Project Report

Climate smart track Using Block-chain

1.INTRODUCTION:

The Climate track smart process using the ethereum blockchain to create a decentralized and immutable ledger that records the entire lifecycle of a product,capturing its environmental impact at every step smart contract on the Ethereum blockchain can automated execution of rules and agreements ,ensuring that the data entered into the system is accurate and transparent[1].

The creating a smart contract on the Ethereum blockchain to track the progress of a climate agreement involves several steps.Ethereum Smart Contract are written in solidity, a Programming language specifically designed for Ethereum[2].Below is a high-level outline of how you might design such a Smart Contract Time-based progress updates,implement a mechanism that prevent backdated progress updated and ensures that the progress is updated at appropriate intervals[3].

Climate change poses an imminent threat to our planet, and there is an urgent need for innovative solutions to address this global crisis[4]. Blockchain technology has emerged as a powerful tool that can revolutionize the way we track, monitor, and address climate-related issues[5]. In this digital era, the integration of blockchain into climate tracking can provide transparency, security, and efficiency, offering a promising path to combat environmental challenges[6].

Climate Track Smart is a groundbreaking initiative that leverages blockchain technology to create a transparent and decentralized system for monitoring and managing climate data[7]. This initiative offers a new approach to climate tracking, allowing for real-time, immutable, and secure record-keeping of critical environmental data[8].

Project Overview

ClimateTrack Smart is an innovative project that leverages blockchain technology to address the critical challenges associated with climate monitoring, data accuracy, and transparent reporting. The project's primary goal is to create a robust and secure platform that empowers governments, organizations, and individuals to monitor, record, and report climate-related data with unparalleled accuracy and transparency. By harnessing the power of blockchain, ClimateTrack Smart aims to revolutionize the way we collect and manage climate data, fostering a global commitment to combat climate change.

Key Components:

1. Blockchain Technology:

- ❖ Implement a blockchain network (public or private) to store climate-related data in a secure, immutable, and decentralized manner.
- ❖ Ensure data integrity, traceability, and transparency through cryptographic verification and smart contracts.

2. Climate Data Collection:

- ❖ Develop a user-friendly mobile application and IoT devices to facilitate the collection of various climate-related data, such as temperature, humidity, air quality, and emissions.
- ❖ Integrate satellite and remote sensing data for a comprehensive global perspective.

3. Data Aggregation and Validation:

- ❖ Employ machine learning algorithms to aggregate, clean, and validate the collected data to ensure accuracy and reliability.
- ❖ Establish data validation processes that involve multiple stakeholders, including scientists, environmental agencies, and the public.

4. Smart Contracts and Incentive Mechanisms:

- ❖ Create smart contracts that reward data contributors for accurate and verified data.
- ❖ Implement incentives to encourage individuals and organizations to participate in data collection and reporting.

5. User-Friendly Dashboard:

- ❖ Develop a web-based dashboard for easy data access and visualization.
- ❖ Enable users to track climate metrics, view historical data, and receive real-time updates.

6. Compliance and Reporting:

- ❖ Incorporate global climate reporting standards and guidelines to ensure data compliance with international agreements like the Paris Agreement.
- ❖ Enable automated, transparent reporting to relevant environmental agencies and institutions.

7. Security and Privacy:

- ❖ Implement robust security measures to protect user data and ensure the confidentiality of sensitive information.
- ❖ Adhere to privacy regulations and user consent for data collection and storage

8. Scalability and Global Reach:

- ❖ Ensure the platform is scalable to accommodate a growing user base and increasing data volumes.
- ❖ Aim for a global reach to cover a wide range of geographic regions and ecosystems.

Purpose:

Climate tracking with blockchain technology serves several essential purposes in the fight against climate change. Blockchain offers a transparent, secure, and decentralized ledger that can revolutionize how we monitor and address environmental issues.

blockchain ensures data integrity and transparency, making it nearly impossible to manipulate or falsify information. This is crucial for tracking carbon emissions, deforestation, and other environmental data, as it builds trust among stakeholders and governments.

it enables efficient and secure carbon credit trading. By recording carbon reduction efforts on the blockchain, companies can transparently buy and sell carbon credits, promoting incentives for businesses to reduce emissions.

blockchain can facilitate supply chain sustainability. By tracking the journey of products from source to consumer, it encourages responsible sourcing and production practices, thus reducing the carbon footprint associated with manufacturing and transportation.

climate tracking using blockchain technology empowers stakeholders with reliable data, encourages sustainable practices through carbon credit trading, and enhances transparency in supply chains. It plays a pivotal role in accelerating the global transition to a more sustainable and eco-friendly future.

2. LITERATURE SURVEY

2.1.Existing problem and References

Introduction to Blockchain Technology and Climate Tracking

- ❖ Provide a brief overview of blockchain technology and its characteristics.
- ❖ Explain the importance of climate tracking and monitoring in the context of environmental sustainability.

Blockchain Technology in Environmental Sustainability:

- Explore how blockchain technology is being used to address environmental and sustainability challenges.
- Discuss the potential benefits of blockchain in improving transparency, data accuracy, and accountability in climate tracking.

Relevant sources: Tapscott, D., & Tapscott, A. (2016). "Blockchain Revolution: How the Technology Behind Bitcoin is Changing Money, Business, and the World."

- Kamilaris, A., Fonts, A., & Prenafeta-Boldú, F. (2019). "The Rise of Blockchain Technology in Agriculture and Food Supply Chains." Sustainability, 11(3), 863.

Climate Tracking and Data Integrity:

- ❖ Examine how blockchain can enhance the integrity and reliability of climate data.
- ❖ Discuss examples of blockchain-based systems for tracking and verifying carbon emissions, deforestation, or renewable energy production.

Relevant sources: - Zohar, A. (2015). "Bitcoin: under the hood." Communications of the ACM, 58(9), 104-113.

- Mycoo, M. K., & Lee, S. Y. (2019). "Blockchain Technology: What Is It Good for?." Frontiers in Blockchain, 2, 20.

Smart Contracts and Climate Agreements:

- ❖ Investigate how smart contracts can automate and enforce climate-related agreements and transactions.
- ❖ Analyze the potential for self-executing contracts in carbon credit trading and environmental conservation efforts.

Relevant sources:- Mougayar, W. (2016). "The Business Blockchain: Promise, Practice, and Application of the Next Internet Technology." Wiley.

Christidis, K., & Devetsikiotis, M. (2016). "Blockchains and smart contracts for the internet of things." IEEE Access, 4, 2292-2303.

Privacy and Security in Climate Data on the Blockchain:*

- ❖ Explore privacy concerns and security considerations when storing sensitive climate data on a public blockchain.
- ❖ Investigate solutions such as permissioned blockchains and data encryption.

Relevant sources:- Kosba, A., Miller, A., Shi, E., Wen, Z., & Papamanthou, C. (2016). "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts."

In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security.

Problem Statement Definition:

Problem statement:

In the face of global climate change, there is an increasing need for a reliable, secure, and transparent system to track, verify, and manage climate-related data and assets. Current methods for monitoring carbon emissions, renewable energy production, or carbon credits trading lack transparency, are often subject to fraud, and have limited cross-border compatibility. To address these challenges, the problem statement is to develop a "climate track smart" system using blockchain technology. This system should enable the secure and decentralized tracking of climate-related activities, assets, and data to ensure accuracy, prevent fraud, and facilitate efficient reporting and trading on a global scale.

Key elements of this problem statement include:

Climate Data Tracking: Designing a system that can accurately track climate-related data, such as carbon emissions, temperature changes, and renewable energy production, in real-time or near-real-time.

Verification and Transparency: Ensuring that the system provides transparent, immutable records that can be verified by relevant stakeholders, including governments, organizations, and the public.

Security and Fraud Prevention: Implementing robust security measures to prevent fraudulent or unauthorized changes to the data and ensure the integrity of the information.

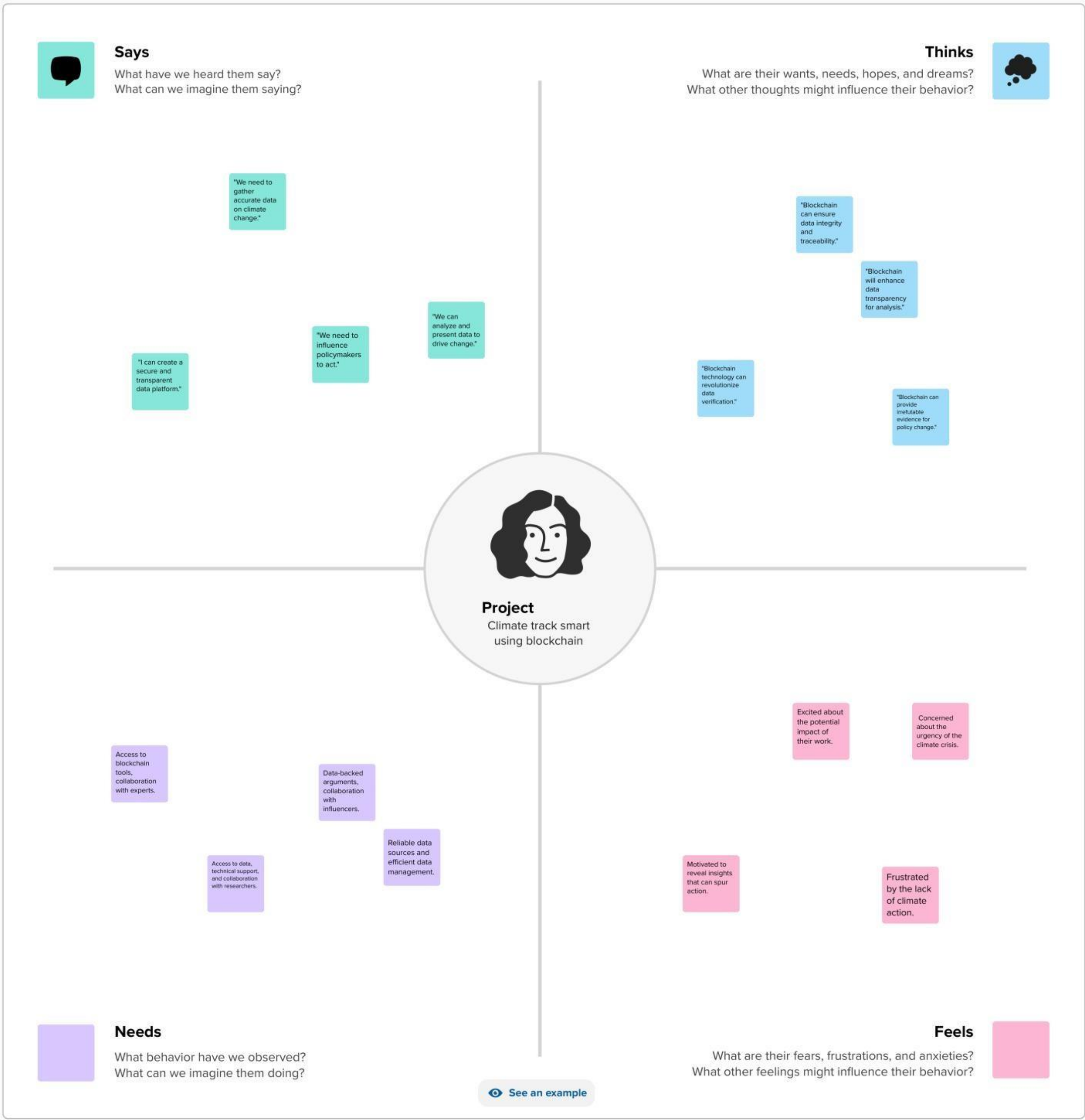
Interoperability: Creating a system that can function across borders and with different types of climate data, enabling global cooperation and consistency.

Efficiency and Automation: Developing smart contract functionalities or automation to simplify processes such as carbon credit trading, compliance reporting, and data sharing.

Problem Statement	I am (User)	I am Trying to	But	Because	Which Makes me feel
PS-1	User	To track Climate in smart contrct using blockchain	High computational Resoureces required	Large Volume of Data storage	Implement the high data Storage device
PS-2	User	To track Climate in smart contrct using blockchain	Inaaccurate Climate curent data	Huge number user use at time	Use high quality & Required Sensor
PS-3	User	To track Climate in smart contrct using blockchain	Limited Scalability Covarage	Verification and Transparency	Use Highly Scalability Devices
PS-4	User	To track Climate in smart contrct using blockchain	To Analyze The data of user form another user	Security and Fraud Prevention	To improve security.

3. IDEATION & PROPOSED SOLUTION

3.1.Empathy Map Canvas



Pain:

Blockchain Developer-Resource Constraints, Complex Problem Solving
Environmental Scientist- Data Accuracy Challenges, Complexity of Blockchain
Policy Advocate- Policy Inertia, Lack of Convincing Data
Data Analyst- Data Accessibility, Technical Challenges

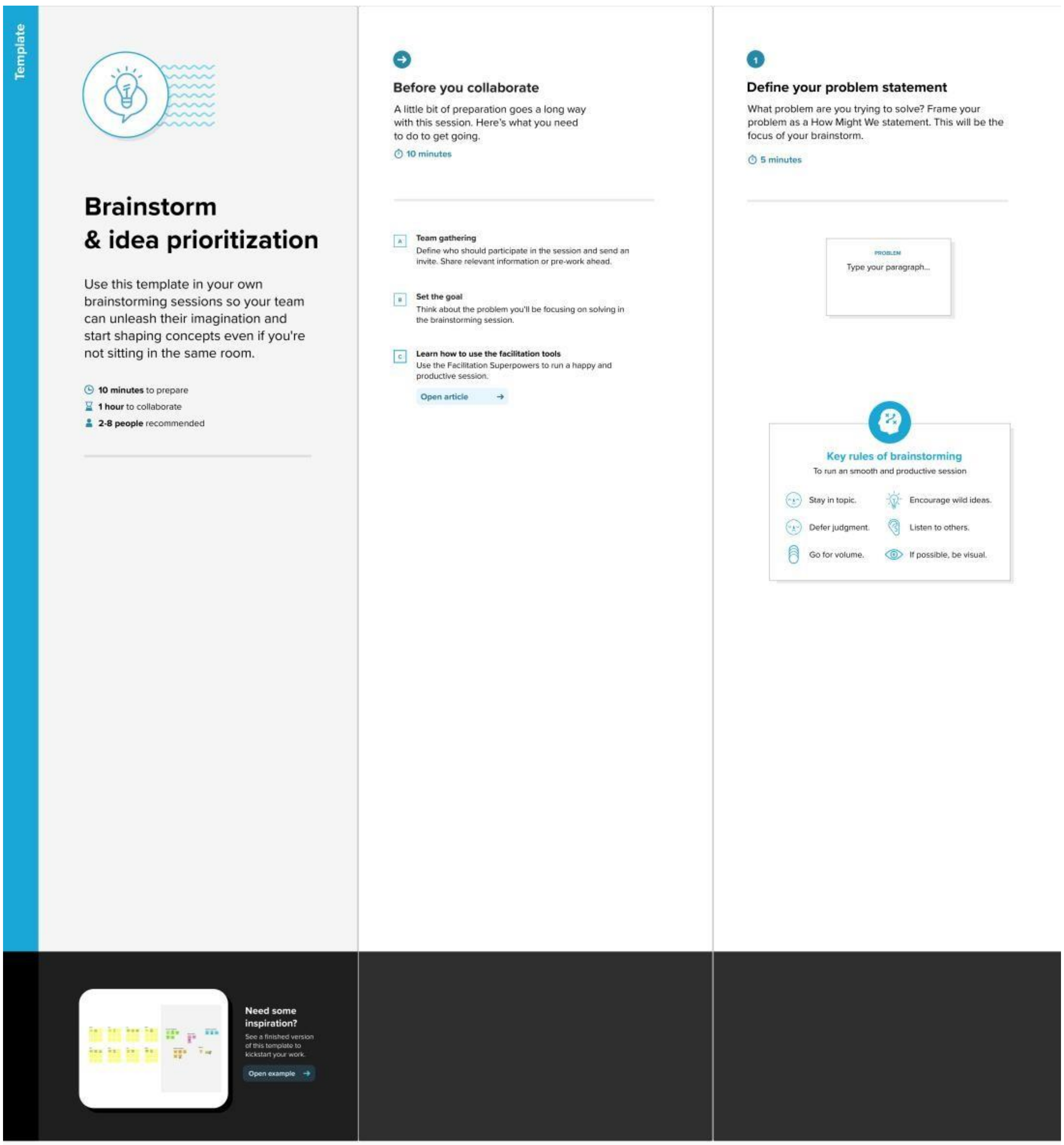
Gain:

- **Data Analyst-** Enhanced Data Transparency, Motivation through Insights
- **Policy Advocate-** Irrefutable Evidence, Collaboration with Influencers
- **Blockchain Developer-** Innovative Solution, Potential Impact
- **Environmental Scientist-** Data Integrity, Contribution to Climate Action

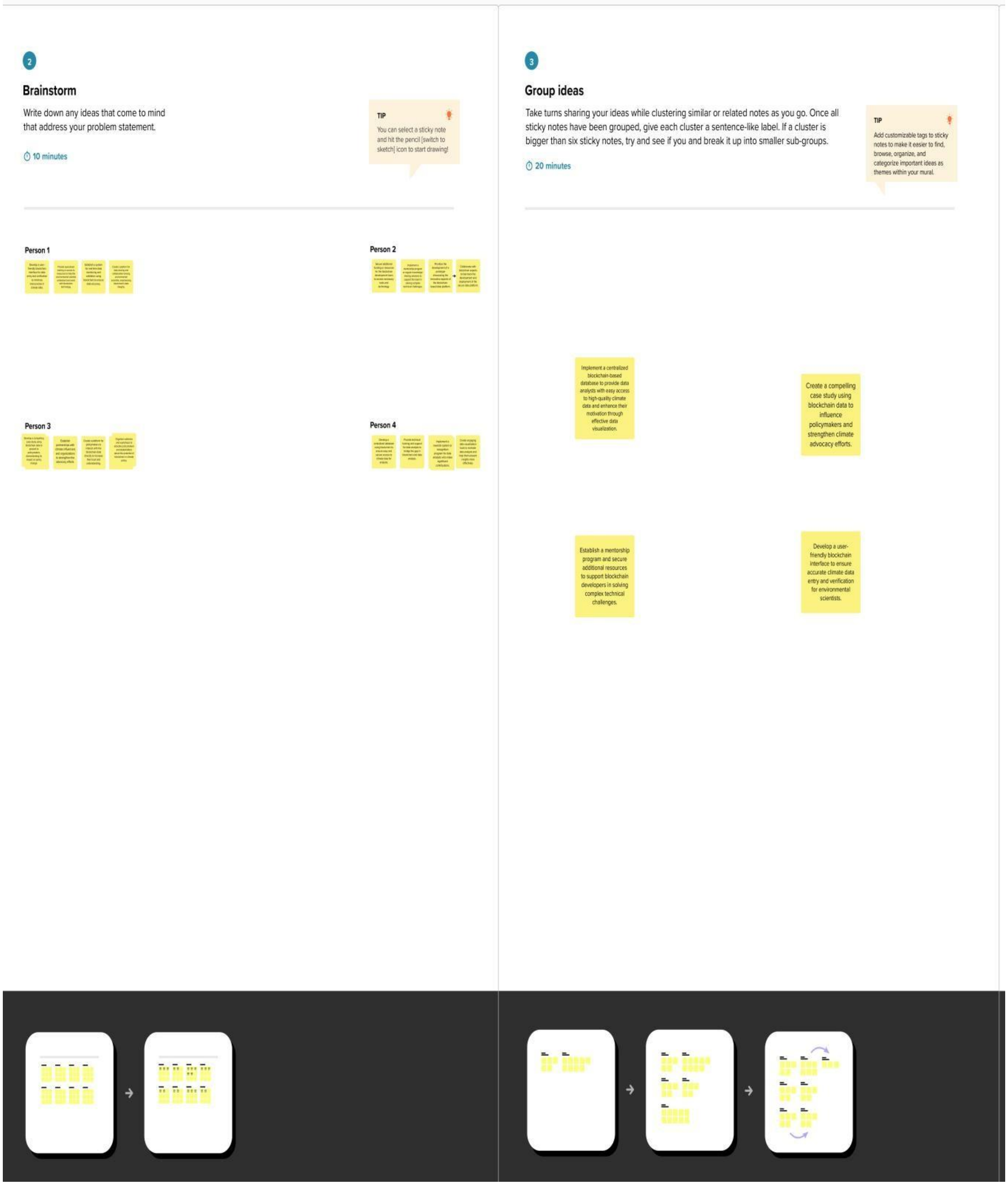
3.2.Ideation & Brainstorming

Brainstorm & Idea Prioritization

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



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization

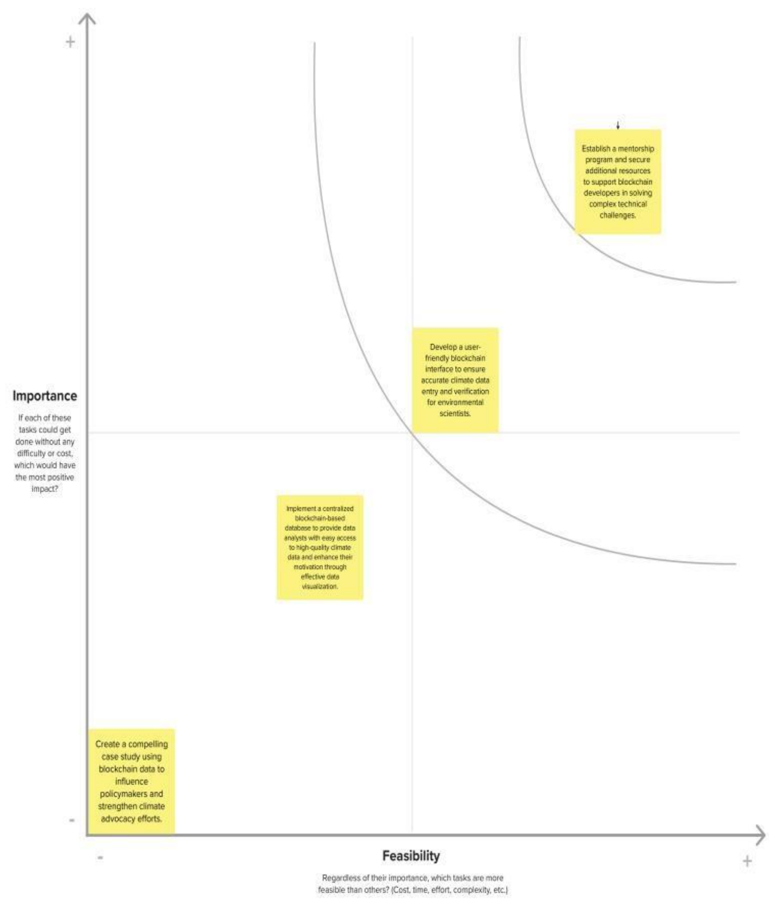
Prioritize

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

Functional requirement

Following are the functional requirement of the Proposed Solution.

FR. no	Functional Requirement(Epic)	Sub-Requirement
1	User Registration and Authentication	Users should be able to register and create accounts securely. Users must authenticate themselves to access the platform.
2	Carbon Emission Tracking:	- Users can record and track their carbon emissions, including transportation, energy consumption, and more. The system should calculate and display users' carbon footprints.

3	Carbon Offset Transactions	Users can purchase carbon offsets or credits using blockchain-based transactions. - Smart contracts manage the issuance and transfer of carbon offsets.
4	Data Integration	Integration with climate data providers and IoT devices to collect real-time climate data. - Data sources should be reliable and accurate.
5	Smart Contracts	- Deploy and execute smart contracts to automate and verify climate-related transactions. - Ensure transparency and trust in carbon offset transactions.
6	Blockchain Transparency	Ensure all carbon offset and emission data is recorded on the blockchain for transparency And trace the origin of carbon

		offsets.

Non-Functional Requirement:

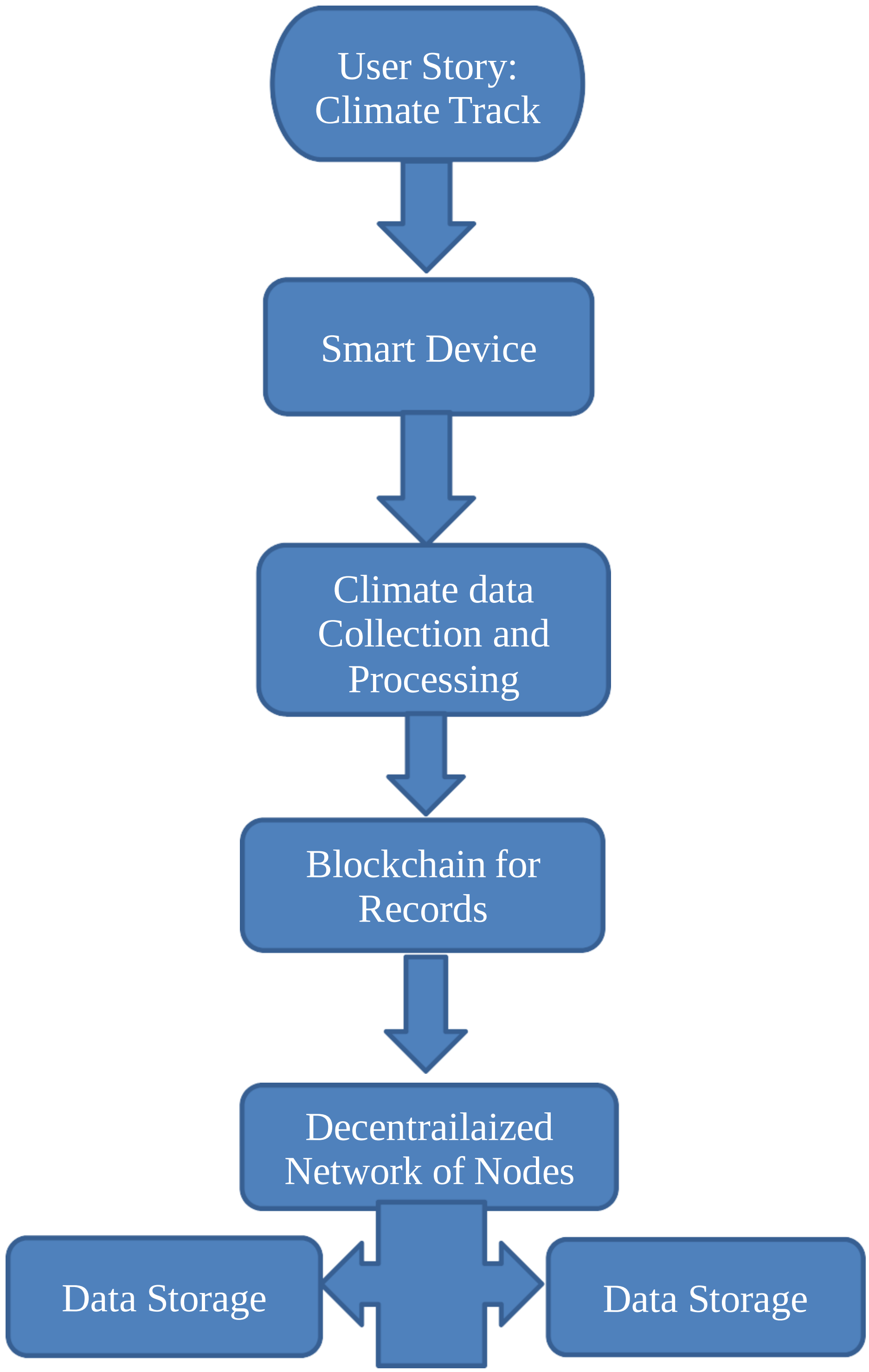
Following are the Non-functional requirement of the Proposed Solution.

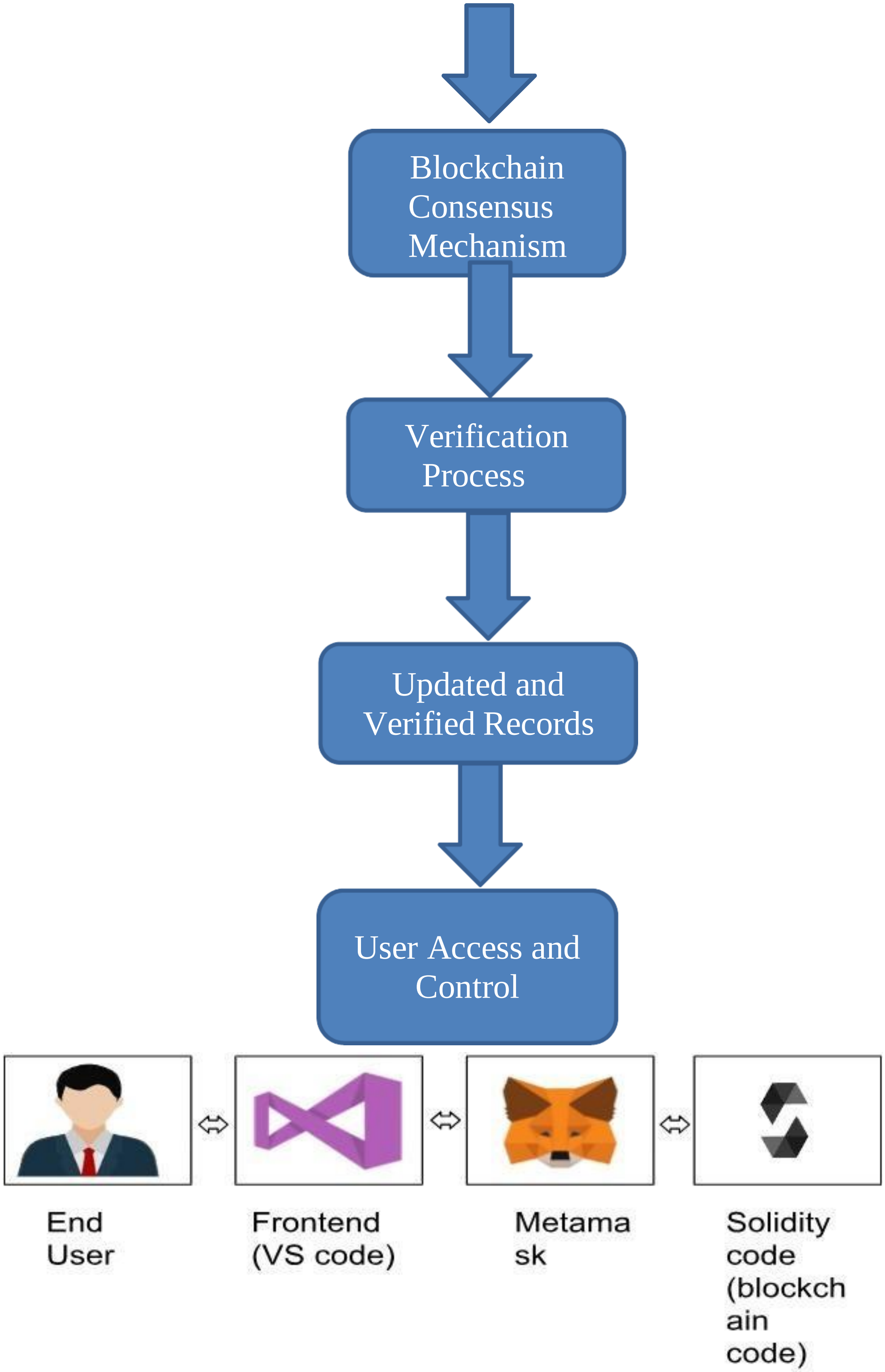
Non-FR. no	Non-Functional Requirement(Epic)	Sub-Requirement
1	Security	- Robust security measures to protect user data and ensure the immutability of the blockchain. - Compliance with data protection regulations (e.g., GDPR).
2	Scalability	The system should be able to handle a growing user base and increasing data volume efficiently.
3	Performance	- Low latency and high availability, especially for real-time climate data processing. - Minimal transaction confirmation times on the blockchain.
4	Reliability	- The system must be highly reliable, with minimal downtime. - Backup and disaster recovery mechanisms should be in place.
5	Interoperability	- Ensure compatibility with different blockchain platforms, data sources, and IoT devices. - Support industry standards for climate data.

5.PROJECT DESIGN

Data Flow Diagrams & User Stories

5.1.1Data Flow Diagrams





Blockchain Consensus Mechanism
Verification Process
Updated and Verified Records
User Access and Control
End User
Frontend (VS code)
Metamask
Solidity code (blockchain code)

5.1.2.User Stories:

User Story	Acceptance Criteria
As a User	I can create an account with my personal information. I can securely login to my account. I can link my bank account for easy transactions.
Tracking Corbon Emission	I can input my daily activities (travel, energy consumption, etc.) to calculate my carbon emissions. I can access a dashboard that displays my emissions history. I receive real-time updates and insights on reducing emissions.
Blockchain Integration	My emissions data is securely stored and timestamped on a blockchain. I can view an immutable ledger of all my emissions data.
Corbon Offset Option	I can choose from a variety of carbon offset projects. I can see the details of each project, including their impact. I can make payments to offset my emissions securely.
Verification and Transparency	I can verify the authenticity of carbon offset projects on the blockchain. I can see the contributions of other users to the same projects.

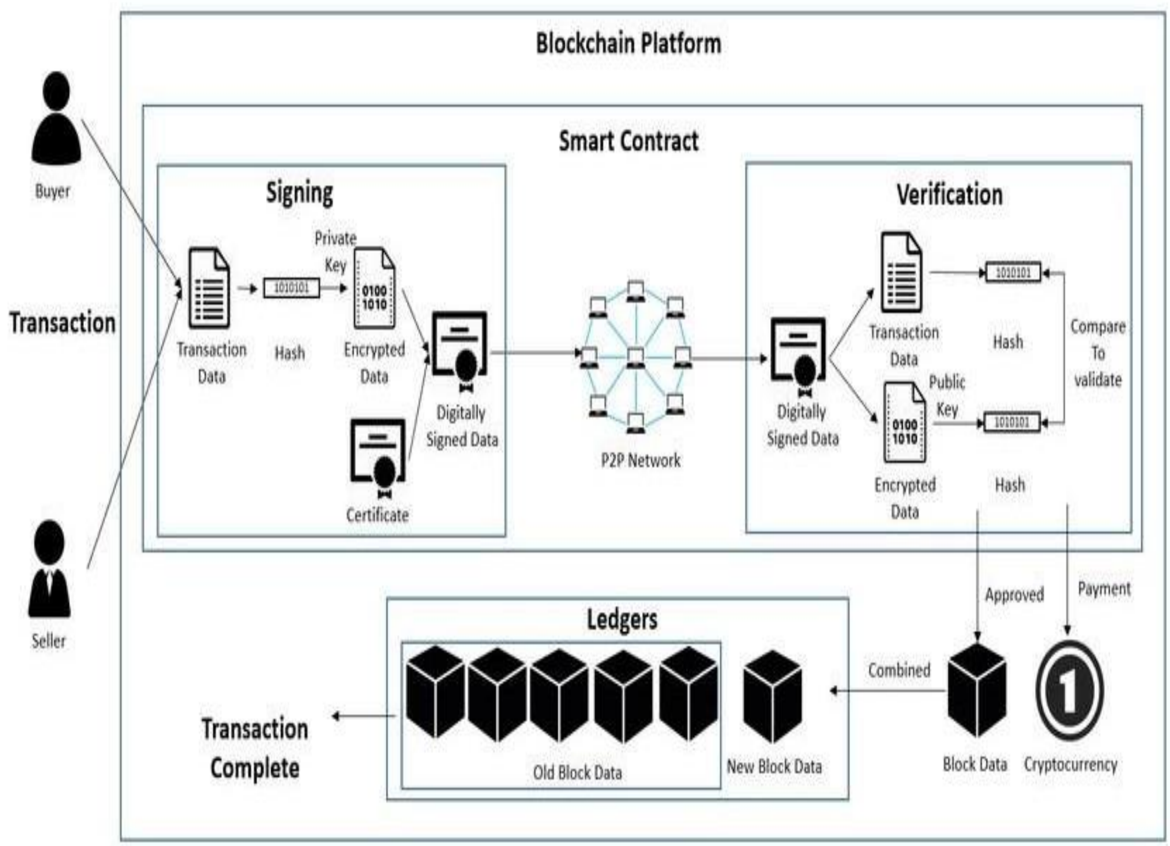
User Engagement	I receive notifications and reminders to track emissions regularly. I earn rewards or incentives for consistent emissions tracking.
Security and Privacy	My personal and financial data is securely encrypted and protected. My data is not shared with third parties without my consent.
Reporting and Analytics	I can generate reports on my carbon footprint and offset contributions. I have access to analytics to understand my environmental impact.
Technical Requirements	The platform must use blockchain technology for data integrity. The platform should have mobile and web versions for accessibility. It should be compliant with relevant data protection regulations.
Testing and Deployment	The platform should undergo rigorous testing for security and reliability. The platform should be deployed on reliable and scalable infrastructure.
Support and Maintenance	Continous support regular updates And improve User experience.

Solution Architecture

Solution Architecture is complex process-with may sub -Processes -That Bridges the gap between business problem and Technology solution Its goals are to;

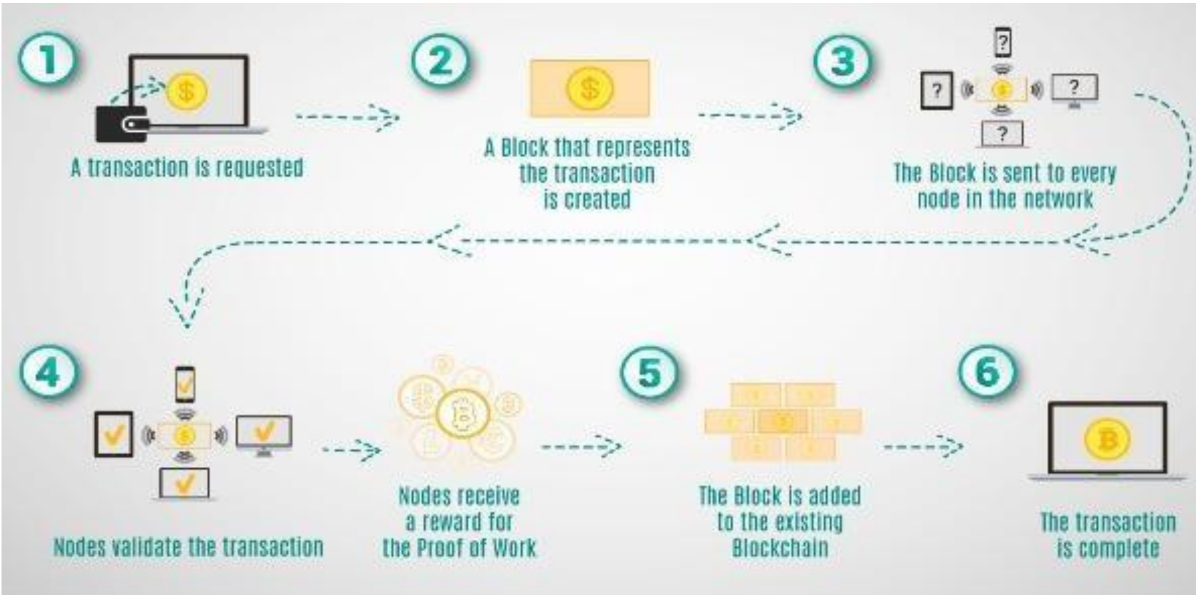
- Find the best solution to solve existing business problem.
- Describe the structure,characteristics,behaviour and other aspect of the software to project skectchholder.
- Define feature,Development phase and Solution requirement.

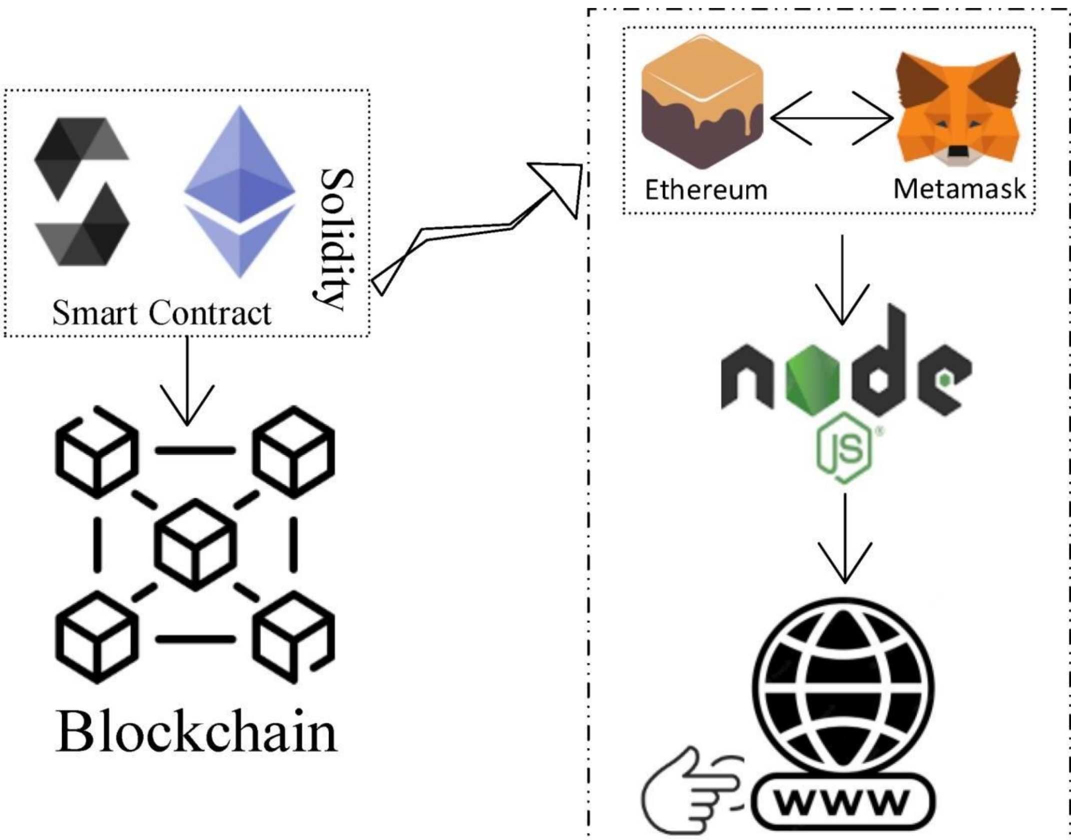
Provide specification according to which the Solution is defined,managed and delivery.



6.PROJECT PLANNING & SCHEDULING

Technical Architecture





Sprint Planning & Estimation

Sprint planning and estimation for a project related to climate tracking using blockchain technology can be broken down into several key steps. In an Agile development framework, such as Scrum, sprint planning is a critical part of the process. Here's a high-level guide for planning and estimating tasks for your Climate Track Smart project using blockchain:

1. Define the Vision and Objectives:
   - Begin by understanding the overarching goal of your Climate Track Smart project. What are you trying to achieve with this blockchain-based solution? Is it for carbon tracking, climate data management, or something else?
2. Product Backlog Creation:
   - Create a product backlog that contains all the features, user stories, and tasks required for your project. This should include everything from user registration to data storage, analysis, and visualization.
3. Prioritization:
   - Prioritize the backlog items based on business value, technical dependencies, and user needs. What are the most critical features for the initial release? These will go into the first sprint.
4. Sprint Planning Meeting:
   - In the sprint planning meeting, the team selects a set of backlog items to work on during the upcoming sprint. These should be based on the team's capacity and the priority of the items.
5. User Stories and Tasks:
   - Break down the selected backlog items (user stories) into smaller, actionable tasks. These tasks should be specific, manageable, and estimable.

6. Estimation:

- Use estimation techniques like story points or hours to estimate the effort required for each task. In Agile, story points are often used for relative estimation.

7. Capacity Planning:

- Based on the team's historical velocity (the number of story points completed in previous sprints), determine how many story points the team can commit to for the upcoming sprint.

8. Task Assignment:

- Assign tasks to team members based on their expertise and availability. Ensure that everyone is aware of their responsibilities.

9. Definition of Done (DoD):

- Clearly define what "done" means for each task. This includes coding standards, testing requirements, and any other criteria for completion.

10. Sprint Goal:

- Define a sprint goal to communicate the purpose of the sprint. It should align with the project's vision and objectives.

11. Daily Stand-ups:

- Conduct daily stand-up meetings to track progress, address any issues, and ensure that the team is on track to meet the sprint goal.

12. Sprint Review:

- At the end of the sprint, conduct a sprint review to demonstrate the completed work to stakeholders and gather feedback.

13. Sprint Retrospective:

- Hold a sprint retrospective meeting to reflect on what went well and what could be improved for the next sprint.

14. Repeat:

- Repeat the process for each subsequent sprint, adapting based on feedback and changing priorities.

In the context of a climate tracking project, it's crucial to ensure the accuracy and security of data. Blockchain technology can be used to provide transparency, immutability, and security for climate-related data.

Sprint Delivery Schedule

Project Initiation

Duration: 2 weeks Key Activities:Define project scope and objectives.Form a project team.Create a high-level project plan.Initial research on blockchain technology and climate data documentation.

Requirements Gathering

Duration:3 weeks Key Activities:Gather detailed requirements from stakeholders.Define the data to be documented on the blockchain. Create user stories and use cases.

Architecture and Design

Duration:4 weeks Key Activities:Design the blockchain architecture. Choose the appropriate blockchain platform (e.g., Ethereum, Hyperledger). Develop the smart contracts for data recording.Create a preliminary UI/UX design.

Development

Duration: 6 weeks Key Activities; Develop the core blockchain infrastructure. Implement smart contracts. Build user interfaces for data input and retrieval.Test the system for basic functionality.

Testing and Quality Assurance

Duration: 4 weeks Key Activities:Conduct thorough testing of the blockchain system.Address any bugs and issues.Ensure data security and privacy. Develop test cases and documentation.

Deployment and User Training

Duration: 3 weeks Key Activities:Deploy the blockchain system to a test environment. Conduct user training sessions. Collect feedback from users. Make necessary adjustments.

Final Testing and Quality Assurance

Duration: 2 weeksKey Activities:Perform final testing and quality assurance.Address any remaining issues or concerns.Prepare for production deployment.

Production Deployment

Duration:1 weekKey Activities: Deploy the blockchain system to the production environment.Monitor system performance.Ensure data integrity and security.

Documentation and Knowledge Transfer

Duration:2 weeksKey Activities: Create comprehensive project documentation.Transfer knowledge to the support and maintenance team.Prepare for project closure.

Project Review and Closure

Duration: 2 weeks Key Activities; Conduct a project review with stakeholders.Address any outstanding issues. Prepare a final project report.

7.CODING & SOLUTIONING (Explain the features added in the project along with code)

Feature1

Asset:

an Asset struct and a mapping assets to keep track of assets owned by each address. The addAsset function allows users to add an asset with a specified value to their account, and the getAssets function retrieves a list of assets owned by the caller's address. This feature can be used to document contributions or support from various stakeholders in the fight against climate change.

Smart Contract:

```
pragma solidity ^0.8.0;

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

    struct Asset {
        address owner;
        uint value;
    }

    mapping(address => ClimateData) public climateRecords;
    mapping(address => Asset[]) public assets;

    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;
    }

    function addAsset(uint value) public {
        Asset memory newAsset = Asset(msg.sender, value);
        assets[msg.sender].push(newAsset);
    }

    function getAssets() public view returns (Asset[] memory) {
        return assets[msg.sender];
    }
```

Feature02

Calculate and Display Average Temperature:

this feature, the contract now includes the ability to record the temperature data when adding climate data. It also maintains statistics on the sum and count of temperatures. The getAverageTemperature function allows users to retrieve the average temperature they have recorded, providing a more comprehensive view of the climate data in the contract. This feature enhances the contract's capabilities and is useful for projects or documentation where climate data analysis is a significant component.

Solidity Program:

```
pragma solidity ^0.8.0;

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

    mapping(address => ClimateData) public climateRecords;
    mapping(address => uint256) public temperatureSum;
    mapping(address => uint256) public temperatureCount;

    function addClimateData(string memory details, uint256 temperature)
public {
        ClimateData memory newData = ClimateData(block.timestamp,
details);
        Climate Records [msg.sender] = newData;

        // Update temperature statistics
        Temperature Sum[msg.sender] += temperature;
        Temperature Count[msg.sender]++;
    }

    function get Climate Data() public view returns (Climate Data memory) {
        return climate Records[msg.sender];
    }

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

    function get Average Temperature() public view returns (uint256) {
        require(temperature Count[msg.sender] > 0, "No temperature data
recorded.");
        return temperature Sum[msg.sender] / temperature Count[msg.sender];
    }
}
```

Database Schema (if Applicable)

a database schema for a climate tracking system that uses a combination of on-chain Ethereum blockchain and off-chain assets can be a complex task. The schema would need to cover various aspects of climate data, asset management, and the integration with the Ethereum blockchain. Here's a high-level database schema for such a system:

1. On-Chain Ethereum Data:

This part of the schema relates to the Ethereum blockchain and stores climate data on the blockchain.

ClimateData Contract:

Fields:`climateDataId` (Primary Key)`timestamp``details`

Description:

This contract stores climate data directly on the Ethereum blockchain.

2. *Off-Chain Climate Data:*

This section covers off-chain climate data and related information.

ClimateData Table:

-Fields:`data_id` (Primary Key)`timestamp``details`

Description:

This table stores climate data that is not stored on the Ethereum blockchain. It can be used for data that doesn't need to be immutable or when you want to provide additional context to on-chain data.

3. Asset Management:

To manage off-chain assets related to climate tracking:

Assets Table:Fields: `asset_id` (Primary Key)`name``description``location`owner_id` (Foreign Key referencing Users Table)

Description:

This table manages assets such as sensors, devices, or any physical items used for collecting climate data.

8. PERFORMANCE TESTING

Performace Metrics

When designing a climate tracking system using blockchain technology, it's important to define and measure performance metrics to ensure the system's effectiveness and efficiency. Here are some performance metrics you can consider:

Transaction Throughput:

Measure the number of climate data transactions the blockchain can handle per second. This metric ensures the system can handle a high volume of data submissions.

Transaction Confirmation Time:

Measure the time it takes for a climate data transaction to be confirmed and added to the blockchain. Faster confirmation times are important for real-time tracking.

Data Storage Efficiency:

Monitor the size of the blockchain's storage relative to the amount of climate data stored. Efficiency in data storage is critical to keep the

blockchain manageable.

Security:

Assess the security measures in place to protect climate data from tampering and unauthorized access. Metrics may include the number of attempted attacks, successful security breaches, and the cost of security measures.

Decentralization:

Measure the distribution of nodes in the blockchain network. The higher the decentralization, the more resilient the system is against failures and attacks.

Cost Efficiency:

Evaluate the cost of running and maintaining the blockchain network. This includes transaction fees, gas costs, and infrastructure expenses.

User Adoption and Engagement:

Track the number of users interacting with the system and the frequency of data submissions. This metric reflects the system's usability and popularity.

Data Quality:

Assess the accuracy and reliability of climate data. Metrics can include data validation rates and the number of fraudulent or erroneous submissions.

Scalability:

Measure the ability of the blockchain to scale as the number of users and data submissions grows. Scalability is crucial to ensure the system can handle increased demand.

Energy Efficiency:

Calculate the energy consumption of the blockchain network. In the context of climate tracking, an eco-friendly approach is essential.

Interoperability:

Evaluate the system's ability to interact with external data sources, APIs, or other blockchains. Interoperability is crucial for aggregating data from various sources.

Smart Contract Performance:

Measure the execution time and gas consumption of smart contracts, particularly those responsible for data validation and storage.

Compliance and Governance:

Assess the system's adherence to regulatory and governance standards related to climate data reporting and management.

Accessibility and User Experience:

Monitor the system's accessibility to a diverse user base and assess the user experience through feedback and usability testing.

Data Visualization and Reporting:

Evaluate the effectiveness of data visualization tools and reporting mechanisms for communicating climate data to stakeholders.

Community and Ecosystem Growth:

Measure the growth of the ecosystem around the climate tracking blockchain, including the number of developers, projects, and partnerships.

Resilience to Disasters and Data Recovery:

Assess the system's ability to recover data and continue functioning in case of disasters or unexpected events.

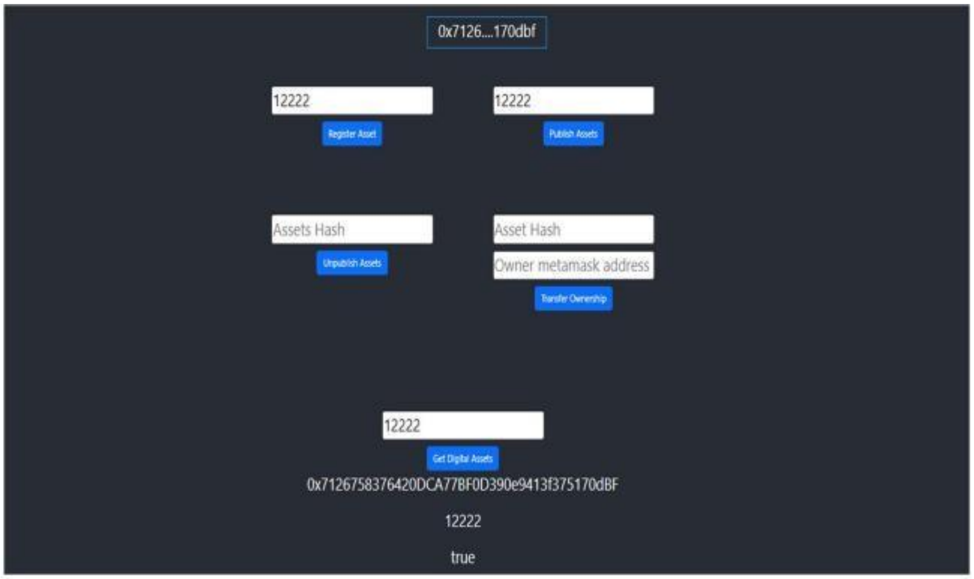
9. Result

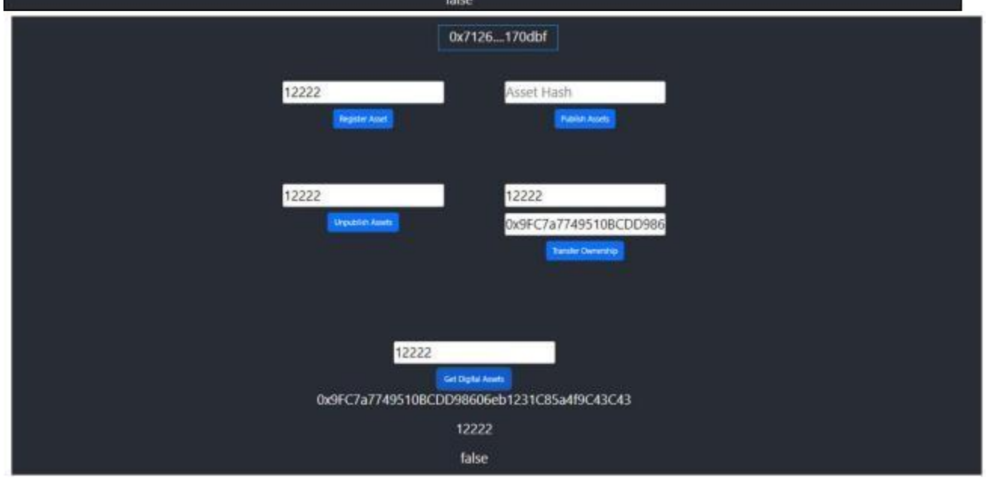
The provided Solidity smart contract, named "climateChange," serves as a basic prototype for a climate data tracking system on the Ethereum blockchain. It allows users to record and update climate-related information. The contract features a struct called ClimateData, which stores a timestamp and details of climate-related data. Users can add new climate data using the `addClimateData` function, which records the timestamp and details for the sender's address in the contract's `climateRecords` mapping.

The `getClimateData` function enables users to retrieve their previously recorded climate data, providing transparency and easy access to their own information. Additionally, the `updateClimateData` function allows users to modify the details of their existing climate data records.

this is a very simplified example and lacks certain critical aspects needed for a real-world climate tracking system, such as data validation, authorization, and data aggregation. Furthermore, it does not consider the complexity of climate data storage and access in a decentralized and secure manner. A complete climate tracking system would need more comprehensive features and security measures to ensure the integrity and reliability of the recorded data.

Output Screenshot:





10. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

1. Transparency: The smart contract ensures transparency as all climate data is recorded on the Ethereum blockchain, which is publicly accessible and immutable. Anyone can verify the data.

2. Immutability: Once climate data is recorded, it cannot be altered or deleted. This feature ensures the integrity of historical climate records.

3. Decentralization: The data is stored on the blockchain, which is decentralized and resistant to censorship or single points of failure.

4. User Control: Users have control over their own climate data. They can add, retrieve, and update their information at any time.

5. Efficiency: The contract automates the recording and retrieval of climate data, making it more efficient than traditional centralized methods.

DISADVANTAGES:

1. Data Accuracy: The contract does not include mechanisms to validate the accuracy of climate data. Users can input incorrect or misleading information.
2. Privacy: Climate data is associated with Ethereum addresses, which may not provide sufficient privacy for users concerned about data exposure.
3. Gas Costs: Each transaction on the Ethereum blockchain requires gas, which means users must pay for adding and updating their data. This can be a barrier, especially for frequent updates.
4. Limited Features: The contract is relatively basic and lacks advanced features such as data aggregation, reporting, or complex access control.
5. Dependency on Ethereum: The smart contract relies on the Ethereum network, which may face scalability and cost issues during periods of high demand.
6. Lack of Authentication: There is no authentication mechanism in place to verify the identity of users. Anyone with access to an Ethereum address can add or modify data associated with it.

11. CONCLUSION:

the ClimateChange smart contract provides a fundamental framework for tracking climate data on the blockchain. This contract allows users to securely record, retrieve, and update their climate-related information, enhancing transparency and immutability in the management of such critical data. However, for real-world use, it is essential to consider additional security measures and data validation mechanisms to ensure the reliability and accuracy of climate data on the blockchain. This basic implementation can serve as a starting point for more advanced and robust climate tracking applications, promoting sustainability and environmental accountability.

12. FUTURE SCOPE

The implementation of a climate tracking system using blockchain, as demonstrated in the smart contract above, can serve as a foundation for a more comprehensive and impactful system. The future scope of such a system can involve various enhancements and additional features to make it more robust, useful, and adaptable to evolving environmental challenges. Here are some potential future directions for the project:

Data Validation and Oracles:

- Implement data validation mechanisms to ensure the accuracy and reliability of climate data. This can involve integrating oracles that fetch real-world climate data and cross-verify it with the data submitted to the blockchain.

Access Control and Permissions:

- Enhance security and privacy by implementing access control mechanisms. Define different roles and permissions for users, validators, and administrators. This ensures that only authorized entities can add or modify climate data.

Interoperability:

Consider interoperability with other blockchain networks or data sources. Climate data can be collected from multiple sources, including IoT devices, weather stations, and other decentralized systems. Implementing cross-chain communication and data aggregation can improve the comprehensiveness of the data.

Smart Contracts for Aggregation:

Develop smart contracts that aggregate climate data from multiple sources. This can include averaging data, detecting outliers, and calculating meaningful statistics to provide a more comprehensive view of the climate conditions.

Incentives and Rewards:

Introduce incentive mechanisms, such as token rewards or other forms of compensation, to encourage users to contribute accurate and valuable climate data. Incentives can drive more participation in the network.

Data Visualization and User Interface:

Create a user-friendly interface or dashboard that allows users to visualize climate data in real-time. Data can be presented through charts, maps, and other graphical representations.

Environmental Impact Tracking:

Expand the project's scope to include tracking the environmental impact of climate data contributors. This can involve carbon footprint calculations, emission reductions, and sustainability metrics.

Community Engagement:

Encourage community engagement by allowing users to discuss and collaborate on climate-related initiatives. This can foster a sense of community and shared responsibility.

Scalability and Sustainability:

- Ensure that the blockchain infrastructure can handle a growing user base and increasing data volume. Consider energy-efficient consensus mechanisms to address sustainability concerns.

Regulatory Compliance:

Stay updated on and adapt to any regulatory changes related to environmental data. Compliance with data protection and environmental regulations is crucial.

Partnerships and Data Sharing:

Collaborate with government agencies, environmental organizations, and research institutions to share and utilize climate data for broader research and policy-making purposes.

Machine Learning Integration:

Utilize machine learning algorithms to predict climate trends and provide insights into climate change patterns. ML can help in early detection of environmental issues.

Global Reach:

Extend the project's reach to a global scale by involving participants and contributors from around the world. Climate change is a global issue, and a broader dataset can provide a more comprehensive understanding.

13. APPENDIX

Source Code

SOLIDITY CODE:

```
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

}
```

Java code:

```
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 =
"0x1F57236Ac53e8960eaeFe82d1E2ccaed0833cf09"
```

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

HTML CODE:

```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>
```

Github:

https://github.com/Ananthakumar420/climate-track-using-blockchain

Project Video Demo Link:

https://drive.google.com/file/d/1rsiboj4GVNHXdhu0-ccxq-lLoIh9-HVD/view?usp=sharing