Blockchain Developer Assignment

Assignment Description

Part 1: Smart Contract Development (Core)

- Build a **Staking Contract** in Solidity with the following features:
    - Users can stake an ERC-20 token (mock token acceptable).
    - Rewards are distributed proportionally to the amount and duration of staking.
    - Implement penalty logic if a user unstakes before a minimum lock period.
    - Ensure gas-efficient reward calculation (no looping over all stakers).
    - Security: Prevent reentrancy and overflow issues.

Part 2: Smart Contract Security & Optimization

- Write unit tests for the smart contract (using Hardhat/Foundry).
- Demonstrate **at least 2 optimizations** for gas usage.
- Add comments in the code explaining **potential vulnerabilities** and how you mitigated them.

Part 3: Integration

- Create a minimal backend or frontend (Node.js/React) to:
    - Connect to the contract.
    - Allow users to stake, unstake, and claim rewards.
    - Display current staking balance and pending rewards.

Part 4: System Design (Short Answer)

- Write a 1–2 page document explaining:
    - How would you scale this staking system if the number of users grows 100x?

- How would you adapt this for a **multi-chain environment** (Ethereum + Polygon)?
- What monitoring/logging mechanisms would you add for production?

Deliverables

1. Solidity code for the staking contract.
2. Unit tests (minimum 5 meaningful test cases).
3. Frontend/backend integration (basic UI or API is fine).
4. System design document.
5. A short **README** with setup/run instructions.

Evaluation Criteria

- **Correctness**: Contract works as expected, test cases pass.
- **Security**: Awareness of vulnerabilities (reentrancy, overflow, replay).
- **Optimization**: Efficient storage and gas usage.
- **Code Quality**: Readable, well-structured, documented.
- **System Thinking**: Ability to think beyond coding (scalability, multi-chain, monitoring).