



School: Campus:

Academic Year: Subject Name: Subject Code:

Semester: Program: Branch: Specialization:

Date:

Applied and Action Learning

(Learning by Doing and Discovery)

Name of the Experiment : Know Your TX – Dissecting a Transaction

Objective/Aim:

To study and understand about TX and how to dissect a transaction.

Apparatus/Software Used:

- Laptop/PC
- Remix IDE
- Ethereum cloud
- Sepolia tesnet
- Internet for research

Theory/Concept:

1. A transaction (TX) in the context of cryptocurrency refers to the movement of assets or data between entities on a blockchain network. It represents the transfer of cryptocurrency from a sender to a receiver, and this exchange is recorded on the blockchain, ensuring transparency and security
2. To dissect a transaction, one can **use a block explorer, which allows users to look up the transaction details and confirmations using the transaction ID (TxID) or transaction hash (Tx Hash)**. The transaction hash is a unique identifier that contains information such as the sender's address, the receiver's address, the amount transferred, the time, and other relevant details.

DEPLOY & RUN TRANSACTIONS

Transactions recorded 6 i

Deployed Contracts 4

SIMPLESTORAGE AT 0xD88...3

Balance: 0 ETH

Low level interactions

CALLDATA

Transact

SimpleStorage.sol

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

contract SimpleStorage {
    uint256 public storedData;

    constructor(uint256 _storedData) payable {
        storedData = _storedData;
    }

    function set(uint256 x) public {
        storedData = x;
    }

    function get() public view returns (uint256) {
        return storedData;
    }
}
```

0: uint256: 321

storedData

SIMPLESTORAGE AT 0xFBE...9F

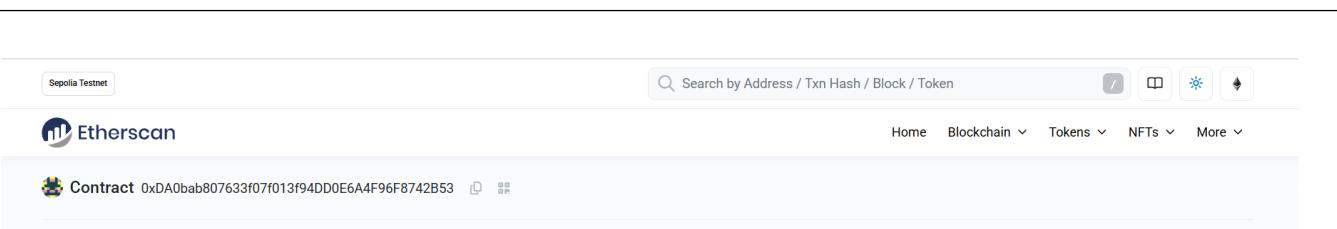
SIMPLESTORAGE AT 0x7EF...BC

SIMPLESTORAGE AT 0xDA0...4C

0 Listen on all transactions

Filter with transaction hash or address

Debug



The screenshot shows the Etherscan interface for a specific Ethereum contract. The top navigation bar includes tabs for Sepolia Testnet, Home, Blockchain, Tokens, NFTs, and More. A search bar at the top right allows users to search by Address / Txn Hash / Block / Token. Below the header, the Etherscan logo is displayed next to the contract address: 0xDA0bab807633f07f013f94DD0E6A4F96F8742B53. The main content area is divided into three sections: Overview, More Info, and Multichain Info. The Overview section shows ETH BALANCE as \$7.302153661442030265 ETH and TOKEN HOLDINGS as \$0.00 (12 Tokens). The More Info section displays the CONTRACT CREATOR as 0x5B38Da6a...f56beddC4, with a timestamp of 2 yrs 248 days ago. The Multichain Info section indicates N/A. At the bottom, there's a banner for advertising with logos for BaseScan, Etherscan, BscScan, PolygonScan, and SolScan. The footer features tabs for Transactions, Internal Transactions, Token Transfers (ERC-20), Contract, and Events, with the Transactions tab currently selected. A table below lists the latest 25 transactions from a total of 928, showing columns for Transaction Hash, Method, Block, Age, From, To, Amount, and Txn Fee.

The screenshot shows the Gwei app interface. At the top, it displays "Account 1" with address "0xDADe4...CCb81". Below this, the balance is shown as "0.05 SepoliaETH" with a "Portfolio" link. There are buttons for "Buy/Sell", "Swap", "Bridge", "Send", and "Receive". A message indicates "Solana is now supported" with a "Create a Solana account to get started" link. The "Tokens" tab is selected, showing "Sepolia" with "SepoliaETH" at 0.05 SepoliaETH and a note "No conversion rate available".

Google Cloud Web3

PYUSD analytics now available in Looker Studio. View dashboard →

Ethereum Sepolia Faucet **BETA**

Get free Sepolia ETH sent directly to your wallet. Brought to you by [Google Cloud for Web3](#).

Drip complete

Testnet tokens sent! Check your wallet address.

Network
Ethereum Sepolia

Recipient
0xDADe4DfDDA3E39e0580D206BcFB59b93cAECC6b81

Transaction hash
0x4dc27ec45a52e1c7144ce977154684115b023166d1bc295789881a3ef8e77

Procedure:

1. **Start the transaction:** Initiate the transaction to group multiple database operations into a single unit of work.
2. **Disable auto-commit:** Turn off the auto-commit mode to prevent each SQL statement from being automatically committed.
3. **Execute SQL statements:** Perform the necessary database operations such as updates, inserts, or deletes within the transaction.
4. **Commit the transaction:** If all operations are successful, commit the transaction to make the changes permanent in the database.
5. **Rollback the transaction:** If any operation fails, rollback the transaction to undo all changes and restore the database to its previous consistent state.
6. **Handle exceptions:** Implement error handling to catch any exceptions that occur during the transaction and decide whether to commit or rollback based on the error.
7. **Close the connection:** Ensure that the database connection is properly closed after the transaction is completed.

Observation Table:

1. Identify transactions and source documents: Transactions must be first identified and corroborated with source documents like receipts, invoices, or bank statements.
2. Analyse transactions using the accounting equation: Each transaction is examined to see how it affects the accounting equation: Assets = Liabilities + Owner's Equity.
3. Record journal entry: After the analysis, the transaction is recorded in the journal as a journal entry, capturing the debit and credit aspects as per double-entry bookkeeping.
4. Post entry to ledger: Finally, the journal entry is posted to the general ledger where it is categorized into the appropriate accounts

:

ASSESSMENT

Rubrics	Full Mark	Marks Obtained	Remarks
Concept	10		
Planning and Execution/ Practical Simulation/ Programming	10		
Result and Interpretation	10		
Record of Applied and Action Learning	10		
Viva	10		
Total	50		

Signature of the Faculty:

Signature of the Student:
 Name : _____
 Regn.No. _____

