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

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

ALGORITHM:

```
1.Set Up Multi-Chain Environments
    • Install Hardhat for EVM-based deployment (BSC / Polygon / Optimism).
    • Configure MetaMask wallet for Binance Smart Chain (BSC) or Layer 2 networks.
2.Write Smart Contracts
    • Create a Solidity smart contract with simple token or data storage logic.
    • Use the same contract to deploy across multiple chains for gas and performance
      comparison.
3.Compile Contracts
    • Run npx hardhat compile to check for syntax or version errors.
    • Ensure compatibility with Solidity compiler version (e.g., ^0.8.0).
4.Configure Networks
    • Add RPC URLs and chain IDs for selected networks in hardhat.config.js.
5.Deploy on Testnets
    • Use deployment script:
          npx hardhat run scripts/deploy.js --network bscTestnet
          npx hardhat run scripts/deploy.js --network polygonMumbai
6.Verify and Test Functionality
    • Check deployed contract addresses on BscScan or PolygonScan.
    • Execute test functions to ensure identical logic execution across chains.
7.Compare Cross-Network Performance
    • Record gas usage, confirmation time, and transaction cost across networks.
    • Evaluate scalability and efficiency improvements on Layer 2 or sidechain solutions.
```

* Software used

1.Hardhat
2.MetaMask
3.Solidity
4.Web3.js / Ethers.js
5.BSC Testnet RPC / Polygon Mumbai RPC

**As applicable according to the experiment. Two sheets per experiment (10-20) to be used.*

* Implementation Phase: Final Output (no error)

Objective:

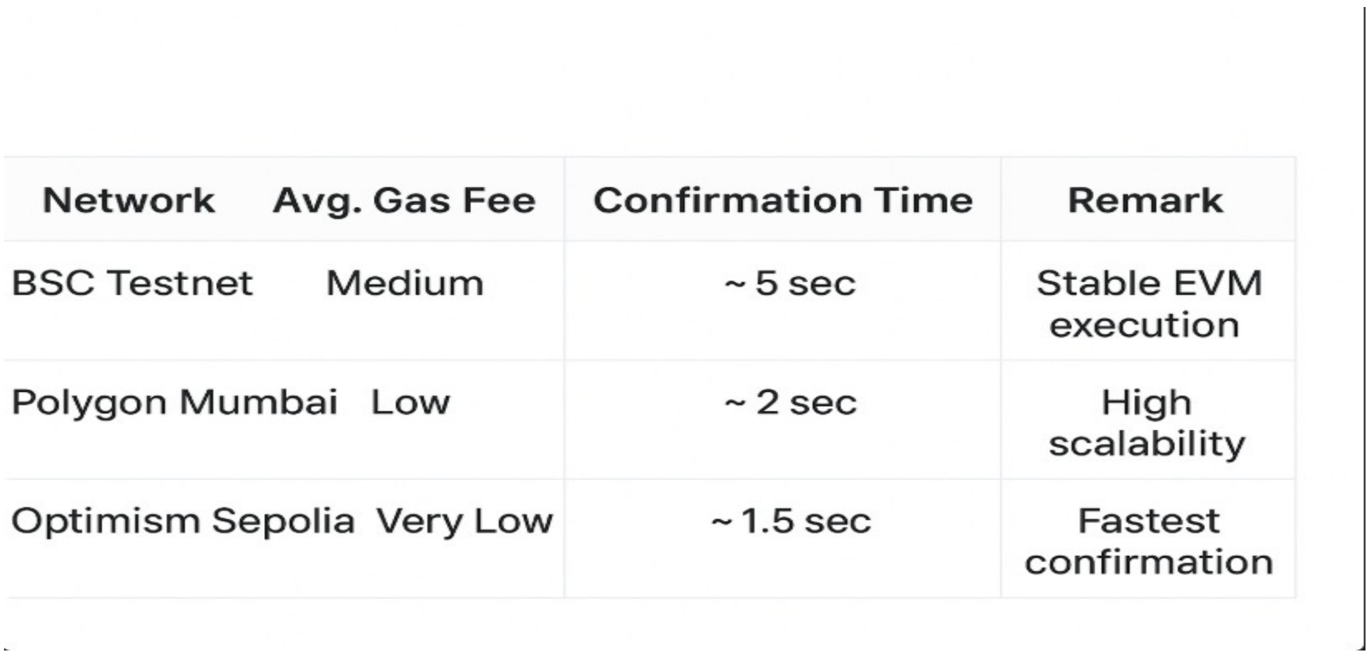
To deploy and test Solidity-based smart contracts on Binance Smart Chain (BSC) and Layer 2 networks such as Polygon or Optimism, demonstrating interoperability, scalability, and efficiency improvements in multi-chain environments.

Steps:

1.Environment Setup – BSC (EVM-Compatible):
    Installed Hardhat and configured BSC Testnet RPC in hardhat.config.js.
    Deployed a Solidity smart contract using MetaMask-connected wallet.
    Verified transaction status on BscScan Testnet Explorer.
2.Environment Setup – Layer 2 (Polygon / Optimism):
    Configured the Polygon Mumbai or Optimism Sepolia network in Hardhat.
    Deployed the same contract for performance benchmarking.
    Verified successful deployment on respective explorers (PolygonScan / Optimistic Etherscan).
3.Testing and Cross-Network Validation:
    Executed identical contract functions on both networks to confirm functional consistency.
    Measured average gas fee and transaction confirmation time.
    Observed that Layer 2 transactions were cheaper and confirmed faster compared to BSC Testnet.

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

* Implementation Phase: Final Output (no error)



Network	Avg. Gas Fee	Confirmation Time	Remark
BSC Testnet	Medium	~ 5 sec	Stable EVM execution
Polygon Mumbai	Low	~ 2 sec	High scalability
Optimism Sepolia	Very Low	~ 1.5 sec	Fastest confirmation

* Observations

1.Multi-chain deployment improves scalability and reduces congestion on Ethereum Mainnet.
2.Layer 2 networks like Polygon and Optimism significantly lower transaction costs and enhance user experience.
3.BSC provides high throughput and easy integration with existing EVM-compatible projects.

ASSESSMENT

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

Signature of the Student:

Name :

Regn. No. :

Signature of the Faculty:

***As applicable according to the experiment.*
*Two sheets per experiment (10-20) to be used.***