Centurion University
Shaping Lives... Empowering Communities...

School: ..... Campus: .....

Academic Year: ..... Subject Name: ..... Subject Code: .....

Semester: ..... Program: ..... Branch: ..... Specialization: .....

Date: .....

Applied and Action Learning

(Learning by Doing and Discovery)

Name of the Experiement : Multi-Chain Deploy – BSC or Layer 2 Experience

Objective/Aim:

To deploy, test, and compare a simple smart contract on two different blockchain environments — Polygon (Layer 2) and Arbitrum — to analyze differences in gas cost, transaction confirmation time, and deployment procedures.

Apparatus/Software Used:

- Hardhat Development Framework
- Solidity Compiler (v0.8.x)
- MetaMask Browser Wallet
- Polygon Mumbai Testnet RPC Endpoint
- Arbitrum Sepolia Testnet RPC Endpoint
- Ethers.js Library
-

Theory/Concept:

- **Multi-Chain Deployment:**
  Refers to deploying identical smart contracts on multiple blockchain networks to expand interoperability, reduce costs, and improve user experience
- .**Network Layers:**

  Layer 1 (Ethereum, BNB Chain): The base blockchain responsible for direct transaction processing and finality.

  Layer 2 (Polygon, Arbitrum, Optimism): Built on top of Layer 1 to offload transaction computation and achieve higher throughput .

- **Consensus Mechanisms:**
  Polygon operates on **Proof of Stake (PoS)**, while Arbitrum uses **Rollup technology** that batches transactions to Ethereum for security and validation.
- **Gas Fees:**
  Each network determines gas fees based on congestion, validator structure, and block time. Layer 2 networks generally provide faster and more affordable transactions compared to Ethereum mainnet.

Procedure

- **Project Setup**
  - Install Hardhat:
    ```
    npm install --save-dev hardhat
    npx hardhat
    ```
  - Initialize a new project and create a `contracts` folder.
- **Create a Smart Contract (Example: Counter.sol)**
  - Write a simple counter contract with `increment()` and `getCount()` functions.
- **Configure Networks** in `hardhat.config.js`:
  - Add Polygon Mumbai and Arbitrum Sepolia network configurations with private keys and RPC URLs.
- **Compile and Deploy the Contract:**
  - Polygon:
    ```
    npx hardhat run scripts/deploy.js --network polygonMumbai
    ```
  - Arbitrum:
    ```
    npx hardhat run scripts/deploy.js --network arbitrumSepolia
    ```
- **Verify Deployments:**
  - Record contract addresses.
  - Verify on:
    - PolygonScan Mumbai
    - Arbiscan Testnet
- **Compare Network Metrics:**
  - Use MetaMask to track transaction speed and gas usage.
  - Note differences in block confirmation time.
  - Compare total deployment cost and efficiency.
- **Interaction Testing:**
  - Call `increment()` function on both networks.
  - Observe updated state values and verify synchronization.

Observation Table:

Network	Avg. Gas Fee (Gwei) Txn	Confirmation Time	Contract Address
Polygon Mumbai	~0.001	~2–3 sec	0x1234...abcd
Arbitrum Sepolia	~0.003	~1–2 sec	0x5678...efgh

ASSESSMENT

Rubrics	Full Mark	Marks Obtained	Remarks
Concept	10		
Planning and Execution/ Practical Simulation/ Programming	10		
Interpretation Result and	10		
Record of Applied and Action Learning	10		
Viva	10		
Total	**50**	.	

Signature of the Faculty:

Signature of the Student:
Name
Regn.No.