Lab project : Blockchain4Coffee is back

We want to develop a smart contract `Block4Coffee` for managing coffee machines which are publicly available. The coffee machines are connected to the internet and to the contract to manage credits and expenses of users and to manage coffee refill. There are three different kinds of users interacting with the contract : coffee drinkers (simply called users in the following), coffee providers and the contract owner.

This project will be done in two steps :

1. You develop a `Block4Coffee` contract and deploy it as a **verified contract** on Goerli. In moodle, you provide the source code of the contract and its URL on Goerli.
2. In moodle you receive the source and URL of several contracts to audit, verify and attack. For each contract, in Moodle, you report if it satisfies properties of Section 2.

1 The `Block4Coffee` functions

Users interact with the contract by calling :

— the `sendMoney` function to add credit to their account ;

— the `getMoneyBack` function to get back all the available money on their account ;

— the `buyCoffee` function to buy a coffee. This function is called by the user. This operation retracts the price of a coffee to the user's account. A maximal debt of 10 Gwei is authorized on the account of each user.

The owner of the contract, can also interact with the contract by calling :

— the `addCoffeeProvider` function which adds an address as a coffee provider ;

— the `fixCoffeePrice` function which fixes the selling price of a coffee ;

— the `changeOwner` function which changes the address of the owner of the contract.

Finally, coffee providers can also interact with the contract by calling :

— the `addCoffee` function to add coffee in the machine. In practice they add coffee in the machine and call this function to declare how many coffee units they added and provide the proof that they did buy the coffee from a shop. [1] They also receive money back : they are refunded for the coffee they bought.

2 Contract properties

Here are the properties that should be ensured by the `Block4Coffee` contract. During the evaluation step you will have to check if those properties are satisfied by the contracts of your mates.

Assertion 1 : A user can send money to its account in the contract. The amount of sent money is added to its account.

Assertion 2 : A user can get back the money available in its account.

Assertion 3 : A user should not be abble to get back money if he has a debt.

Assertion 4 : A user can buy a coffee. This operation retracts the price of a coffee to the user's account.

Assertion 5 : A maximal debt of 10 Gwei is authorized on the account of each user.

Assertion 6 : Only coffee providers can add coffee units in the contract.

Assertion 7 : When a coffee provider adds coffee units in the contract, she also provides the proof that she effectively bought coffee from a shop.

1. e.g. a string containing the URL or a hash of the scanned ticket.

Assertion 8 : When a coffee provider adds coffee units in the contract he also receives money from the contract.

Assertion 9 : Only the owner of the contract can declare a new coffee provider.

Assertion 10 : Only the owner of the contract can change the selling price of one coffee.

Assertion 11 : Only the owner of the contract can change the ownership of the contract.

Assertion 12 : The money cannot be drained from the contract by **users**, which are not owner or coffee provider, using a reentrancy attack. [2]

Assertion 13 : There are no dangerous overflows/underflow errors in the contract.

2. Malicious owner or coffee providers can drain money from the contract using `addCoffee`