7️⃣

Lecture 07: Ethereum and smart Contract and Doubt session

🌳 Fork Process in Blockchain

In blockchain networks (like Bitcoin, Ethereum), all nodes try to agree on the same chain of blocks — this is called **consensus**.

But sometimes, two valid blocks are produced at the same time. This creates a **temporary fork** — the blockchain splits into two possible paths.

Here's how it works step by step:

🔄 Why Do Forks Happen?

✅ Two miners solve the **Proof of Work (PoW)** puzzle almost simultaneously.

✅ Both broadcast their block to the network.

✅ Some nodes see one block first, others see the other — so the chain "forks" into two branches.

Visual:

```
less
CopyEdit
      o → o → o
              |
              |→ Block A
              |
              |→ Block B
```

Both `Block A` and `Block B` are valid — but the blockchain can only continue on one path in the long run.

⚖️ Resolving a Fork: Rules of Truth

The network uses **two rules of consensus** to resolve forks and maintain integrity:

✅ Rule 1: Work Truth (Longest Chain Wins)

> The branch with the most cumulative Proof of Work (PoW) is considered the valid chain.

- Miners keep building on the chain they received first.
- Eventually, one branch becomes longer (has more work) than the other.
- Nodes switch to the longer chain and abandon the shorter one (orphan blocks).

Reason: The longest chain represents the majority of the network's work and is the "truth."

✅ Rule 2: Honest Majority

> The majority (>50%) of mining power is assumed to be honest.

This ensures no malicious miner can easily take over the chain by extending their own fork faster than everyone else.

If a malicious actor controls more than 50% of the network's total hash power, they could theoretically build a longer (fraudulent) chain — this is called a **51% attack**.

🔗 In Summary:

- Temporary forks happen naturally because of network latency and simultaneous block discovery.
- The chain with the **most cumulative work (longest)** is the valid one.

- Honest majority ensures trust in the system.

📌 Final Notes:

✅ Forks are resolved automatically by the protocol.

✅ Blocks in the shorter branch become **orphaned** and discarded.

✅ Users don't need to do anything — their node will always follow the valid chain.

Part 1

🎯 Problem: Crowdfunding with a Refund if Goal Not Met

We want to create a **crowdfunding contract** where:

- A funding goal (`1 lakh BTC`) must be met **before 31 July**.
- If goal is met → project owner can claim the funds.
- If goal is **not met by 31 July** → all funds are **returned to contributors automatically.**

❌ Why Bitcoin Cannot Solve This (On-Chain)

Bitcoin's scripting language is:

- Very limited (to keep it simple and secure).
- Not *Turing-complete* — it cannot express complex conditions or loops.
- No way to write a smart contract that says:

> "If block time > 31 July and goal not reached, then allow refunds per contributor."

Bitcoin **can only do basic scripts:**

- Multi-signature payments.

- Timelocks (only lock/unlock at a fixed time, but no logic to check goals or amounts).

So: You would need to run an **off-chain process** to check if the goal was met, and manually send refunds — not trustless and not automatic.

✅ Why Ethereum Can Solve This

Ethereum was **designed for programmable logic on-chain** — it is a **Turing-complete smart contract platform.**

You can write a **smart contract (in Solidity)** that does exactly this:

- Accept contributions and keep track of each contributor and how much they sent.
- Check the total amount of funds collected.
- Store a deadline timestamp (e.g., block.timestamp ≤ 31 July).
- Define two possible outcomes:
  1. If funding ≥ 1 lakh BTC (or equivalent ETH) **before deadline** → owner can withdraw.
  2. If funding < 1 lakh BTC **after deadline** → contributors can withdraw their own contributions.

👨‍💻 How it Works in Ethereum — Logic Example

Contract Logic:

Condition	Action
`now < 31 July`	Accept contributions, update mapping of who sent how much
`now > 31 July && total ≥ goal`	Allow owner to withdraw
`now > 31 July && total < goal`	Allow contributors to withdraw their own funds

Solidity Example (Simplified Pseudo-Code)

```
solidity
CopyEdit
mapping(address ⇒ uint) public contributions;
uint public total;
uint public goal = 100000 * 1 ether;
uint public deadline = 1753910400; // 31 July timestamp
address public owner;

function contribute() public payable {
    require(block.timestamp <= deadline);
    contributions[msg.sender] += msg.value;
    total += msg.value;
}

function withdrawOwner() public {
    require(block.timestamp > deadline);
    require(total >= goal);
    require(msg.sender == owner);
    payable(owner).transfer(address(this).balance);
}

function refund() public {
    require(block.timestamp > deadline);
    require(total < goal);
    uint amount = contributions[msg.sender];
    contributions[msg.sender] = 0;
    payable(msg.sender).transfer(amount);
}
```

🔷 Why Ethereum Can Do This

✅ Smart contracts are programmable & autonomous.

✅ All logic is enforced on-chain — no need for trust.

✅ Every participant can interact with the contract directly.

✅ No central authority is needed to process refunds.

Summary Table:

Feature	Bitcoin	Ethereum
Programmable logic?	❌ Very limited	✅ Full smart contracts
Check deadline + goal?	❌ Off-chain	✅ On-chain
Automatic refunds?	❌ No	✅ Yes
Trustless?	❌ Partially	✅ Fully

Part 2

📹 Why YouTube Is Not Built on Blockchain

YouTube today is a **centralized video hosting and streaming platform.**

Blockchain is great for some use cases — but not all.

Let's analyze why 👇

🔷 1. Blockchains are *slow* and low-throughput

- Bitcoin → ~7 transactions per second.
- Ethereum → ~15–30 transactions per second (pre-L2s).
- Even newer chains can only handle a few thousand transactions per second.
- YouTube handles:

  ✅ Billions of video views per day.

  ✅ Millions of uploads per day.

  ✅ Massive data requests every second.

Problem: Blockchain consensus is too slow and too resource-intensive for streaming high-quality video to billions of users in real time.

🔷 2. Blockchains are expensive

- Every action you take (uploading, commenting, liking) would require an on-chain transaction.
- Even on cheap chains, gas fees add up when you have billions of users doing millions of actions per minute.
- Imagine paying a fee just to press the “Like” button — bad UX.

🔷 3. Blockchains are not good for large files

- Blockchain is great for *small, immutable records* (like token balances, ownership records).
- Video files are massive — even one 4K video is hundreds of megabytes or gigabytes.
- Storing even a few terabytes of video content directly on a blockchain would make it unmanageable and incredibly costly.

🔷 4. Centralized infrastructure is more efficient

YouTube’s servers can deliver:

- Video compression & encoding optimized for your device.
- Content delivery networks (CDNs) that cache videos close to you.
- Adaptive bitrate streaming for smooth playback even on weak connections.
- Blockchains simply cannot replicate this level of performance yet.

🔷 5. Censorship and moderation

- Blockchains are immutable and decentralized — great for censorship resistance.
- But what about illegal, abusive, or harmful content?

  On YouTube, moderators and AI can remove it quickly.

  On a blockchain, once it’s published, it’s there forever — a huge problem for legality & ethics.

✅ What Could Blockchain Add to Video Platforms?

While blockchain isn't suitable for fully hosting YouTube, it *can improve certain aspects*:

- **Ownership of content:** Prove you created and own a video (NFTs).
- **Decentralized payments:** Creators can get paid directly in crypto.
- **Censorship-resistant publishing:** For niche, freedom-focused platforms.
- **Transparent ad revenue sharing.**

🔷 TL;DR

Feature	YouTube (Centralized)	Blockchain
Video hosting & streaming	✅ Fast & efficient	❌ Slow & expensive
Scalability	✅ Handles billions	❌ Limited
Moderation	✅ Enforceable	❌ Hard to remove bad content
Transparency & trust	❌ Opaque	✅ Transparent
Censorship resistance	❌	✅

🌱 Future?

Some projects (like Livepeer, Theta, Audius) are experimenting with *hybrid models*:

- Video stored & streamed via decentralized CDNs.
- Metadata & ownership on blockchain.
- Payments & governance on blockchain.

But for now — YouTube's centralized model is far more practical.
