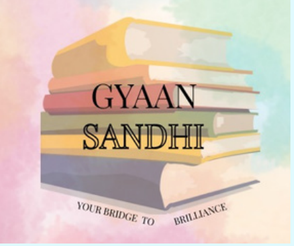



SPPU

# BLOCKCHAIN TECHNOLOGY

## UNIT 5

## Blockchain Ethereum Platform using Solidity

# CONTENTS

# ETHEREUM

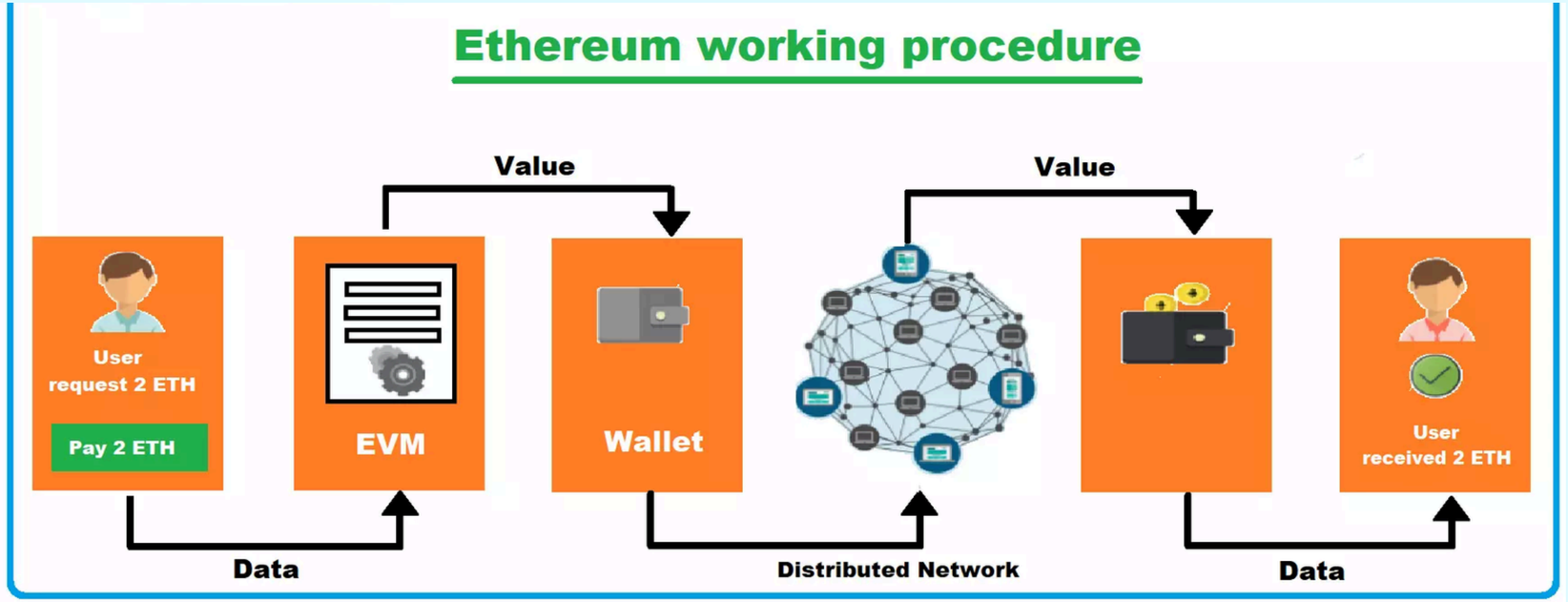
- Ethereum is a decentralized blockchain platform used to build and run smart contracts and dApps (decentralized applications).
- It uses Ether (ETH) as its native currency to pay gas fees for transactions and smart-contract execution.
- Powered by the Ethereum Virtual Machine (EVM), allowing developers worldwide to deploy programmable logic on the blockchain.
- Shifted from Proof-of-Work to Proof-of-Stake (The Merge), making the network more energy-efficient and secure.
- Supports major innovations like :

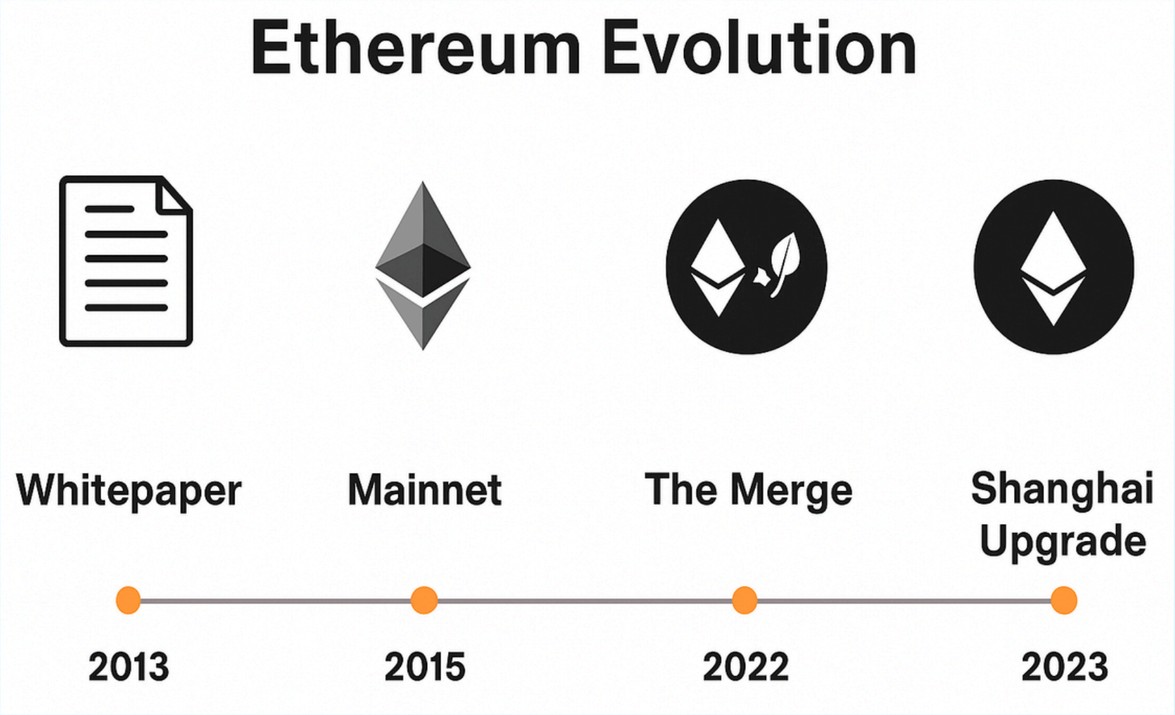
  DeFi: lending, borrowing, DEXs (Uniswap, Aave)

  NFTs & digital ownership: collectibles, art, gaming

  DAOs: decentralized governance organizations

  dApps: social, identity, supply chain, gaming, oracles integration


Ethereum working procedure
Value
Value
User
request 2 ETH
Pay 2 ETH
EVM
Wallet
User
received 2 ETH
Data
Distributed Network
Data


Ethereum Evolution
Whitepaper
Mainnet
The Merge
Shanghai
Upgrade
2013
2015
2022
2023

# TYPES OF ETHEREUM NETWORKS

## Mainnet (Main Network)

- Purpose: Real-world transactions
- Use Case: Actual ETH transfer, DeFi, NFTs, smart contracts
- Currency: Real ETH (high value)

✓ Key Points

- It is the primary, public, live Ethereum blockchain.
- All transactions are permanent and cost real ETH.
- Used for real applications: Uniswap, OpenSea, Lido, etc.
- Very secure because it has thousands of validators.

## Testnets (Testing Networks)

- Testnets are safe environments to test smart contracts without spending real ETH.
- Popular Ethereum Testnets:

a) Goerli Testnet (Old – now deprecated in 2023)

- Used earlier to test dApps.
- Test ETH was free (faucets).
- Now mostly replaced.

b) Sepolia Testnet (Current official testnet)

- Lightweight, stable testnet.
- Developers use it to test contracts before deploying to Mainnet.
- Uses Sepolia ETH (worthless, only for testing).

c) Holesky Testnet (New – for staking & infrastructure testing)

- Launched in 2023 for validator/staking testing.
- Supports more validators than Mainnet.

# TYPES OF ETHEREUM NETWORKS

## Local Networks (Developer Networks)

These are private, local blockchains used by developers on their own PC.

Examples:

- Ganache
- Hardhat Network
- Foundry (Anvil)
- Geth private network

✓ Key Points

- Developers use them for fast testing.
- No internet connection required.
- Blocks are mined instantly.
- Full control over accounts, balances, gas fees.

## Private Networks (Enterprise / Organization Networks)

These are custom Ethereum networks created by companies or research teams.

Examples:

- Quorum (by JPMorgan)
- Hyperledger Besu Enterprise Network

✓ Key Points

- Used in enterprises for supply chain, banking, identity.
- Controlled access (not public).
- Very secure and customizable.
- Often used for confidential business transactions.

# EVM

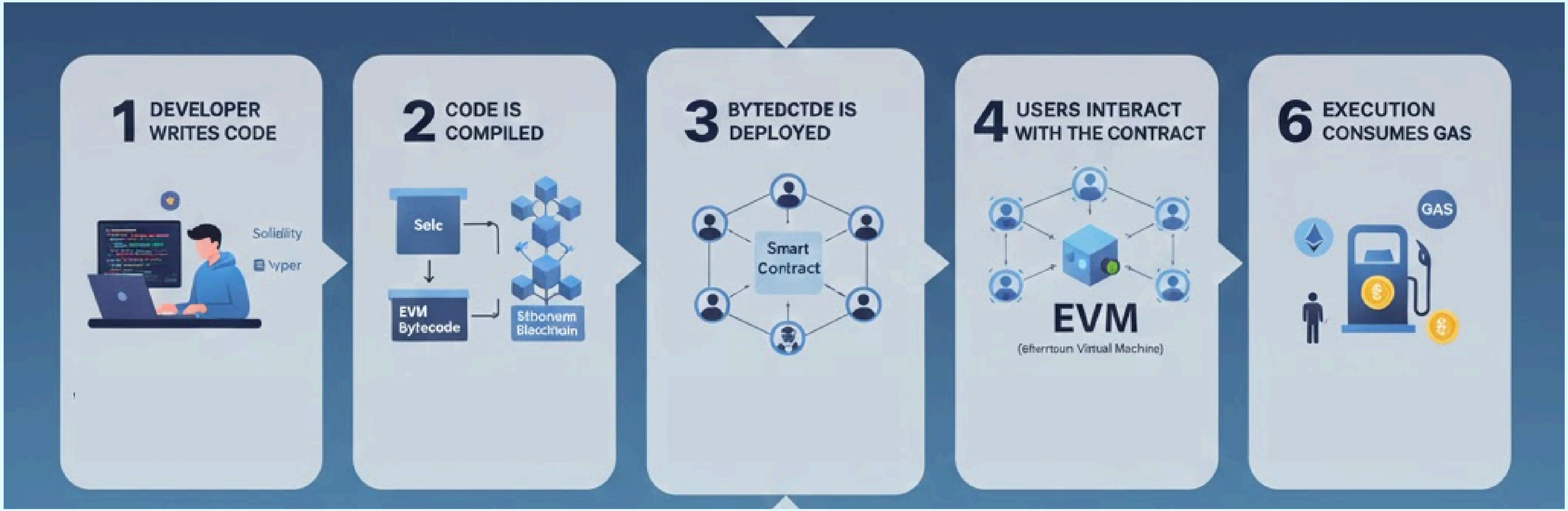
EVM is the “brain” of Ethereum — a global computer that executes smart contracts exactly as programmed, without downtime or manipulation.

- EVM is a virtual computer that runs on every Ethereum node across the world.
- It executes smart contract code written in languages like Solidity.
- Converts high-level code → EVM Bytecode, so all nodes can understand it.
- Makes Ethereum programmable, enabling dApps, DeFi, NFTs, DAOs, etc.
- Ensures deterministic execution → same input = same output on every node.
- Uses gas to measure computational work and prevent spam.

**Why EVM is Important**

- It creates a common environment where all smart contracts run.
- Ensures security and trustless execution.
- Allows thousands of decentralized apps to run without servers.
- Enables Ethereum’s ability to function like a global supercomputer.

# WORKING OF EVM



# INTRODUCTION TO SMART CONTRACTS

Smart Contracts are self-executing digital agreements where the terms are written directly in code.

- They run automatically on blockchain platforms like Ethereum using the EVM (Ethereum Virtual Machine).
- They require no third party (bank, broker, middleman).
- Once deployed, they are immutable and transparent.
- They execute automatically when predefined conditions are met.
- Example:
- "If A sends ₹1000, then B will receive digital ownership of a document."


**Purpose of Smart Contracts**

Smart Contracts are designed to provide:

1. Automation : Executes actions automatically when conditions are satisfied.
2. Trust & Transparency : Everyone can verify the code on the blockchain.
3. Security : Stored on distributed networks — tamper-proof and hack-resistant.
4. Cost Saving : No need for lawyers, banks, or intermediaries.
5. Speed : Eliminates paperwork and manual verification.
6. Accuracy : Exact execution according to defined rules.

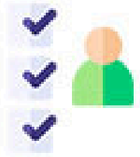
# TYPES OF SMART CONTRACTS

1. Deterministic Smart Contracts
   - Produce the same output for the same input.
   - Used in payments, escrow, token transfers.
2. Smart Legal Contracts
   - Combine legal agreements + automation.
   - Used in: insurance, real estate, legal settlements.
3. Decentralized Autonomous Organizations (DAO) Contracts
   - Govern entire organizations.
   - Voting, treasury management, governance rules.
4. Application Logic Contracts (ALC)
   - Connect blockchain with frontend applications.
   - Used in DApps, DeFi, NFT platforms.
5. Token Smart Contracts
   - Used for creating cryptocurrencies and NFTs.
   - Example: ERC-20, ERC-721.

# How does a Smart Contract Work?

### Identify Agreement

Multiple parties identify the cooperative opportunity and desired outcomes.



### Set conditions

Smart contracts are executed automatically when certain conditions are met.

### Code business logic

A computer program is written

### Encryption and blockchain technology

Encryption provides a secure transfer of messages between parties.

### Execution and processing

The code is executed and outcomes are memorialized.

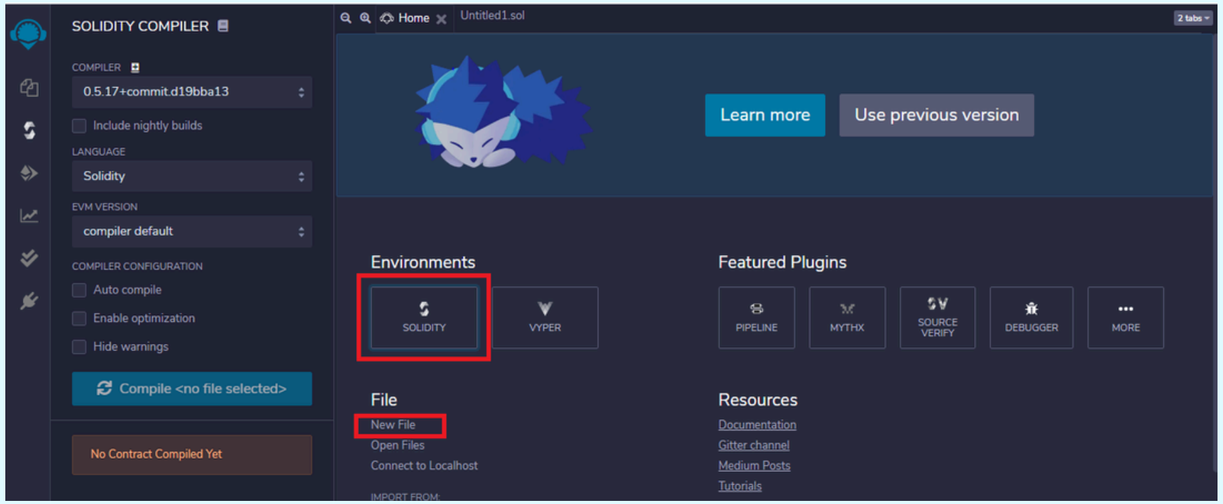
### Network updates

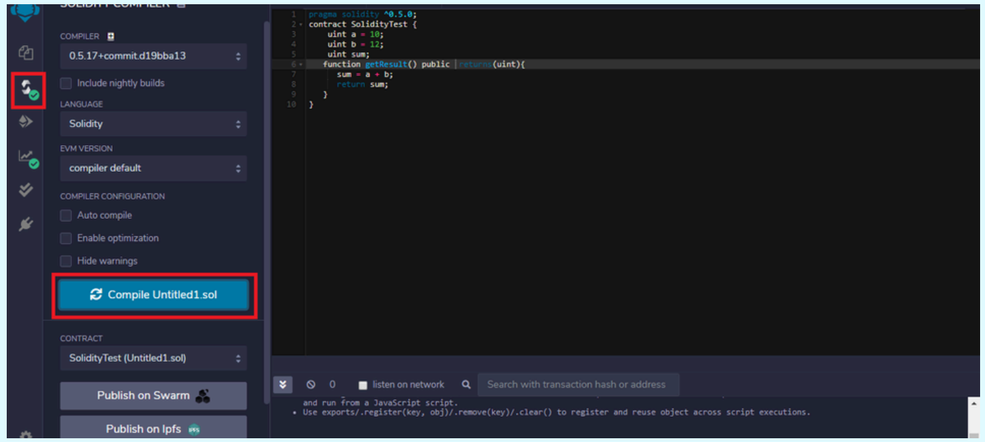
All the nodes on the network update their ledger.

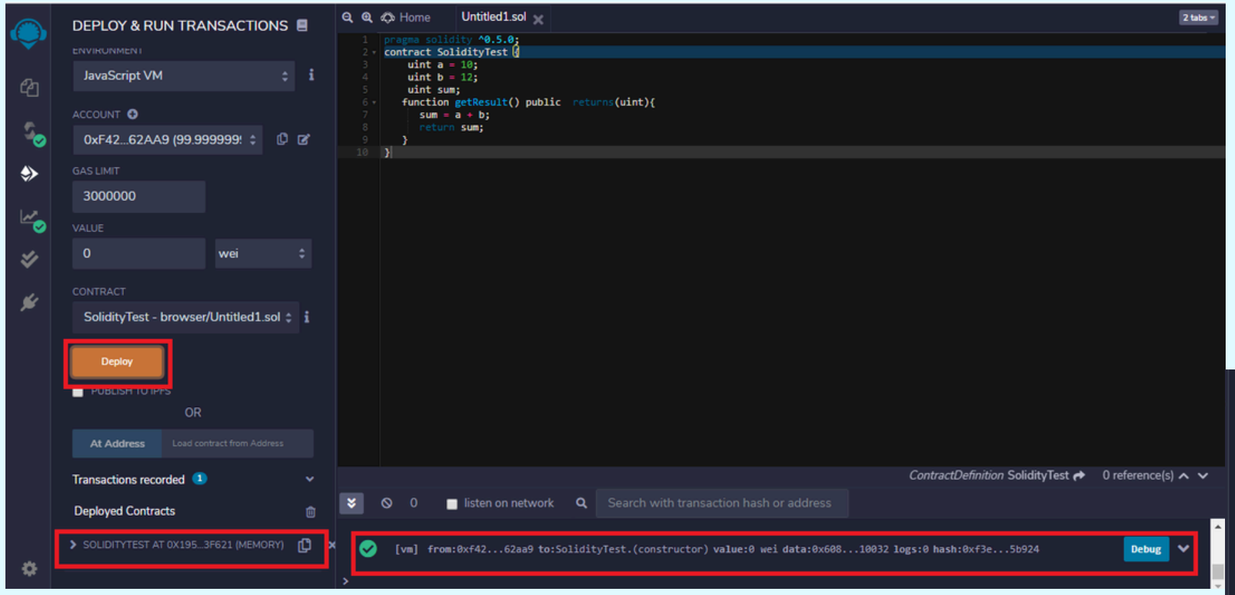
# IMPLEMENTING AND DEPLOYING SMART CONTRACTS USING SOLIDITY.

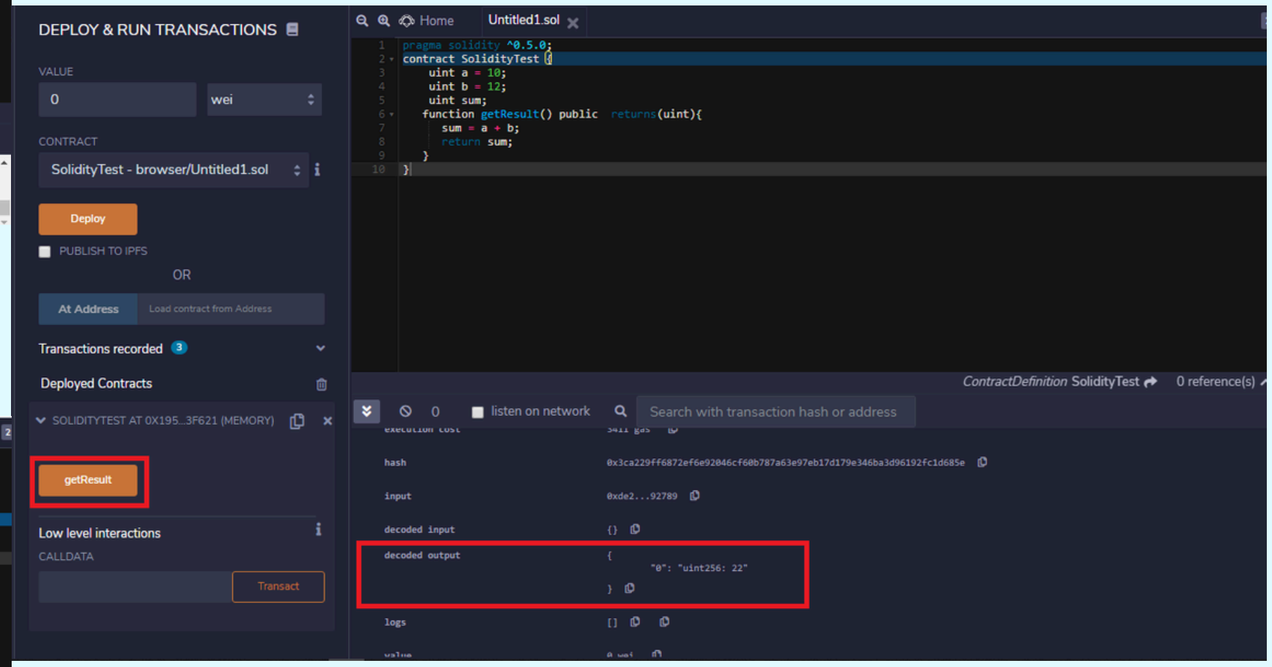
Solidity is a high-level, object-oriented programming language used to write smart contracts on Ethereum. It looks similar to JavaScript, C++, and Python.

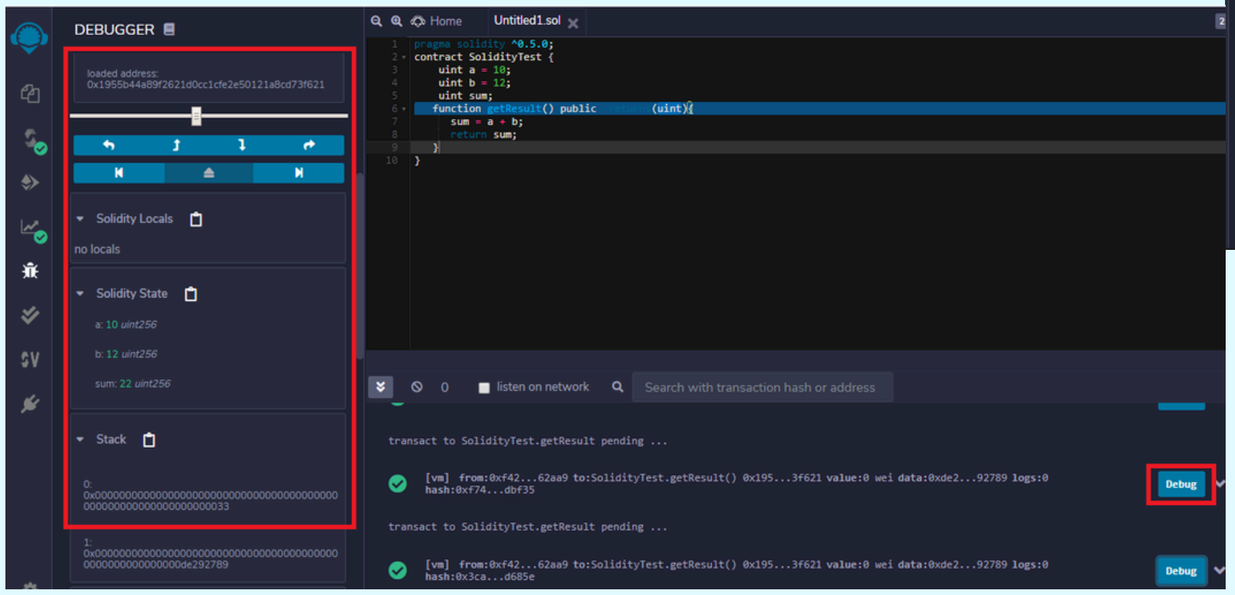
- Set Up a Development Environment
- Connect to the Ethereum Network
- Create and Configure Your Wallet
- Write Your Solidity Contract
- Compile the Contract
- Deploy to a Testnet
- Interact With Your Contract




DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
JavaScript VM
ACCOUNT
0xF42...62AA9 (99.999999
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
SolidityTest - browser/Untitled1.sol
Deploy
PUBLISH TO IPFS
OR
At Address
Load contract from Address
Transactions recorded 1
Deployed Contracts
SOLIDITYTEST AT 0X195...3F621 (MEMORY)
Home
Untitled1.sol
2 tabs
pragma solidity ^0.5.0;
contract SolidityTest {
uint a = 10;
uint b = 12;
uint sum;
function getResult() public returns(uint){
sum = a + b;
return sum;
}
}
ContractDefinition SolidityTest
0 reference(s)
listen on network
Search with transaction hash or address
[vm] from:0xf42...62aa9 to:SolidityTest.(constructor) value:0 wei data:0x608...10032 logs:0 hash:0xf3e...5b924
Debug


DEPLOY & RUN TRANSACTIONS
VALUE
0
wei
CONTRACT
SolidityTest - browser/Untitled1.sol
Deploy
PUBLISH TO IPFS
OR
At Address
Load contract from Address
Transactions recorded 3
Deployed Contracts
SOLIDITYTEST AT 0X195...3F621 (MEMORY)
getResult
Low level interactions
CALLDATA
Transact
Home
Untitled1.sol
pragma solidity ^0.5.0;
contract SolidityTest {
uint a = 10;
uint b = 12;
uint sum;
function getResult() public returns(uint){
sum = a + b;
return sum;
}
}
ContractDefinition SolidityTest
0 reference(s)
listen on network
Search with transaction hash or address
hash
0x3ca229ff6872ef6e92046cf60b787a63e97eb17d179e346ba3d96192fc1d685e
input
0xde2...92789
decoded input
{}
decoded output
{
"0": "uint256: 22"
}
logs
[]


DEBUGGER
loaded address:
0x1955b44a89f2621d0cc1cfe2e50121a8cd73f621
Solidity Locals
no locals
Solidity State
a: 10 uint256
b: 12 uint256
sum: 22 uint256
Stack
0:
0x0000000000000000000000000000000000000000000000000000000000000033
1:
0x0000000000000000000000000000000000000000000000000000000000de292789
Home
Untitled1.sol
pragma solidity ^0.5.0;
contract SolidityTest {
uint a = 10;
uint b = 12;
uint sum;
function getResult() public (uint){
sum = a + b;
return sum;
}
}
listen on network
Search with transaction hash or address
transact to SolidityTest.getResult pending ...
[vm] from:0xf42...62aa9 to:SolidityTest.getResult() 0x195...3f621 value:0 wei data:0xde2...92789 logs:0 hash:0xf74...dbf35
Debug
transact to SolidityTest.getResult pending ...
[vm] from:0xf42...62aa9 to:SolidityTest.getResult() 0x195...3f621 value:0 wei data:0xde2...92789 logs:0 hash:0x3ca...d685e
Debug

# SWARM (DECENTRALIZED STORAGE PLATFORM),

Swarm is a decentralized storage platform designed for the Ethereum ecosystem.

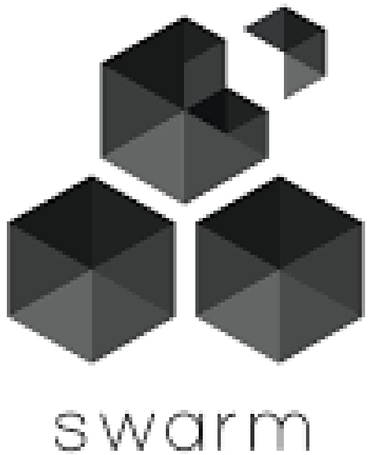
It allows users to store, share, and distribute data without relying on centralized servers.

Key Features

- ✓ Decentralized Storage :Files are broken into chunks and stored across many nodes.
- ✓ Redundancy & Fault Tolerance : Multiple copies ensure data is always available.
- ✓ Incentive System (SWARM tokens) : Nodes get rewarded for storing and serving files.
- ✓ Content-Addressed Storage : Files are retrieved using a hash (content ID), not a server location.
- ✓ Immutable and Censorship-Resistant : No single authority can remove or alter data.

Use Cases

- Decentralized websites
- DApps data storage
- Media streaming
- Backup & archival storage
- NFT metadata storage



# WHISPER (DECENTRALIZED MESSAGING PLATFORM)

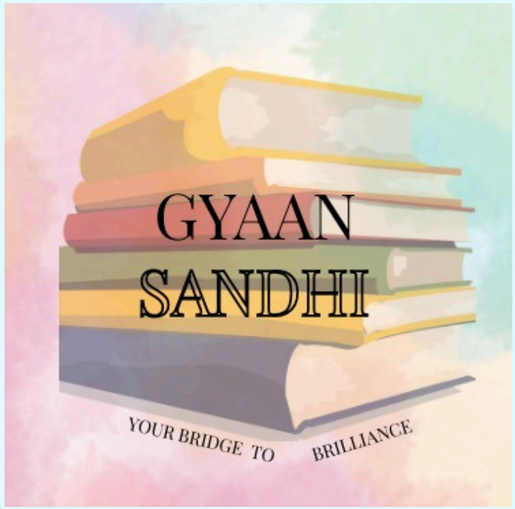
Whisper is a peer-to-peer, decentralized messaging protocol used in Ethereum for sending secure, anonymous, and encrypted messages between users or DApps.

Key Features

- ✓ End-to-End Encryption : Ensures message privacy.
- ✓ Anonymity : Sender and receiver identities are hidden.
- ✓ Decentralized Communication : No server or central authority involved.
- ✓ Low-level messaging : Useful for DApps that need private chat or signaling.
- ✓ Off-chain Communication : Messages are not stored on the blockchain, reducing cost and congestion.

Use Cases

- Private messaging/chat between DApp users
- Secure trading signals in decentralized exchanges
- Anonymous notifications
- DAO voting messages
- Peer-to-peer coordination




# THANK YOU


Share