FOOD TRACKING SYSTEM

PROJECT DOCUMENTATION

Submitted by

Team ID:	NM2023TMID03807
Team Leader:	KARTHIKA P
Team Member:	SAROJINI P
Team Member:	PRIYADHARSHNI A
Team Member:	DHEENA FATHIMA N

INDEX

1. INTRODUCTION

1.1Project Overview:

The Advanced Food Tracking System is a state-of-the-art platform leveraging cutting-edge technology to revolutionize the way we monitor, manage, and analyze food-related information. Built upon a foundation of robust hardware and software components, it employs advanced techniques to ensure accuracy, security, and efficiency in tracking food from production to consumption.

Smart Sensors and IoT Devices:

Utilizes advanced sensors and Internet of Things (IoT) devices to collect real-time data on food production, storage, and transportation. Sensors monitor temperature, humidity, location, and other relevant parameters to ensure optimal conditions throughout the supply chain.

Blockchain Technology:

Implements blockchain for secure and immutable record-keeping. Each transaction in the food supply chain is recorded in a tamper-proof ledger, providing transparency and traceability. Artificial Intelligence (AI) and Machine Learning (ML): Employs AI algorithms to analyze data patterns, predict demand, and optimize logistics for efficient distribution. ML models are used to detect anomalies, ensuring quality control and preventing contamination or spoilage.

Augmented Reality (AR) and Virtual Reality (VR):

Integrates AR and VR for enhanced user experiences. This can include visualizing supply chain routes, providing training simulations, and assisting in quality inspections.

Big Data Analytics:

Harnesses the power of big data analytics to process and derive meaningful insights from vast amounts of food-related data. This aids in optimizing inventory, reducing waste, and improving overall efficiency.

Biometric Authentication:

Implements advanced biometric authentication methods to ensure that only authorized personnel access critical data, enhancing security and compliance.

Machine Vision and Image Recognition:

Utilizes machine vision to identify and categorize food products, aiding in automated quality control and inventory management.

Geo-fencing and GPS Tracking:

Incorporates geo-fencing and GPS tracking for real-time monitoring of food shipments, allowing for precise location tracking and ensuring compliance with designated routes.

1.2Project Purpose:

Enhanced Food Safety:

Minimizes the risk of contamination and spoilage through real-time monitoring and quality control measures.

Supply Chain Efficiency:

Optimizes logistics and inventory management, reducing waste and ensuring timely delivery.

Transparency and Traceability:

Provides consumers with detailed information about the origin and journey of their food, fostering trust and confidence.

Compliance and Regulation

Facilitates adherence to regulatory standards by maintaining comprehensive and auditable records.

Data-Driven Decision Making:

Empowers stakeholders with actionable insights derived from extensive data analysis. **Reduced Environmental Impact**:

Minimizes food waste and energy consumption through optimized supply chain processes.

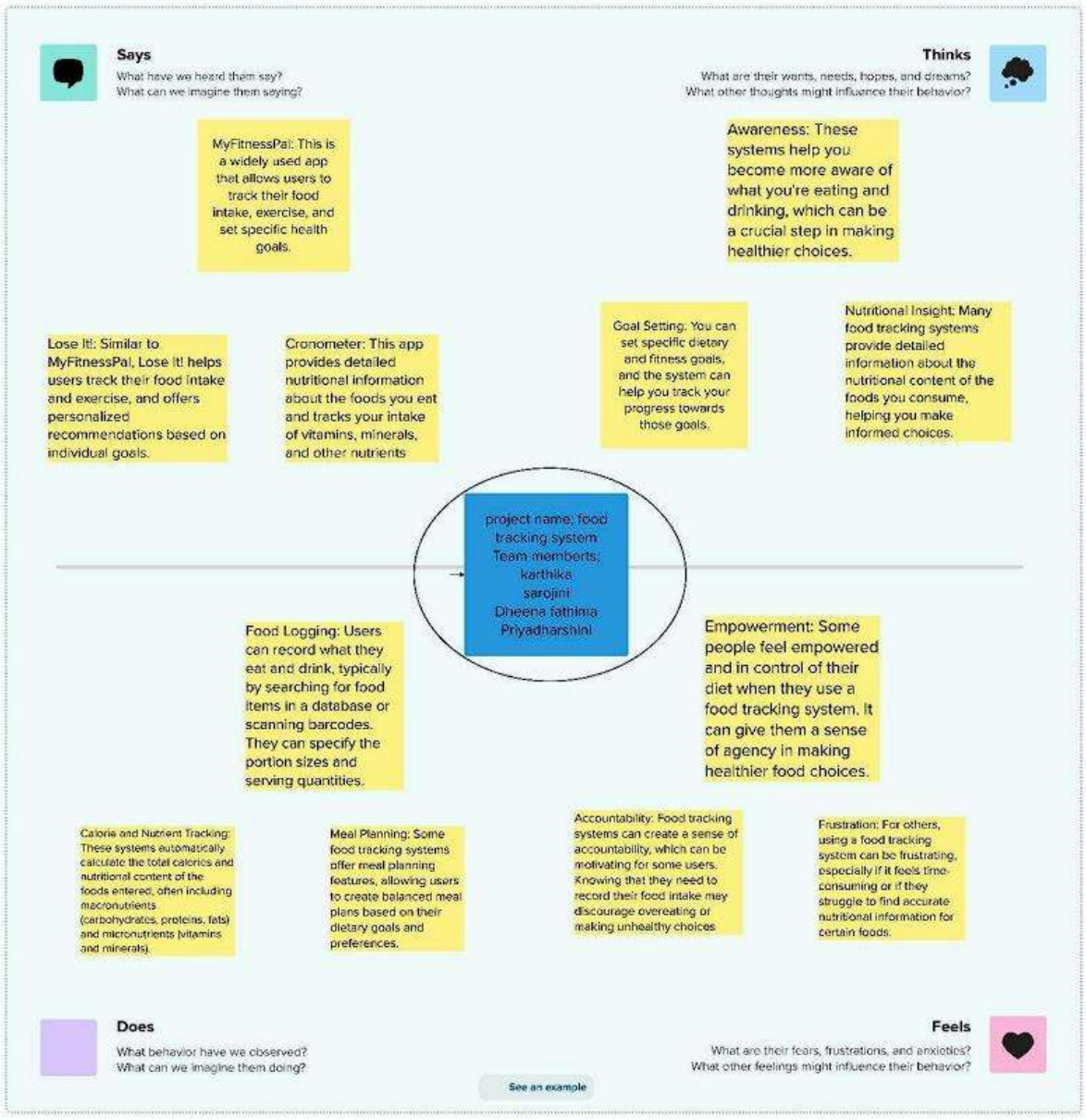
2. IDEATION & PROPOSED SOLUTION

2.1 Empathy Map Canvas

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user behavior and Attitudes.

It is a useful tool to helps teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the use

VACCINE TRACKING

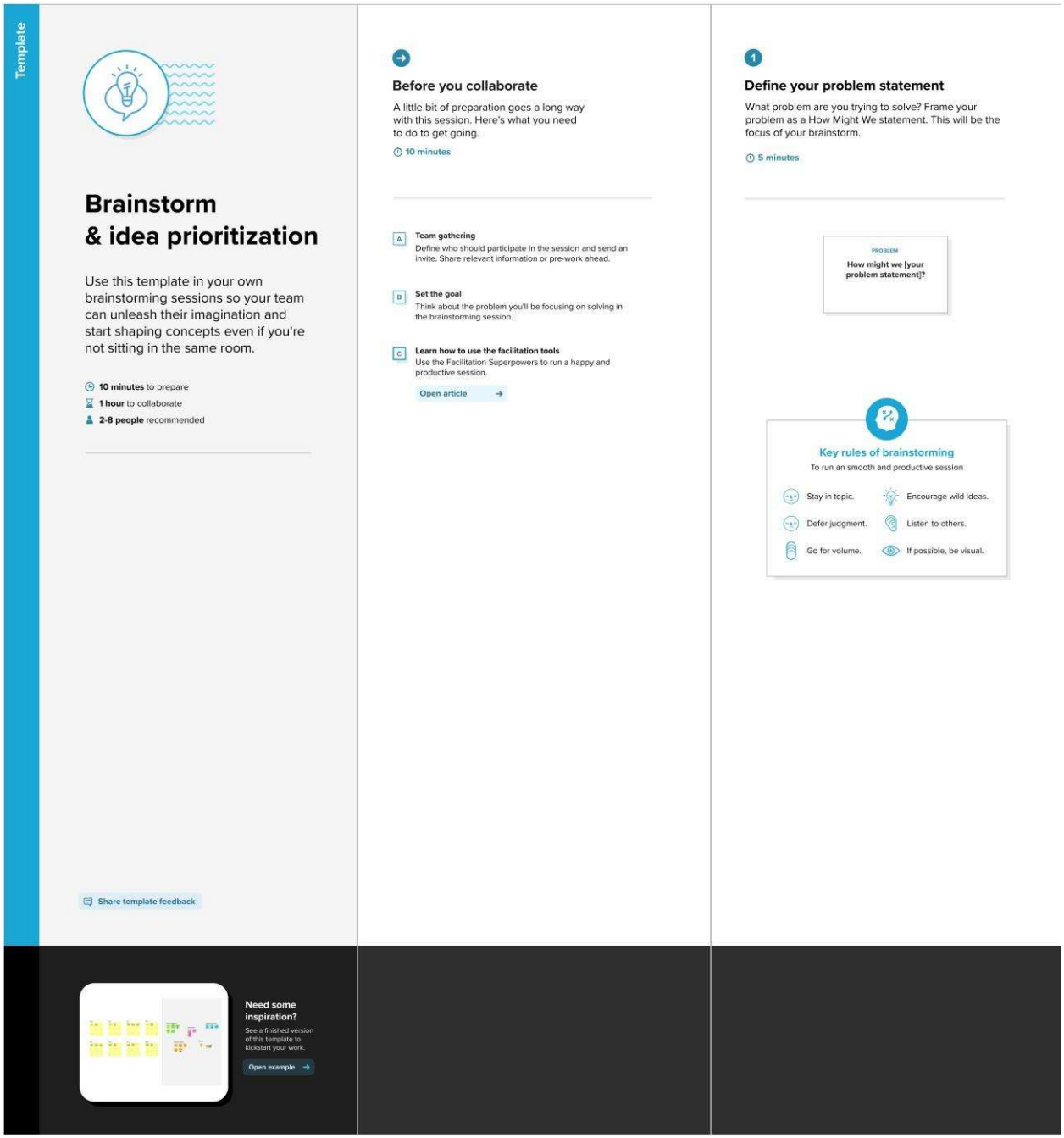


2.2 Ideation & Brainstorming

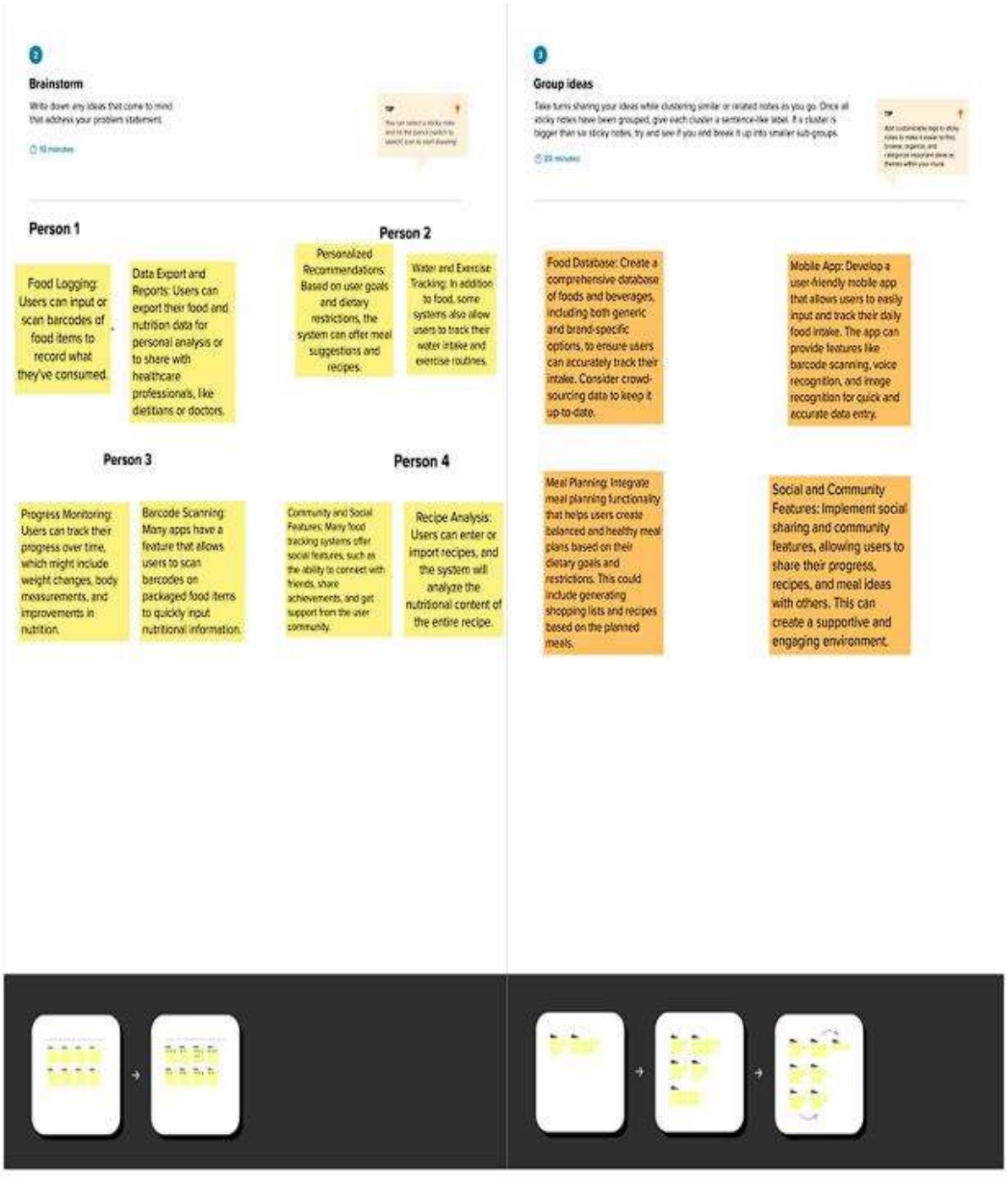
Brainstorming provides a free and open environment that encourages everyone within a team participate in the creative thinking process thatleads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the dame room.

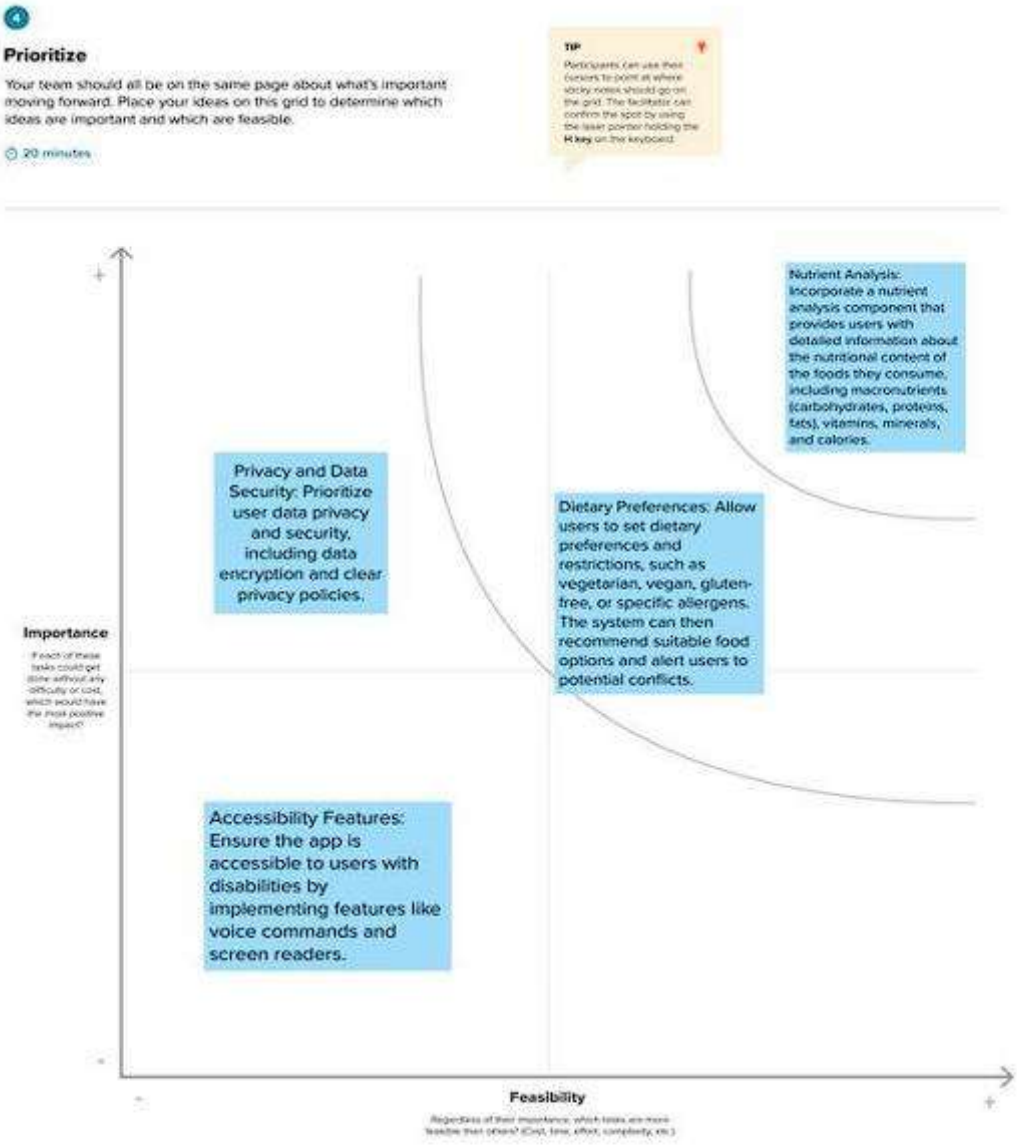
Step-1: Team Gathering, Collaboration and Select the Problem Statement:

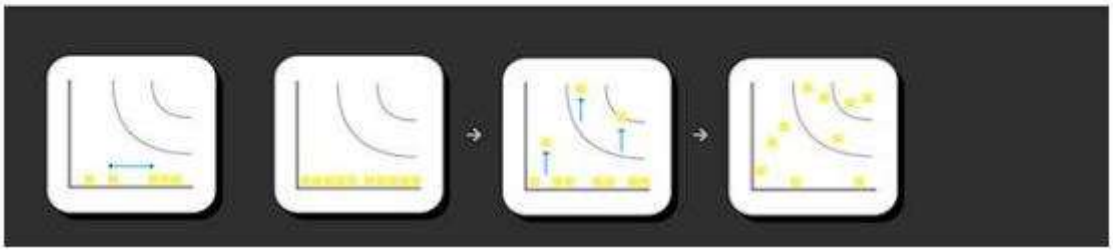


Step-2: Brainstorm, Idea Listing and grouping



Step-3: Idea Prioritization





3. REQUIREMENT ANALYSIS

3.1Functional Requirements:

Real-time Sensors and IoT Devices: Incorporate sensors that can monitor food temperature, freshness, and other relevant parameters in real-time. IoT devices can be used to transmit this data to a central processing unit.

Artificial Intelligence (AI): Implement AI algorithms to process the data from sensors. This could include image recognition for identifying food items, natural language processing for understanding user input, and machine learning for predicting food spoilage.

Mobile App or Web Interface: Develop a user-friendly interface for consumers to interact with the system. This could include features like barcode scanning, voice recognition, or image recognition to input food items.

Database and Cloud Storage: Store data on a robust, scalable database or cloud platform to ensure easy access and data integrity.

User Profiles and Preferences: Allow users to create profiles with preferences like dietary restrictions, allergies, and taste preferences. This information can be used to provide personalized recommendations.

Nutrition Database and Analysis: Integrate a comprehensive nutrition database to provide detailed information about the nutritional content of various foods.

Alerts and Notifications: Implement a notification system that alerts users about impending food expirations, recalls, or offers personalized recipe suggestions based on available ingredients.

3.2Non-functional Requirements:

Scalability and Performance: Ensure that the system can handle a large user base and process data efficiently, especially during peak usage times.

Security and Privacy: Implement strong security measures to protect user data, especially sensitive information like dietary restrictions and health conditions.

Reliability and Availability: The system should be available and reliable 24/7. Implement redundancy and failover mechanisms to minimize downtime.

Compliance and Regulations: Ensure the system complies with relevant food safety and privacy regulations, such as FDA guidelines, GDPR, or other local data protection laws.

Integration and Interoperability: Make sure the system can integrate with other platforms or services, like e-commerce sites, grocery delivery services, and recipe databases.

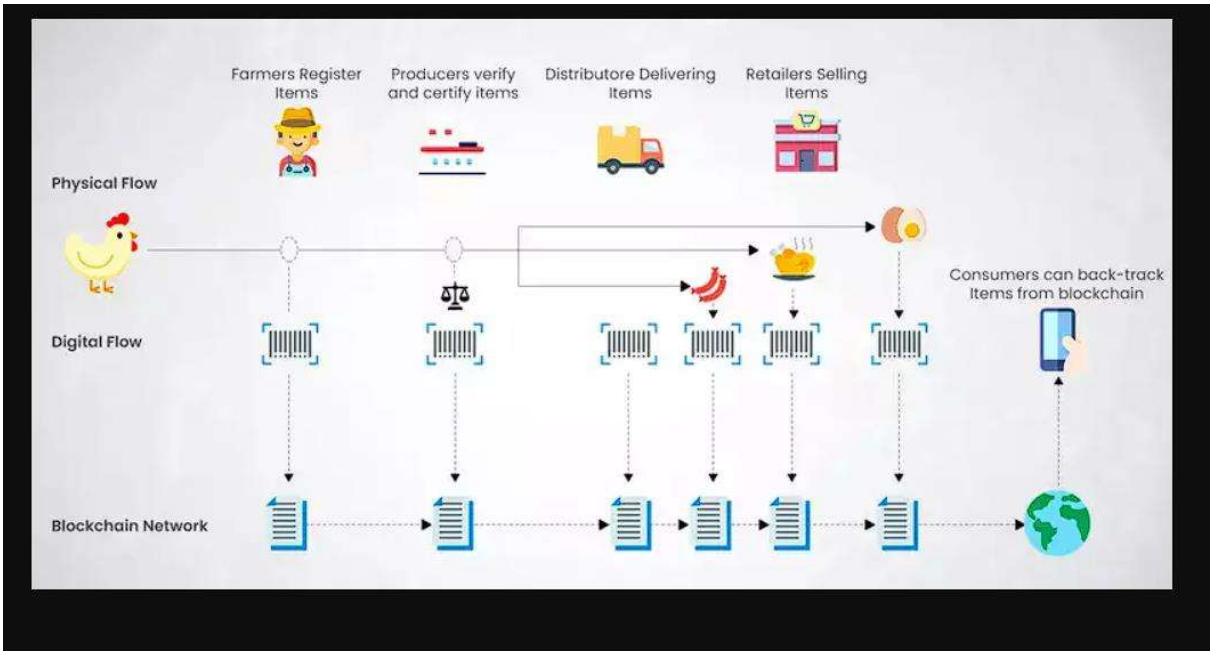
Usability and Accessibility: Design the user interface to be intuitive, accessible to people with disabilities, and available in multiple languages if applicable.

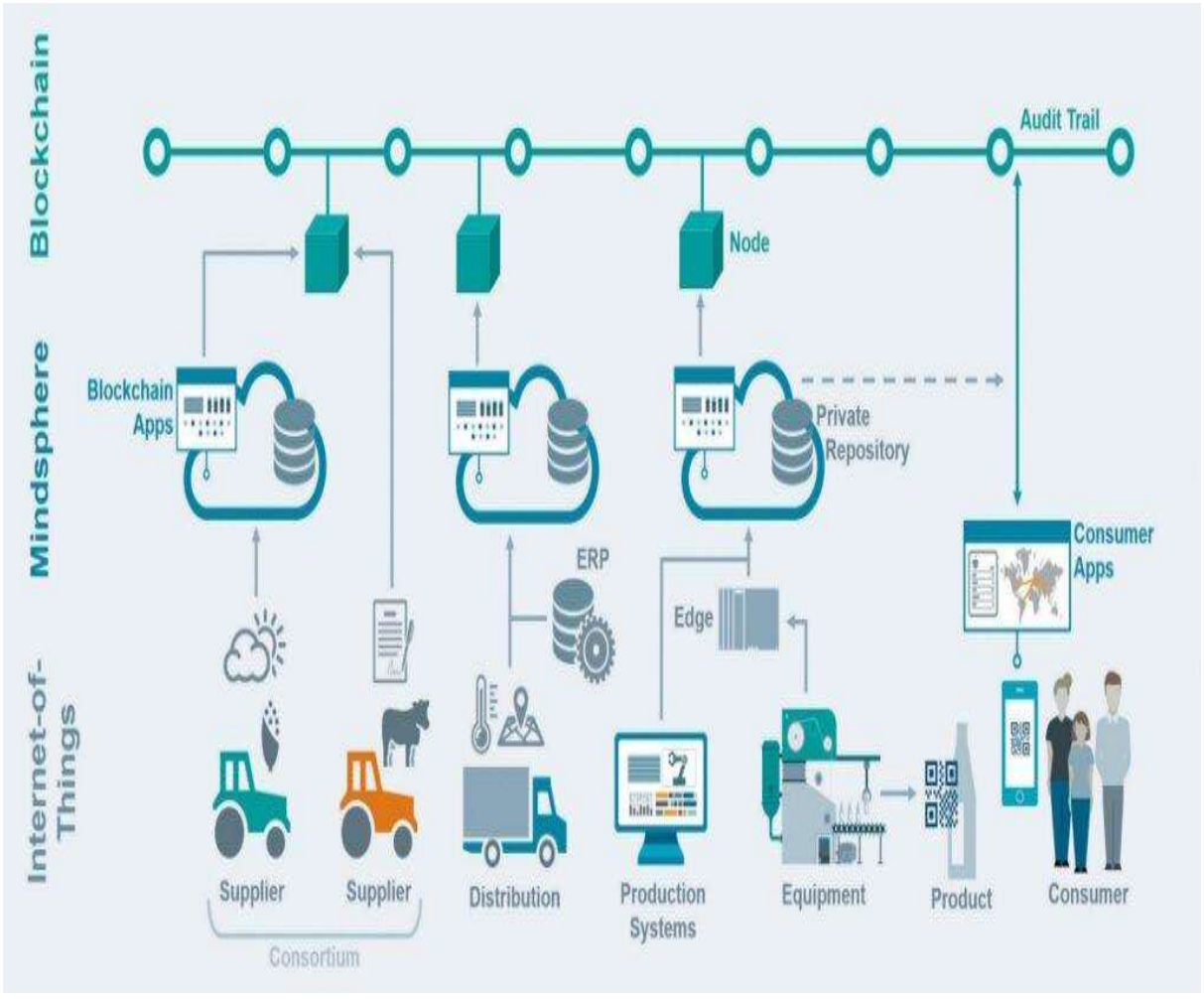
Feedback and Improvement: Incorporate mechanisms for user feedback to continuously improve the system based on user needs and preferences.

4. PROJECT DESIGN

4.1Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of thesystem requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



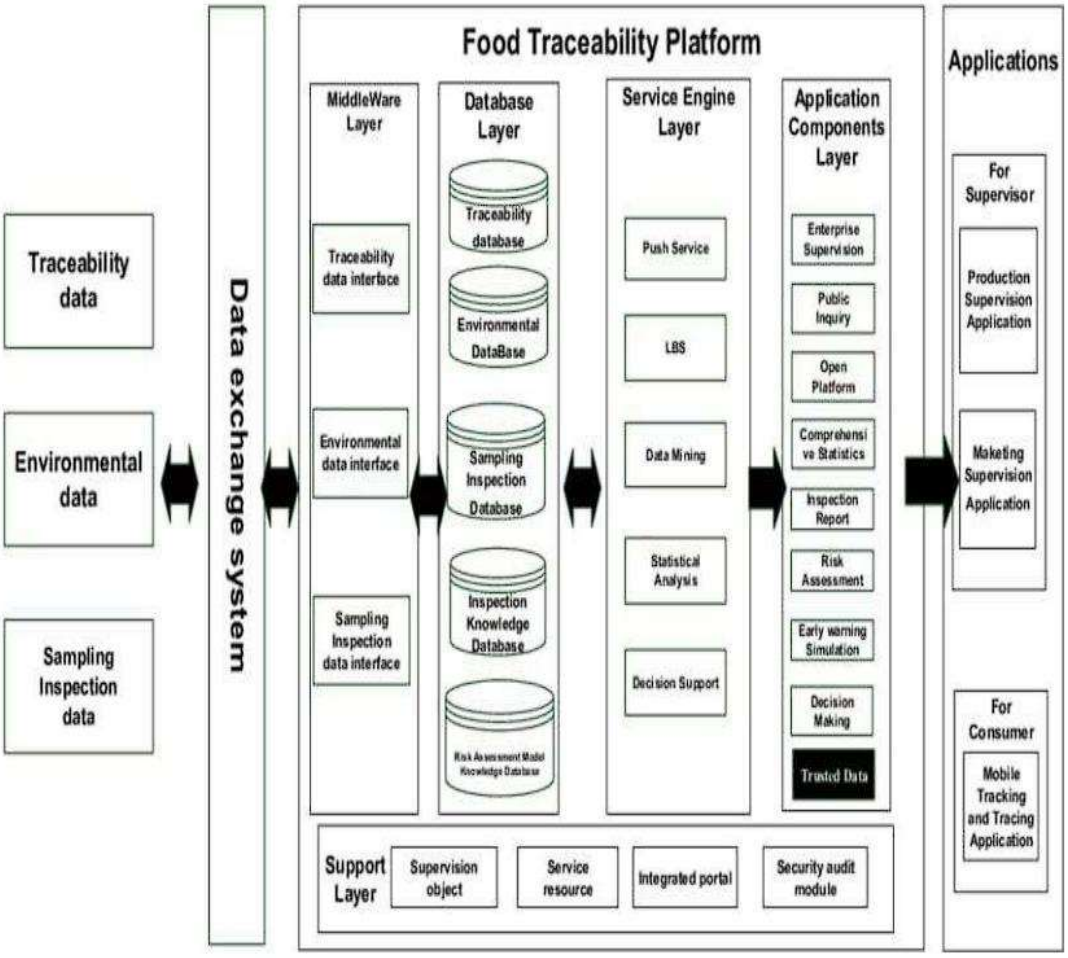

Blockchain
Mindsphere
Internet-of-Things
Audit Trail
Node
Blockchain Apps
Private Repository
ERP
Edge
Consumer Apps
Supplier
Supplier
Consortium
Distribution
Production Systems
Equipment
Product
Consumer

4.2Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gapbetween business problems and technology solutions.

Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed and delivered.


Traceability data
Environmental data
Sampling Inspection data
Data exchange system
Food Traceability Platform
MiddleWare Layer
Traceability data interface
Environmental data interface
Sampling Inspection data interface
Database Layer
Traceability database
Environmental DataBase
Sampling Inspection Database
Inspection Knowledge Database
Risk Assessment Model Knowledge Database
Service Engine Layer
Push Service
LBS
Data Mining
Statistical Analysis
Decision Support
Application Components Layer
Enterprise Supervision
Public Inquiry
Open Platform
Comprehensive Statistics
Inspection Report
Risk Assessment
Early warning Simulation
Decision Making
Trusted Data
Support Layer
Supervision object
Service resource
Integrated portal
Security audit module
Applications
For Supervisor
Production Supervision Application
Maketing Supervision Application
For Consumer
Mobile Tracking and Tracing Application

4.3User Stories:

- Creating a food tracking system using cutting-edge or "black" technology involves advanced and innovative features. Here are some user stories that could be considered for such a system:
- As a health-conscious user, I want to be able to scan barcodes on food items using my smartphone's camera, so that I can easily track the nutritional information and log it into my daily intake.
- As a user with dietary restrictions, I want the system to provide real-time alerts when a scanned item contains allergens or ingredients I need to avoid, ensuring I make safe food choices.
- As a fitness enthusiast, I want the system to integrate with my smartwatch or fitness tracker to automatically adjust my recommended calorie intake based on my activity level and goals.
- As a person with specific health goals, I want personalized meal plans generated based on my dietary preferences, restrictions, and fitness objectives, ensuring I stay on track.
- As a busy professional, I want the system to have a voice-activated assistant that allows me to log my meals and snacks quickly and efficiently, even when I'm on the go.
- As a parent, I want to have the ability to track the nutritional intake of my children and receive suggestions for balanced meals that cater to their specific needs and preferences.
- As a user interested in sustainability, I want the system to provide information on the environmental impact of the foods I consume, including carbon footprint and water usage, so I can make eco-conscious choices
- As a user concerned about food waste, I want the system to suggest recipes based on the ingredients I already have at home, reducing the likelihood of unused groceries.
- As a person aiming to improve their cooking skills, I want access to a database of recipes with step-by-step instructions and video tutorials, tailored to my skill level and available ingredients.
- As a user with a chronic health condition, I want the system to provide insights and recommendations on foods that can help manage my condition, based on the latest medical research and personalized data.
- As a social user, I want the option to connect with friends and family on the platform, share recipes, and see each other's progress towards health and fitness goals.
- As a user conscious about privacy, I want the system to have robust security measures in place to protect my personal data, and the option to control who has access to my food tracking information.

5.CODING & SOLUTIONING

Coding refers to the process of translating a software design into a functional program using programming languages like JavaScript, Python, or Java. It involves writing code, debugging, and optimizing algorithms to create a solution for a specific problem or requirement.

Solutioning involves designing a comprehensive solution strategy before coding. It includes problem analysis, architectural design, selecting appropriate technologies, and planning for scalability and security. Solutioning ensures that the software addresses the problem effectively and aligns with the project's goals.

5.1 FEATURE 1

Feature 1: User Authentication

```
function authenticateUser(username, password) {
// Code to validate username and password
        if (isValidCredentials(username, password)) {
                return "Authentication successful";
        }
        else {
                return "Authentication failed";
        }
}
function isValidCredentials(username, password) {
// Code to check credentials against database or backend
system
// Return true if valid, false otherwise
}
```

Explanation:

The code snippet checks the provided username and password against a database or backend system. If the credentials are valid, the function returns "Authentication successful"; otherwise, it returns "Authentication failed." This feature ensures secure user access to the system.

5.2FEATURE 2

Feature 2: Data Encryption

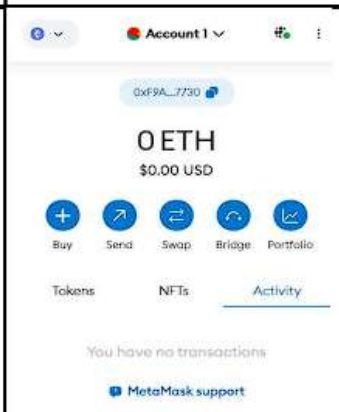
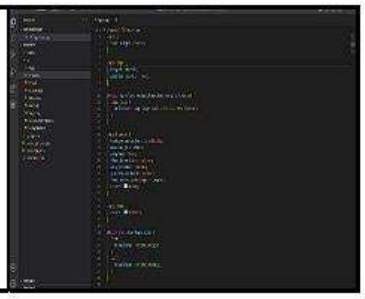
```
const crypto = require('crypto');
function encryptData(data, key) {
    const cipher = crypto.createCipher('aes-256-cbc', key);
    let encryptedData = cipher.update(data, 'utf-8', 'hex');
    encryptedData += cipher.final('hex');
    return encryptedData;
}
function decryptData(encryptedData, key) {
    const decipher = crypto.createDecipher('aes-256-cbc', key);
    let decryptedData = decipher.update(encryptedData, 'hex',
    'utf-8');
    decryptedData += decipher.final('utf-8');
    return decryptedData;
}
```

Explanation:

This code snippet demonstrates data encryption and decryption using the AES encryption algorithm. encryptData function takes data and a key, encrypts it, and returns the encrypted data in hexadecimal format. decryptData function reverses the process, decrypting the data using the same key.

6.RESULTS

6.1 Performance Metrics:

S.No.	Parameter	Values	Screenshot
1.	Information gathering	Setup all the Prerequisite:	
2.	Extract the zip files	Open to vs code	

3.	Remix Ide platform explorting	Deploy the smart contract code Deploy and run the transaction. By selecting the environment - inject the MetaMask.	
4	Open file explorer	Open the extracted file and click on the folder. Open src, and search for utiles. Open cmd enter commands 1.npm install 2.npm bootstrap 3. npm start	
5	{LOCALHOST IP ADDRESS	copy the address and open it to chrome so you can see the front end of your project.	

7. ADVANTAGES & DISADVANTAGES

Advantages:

Precision and Accuracy: Advanced technology can provide precise measurements and accurate data about the nutritional content of foods. This is especially important for individuals with specific dietary needs, such as those managing allergies, diabetes, or other health conditions.

Real-time Monitoring: With the integration of sensors and IoT devices, a black technology-based food tracking system can offer real-time monitoring. This enables users to make immediate adjustments to their diet based on their nutritional goals.

Customization and Personalization: The system can be designed to cater to individual preferences and requirements. It can suggest personalized meal plans, recipes, and dietary recommendations based on a user's specific health goals and constraints.

AI-driven Insights: Incorporating artificial intelligence can provide users with intelligent insights based on their dietary habits. It can offer recommendations for healthier alternatives or adjustments to meet specific nutritional targets.

Automated Data Entry: Advanced technology can automate the process of logging food intake. This reduces the burden on users to manually enter every detail, making the tracking process more seamless and user-friendly.

Integration with Wearable Devices: The system can integrate with wearable technology like smartwatches or fitness trackers to gather additional health-related data. This holistic approach provides a more comprehensive view of a person's overall health.

Database of Food Information: A comprehensive database of food information, including nutritional values, portion sizes, and preparation methods, can be compiled. This ensures that users have access to a wide range of food options for accurate tracking.

Allergen and Contaminant Detection: Advanced technology can be used to detect allergens and contaminants in food products, which is crucial for individuals with allergies or sensitivities.

Compliance and Regulation: A black technology-based food tracking system can help individuals and businesses comply with regulatory requirements related to food safety, labeling, and nutritional information.

Research and Analytics: The collected data can be used for research purposes, enabling scientists and nutritionists to gain insights into dietary trends, preferences, and their impact on health.

Reduced Food Waste: By monitoring food consumption patterns, the system can help users minimize food waste by providing recommendations on portion sizes and suggesting ways to use leftover ingredients.

Improved Health Outcomes: Ultimately, a sophisticated food tracking system can contribute to better health outcomes for individuals by helping them make informed, data-driven decisions about their diet.

Disadvantages:

Cost: Advanced technology often comes with a higher price tag. This could make the system more expensive for both developers and end-users, potentially limiting its accessibility.

Complexity: Highly advanced technology may require specialized knowledge to develop, implement, and maintain. This can pose a barrier for smaller businesses or organizations **without the necessary expertise or resources.**

Dependency on Infrastructure: Cutting-edge technology often relies on a stable and reliable technology infrastructure. If there are disruptions or failures in this infrastructure, it could lead to system downtime or data loss.

Privacy and Security Concerns: More advanced technology may collect and process more sensitive data. This can raise concerns about privacy and security, especially if the system is not properly secured.

Compatibility Issues: Cutting-edge technology might not always be compatible with existing systems or hardware. This could require additional investments in upgrading or replacing current infrastructure.

Limited Adoption: New and advanced technologies may take time to gain widespread adoption. This could result in a smaller user base initially, which might limit the effectiveness or impact of the system.

Potential for Technical Failures: Advanced technology can sometimes be more prone to technical issues or bugs, especially if it's still in the early stages of development or implementation.

Training and Familiarity: Users and operators may require additional training to effectively use and maintain a system based on advanced technology. This can add to the overall cost and time investment.

Regulatory and Compliance Challenges: Depending on the region and industry, there may be specific regulations and compliance requirements that must be met when using advanced technology in food tracking systems.

8. CONCLUSION

- **Precision and Accuracy:** Advanced technologies such as artificial intelligence, machine learning, and sensor-based tracking can provide highly accurate data about food intake, including nutritional content.

- **Personalized Nutrition:** These systems can offer tailored dietary recommendations based on individual needs, taking into account factors like age, activity level, and health conditions.

- **Health and Wellness Management:** The system can serve as a powerful tool for managing health conditions like diabetes, obesity, and allergies by providing real-time feedback and insights.

- **Environmental Impact:** By tracking food sources and supply chains, the system can contribute to sustainability efforts, helping consumers make more environmentally-friendly food choices.

- **Data-Driven Insights:** The data collected can be used for research and policy-making, providing valuable insights into eating habits and trends on a large scale.

9. FUTURE SCOPE

- **Integration with Wearable Devices:** Future iterations could seamlessly integrate with wearables, providing continuous monitoring and feedback on dietary habits.

- **AI-Powered Meal Planning:** AI algorithms could generate personalized meal plans based on dietary preferences, nutritional needs, and health goals.

- **Blockchain and Supply Chain Transparency:** Utilizing blockchain technology can enhance traceability, ensuring consumers have access to accurate information about the origin of their food.

- **Augmented Reality (AR) for Food Recognition:** AR technology could enable real-time food recognition, making it easier for users to track their meals.

- **IoT Integration for Smart Kitchen Appliances:** Connected kitchen appliances could automatically log food items and their nutritional information, streamlining the tracking process.

- **Virtual Nutritionist and Cooking Coach:** Advanced AI could offer real-time advice on meal preparation and suggest healthier alternatives.

- **Global Health Initiatives:** Governments and health organizations could leverage this technology to promote healthier eating habits and combat diet-related health issues on a larger scale.

- **AI-Powered Dietary Research and Development:** Advanced analytics could be used to optimize food production and development, creating more nutritious and sustainable options.

10. APPENDIX

Define Objectives and Features: Clearly define the objectives of your food tracking system. What do you want it to achieve? For example, is it for dietary monitoring, allergen tracking, or something else? List the features you want to include, such as barcode scanning, image recognition, nutritional analysis, meal planning, etc.

Research Advanced Technologies: Investigate any available "black technologies" or cutting-edge solutions that can be integrated into your system. This might include advanced sensors, AI algorithms, or other proprietary technologies.

Data Collection: Determine how you'll collect data. This could involve sensors, image recognition, or other methods to identify and quantify food items.

Data Processing and Analysis: Use advanced algorithms and AI techniques to process and analyze the data. This might involve machine learning models to recognize food items, estimate portion sizes, and calculate nutritional content.

User Interface: Design an intuitive and user-friendly interface for your system. This could be a mobile app, web application, or even a specialized device with a screen.

Integration with Existing Systems: If applicable, ensure that your system can integrate with existing platforms or databases for enhanced functionality.

Security and Privacy: Implement robust security measures to protect user data and privacy, especially if personal health information is involved.

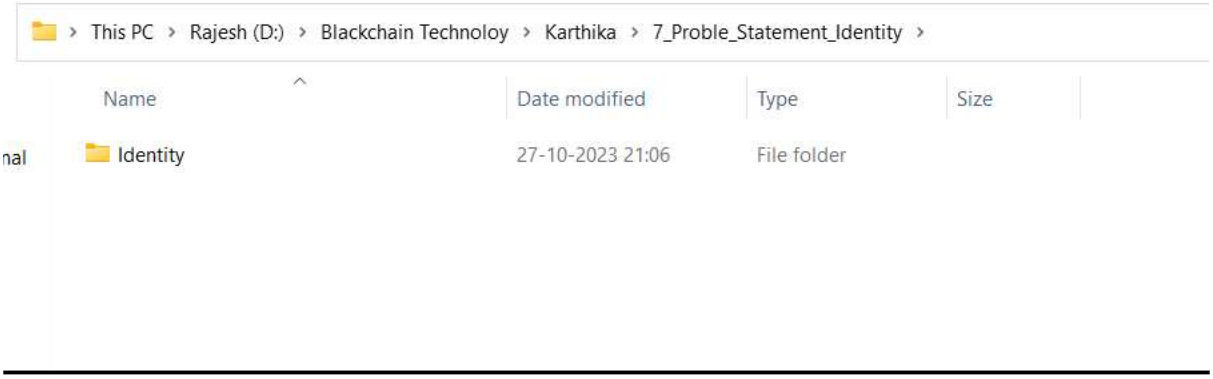
Testing and Validation: Rigorously test the system to ensure accuracy, reliability, and usability. This may involve both controlled lab tests and real-world trials.

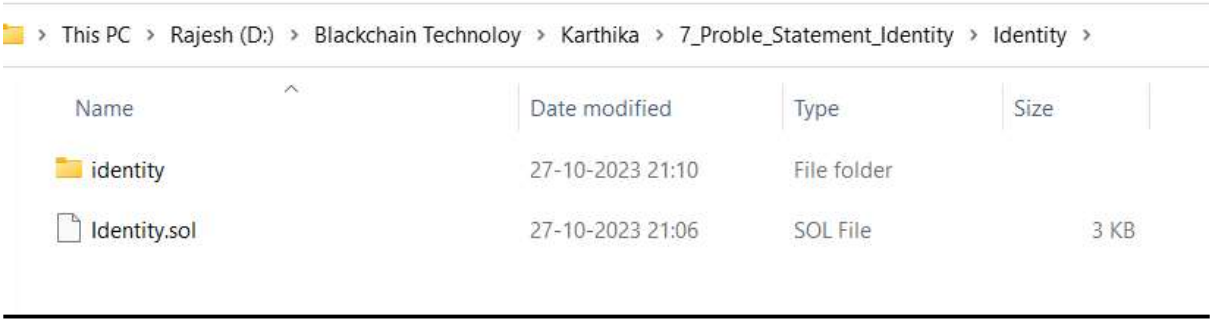
Regulatory Compliance: Depending on your location and the nature of the system, you may need to comply with various regulations related to health data, privacy, and medical devices.

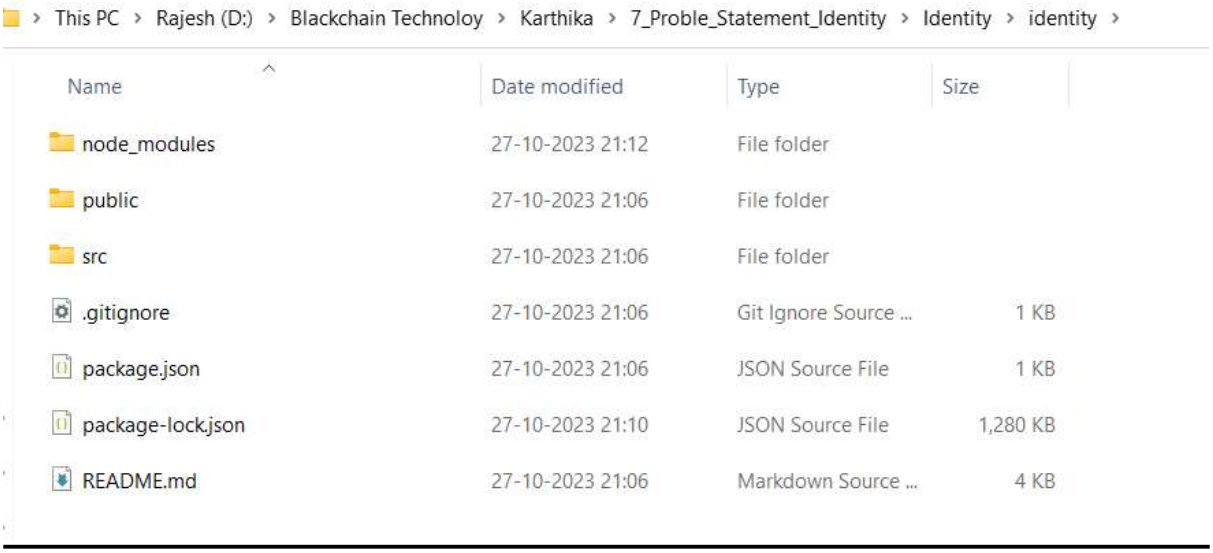
Deployment and Maintenance: Deploy your system and provide ongoing maintenance and support. This could include regular updates, bug fixes, and user support.

11.SOURCE CODE

Folder Structure:








This PC › Rajesh (D:) › Blackchain Technoloy › Karthika › 7_Proble_Statement_Identity › Identity › identity › src ›
Name
Date modified
Type
Size
Page
27-10-2023 21:06
File folder
App.css
27-10-2023 21:06
CSS Source File
1 KB
App.js
27-10-2023 21:06
JavaScript Source ...
1 KB
App.test.js
27-10-2023 21:06
JavaScript Source ...
1 KB
index.css
27-10-2023 21:06
CSS Source File
1 KB
index.js
27-10-2023 21:06
JavaScript Source ...
1 KB
logo.svg
27-10-2023 21:06
Microsoft Edge HT...
3 KB
reportWebVitals.js
27-10-2023 21:06
JavaScript Source ...
1 KB
setupTests.js
27-10-2023 21:06
JavaScript Source ...
1 KB

INDEX.HTML

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
    <!--
      manifest.json provides metadata used when your web app is installed on a
      user's mobile device or desktop. See
https://developers.google.com/web/fundamentals/web-app-manifest/
    -->
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <!--
      Notice the use of %PUBLIC_URL% in the tags above.
      It will be replaced with the URL of the `public` folder during the
build.
      Only files inside the `public` folder can be referenced from the HTML.

      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
      work correctly both with client-side routing and a non-root public URL.
      Learn how to configure a non-root public URL by running `npm run build`.
    -->
    <title>React App</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
    <!--
      This HTML file is a template.
      If you open it directly in the browser, you will see an empty page.

      You can add webfonts, meta tags, or analytics to this file.
      The build step will place the bundled scripts into the <body> tag.

      To begin the development, run `npm start` or `yarn start`.
      To create a production bundle, use `npm run build` or `yarn build`.
    -->
  </body>
</html>
```

Manifest.json

```
{
  "short_name": "React App",
  "name": "Create React App Sample",
  "icons": [
    {
      "src": "favicon.ico",
      "sizes": "64x64 32x32 24x24 16x16",
      "type": "image/x-icon"
    },
    {
      "src": "logo192.png",
      "type": "image/png",
      "sizes": "192x192"
    },
    {
      "src": "logo512.png",
      "type": "image/png",
      "sizes": "512x512"
    }
  ],
  "start_url": ".",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#ffffff"
}
```

App.css

```css
.App {
  text-align: center;
}

.App-logo {
  height: 40vmin;
  pointer-events: none;
}

@media (prefers-reduced-motion: no-preference) {
  .App-logo {
    animation: App-logo-spin infinite 20s linear;
  }
}

.App-header {
  background-color: #282c34;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}

.App-link {
  color: #61dafb;
}

@keyframes App-logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}
```

App.js

```
import './App.css';
import Home from './Page/Home'

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <Home />
      </header>
    </div>
  );
}

export default App;
```

Home.js

```
import React, { useState } from "react";
import { Button, Container, Row, Col } from 'react-bootstrap';
import '../../node_modules/bootstrap/dist/css/bootstrap.min.css';
import { contract } from "./connector";

function Home() {
 const [Id, setId] = useState("");
 const [name, setName] = useState("");
 const [email, setEmail] = useState("");
 const [addr, setAddr] = useState("");
 const [mAddr, setmAddr] = useState("");
 const [data, setdata] = useState("");
   const [Wallet, setWallet] = useState("");



 const handleId = (e) => {
  setId(e.target.value)
 }

 const handleName = (e) => {
  setName(e.target.value)
 }
```

```
const handleEmail = (e) => {
 setEmail(e.target.value)
}

const handleAddr = (e) => {
 setAddr(e.target.value)
}

const handleRegisterIdentity = async () => {
 try {
  let tx = await contract.registerIdentity(Id.toString(), name,  email, addr)

 let wait =  await tx.wait()
 alert(wait.transactionHash)
 console.log(wait);
 } catch (error) {
  alert(error)
 }
}

const handleMetamaskAddr = (e) => {
 setmAddr(e.target.value)
}


const handleAddrDetails = async () => {
 try {
  let tx = await contract.getIdentityDetails(mAddr)
  let arr = []
  tx.map(e => arr.push(e))
  setdata(arr)
  // alert(tx)
  console.log(tx);
 } catch (error) {
  alert(error)
 }
}

  const handleWallet = async () => {
     if (!window.ethereum) {
        return alert('please install metamask');
     }

     const addr = await window.ethereum.request({
        method: 'eth_requestAccounts',
     });

     setWallet(addr[0])
```

```
   }

 return (
  <div>
   <h1 style={{ marginTop: "30px", marginBottom: "80px" }}>Identity On
Blockchain</h1>
       {!Wallet ?

          <Button onClick={handleWallet} style={{ marginTop: "30px",
marginBottom: "50px" }}>Connect Wallet </Button>
          :
          <p style={{ width: "250px", height: "50px", margin: "auto",
marginBottom: "50px", border: '2px solid #2096f3' }}>{Wallet.slice(0,
6)}....{Wallet.slice(-6)}</p>
       }
   <Container>
    <Row>
     <Col style={{marginRight:"100px"}}>
      <div>


       <input style={{ marginTop: "10px", borderRadius: "5px" }}
onChange={handleId} type="number" placeholder="Enter Identity ID" value={Id}
/> <br />

       <input style={{ marginTop: "10px", borderRadius: "5px" }}
onChange={handleName} type="string" placeholder="Enter Name" value={name} />
<br />

       <input style={{ marginTop: "10px", borderRadius: "5px" }}
onChange={handleEmail} type="string" placeholder="Enter Email" value={email}
/><br />

       <input style={{ marginTop: "10px", borderRadius: "5px" }}
onChange={handleAddr} type="string" placeholder="Enter Address" value={addr}
/><br />

       <Button onClick={handleRegisterIdentity} style={{ marginTop: "10px" }}
variant="primary">Register Identity</Button>
      </div>
     </Col>
     <Col>
      <div>
       <input style={{ marginTop: "10px", borderRadius: "5px" }}
onChange={handleMetamaskAddr} type="string" placeholder="User Metamask
address" value={mAddr} /><br />
```

```
      <Button onClick={handleAddrDetails} style={{ marginTop: "10px" }}
variant="primary">Get Identity Details</Button>
      {data ? data?.map(e => {
         return <p>{e.toString()}</p>
      }) : <p></p>}
     </div>
    </Col>
   </Row>

  </Container>

 </div>
 )
}

export default Home;
```

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;
}
```

Index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);

// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

Reportwebvitals.js

```
const reportWebVitals = onPerfEntry => {
  if (onPerfEntry && onPerfEntry instanceof Function) {
    import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) =>
{
      getCLS(onPerfEntry);
      getFID(onPerfEntry);
      getFCP(onPerfEntry);
      getLCP(onPerfEntry);
      getTTFB(onPerfEntry);
    });
  }
};
```

```
export default reportWebVitals;
```

Setuptest.js

```
// jest-dom adds custom jest matchers for asserting on DOM nodes.
// allows you to do things like:
// expect(element).toHaveTextContent(/react/i)
// learn more: https://github.com/testing-library/jest-dom
import '@testing-library/jest-dom';
```

Package.json

```
{
  "name": "identity",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.17.0",
    "@testing-library/react": "^13.4.0",
    "@testing-library/user-event": "^13.5.0",
    "bootstrap": "^5.3.1",
    "ethers": "^5.6.6",
    "react": "^18.2.0",
    "react-bootstrap": "^2.8.0",
    "react-dom": "^18.2.0",
    "react-scripts": "5.0.1",
    "web-vitals": "^2.1.4"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
```

```
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```

Connector.js

```
const { ethers } = require("ethers");

const abi = [
 {
  "inputs": [],
  "stateMutability": "nonpayable",
  "type": "constructor"
 },
 {
  "anonymous": false,
  "inputs": [
   {
    "indexed": true,
    "internalType": "address",
    "name": "owner",
    "type": "address"
   },
   {
    "indexed": false,
    "internalType": "string",
    "name": "identityId",
    "type": "string"
   },
   {
    "indexed": false,
    "internalType": "string",
    "name": "name",
    "type": "string"
```

```
  },
  {
   "indexed": false,
   "internalType": "string",
   "name": "email",
   "type": "string"
  },
  {
   "indexed": false,
   "internalType": "uint256",
   "name": "registrationTimestamp",
   "type": "uint256"
  }
 ],
 "name": "IdentityRegistered",
 "type": "event"
},
{
 "inputs": [
  {
   "internalType": "address",
   "name": "userAddress",
   "type": "address"
  }
 ],
 "name": "getIdentityDetails",
 "outputs": [
  {
   "internalType": "string",
   "name": "",
   "type": "string"
  },
  {
   "internalType": "string",
   "name": "",
   "type": "string"
  },
  {
   "internalType": "string",
   "name": "",
   "type": "string"
  },
  {
   "internalType": "string",
   "name": "",
   "type": "string"
  },
  {
```

```
    "internalType": "uint256",
    "name": "",
    "type": "uint256"
   }
  ],
  "stateMutability": "view",
  "type": "function"
 },
 {
  "inputs": [
   {
    "internalType": "address",
    "name": "",
    "type": "address"
   }
  ],
  "name": "identities",
  "outputs": [
   {
    "internalType": "string",
    "name": "identityId",
    "type": "string"
   },
   {
    "internalType": "string",
    "name": "name",
    "type": "string"
   },
   {
    "internalType": "string",
    "name": "email",
    "type": "string"
   },
   {
    "internalType": "string",
    "name": "contactAddress",
    "type": "string"
   },
   {
    "internalType": "uint256",
    "name": "registrationTimestamp",
    "type": "uint256"
   }
  ],
  "stateMutability": "view",
  "type": "function"
 },
 {
```

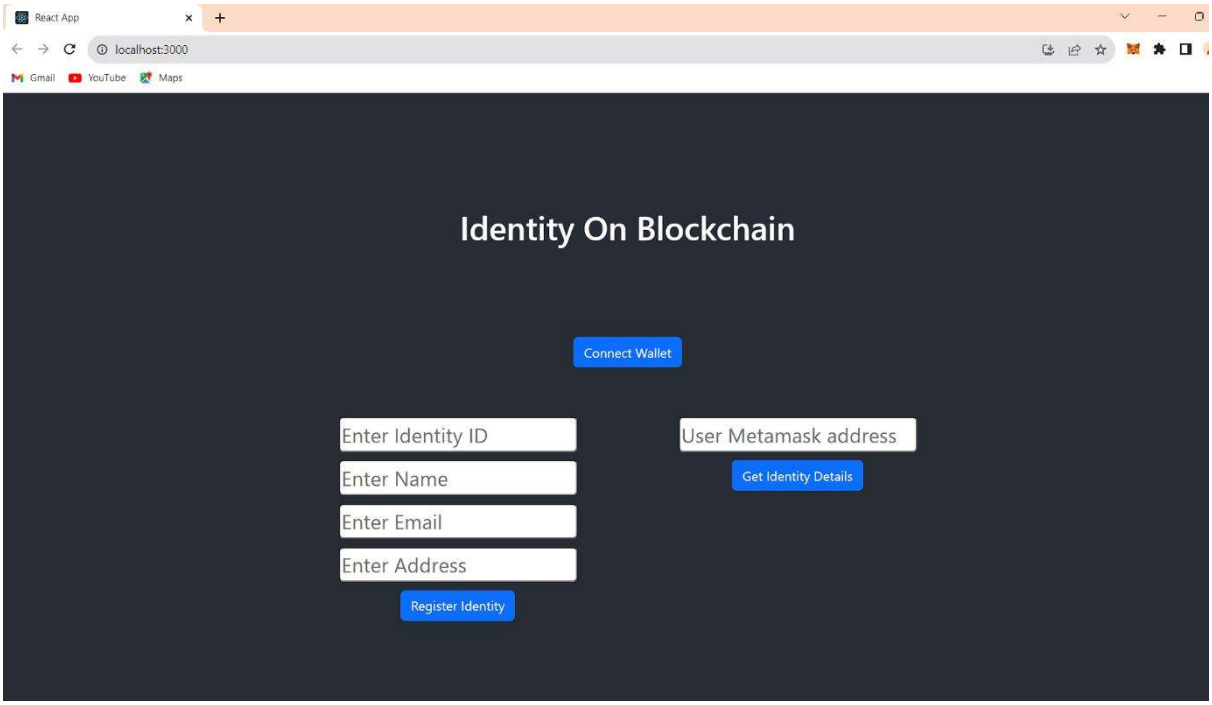
```
  "inputs": [],
  "name": "owner",
  "outputs": [
   {
    "internalType": "address",
    "name": "",
    "type": "address"
   }
  ],
  "stateMutability": "view",
  "type": "function"
 },
 {
  "inputs": [
   {
    "internalType": "string",
    "name": "identityId",
    "type": "string"
   },
   {
    "internalType": "string",
    "name": "name",
    "type": "string"
   },
   {
    "internalType": "string",
    "name": "email",
    "type": "string"
   },
   {
    "internalType": "string",
    "name": "_address",
    "type": "string"
   }
  ],
  "name": "registerIdentity",
  "outputs": [],
  "stateMutability": "nonpayable",
  "type": "function"
 }
]

if (!window.ethereum) {
 alert('Meta Mask Not Found')
 window.open("https://metamask.io/download/")
}

export const provider = new ethers.providers.Web3Provider(window.ethereum);
```

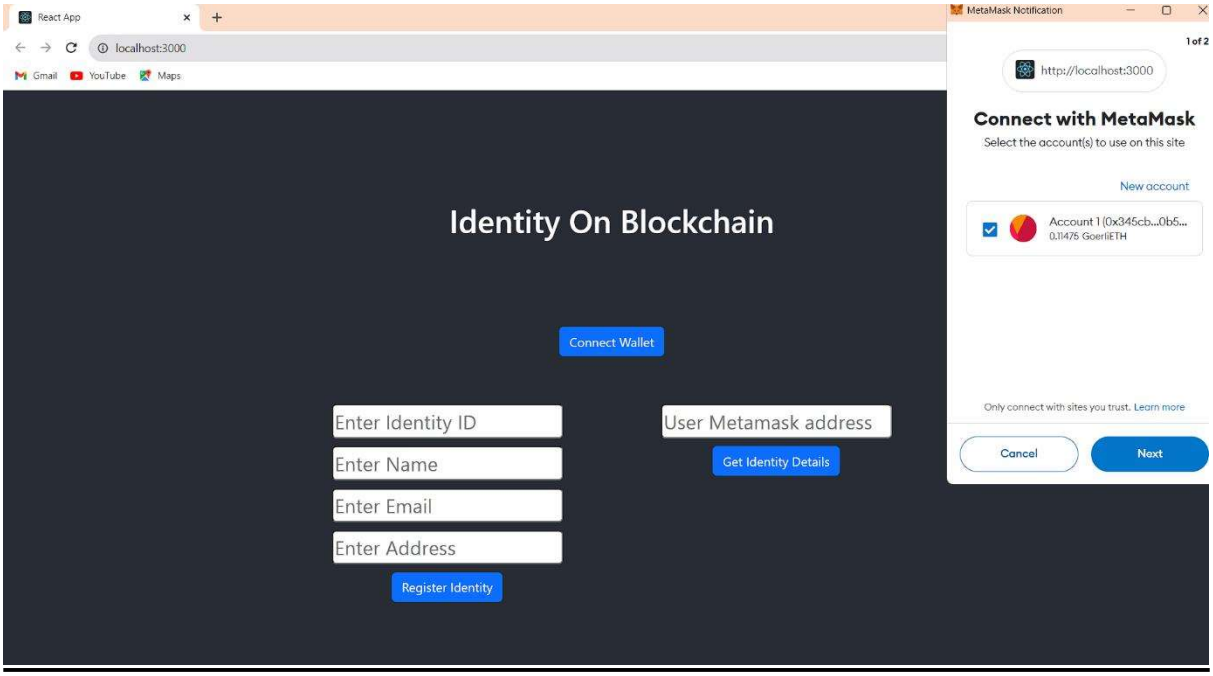
```
export const signer = provider.getSigner();
export const address = "0xd9145CCE52D386f254917e481eB44e9943F39138"

export const contract = new ethers.Contract(address, abi, signer)
```

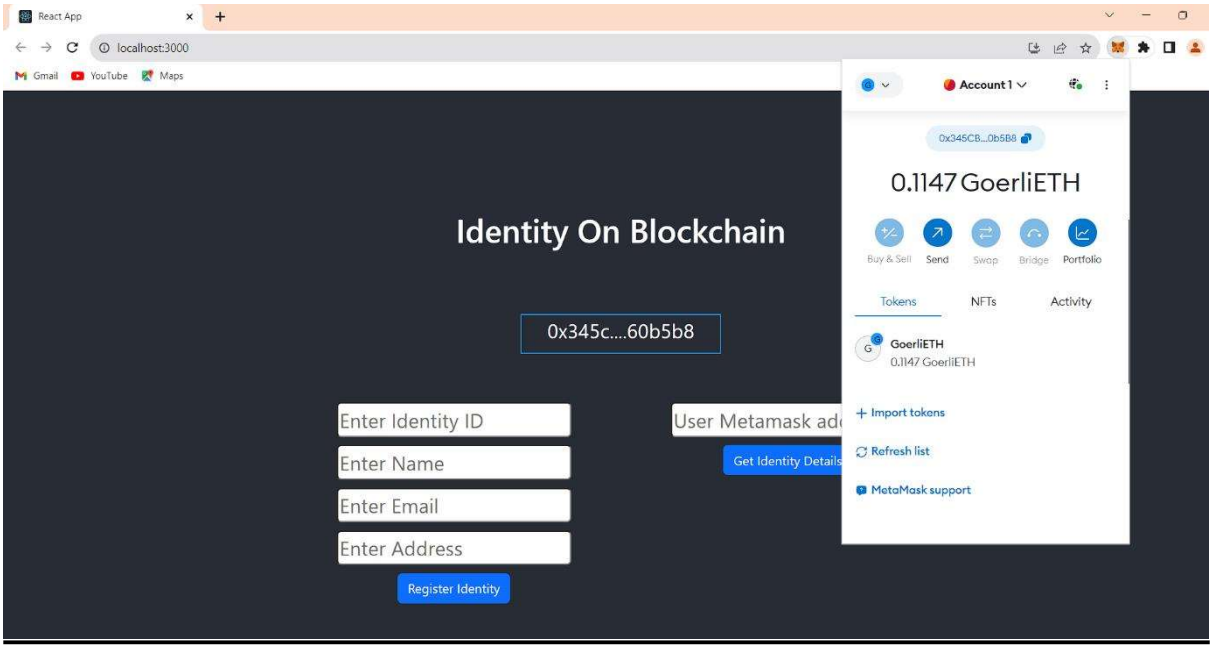
GitHub Link: https://github.com/Karthika46pk/Food-Tracking-system

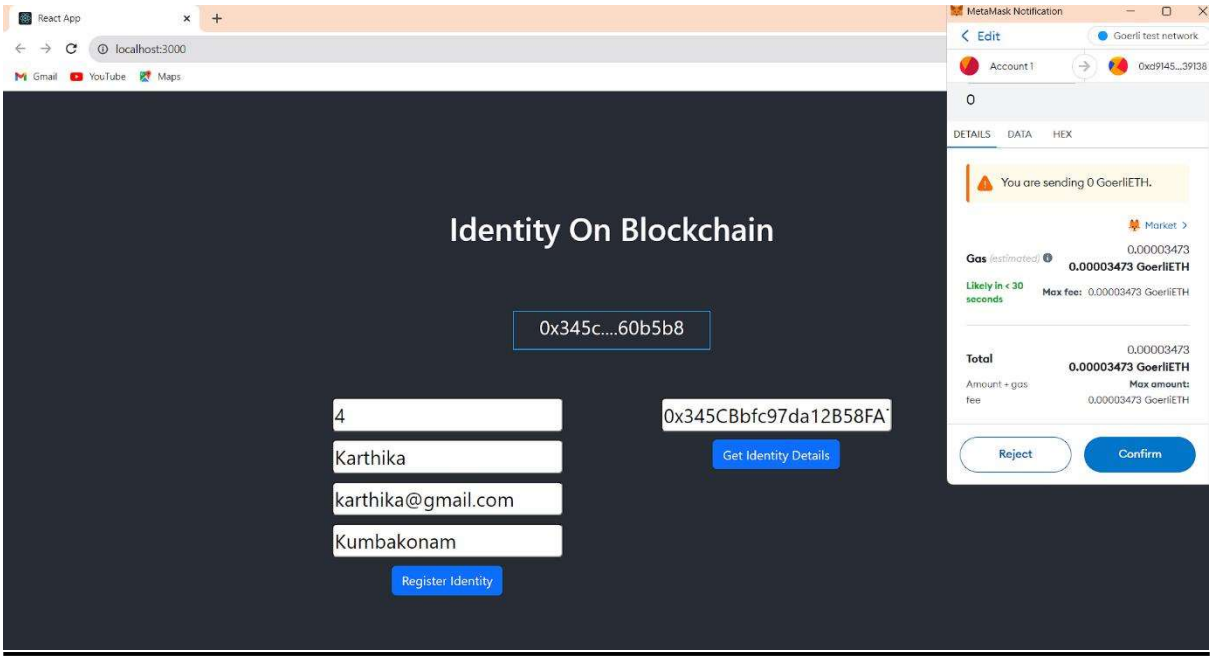
Demo Link: https://drive.google.com/file/d/1JJK_l66cFEEKHqo-4XtzkY5H0dQdQVMK/view?usp=drive_link

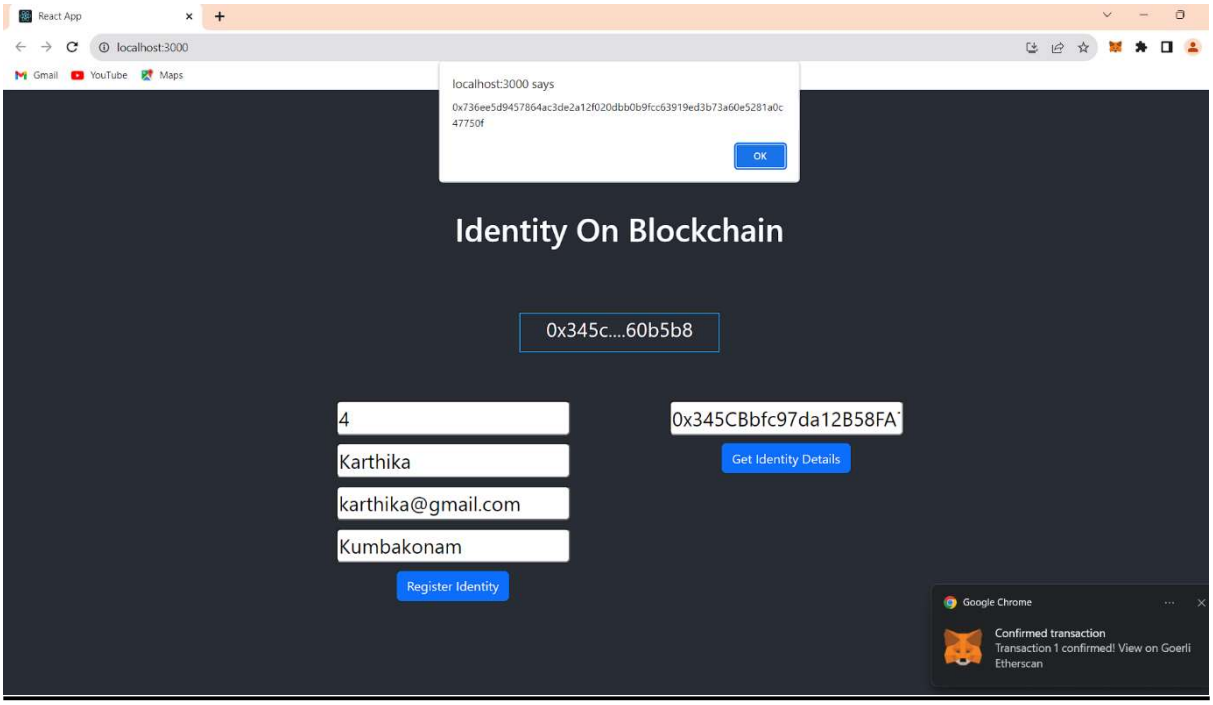
11.OUTPUT SCREENSHOTS




React App
localhost:3000
Gmail
YouTube
Maps
Identity On Blockchain
Connect Wallet
Enter Identity ID
Enter Name
Enter Email
Enter Address
Register Identity
User Metamask address
Get Identity Details
MetaMask Notification
1 of 2
http://localhost:3000
Connect with MetaMask
Select the account(s) to use on this site
New account
Account 1 (0x345cb...0b5...
0.11475 GoerliETH
Only connect with sites you trust. Learn more
Cancel
Next


React App
localhost:3000
Gmail
YouTube
Maps
Identity On Blockchain
0x345c....60b5b8
Enter Identity ID
Enter Name
Enter Email
Enter Address
Register Identity
User Metamask ad
Get Identity Details
Account 1
0x345CB...0b5B8
0.1147 GoerliETH
Buy & Sell
Send
Swap
Bridge
Portfolio
Tokens
NFTs
Activity
GoerliETH
0.1147 GoerliETH
Import tokens
Refresh list
MetaMask support


React App
localhost:3000
Gmail
YouTube
Maps
Identity On Blockchain
0x345c....60b5b8
4
Karthika
karthika@gmail.com
Kumbakonam
Register Identity
0x345CBbfc97da12B58FA
Get Identity Details
MetaMask Notification
Edit
Goerli test network
Account 1
0xd9145...39138
0
DETAILS
DATA
HEX
You are sending 0 GoerliETH.
Market
Gas (estimated)
0.00003473
0.00003473 GoerliETH
Likely in < 30 seconds
Max fee: 0.00003473 GoerliETH
Total
0.00003473
0.00003473 GoerliETH
Amount + gas fee
Max amount:
0.00003473 GoerliETH
Reject
Confirm


React App
localhost:3000
Gmail
YouTube
Maps
localhost:3000 says
0x736ee5d9457864ac3de2a12f020dbb0b9fcc63919ed3b73a60e5281a0c47750f
OK
Identity On Blockchain
0x345c....60b5b8
4
Karthika
karthika@gmail.com
Kumbakonam
Register Identity
0x345CBbfc97da12B58FA
Get Identity Details
Google Chrome
Confirmed transaction
Transaction 1 confirmed! View on Goerli Etherscan