SEAS 4.0 – Industry 4.0 Enabling Technologies

Final project – smart contract report

Date: May 16th, 2024

Authors:

Mohammed Ahmed (m.ahmed1@udc.es)

Valentina Román Piedrahita (v.romanp@udc.es)

Devaprasad Puthuvalsthalath Sudheer (devaprasad.sudheer@udc.es)

Instructors:

Professor. Paula Fraga Lamas (paula.fraga@udc.es)

Professor. Tiago Manuel Fernández Caramés (tiago.fernandez@udc.es)

A review report for a smart contract for the insurance claims related to ship flooding detection system

Table of Contents

Report: Insurance Contract for ship flooding detection system

1 Motivation

In our IoT project, we are developing a system of sensors to monitor water levels and temperature in ship compartments. These sensors play a crucial role in detecting potential flooding incidents and abnormal temperature conditions, which are common risks in maritime environments. To mitigate the risks associated with such incidents, it's essential to have an automated insurance claim system that can efficiently process claims based on sensor data.

The insurance claim requests will correlate with the severity of flooding incidents. Sensor data, including water level and temperature readings from flooded compartments, will drive claim initiation. These precise measurements enable ship owners to file claims with unprecedented accuracy. Furthermore, insurance assessments will leverage real-time sensor data for risk evaluation and cost estimation.

This streamlined procedure empowers ship owners to make informed decisions based on comprehensive sensor data, while insurance assessments are fortified with up-to-the-minute information from onboard sensors.

Of course, due to simplicity of the IoT system the smart contract is very simple, but for a big scale system, this smart contract could be very efficient and reliable to avoid much investigations about the flooding events.

2 Case Definition and Objective

2.1 Case Definition

The objective of this smart contract is to facilitate the initiation of insurance claims based on sensor data collected from ship compartments. The contract defines thresholds for water level and temperature, and when these thresholds are exceeded, a claim can be filed.

Authorization check is also required for the claiming process.

2.2 Objective:

1. Develop a smart contract that allows authorized addresses to initiate insurance claims based on sensor data.
2. Define thresholds for water level and temperature to verify insurance claim terms and conditions.
3. Ensure secure and transparent processing of insurance claims on the Ethereum blockchain.

3 Design Aspects of the Smart Contract

3.1 Features:

- Authorization mechanism to control access to claim initiation functionalities.
- Pre-defined thresholds for water level and temperature based on the agreement with the insurance company.
- Functions for checking claim conditions and initiating insurance claims.
- Event logging to record initiated insurance claims on the blockchain.

3.2 Implementation:

- Struct to store sensor data including timestamp, water level, and temperature.
- Mapping to store authorized addresses and track claim initiation permissions.
- Functions for authorizing addresses, checking claim conditions, and initiating claims.

The following figure shows the code of these code functions, struct and mapping:

```
contract InsuranceContract {
    // Structure to store sensor data
    struct SensorData {
        uint256 timestamp;
        uint256 waterLevel; // Assuming water level is represented in some unit
        uint256 temperature; // Assuming temperature is represented in Celsius
    }

    // Pre-defined thresholds for triggering a claim
    uint256 public constant waterLevelThreshold = 20; // Example threshold for water level
    uint256 public constant temperatureThreshold = 17; // Example threshold for temperature

    // Mapping to store authorized addresses
    mapping(address => bool) public authorizedAddresses;

    // Event to log insurance claims
    event ClaimInitiated(address indexed claimAddress, uint256 timestamp);

    // Function to authorize an address
    function authorizeAddress(address _address) public {   24857 gas
        authorizedAddresses[_address] = true;
    }

    // Function to check if claim conditions are met and emit appropriate message
    function checkClaimConditions(uint256 _waterLevel, uint256 _temperature) public pure returns (string memory) {   infinite gas
        if (_waterLevel > waterLevelThreshold && _temperature > temperatureThreshold) {
            return "Claim has been filed, insurence will contact you soon";
        }
        return "Claim could not be filed, please check the insurance policy";
    }

    // Function to initiate an insurance claim
    function initiateInsuranceClaim(uint256 _claimID, address _claimAddress, uint256 _waterLevel, uint256 _temperature) public {   infinite gas
        require(_claimID >= 100000 && _claimID <= 999999, "Invalid claim ID");
        require(authorizedAddresses[msg.sender], "Unauthorized");

        // Check if claim conditions are met
        if (_waterLevel > waterLevelThreshold && _temperature > temperatureThreshold) {
            // Trigger insurance claim
            emit ClaimInitiated(_claimAddress, block.timestamp);
        } else {
            revert("Claim conditions not met");
        }
    }
```

It is clear that this code will implement main parts which are:

1. Struct to store the sensors data.
2. Mapping to store the authorized address.
3. An event to log the claim event.
4. Function to check the authorized address.
5. Function to check the claim conditions according to the agreed thresholds, and based on that a result of compliance or non-compliance with the agreement will pop up.

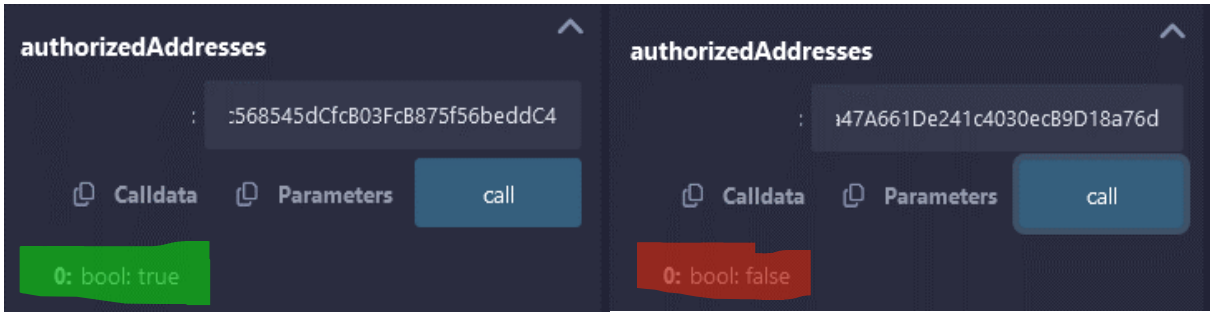
The code has been compiled without any issues and deployed to test and validate it.

4 Testing and Validation

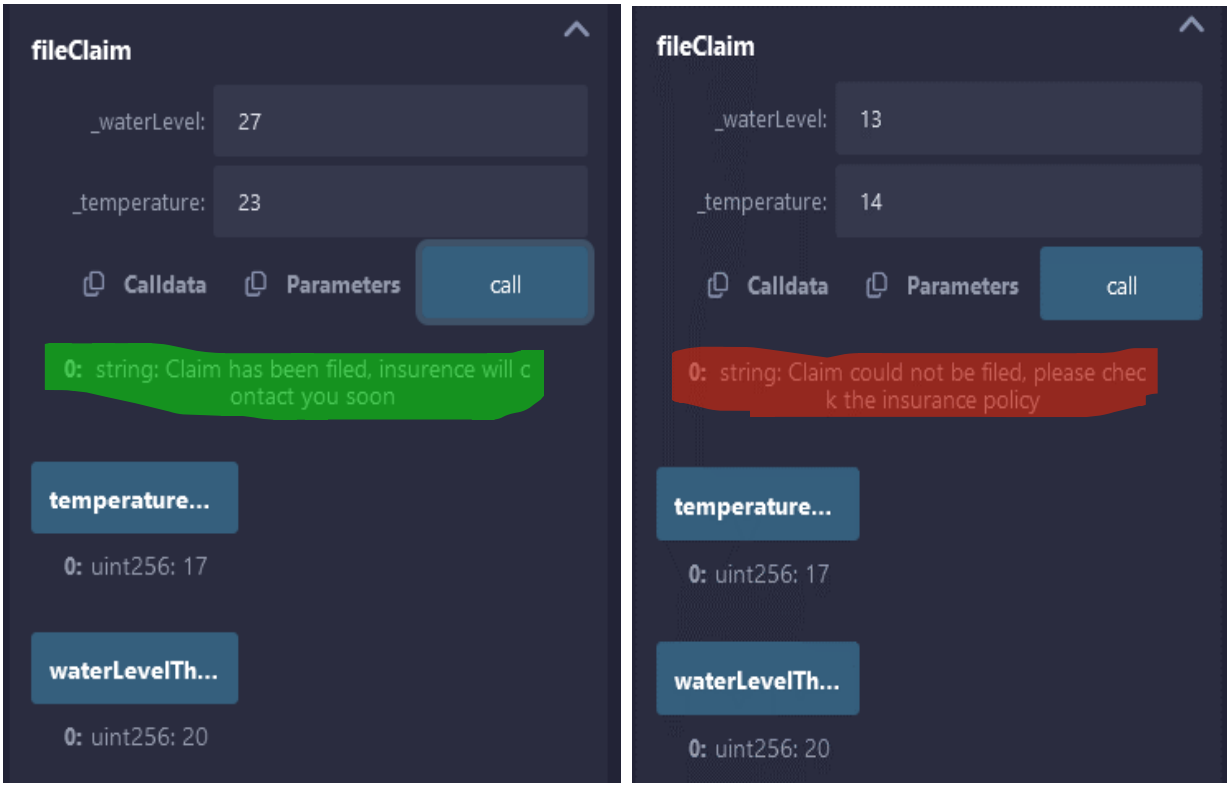
Remix Ethereum IDE has been used for contract development and testing.

4.1 Test Cases

1. Verify authorization mechanism: Ensure that only authorized addresses can initiate insurance claims. The following two pictures show whether the authorized address is true or false.



2. Test claim conditions: Check if claims can be initiated when sensor data exceeds pre-defined thresholds.



Ass seen in these two figures, when the claim conditions are met, the claim will be filed, otherwise it will be rejected automatically.

5 Conclusions and Recommendations

5.1 Conclusions:

- The “InsuranceContract” smart contract provides a robust solution for automating insurance claims based on IoT sensor data in ship compartments.
- Authorization mechanisms and threshold-based claim initiation ensure security, reliability, and integrity in the claim initiation process.

5.2 Recommendations:

- Consider extending the contract to include additional functionalities such as claim processing and settlement.
- Conduct further testing and validation in different deployment environments to ensure contract reliability and performance.
- Increase the complexity of the smart contract by optimizing it to meet the need of more complex IoT flood detection systems.