Smart contract and Ethereum

1.Why do we use Ethereum?

2.Why do we need it over Bitcoin?

3.What problem does Ethereum solve?

1. Bitcoin's limitation

Bitcoin was made mainly for **digital money**.

- Send money
- Receive money
- Store value (like digital gold)

Problem:
Bitcoin **cannot run complex programs**.
It only does payments very well.

2. Ethereum's solution

Ethereum was created to go **beyond money**.

It introduced **Smart Contracts**.

Smart Contract = self-running program on blockchain

Example:

- If conditions are true → action happens automatically
- No middleman needed

3. What Ethereum can do (Bitcoin can't easily)

With Ethereum, you can build:

- **DeFi** (loans, staking, exchanges)
- **NFTs**
- **Decentralized Apps (DApps)**
- **DAOs** (organizations without a boss)

Bitcoin can't do all this natively.

4. Real-world example

Bitcoin example:

"Send 1 BTC from A to B"

Ethereum example:

"If Vijay sends 1 ETH before Friday, release NFT automatically, else refund"

This logic runs **without trust**.

4.Smart Contract (short & simple):

A **smart contract** is a **self-executing program on blockchain** that **automatically runs when conditions are met**, without any middleman.

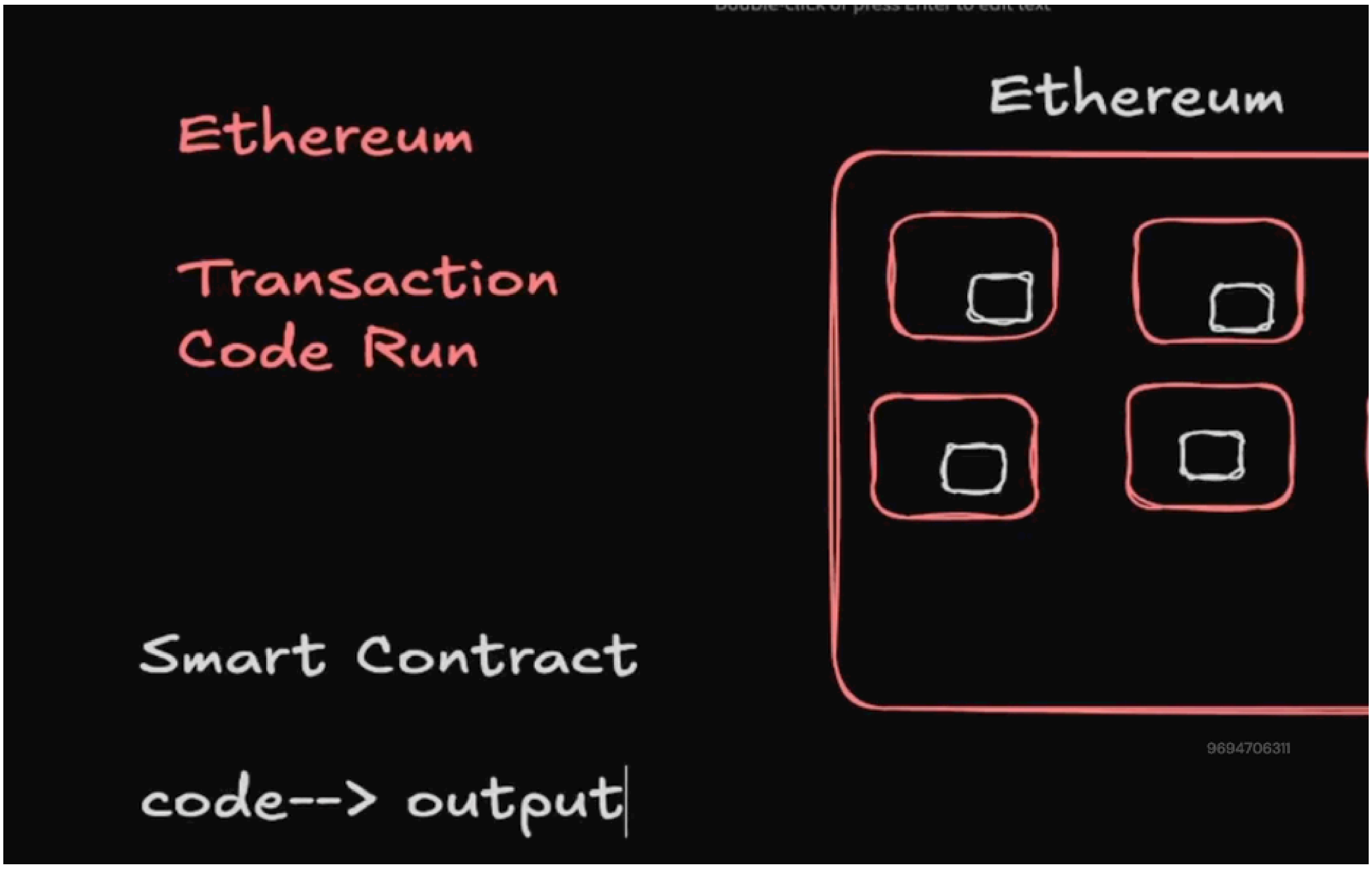
it is a piece of code

Example:
If payment is received → product/NFT is sent automatically.

No trust needed. No human control.

Difficulties/problems in smart contracts

1. **Bugs & errors**
   Once deployed, smart contracts **cannot be easily changed**.
   A small bug can cause **huge money loss**.
2. **Security attacks**
   Hackers can exploit code (reentrancy, overflow, etc.).
3. **No real-world data by default**
   Smart contracts **can't access outside data** (price, weather).
   They need **oracles**, which adds risk.
4. **Gas fees**
   Running contracts costs **gas**.
   Complex contracts = **high fees**.
5. **Immutability problem**
   If logic is wrong, you **can't fix it easily** after deployment.
6. **Scalability limits**
   Network congestion makes transactions **slow and expensive**.

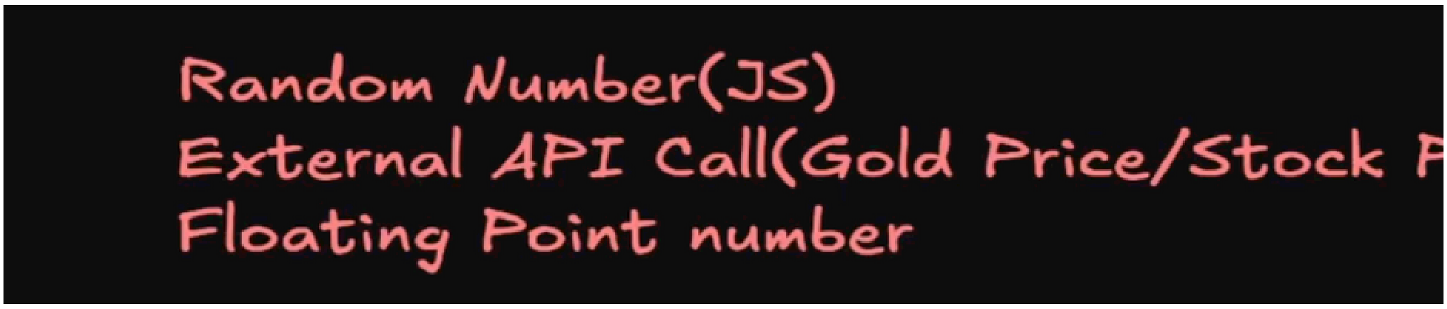


Ethereum

Transaction
Code Run

Smart Contract

code--> output



Random Number(JS)
External API Call(Gold Price/Stock
Floating Point number

solidity language we will use to create smart contract

C++ and JavaScript are not used because smart contracts must run safely, deterministically, and identically on every blockchain node.

Why not C++

- Too **complex & unsafe** (memory errors, undefined behavior)
- Different compilers → **different results** on nodes

Why not JavaScript

- **Non-deterministic** (random, time, async, APIs)
- Depends on runtime (Node, browser) → **not trustless**

5.What is Solidity?

Solidity is a **high-level programming language** used to **write smart contracts** on **Ethereum and EVM-based blockchains**.

It is **deterministic**, **secure**, and **designed specially for blockchain**, unlike JavaScript or C++.

6.What is EVM (Ethereum Virtual Machine)?

The EVM (Ethereum Virtual Machine) is a global virtual computer that runs smart contracts and ensures every Ethereum node executes the same code and gets the same result.

Why EVM is needed (problem → solution)

Problem (without EVM)

- **Thousands of nodes run contracts**
- **Different OS, hardware, compilers**
- **Results could differ → blockchain breaks**

Solution (EVM)

- **A standard execution environment**
- **Same rules, same instructions**
- **Result is 100% deterministic**

How EVM works

You write smart contract in Solidity

1. **Solidity → Bytecode**
2. **Bytecode runs inside EVM**
3. **Every node runs the same bytecode**
4. **Same output → consensus**

Real-world example

Think of EVM like an ATM Machine

- **Your bank → SBI / HDFC / ICICI (different systems)**
- **ATM → same interface & rules**
- **Card + PIN → same result everywhere**

ATM = EVM
Banks = different nodes

No matter which bank's ATM, rules are same.

Another real-world example (Office Rules)

- 100 employees
- Different laptops, OS
- Company gives one rulebook

Rulebook = EVM

Employees = nodes

Everyone follows same rules → no confusion.

What EVM does NOT do

- It does not access internet
- It does not call APIs
- It does not allow randomness

(For determinism)

Ethereum has two types of accounts.

Externally Owned Account (EOA)

What it is:
An account controlled by a person using a private key.

Features:

- Created by a wallet (MetaMask, Trust Wallet)
- Can send transactions
- Can call smart contracts
- Has ETH balance
- No code inside it

Real-world example:
Like your personal bank account
You sign (private key) → money is sent.I will control this

Example:
Vijay opens MetaMask → gets address → sends ETH.

Contract Account (Smart Contract Account)

What it is:
An account controlled by code, not by a human.

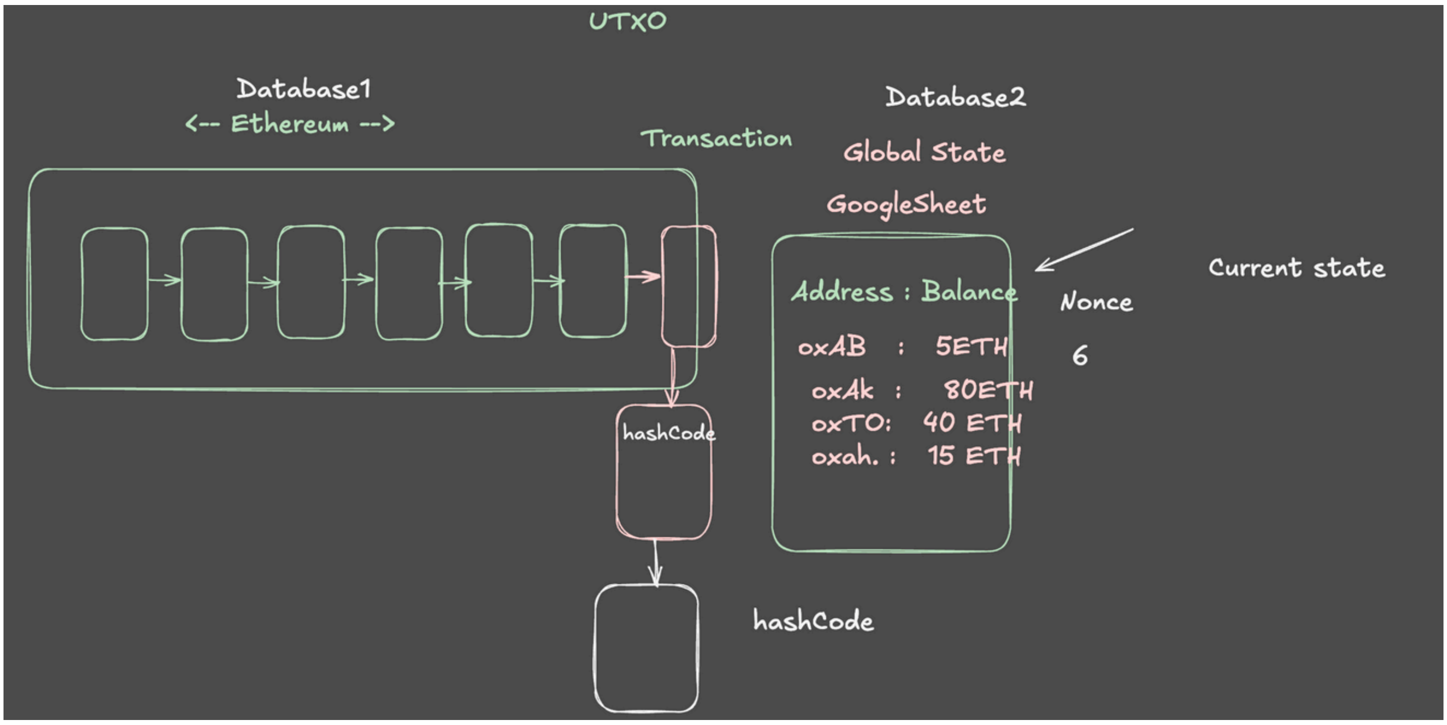
Features:

- Created when a smart contract is deployed
- Has code

- Cannot start a transaction by itself
- Runs only when called
- Can hold ETH & tokens

Real-world example:
Like a vending machine
You press button + insert money → machine reacts.

Example:
Send ETH to NFT contract → NFT is minted automatically.


UTXO
Database1
<-- Ethereum -->
Transaction
Database2
Global State
GoogleSheet
Current state
Address : Balance
oxAB : 5ETH
oxAk : 80ETH
oxTO: 40 ETH
oxah. : 15 ETH
Nonce
6
hashCode
hashCode