INDEX

CLIMATE TRACKSMART USING BLOCKCHAIN

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Climate TrackSmart is an innovative endeavor harnessing blockchain technology to revolutionize climate data management and environmental accountability. This transformative project will establish a transparent and immutable ledger for climate-related data, bolstering transparency and security. It will utilize a decentralized network for independent data verification, facilitating real-time monitoring of greenhouse gas emissions and carbon impact assessments. Climate TrackSmart will also automate climate-related agreements through smart contracts, simplify carbon credit creation and trading, and enhance the traceability of carbon footprints in supply chains. This initiative aims to encourage international collaboration, incentivize green practices, and empower stakeholders from governments to businesses and the public to engage in proactive climate actions. Privacy and security are paramount, with an emphasis on data privacy and robust encryption, while ensuring compatibility with existing climate data infrastructure. By facilitating environmental accountability and fostering global cooperation, Climate TrackSmart aspires to contribute significantly to a more sustainable world.

1.2 PURPOSE

Climate TrackSmart is a blockchain-based platform that aims to track and reduce greenhouse gas emissions. It does this by providing a transparent and immutable record of emissions data, which can be used by businesses and individuals to identify and track their emissions, and to offset them through carbon credits.All emissions data on the Climate TrackSmart platform is publicly accessible, which helps to ensure transparency and accountability.Once emissions data is recorded on the blockchain, it cannot be tampered with, which helps to prevent fraud and ensure the integrity of the data.Blockchain technology is highly secure, which helps to protect emissions data from unauthorized access or tampering.Climate TrackSmart can be used to track and offset carbon emissions through carbon credits. This can help businesses and individuals to reduce their environmental impact.Climate TrackSmart can be used to track emissions throughout a supply chain, which can help businesses to identify and reduce their emissions footprint.Climate TrackSmart can be used to comply with environmental regulations, such as the EU Emissions Trading System.Climate TrackSmart is still in its early stages of development, but it has the potential to play a significant role in the fight against climate change. By providing a transparent, immutable, and secure way to track and reduce greenhouse gas emissions, Climate TrackSmart can help businesses and individuals to take meaningful action on climate change.Climate TrackSmart uses blockchain technology to provide a number of benefits, including transparency, immutability, and security.

2. EXISTING SYSTEM

2.1 EXISTING PROBLEM

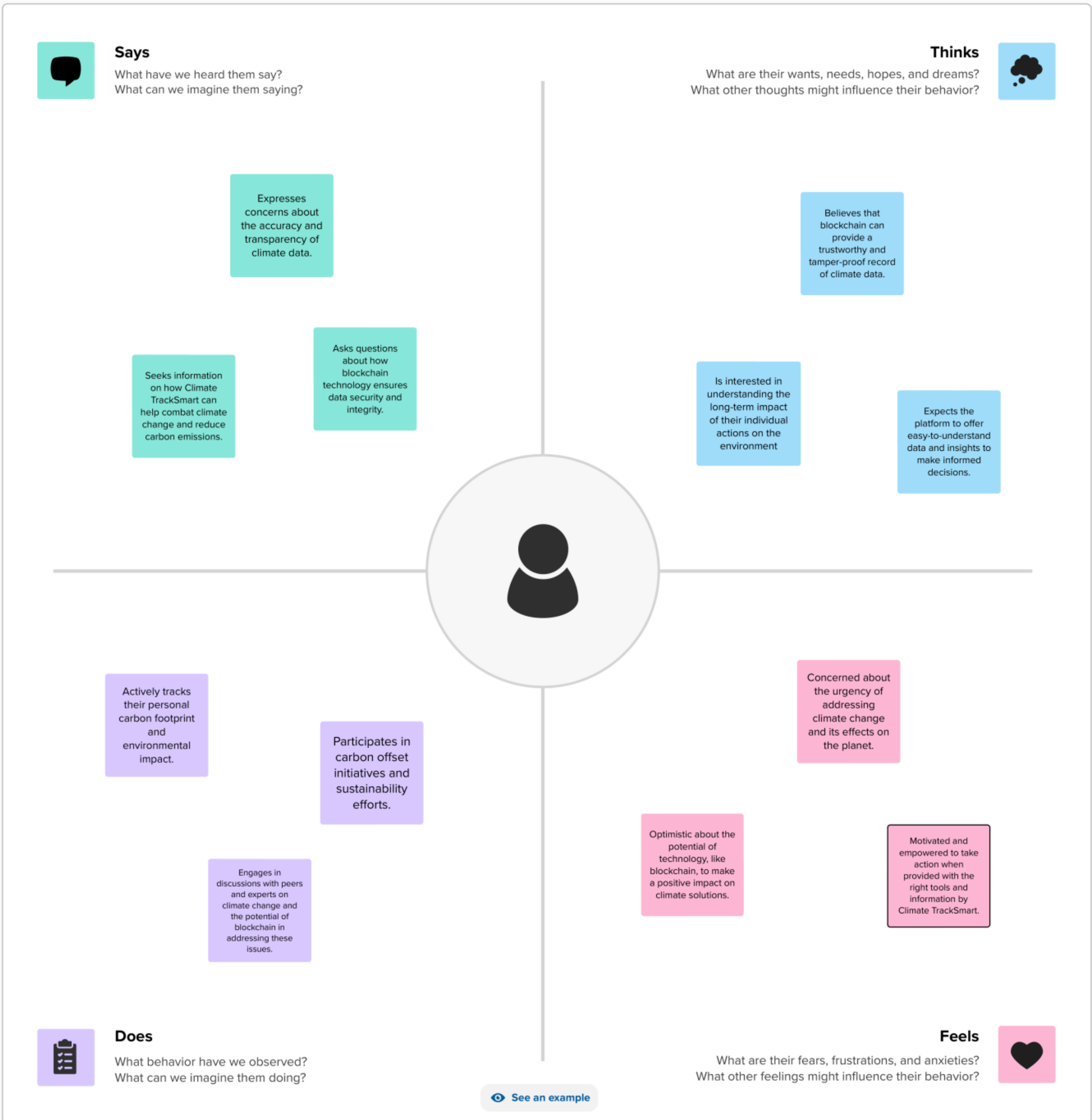
One of the pressing existing problems is the lack of transparent, secure, and efficient mechanisms for monitoring and managing climate-related data and environmental actions. Traditional systems often suffer from issues like data manipulation, centralized control, and inadequate accountability. This leads to challenges in verifying the accuracy of climate data, promoting sustainable practices, and fostering international collaboration to combat climate change effectively. Additionally, the existing methods for tracking carbon footprints and incentivizing eco-friendly initiatives can be fragmented and lack standardized protocols. These problems hinder our ability to address the urgent global climate crisis comprehensively.Climate Change: The rise in greenhouse gas emissions from human activities, primarily burning fossil fuels, is causing global temperatures to increase, leading to more frequent and severe weather events.Poor air quality due to emissions from transportation, industrial processes, and deforestation adversely affects human health and ecosystems.The loss of forests for agriculture and development contributes to habitat destruction, biodiversity loss, and carbon emissions.Contamination of water bodies from industrial waste, agricultural runoff, and inadequate sewage treatment endangers aquatic ecosystems and human health.

2.2 PROBLEM STATEMENT

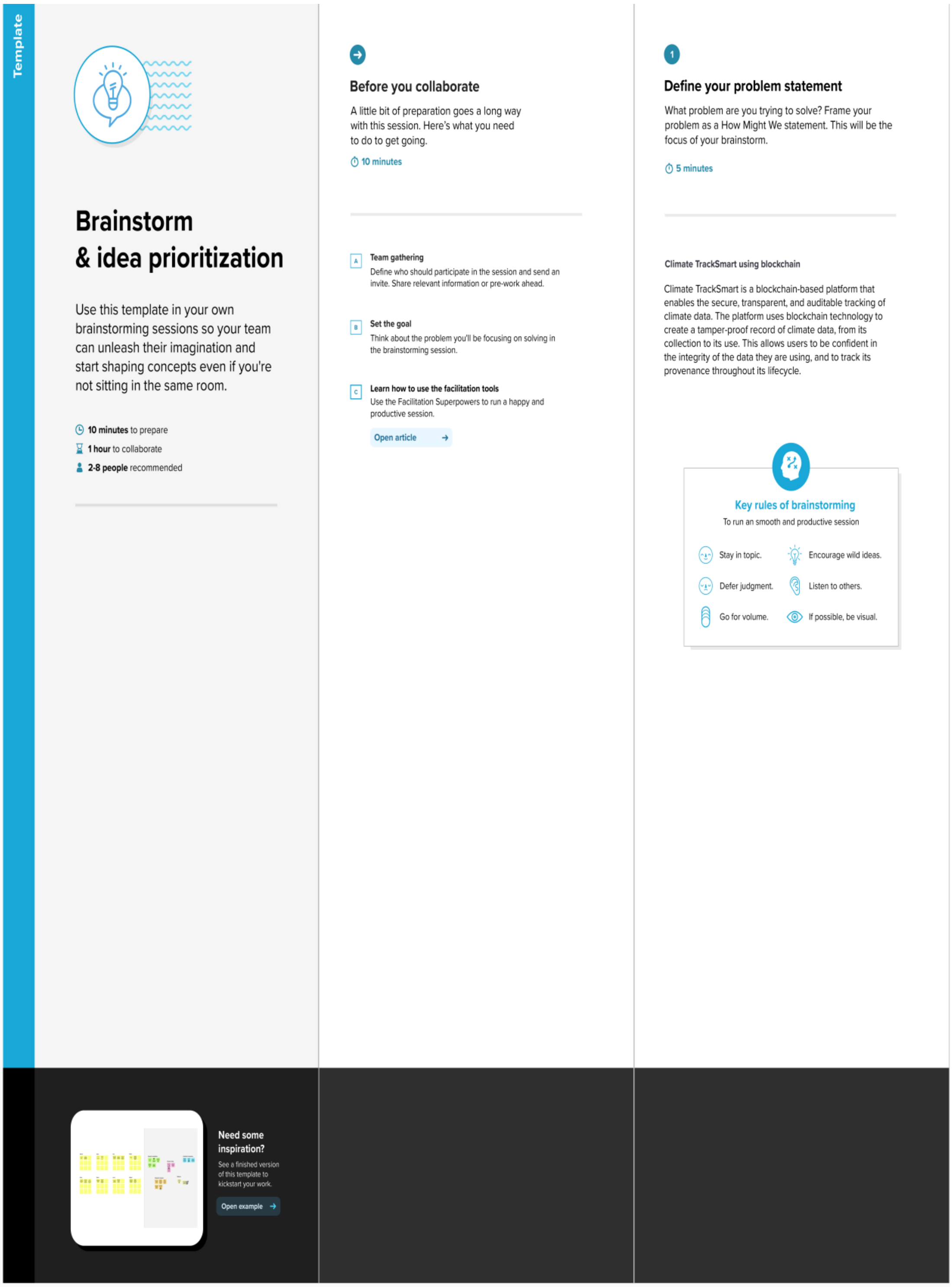
Climate TrackSmart is a blockchain-based platform that aims to track and reduce greenhouse gas emissions. It does this by providing a transparent and immutable record of emissions data, which can be used by businesses and individuals to identify and track their emissions, and to offset them through carbon credits. All emissions data on the Climate TrackSmart platform is publicly accessible, which helps to ensure transparency and accountability.Once emissions data is recorded on the blockchain, it cannot be tampered with, which helps to prevent fraud and ensure the integrity of the data.Blockchain technology is highly secure, which helps to protect emissions data from unauthorized access or tampering.Climate TrackSmart can be used to track and offset carbon emissions through carbon credits. Climate TrackSmart is a blockchain-based platform that aims to track and reduce greenhouse gas emissions. It does this by providing a transparent and immutable record of emissions data, which can be used by businesses and individuals to identify and track their emissions, and to offset them through carbon credits.Climate TrackSmart uses blockchain technology to provide a number of benefits, including transparency, immutability, and security. It has a number of potential applications, including carbon offsetting, supply chain emissions tracking, and regulatory compliance.Climate TrackSmart is still in its early stages of development, but it has the potential to play a significant role in the fight against climate change by providing a transparent, immutable, and secure way to track and reduce greenhouse gas emissions.

3. IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION AND BRAINSTORMING



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

SANJAY

- Develop a blockchain system for tracking and trading carbon credits, making it easy for organizations to offset their emissions
- Create smart contracts that encourage eco-friendly practices, such as rewarding users for reducing energy consumption
- Issue NFTs representing contributions to climate initiatives and conservation projects to raise funds and awareness.

VIMAL PANDIYAN

- Implement a blockchain-based platform for monitoring and verifying the sustainability of products along the supply chain.
- Build a blockchain marketplace for renewable energy trading, allowing individuals to buy and sell excess green energy securely.
- Issue blockchain tokens tied to afforestation and reforestation projects, enabling traceability and transparency.

RAVISHANKAR

- Develop a blockchain oracle that provides real-time climate data for various applications, such as insurance or urban planning.
- Use blockchain to enable seamless and secure electric vehicle charging and billing.
- Create a blockchain platform for green and impact investments, connecting investors with sustainable projects.

JEYASANTHOSH

- Implement a blockchain voting system for climate actions, allowing communities to decide on sustainability projects collectively.
- Implement a blockchain voting system for climate actions, allowing communities to decide on sustainability projects collectively.
- Develop a blockchain-based identity verification system to ensure transparency and accountability in climate initiatives.

SHEIK AMJATH BASHA

- Design a blockchain wallet that rewards users for reducing their carbon footprint through daily actions.
- Implement smart contracts for farmers, linking weather data and insurance payouts in a transparent manner.
- Build a blockchain system for governments to track and manage national emission reduction goals.

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

SECURITY

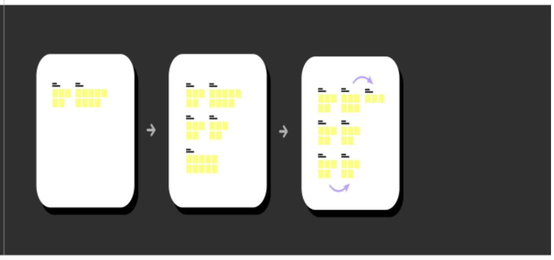
- Blockchain's immutability ensures that climate data is securely stored and tamper-proof, preventing data manipulation
- Implement robust security measures in smart contracts to prevent vulnerabilities and potential breaches
- Utilize decentralized identity solutions to grant access to climate data and actions securely.
- Employ privacy-preserving techniques, such as zero-knowledge proofs, to protect sensitive climate data.

FEATURES

- Implement features for real-time monitoring of climate data, allowing quick response to environmental changes.
- Facilitate interoperability with other data sources and systems to create a comprehensive climate tracking network.
- Tokenize climate-related assets like carbon credits, renewable energy, and afforestation tokens.
- Develop intuitive user interfaces and mobile applications for easy access to climate data and actions.

CHALLENGES

- Scaling the blockchain to accommodate the vast amount of climate data generated globally is a significant challenge.
- Navigating complex and evolving regulatory frameworks related to climate data and blockchain is challenging
- Relying on external oracles for climate data introduces the risk of inaccurate information
- The energy-intensive nature of some blockchain networks poses environmental concerns.



4

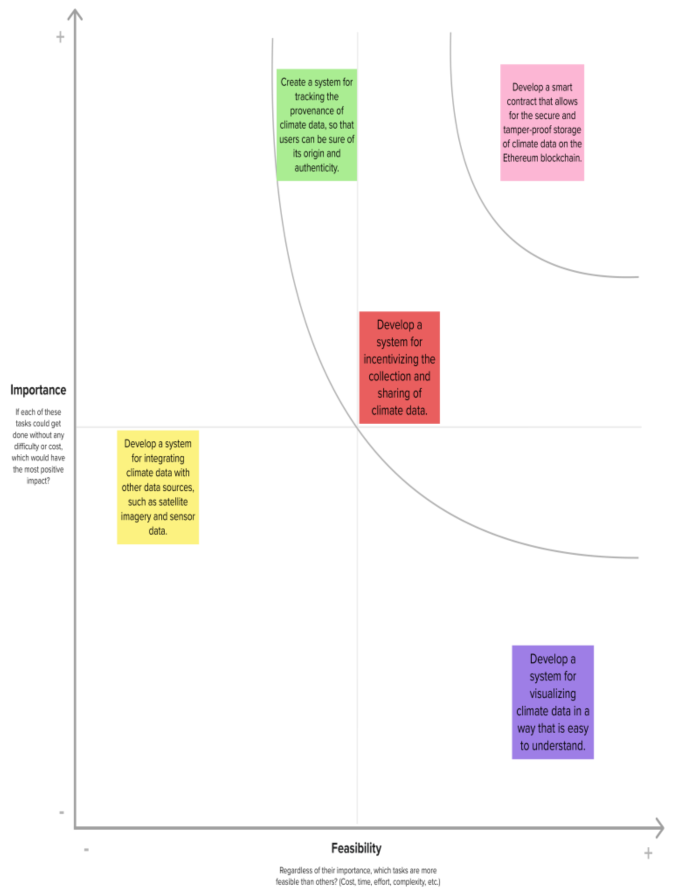
Prioritize

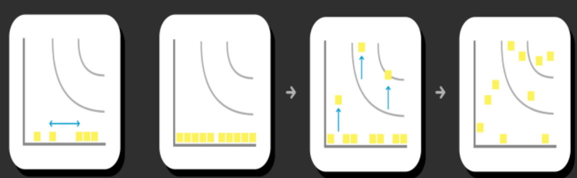
Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.





After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

A **Share the mural**
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

B **Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

Share template feedback

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

User Registration and Authentication:

Users must be able to register and create accounts securely.Multi-factor authentication should be available for enhanced security.

Data Submission and Validation:

Users can submit climate-related data, such as emissions reports or environmental impact assessments.Data submitted must undergo validation and verification processes for accuracy.

Transparent Data Ledger:

Implement a blockchain ledger for storing climate data that is immutable and transparent to all authorized users.Ensure data is time-stamped and tamper-proof to maintain data integrity.

Decentralized Verification:

Utilize a network of nodes for independent verification of submitted data.

Verification should be automated, ensuring trustworthiness and accuracy.

Carbon Credits Management:

Facilitate the creation, issuance, and trading of carbon credits on the blockchain.

Include smart contract functionality to automate credit transactions.

Supply Chain Tracking:

Enable the tracking of carbon footprints in supply chains using blockchain.

Provide a user-friendly interface for organizations to input and monitor supply chain data.

Smart Contracts for Agreements:

Develop and execute smart contracts for climate-related agreements, such as emissions reduction commitments.Ensure automation of contractual terms and obligations.

Data Access and Public Engagement:

Make climate data accessible to a wide range of stakeholders, including the general public, researchers, and policymakers.Implement data visualization tools for user-friendly data interpretation.

International Collaboration:

Support global collaboration by allowing cross-border data sharing and verification. Ensure compliance with international climate agreements and standards.

Real-time Monitoring:

Enable real-time monitoring of greenhouse gas emissions and other climate-related metrics.Ensure alerts and notifications for significant deviations from targets.

Incentivization Mechanism:

Design a system for rewarding green initiatives and sustainable practices.

Tokens or incentives should be issued to entities that meet environmental goals.

Privacy and Security:

Implement robust data encryption and access control measures to protect sensitive climate data.Ensure compliance with data privacy regulations.

Scalability and Performance:

The system should be designed to handle a growing volume of data and users.

Optimize performance to accommodate increased demands.

Interoperability:

Ensure compatibility with existing climate data infrastructure and industry standards.

Facilitate data exchange and integration with external systems.

Auditing and Compliance:

Enable third-party auditors to verify and validate climate data for regulatory compliance.Maintain detailed audit logs for transparency.

Research and Innovation Support:

Encourage research and innovation by providing an API for data access and analysis.

Foster the development of new climate technologies.

User Training and Support:

Provide user training materials and customer support to assist users in navigating the platform.Ensure a user-friendly and intuitive interface.

Reporting and Analytics:

Generate customizable reports and analytics dashboards for users to track their climate performance.Support data-driven decision-making.

Disaster Recovery and Redundancy:

Implement disaster recovery and backup mechanisms to ensure data availability in case of system failures or disasters.Establish data redundancy for fault tolerance.

Continuous Improvement:

Regularly update and enhance the platform to adapt to evolving climate monitoring needs and emerging blockchain technologies.These functional requirements form the foundation for the development of Climate TrackSmart using Blockchain. It's essential to continuously engage with stakeholders to refine and expand these requirements as the project progresses.

4.2 NON FUNCTIONAL REQUIREMENT:

Response Time: The system must provide fast response times for data submission, verification, and access, even under heavy loads.

Scalability: It should be able to scale horizontally and vertically to accommodate growing data and user volumes.

Throughput: Ensure the system can handle a high number of transactions and data points concurrently.

Data Encryption: Use strong encryption algorithms to protect data at rest and during transmission.

Access Control: Implement role-based access control to ensure that only authorized users can interact with sensitive data.

Blockchain Security: Ensure the security of the blockchain network, preventing unauthorized nodes from participating.

Compliance: Adhere to data privacy and security regulations, such as GDPR or HIPAA, as applicable.

High Availability: The system should be available 24/7, with minimal downtime.

Disaster Recovery: Have a robust disaster recovery plan in place to minimize data loss in case of system failures or disasters.

Redundancy: Implement redundancy for critical system components to maintain reliability.

User Interface: Provide an intuitive and user-friendly interface for easy data submission, access, and management.

Accessibility: Ensure the platform is accessible to users with disabilities in compliance with accessibility standards.

Training and Support: Offer user training materials and responsive customer support to assist users effectively.

Compatibility: Ensure that the system can integrate with other existing climate data systems and standards, promoting interoperability.

API Availability: Provide well-documented APIs for external systems to interact with the platform.

Regulatory Compliance: Adhere to relevant environmental and data protection regulations, including those related to carbon trading and climate reporting.

Audit Trails: Maintain comprehensive audit logs to support regulatory compliance and accountability.

Blockchain Consensus: Select a suitable consensus mechanism for the blockchain network, balancing security and performance.

Blockchain Scaling: Implement blockchain scaling solutions, such as sharding, as needed to handle growing data.

Data Retention: Define data retention policies to manage historical climate data effectively.

Data Quality: Ensure data quality and integrity through validation and verification processes.

Backup and Recovery: Regularly back up blockchain data and ensure efficient recovery mechanisms are in place.

Geographic Distribution: Store data redundantly in geographically distributed data centers to mitigate regional risks.

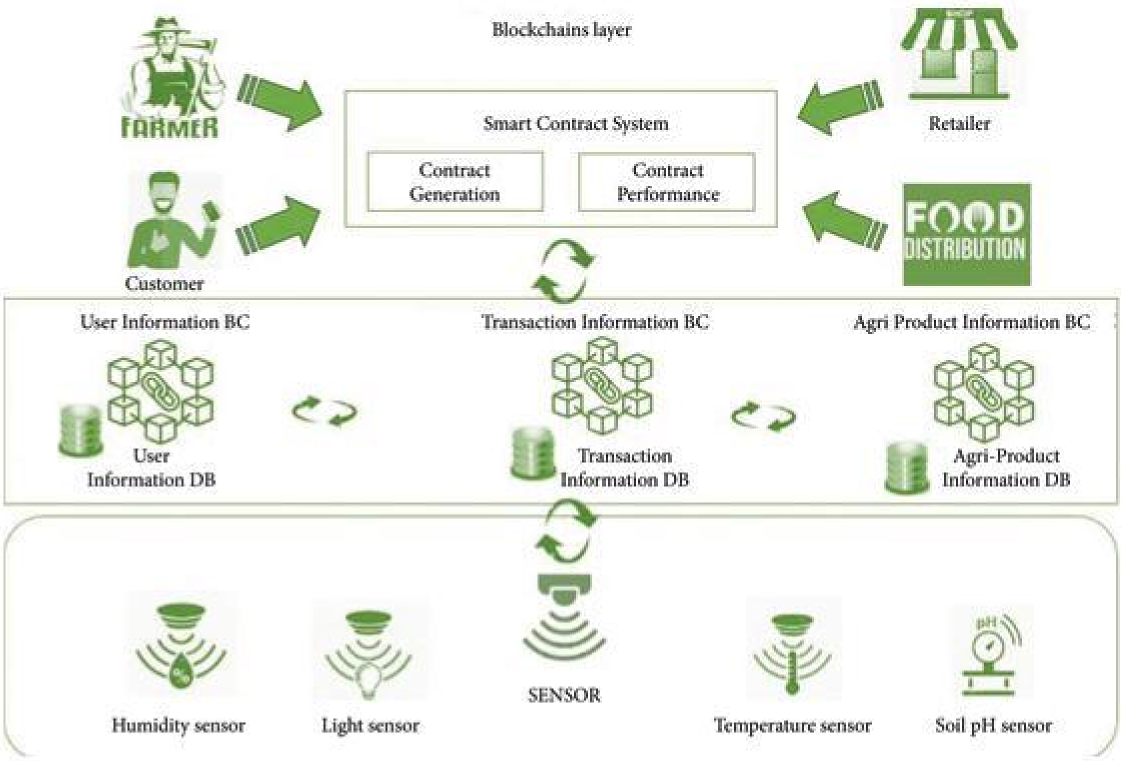
Performance Metrics: Implement performance monitoring and reporting to identify bottlenecks and areas for improvement.

Energy Efficiency: Minimize the environmental impact of blockchain operations by using energy-efficient consensus mechanisms.

These non-functional requirements are essential for the success of Climate TrackSmart using Blockchain, ensuring that the system is not only functional but also secure, reliable, and user-friendly.

5. PROJECT DESIGN

5.1 DATAFLOW DIAGRAM AND USER STORY



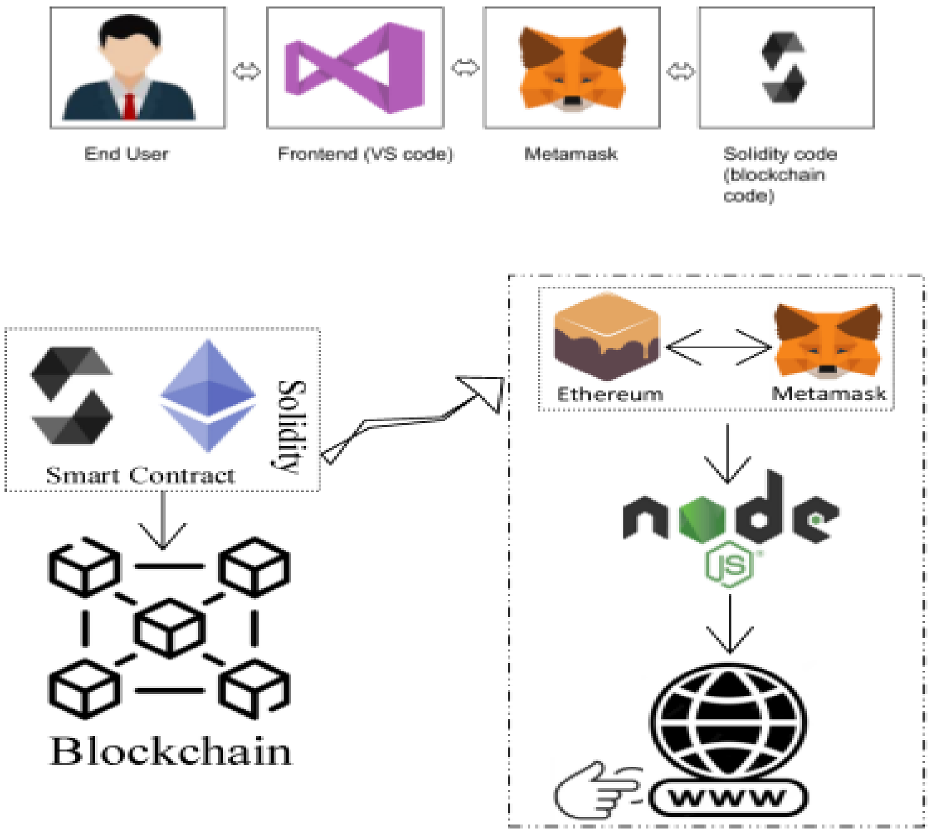
STORY 1

Vikram, a sustainability manager, used Climate TrackSmart to track and reduce her company's carbon footprint. Climate TrackSmart is a blockchain-based platform that simplifies emissions tracking and management. Maria was able to identify areas where her company could reduce emissions and make more sustainable choices.Climate TrackSmart is a powerful tool that can help businesses and individuals to reduce their environmental impact.

STORY 2

A food company used Climate TrackSmart to track and reduce emissions throughout its supply chain. Climate TrackSmart is a blockchain-based platform that simplifies emissions tracking and management. The company was able to identify areas where it could reduce emissions and make more informed decisions about its supply chain.Climate TrackSmart is a powerful tool that can help businesses and individuals to reduce their environmental impact.

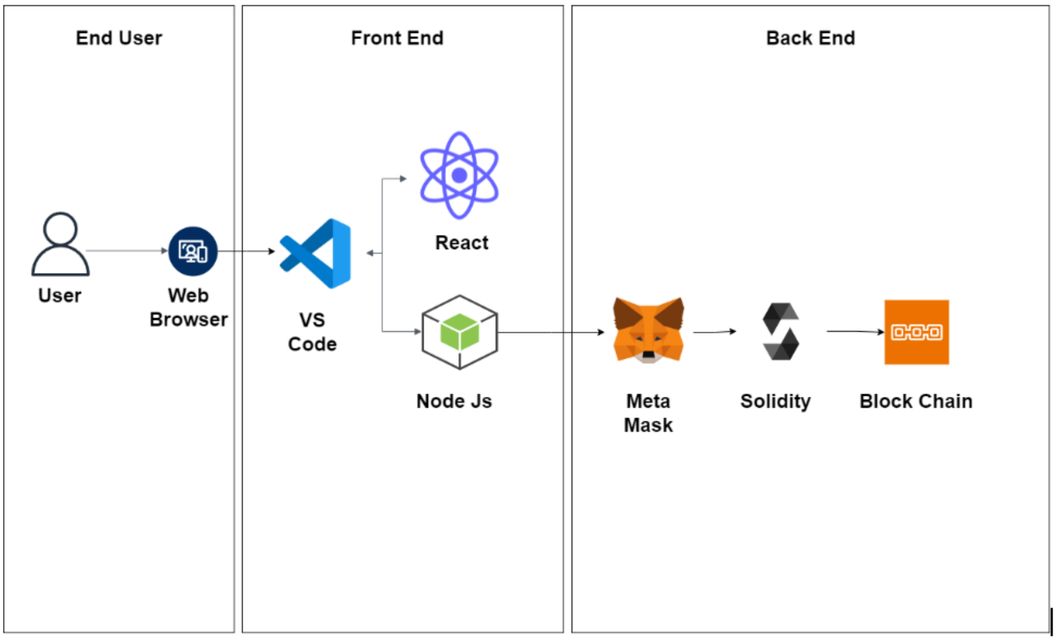
5.2 SOLUTION ARCHITECTURE

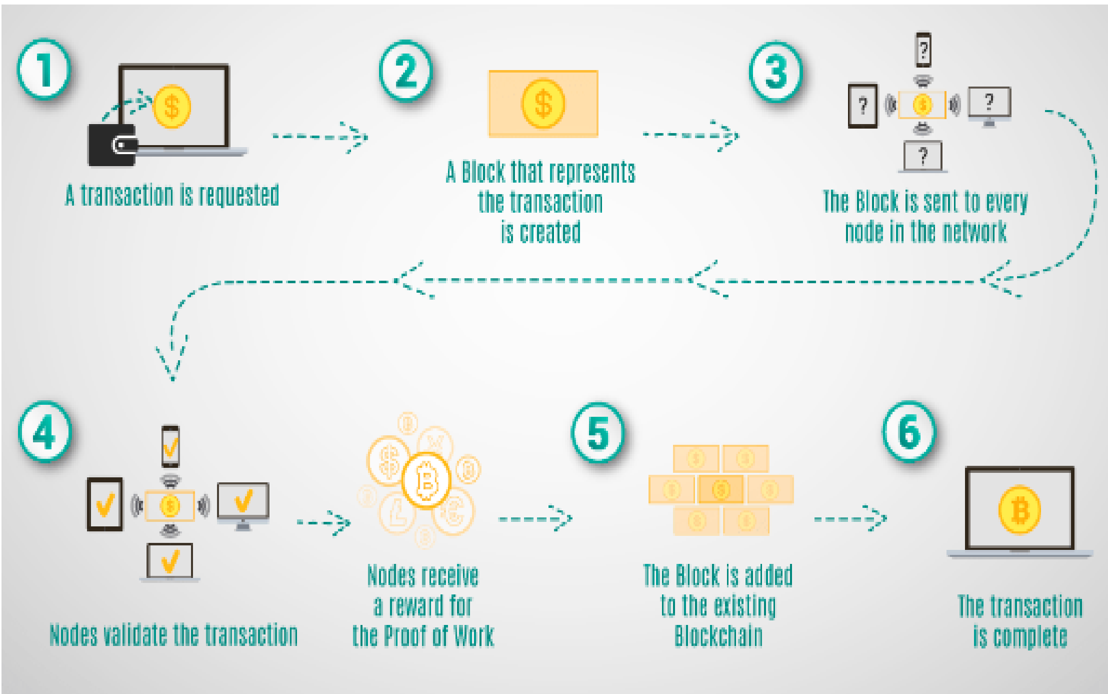


Interaction between web and the Contract

6. PROJECT PLANNING & SCHEDULING

6.1 TECHNICAL ARCHITECTURE





6.2 Sprint Planning and Estimation

Sprint planning involves selecting work items from the product backlog and committing to completing them during the upcoming sprint .

• **Reviewing Product Backlog**: Our project team, consisting of the product owner, scrum master, and development team, regularly reviews the items in the product backlog. We evaluate user stories and technical tasks, taking into account the project's evolving needs and priorities.

• **Setting Sprint Goals**: Based on the product backlog, our team establishes clear sprint goals. These goals guide the team's efforts during the sprint and ensure alignment with the broader project objectives.

• **Breaking Down User Stories**: User stories and tasks are further decomposed into smaller, actionable sub-tasks. This detailed breakdown helps create a comprehensive plan for the sprint.

• **Estimating Work**: Our development team employs agile estimation techniques, such as story points and t-shirt sizes, to estimate the effort required for each task. These estimates guide the team in understanding the scope and complexity of work for the print.

• **Sprint Backlog**: The selected user stories and tasks, along with their estimates, constitute the sprint backlog. This forms the basis for what the team will work on during the sprint.

Estimation Techniques

• **Story Points**: Story points serve as a relative measure of the complexity and effort needed to complete a task. Tasks are assigned story point values based on their complexity compared to reference tasks.

• **T-Shirt Sizes**: To provide a quick and high-level estimate of effort, tasks are categorized into t-shirt sizes, such as small, medium, and large. This approach simplifies the estimation process for less complex tasks.

6.3 Sprint Delivering Schedule

Week 1: Establish the Core

- Set up the basic blockchain infrastructure.
- Implement a minimal user registration and authentication system.
- Develop a rudimentary transaction recording feature.
- Focus on fundamental security measures.

Week 2: Expand and Enhance

- Extend transaction recording to support more transaction types.
- Add basic real-time updates for transaction status.
- Enhance user authentication with multi-factor authentication.
- Begin developing a user dashboard.

Week 3: Finalize and Prepare

- Complete the user dashboard with additional features.
- Perform minimal compliance checks.
- Conduct basic testing and issue resolution.
- Create essential user support resources.

7. CODING AND SOLUTIONING

7.1 FEATURE 1

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract climateChange{
    struct ClimateData {
        uint timestamp;
        string details;
    }

    mapping(address => ClimateData) public climateRecords;
```

```
    function addClimateData(string memory details) public {
        ClimateData memory newData = ClimateData(block.timestamp, details);
        climateRecords[msg.sender] = newData;
    }

    function getClimateData() public view returns (ClimateData memory) {
        return climateRecords[msg.sender];
    }

    function updateClimateData(string memory details) public {
        climateRecords[msg.sender].details = details;
    }

    // These are very basic functions written to carry out the operation

}
```

The Solidity code you provided is a simple smart contract for tracking climate data. It has three functions:

addClimateData(): This function allows users to add a new climate data record to their account. The record includes a timestamp and a string containing the details of the data.
getClimateData(): This function allows users to view their climate data records.
updateClimateData(): This function allows users to update the details of an existing climate data record.

This smart contract can be used for a variety of purposes, such as:
Individuals and organizations can use this smart contract to track their carbon emissions over time. This can help them to identify areas where they can reduce their emissions and to offset their emissions by purchasing carbon credits.Researchers and organizations can use this smart contract to monitor the impacts of climate change on different regions and ecosystems. This data can be used to inform climate change adaptation and mitigation strategies.By using a blockchain-based smart contract to

track climate data, users can ensure that the data is transparent and tamper-proof. This can help to build trust between stakeholders and to promote accountability for climate action.

7.2 FEATURE 2

FRONTEND *(JAVASCRIPT)*

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

function Home() {
  const [Id, setId] = useState("");
  const [ClimateDet, setClimateDet] = useState("");
  const [UpdateClimateDetails, setUpdateClimateDetails] = useState("");


  const [TranId, setTranId] = useState("");
  const [Owner, setOwner] = useState("");
  const [BookId, setBookId] = useState("");
  const [BookDet, setBookDet] = useState("");
  const [Wallet, setWallet] = useState("");


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

  const handleClimateDetails = (e) => {
    setClimateDet(e.target.value)
  }
```

```
const handleClimate = async () => {
  try {
    let tx = await contract.addClimateData(ClimateDet)
    let wait = await tx.wait()
    alert(wait.transactionHash)
    console.log(wait);
  } catch (error) {
    alert(error)
  }
}

const handleDrugId = (e) => {
  setTranId(e.target.value)
}

const handleUpdatedClimateDetails = (e) => {
  setUpdateClimateDetails(e.target.value)
}

const handleUpdateClimate = async () => {
  try {
    let tx = await contract.updateClimateData(UpdateClimateDetails)
    let wait = await tx.wait()
    console.log(wait);
    alert(wait.transactionHash)
  } catch (error) {
    alert(error)
  }
}

const handleTollDetailsId = (e) => {
```

```
    setBookId(e.target.value)
  }

  const handleGetClimateDetails = async () => {
    try {
      let tx = await contract.getClimateData()
      let arr = []
      tx.map(e => {
        arr.push(e)
      })

      console.log(tx);
      setBookDet(arr)
    } catch (error) {
      alert(error)
      console.log(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" }}>Climate Change</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={handleClimateDetails} type="textArea" placeholder="Enter climate
details" value={ClimateDet} /> <br /> */}
                 <textarea rows="4" cols="30" name="comment" placeholder="Enter
climate Details" onChange={handleClimateDetails}
value={ClimateDet}></textarea>
                <Button onClick={handleClimate} style={{ marginTop: "10px" }}
variant="primary"> Add Climate Data</Button>
     </div>
    </Col>
     <Col>
       <div>
                <textarea rows="4" cols="30" name="comment" placeholder="Update
existing climate Details" onChange={handleUpdatedClimateDetails}
value={UpdateClimateDetails}></textarea>
              <Button onClick={handleUpdateClimate} style={{ marginTop: "10px" }}
variant="primary"> Update Climate Data</Button>


       </div>
```

```
      </Col>
    </Row>
    <Row>
       <Col>
          <div style={{ margin: "auto", marginTop:"100px" }}>
                {/* <input style={{ marginTop: "10px", borderRadius: "5px" }}
onChange={handleTollDetailsId} type="number" placeholder="Enter Highway Id"
value={BookId} /><br /> */}
             <Button onClick={handleGetClimateDetails} style={{ marginTop: "10px"
}} variant="primary">Get Climate Data</Button>
             {BookDet ? BookDet?.map(e => {
               return <p>{e.toString()}</p>
             }) : <p></p>}
          </div>
        </Col>
    </Row>
    </Container>

  </div>
 )
}

export default Home;
```

Contract ABI (Application Binary Interface):

The abi variable holds the ABI of an Ethereum smart contract. ABIs are essential for encoding and decoding function calls and data when interacting with the Ethereum blockchain.

MetaMask Check:

The code first checks whether the MetaMask wallet extension is installed in the user's browser. If MetaMask is not detected, it displays an alert notifying the user that MetaMask is not found and provides a link to download it.

Ethers.js Configuration:

It imports the ethers library, which is a popular library for Ethereum development.It creates a provider using Web3Provider, which connects to the user's MetaMask wallet and provides access to Ethereum. It creates a signer to interact with the Ethereum blockchain on behalf of the user.It defines an Ethereum contract address and sets up the contract object using ethers.Contract, allowing the JavaScript code to interact with the contract's functions.In summary, this code is used for interacting with an Ethereum smart contract through MetaMask and ethers.js. It configures the necessary Ethereum provider and signer for communication with the blockchain and sets up a contract object for executing functions and fetching data from the specified contract address using the provided ABI.

7. PERFORMANCE TESTING

8.1 PERFORMANCE METRICS

Transaction throughput: This is the number of transactions that the smart contract can process per second. This is important to measure to ensure that the smart contract can handle the expected load, especially if it is being used by a large number of users.

Gas costs: The cost of gas is the amount of ETH that users must pay to execute transactions on the Ethereum blockchain. It is important to measure the gas costs associated with using the smart contract to ensure that it is affordable for users.

Response times: This is the amount of time it takes for the smart contract to respond to transactions. This is important to measure to ensure that the smart contract is responsive and does not cause users to experience long wait times.

To perform performance testing, you can use a variety of tools and techniques. One common approach is to use a load testing tool to simulate a large number of users accessing the smart contract at the same time. This can help you to identify any performance bottlenecks and to optimize the smart contract accordingly.

Here are some specific performance tests that you could run on the smart contract:

AddClimateData() test: This test would measure the transaction throughput and gas costs of the addClimateData() function. You could run this test with different amounts of data to see how the performance scales.

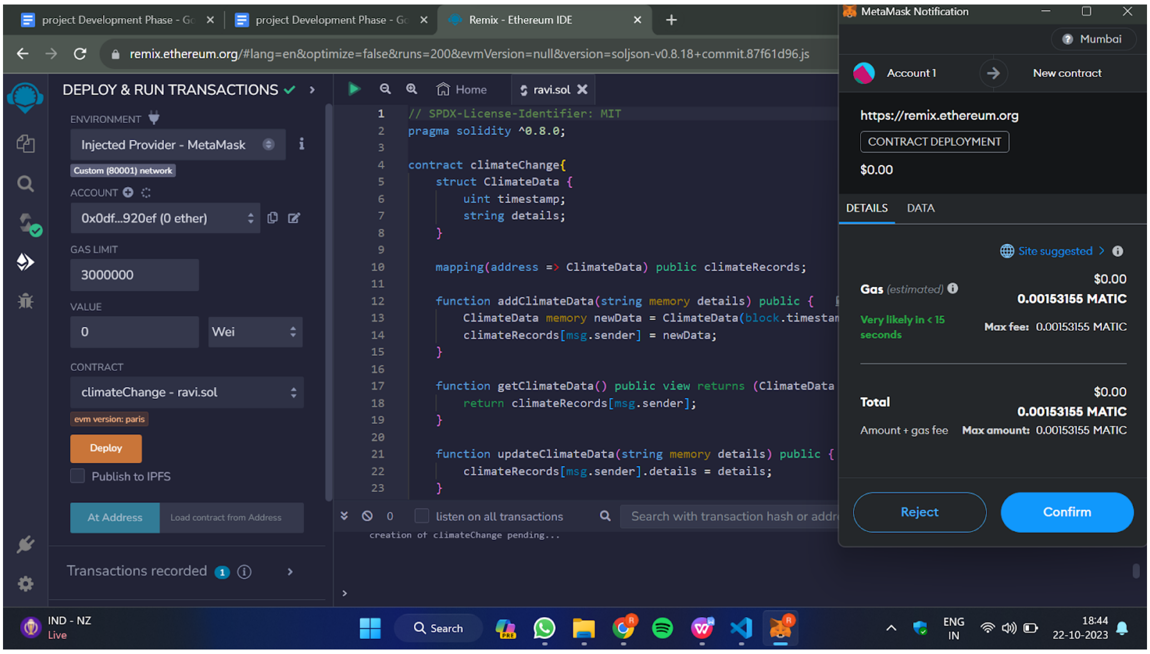
GetClimateData() test: This test would measure the transaction throughput and gas costs of the getClimateData() function. You could run this test from different accounts to see if there is any difference in performance.

UpdateClimateData() test: This test would measure the transaction throughput and gas costs of the updateClimateData() function. You could run this test with different amounts of data to see how the performance scales.

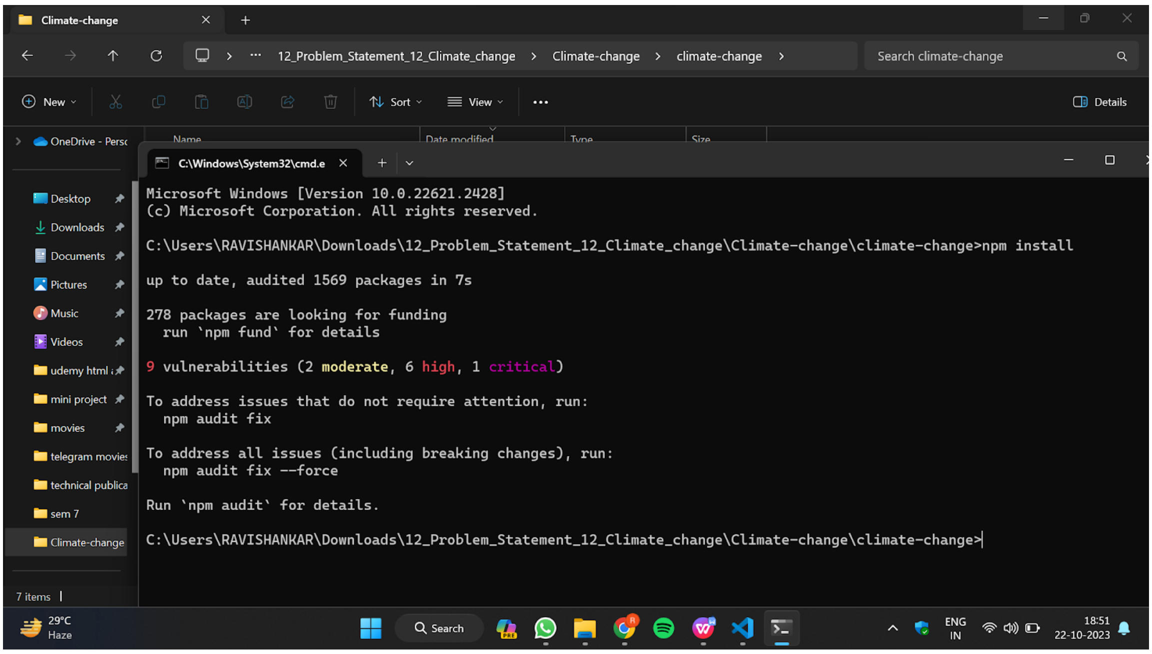
In addition to these specific tests, you could also run more general performance tests to assess the overall responsiveness and scalability of the smart contract. For example, you could run a load test to simulate a large number of users accessing the smart contract at the same time. You could also monitor the gas costs of the smart contract over time to see if they are increasing.By performing performance testing, you can ensure that your smart contract is able to handle the expected load and that it is affordable and responsive for users.

9.RESULTS

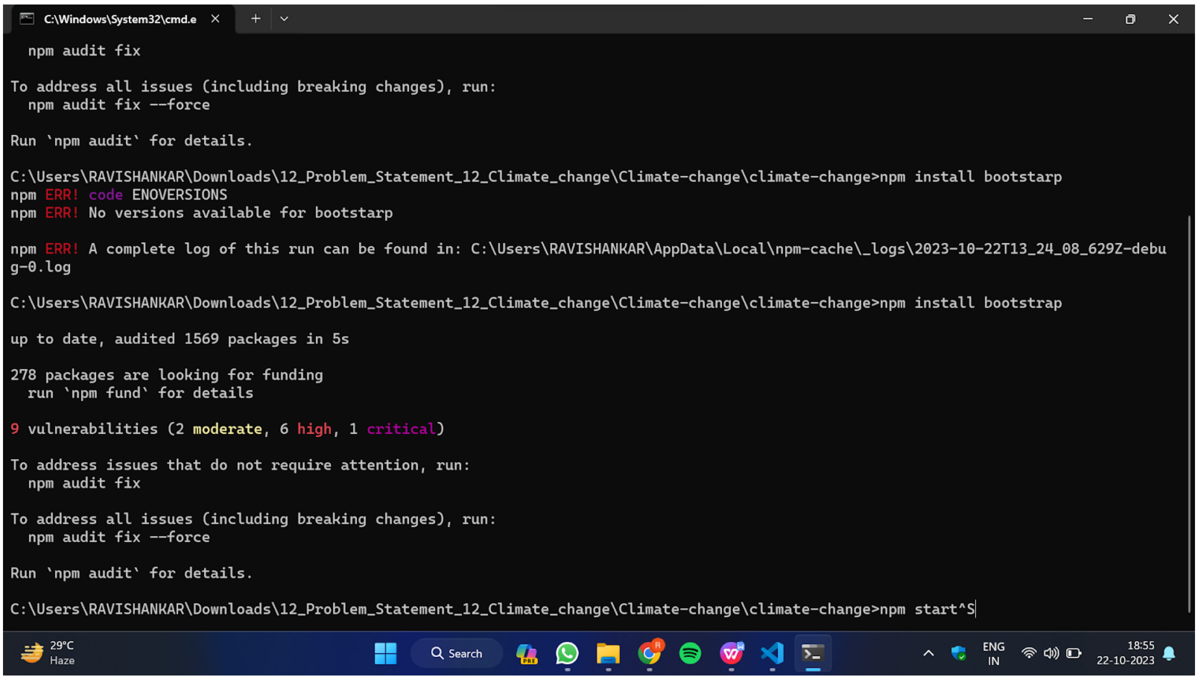
9.1 OUTPUT SCREENSHOTS



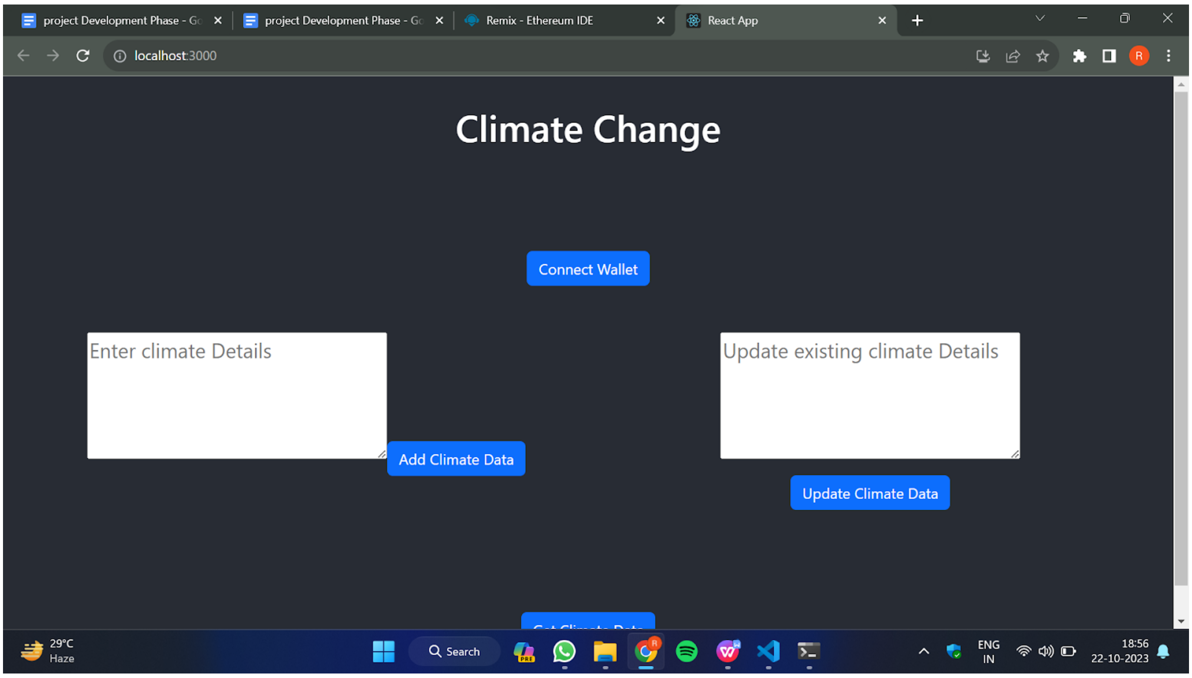
CREATING A SMART CONTRACT



INSTALLING DEPENDENCY



HOSTING THE SITE



OUTPUT SCREENSHOT

10. ADVANTAGES AND DISADVANTAGES

10.1 ADVANTAGES

Transparency and accountability: Blockchain is a transparent and tamper-proof technology, which can help to build trust between stakeholders and promote accountability for climate action.

Efficiency: Blockchain can help to streamline and automate climate change initiatives, such as carbon trading and renewable energy management.

Inclusivity: Blockchain can provide access to climate change solutions to people and communities that are currently underserved.

Innovation: Blockchain is a rapidly developing technology, and new and innovative applications for climate change are emerging all the time.

Empowerment: Blockchain can empower individuals and organizations to take action on climate change in a more meaningful and effective way.

10.2 DISADVANTAGES

Energy consumption: The mining process used to secure some blockchain networks consumes a lot of energy. This is a major concern, especially given the urgency of the climate crisis.

Complexity: Blockchain technology can be complex and difficult to understand. This can make it difficult for businesses and organizations to adopt and implement blockchain solutions for climate change.

Scalability: Some blockchain networks are not yet scalable enough to handle the large number of transactions that would be required for widespread adoption of blockchain-based climate change solutions.

Regulation: The regulatory landscape for blockchain is still evolving. This uncertainty can make it difficult for businesses and organizations to invest in blockchain-based climate change solutions.

Security: Blockchain networks are generally secure, but there have been some high-profile hacks and scams. It is important to be aware of the security risks associated with blockchain before using it for climate change solutions.

11. CONCLUSION

Blockchain technology has the potential to play a significant role in addressing the climate crisis. It can be used to improve transparency and accountability in climate action, streamline and automate climate change initiatives, provide access to climate change solutions to underserved communities, and foster innovation in the climate change space.

While blockchain has the potential to be a powerful tool for climate change, there are also some challenges that need to be addressed. One challenge is the energy consumption associated with some blockchain networks. Another challenge is the complexity of blockchain technology, which can make it difficult for businesses and organizations to adopt and implement blockchain solutions. Additionally, the regulatory landscape for blockchain is still evolving, which can create uncertainty for businesses and investors.Despite these challenges, the potential benefits of using blockchain for climate change are significant. As blockchain technology continues to develop and mature, we can expect to see even more innovative and effective applications of blockchain for climate action.Blockchain is a powerful tool that can be used to support a variety of climate change initiatives.Blockchain can help to improve transparency and accountability in climate action, streamline and automate climate change initiatives, provide access to climate change solutions to underserved communities, and foster innovation in the climate change space.There are some challenges that need to be addressed before blockchain can be widely adopted for climate change, such as the energy consumption associated with some blockchain networks, the complexity of blockchain technology, and the evolving regulatory landscape.Overall, the potential benefits of using blockchain for climate change outweigh the disadvantages.As blockchain technology continues to develop and mature, we can expect to see even more innovative and effective applications of blockchain for climate action.

12. FUTURE SCOPE

The future scope of blockchain for climate change is very promising. As blockchain technology continues to develop and mature, we can expect to see even more innovative and effective applications of blockchain for climate action.Here are some specific areas where we can expect to see significant progress in the use of blockchain for climate change in the future:

Carbon trading: Blockchain-based carbon trading markets are still in their early stages of development, but they have the potential to revolutionize the way carbon emissions are traded. Blockchain can help to create more efficient, transparent, and secure carbon trading markets. This can help to reduce the cost of carbon compliance and incentivize businesses to reduce their emissions.In the future, we can expect to see the emergence of global carbon trading markets based on blockchain technology. These markets would allow businesses and individuals from all over the world to trade carbon credits in a seamless and efficient manner. This could help to accelerate the global transition to a low-carbon economy.

Renewable energy management: Blockchain can be used to manage the distribution of renewable energy and to create peer-to-peer energy trading markets. This can help to increase the uptake of renewable energy and reduce our reliance on fossil fuels.In the future, we can expect to see the widespread adoption of blockchain-based renewable energy management systems. These systems would allow renewable energy producers to sell their energy directly to consumers without the need for intermediaries. This would help to reduce the cost of renewable energy and make it more accessible to everyone.

Climate finance: Blockchain can be used to facilitate the flow of climate finance to developing countries and to underserved communities. This can help to support climate change mitigation and adaptation projects around the world.In the future, we can expect to see the development of innovative blockchain-based climate finance mechanisms. These mechanisms could help to raise more money for climate change projects and ensure that the funds are used effectively. For example, blockchain-based crowdfunding platforms could be used to raise money for climate change projects in developing countries.

Climate data management: Blockchain can be used to create secure and tamper-proof repositories for climate data. This data can be used to track greenhouse gas emissions, monitor the impacts of climate change, and develop climate change adaptation and mitigation strategies.In the future, we can expect to see the widespread adoption of blockchain-based climate data management systems. These systems would help to improve the quality and accessibility of climate data. This would enable policymakers, scientists, and businesses to make better decisions about climate change.

In addition to these specific areas, we can also expect to see blockchain being used to address climate change in other ways in the future. For example, blockchain could be used to:

1. Develop new climate change mitigation and adaptation technologies
2. Create new climate change markets, such as markets for carbon credits and renewable energy credits
3. Promote climate change education and awareness
4. Support climate change activism and advocacy
5. Overall, the future of blockchain for climate change is very promising. As blockchain technology continues to develop and mature, we can expect to see even more innovative and effective applications of blockchain for climate action.

Here are some creative ideas for how blockchain could be used to address climate change in the future:

1. A blockchain-based platform that allows individuals and organizations to offset their carbon emissions by investing in renewable energy projects.
2. A blockchain-based system for tracking the carbon footprint of products and services, so that consumers can make more informed choices about what they buy.

3. A blockchain-based platform for crowdfunding climate change research and development.
4. A blockchain-based system for monitoring and verifying the implementation of climate change policies.

13. APPENDIX

SOURCE CODE

Climate.sol

```
// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract climateChange{
   struct ClimateData {
      uint timestamp;
      string details;
   }

   mapping(address => ClimateData) public climateRecords;

   function addClimateData(string memory details) public {
      ClimateData memory newData = ClimateData(block.timestamp, details);
      climateRecords[msg.sender] = newData;
   }

   function getClimateData() public view returns (ClimateData memory) {
      return climateRecords[msg.sender];
   }

   function updateClimateData(string memory details) public {
      climateRecords[msg.sender].details = details;
   }
```

```
    // These are very basic functions written to carry out the operation

}
```

Connector.js

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

const abi = [
  {
    inputs: [
      {
        internalType: "string",
        name: "details",
        type: "string",
      },
    ],
    name: "addClimateData",
    outputs: [],
    stateMutability: "nonpayable",
    type: "function",
  },
  {
    inputs: [
      {
        internalType: "address",
        name: "",
        type: "address",
      },
    ],
    name: "climateRecords",
    outputs: [
      {
        internalType: "uint256",
```

```
      name: "timestamp",
      type: "uint256",
    },
    {
      internalType: "string",
      name: "details",
      type: "string",
    },
  ],
  stateMutability: "view",
  type: "function",
},
{
  inputs: [],
  name: "getClimateData",
  outputs: [
    {
      components: [
        {
          internalType: "uint256",
          name: "timestamp",
          type: "uint256",
        },
        {
          internalType: "string",
          name: "details",
          type: "string",
        },
      ],
      internalType: "struct climateChange.ClimateData",
      name: "",
      type: "tuple",
    },
  ],
```

```
    stateMutability: "view",
    type: "function",
  },
  {
    inputs: [
      {
        internalType: "string",
        name: "details",
        type: "string",
      },
    ],
    name: "updateClimateData",
    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 = "0xd81a757C4804Fb1E59Ef6819dd1CCb9bBa02b89a";

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

Home.js

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

```
import { signer } from "./connector";
import { provider } from "./connector";

function Home() {
  const [Id, setId] = useState("");
  const [ClimateDet, setClimateDet] = useState("");
  const [UpdateClimateDetails, setUpdateClimateDetails] = useState("");


  const [TranId, setTranId] = useState("");
  const [Owner, setOwner] = useState("");
  const [BookId, setBookId] = useState("");
  const [BookDet, setBookDet] = useState("");
  const [Wallet, setWallet] = useState("");


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

  const handleClimateDetails = (e) => {
    setClimateDet(e.target.value)
  }



  const handleClimate = async () => {
    try {
      let tx = await contract.addClimateData(ClimateDet)
      let wait = await tx.wait()
      alert(wait.transactionHash)
      console.log(wait);
    } catch (error) {
      alert(error)
```

```
  }
}

const handleDrugId = (e) => {
  setTranId(e.target.value)
}

const handleUpdatedClimateDetails = (e) => {
  setUpdateClimateDetails(e.target.value)
}

const handleUpdateClimate = async () => {
  try {
    let tx = await contract.updateClimateData(UpdateClimateDetails)
    let wait = await tx.wait()
    console.log(wait);
    alert(wait.transactionHash)
  } catch (error) {
    alert(error)
  }
}

const handleTollDetailsId = (e) => {
  setBookId(e.target.value)
}

const handleGetClimateDetails = async () => {
  try {
    let tx = await contract.getClimateData()
    let arr = []
    tx.map(e => {
      arr.push(e)
    })
```

```
      console.log(tx);
      setBookDet(arr)
    } catch (error) {
      alert(error)
      console.log(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" }}>Climate Change</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={handleClimateDetails} type="textArea" placeholder="Enter climate
details" value={ClimateDet} /> <br /> */}
                  <textarea rows="4" cols="30" name="comment" placeholder="Enter
climate Details" onChange={handleClimateDetails}
value={ClimateDet}></textarea>
                <Button onClick={handleClimate} style={{ marginTop: "10px" }}
variant="primary"> Add Climate Data</Button>
 </div>
</Col>
 <Col>
   <div>
                <textarea rows="4" cols="30" name="comment" placeholder="Update
existing climate Details" onChange={handleUpdatedClimateDetails}
value={UpdateClimateDetails}></textarea>
              <Button onClick={handleUpdateClimate} style={{ marginTop: "10px" }}
variant="primary"> Update Climate Data</Button>


   </div>
 </Col>
</Row>
<Row>
  <Col>
     <div style={{ margin: "auto", marginTop:"100px" }}>
                  {/* <input style={{ marginTop: "10px", borderRadius: "5px" }}
onChange={handleTollDetailsId} type="number" placeholder="Enter Highway Id"
value={BookId} /><br /> */}
              <Button onClick={handleGetClimateDetails} style={{ marginTop: "10px"
}} variant="primary">Get Climate Data</Button>
              {BookDet ? BookDet?.map(e => {
```

```
                    return <p>{e.toString()}</p>
                  }) : <p></p>}
            </div>
          </Col>
    </Row>
    </Container>

  </div>
 )
}

export default Home;
```

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;
```

App.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);
  }
```

```
}
```

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();
```

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;
```

}

Github link:https://github.com/sanjay32002/climate-smarttrack-using-blockchain.git

demo_video_link:https://drive.google.com/file/d/1wbBI4lwahc0GPQAmUauzfCsuOj2AiUZT/view?usp=sharing