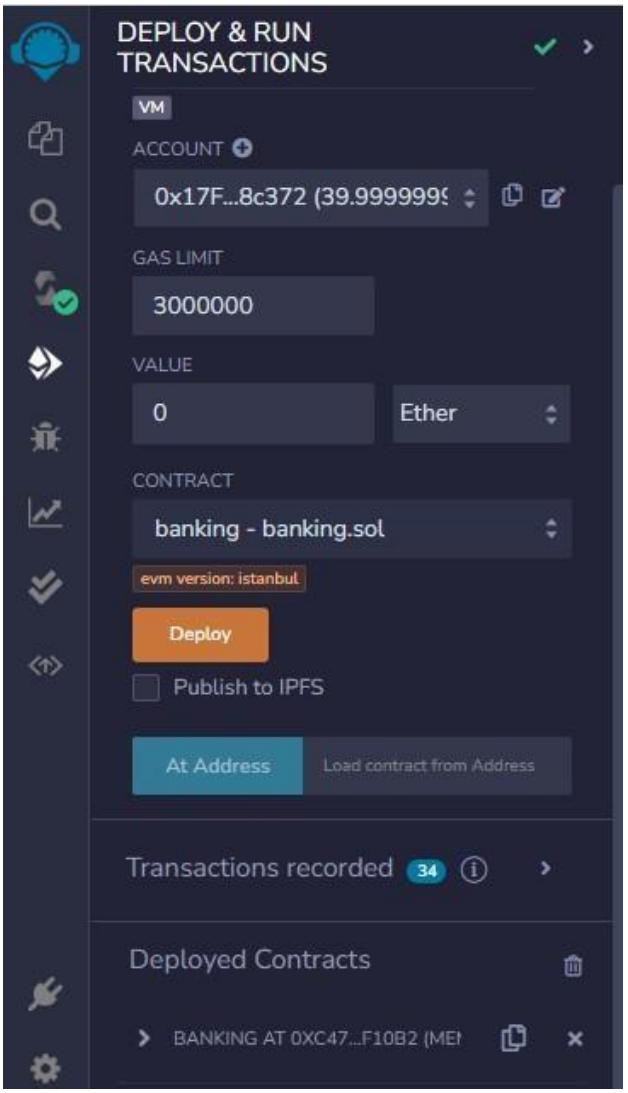
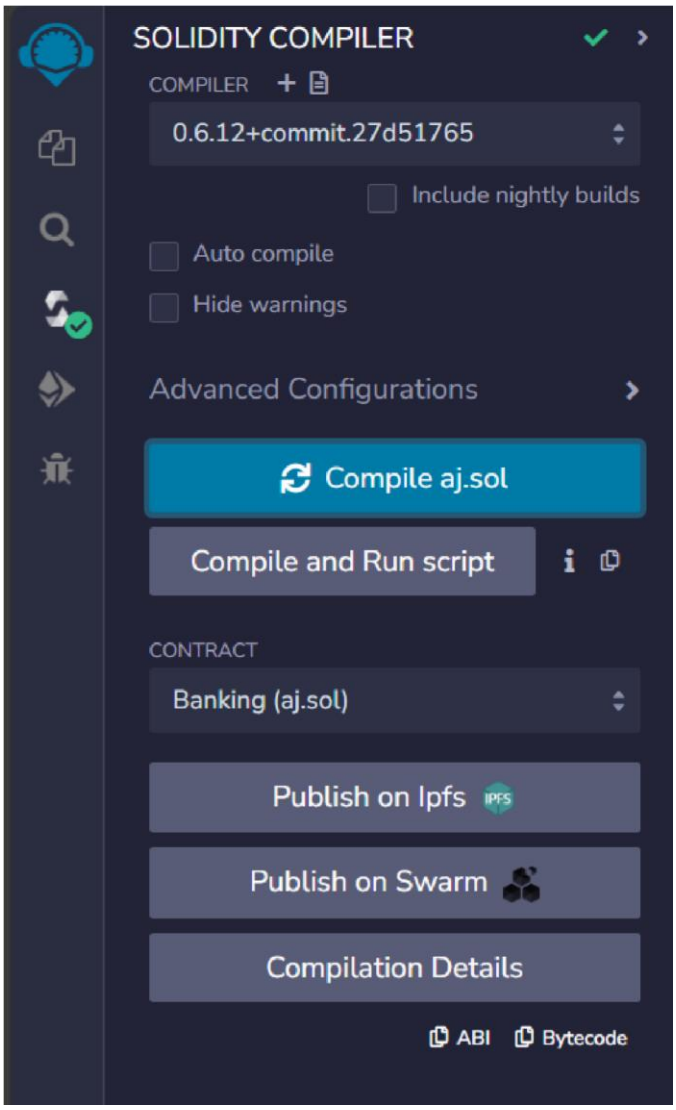
Name Vedant
Roll No. COBB033
Subject: BT

```
pragma solidity ^0.6;
contract banking
{
mapping(address=>uint) public user_account;
mapping(address=>bool) public user_exists;
function create_account() public payable returns(string memory)
{
require(user_exists[msg.sender]==false,'Account already created');
if(msg.value==0)
{
user_account[msg.sender]=0;
user_exists[msg.sender]=true;
return "Account created";
}
require(user_exists[msg.sender]==false,"Account already created");
user_account[msg.sender]=msg.value;
user_exists[msg.sender]=true; return
"Account created";
}
function deposit() public payable returns(string memory)
{
require(user_exists[msg.sender]==true,"Account not created");
require(msg.value>0,"Value for deposit is Zero");
user_account[msg.sender]=user_account[msg.sender]+msg.value;
return "Deposited Successfully";
}
function withdraw(uint amount) public payable returns(string memory)
{
   require(user_account[msg.sender]>amount,"Insufficient Balance");
require(user_exists[msg.sender]==true,"Account not created");
require(amount>0,"Amount should be more than zero");
user_account[msg.sender]=user_account[msg.sender]-amount;
msg.sender.transfer(amount);
return "Withdrawl Successful";
}
function transfer(address payable userAddress, uint amount) public returns(string
memory)
{
require(user_account[msg.sender]>amount,"Insufficient balance in Bank
account"); require(user_exists[msg.sender]==true,"Account is not created");
require(user_exists[userAddress]==true,"Transfer account does not exist");
require(amount>0,"Amount should be more than zero");
user_account[msg.sender]=user_account[msg.sender]-amount;
```

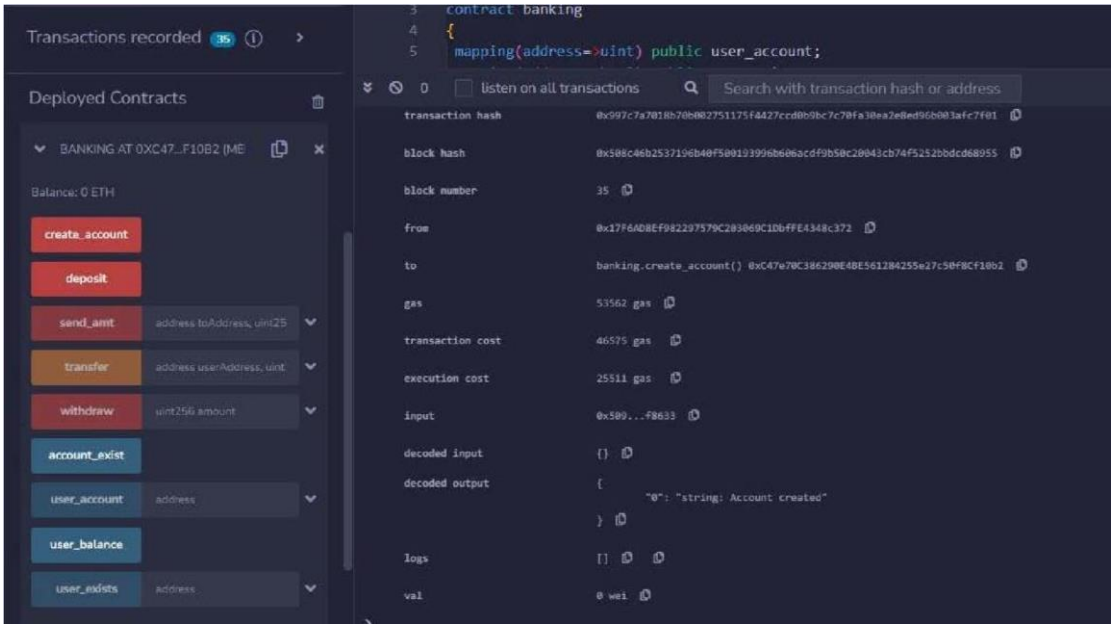
```
user_account[userAddress]=user_account[userAddress]+amount;
return "Transfer Successful";
}
function send_amt(address payable toAddress, uint256 amount) public payable returns(string memory)
{
require(user_account[msg.sender]>amount,"Insufficeint balance in Bank account"); require(user_exists[msg.sender]==true,"Account is not created"); require(amount>0,"Amount should be more than zero");
user_account[msg.sender]=user_account[msg.sender]-amount;
toAddress.transfer(amount);
return "Transfer Success";
}
function user_balance() public view returns(uint)
{
return user_account[msg.sender];
}
function account_exist() public view returns(bool)
{
return user_exists[msg.sender];
}
}
```

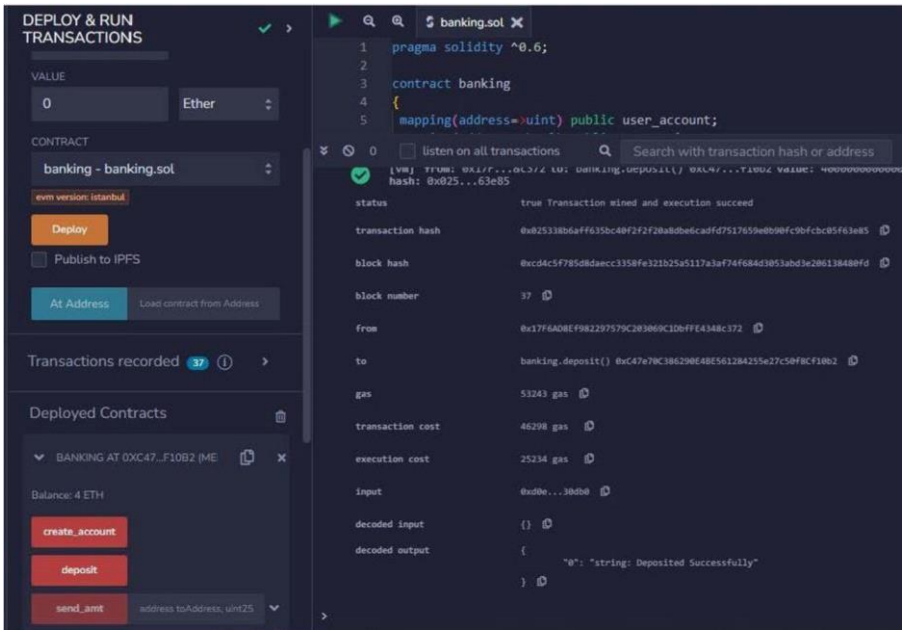
Compile SOL code Deploy contract



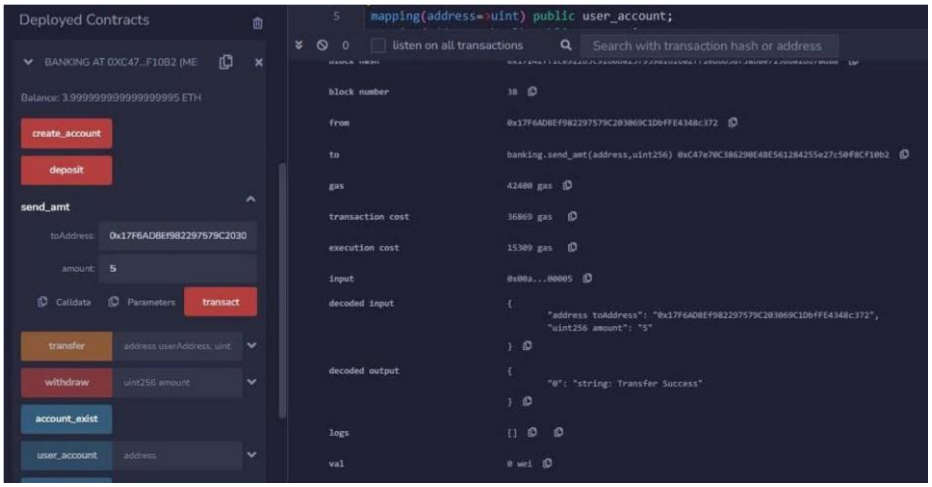

SOLIDITY COMPILER
COMPILER
0.6.12+commit.27d51765
Include nightly builds
Auto compile
Hide warnings
Advanced Configurations
Compile aj.sol
Compile and Run script
CONTRACT
Banking (aj.sol)
Publish on Ipfs
Publish on Swarm
Compilation Details
ABI
Bytecode

Create Account



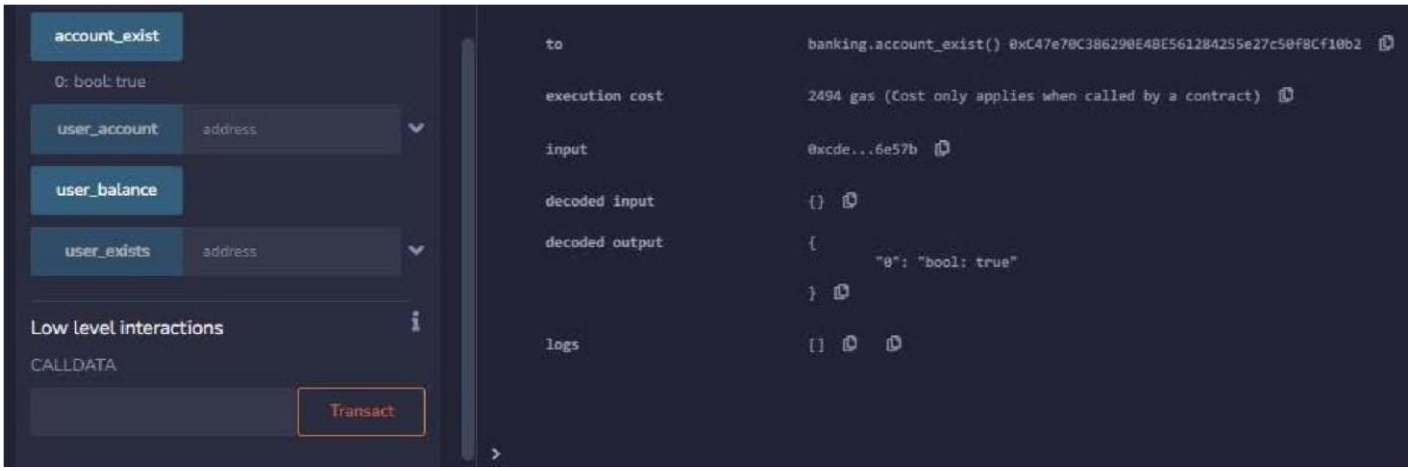


Deposit Amount

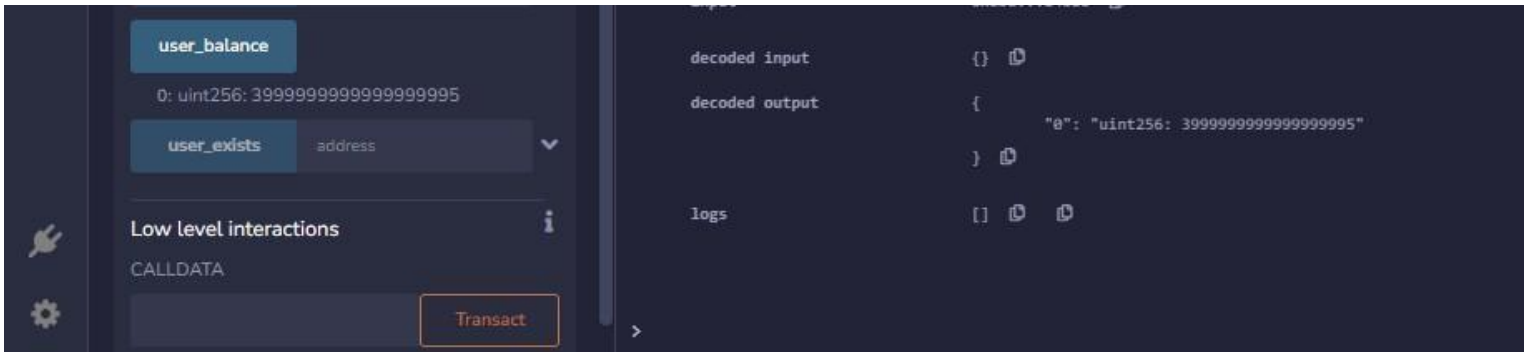


Send Amount

Check Account Exists



Check User balance



Withdraw Amount