Supply Chain Management

Smart Contract Solution

PHASE I : PROJECT OVERVIEW

1) The different participants of the Business Network: *check Excel Sheet*
2) Role of each Participant: *check Excel Sheet*

3) This blockchain is a federated blockchain. The smart city involves multiple stakeholders, including government entities, private sector participants, and citizens. With definition, a federated blockchain allows these participants to collaborate without one entity having full control, which means that it also controls access, ensures transparency with privacy, time efficient compared to public blockchains (when managing real-time applications like energy trading, waste management, and transportation), and handles large-scale operations while allowing different organizations to manage specific tasks without overwhelming the system.
4) Smart Contracts from each Scenario:

1. Data Sharing and privacy:
    a. ***Secure IoT Integration****:* A smart contract for secure IoT data sharing in a smart city would ensure that IoT devices automatically and securely **log data** onto the blockchain. The contract would **define who can access this data**, **how it can be shared**, and **enforce strict privacy policies**.
        i. (smart contracts can only grant access to specific parties (ex: government bodies, authorized service providers) and can ensure that any access to sensitive data is logged for accountability. The contract would also include conditions to prevent unauthorized access by triggering penalties or alerts)

    b. ***Citizen Participation****:*
        i. Citizens would opt-in or opt-out of data-sharing with specific services (example with healthcare or transportation).
        ii. The contract logs each citizen's consent choices and ensures that their data is shared only with the selected services.
        iii. If a citizen wishes to withdraw consent, the contract would automatically stop further sharing and delete the shared data from services that no longer have consent.
        iv. The contract could also automatically reward or compensate citizens for sharing their data with the city (like earning tokens for participation in city programs or health tracking).

2. Real Estate and Land Registry:
    a. **Immutable Property Records**:
        i. Automatically verify the **transfer of ownership** by checking that both parties (buyer and seller) fulfill predefined conditions (example: successful payment of funds).

   ii. Create an **immutable digital title** that cannot be tampered with after it is added to the blockchain.
   iii. Verify that all **legal documents** (like contracts, tax records, or verification of ownership) are in place before completing the transaction.
   iv. Once these <u>conditions are met</u>, the contract <u>executes the transfer of ownership</u>, making it transparent and auditable, reducing the risk of fraud.
   v. The contract could also include a **dispute resolution** process to handle any conflicts related to ownership.

b. **Rental Agreements**:
   i. **Rent Payment Automation**: It could include a condition that triggers the release of rental payments when tenants fulfill payment obligations (e.g., monthly rent). The contract might check if the payment was made on time and release funds to the landlord once verified.
   ii. **Maintenance Responsibilities**: The contract could track and automatically execute maintenance tasks. If for example a tenant reports a maintenance issue, the contract could automatically initiate the request and ensure that the service provider completes the work.
   iii. **Penalties for Late Payment**: If a tenant misses a rent payment or violates the contract, the smart contract could trigger penalties, such as late fees, or restrict access to services (example heating, water).
   iv. The contract would also enforce renewal clauses (e.g., if the tenant decides to stay after the contract period ends, the contract could automatically renew unless either party opts out).

3. Public Safety and Security:
   a. **Crime Prevention**:
      i. Automatically log evidence, such as **surveillance footage** or sensor data, onto the blockchain.
      ii. Enforce **access control**: Only law enforcement or authorized personnel would be allowed to view or extract data, based on predefined conditions.
      iii. **Transparency**: Each access request would be recorded on the blockchain, creating an immutable record of who accessed the data and when.
      iv. If someone attempts to alter or tamper with the data, the smart contract could trigger an alert or initiate legal action.
   b. **Emergency Response Coordination**:
      i. Monitor real-time emergency situations, such as the location of incidents and the status of available resources (like ambulances, fire trucks).
      ii. Once an emergency is reported, the smart contract could automatically **dispatch the appropriate services** based on predefined conditions (nearest available unit, severity of the event).
      iii. It could also share **real-time updates** between agencies and ensure that all parties have the information needed to act swiftly, improving response times and minimizing confusion.

4. Urban Planning and Governance:
    a. **Transparent City Budgets**:
        i. **Log transactions** (like fund allocation, expenditures) on the blockchain, ensuring every financial decision is transparent and auditable.
        ii. Automatically enforce budgetary **constraints**: The contract would only allow the transfer of funds to approved departments or projects, based on the predefined city budget.
        iii. **Track and report**: The contract could provide real-time access to citizens to see how public funds are spent and ensure accountability by automatically publishing transaction reports.
        iv. Citizens could even propose budget changes or vote on specific expenditure items using **decentralized governance**, with the contract executing changes once a majority is reached.

    b. **Decentralized Urban Governance**:
        i. Enable **voting** on key city issues (like infrastructure projects, zoning laws) where each citizen can cast votes securely via the blockchain.
        ii. Ensure **voter anonymity** and **integrity** of the voting process, so votes cannot be tampered with.
        iii. Automatically **implement decisions** once the voting process is completed and a threshold is met (for ex 51% or 70% of votes).
        iv. Provide a **transparent audit trail** for all voting activity, ensuring citizens can trust the process and see the outcomes of their participation.

5. Waste Management:
    a. **Monitor waste collection** events, ensuring that waste management companies are collecting waste on time and to the correct locations.
    b. **Verify** that the waste is disposed of properly, ensuring that recyclables are sent to recycling facilities and hazardous waste is safely handled.
    c. Automatically **penalize companies** for improper disposal (e.g., illegal dumping) or reward them for efficient recycling practices.
    d. **Track recycling rates**: If citizens participate in recycling programs, the contract could automatically reward them with incentives, such as tokens or discounts on services.
6. Energy Management
    a. **Decentralized Energy Trading:**
        i. Smart contracts in a decentralized energy trading scenario would ensure that households with renewable energy (e.g., solar power) can sell excess energy to others.
        ii. Once an agreement is made, the contract would automatically execute payment from the buyer to the seller once the energy has been delivered.
        iii. The blockchain would ***record all transactions***, ensuring transparency in energy trading and consumption.

    iv. ***Penalties*** could be automatically triggered if energy quality or transaction conditions are violated.

  b. **Transparency in Energy Distribution:**

    i. The smart contract ensures **transparency** in energy distribution, *tracking the flow of energy from generation to consumption.*

    ii. Each transaction is recorded on the blockchain, including the amount of energy distributed, the consumer, and the price.

    iii. The contract would enable **real-time monitoring and reporting** of energy usage to prevent fraud and inefficiencies.

    iv. Alerts would be triggered if there are any discrepancies in energy usage or **price fluctuations** outside predefined conditions.

    v. Citizens and organizations can access this data, ensuring fairness and trust in the energy distribution system.

7. Supply Chain Management

  Food and Goods Tracking:

    i. A smart contract would ensure that food products <u>are tracked from the farm to the consumer</u>, ensuring transparency in the supply chain.

    ii. The contract would **log the product details**, including its origin, journey, and quality checks, onto the blockchain, creating an immutable record.

    iii. It would include ***conditions to ensure that the product passes all necessary quality checks*** before it moves to the next stage in the supply chain.

    iv. Payments would be automatically processed once the goods reach the next destination (e.g., retailer), and the blockchain would ensure that all transactions are transparent and secure.

    v. If any *discrepancy or tampering is detected, the contract could automatically trigger alerts*, ensuring the product's integrity throughout its journey.

8. Smart Contracts for Public Services

  a. **Automated Contract Execution (e.g., Waste Management):**

    i. A smart contract would automatically trigger actions related to waste management **based on data inputs from sensors or external systems** (e.g., full waste bins or scheduled collection times).

    ii. The contract would verify that the waste collection service provider is authorized to perform the task.

    iii. It would automatically create and send a payment from the city or service user to the service provider once the collection is completed as per the agreement.

    iv. The contract would also track the timeliness of waste collection and penalize the service provider if they fail to meet agreed-upon deadlines.

    v. If the waste is disposed of incorrectly (e.g., non-recyclable materials in recycling bins), the contract would *<u>trigger a penalty or fine</u>* for the service provider.

b. **Public Procurement:**
    i. Smart contracts would ensure transparency and fairness in public procurement processes.
    ii. The contract would **verify that all bidders meet the predefined criteria** (e.g., qualifications, budget limits, compliance with regulations).
    iii. Once bids are submitted, the smart contract would **automatically evaluate** them based on predefined conditions (e.g., best price, quickest delivery time).
    iv. The winning bid would be selected and the contract would execute the formal agreement between the city and the provider.
    v. <u>The contract would ensure that funds are only released when predefined milestones</u> (e.g., project completion, delivery of goods/services) <u>are met</u>, ensuring accountability in public spending.

9. Identity Management
    a. **Decentralized Digital Identity:**
        i. Smart contracts would allow citizens to **control their digital identities** within the smart city ecosystem.
        ii. The contract would ensure that a citizen’s identity is stored securely on the blockchain, granting access to city services (e.g., healthcare, education, and banking) based on verified identity attributes.
        iii. **Only authorized parties** (e.g., city services, service providers) would be allowed to access the citizen's identity information, and all access would be logged on the blockchain for accountability and transparency.
        iv. Citizens could modify or revoke their access permissions at any time, with the smart contract ensuring that any unauthorized attempts to access their identity are blocked.
        v. The contract would ensure that only the minimal necessary data (e.g., name, age, address) is shared with services, <u>*ensuring privacy*</u> and <u>*compliance with data protection laws.*</u>
    b. **Secure Voting:**
        i. Smart contracts would create a **secure, transparent, and immutable** voting system for citizens to participate in city governance.
        ii. Each eligible voter would have a unique, verified digital identity stored on the blockchain to prevent fraud.
        iii. The contract would automatically ensure that each voter can cast only one vote, maintaining the integrity of the voting process.
        iv. Once votes are cast, the contract would automatically tally the results, ensuring no tampering or manipulation.
        v. After the voting period ends, the results would be published transparently, with a full audit trail recorded on the blockchain for public scrutiny.
        vi. The contract would also <u>ensure voter anonymity</u> and prevent any unauthorized access to voting records.

10. Transportation and Mobility
    a. **Autonomous Vehicles:**

    i. Smart contracts would enable seamless interaction between autonomous vehicles and city infrastructure, ensuring smooth traffic flow and regulatory compliance.
    ii. The contract would **verify that the vehicle meets all safety and regulatory** standards before it operates on the city's roads.
    iii. It would automatically communicate with traffic lights and city infrastructure to ensure optimal travel, such as triggering green lights when an autonomous vehicle is approaching an intersection.
    iv. Payments <u>for tolls, fines, or services</u> (e.g., parking) would be automatically processed through the smart contract, ensuring an efficient and cashless system.
    v. The contract would track the vehicle's performance and route, logging the data on the blockchain to ensure accountability and transparency.
    vi. If the vehicle *operates in violation of any rules* (e.g., exceeding speed limits, unauthorized areas), the contract would automatically trigger fines or penalties.

b. **Carpooling and Ride-Sharing:**
    i. Smart contracts would govern the interactions between drivers and passengers in a *decentralized carpooling or ride-sharing platform*.
    ii. The contract would <u>match drivers and passengers based on location, timing, and preferences.</u>
    iii. It would automatically calculate and agree on the fare between the rider and driver before the ride begins.
    iv. The contract would **automatically release payment from the rider to the driver once the trip is completed**, ensuring both parties fulfill their obligations.
    v. If there are any disputes, the contract <u>would initiate a dispute resolution</u> process, such as reviewing driver and passenger ratings or investigating the complaint.
    vi. The contract would **verify that both the driver and passenger meet safety criteria, such as verified identities**, before proceeding with the trip.

5) Assets of the Blockchain Application:

The following assets can be identified:

1. Identity Assets

- **Digital Identity**: This represents the unique, verified identity of citizens within the smart city ecosystem, used to grant access to various services like healthcare, education, and transportation.

- **Identity Attributes**: Includes personal details (e.g., name, age, address, health status) that are securely stored and shared with authorized parties via the blockchain.
- **Voter Identity**: An identity asset tied to citizens' right to vote in city governance, ensuring secure and transparent voting processes.

2. Transaction Assets

- **Financial Transactions**: These assets represent any form of monetary exchange within the smart city, including payments for services like waste management, energy trading, public transport fares, and real estate transactions.
- **Public Procurement Transactions**: Includes bidding, contract payments, and procurement-related exchanges between the government and service providers or suppliers.
- **Rental Payment Transactions**: For rental agreements, where payments are tracked, verified, and executed automatically through smart contracts.

3. Real Estate and Property Assets

- **Property Ownership Records**: Immutable digital records of property titles and ownership that are verified and transferred through smart contracts, ensuring transparency and security in real estate transactions.
- **Land Registry**: This asset represents the ownership and transfer of land or property within the smart city, ensuring that all property transactions are securely recorded and auditable.
- **Rental Agreements**: Smart contracts governing the rental agreements, detailing the terms, payment conditions, and responsibilities of tenants and landlords.

4. Energy Assets

- **Energy Supply and Consumption Records**: These assets track energy production, distribution, and consumption, ensuring transparency in energy management within the city’s infrastructure.
- **Decentralized Energy Trading Units**: Represent the tokens or credits issued for peer-to-peer energy trading, facilitating transactions between households, businesses, and other energy producers.

5. Service and Infrastructure Assets

- **Waste Management Records**: Includes data related to waste collection, recycling, and disposal, tracked and executed through smart contracts.
- **Transportation Assets**: Represents data related to transportation services, including vehicles (e.g., autonomous vehicles), traffic systems, and public transportation services. These assets ensure optimal routing, fare management, and accountability in the transportation system.

- **IoT Device Data**: Includes data from various IoT devices used in the smart city, such as sensors, cameras, or environmental monitoring devices, with access and data-sharing governed by smart contracts.

6. Data Assets

- **IoT Data**: Data from smart devices and sensors that is logged and shared securely on the blockchain, ensuring integrity, transparency, and privacy for city management services.
- **Citizen Data**: This includes personal and behavioral data shared by citizens, with clear consent protocols governed by smart contracts to ensure privacy and proper use.
- **Real-Time Event Data**: Data generated by city services (e.g., emergency response systems, traffic control, or utility monitoring) that is logged and acted upon by smart contracts in real time.**7. Governance and Decision-Making Assets**
- **Voting Assets**: Represent citizens' votes in decentralized urban governance, ensuring secure, transparent, and auditable voting records.
- **Budget and Financial Allocations**: Financial assets related to the allocation and use of public funds, tracked and managed using smart contracts to ensure transparency in government spending.
- **Decentralized Proposals and Polls**: Assets that facilitate decentralized decision-making, where citizens can propose or vote on city initiatives, ensuring collective participation in urban planning and governance.

PHASE II: Development of the Blockchain Solution

Why Ethereum-Based Blockchain Appealed to us

Ethereum’s support for smart contracts made it an appealing choice for my federated blockchain solution. Since I have chosen to work on the Supply Chain Scenario, the information related to this scenario will be public since we are aiming for transparency in the chain; from the original supplier to the last consumer. The **developer-friendly ecosystem of Ethereum** was another major factor in my decision. I used **Remix**, a web-based Integrated Development Environment (IDE), to develop my project. Remix provided a simple and efficient platform to code, test, and deploy smart contracts without requiring extensive setup. Its built-in tools, including a Solidity compiler and debugger, allowed me to focus on implementing and refining the functionality of my blockchain without technical hurdles.

I did not choose Hyperledger because it is complex in its implementation. My project required a federated blockchain for a smart city involving diverse, independent stakeholders where decentralization, selective transparency, and flexibility were essential and so ETH made sense.

Choice of Scenario: *Supply Chain Management*

We have developed 3 contracts that would be implemented in the supply chain management process. ***<u>You will find the code in folder "project/blockchain-solution"</u>***

The three smart contracts developed for the supply chain management system aim to streamline and automate key functions within the supply chain. These contracts include:

1. **Provenance Tracking Contract** - Ensures the origin and traceability of goods.
2. **Shipment Tracking Contract** - Manages the shipment process and automates payments based on delivery confirmation.
3. **Reputation Management Contract** - Tracks and updates supplier reputation scores.

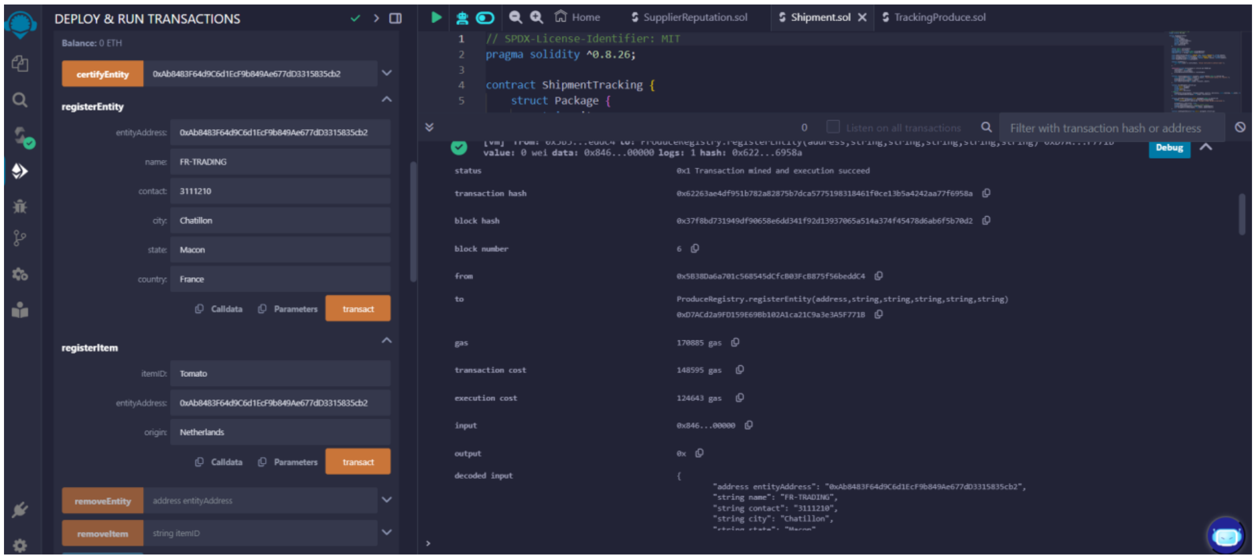
Each contract has been enhanced with new features to improve functionality, dynamic tracking, and payment systems, making the entire supply chain process more efficient and transparent.

1. Provenance Tracking Contract

The **ProvenanceRegistry** contract ensures goods' provenance is recorded immutably on the blockchain, providing transparency into their origin and certification. The contract has been enhanced to include stock management, price tracking, and more detailed permissions for entities.

Key Functions:

- **Entity Registration:** Allows the admin to register new entities (e.g., suppliers).



Balance: 0 ETH

certifyEntity: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2

registerEntity
entityAddress: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2
name: FR-TRADING
contact: 3111210
city: Chatillon
state: Macon
country: France

registerItem
itemID: Tomato
entityAddress: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2
origin: Netherlands

removeEntity: address entityAddress
removeItem: string itemID

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.26;

contract ShipmentTracking {
    struct Package {
```

```
value: 0 wei data: 0x846...00000 logs: 1 hash: 0x622...6958a
status              0x1 Transaction mined and execution succeed
transaction hash    0x62263ae4df951b782a82875b7dca5775198318461f0ce13b5a4242aa77f6958a
block hash          0x37f8bd731949df90658e6dd341f92d13937065a514a374f45478d6ab6f5b70d2
block number        6
from                0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                  ProduceRegistry.registerEntity(address,string,string,string,string,string)
                    0xD7ACd2a9FD159E69Bb102A1ca21C9a3e3A5F771B
gas                 170885 gas
transaction cost    148595 gas
execution cost      124643 gas
input               0x846...00000
output              0x
decoded input       {
                        "address entityAddress": "0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2",
                        "string name": "FR-TRADING",
                        "string contact": "3111210",
                        "string city": "Chatillon",
```

- **Certification:** Admin can certify entities to confirm their legitimacy.

- **Item Registration:** Tracks individual goods and their origin, along with their available units and price per unit.

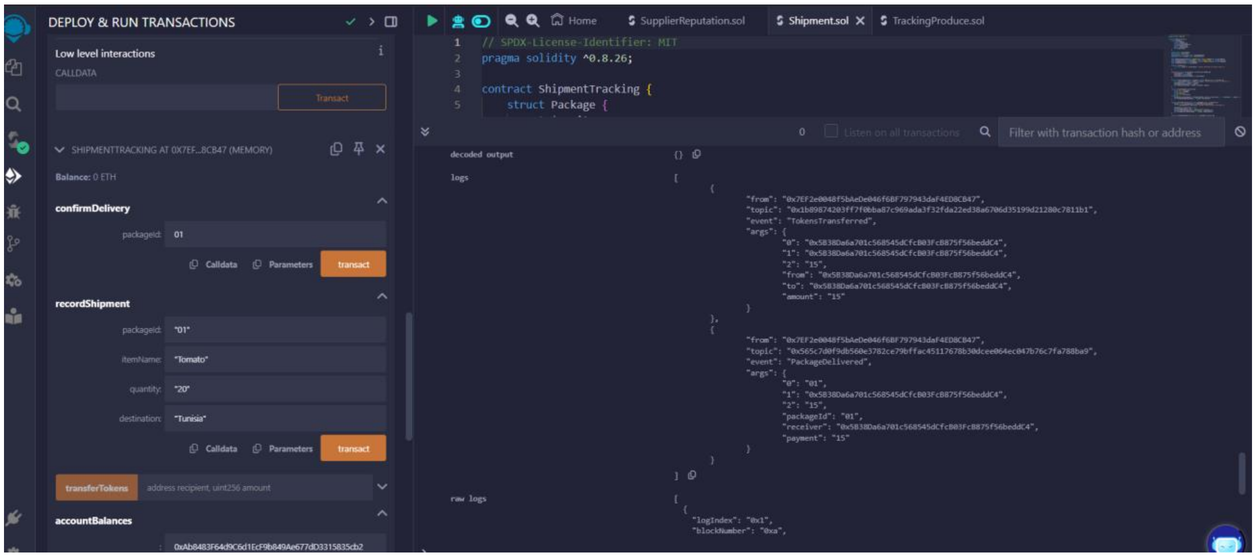
- **Stock Updates:** Items can have their stock and price updated as needed.
- **Shipment Integration:** Once an item is shipped, the available units are reduced accordingly.
- **Audit Log:** Each action (e.g., registration, updates) is logged for transparency.
- **Audit Trail for Changes:** Tracks every modification made to entities and items, ensuring accountability.
- **Item Stock Management:** Adds functionality to manage item stock (available units) and updates prices dynamically.
- **Permissions Control:** Only certified entities are allowed to register and update items.
- **Item Decrease upon Shipment:** As items are shipped and delivered, their stock is automatically reduced, ensuring inventory control.

2. Shipment Tracking Contract

The **ShipmentTracking** contract handles the shipment of goods, including dynamic cost calculation and payment automation. It tracks the entire shipping process from dispatch to delivery and automates payments based on the successful receipt of goods.

Key Functions:

- **Shipment Record:** Logs shipment details, including item name, quantity, destination, and price.
- **Payment and Tokens Transfer:** Automates token transfer to the sender after delivery is confirmed.
- **Dynamic Shipping Costs:** Shipping costs are dynamically calculated based on the destination.
- **Unit Deduction on Delivery:** The shipment's quantity is deducted from the origin inventory once the delivery is confirmed.
- **Shipping Cost Calculation:** A dynamic shipping cost calculation is introduced, allowing flexibility based on destination or other variables.
- **Payment Automation:** Upon successful delivery confirmation, the payment is automatically transferred to the sender, ensuring fast and secure transactions.

- 

Balance: 0 ETH

confirmDelivery
packageId: 01

recordShipment
packageId: "01"
itemName: "Tomato"
quantity: "20"
destination: "Tunisia"

transferTokens: address recipient, uint256 amount

accountBalances
: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.26;

contract ShipmentTracking {
    struct Package {
```

```
decoded output                {}

logs                          [
                                  {
                                      "from": "0x7EF2e0048f5bAeDe046f6BF797943daF4ED8CB47",
                                      "topic": "0x1b89874203ff7f0bba87c969ada3f32fda22ed38a6706d35199d21280c7811b1",
                                      "event": "TokensTransferred",
                                      "args": {
                                              "0": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4",
                                              "1": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4",
                                              "2": "15",
                                              "from": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4",
                                              "to": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4",
                                              "amount": "15"
                                      }
                                  },
                                  {
                                      "from": "0x7EF2e0048f5bAeDe046f6BF797943daF4ED8CB47",
                                      "topic": "0x565c7d0f9db560e3782ce79bffac45117678b30dcee064ec047b76c7fa788ba9",
                                      "event": "PackageDelivered",
                                      "args": {
                                              "0": "01",
                                              "1": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4",
                                              "2": "15",
                                              "packageId": "01",
                                              "receiver": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4",
                                              "payment": "15"
                                      }
                                  }
                              ]

raw logs                      [
                                {
                                  "logIndex": "0x1",
                                  "blockNumber": "0xa",
```

- **Penalty Mechanism:** Delays in delivery can result in penalties or compensation based on the contract's conditions.
- **Multi-Item Shipments:** Allows multiple items to be shipped together, updating the inventory of all items in one transaction.

3. Reputation Management Contract

The **SupplierReputation** contract helps track and update supplier reputation scores based on performance feedback. Suppliers are rated, and their reputation scores are updated dynamically to reflect their ongoing performance in the supply chain.

Key Functions:

- **Supplier Registration:** Allows admins to register suppliers with their name, location, and product category.
- **Reputation Updates:** Admin can update reputation scores or rely on buyer feedback to adjust scores.
- **Feedback Mechanism:** Buyers leave feedback that is incorporated into the supplier's reputation score.
- **Supplier Querying:** Reputation can be queried by category, location, or score range to facilitate better decision-making.
- **Dynamic Reputation Scoring:** Reputation scores are updated based on user feedback, allowing for real-time changes reflecting performance.
- **Feedback System:** Buyers can leave feedback, which affects the supplier’s reputation score.
- **Reputation Query by Category:** Suppliers are grouped by categories, and their reputations can be queried based on location, category, or score range.
- **Auditing:** Reputation changes are auditable to ensure transparency.

Phase 3: Blockchain Evaluation

Phase 4: Towards a Nested Blockchain

1. Choosing a Nested Blockchain to Work On:

We chose **Polkadot** as our nested blockchain platform.

Polkadot is a blockchain network designed with a **multi-chain architecture**. It allows multiple independent blockchains, called **parachains**, to connect to a central blockchain called the **Relay Chain**. The Relay Chain is responsible for providing security and consensus, while the parachains handle specialized tasks independently.

We'll use Polkadot as our nested blockchain platform because of its robust framework for creating interconnected blockchains that can each focus on different tasks.

2. Nested Blockchain Description:

Our nested blockchain solution will have the following components:

Parent Chain (Relay Chain):

The **Relay Chain** will serve as the central chain providing:

- **Security**: Ensures all parachains are secure by using the shared security model.
- **Governance**: Coordinates the overall governance of the network and makes decisions on updates, upgrades, and rule changes.
- **Consensus**: Validates transactions and ensures the integrity of the system.

Child Chains (Parachains):

Each **parachain** will have a specific functionality and can operate independently but still interact with other parachains through the Relay Chain. Here are a few examples of child chains:

1. **Provenance Parachain**: This child chain will handle the **provenance tracking** of goods in a supply chain, tracking their origin, certification, and history.
2. **Shipment Parachain**: This child chain will manage **shipment tracking** and ensure payments are automated once goods are delivered.
3. **Reputation Parachain**: This child chain will track the **reputation scores** of suppliers and other entities within the supply chain, updating scores based on feedback and performance.

Each parachain will operate independently but rely on the Relay Chain for security and coordination.

Nested Blockchain Design:

*Code was provided in folder **“project/nested-blockchain”***

Parent Chain (Relay Chain) Design:

- **Consensus Mechanism**: The Relay Chain will use **Nominated Proof of Stake (NPoS)** to ensure that validators are chosen based on their stake and reputation.
- **Governance Model**: **On-chain governance** will allow token holders to propose and vote on decisions regarding the network’s future upgrades, parachain slots, and protocol adjustments.
- **Cross-Chain Communication**: The Relay Chain will enable **cross-chain messaging** (XCM) between parachains, allowing them to interact and share information, such as when goods are shipped or when a reputation score is updated.

Child Chains (Parachains) Design:

Each parachain will have its own:

- **Consensus**: Can use different consensus mechanisms (like Proof of Authority, Proof of Stake, or others) tailored to its specific needs.
- **Smart Contracts**: The parachains will host smart contracts for specific functionalities, such as provenance tracking, shipment tracking, and reputation management.
- **Cross-Chain Communication**: Parachains will use the Relay Chain to communicate and exchange data or assets. For example, when a shipment is confirmed, the Shipment Parachain will notify the Provenance Parachain to update stock availability.

Data Structure:

- The data on each parachain will be stored in a decentralized manner, with important information such as:
    - Goods' provenance (origin, certification, and transaction history).
    - Shipment details (quantities, destinations, prices, and payment status).
    - Reputation scores based on performance and feedback.

Evaluation and Comparison to the Simple Blockchain:

Nested Blockchain (Polkadot) Evaluation:

- **Scalability**: The nested blockchain model significantly improves scalability because different tasks (e.g., provenance tracking, shipment tracking, reputation management) are offloaded to specialized parachains. This reduces the load on the Relay Chain and avoids congestion on a single blockchain.

- **Security**: The parent Relay Chain ensures security for all parachains through shared security. Each parachain benefits from this without needing to establish its own separate security mechanisms.
- **Customization**: Each parachain can be customized for specific functionalities (e.g., different consensus mechanisms or smart contract structures for each use case), which provides greater flexibility.
- **Cross-Chain Interoperability**: Parachains can communicate with each other through the Relay Chain, allowing them to synchronize and share data.
- **Complexity**: The architecture is more complex than a simple blockchain due to the need for multiple chains and cross-chain communication mechanisms. This requires more sophisticated development and management.

Simple Blockchain Evaluation (Single Blockchain):

- **Scalability**: A simple blockchain (e.g., Ethereum or Bitcoin) handles all tasks within a single chain. As more functions are added (like provenance tracking, shipment tracking, and reputation management), the blockchain can become congested, leading to slower transaction speeds and higher costs.
- **Security**: Security is provided at the blockchain level, but if the blockchain becomes too congested or complex, it can introduce vulnerabilities.
- **Customization**: A simple blockchain offers limited customization because all tasks must be handled within the same network, which can make it less flexible for specialized needs.
- **Cross-Chain Interoperability**: A simple blockchain doesn't support cross-chain communication natively. To interact with other blockchains, complex bridges or third-party solutions would be needed.
- **Simplicity**: It's easier to develop and manage because there is only one blockchain to maintain, but it comes at the cost of scalability and flexibility.

Comparison:

Feature	Nested Blockchain (Polkadot)	Simple Blockchain
Scalability	High: Tasks are offloaded to parachains.	Low: Single chain can get congested.

Security	High: Shared security from Relay Chain.	Moderate: Depends on the chain's consensus.
Customization	High: Each parachain is specialized.	Low: All tasks are handled in one chain.
Cross-Chain Communication	Easy via XCM protocol.	Requires third-party solutions.
Complexity	High: Involves managing multiple chains.	Low: Simple to implement but limited in scope.