FINAL EXAMINATION PROJECT

Course: *Blockchain 1*

Format: Group Project

Technology Stack: Solidity, JavaScript, MetaMask, Ethereum (Testnet)

RULES:

1) ***NO LATE SUBMISSION***
2) ***Upload PDF documentation with github link with code***
3) ***Upload Presentation***
4) ***Each member of the team needs to upload***

1. Purpose of the Final Project

CHOOSE a free topic!

The purpose of the final examination project is to comprehensively assess students' theoretical knowledge and practical skills in blockchain technology, smart contract development, and decentralized application (DApp) engineering.

The project evaluates the student's ability to:

- design and implement smart contracts using Solidity;
- develop client-side blockchain interaction using JavaScript;
- integrate decentralized applications with the MetaMask wallet;
- work with Ethereum test networks and token standards;
- apply architectural principles of decentralized applications.

2. Project Overview

Students are required to develop a decentralized crowdfunding application operating exclusively on an Ethereum test network and utilizing free test tokens only.

The developed application must enable:

- creation of crowdfunding campaigns;

- participation of users as contributors;
- issuance of internal reward tokens for contributions;
- secure interaction with the blockchain via MetaMask.

The use of real cryptocurrency or Ethereum mainnet is strictly prohibited.

3. Functional System Requirements

3.1 General Requirements

The project must:

- operate exclusively on an Ethereum test network (Sepolia, Holesky, or local network);
- integrate with the MetaMask wallet;
- use only free test ETH and test tokens;
- demonstrate real blockchain interaction.

3.2 Smart Contract Requirements

The smart contracts must provide the following functionality:

1. Creation of crowdfunding campaigns with defined parameters:
    - campaign title;
    - funding goal;
    - campaign duration or deadline.
2. Contribution of test ETH to active campaigns.
3. Accurate tracking of individual contributions.
4. Finalization of campaigns upon reaching the deadline.
5. Issuance of internal reward tokens proportional to user contributions.

3.3 Tokenization Requirements

The project must include a custom ERC-20 token that:

- is minted automatically during user participation;
- has no real monetary value;
- is used solely for educational purposes;
- demonstrates core tokenization concepts.

3.4 Frontend Requirements

The client-side application must:

- connect to MetaMask;
- display the connected wallet address;
- verify the selected blockchain network;
- allow users to:
    - create campaigns;
    - contribute to campaigns;
    - monitor transaction outcomes;
- display user balances of test ETH and reward tokens.

4. MetaMask Integration Requirements

MetaMask integration is mandatory.

The application must demonstrate:

- requesting user permission to access wallet accounts;

- validation of the active Ethereum test network;
- execution of blockchain transactions through MetaMask.

5. Project Documentation Requirements

Students must submit technical documentation that includes:

1. Overview of the application architecture.
2. Explanation of design and implementation decisions.
3. Description of smart contract logic.
4. Explanation of frontend-to-blockchain interaction.
5. Deployment and execution instructions.
6. Description of the process for obtaining test ETH.

6. Academic and Technical Constraints

- Deployment on Ethereum mainnet is strictly forbidden.
- Use of real cryptocurrency is not allowed.
- The project must be completed individually.
- Plagiarism and reuse of ready-made solutions are prohibited.

7. Assessment Criteria

The final project will be evaluated based on the following criteria:

Criterion	Points
Smart contract implementation	24
Correct crowdfunding logic	6
ERC-20 token usage	6

MetaMask integration	6
Test network usage	6
Quality of documentation	6
Architecture and code structure	6
Total	60

40 Points is defence !

Best regards and Good luck !