ABHYASMITRA.IN

COMPREHENSIVE UNIVERSITY STUDY NOTES

BLOCKCHAIN TECHNOLOGY

Unit IV — Ethereum, EVM, Smart Contracts & DApps

Topics Covered in this Unit

Ethereum Platform Architecture: Need for Ethereum, Account Models (EOAs vs. Contract Accounts)
Ethereum Virtual Machine (EVM): Architecture, Stack Memory, Storage & Turing-Completeness
Gas Economics & Transaction Fees: Gas Limits, Gas Price (Gwei), EIP-1559 Base Fee Burn & OOG Errors
Smart Contract Fundamentals: Self-Executing Deterministic Code & State Immutability
Solidity Programming Language: Syntax, Types, Modifiers, Events, Error Handling (require/revert/assert)
Standard Token Standards: ERC-20 (Fungible Tokens) & ERC-721 (Non-Fungible Tokens NFTs)
Smart Contract Tooling: Development, Testing, Compilation & Deployment using Remix IDE
Decentralized Applications (DApps): Full-Stack Web3 Architecture & RPC State Interaction
Decentralized Storage: InterPlanetary File System (IPFS), Content Identifiers (CIDs) & BitSwap
Web3.js & Ethers.js Basics: Providers, Signers, Contract ABI & JSON-RPC Client Integration
High-Yield University Exam Question Bank with Detailed Model Answers & Unit Summary

Compiled in a structured, easy-to-understand, textbook style format
Specially curated for B.Tech / B.E. / BCA / MCA / B.Sc Computer Science & IT Students

COURSE SYLLABUS & MAPPING

[UNIT IV SYLLABUS — PRESCRIBED UNIVERSITY CURRICULUM]

This section contains the official syllabus for Unit IV. Verify with your university curriculum mapping.

Unit IV Syllabus Breakdown:

Unit IV - Ethereum and Smart Contracts: Ethereum Platform Architecture: Need of Ethereum, Type of Ethereum Platforms, Ethereum Virtual Machine (EVM), Gas and Transaction Fees, Smart Contract Fundamentals, Solidity Programming Language, Deploying And Interacting with Smart Contracts using Remix, Decentralized Applications (Dapps), Decentralized Storage (IPFS) Web3.Js Basics.

• Course Code: _____ • Semester / Year: _____

TABLE OF CONTENTS

4.1 Ethereum Platform Architecture & Account Models

Definition: Ethereum (Vitalik Buterin, 2013)

Ethereum is an open-source, globally decentralized, programmable blockchain platform that introduces a Turing-complete state machine (the Ethereum Virtual Machine - EVM) enabling developers to deploy and execute immutable Smart Contracts and Decentralized Applications (DApps).

Externally Owned Accounts (EOAs) vs. Contract Accounts

Comparison Dimension	Externally Owned Account (EOA)	Smart Contract Account
Control Mechanism	Controlled by a private cryptographic key held by a human user.	Controlled autonomously by its compiled deployed bytecode logic.
Storage & Code	Contains ZERO code and zero internal storage (Balance + Nonce only).	Contains EVM executable bytecode and a permanent persistent key-value Storage Root.
Transaction Initiation	Can actively initiate transactions and state changes on the network.	Passive: can only execute and send internal messages when triggered by an EOA or another contract.
Creation Cost	Completely free (derived offline via elliptic curve math).	Requires paying network Gas fees to deploy bytecode into global state.

4.2 The Ethereum Virtual Machine (EVM) & Gas Economics

EVM Architectural Components

- *Stack: 1024-element deep Last-In-First-Out (LIFO) stack with 256-bit word size.*
- *Memory: Ephemeral, byte-addressable volatile memory initialized to zero per transaction.*
- *Storage: Persistent, expensive key-value mapping (2^256 -> 2^256 words) stored permanently on global disk trie (SSTORE / SLOAD opcodes).*
- *ROM / Code: Immutable bytecode containing executable contract instructions.*

Gas Economics & The EIP-1559 Fee Mechanism

Because the EVM is Turing-complete, it is mathematically vulnerable to the Halting Problem (an infinite loop crashing validator nodes). Gas solves this by charging a computational execution fee for every single opcode:

- **1. Gas Formula:** Total Transaction Fee = Gas Units Consumed * Gas Price (in Gwei, where 1 ETH = 10^9 Gwei = 10^18 Wei).
- **2. EIP-1559 Modern Fee Structure:**
  - Base Fee: Dynamically adjusted by network block utilization; permanently BURNED (destroyed), making ETH deflationary under high network demand.
  - Priority Fee (Tip): Paid directly to validators to prioritize inclusion in the upcoming block.
- **3. Out of Gas (OOG) Exception:** If a transaction exceeds its user-specified Gas Limit, execution immediately halts, the entire state reverts, but ALL consumed gas is forfeited to validators to penalize spam/infinite loops.

4.3 Smart Contracts & Solidity Programming

Definition: Smart Contracts (Nick Szabo, 1994)
A Smart Contract is a self-executing, tamper-proof computer program stored directly on the blockchain whose contractual terms and execution logic are enforced autonomously by consensus rules without human intermediary intervention.

Core Syntax & Building Blocks of Solidity

- **1. Variable Types:** uint256 (unsigned integer), address (20-byte Ethereum account), bool, string, mapping(key => value), struct (custom data structure).
- **2. Visibility Modifiers:**
  - public: Accessible both internally and externally via RPC calls.
  - private: Accessible ONLY within the defining contract.
  - external: Accessible ONLY from external transactions/contracts.
  - internal: Accessible within defining contract and derived child contracts.
- **3. State Mutability:** view (reads state, no modification), pure (neither reads nor writes state), payable (can receive native Ether).
- **4. Error Handling:** require(condition, 'Error message') (reverts and refunds remaining gas), revert(), assert() (used for internal invariants; burns gas).

Standard Token Interfaces: ERC-20 vs. ERC-721 (NFT)

Token Standard	Nature & Fungibility	Core Interface Functions	Major Applications
ERC-20	Fungible (Identical & Interchangeable: 1 Token A = 1 Token A).	totalSupply(), balanceOf(account), transfer(to, amount), approve(spender, amount), transferFrom(from, to, amount).	Utility tokens, stablecoins (USDT, USDC), governance tokens (UNI, AAVE).
ERC-721	Non-Fungible (Unique & Indivisible with distinct uint256 tokenId).	ownerOf(tokenId), transferFrom(from, to, tokenId), approve(to, tokenId), tokenURI(tokenId).	Digital art, gaming assets, domain names (ENS), real estate property deeds.

4.4 DApps, IPFS Decentralized Storage & Web3.js

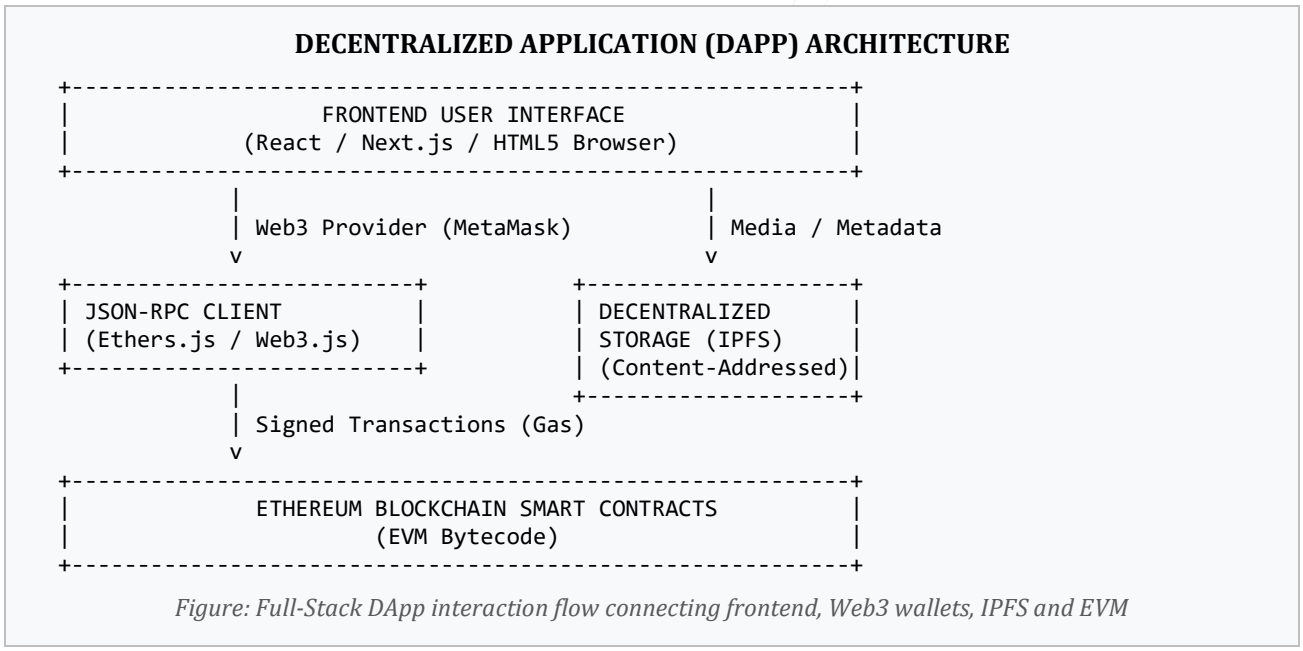

Figure: Full-Stack DApp interaction flow connecting frontend, Web3 wallets, IPFS and EVM

InterPlanetary File System (IPFS)

Storing large files (images, videos, PDFs) directly in Ethereum storage is prohibitively expensive ($ millions per gigabyte). IPFS provides decentralized off-chain storage:

- **Content-Addressing (CID):** Files are addressed by their cryptographic hash (Content Identifier CID, e.g., QmXoypiz...) rather than physical location (URLs). Files are immutable; changing one byte alters the CID.
- **Storage Mechanics:** Uses Kademlia Distributed Hash Tables (DHT) for peer discovery and BitSwap peer-to-peer data chunk exchange.

Web3.js / Ethers.js Fundamentals

- **1. Provider:** Read-only abstraction connecting to the blockchain node (e.g., Infura, Alchemy, window.ethereum).
- **2. Signer:** Abstraction of an Ethereum account capable of signing messages and sending transactions that alter state.
- **3. Application Binary Interface (ABI):** A JSON-formatted specification describing contract functions, parameters, and return types, enabling frontend libraries to encode/decode EVM calls.

4.5 High-Yield University Exam Question Bank

Part A: Short Answer Questions (2 to 5 Marks)

Q1. What is the Ethereum Virtual Machine (EVM)? Why is Gas required? [3-4 Marks]

- **Answer:** The EVM is a stack-based, Turing-complete virtual runtime environment executing smart contract bytecode across all Ethereum nodes. Gas is a metered execution fee that solves the Halting Problem, preventing infinite loops and spam from crashing the network.

Q2. Differentiate between ERC-20 and ERC-721 token standards. [3 Marks]

- **Answer:** ERC-20 defines interchangeable, fungible tokens where every unit is identical (e.g., currencies). ERC-721 defines non-fungible tokens (NFTs) where every token has a globally unique uint256 tokenId and distinct metadata URI.

Q3. What is IPFS and why is it used alongside Ethereum DApps? [3 Marks]

- **Answer:** IPFS (InterPlanetary File System) is a decentralized, peer-to-peer, content-addressed storage protocol. It is used to store large DApp media and NFT files cheaply off-chain, storing only the short 32-byte IPFS hash (CID) on the expensive Ethereum blockchain.

Part B: Long Answer Questions (10 to 15 Marks Outline Guides)

- **Q4. Explain the Ethereum architecture: EOAs vs Contract accounts, EVM Stack/Memory/Storage components, and the EIP-1559 Gas fee mechanism. [12 Marks]**
- **Q5. Write a simple Solidity Smart Contract for an ERC-20 token or Crowdfunding escrow. Explain visibility modifiers, require statements, and events. [12 Marks]**
- **Q6. Describe the end-to-end process of writing, compiling, deploying, and testing a smart contract using Remix IDE and MetaMask. [10 Marks]**

- **Q7. Detail the complete architecture of a Decentralized Application (DApp) integrating React frontend, Web3.js/Ethers.js, MetaMask, Smart Contracts, and IPFS. [12 Marks]**

4.6 Unit Summary & Key Takeaways

- Ethereum extends blockchain with a Turing-complete state machine (EVM) for decentralized smart contracts.
- Accounts consist of Externally Owned Accounts (EOAs - private keys) and Contract Accounts (immutable bytecode).
- EVM uses Stack, Memory, and persistent Storage, metered via Gas (EIP-1559 base fee burning).
- Solidity is the primary language for smart contracts, featuring strict visibility modifiers, events, and error handling.
- Token standards include ERC-20 (fungible currencies) and ERC-721 (non-fungible NFTs).
- DApps combine frontend UIs, Web3 RPC client libraries (Ethers.js/Web3.js), MetaMask signers, and IPFS decentralized content-addressed storage.