— INTRODUCTION TO SMART CONTRACTS

◆ What Is a Smart Contract?

A **smart contract** is a computer program stored and executed on a blockchain.
It runs automatically when predefined conditions are met.

Think of it as a **robot agreement**:

- It cannot lie
- It cannot forget
- It cannot cheat
- It cannot change its mind
- It only does *exactly* what you coded

Unlike traditional contracts:

- No lawyers
- No government office
- No paper
- No human “approval” step

It’s pure **if-this-then-that** logic.

Example:

"If user sends 1 ETH → release the NFT."

That’s it. No trust required.
The blockchain acts as the judge, the server, and the executor.

WHY SMART CONTRACTS EXIST

🔹 The Problem With Traditional Systems

Normal digital systems rely on **centralized authorities**:

- Banks
- Companies
- Servers
- Databases
- Admins

This creates problems:

- Downtime
- Corruption
- Fraud
- Data manipulation
- Human mistakes
- Trust issues

Smart contracts solve these issues by removing all middlemen.

🔹 Why Decentralization Matters

In decentralized systems:

- Data is stored across thousands of nodes
- No single person controls it
- No one can secretly change something

Smart contracts work on top of this decentralized network.
So they inherit blockchain qualities like:

- **Immutability**

- **Transparency**
- **Security**

This makes smart contracts extremely trustworthy.

HOW SMART CONTRACTS WORK INTERNALLY

Let’s break down the full lifecycle.

🔸 1. Writing the Contract

Usually done with:

- Solidity (Ethereum)
- Rust (Solana)
- Vyper (Ethereum alt language)

Example of contract logic:

```
if payment_received:
    give_access()
else:
    reject()
```

🔸 2. Compiling It

The code is converted into **bytecode** (machine-readable instructions).

🔸 3. Deploying to Blockchain

Once deployed:

- It gets a **permanent blockchain address**

- It becomes immutable
- Everyone can interact with it

Deployment requires **gas fees** (paid in ETH, BNB, etc.).

🔸 4. Execution

Every time someone interacts with it:

- Nodes simulate the contract execution
- The network comes to a consensus
- The result is recorded on-chain

This means:

- Guaranteed output
- No server failures
- No admin interference

CORE PROPERTIES OF SMART CONTRACTS

⭐ 1. Immutability

Once deployed, code cannot be changed.

Why it matters:

- No admin can secretly update rules
- No company can change fees
- No government can censor it

Downside:

- If you make a bug → too bad
- Many hacks happened due to this

⭐ 2. Transparency

Anyone can view:

- The code
- Transactions
- Balances
- Logic

This builds trust, especially in financial systems.

⭐ 3. Deterministic Execution

Given the same input → always gives the same output.

You never get random results.
This consistency is necessary for consensus.

⭐ 4. Decentralized Security

Smart contracts are protected by:

- Cryptography
- Consensus algorithms
- Decentralized networks

Hacking one node does NOTHING, because:

- You’d need to attack thousands
- And rewrite the blockchain history
- Which is almost impossible

⭐ 5. Autonomy

After deployment, the contract runs itself.

No one “manually approves” anything.

COMPONENTS OF A SMART CONTRACT SYSTEM

📌 1. Blockchain Network

Examples:

- Ethereum
- BNB Chain
- Polygon
- Solana

The network verifies and runs contract code.

📌 2. Virtual Machine

Every blockchain needs something to *run* the code.

Ethereum uses:

- **EVM → Ethereum Virtual Machine**

Others:

- Solana VM
- WASM-based VMs

The VM executes instructions safely and deterministically.

📌 3. Gas System

Every contract operation costs “gas”.

Why?

To prevent:

- Infinite loops
- Spam
- Resource abuse

Gas = computational cost
Paid using blockchain’s token.

Example:

- Ethereum → gas paid in ETH
- BNB chain → gas paid in BNB

📌 4. State Storage

Smart contracts store:

- Balances
- Ownerships
- Variables
- Settings

This state lives forever on the blockchain.

TYPES OF SMART CONTRACTS

1️⃣ Token Contracts

Examples:

- ERC-20 tokens (cryptocurrencies)
- ERC-721 (NFTs)
- ERC-1155 (multi-tokens)

These define:

- How tokens are created
- How they move
- How ownership works

2️⃣ Financial Contracts (DeFi)

Used in:

- Lending pools
- Borrowing
- Staking
- Yield farming
- Flash loans

These replace banks completely.

3️⃣ Governance Contracts

Used in DAOs (Decentralized Autonomous Organizations).

DAO = company with no CEO
Rules are coded in smart contracts.

4️⃣ Escrow Contracts

For safe transactions.

Example:
Buyer sends money → Contract holds it → Seller delivers → Contract releases.

No cheating from either side.

5️⃣ Marketplace Contracts

Used in:

- NFTs
- Auctions
- Bidding
- Royalty systems

6️⃣ Identity & Access Control Contracts

Used in:

- Voting systems
- Attendance systems
- Membership passes (NFT access)