**Blockchain and Supply Chain Management System**

This README provides a detailed explanation of the functions and classes implemented in the provided Python code for a Blockchain-based Supply Chain Management System.

**Team Members – (GROUP 37):**

1. Mohammed Asad Bakhsh (2021AAPS2706H)
2. Abhimanyu Thapliyal (2021A3PS2940H)
3. Utkarsh Sinha (2021A3PS2938H)

**Table of Contents :**

1. [**Introduction**](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#introduction)
2. **Objective**
3. [**Classes**](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#classes)

* [Transaction](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#transaction)
* [Block](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#block)
* [Blockchain](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#blockchain)
* [Client](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#client)
* [Distributor](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#distributor)
* [Manufacturer](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#manufacturer)

1. [**Functions**](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#functions)

* [generate\_qr\_code](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#generate_qr_code)
* [add\_transaction](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#add_transaction)
* [mine\_pending\_transactions](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#mine_pending_transactions)
* [print\_blockchain](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#print_blockchain)
* [confirm\_delivery](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#confirm_delivery)
* [user\_menu](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#user_menu)
* [send\_order](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#send_order)
* [get\_order](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#get_order)
* [add\_order](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#add_order)
* [main\_menu](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#main_menu)
* [create\_user](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#create_user)
* [login](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#login)
* [main\_loop](https://chat.openai.com/c/23135ef5-e384-4b33-8177-c9854d68b7fe#main_loop)

1. **Blockchain Features and their codes**
2. **Implementation**
3. **Introduction**

This Python code represents a simplified supply chain management system built on blockchain technology, employing the Proof of Work (PoW) consensus algorithm. It introduces essential classes for transactions, blocks, a blockchain, clients, distributors, and a manufacturer. This system empowers users to seamlessly create and manage orders, efficiently monitor order statuses, and securely add new blocks to the blockchain.

The integration of the PoW consensus algorithm ensures the immutability and transparency of supply chain transactions. Users can navigate the complexities of supply chain management with confidence, utilizing a system that offers a robust and secure framework for their operations.

Explore the README further to delve into the code's structure, functionalities, and how it streamlines supply chain management while leveraging blockchain technology with the PoW consensus algorithm to enhance data integrity and trust.

1. **Objective**

**Project Objective:** The project aims to create a robust blockchain-based Supply Chain Management System. It facilitates transparent and secure product tracking and delivery. The system's objectives include registering clients, distributors, and manufacturers, each contributing a security deposit. A Proof of Work consensus algorithm enhances security. Implementing a Merkle tree for transaction hashing ensures data integrity, and QR codes enable real-time product status tracking. Additionally, the system features a distributor-client transaction confirmation mechanism. It addresses challenges, such as handling disputes where a distributor dispatched a product, but the client denies it or when the distributor hasn't dispatched, but the client claims non-receipt, fostering trust and reliability in the supply chain.

1. **Classes**

* **Transaction**

The Transaction class represents a single transaction within the supply chain. It includes information such as product name, timestamp, manufacturer, distributor, client, status, order ID, and a hash of the transaction. Key functions and attributes:

\_\_init\_\_(self, order\_id, product\_name, status, manufacturer=None, distributor=None, client=None, timestamp=time.ctime()): Initializes a transaction object with the provided parameters.

calculate\_hash(self): Calculates a hash for the transaction using the order ID and timestamp.

\_\_str\_\_(self): Converts the transaction object to a string for easy printing.

* **Block**

The Block class represents a block within the blockchain. Each block contains a list of transactions, a timestamp, a previous block's hash, a nonce, a difficulty level, and its own hash. Key functions and attributes:

\_\_init\_\_(self, timestamp, transactions, prev\_hash, nonce, difficulty): Initializes a block object with the provided parameters.

calculate\_hash(self): Calculates a hash for the block using transaction hashes, timestamp, previous hash, and nonce.

mine(self): Mines the block by finding a hash that meets the required difficulty.

calculate\_merkle\_root(self): Calculates the Merkle root of transactions in the block.

\_\_str\_\_(self): Converts the block object to a string for easy printing.

* **Blockchain**

The Blockchain class represents the entire blockchain. It maintains a chain of blocks, a difficulty level, and a list of pending transactions. Key functions and attributes:

generate\_qr\_code(self, order\_id): Generates a QR code for a specific order ID and its details.

add\_transaction(self, transaction): Adds a transaction to the list of pending transactions.

mine\_pending\_transactions(self): Mines a new block with pending transactions.

print\_blockchain(self): Prints the entire blockchain.

\_\_str\_\_(self): Converts the blockchain object to a string for easy printing.

* **Client**

The Client class represents a client within the supply chain. Clients can place orders, confirm deliveries, and view their orders. Key functions and attributes:

confirm\_delivery(self): Allows clients to confirm the delivery of an order.

user\_menu(self): Displays a menu for clients to interact with the system.

\_\_str\_\_(self): Converts the client object to a string for easy printing.

* **Distributor**

The Distributor class represents a distributor within the supply chain. Distributors can get orders from the manufacturer and send orders to clients. Key functions and attributes:

send\_order(self): Sends an order to a client.

user\_menu(self): Displays a menu for distributors to interact with the system.

\_\_str\_\_(self): Converts the distributor object to a string for easy printing.

* **Manufacturer**

The Manufacturer class represents the manufacturer in the supply chain. Manufacturers can get orders from clients and distribute them to distributors. Key functions and attributes:

get\_order(self, distributor): Gets an order from a distributor.

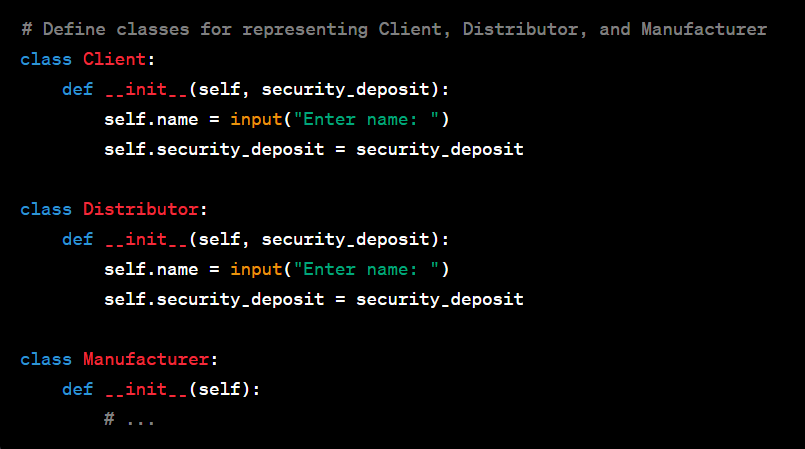
add\_order(self, client): Adds an order to the list of client orders.

\_\_str\_\_(self): Converts the manufacturer object to a string for easy printing.

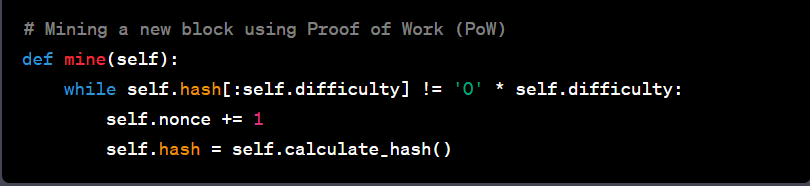
1. **Functions**

* **generate\_qr\_code(self, order\_id**): Generates a QR code for a specific order ID and its details.
* **add\_transaction(self, transaction):** Adds a transaction to the list of pending transactions.
* **mine\_pending\_transactions(self):** Mines a new block with pending transactions.
* **print\_blockchain(self):** Prints the entire blockchain.
* **confirm\_delivery(self):** Allows clients to confirm the delivery of an order.
* **user\_menu(self):** Displays a menu for clients and distributors to interact with the system.
* **send\_order(self):** Sends an order to a client.
* **get\_order(self, distributor):** Gets an order from a distributor.
* **add\_order(self, client):** Adds an order to the list of client orders.
* **main\_menu():** Displays the main menu for the program.
* **create\_user():** Allows the creation of a new user (either distributor or client).
* **login():** Allows user login and interaction with the system.
* **main\_loop():** The main loop of the program that executes user interactions and actions.

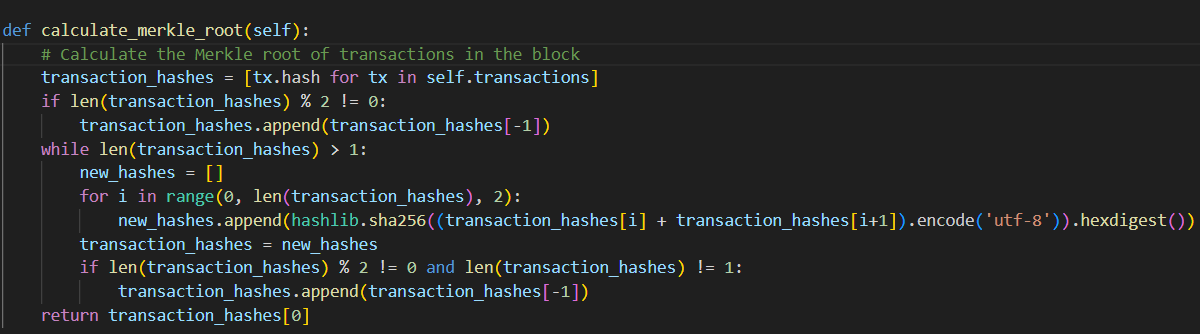
1. **Blockchain Features and their codes**
2. Registering New Clients, Distributors, and a Manufacturer with Security Deposits:



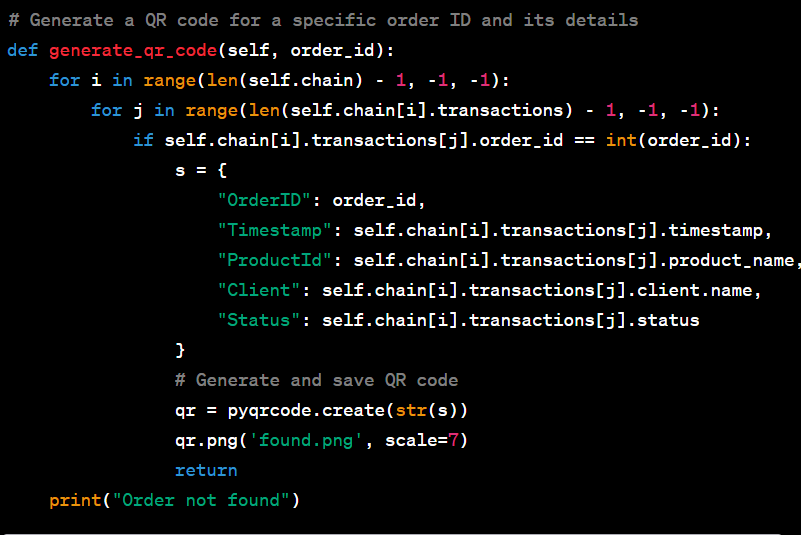
1. Incorporating a Consensus Algorithm (Proof of Work - PoW):



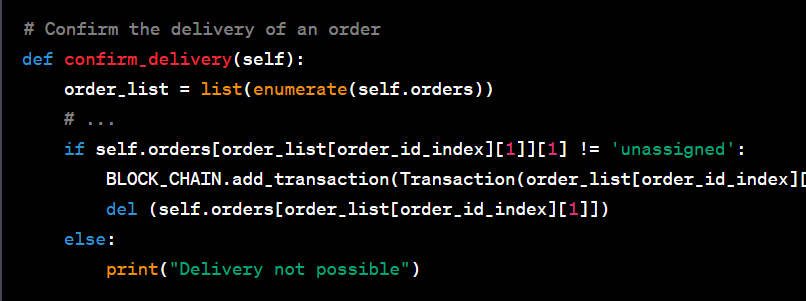
1. Implementing Merkle Tree for Transaction Hash Calculation:

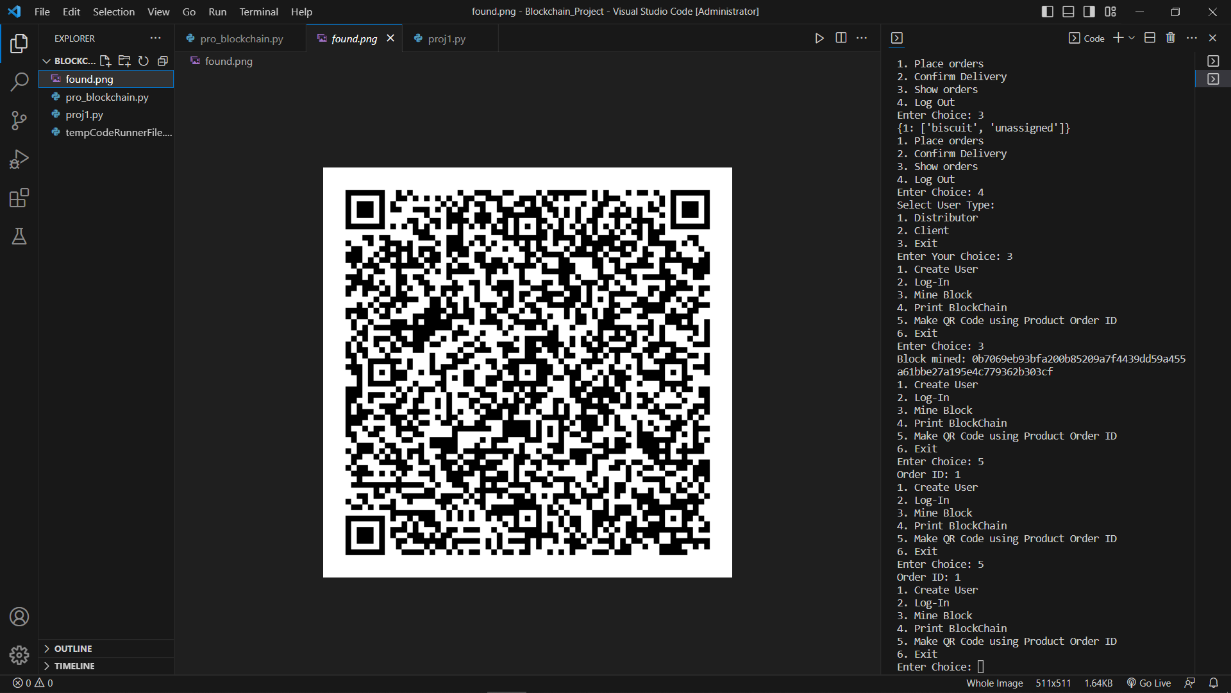


1. Viewing Product Status Using QR Code:



1. Distributor-Client Transaction Confirmation Mechanism:



****

1. **Implementation :**

1. Create User

2. Log-In

3. Mine Block

4. Print BlockChain

5. Make QR Code using Product Order ID

6. Exit

Enter Choice: 1

Select User Type:

1. Distributor

2. Client

Enter Choice: 2

Enter name: Abhimanyu

1. Create User

2. Log-In

3. Mine Block

4. Print BlockChain

5. Make QR Code using Product Order ID

6. Exit

Enter Choice: 2

Select User Type:

1. Distributor

2. Client

3. Exit

Enter Your Choice: 2

Enter Client Name: Abhimanyu

1. Place orders

2. Confirm Delivery

3. Show orders

4. Log Out

Enter Choice: 1

0 chips 10

1 biscuit 10

Enter Index: 1

1. Place orders

2. Confirm Delivery

3. Show orders

4. Log Out

Enter Choice: 3

{1: ['biscuit', 'unassigned']}

1. Place orders

2. Confirm Delivery

3. Show orders

4. Log Out

Enter Choice: 4

Select User Type:

1. Distributor

2. Client

3. Exit

Enter Your Choice: 3

1. Create User

2. Log-In

3. Mine Block

4. Print BlockChain

5. Make QR Code using Product Order ID

6. Exit

Enter Choice: 3

Block mined: 08bfffa45d7b12d8f134e99085197fd275dda738515b2c8b8a00d2c5dd4e93c2

1. Create User

2. Log-In

3. Mine Block

4. Print BlockChain

5. Make QR Code using Product Order ID

6. Exit

Enter Choice: 4

Block {'timestamp': 1696256306.8685203, 'transactions': ["{'product\_name': 'biscuit', 'timestamp': 'Mon Oct 2

19:47:43 2023', 'manufacturer': 1, 'distributor': None,

'client': <\_\_main\_\_.Client object at 0x0000023BC63F4E10>, 'status': 'ORDERED By Client', 'order\_id': 1, 'hash':

'be504d9488e9069437da9d9c71bdf7129236cd8a3eb6c9c1e32bdb1beffb71b2'}"], 'prev\_hash': '0', 'nonce': 0, 'difficulty': 1, 'hash': '08bfffa45d7b12d8f134e99085197fd275dda738515b2c8b8a00d2c5dd4e93c2', 'merkle\_root': 'a99852528402f6d2b1aeb7a14a4e1400770a01e7e9c6fc79ac9dd1282346ec49'}

1. Create User

2. Log-In

3. Mine Block

4. Print BlockChain

5. Make QR Code using Product Order ID

6. Exit

Enter Choice: 6

PS E:\Blockchain\_Project>