



Bhartiya Vidya Bhavan's

Sardar Patel Institute of Technology, Mumbai-400058
Department of Computer Science and Engineering
OEIT1:Blockchain Technology and Applications

MINI PROJECT REPORT

BRANCH: CSE (DS)

BATCH: E

DATE: 30/04/2024

Submitted By

UID:	Name:
2021700001	Anushka Acharya
2021700007	Asmi Bhanushali
2021700009	Aditi Bhutada
2021200114	Divya Suvarna

OUTLINE

PROBLEM STATEMENT	2
OBJECTIVES	2
TECH STACK	2
METHODOLOGY	4
DAPP SERVER	4
GANACHE TEST BLOCKCHAIN	5
EXPERIMENTAL SETUP	5
CONCLUSION	13

PROBLEM STATEMENT

To develop a consumer friendly product verification application using Blockchain Technology for detecting genuine products in supply chain companies using QR code scanning.

OBJECTIVES

1. Create a reliable blockchain database that is connected to QR codes that are linked to every product With the help of this system, consumers will be able to easily authenticate products by simply scanning the QR code.
2. Develop a user-friendly verification application accessible to consumers, enabling real-time authentication of products.

TECH STACK

1. Solidity:

Solidity is a programming language used for writing smart contracts on the Ethereum blockchain. Your project likely includes Solidity smart contracts for implementing the logic of identifying fake products and managing interactions between various parties on the blockchain.

2. Truffle:

Truffle is a popular development framework for Ethereum that simplifies the process of building and deploying smart contracts. Your project uses Truffle for compiling Solidity contracts, managing migrations, and providing a development environment for testing and deployment.

3. Ganache:

Ganache is a local blockchain emulator that provides a simulated Ethereum blockchain environment for development and testing purposes. Your project uses Ganache to set up a local blockchain network where you can deploy and interact with smart contracts without incurring real Ethereum transaction costs.

4. Web3.js:

Web3.js is a JavaScript library that allows interaction with Ethereum nodes. It provides an interface for connecting to Ethereum networks, deploying smart contracts, and interacting with them. Your project uses Web3.js for connecting the front end to the blockchain network, enabling users to interact with smart contracts through the web interface.

5. Metamask:

Metamask is a browser extension that allows users to interact with Ethereum-based decentralized applications directly from their web browsers. Your project integrates with Metamask to enable users to connect their Ethereum accounts, sign transactions, and interact with the smart contracts deployed on the blockchain network.

6. Node.js and npm:

Node.js and npm (Node Package Manager) are used for managing project dependencies and running the development server.

- Frontend Web Application: Project includes a frontend web application built with **HTML, CSS, and JavaScript**. This application provides a user interface for interacting with the smart contracts deployed on the **Ethereum blockchain**.
- Users can perform actions such as registering products, verifying product authenticity, and viewing product information through this interface.

Overall, the project leverages these blockchain technologies to implement a decentralized application for verifying products using blockchain technology, enabling secure and transparent product verification on the Ethereum blockchain.

METHODOLOGY

DAPP SERVER

In your project setup, there's no explicit mention of a DApp (decentralized application) server. However, it does involve serving web content through a development server, which is commonly used in DApp development for hosting the front end of the application. This development server typically serves HTML, CSS, and JavaScript files to users' browsers and facilitates interaction with the blockchain through Web3.js.

Here's a breakdown of the components and their roles:

1. Frontend (HTML, CSS, JavaScript):

- The frontend components, including HTML for structure, CSS for styling, and JavaScript for interactivity, are served to users' browsers by the development server.
- These files contain the user interface elements of your DApp, allowing users to interact with the underlying blockchain functionalities.

2. Development Server:

- The development server is responsible for serving the frontend files to users' browsers.
- It may also handle routing requests and responses between the frontend and backend components of your application, though in a decentralized context, much of the application logic may reside in the frontend.

3. Web3.js:

- Web3.js is used in the frontend JavaScript code to interact with the Ethereum blockchain.
- It allows your DApp to send transactions, read data from smart contracts, and listen for events emitted by the blockchain.

4. Metamask Integration:

- DApp integrates with Metamask, a browser extension that provides users with a secure way to manage their Ethereum accounts and interact with decentralized applications.

- Metamask allows users to sign transactions and securely communicate with the Ethereum blockchain directly from their browsers.

GANACHE TEST BLOCKCHAIN

In this project, Ganache is used as a local blockchain for development and testing purposes. Here's how we implemented it:

1. Setup:

- After installation, we launched Ganache, to start a local Ethereum blockchain environment.

2. Integration with Truffle:

- Truffle, a popular development framework for Ethereum, is used to compile, deploy, and test smart contracts.
- We configured Truffle to connect to the Ganache blockchain environment for deployment and testing.
- This connection is established by specifying the appropriate network configuration in the Truffle project's configuration file (`truffle-config.js` or `truffle.js`).

3. Deployment:

- After writing and compiling smart contracts using Truffle, we deployed them to the Ganache blockchain environment.
- Truffle migrations handle the deployment process, ensuring that smart contracts are deployed to the correct Ganache network.

Overall, Ganache serves as a local Ethereum blockchain environment for development and testing, seamlessly integrated with Truffle to streamline the smart contract development workflow in the project.

EXPERIMENTAL SETUP

CLIENT SERVICE

In our project, the client service typically refers to the frontend or client-side application that interacts with the blockchain network and smart contracts.

1. Technologies:

- HTML, CSS, and JavaScript are used to build the frontend interface.
- Libraries and frameworks such as jQuery may also be utilized for DOM manipulation and asynchronous request handling.
- Web3.js is employed to interact with the Ethereum blockchain and smart contracts from the client-side.

2. Functionality:

- The client service provides a user interface through which users can interact with the blockchain application.
- Users can initiate transactions, and view relevant information presented in a user-friendly format.
- Forms and input fields are included to gather user input required for executing transactions and invoking smart contract functions.

3. Integration with Blockchain:

- Web3.js is used to establish a connection between the client-side application and the Ethereum blockchain network.
- Smart contract ABI (Application Binary Interface) is provided to interact with deployed smart contracts from the client-side.
- Contract methods are accessed through Web3.js to perform transactions and call functions.

4. User Experience:

- The client service aims to provide an intuitive and seamless user experience, guiding users through the process of interacting with the blockchain application.
- User-friendly interfaces, clear instructions, and visual feedback contribute to enhancing the usability of the application.

RESULT SCREENSHOTS

<div>ACCOUNTS BLOCKS TRANSACTIONS CONTRACTS EVENTS LOGS</div> <div>CURRENT BLOCK10GAS PRICE2000000000GAS LIMIT6721975HARDFORKMERGENETWORK ID5777RPC SERVERHTTP://127.0.0.1:7545MINING STATUSAUTOMININGWORKSPACEBCT PROJECTSWITCH</div> <div>SEARCH FOR BLOCK NUMBERS OR TX HASHES</div>									
MNEMONIClonely album make slim cloud interest feed pole index science better logic					HD PATHm44'60'0'0account_index				
ADDRESS	BALANCE				TX COUNT	INDEX			
0x81C991c60E9b039e023342c959712B8d6B00761B	99.99 ETH				10	0			
ADDRESS	BALANCE				TX COUNT	INDEX			
0x3514E435d63917C140d85cf3F8Cd228D670156f7	100.00 ETH				0	1			
ADDRESS	BALANCE				TX COUNT	INDEX			
0x35d71AA70BD4bCc9bF7942b68194A642bcd2ef8E	100.00 ETH				0	2			
ADDRESS	BALANCE				TX COUNT	INDEX			
0x0d6D61D4a28cad2079220B4FBe57A5b3fAB4E4Ef	100.00 ETH				0	3			
ADDRESS	BALANCE				TX COUNT	INDEX			
0xD73ACDcf2761B8f3559941BFb1CCfBd8EaE33378	100.00 ETH				0	4			
ADDRESS	BALANCE				TX COUNT	INDEX			
0xf5471cF5F1ECBFb09c2118415FC47C4090E4EdF0	100.00 ETH				0	5			

GANACHE

BCT PROJECT - Google Docs

MetaMask

chrome-extension://nkbihfbeogaeaoehlefnkodbefgpgknn/home.html

METAMASK

Account 2

0x81C99...0761B

99.9899 ETH

Buy & Sell

Send

Swap

Bridge

Portfolio

Tokens

NFTs

Activity

May 6, 2024

Contract interaction

Confirmed

-0 ETH

Contract interaction

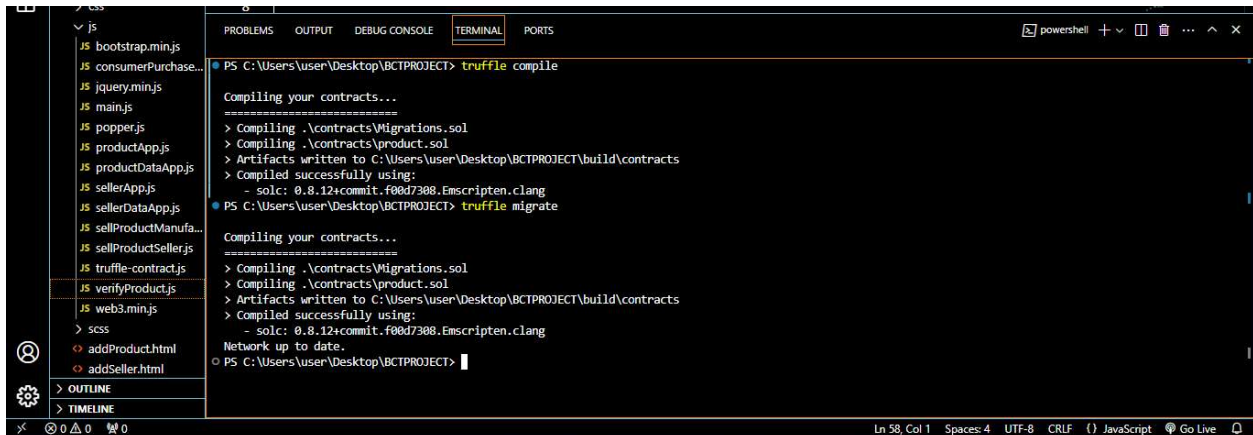
Confirmed

-0 ETH

Contract interaction

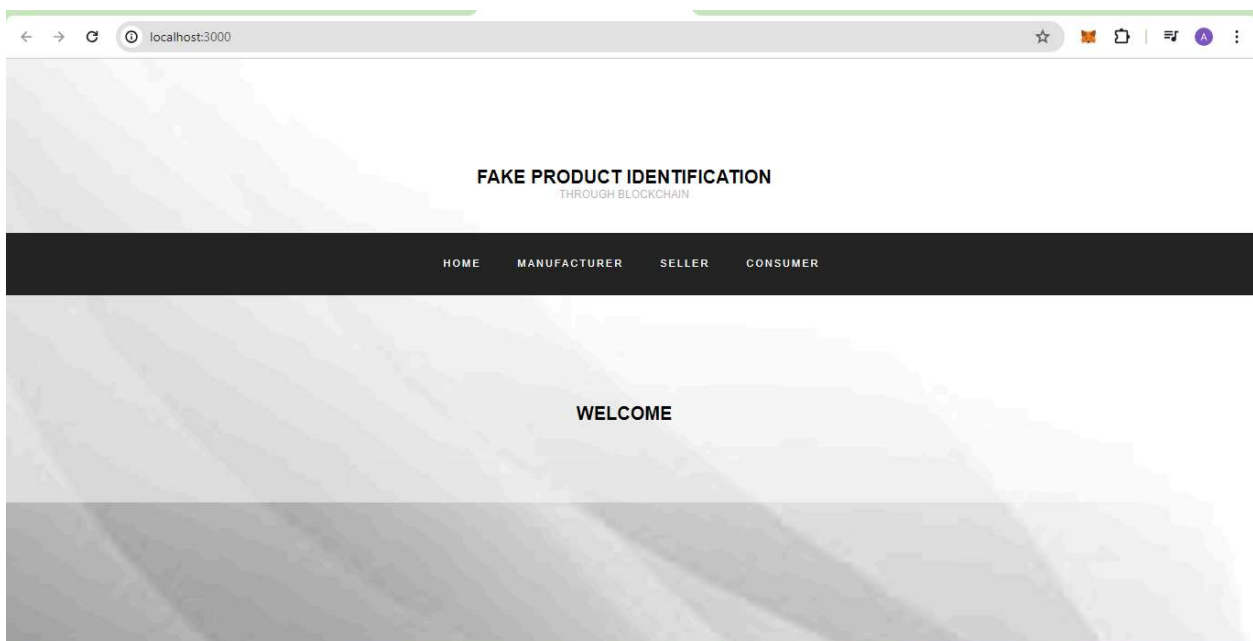
-0 ETH

METAMASK

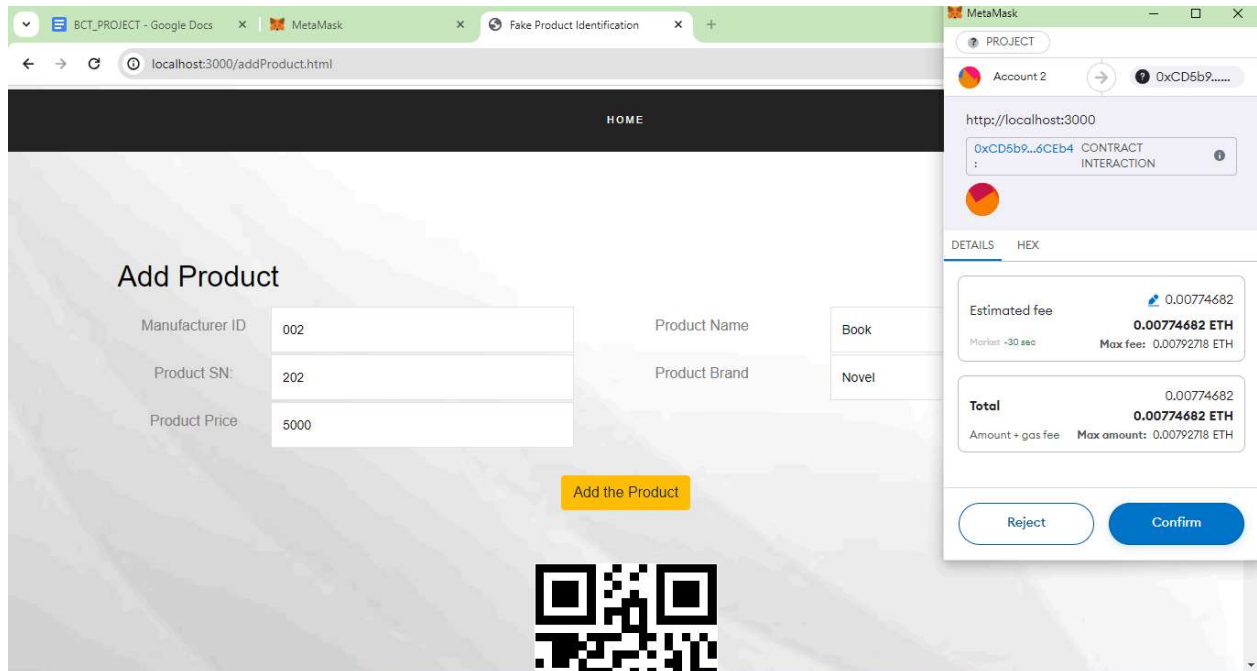


The screenshot shows a VS Code editor with a file explorer on the left containing files like bootstrap.min.js, consumerPurchase..., jquery.min.js, main.js, popper.js, productApp.js, productDataApp.js, sellerApp.js, sellerDataApp.js, sellProductManufa..., sellProductSeller.js, truffle-contract.js, verifyProduct.js, web3.min.js, sccs, addProduct.html, and addSeller.html. The terminal window is open, showing the execution of Truffle commands. The first command is `truffle compile`, which outputs the compilation of `\contracts\Migrations.sol` and `\contracts\product.sol`, stating that artifacts were written to `C:\Users\user\Desktop\BCTPROJECT\build\contracts` and compiled successfully using `solc: 0.8.12+commit.f08d7308.Emscripten.clang`. The second command is `truffle migrate`, which outputs the compilation of the same contracts, states the network is up to date, and shows the prompt `PS C:\Users\user\Desktop\BCTPROJECT>`.

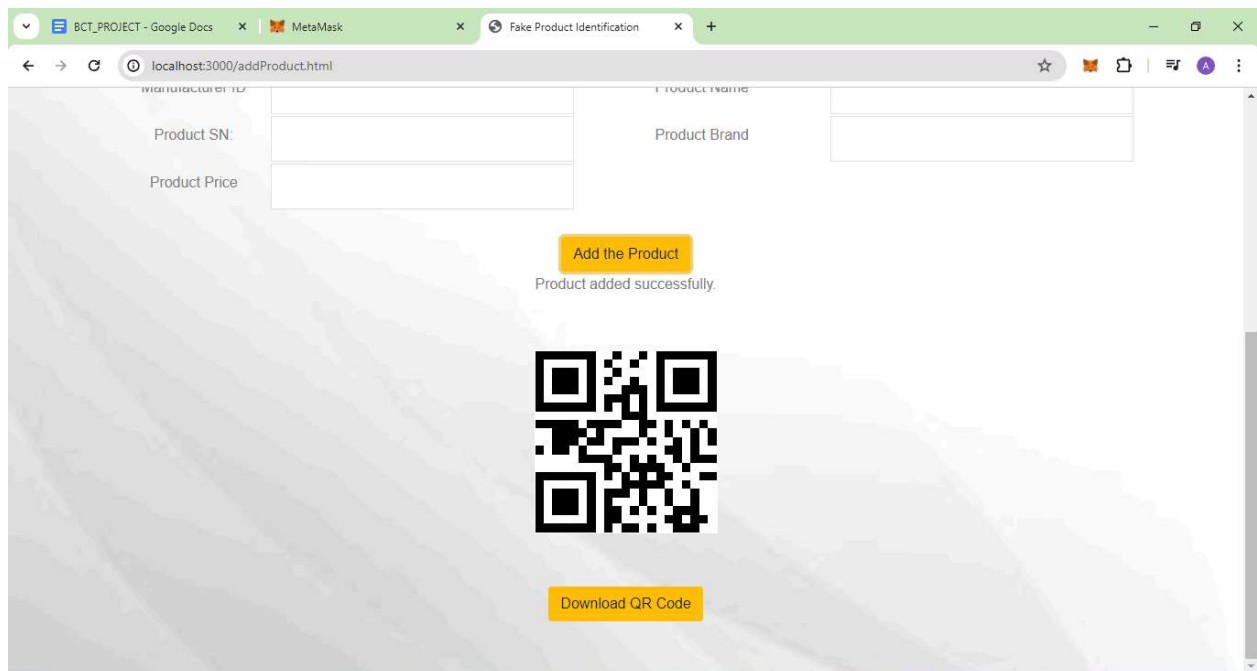
COMPILING CONTRACTS AND MIGRATING



WEBSITE (HOMEPAGE) DEPLOYED ON LOCALHOST



ADDING PRODUCT



PRODUCT ADDED SUCCESSFULLY AND QR CODE GENERATED

CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE	SWITCH	SETTINGS
BLOCK 11	MINED ON 2024-05-06 14:11:22					GAS USED 205818	BCT PROJECT		
BLOCK 10	MINED ON 2024-05-06 12:29:44					GAS USED 51653			
BLOCK 9	MINED ON 2024-05-06 12:29:09					GAS USED 211508			
BLOCK 8	MINED ON 2024-05-06 12:28:20					GAS USED 185930			
BLOCK 7	MINED ON 2024-05-06 12:10:31					GAS USED 88653			
BLOCK 6	MINED ON 2024-05-06 12:09:50					GAS USED 225808			
BLOCK 5	MINED ON 2024-05-06 12:08:59					GAS USED 183130			
BLOCK 4	MINED ON 2024-05-06 11:53:06					GAS USED 28813			
BLOCK 3	MINED ON 2024-05-06 11:53:06					GAS USED 2002117			

PRODUCT DATA ADDED TO BLOCKCHAIN : SUCCESSFUL TRANSACTION

FAKE PRODUCT IDENTIFICATION
THROUGH BLOCKCHAIN

HOME

Add Seller

Seller Name	Divya	Seller Brand	ABCD
Seller Code	202	Seller Phone Number	01234567
Seller Manager	Aditi	Seller Address	bbbb
Manufacturer ID	002		

Add the Seller

PROJECT

Account 2 → 0xCD5b9.....

http://localhost:3000

0xCD5b9...6CEb4 CONTRACT INTERACTION

DETAILS HEX

Estimated fee 0.00763583
Market -30 sec **0.00763583 ETH**
Max fee: 0.00779399 ETH

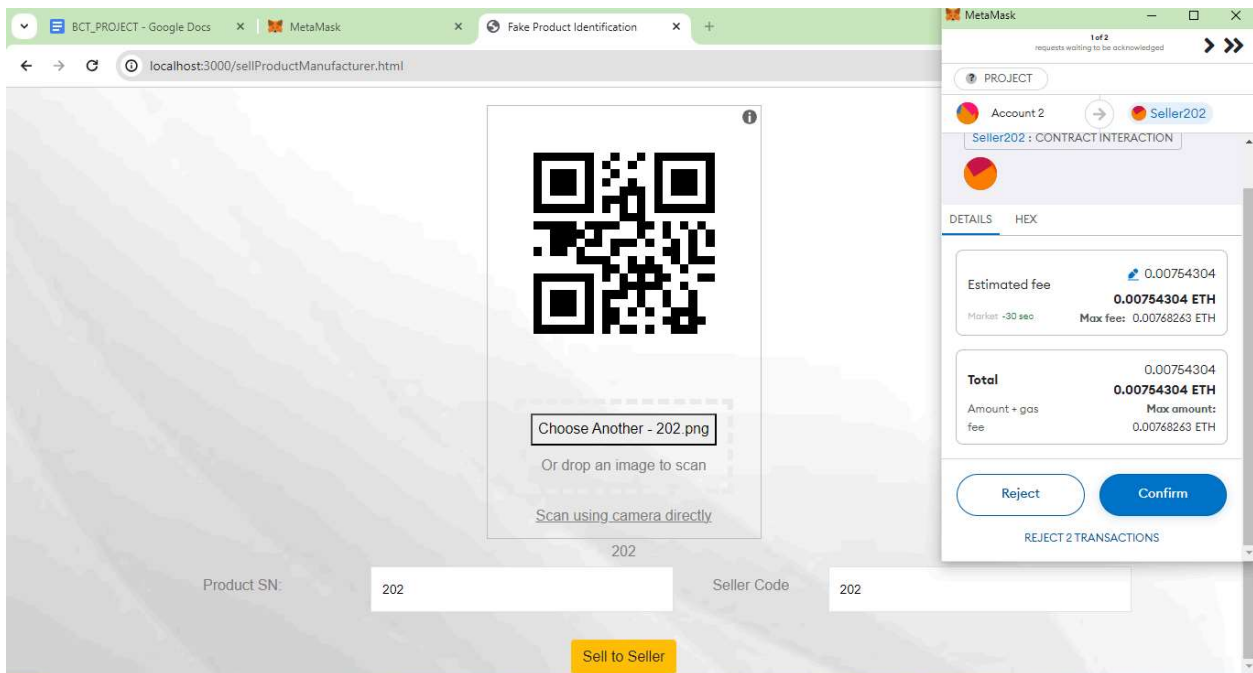
Total 0.00763583
0.00763583 ETH
Amount + gas fee Max amount: 0.00779399 ETH

Reject Confirm

SELLER ADDED SUCCESSFULLY

BLOCK	MINED ON	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	GAS USED	TRANSACTIONS
BLOCK 12	2024-05-06 14:13:35	20000000000	6721975	MERGE	5777	HTTP://127.0.0.1:7545	AUTOMINING	228596	1 TRANSACTION
BLOCK 11	2024-05-06 14:11:22							205818	1 TRANSACTION
BLOCK 10	2024-05-06 12:29:44							51653	1 TRANSACTION
BLOCK 9	2024-05-06 12:29:09							211508	1 TRANSACTION
BLOCK 8	2024-05-06 12:28:20							185930	1 TRANSACTION
BLOCK 7	2024-05-06 12:10:31							88653	1 TRANSACTION
BLOCK 6	2024-05-06 12:09:50							225808	1 TRANSACTION
BLOCK 5	2024-05-06 12:08:59							183130	1 TRANSACTION
BLOCK 4	2024-05-06 11:53:06							28813	1 TRANSACTION

SUCCESSFUL TRANSACTION OF ADDING SELLER



UPLOADING QR TO SELL PRODUCT TO SELLER : PRODUCT SOLD SUCCESSFULLY

The screenshot shows the Ganache application interface. At the top, there are tabs for ACCOUNTS, BLOCKS, TRANSACTIONS, CONTRACTS, EVENTS, and LOGS. Below these tabs, a summary bar displays various network statistics. The main area shows a list of blocks, each with a 'BLOCK' number, 'MINED ON' timestamp, and 'GAS USED' amount. Each block entry has a corresponding '1 TRANSACTION' button.

CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
14	20000000000	6721975	MERGE	5777	HTTP://127.0.0.1:7545	AUTOMINING	BCT PROJECT

BLOCK	MINED ON	GAS USED	TRANSACTION
14	2024-05-06 14:15:55	51653	1 TRANSACTION
13	2024-05-06 14:15:39	88653	1 TRANSACTION
12	2024-05-06 14:13:35	228596	1 TRANSACTION
11	2024-05-06 14:11:22	265818	1 TRANSACTION
10	2024-05-06 12:29:44	51653	1 TRANSACTION
9	2024-05-06 12:29:09	211568	1 TRANSACTION
8	2024-05-06 12:28:20	185930	1 TRANSACTION
7	2024-05-06 12:10:31	88653	1 TRANSACTION
6	2024-05-06 12:09:50	225808	1 TRANSACTION

SUCCESSFUL TRANSACTION FOR SELL PRODUCT TO SELLER

The screenshot shows a web application running on localhost:3000. It features a QR code for product identification. Below the QR code, there is a text input field for 'Product SN' with the value '202' and another for 'Consumer Code' with the value '302'. A 'Sell to Consumer' button is visible, and a message at the bottom states 'Product sold successfully to consumer'.

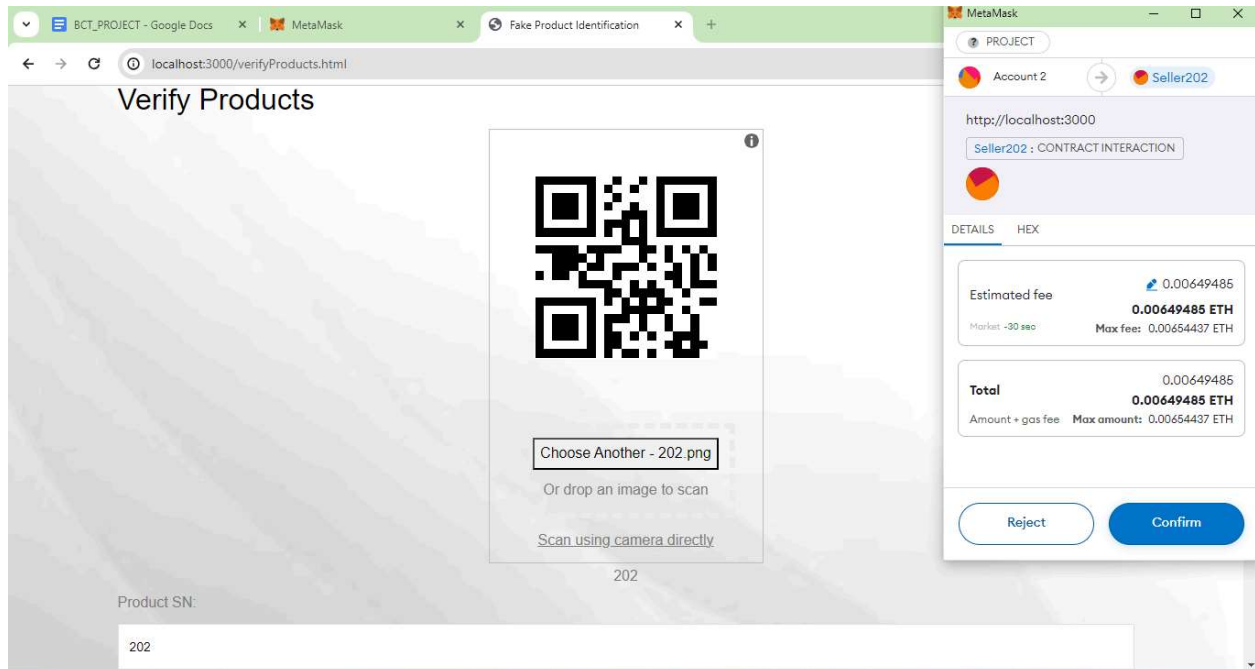
Choose Another - 202.png
Or drop an image to scan
[Scan using camera directly](#)

Product SN: 202 Consumer Code 302

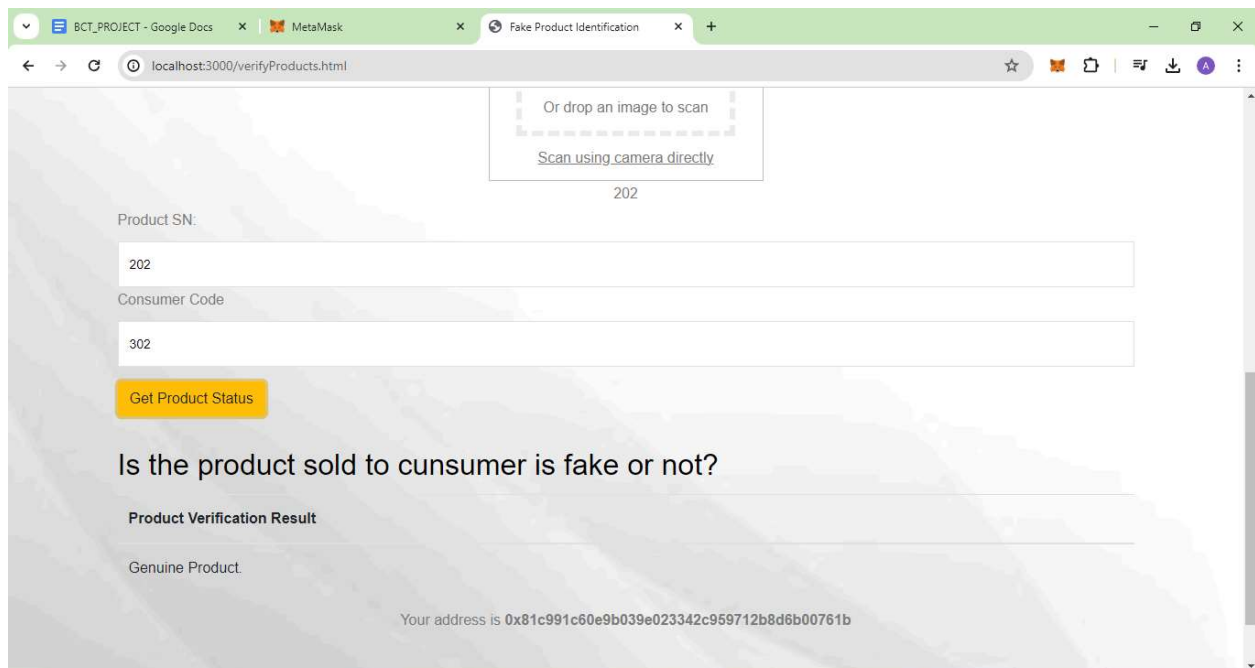
[Sell to Consumer](#)

Product sold successfully to consumer

UPLOADING QR TO SELL PRODUCT TO CONSUMER : PRODUCT SOLD SUCCESSFULLY



UPLOADING QR FOR VERIFICATION



GENUINE PRODUCT DETECTED WITH ACCOUNT ADDRESS

ACCOUNTS	BLOCKS	TRANSACTIONS	CONTRACTS	EVENTS	LOGS
CURRENT BLOCK 21	GAS PRICE 2000000000	GAS LIMIT 6721975	HARDFORK MERGE	NETWORK ID 5777	RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING					WORKSPACE BCT PROJECT
SWITCH					⚙️
BLOCK 21	MINED ON 2024-05-06 14:22:22	GAS USED 24433	1 TRANSACTION		
BLOCK 20	MINED ON 2024-05-06 14:21:36	GAS USED 33320	1 TRANSACTION		
BLOCK 19	MINED ON 2024-05-06 14:21:02	GAS USED 62033	1 TRANSACTION		
BLOCK 18	MINED ON 2024-05-06 14:19:31	GAS USED 103642	1 TRANSACTION		
BLOCK 17	MINED ON 2024-05-06 14:18:49	GAS USED 76672	1 TRANSACTION		
BLOCK 16	MINED ON 2024-05-06 14:17:23	GAS USED 76672	1 TRANSACTION		
BLOCK 15	MINED ON 2024-05-06 14:16:59	GAS USED 60365	1 TRANSACTION		
BLOCK 14	MINED ON 2024-05-06 14:15:55	GAS USED 51653	1 TRANSACTION		
BLOCK 13	MINED ON 2024-05-06 14:15:39	GAS USED 88653	1 TRANSACTION		

SUCCESSFUL TRANSACTION FOR VERIFICATION

CONCLUSION

By leveraging the decentralized nature of blockchain, the project ensures that product information, including serial numbers and transaction history, is securely recorded on the blockchain ledger. This provides consumers, manufacturers, and sellers with a reliable and transparent means of verifying the authenticity of products at every stage of the supply chain.