VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Project on

Supply chain management using blockchain technology

for the course

Blockchain Architecture Design And Use Cases

(BKT4005)

By:

Anirudh Kaushik(20BKT0038)

C M Vivek(20BKT0134)

Chhavi Jain (20BKT0144)

Submitted to:

Prof. Lokesh Kumar R

TABLE OF CONTENTS

ABSTRACT

The current supply chain contains a large number of counterfeit goods. It is essential to have a system in place that allows customers to verify all the information about the product they are purchasing in order to determine whether or not it is authentic.

Blockchain technology is used to identify genuine goods and segregate counterfeit goods. Blockchain technology is a distributed, decentralized, and digital ledger that keeps track of transactions in a large number of databases and node computers linked via chains.

Blockchain technology is secure because no block can be altered or compromised because the data is immutable once it has been added to the chain. Customers or consumers do not have to rely on the trust of third parties to certify the authenticity and safety of a product thanks to Blockchain technology.

LIST OF ABBREVIATIONS

SCM: supply chain management.
Blockchain: A blockchain is a distributed ledger with growing lists of records (blocks) that are securely linked together via cryptographic hashes.
Distributed Ledger: A distributed ledger (also called a shared ledger or distributed ledger technology or DLT) is the consensus of replicated, shared, and synchronized digital data that is geographically spread (distributed) across many sites, countries, or institutions
Block: A block is a place in a blockchain where information is stored and encrypted, identified by long numbers that include encrypted transaction information from previous blocks and new transaction information.
Nonce: A nonce is a random or semi-random number that is generated for a specific use.
Cryptocurrency: Cryptocurrency is a form of digital or virtual currency designed to work as a medium of exchange through a computer network that is not reliant on any central authority to uphold or maintain it. Cryptocurrencies don't have a central issuing or regulating authority, instead using a decentralized system to record transactions and issue new units.

1. INTRODUCTION

1.1 SYSTEM OVERVIEW

Supply chain management is a critical aspect of modern business operations, involving the coordination of various parties involved, including manufacturers, suppliers, distributors, and retailers, to ensure the timely delivery of goods and services to customers. However, supply chain management is often plagued by inefficiencies, such as opaque processes, lack of transparency, and mistrust between parties, leading to

increased costs, miscommunication, and delays. Blockchain technology, with its ability to provide secure and transparent transaction tracking, has the potential to address many of these challenges which currently plague the supply chain management system.

1.2 OBJECTIVE

The objective of this project is to explore the use of blockchain technology in supply chain management and its potential benefits, challenges, and applications. The project hopes to provide an overview of using blockchain technology in supply chain management and its respective implications in the field of supply chain management, including the potential to increase efficiency, reduce costs, and improve collaboration among stakeholders.

1.3 APPLICATIONS

Blockchain technology has numerous potential applications in supply chain management, including product provenance tracking, inventory management, supply chain finance, and more. By providing a secure and transparent way to track and verify transactions, blockchain can enhance supply chain efficiency, reduce costs, and increase trust and collaboration among stakeholders. Blockchains are decentralized networks where the miners have incentives to keep the network secure. This allows for models in which partition tolerance

1.4 LIMITATIONS

Despite the potential offered by blockchain technology, there are some limitations and challenges when dealing with the space of supply chain management. These include issues related to scalability, interoperability, and data privacy, as well as the need for collaboration and standardization among stakeholders.

2 SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

The existing supply chain management system is often characterized by inefficiencies, such as opaque processes, lack of transparency, and mistrust between parties. These inefficiencies can lead to increased costs and delays, as well as increased risk of fraud and errors.

2.2 PROPOSED SYSTEM

Our proposed system involves incorporating blockchain technology into the supply chain management process. By using a blockchain-based system, supply chain stakeholders can track and verify transactions in a secure and transparent manner, reducing the risk of fraud and errors, as well as improving efficiency and collaboration among stakeholders.

2.2.1 Benefits of Proposed System

The proposed system offers several benefits over the existing system, including but not limited to:

Increased transparency: By using a blockchain-based system, supply chain stakeholders can track and verify transactions in real-time, providing increased transparency and visibility into the supply chain.

Improved efficiency: Blockchain technology can automate many supply chain processes, reducing the need for manual intervention and increasing efficiency.

Reduced costs: The increased efficiency and transparency provided by the blockchain-based system can reduce costs associated with supply chain management, such as those related to inventory management and logistics.

Enhanced security: Blockchain technology provides a secure and tamper-proof system for tracking and verifying transactions, reducing the risk of fraud and errors.

Improved collaboration: The transparent and secure nature of blockchain technology can improve trust and collaboration among supply chain manufacturers, distributors, retailers and customers leading to a more effective supply chain management process centred on the products.

3 REQUIREMENT SPECIFICATION

3.1 Hardware Requirements:

Processor: Intel Core i5 or equivalent

RAM: 8 GB or higher

Storage: 1 GB or higher

Network: 1 Mbps or higher

3.2 Software Requirements:

Operating System: Windows 10 or Linux Ubuntu 18.04 or higher

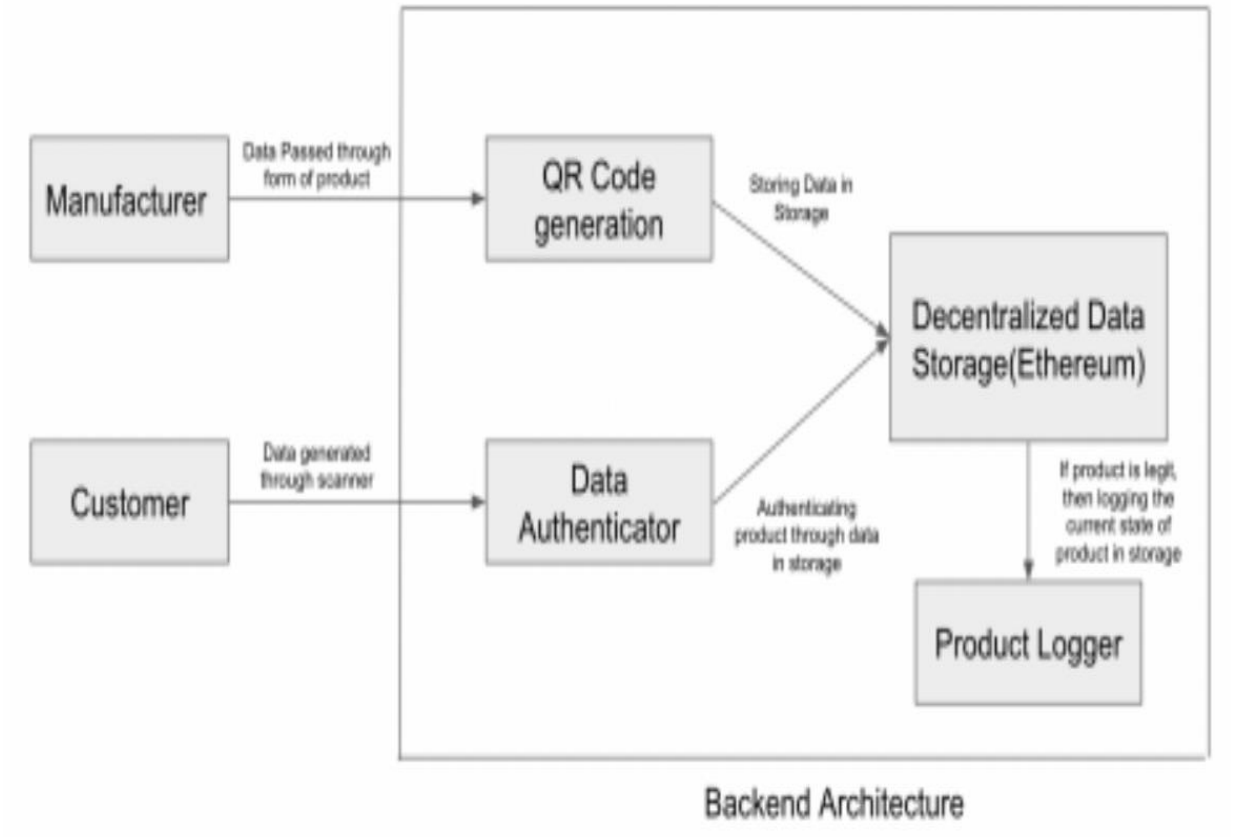
Blockchain Platform: Polygon Mumbai test net(alchemy)

Programming Languages: Solidity, NodeJS, HTML, CSS,React

Development Tools: Truffle & Ganache, Remix, or similar

4 SYSTEM DESIGN SPECIFICATION

4.1 SYSTEM ARCHITECTURE



Backend Architecture

5 SYSTEM IMPLEMENTATION

The smart contract is deployed onto alchemy's polygon Mumbai test net. The website is able to access the smart contract (and in the process the blockchain) via an API key generated for the app. The frontend gives users a more interactable user interface to the smart contract. Whenever attempting to add data to the blockchain, cryptocurrency needs to be paid for the transaction to be mined by nodes on the blockchain. The interaction between the website and the blockchain occurs by calling the functions of the deployed smart contract.

5.1 MODULE DESCRIPTION

The smart contract:

This smart contract allows users to track the production and distribution of a product from the manufacturer to the retailer. Several structs are defined including prodMan, prodDis, prodRet, and prodData, which contain information about the product, the manufacturer, the distributor, and the retailer. The contract also defines several events, such as prodAdd, pmanAdd, pdisAdd, pretAdd, pdistrans, pmandis, pdisret, and psold, which are triggered when certain actions occur.

The contract provides functions to add a product, add a product distributor, add a product retailer, add a product manufacturer, add a product distribution, add a product retail, and sell a product. The addProd function allows a manufacturer to add a new product, while addProdDis and addProdRet functions allow distributors and retailers to add their information. The addMan, addDis, and addRet functions allow users to add new manufacturers, distributors, and retailers, respectively after validation of the data provided as input.

The contract also provides functions to track the movement of a product from the manufacturer to the retailer. The addProdDis function allows a distributor to receive the product, while the addProdRet function allows a retailer to receive the product. The SellProd function allows a retailer to sell the product, and the contract updates the status of the product to "SOLD". The contract ensures that only valid products can be sold by using the verify function. The viewProdRoute function provides the overall movement of a product throughout the supply chain.

The front end:

addDis.jsx: react component that enables the functionality to add a distributor.
addMan.jsx: react component that enables the functionality to add a manufacturer.
addRet.jsx: react component that enables the functionality to add a retailer.
Navbar.jsx: react component that deals with the nav bar.

6 CONCLUSION AND FUTURE ENHANCEMENTS

This project demonstrates tracking of a product through the supply chain network(from manufacturers to retailers and finally the customer).

7 APPENDICES

7.1 APPENDIX 1 - SAMPLE SOURCE CODE

The smart contract:

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.0 <0.9.0;
contract PMD{
    enum state{NOTEXIST,CREATED,ONTHEWAY,ATSTORE,SOLD}
    state constant defaultChoice=state.NOTEXIST;
    //dis->distributor, Man->manefacturer, Ret->retailer, prod->Product
    struct prodMan{
        string loc;
        string org;
        string name;
        uint produce;
    }
    struct prodDis{
        string loc;
        string name;
        string org;
        uint del;
    }
    struct prodRet{
        string loc;
        string org;
        string name;
        uint sold;
    }
    struct prodData{
        string name;
        uint manId;
        uint[] disId;
        uint retId;
        state curstate;
        string des;
    }
    uint counterDis=0;
    uint counterRet=0;
```

```
    uint counterMan=0;
    uint counterProd=0;
    mapping(uint256=>prodMan) prodManInfo;
    mapping(uint256=>prodDis) prodDisInfo;
    mapping(uint256=>prodRet) prodRetInfo;
    mapping(uint256=>prodData) prodInfo;
    event prodAdd(uint,uint,string,uint);//man and prod
    event pmanAdd(uint,string);//new man added
    event pdisAdd(uint,string);//new dis added
    event pretAdd(uint,string);//new ret added
    event pdistrans(uint,uint);// dis transferred
    event pmandis(uint,uint,uint);//man -> dis,prod
    event pdisret(uint,uint,uint);//dis -> ret,prod
    event psold(uint,uint);//ret,prod
    function addProd(uint256 _manID,string memory _name,string memory _des)
public{
        require(_manID<counterMan);
        prodData memory nprod;
        nprod.manId=_manID;
        nprod.curstate=state.CREATED;
        nprod.name=_name;
        nprod.des=_des;
        prodInfo[counterProd]=nprod;
        emit prodAdd(_manID,counterProd,_name,counterProd);
        counterProd++;
    }
    function addProdDis(uint256 _id,uint256 _disId ) public{
        require(_id<counterProd,"counterProd");
        require(prodInfo[_id].curstate==state.ONTHEWAY||
        prodInfo[_id].curstate==state.CREATED,"product is not available for
shipment");
        if(prodInfo[_id].curstate==state.CREATED){
            prodInfo[_id].curstate=state.ONTHEWAY;
            prodInfo[_id].disId.push(_disId);
            prodManInfo[prodInfo[_id].manId].produce++;
            emit pmandis(prodInfo[_id].manId,_disId,_id);
        }
        else{
            prodInfo[_id].disId.push(_disId);
            uint x=prodInfo[_id].disId.length-2;
            prodDisInfo[prodInfo[_id].disId[x]].del++;
            emit pdistrans(prodInfo[_id].disId[x],_disId);
        }
    }
    function addProdRet(uint256 _id,uint256 _retId) public{
```

```
        require(_id<counterProd,"counterProd");
        require(_retId<counterRet,"counterRet");
        require(prodInfo[_id].curstate==state.ONTHEWAY,"product is not available
for dispatch");
        prodInfo[_id].retId=_retId;
        prodInfo[_id].curstate=state.ATSTORE;
        uint x=prodInfo[_id].disId.length-1;
        prodDisInfo[prodInfo[_id].disId[x]].del++;
        emit pdisret(prodInfo[_id].disId[prodInfo[_id].disId.length-
1],_retId,_id);
    }
    function SellProd(uint256 _id)public {
        require(_id<counterProd,"counterProd");
        require(prodInfo[_id].curstate==state.ATSTORE,"product is not available
at store");
        require(verify(_id,prodInfo[_id].retId),"invalid product!");
        prodInfo[_id].curstate=state.SOLD;
        prodRetInfo[prodInfo[_id].retId].sold++;
        emit psold(prodInfo[_id].retId,_id);
    }
    function addMan(string memory _loc,string memory _org,string memory
_name)public  returns (prodMan memory){
        prodMan memory nMan;
        nMan.loc=_loc;
        nMan.org=_org;
        nMan.name=_name;
        nMan.produce=0;
        prodManInfo[counterMan]=nMan;
        emit pmanAdd(counterMan,_org);
        counterMan++;
        return nMan;
    }
    function addDis(string memory _loc,string memory _org,string memory
_name)public  returns(prodDis memory){
        prodDis memory nDis;
        nDis.loc=_loc;
        nDis.org=_org;
        nDis.name=_name;
        nDis.del=0;
        prodDisInfo[counterDis]=nDis;
        emit pdisAdd(counterDis,_org);
        counterDis++;
        return nDis;
    }
```

```
    function addRet(string memory _loc,string memory _org,string memory
_name)public  returns(prodRet memory){
        prodRet memory nRet;
        nRet.loc=_loc;
        nRet.org=_org;
        nRet.name=_name;
        nRet.sold=0;
        prodRetInfo[counterRet]=nRet;
        emit pretAdd(counterRet,_org);
        counterRet++;
        return nRet;
    }
    function verify(uint _id,uint _retId) public view returns (bool){
        return (prodInfo[_id].curstate!=state.NOTEXIST)&&

(prodInfo[_id].retId==_retId)&&(prodInfo[_id].curstate==state.ATSTORE);
    }
    function viewRet(uint _id) public view returns(prodRet memory){
        return prodRetInfo[_id];
    }
    function viewDis(uint _id) public view returns(prodDis memory){
        return prodDisInfo[_id];
    }
    function viewMan(uint _id) public view returns(prodMan memory){
        return prodManInfo[_id];
    }
    function viewinfo(uint _id) public view returns(prodData memory){
        return prodInfo[_id];
    }
    function viewstate(uint _id) public view returns(state){
        return prodInfo[_id].curstate;
    }
    //demo only
    function viewProdRoute(uint _id) public view returns(string memory){
        prodData memory x=prodInfo[_id];
        string memory res;
        if(x.curstate==state.NOTEXIST){
            res="item does not exist!";
        }
        if(x.curstate>=state.CREATED){
            res=string.concat("item created by:     ",prodManInfo[x.manId].name);
        }
        if(x.curstate>=state.ONTHEWAY){
            res=string.concat(res," distributors:    ");
            for(uint i=0;i<x.disId.length;i++){
```

```
                res=string.concat(res,prodDisInfo[x.disId[i]].name);
                res=string.concat(res," ");
            }
        }
        if(x.curstate>=state.ATSTORE){
            res=string.concat(res," retailer:   ");
            res=string.concat(res,prodRetInfo[x.retId].name);
        }
        if(x.curstate==state.SOLD){
            res=string.concat(res," item sold");
        }
        return res;
    }
}
```

AddDis.jsx:

```
import { Table, TextField } from "@mui/material";
import React from "react";
import { useState } from "react";
import {
  Backdrop,
  CircularProgress,
  Typography,
  Button,
  Box,
} from "@mui/material";
import ConfirmDis from "./ConfirmDis";

function AddDis() {
  const [loc, setloc] = useState();
  const [org, setorg] = useState();
  const [name, setname] = useState();
  const [isClicked, setisClicked] = useState(false);

//   function handleDis(e) {
//     let Dis = e.target.value.split(" ");
//     console.log(Dis);
//     setloc(Dis[0]);
//     setorg(Dis[1]);
//     setname(Dis[2]);
//   }
  if (isClicked) {
```

```
  return <ConfirmDis loc={loc} org={org} name={name}></ConfirmDis>;
}

return (
  <Box display="flex" justifyContent="center" mt="15%">
    <form>
      <Table>
      <tr>
          <td>
            <TextField
              label="Name"
              onChange={(e) => {
                  setname(e.target.value);
                }}              ></TextField>
          </td>
        </tr>
        <tr>
          <td>
            <TextField
              label="location"
              placeholder="Enter location"
              onChange={(e) => {
                setloc(e.target.value);
              }}
            ></TextField>
          </td>
        </tr>
        <tr>
          <td>
            <TextField
              label="Organization"
              placeholder="Enter organization"
              onChange={(e) => {
                  setorg(e.target.value);
                }}
            ></TextField>
          </td>
        </tr>
      </Table>
      <Typography display="flex" justifyContent="center">
        <Button
          onClick={() => {
            setisClicked(true);
          }}
        >
```

```
            Create
          </Button>
        </Typography>
      </form>
    </Box>
  );
}

export default AddDis;
```

AddMan.jsx:

```
import { Table, TextField } from "@mui/material";
import React from "react";
import { useState } from "react";
import {
  Backdrop,
  CircularProgress,
  Typography,
  Button,
  Box,
} from "@mui/material";
import ConfirmMan from "./ConfirmMan";

function AddMan() {
  const [loc, setloc] = useState();
  const [org, setorg] = useState();
  const [name, setname] = useState();
  const [isClicked, setisClicked] = useState(false);

//   function handleMan(e) {
//     let Man = e.target.value.split(" ");
//     console.log(Man);
//     setloc(Man[0]);
//     setorg(man[1]);
//     setname(man[2]);
//   }
  if (isClicked) {
    return <ConfirmMan loc={loc} org={org} name={name}></ConfirmMan>;
  }

  return (
```

```
<Box display="flex" justifyContent="center" mt="15%">
  <form>
    <Table>
    <tr>
        <td>
          <TextField
            label="Name"
            onChange={(e) => {
                setname(e.target.value);
              }}              ></TextField>
        </td>
      </tr>
      <tr>
        <td>
          <TextField
            label="location"
            placeholder="Enter location"
            onChange={(e) => {
              setloc(e.target.value);
            }}
          ></TextField>
        </td>
      </tr>
      <tr>
        <td>
          <TextField
            label="Organization"
            placeholder="Enter organization"
            onChange={(e) => {
                setorg(e.target.value);
              }}
          ></TextField>
        </td>
      </tr>
    </Table>
    <Typography display="flex" justifyContent="center">
      <Button
        onClick={() => {
          setisClicked(true);
        }}
      >
        Create
      </Button>
    </Typography>
  </form>
```

```
    </Box>
  );
}

export default AddMan;
```

AddRet.jsx:

```
import { Table, TextField } from "@mui/material";
import React from "react";
import { useState } from "react";
import {
  Backdrop,
  CircularProgress,
  Typography,
  Button,
  Box,
} from "@mui/material";
import ConfirmRet from "./ConfirmRet";

function AddRet() {
  const [loc, setloc] = useState();
  const [org, setorg] = useState();
  const [name, setname] = useState();
  const [isClicked, setisClicked] = useState(false);

//   function handleRet(e) {
//     let Ret = e.target.value.split(" ");
//     console.log(Ret);
//     setloc(Ret[0]);
//     setorg(Ret[1]);
//     setname(Ret[2]);
//   }
  if (isClicked) {
    return <ConfirmRet loc={loc} org={org} name={name}></ConfirmRet>;
  }

  return (
    <Box display="flex" justifyContent="center" mt="15%">
      <form>
        <Table>
        <tr>
```

```
            <td>
              <TextField
                label="Name"
                onChange={(e) => {
                    setname(e.target.value);
                  }}              ></TextField>
            </td>
          </tr>
          <tr>
            <td>
              <TextField
                label="location"
                placeholder="Enter location"
                onChange={(e) => {
                  setloc(e.target.value);
                }}
              ></TextField>
            </td>
          </tr>
          <tr>
            <td>
              <TextField
                label="Organization"
                placeholder="Enter organization"
                onChange={(e) => {
                    setorg(e.target.value);
                  }}
              ></TextField>
            </td>
          </tr>
        </Table>
        <Typography display="flex" justifyContent="center">
          <Button
            onClick={() => {
              setisClicked(true);
            }}
          >
            Create
          </Button>
        </Typography>
      </form>
    </Box>
  );
}
```

```
export default AddRet;
```

ConfirmDis.jsx:

```
import React from "react";
import {
  Backdrop,
  CircularProgress,
  Typography,
  Button,
  Box,
} from "@mui/material";
import { Bote, abi_Bote } from "../new_deployments";
import { usePrepareContractWrite, useContractWrite } from "wagmi";

function ConfirmDis({ loc, org, name }) {
  console.log(loc);
  console.log(org);
  console.log(name);
  const { config } = usePrepareContractWrite({
    address: Bote,
    abi: abi_Bote,
    functionName: "addDis",
    args: [loc, org, name],
  });
  const { writeAsync, isLoading, isSuccess } = useContractWrite(config);

  if (isSuccess) {
    return (
      <Typography variant="h5" justifyContent="center" display="flex" mt="15%" >
        Distributor successfully added
      </Typography>
    );
  }
  if (isLoading) {
    return (
      <Backdrop
        open={true}
        sx={{ color: "#3d6f79", zIndex: (theme) => theme.zIndex.drawer + 1 }}
      >
        <CircularProgress color="inherit"></CircularProgress>
      </Backdrop>
```

```
    );
  }
  return (
    <>
      <Typography display="flex" justifyContent="center" mt="20%">
        <Button onClick={writeAsync}>Confirm</Button>
      </Typography>
    </>
  );
}

export default ConfirmDis;
```

ConfirmMan.jsx:

```
import React from "react";
import {
  Backdrop,
  CircularProgress,
  Typography,
  Button,
  Box,
} from "@mui/material";
import { Bote, abi_Bote } from "../new_deployments";
import { usePrepareContractWrite, useContractWrite } from "wagmi";

function ConfirmMan({ loc, org, name }) {
  console.log(loc);
  console.log(org);
  console.log(name);
  const { config } = usePrepareContractWrite({
    address: Bote,
    abi: abi_Bote,
    functionName: "addMan",
    args: [loc, org, name],
  });
  const { writeAsync, isLoading, isSuccess } = useContractWrite(config);

  if (isSuccess) {
    return (
```

```
      <Typography variant="h5" justifyContent="center" display="flex" mt="15%" >
        Manefacturer successfully added
      </Typography>
    );
  }
  if (isLoading) {
    return (
      <Backdrop
        open={true}
        sx={{ color: "#3d6f79", zIndex: (theme) => theme.zIndex.drawer + 1 }}
      >
        <CircularProgress color="inherit"></CircularProgress>
      </Backdrop>
    );
  }
  return (
    <>
      <Typography display="flex" justifyContent="center" mt="20%">
        <Button onClick={writeAsync}>Confirm</Button>
      </Typography>
    </>
  );
}

export default ConfirmMan;
```

ConfirmRet.jsx:

```
import React from "react";
import {
  Backdrop,
  CircularProgress,
  Typography,
  Button,
  Box,
} from "@mui/material";
import { Bote, abi_Bote } from "../new_deployments";
import { usePrepareContractWrite, useContractWrite } from "wagmi";

function ConfirmRet({ loc, org, name }) {
  console.log(loc);
  console.log(org);
```

```
  console.log(name);
  const { config } = usePrepareContractWrite({
    address: Bote,
    abi: abi_Bote,
    functionName: "addRet",
    args: [loc, org, name],
  });
  const { writeAsync, isLoading, isSuccess } = useContractWrite(config);

  if (isSuccess) {
    return (
      <Typography variant="h5" justifyContent="center" display="flex" mt="15%" >
        Retailer successfully added
      </Typography>
    );
  }
  if (isLoading) {
    return (
      <Backdrop
        open={true}
        sx={{ color: "#3d6f79", zIndex: (theme) => theme.zIndex.drawer + 1 }}
      >
        <CircularProgress color="inherit"></CircularProgress>
      </Backdrop>
    );
  }
  return (
    <>
      <Typography display="flex" justifyContent="center" mt="20%">
        <Button onClick={writeAsync}>Confirm</Button>
      </Typography>
    </>
  );
}

export default ConfirmRet;
```

Navbar.jsx:

```
```

```
import * as React from "react";
import AppBar from "@mui/material/AppBar";
import Box from "@mui/material/Box";
import Toolbar from "@mui/material/Toolbar";
import IconButton from "@mui/material/IconButton";
import Typography from "@mui/material/Typography";
import Menu from "@mui/material/Menu";
import MenuIcon from "@mui/icons-material/Menu";
import Container from "@mui/material/Container";
import Button from "@mui/material/Button";
import MenuItem from "@mui/material/MenuItem";
import { Link } from "react-router-dom";
import { Tab, Tabs } from "@mui/material";
import { ConnectButton } from "@rainbow-me/rainbowkit";

const pages = ["AddMan", "AddDis","AddRet"]//,"AddProd", "AddProdDis",
"AddProdMan","AddProdRet","SellProd" ,"Verify" ,"ViewRoute"];

const ResponsiveAppBar = () => {
  const [anchorElNav, setAnchorElNav] = React.useState(null);
  const [anchorElLens,setanchorElLens]=React.useState(null);
  const [Profile,setProfile]=React.useState();

  const handleOpenNavMenu = (event) => {
    setAnchorElNav(event.currentTarget);
  };

  const handleCloseNavMenu = () => {
    setAnchorElNav(null);
  };

  // function changeColor(event) {
  //   console.log(event);
  //   event.target.style.color = "Red";
  // }
  function handler() {
    handleCloseNavMenu();
  }

  function SignUpHandler(){

  }

  //  FOR PRESERVED COLORS
```

```
  // const [isClicked, setisClicked] = React.useState(["False", "False",
"False"]);

  return (
    <AppBar position="static" sx={{ backgroundColor: "#a31b5d", zIndex: "1" }}>
      <Container maxWidth="xl">
        <Toolbar disableGutters>
          <Typography
            variant="h6"
            noWrap
            component="a"
            href="/"
            sx={{
              mr: 2,
              display: { xs: "none", md: "flex" },
              fontFamily: "monospace",
              fontWeight: 700,
              letterSpacing: ".3rem",
              color: "white",
              textDecoration: "none",
            }}
          >
            HOME
          </Typography>

          <Box sx={{ flexGrow: 1, display: { xs: "flex", md: "none" } }}>
            <IconButton
              size="large"
              aria-label="account of current user"
              aria-controls="menu-appbar"
              aria-haspopup="true"
              onClick={handleOpenNavMenu}
              color="inherit"
            >
              <MenuIcon />
            </IconButton>
            <Menu
              id="menu-appbar"
              anchorEl={anchorElNav}
              anchorOrigin={{
                vertical: "bottom",
                horizontal: "left",
              }}
              keepMounted
              transformOrigin={{
```

```
            vertical: "top",
            horizontal: "left",
          }}
          open={Boolean(anchorElNav)}
          onClose={handleCloseNavMenu}
          sx={{
            display: { xs: "block", md: "none" },
          }}
        >
          {pages.map((page) => (
            <MenuItem
              key={page}
              onClick={handleCloseNavMenu}
              sx={{
                backgroundColor: "#004b8f",
                "&:hover": { backgroundColor: "black" },
              }}
            >
              <Typography textAlign="center">
                <Link
                  to={`/${page}`}
                  style={{ textDecoration: "none", color: "white" }}
                >
                  {page}
                </Link>
              </Typography>
            </MenuItem>
          ))}
        </Menu>
      </Box>
      <Typography
        variant="h5"
        noWrap
        component="a"
        href=""
        sx={{
          mr: 2,
          display: { xs: "flex", md: "none" },
          flexGrow: 1,
          fontFamily: "monospace",
          fontWeight: 700,
          letterSpacing: ".3rem",
          color: "inherit",
          textDecoration: "none",
        }}
```

```
          >

          </Typography>
          <Box sx={{ flexGrow: 1, display: { xs: "none", md: "flex" } }}>
            {pages.map((page) => (
              <Button
                key={page}
                onClick={handler}
                sx={{ my: 2, color: "red", display: "block" }}
              >
                <Link
                  to={`/${page}`}
                  style={{ textDecoration: "none", color: "#ffffff" }}
                >
                  {page}
                </Link>
              </Button>
            ))}
          </Box>
          {/* {
            Profile ? (
              <div>user profile</div>
            ):(
              <div>
              <Link to="/Login">
              <Button variant="contained">Login</Button>
              </Link>

              <Link to="/SignUp">
              <Button variant="contained" sx={{ ml: "8px" }}
onClick={SignUpHandler}>
                   Sign Up
              </Button>
              </Link>

              </div>
            )
          } */}
          <ConnectButton/>
        </Toolbar>
      </Container>
    </AppBar>
  );
};
export default ResponsiveAppBar;
```

Home.jsx:

```
import { Typography } from '@mui/material';
import React,{ Component } from 'react';
function Home() {
    const myStyle={
        backgroundImage:
 "D:\\programs\\SCM\\src\\background.avif",
        height:'100vh',
        marginTop:'-70px',
        fontSize:'50px',
        backgroundSize: 'cover',
        backgroundRepeat: 'no-repeat',
    };
    return (
      <div style={myStyle}>
    <Typography variant='h3' justifyContent="center" display="flex" mt="20%">
    Welcome
</Typography>
<Typography variant='h6' justifyContent="center" display="flex" mt="23%">
  Supply chain management
</Typography>
      </div>
    );
  }

export default Home
```

GetCard.jsx:

```
import {
  Card,
  CardActions,
  CardContent,
  Typography,
  Button,
  Backdrop,
  CircularProgress,
```

```
} from "@mui/material";
import React, { useState } from "react";
import { SCM, abi_SCM } from "../new_deployments";
import { useAccount, useContractWrite, usePrepareContractWrite } from "wagmi";

function GetCard({ pollId, state }) {
  const [Clicked, setClicked] = useState(false);
  const { address } = useAccount();
  const { config } = usePrepareContractWrite({
    address: SCM,
    abi: abi_SCM,
    functionName: "startPoll",
    args: [pollId],
    overrides:{from:address}
  });
  const { writeAsync, isLoading, isSuccess } = useContractWrite(config);
  if(isSuccess){
    return(
        <Typography variant="h5" justifyContent="center" display="flex" mt="15%">
        Poll started!
      </Typography>
    )
  }
  if (isLoading) {
    return (
      <Backdrop
        open={true}
        sx={{ color: "#fff", zIndex: (theme) => theme.zIndex.drawer + 1 }}
      >
        <CircularProgress color="inherit"></CircularProgress>
      </Backdrop>
    );
  }
  return (
    <div>
      <Card>
        <CardContent>
          <Typography gutterBottom variant="h5" component="div">
            {pollId}
          </Typography>
        </CardContent>
        <CardActions>
          {state == 2 ? (
            Clicked ? (
              <Button size="small" onClick={writeAsync}>
```

```
              Confirm
            </Button>
          ) : (
            <Button
              size="small"
              onClick={() => {
                setClicked(true);
              }}
            >
              Start
            </Button>
          )
        ) : (
          <></>
        )}
      </CardActions>
    </Card>
  </div>
 );
}

export default GetCard;
```

App.js:

```
import "./App.css";
import { Route, Routes, Link, Router } from "react-router-dom";
import Home from "./components/Home";
import ResponsiveAppBar from "./components/Navbar";
import AddMan from "./components/AddMan";
import AddDis from "./components/AddDis";
import AddRet from "./components/AddRet";
import "@rainbow-me/rainbowkit/styles.css";
import { getDefaultWallets, RainbowKitProvider,darkTheme } from "@rainbow-
me/rainbowkit";
import {
  chain,
  configureChains,
  createClient,
  useSigner,
  WagmiConfig,
} from "wagmi";
```

```
import { alchemyProvider } from "wagmi/providers/alchemy";
import { publicProvider } from "wagmi/providers/public";
import { ThemeProvider, createTheme } from '@mui/material/styles';
import CssBaseline from '@mui/material/CssBaseline';

const darkTheme_MUI = createTheme({
  palette: {
    mode: 'dark',
  },
});

const { chains, provider, webSocketProvider } = configureChains(
  [chain.polygonMumbai],
  [
    alchemyProvider({ apiKey: "9Wc5W1c7UekbGVThR6XdflYMiWEKQ9p5" }),
    publicProvider(),
  ]
);

const { connectors } = getDefaultWallets({
  appName: "RainbowKit demo",
  chains,
});

const wagmiClient = createClient({
  autoConnect: true,
  connectors,
  provider,
  webSocketProvider,
});

function App() {
  return (
    <>
    <ThemeProvider theme={darkTheme_MUI}>
      <CssBaseline />
      <WagmiConfig client={wagmiClient}>
        <RainbowKitProvider chains={chains} theme={darkTheme()}>
          <ResponsiveAppBar></ResponsiveAppBar>
        </RainbowKitProvider>
      </WagmiConfig>
      <WagmiConfig client={wagmiClient}>
        <Routes>
          <Route path="/" element={<Home></Home>}></Route>
          {/* <Route path="/AddProd" element={<Addprod></Addprod>}></Route> */}
```

```
          <Route path="/AddDis" element={<AddDis></AddDis>}></Route>
          <Route path="/AddMan" element={<AddMan></AddMan>}></Route>
          <Route path="/AddRet" element={<AddRet></AddRet>}></Route>
          {/* <Route path="/AddProdDis"
element={<AddProdDis></AddProdDis>}></Route>
          <Route path="/AddProdMan" element={<AddProdMan></AddProdMan>}></Route>
          <Route path="/AddProdRet" element={<AddProdRet></AddProdRet>}></Route>
          <Route path="/SellProd" element={<SellProd></SellProd>}></Route>
          <Route path="/Verify" element={<Verify></Verify>}></Route>
          <Route path="/ViewRoute" element={<ViewRoute></ViewRoute>}></Route> */}
        </Routes>
      </WagmiConfig>
      </ThemeProvider>
    </>
  );
}

export default App;
```

new_deployments.js:

```
export const SCM= "0x64df6D13293Fbd3aBceE0d43DCeb6367DDE0BAcb";
export const abi_SCM = [<contract ABI>]
```

App.test.js:

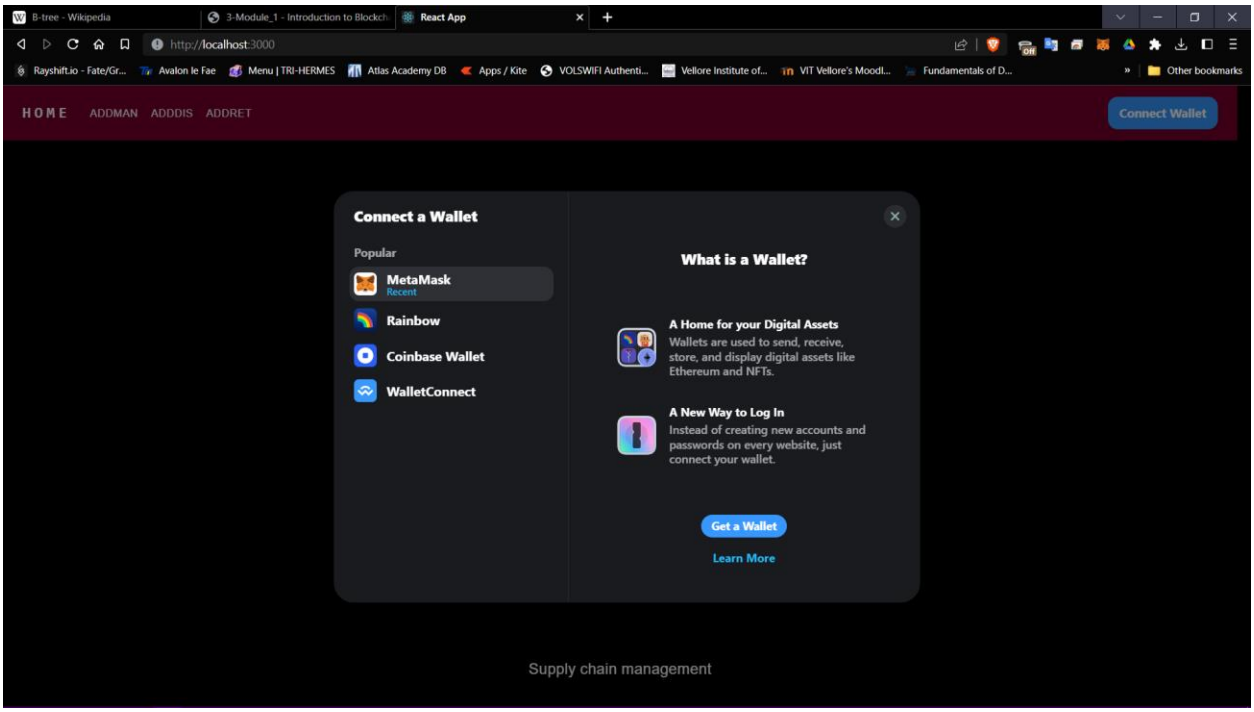
```
import { render, screen } from '@testing-library/react';
import App from './App';

test('renders learn react link', () => {
  render(<App />);
  const linkElement = screen.getByText(/learn react/i);
  expect(linkElement).toBeInTheDocument();
});
```

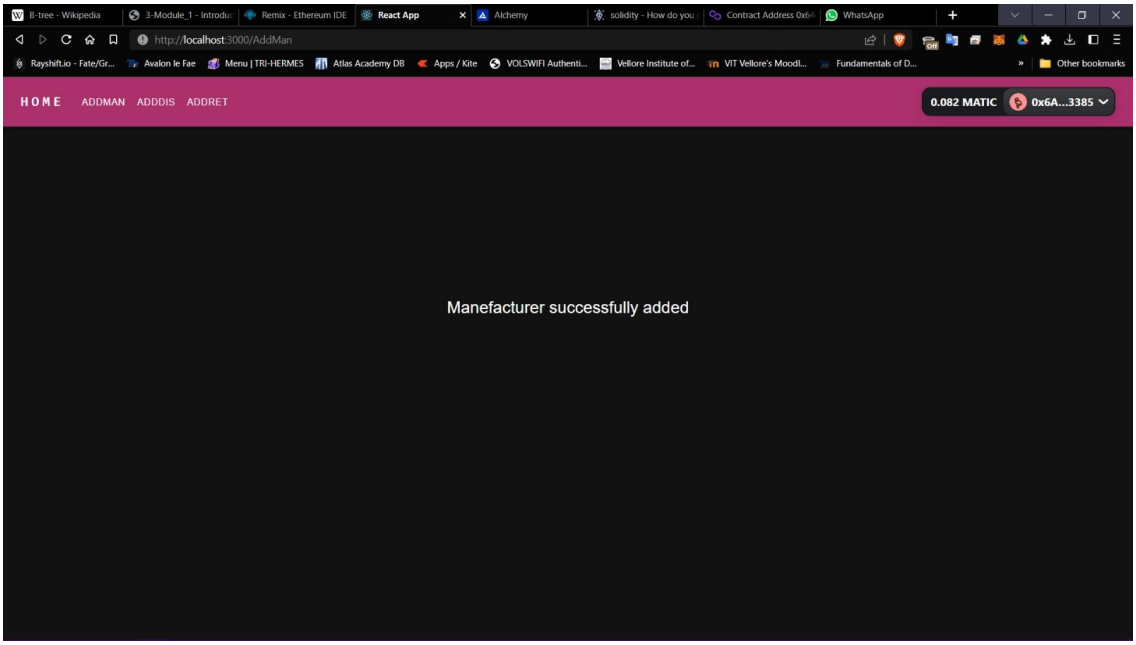
Index.css:

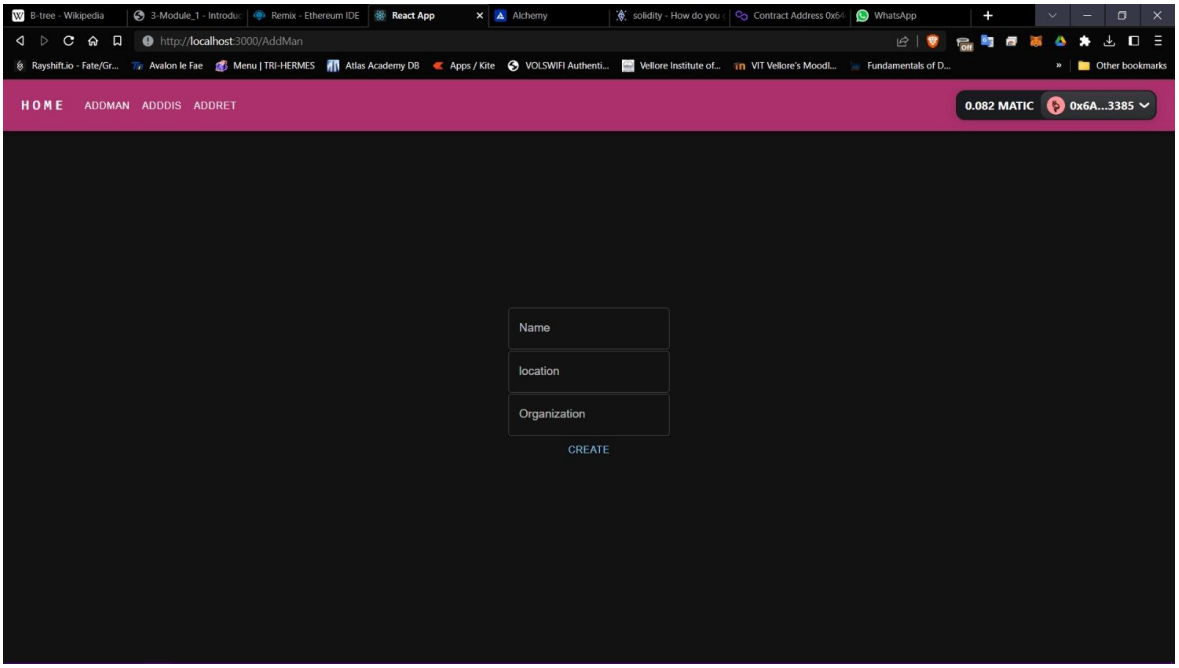
```
body {
  margin: 0;
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
    'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
    sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}

code {
  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
    monospace;
}
```

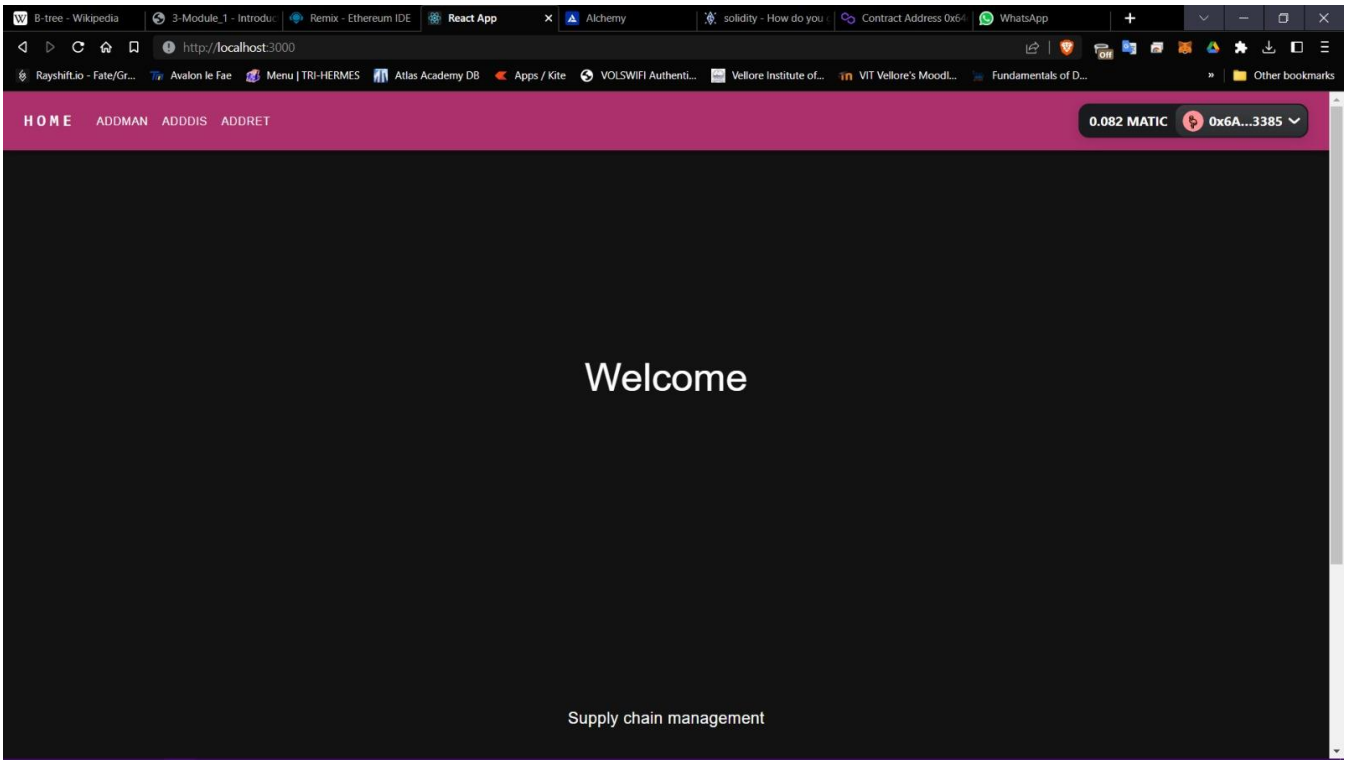
7.2 APPENDIX 2 - SCREEN SHOTS /OUTPUTs

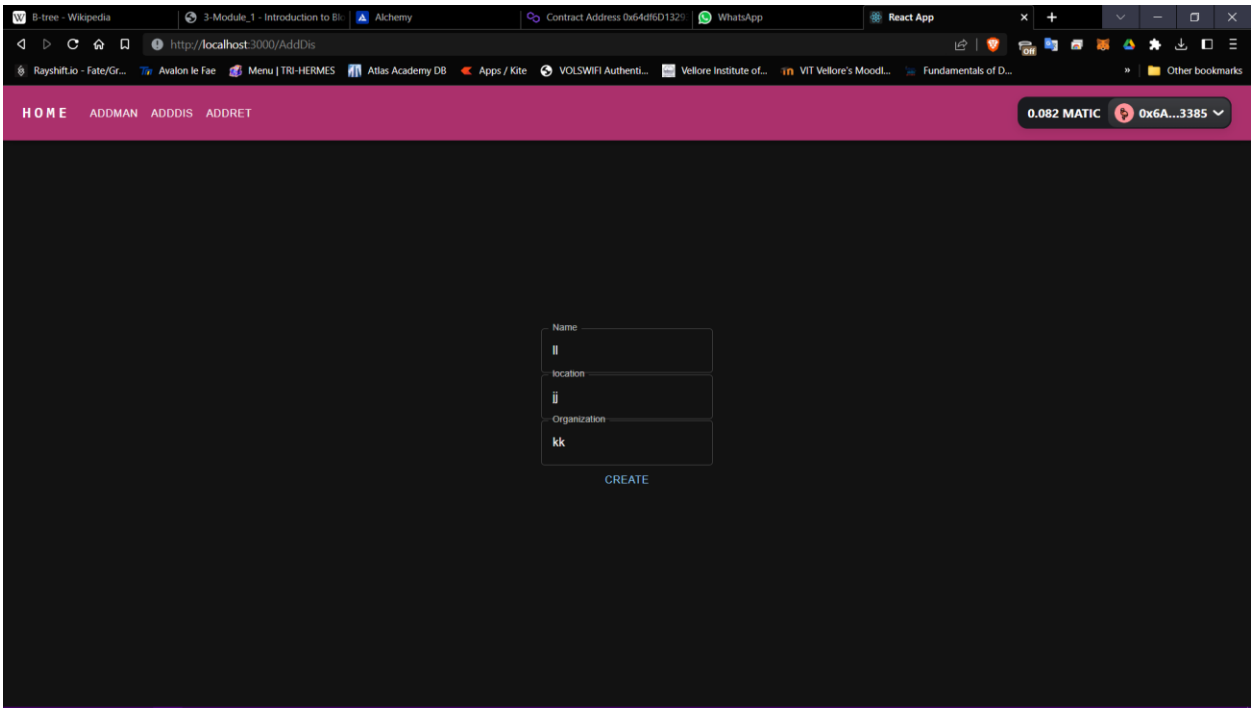
Logging in:

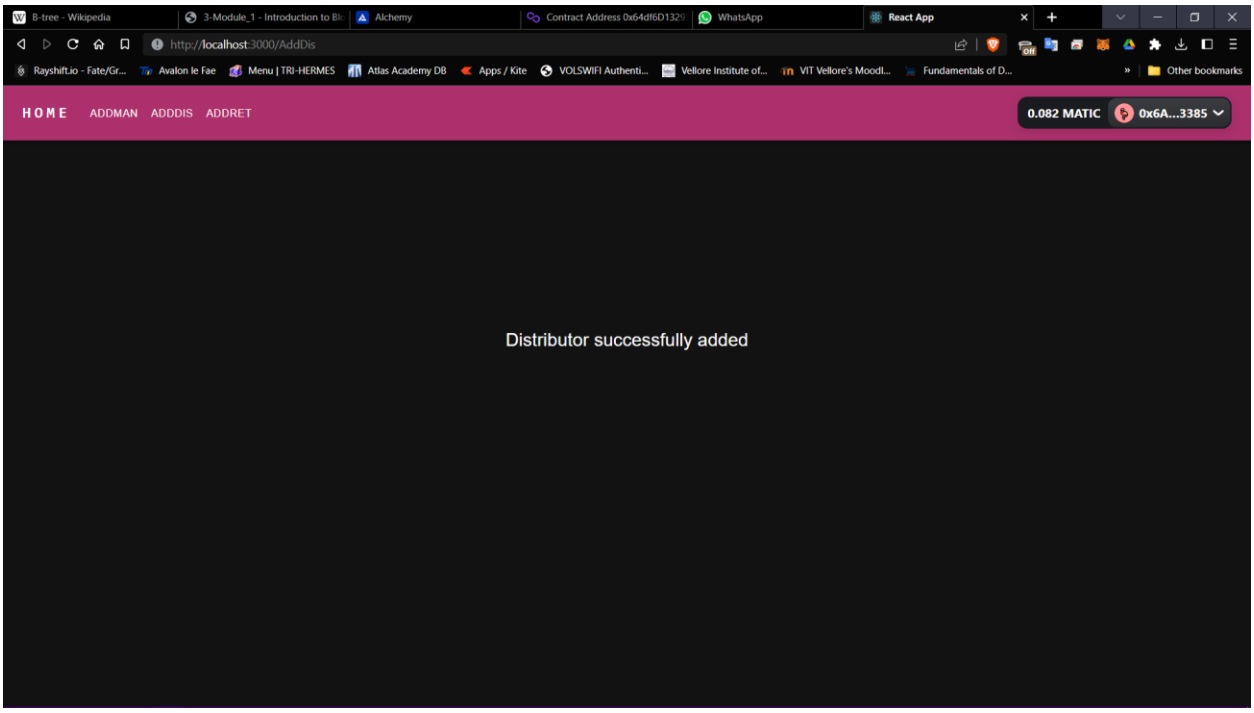


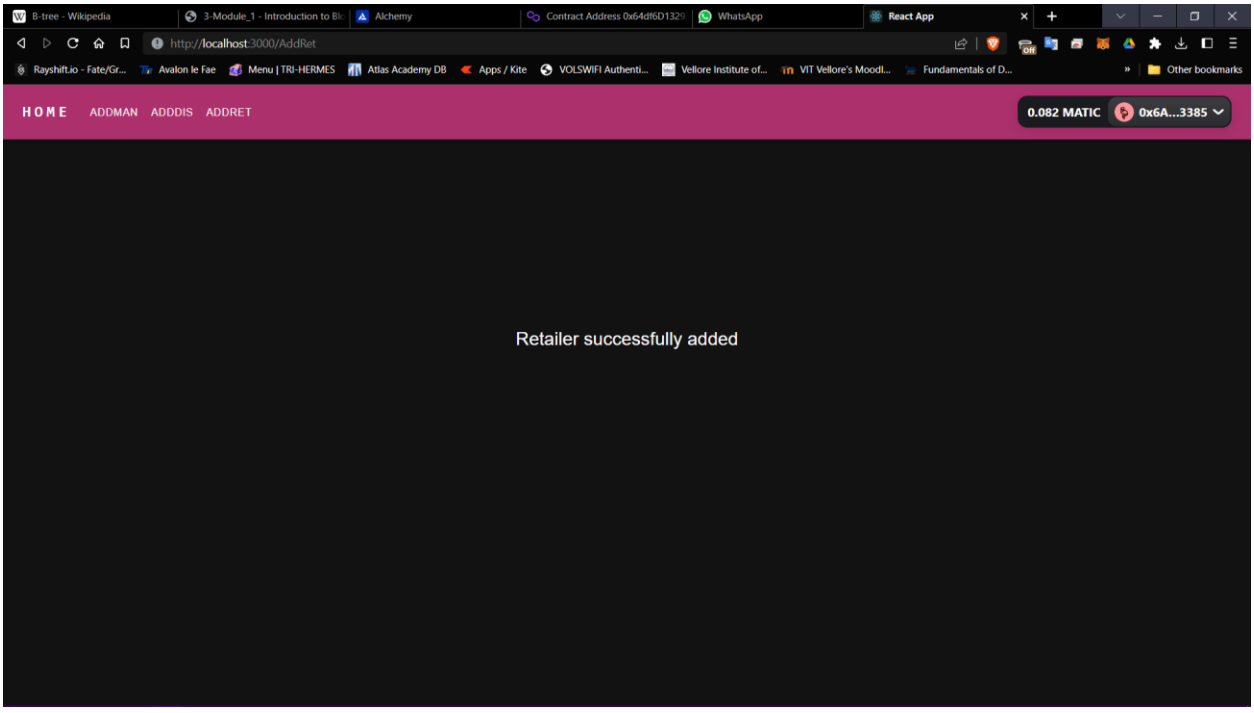

B-tree - Wikipedia
3-Module_1 - Introduc
Remix - Ethereum IDE
React App
Alchemy
solidity - How do you
Contract Address 0x64
WhatsApp
http://localhost:3000/AddMan
Rayshift.io - Fate/Gr...
Avalon le Fae
Menu | TRI-HERMES
Atlas Academy DB
Apps / Kite
VOLSWIFI Authenti...
Vellore Institute of...
VIT Vellore's Moodl...
Fundamentals of D...
Other bookmarks
HOME
ADDMAN
ADDDIS
ADDRET
0.082 MATIC
0x6A...3385
Manefacturer successfully added

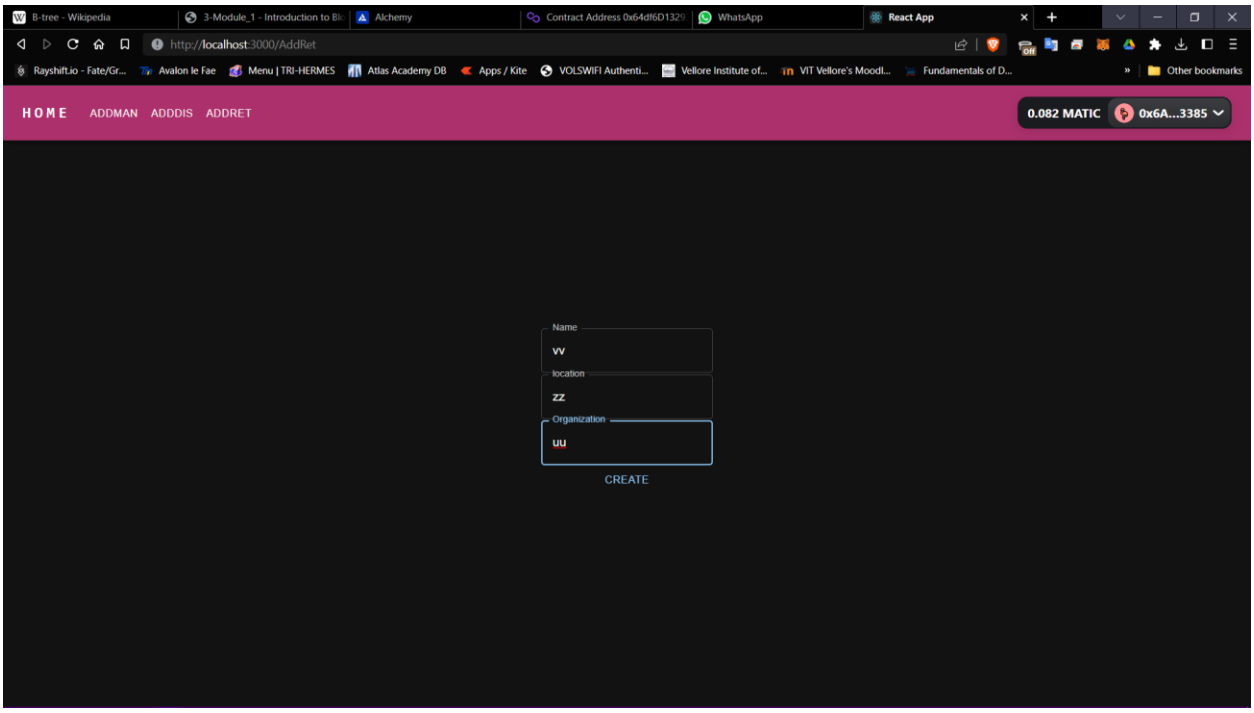

B-tree - Wikipedia
3-Module_1 - Introduc
Remix - Ethereum IDE
React App
Alchemy
solidity - How do you
Contract Address 0x64
WhatsApp
http://localhost:3000/AddMan
Rayshift.io - Fate/Gr...
Avalon le Fae
Menu | TRI-HERMES
Atlas Academy DB
Apps / Kite
VOLSWIFI Authenti...
Vellore Institute of...
VIT Vellore's Moodl...
Fundamentals of D...
Other bookmarks
HOME
ADDMAN
ADDDIS
ADDRET
0.082 MATIC
0x6A...3385
Name
location
Organization
CREATE

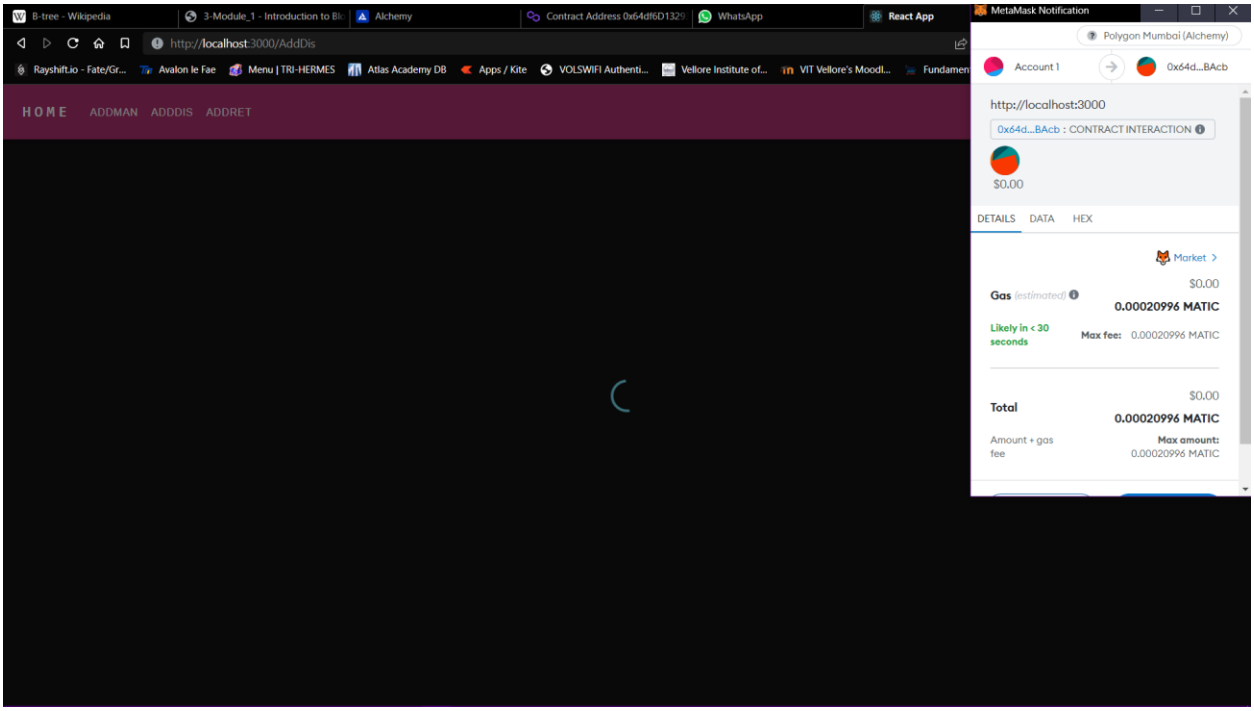

B-tree - Wikipedia
3-Module_1 - Introduc
Remix - Ethereum IDE
React App
Alchemy
solidity - How do you
Contract Address 0x64
WhatsApp
http://localhost:3000
Rayshift.io - Fate/Gr...
Avalon le Fae
Menu | TRI-HERMES
Atlas Academy DB
Apps / Kite
VOLSWIFI Authenti...
Vellore Institute of...
VIT Vellore's Moodl...
Fundamentals of D...
Other bookmarks
HOME
ADDMAN
ADDDIS
ADDRET
0.082 MATIC
0x6A...3385
Welcome
Supply chain management


B-tree - Wikipedia
3-Module_1 - Introduction to Bl
Alchemy
Contract Address 0x64df6D1329
WhatsApp
React App
http://localhost:3000/AddDis
Rayshift.io - Fate/Gr...
Avalon le Fae
Menu | TRI-HERMES
Atlas Academy DB
Apps / Kite
VOLSWIFI Authenti...
Vellore Institute of...
VIT Vellore's Moodl...
Fundamentals of D...
Other bookmarks
HOME
ADDMAN
ADDDIS
ADDRET
0.082 MATIC
0x6A...3385
Name
ll
location
jj
Organization
kk
CREATE

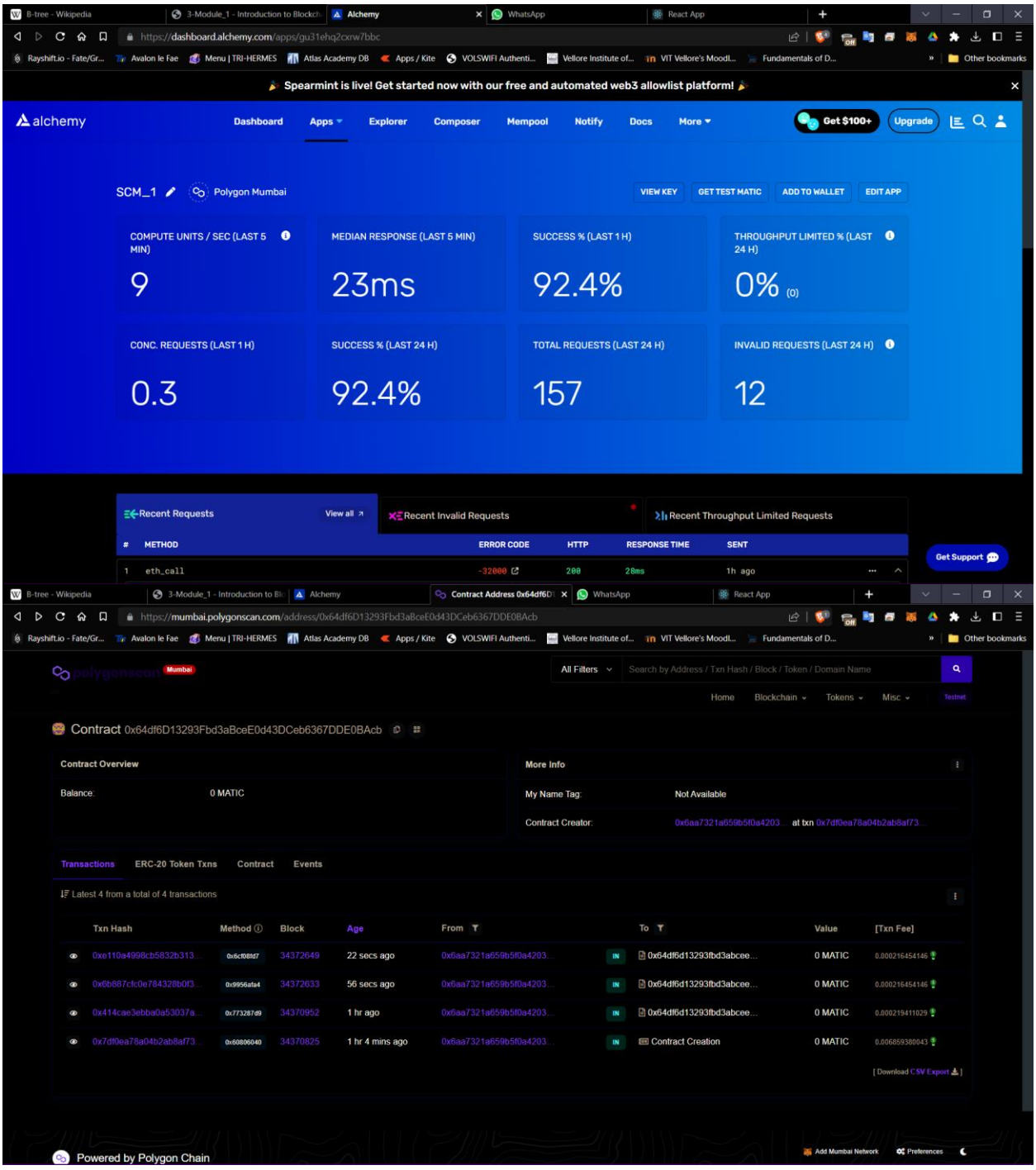

B-tree - Wikipedia
3-Module_1 - Introduction to Bl
Alchemy
Contract Address 0x64df6D1329
WhatsApp
React App
http://localhost:3000/AddDis
Rayshift.io - Fate/Gr...
Avalon le Fae
Menu | TRI-HERMES
Atlas Academy DB
Apps / Kite
VOLSWIFI Authenti...
Vellore Institute of...
VIT Vellore's Moodl...
Fundamentals of D...
Other bookmarks
HOME
ADDMAN
ADDDIS
ADDRET
0.082 MATIC
0x6A...3385
Distributor successfully added


B-tree - Wikipedia
3-Module_1 - Introduction to Bl
Alchemy
Contract Address 0x64df6D1329
WhatsApp
React App
http://localhost:3000/AddRet
Rayshift.io - Fate/Gr...
Avalon le Fae
Menu | TRI-HERMES
Atlas Academy DB
Apps / Kite
VOLSWIFI Authenti...
Vellore Institute of...
VIT Vellore's Moodl...
Fundamentals of D...
Other bookmarks
HOME
ADDMAN
ADDDIS
ADDRET
0.082 MATIC
0x6A...3385
Retailer successfully added





Contract deployed on polygon testnet:



8 REFERENCES

https://www.tandfonline.com/doi/abs/10.1080/00207543.2018.1533261 "Current state of blockchain in supply chain management" (source: Journal of the Operational Research Society, 2018): The possible applications of blockchain technology in supply chain management are covered in this article, along with

advantages including improved traceability and transparency. The authors give examples of businesses that have integrated blockchain technology into their supply chains while also highlighting some of the difficulties and restrictions associated with doing so.

https://link.springer.com/chapter/10.1007/978-3-540-74512-9_2 "Processes involved in supply chain management" (source: Lecture Notes in Computer Science, 2007): An overview of the various supply chain management processes, including production, shipping, warehousing, and customer service, is given in this chapter. The writers go over the significance of each process, the difficulties in managing them well, and some of the tools and methods that can be applied to enhance supply chain performance.

https://www.nber.org/papers/w1876 "Effects of counterfeit trade" (source: National Bureau of Economic Research, 1986): The economic consequences of the counterfeit trade are examined in this essay, along with how it affects consumer welfare, manufacturer profitability, and tax income. The authors investigate some of the policy measures that can be utilised to combat it as well as some of the causes that contribute to the expansion of the counterfeit trade, such as changes in trade rules and technological advancements.

https://link.springer.com/chapter/10.1007/978-3-540-71641-9_9 "Product authentication through RFID" (source: Lecture Notes in Computer Science, 2007): In order to confirm that a product is authentic and unaltered, this chapter examines the usage of RFID technology for product authentication. The writers go over the many parts of an RFID system, how it may be used to track products along the supply chain, some of the difficulties in putting one in place, and some of the benefits. They also go over some possible advantages of employing RFID for product authentication, like boosted consumer confidence and enhanced supply chain efficiency.