**Short Write-Up**

**a. Where enforcement was applied**

* **Cap**: The token cap (maximum supply) was enforced in the ERC-20 contract logic. This ensured no minting beyond the limit defined in .env (TOKEN\_CAP=2000000).
* **Pause**: A pause mechanism was added so transfers and approvals can be temporarily disabled by the admin if needed. This provides control in emergencies or upgrades.
* **Roles**: Specific roles were assigned (e.g., **owner/admin**). Only authorized accounts can mint tokens, pause/unpause the contract, or manage special functions. This prevents unauthorized token manipulation.

**b. Why batch airdrop saved gas**

In the experiment:

* **Single transfers gas**: 67,226
* **Batch airdrop gas**: 51,216
* **Gas saved**: ~23.82%

Batching reduces repeated overhead (like initialization, event emissions, storage updates) by grouping multiple transfers into one transaction. Instead of paying the fixed base transaction cost (21,000 gas) for each individual transfer, batch airdrop consolidates this cost across multiple recipients. That explains the ~24% savings.

**c. Issues faced and fixes**

1. **Invalid address errors**
   * Problem: Mixed-case or incorrect checksum caused InvalidAddressError.
   * Fix: Updated .env addresses to proper checksum format (EIP-55).
2. **BigInt parsing error in transfer script**
   * Problem: parseUnits failed when a BigInt was passed instead of a string.
   * Fix: Converted all amounts to strings before calling parseUnits.
3. **Logs query block range error**
   * Problem: Initially queried from a negative block (-1994).
   * Fix: Adjusted logic to query from Math.max(0, currentBlock - 2000).
4. **MetaMask custom network not connecting**
   * Problem: Used 127.0.0.1:8545 but MetaMask requires a hostname.
   * Fix: Used http://localhost:8545 or team RPC (hh-12.didlab.org).
5. **Token not auto-detected in MetaMask**
   * Problem: Local network doesn’t support automatic token detection.
   * Fix: Manually imported the deployed TOKEN\_ADDRESS.