Sandbox Blockchain technical tests

General Questions

A) In Ethereum and blockchain ecosystem in general, what are the use-cases of hash functions?

B) What are the bottlenecks in terms of performance when operating on a network like Ethereum? What kind of solutions can be utilized to overcome them?

C) What are the different types of bridges and how do they work ?
Explain in detail step by step how bridging an ERC721 would work.

D) Describe the EIP-2771 standard with your own words and describe some use cases using this EIP.

E) What are the centralization issues to be aware of when developing a smart contract ? Propose a technical solution for each of them.

F) What process (step by step) do you follow to build a protocol (a set of smart contracts) from given specifications ?

Solidity Questions

G) After analyzing the file “Signature.sol”, describe the use case of this contract, how to use it and all the technical steps (off-chain & on-chain) & key methods of the contract.

H) Perform an audit of the contract “Staking.sol”, find at least 3 technical technical, logic issues or hacks, explain why it is an issue and provide a way to fix those.

I) How would you build a proxy contract where the implementation lives in another contract in solidity (do not worry about syntax error or mispel)

J) In this exercise, you have been given specifications and your goal is to develop a smart contract that satisfies those. This contract should be named “CampaignSale.sol” and implements the interface “ICampaignSale.sol” included in this test. This interface includes comments that should help you build your contract. Here are the specifications:

We are building a fundraising dApp that enables creators to publish their projects in order to raise money in the form of ERC20 tokens. A campaign lasts for a duration and has to raise a minimum goal to unlock the development of the project & the tokens invested by contributors. Otherwise, the contributors can get refunded.

Launching a campaign
A campaign can be created by any user and should include:

- the starting date of the campaign that should be in the future
- the ending date of the campaign that should be in the future, greater than the starting date. Also a campaign should last at max 90 days.
- The goal to reach (token amount)

Canceling a campaign
A campaign can be canceled before it starts. The campaign is then no longer accessible and should be deleted. Only the creator of the campaign can cancel it

Contributing to a campaign
A user can contribute to the campaign by sending an amount of tokens to the contract only if the campaign is running.

Withdrawing from a campaign
A user can withdraw an amount of token from a campaign only if the campaign is running. Then, that amount of tokens is sent back to the user.

Claiming the token from a campaign
Once a campaign is over, the creator of the campaign is able to claim all the tokens only if the goal of the campaign has been reached. A campaign can be claimed only once.

Refunding a campaign
If the campaign didn't reach the goal after the campaign is over, the contributors can still get refunded. A contributor will receive back his whole contribution (tokens) to the campaign.