Activity

Build a Voting DApp using Hardhat and React

🎯 Objective

Develop a decentralized **Voting Application (DApp)** that allows users to vote for candidates in a transparent and tamper-proof way using **Ethereum smart contracts** deployed via **Hardhat**, with a **React frontend** connected to MetaMask.

⚙️ Core Requirements

Smart Contract (Solidity)

Each student must:

1. Create a unique contract named after their own name or roll number e.g.: contract Voting_YourName { ... }
2. The contract should:
   - Allow an **admin (deployer)** to add candidates.
   - Allow each **address to vote only once**.
   - Store all votes on-chain.
   - Provide a **function to view results**.
   - Prevent voting once the election ends.

💻 Frontend (React + MetaMask)

Students must build a **simple React app** that:

1. Connects to **MetaMask** (via ethers.js).
2. Displays a **list of candidates** fetched from the contract.
3. Allows the user to:
   - **Add candidates** (if admin)
   - **Vote** for one candidate
   - **View live results**

4. Shows wallet address and current voting status.

Personalization (Anti-Plagiarism Rule)

Each student must:

- Name their contract as Voting_<StudentName>
- Display your name in the React app footer.

Features

- Allow users to **view remaining voting time** (countdown).
- Add **event logs** for “VoteCast” and “VotingEnded”.
- Display **winner** automatically after voting ends.
- Deploy on a testnet.

🧩 **Deliverables**

1. Hardhat project folder (contracts, deployment scripts).
2. React project folder (frontend).
3. Screenshots:
    - Contract deployed
    - Voting process
    - Result display