Ziad Ibrahim elbatal

2205075

Smart Contract Deployment Report

**Problem Statement**

Deploy a contract that:

1. Stores two numbers (A and B)
2. Calculates and emits their sum, difference, product, and quotient
3. Emits the stored values
4. Uses separate functions for each operation

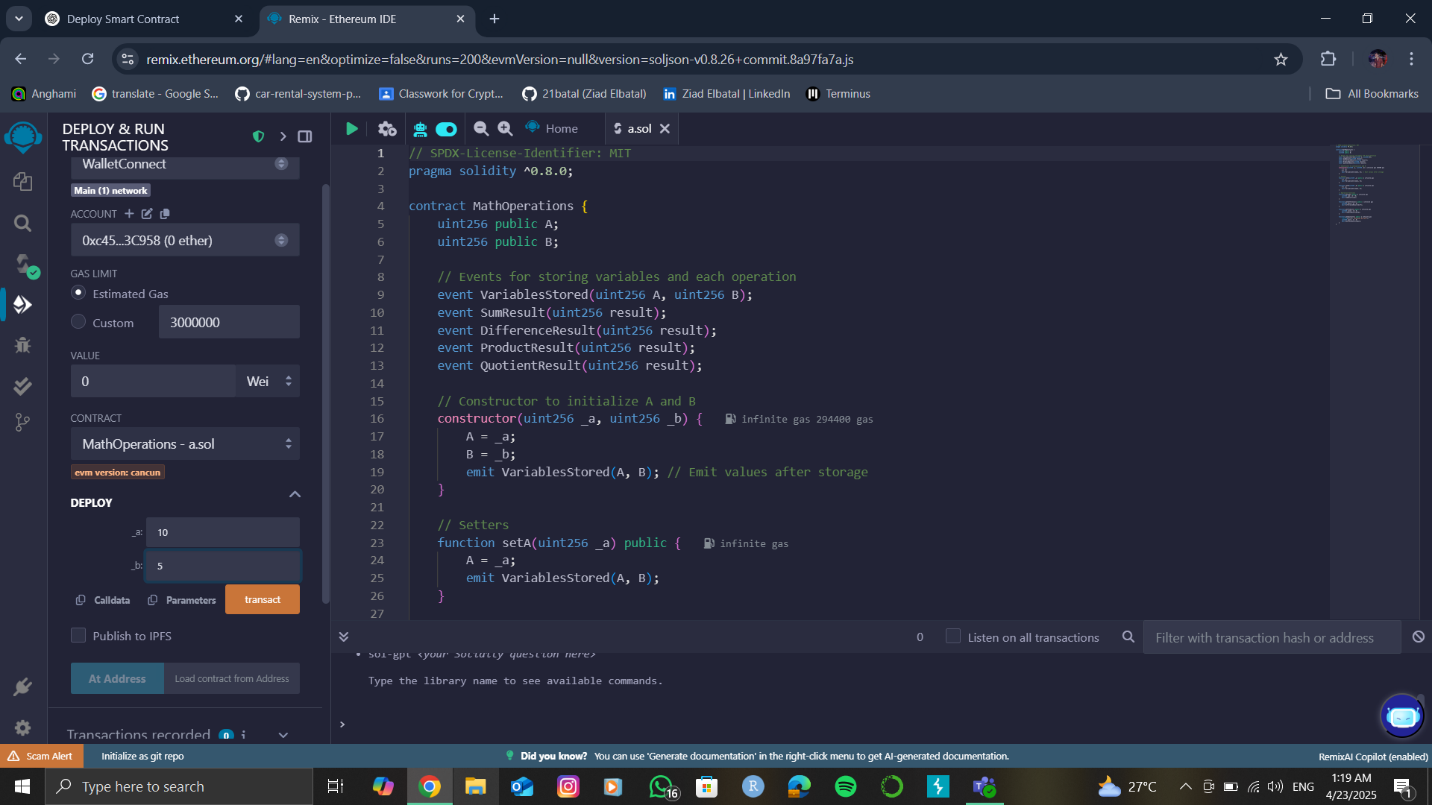
**Solution**

Created MathOperations.sol contract with:

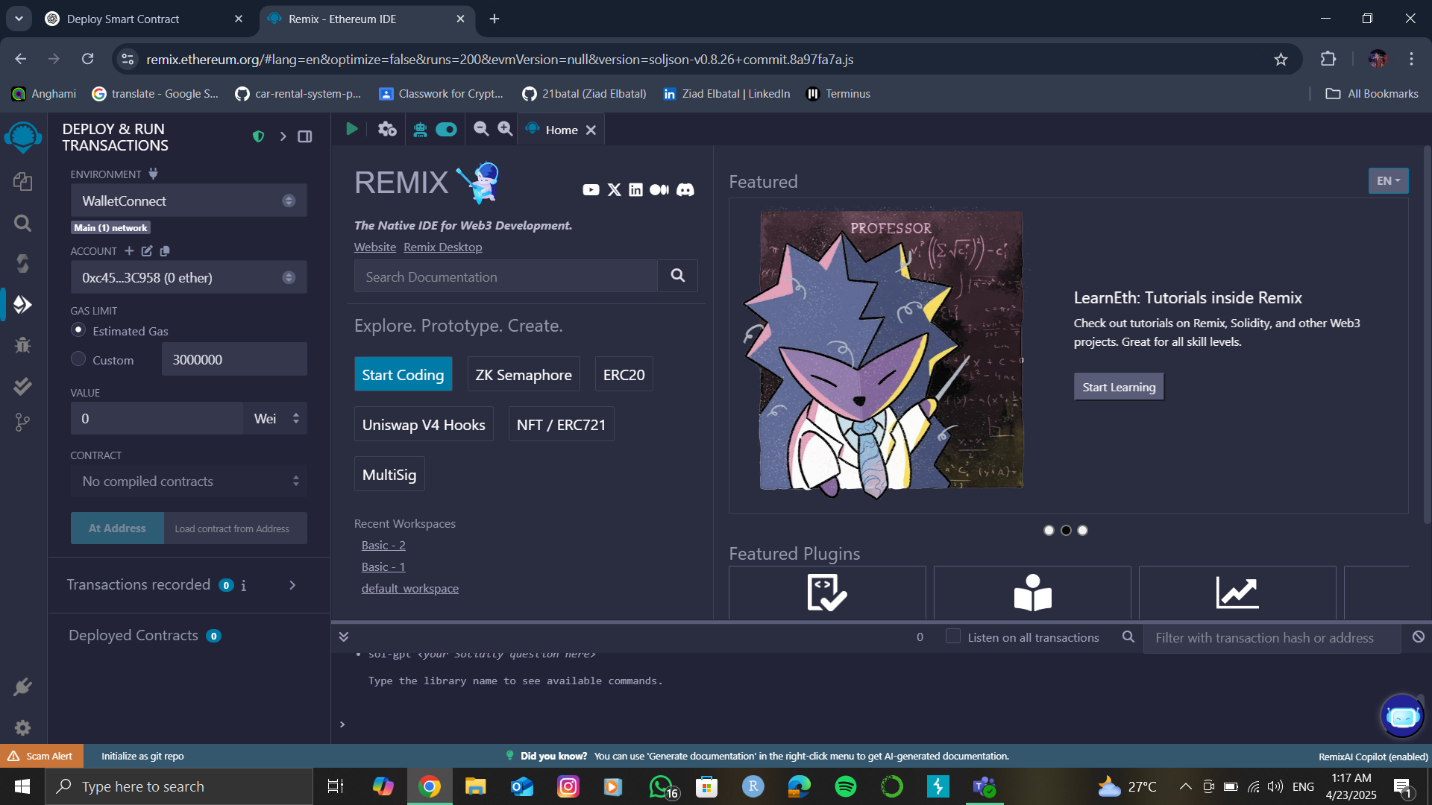
* Variables A and B
* Events for each operation
* Constructor to set initial values
* Functions for all calculations

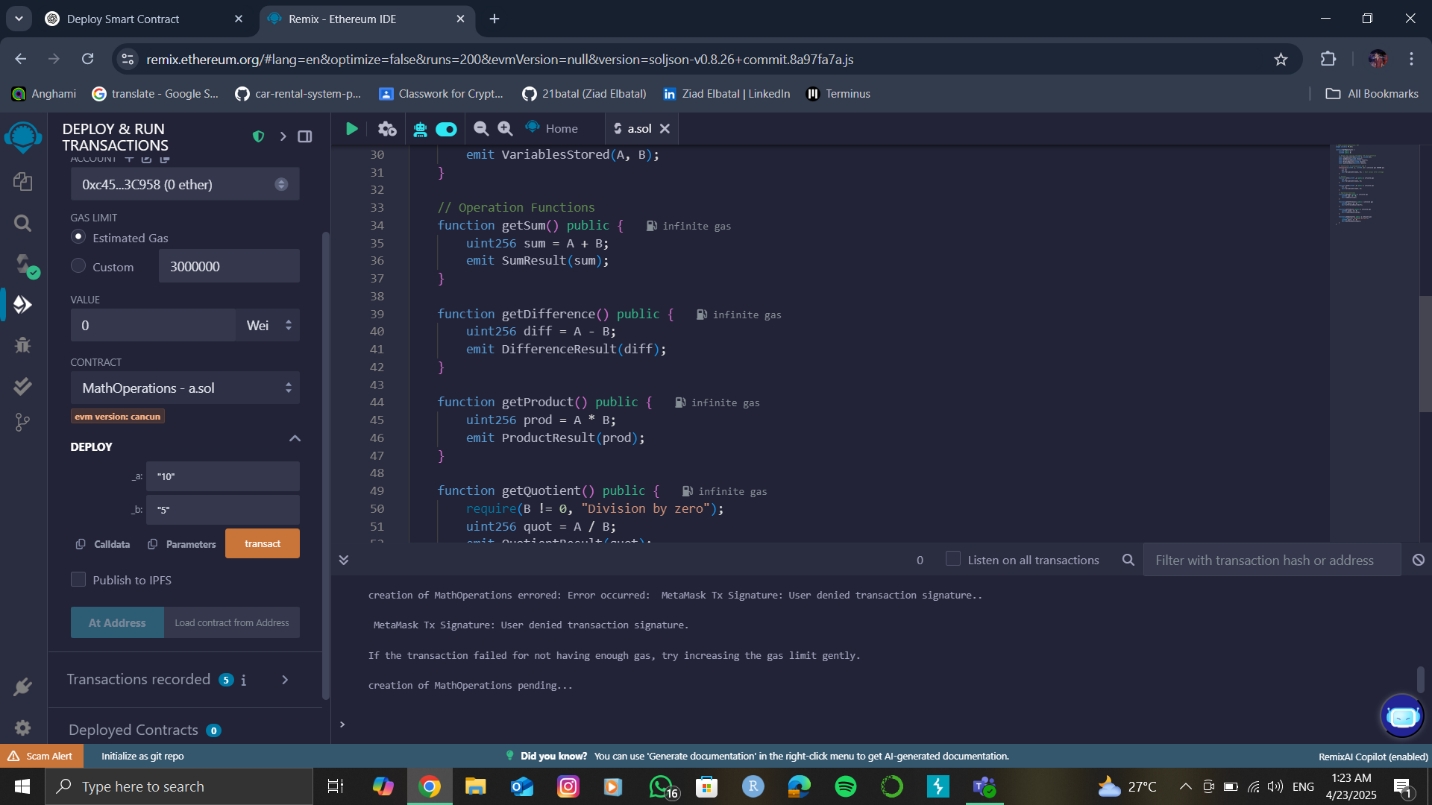
**Deployment Steps**

1. **Wrote the contract** in Remix IDE



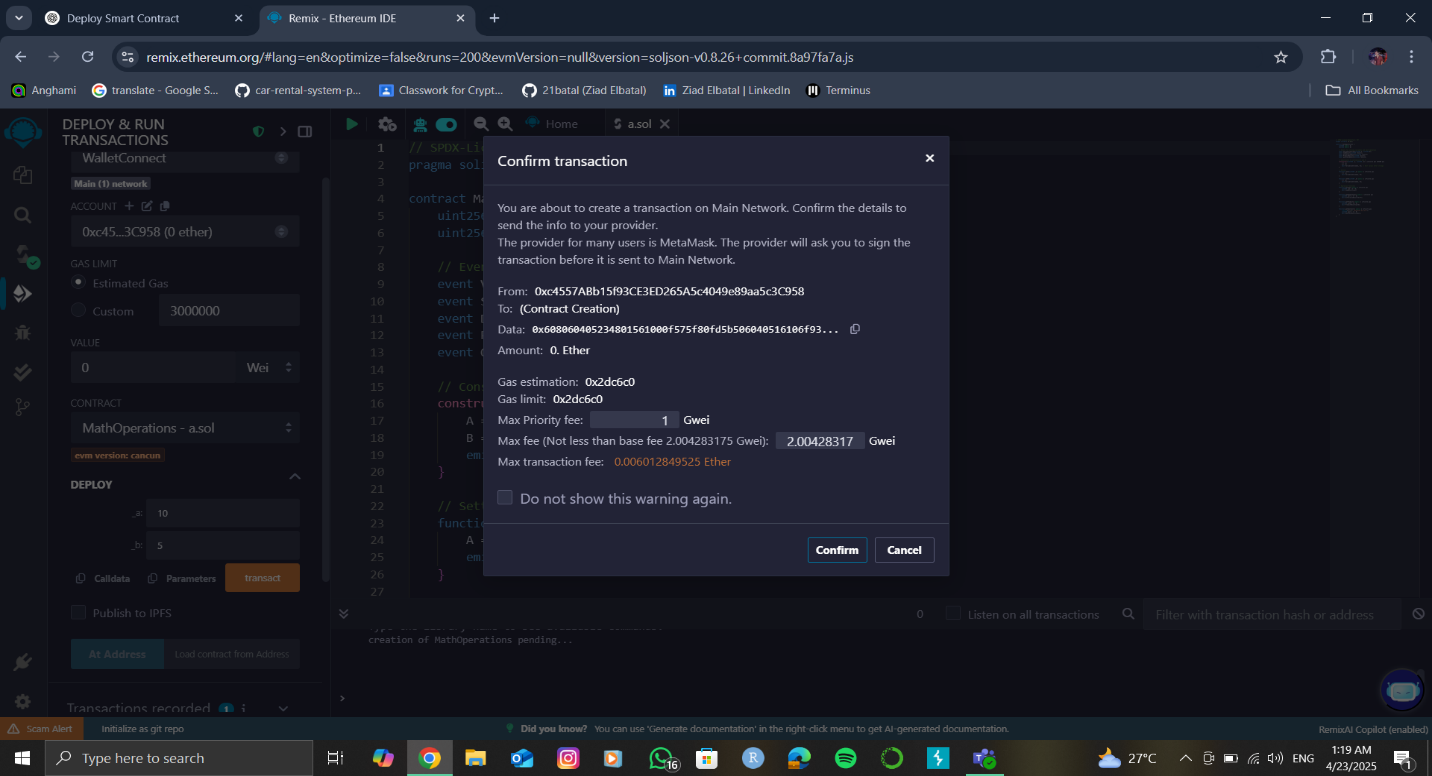
**2.Connected MetaMask**



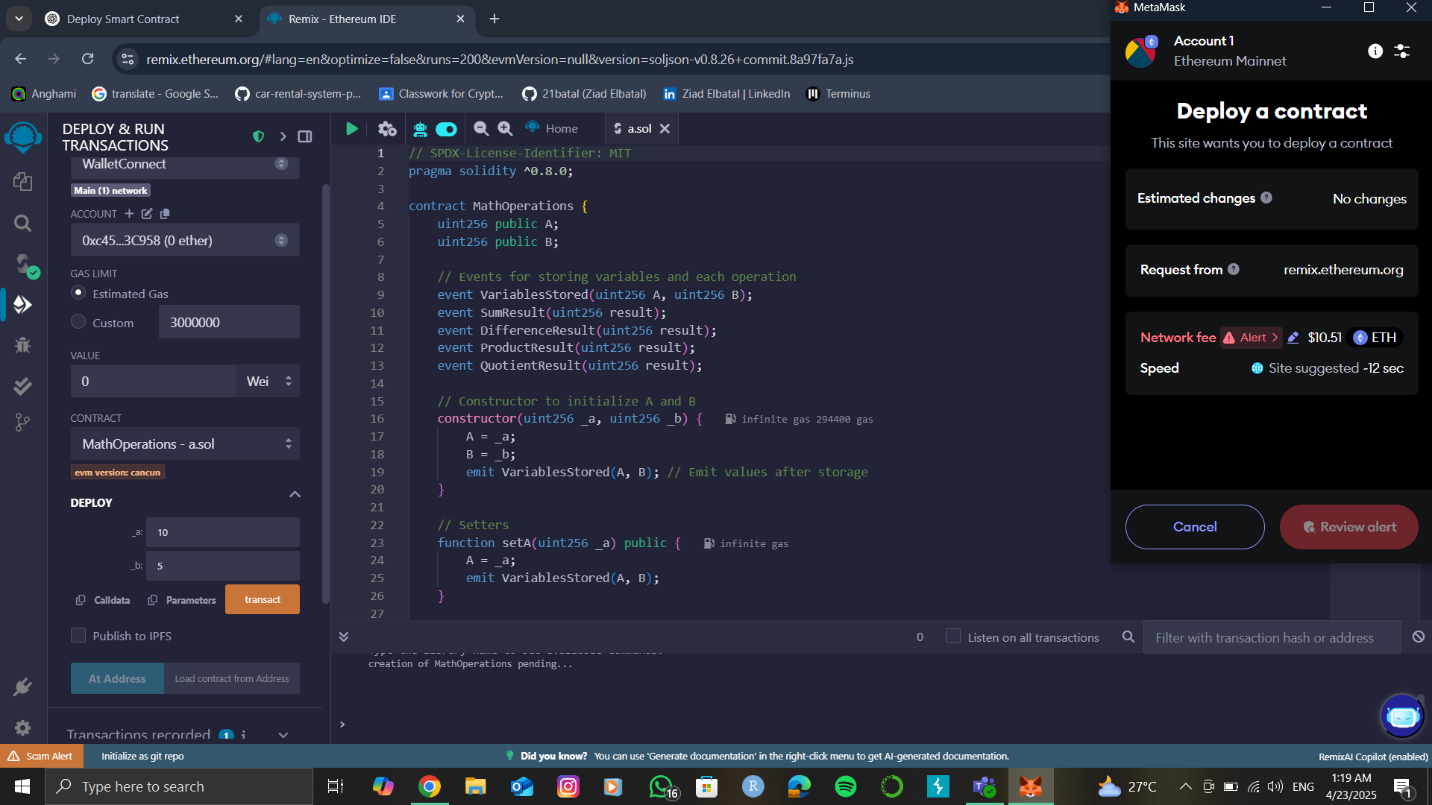
**3.Set deployment parameters**

* Gas limit: 3,000,000
* 0 ETH value
* Selected correct contract

**4.Confirmed in MetaMask**

****

1. **Successfully deployed**



Code: // SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract MathOperations {

uint256 public A;

uint256 public B;

event VariablesStored(uint256 a, uint256 b);

event SumResult(uint256 result);

event DifferenceResult(uint256 result);

event ProductResult(uint256 result);

event QuotientResult(uint256 result);

constructor(uint256 \_a, uint256 \_b) {

A = \_a;

B = \_b;

emit VariablesStored(A, B);

}

function getSum() public {

emit SumResult(A + B);

}

function getDifference() public {

emit DifferenceResult(A - B);

}

function getProduct() public {

emit ProductResult(A \* B);

}

function getQuotient() public {

require(B != 0, "Can't divide by zero");

emit QuotientResult(A / B);

}

}