

# IT465: Cryptocurrencies and Blockchain Technologies

## Lab Assignment 2

Name: Chinmayi C. Ramakrishna

Date: 18<sup>th</sup> Oct, 2021

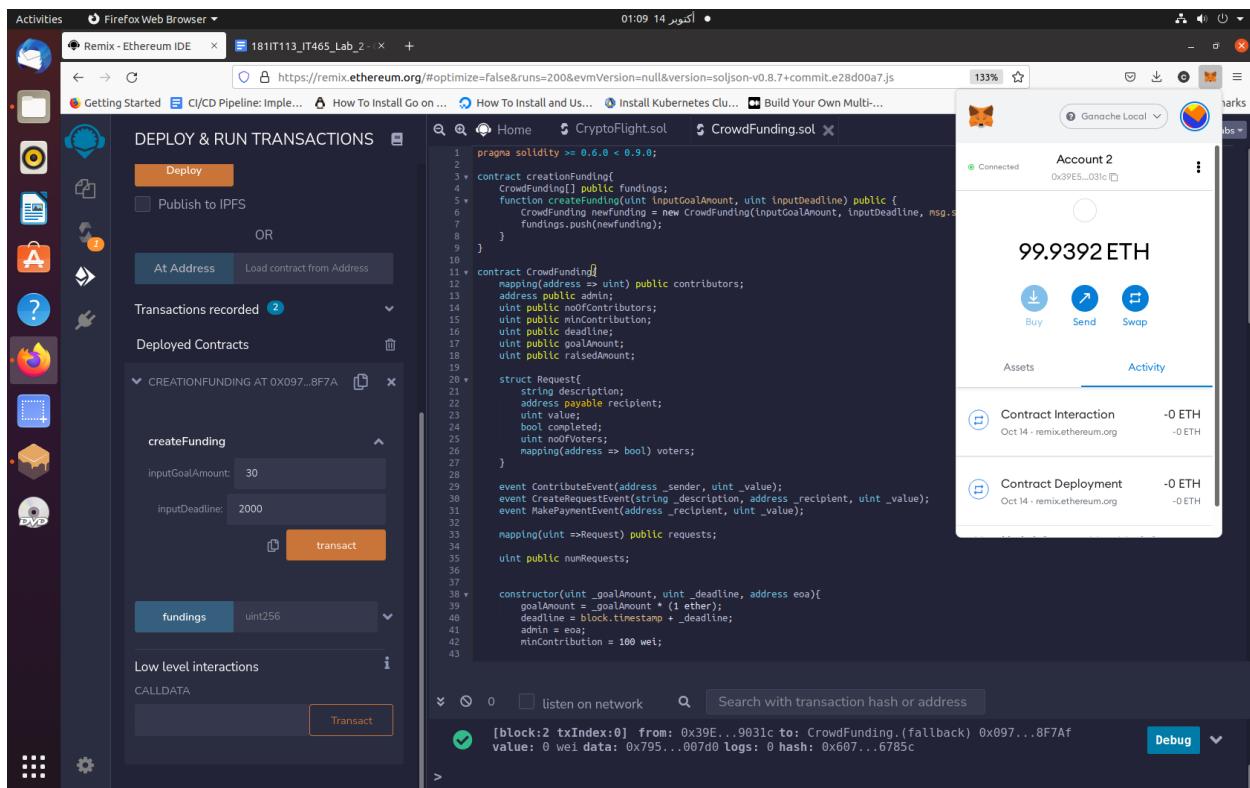
Roll No.: 181IT113

---

### Ganache and Metmask Implementation of the two applications:

#### 1) CrowdFunding

Deploying creationFunding contract with Account 2 as admin.



The screenshot shows the Remix Ethereum IDE interface. On the left, there's a sidebar with various development tools like terminal, file manager, and debugger. The main area has tabs for "Remix - Ethereum IDE" and "181IT113\_IT465\_Lab\_2". The current tab displays the Solidity code for "CrowdFunding.sol".

```
pragma solidity >= 0.6.0 < 0.9.0;
contract creationFunding{
    CrowdFunding[] public fundings;
    function createFunding(uint inputGoalAmount, uint inputDeadline) public {
        CrowdFunding newFunding = new CrowdFunding(inputGoalAmount, inputDeadline, msg.sender);
        fundings.push(newFunding);
    }
}
contract CrowdFunding{
    mapping(address => uint) public contributors;
    address public admin;
    uint public noOfContributors;
    uint public minContribution;
    uint public deadline;
    uint public goalAmount;
    uint public raisedAmount;
    struct Request{
        string description;
        address payable recipient;
        uint value;
        bool completed;
        uint noOfVoters;
        mapping(address => bool) voters;
    }
    event ContributeEvent(address _sender, uint _value);
    event CreateRequestEvent(string _description, address _recipient, uint _value);
    event MakePaymentEvent(address _recipient, uint _value);
    mapping(uint =>Request) public requests;
    uint public numRequests;
    constructor(uint _goalAmount, uint _deadline, address _eoA){
        goalAmount = _goalAmount * (1 ether);
        deadline = block.timestamp + _deadline;
        admin = _eoA;
        minContribution = 100;
    }
}
```

On the right, the results of a transaction are shown:

- deadline**: 0: uint256: 1634164702
- getBalance**: 0: uint256: 0
- goalAmount**: 0: uint256: 30000000000000000000000000
- minContribution**: 0: uint256: 100
- noOfContributors**: 0: uint256: 0
- numRequests**: 0: uint256: 0
- raisedAmount**: 0: uint256: 0

At the bottom, the transaction details are:

CALL [call] from: 0x30E5813fa939Dc6aB1F824653d64DcA62599031c to: CrowdFunding.raisedAmount()  
data: 0xc59...eeldc

Buttons at the bottom include "Debug" and a dropdown menu.

20 ethers contributed by Account 3.

The screenshot shows the Remix Ethereum IDE interface. On the left, there's a sidebar with various icons for different tools and features. The main area has tabs for "Home", "CryptoFlight.sol", and "CrowdFunding.sol". The "CrowdFunding.sol" tab is active, displaying the following Solidity code:

```
pragma solidity >=0.6.0 <0.8.0;

contract creationFunding{
    CrowdFunding[] public fundings;
    function createFunding(uint inputGoalAmount, uint inputDeadline) public {
        CrowdFunding newFundng = new CrowdFunding(inputGoalAmount, inputDeadline, msg.sender);
        fundings.push(newFundng);
    }
}

contract CrowdFunding{
    mapping(address payable) public contributors;
    address payable admin;
    uint public noOfContributors;
    uint public minContribution;
    uint public deadline;
    uint public goalAmount;
    uint public raisedAmount;
    struct Request{
        string description;
        address payable recipient;
        uint value;
        bool completed;
        uint nofVoters;
        mapping(address => bool) voters;
    }
    event ContributeEvent(address _sender, uint _value);
    event CreateRequestEvent(string _description, address _recipient, uint _value);
    event MakePaymentEvent(address _recipient, uint _value);
    mapping(uint =>Request) public requests;
    uint public numRequests;
    constructor(uint _goalAmount, uint _deadline, address eos){
        goalAmount = _goalAmount * (1 ether);
        deadline = block.timestamp + _deadline;
        admin = eos;
        minContribution = 100 wei;
    }
}
```

Below the code editor, there's a search bar with placeholder text "listen on network" and a "Search with transaction hash or address" field. A green checkmark icon indicates a successful transaction: "[block:3 txIndex:0] from: 0x526...A4703 to: CrowdFunding.contribute() 0x9B0...D24BA value: 26000000000000000000000000000000 wei data: 0xd7b...b99ba logs: 1 hash: 0x48a...f8313".

10 ethers contributed by Account 4.

The screenshot shows the Remix Ethereum IDE interface. On the left, the sidebar contains various development tools like Truffle, Hardhat, and Etherscan. The main workspace is titled "DEPLOY & RUN TRANSACTIONS". A sidebar on the left lists functions: contribute, createRequest, makePayment, refund, voteRequest, admin, contributors, deadline, and getBalance. Below these are their respective input fields and values (e.g., address 0x39E5813fa939Dc6aB1F824653d40cA62590931c for contributors). The central area displays the Solidity code for the CrowdFunding contract. The right side features the MetaMask wallet interface, showing 89.9988 ETH, three transaction buttons (Buy, Send, Swap), and a history section for a recent contribution to the contract. At the bottom, there's a search bar for transaction hashes and a "Debug" button.

```
pragma solidity >= 0.6.0 < 0.9.0;

contract CrowdFunding {
    mapping(address => uint) public contributions;
    address public admin;
    uint public numContributors;
    uint public minContribution;
    uint public deadline;
    uint public goalAmount;
    uint public raisedAmount;
}

struct Request {
    string description;
    address payable recipient;
    uint value;
    bool completed;
    uint numVoters;
    mapping(address => bool) voters;
}

event ContributeEvent(address _sender, uint _value);
event CreateRequestEvent(string _description, address _recipient, uint _value);
event MakePaymentEvent(address _recipient, uint _value);

mapping(uint => Request) public requests;
uint public numRequests;

constructor(uint _goalAmount, uint _deadline, address _eos) {
    goalAmount = _goalAmount * (1 ether);
    deadline = block.timestamp + _deadline;
    admin = _eos;
    minContribution = 100 wei;
}
```

The screenshot shows the Remix Ethereum IDE interface. On the left is a sidebar with various icons for different tools and features. The main area is titled "DEPLOY & RUN TRANSACTIONS". It displays several buttons with placeholder values: "deadline" (0: uint256: 1634164702), "getBalance" (0: uint256: 300000000000000000000000000000000), "goalAmount" (0: uint256: 300000000000000000000000000000000), "minContribution" (0: uint256: 100), "noOfContribut..." (0: uint256: 2), "numRequests" (0: uint256: 0), and "raisedAmount" (0: uint256: 300000000000000000000000000000000). To the right of these buttons is the Solidity code for the "CrowdFunding.sol" contract. The code defines a "creationFunding" contract with a "funding" array and a "createFunding" function. It also defines the "CrowdFunding" contract with variables for goalAmount, deadline, minContribution, and raisedAmount, along with Request, ContributeEvent, CreateRequestEvent, and MakePaymentEvent events. The constructor initializes these variables. At the bottom, there are tabs for "listen on network" and "Search with transaction hash or address", and a "Debug" button.

```
pragma solidity >= 0.6.0 < 0.9.0;

contract creationFunding{
    CrowdFunding[] public fundings;
    function createFunding(uint inputGoalAmount, uint inputDeadline) public {
        CrowdFunding newFunding = new CrowdFunding(inputGoalAmount, inputDeadline, msg.sender);
        fundings.push(newFunding);
    }
}

contract CrowdFunding{
    mapping(address admin);
    uint noOfContributors;
    uint minContribution;
    uint deadline;
    uint goalAmount;
    uint raisedAmount;
    struct Request{
        string description;
        address payable recipient;
        uint value;
        bool completed;
        uint nofVoters;
        mapping(address => bool) voters;
    }
    event ContributeEvent(address _sender, uint _value);
    event CreateRequestEvent(string _description, address _recipient, uint _value);
    event MakePaymentEvent(address _recipient, uint _value);
    mapping(uint =>Request) public requests;
    uint public numRequests;
    constructor(uint _goalAmount, uint _deadline, address eos){
        goalAmount = _goalAmount * (1 ether);
        deadline = block.timestamp + _deadline;
        admin = eos;
        minContribution = 100 wei;
    }
}
```

createRequest by admin(Account 2) for 20 wei with Account 3 as recipient.

The screenshot shows the Remix Ethereum IDE interface. On the left, there's a sidebar with various icons. In the center, the Ethereum IDE window is open with the title "CryptoFlight.sol". A sub-menu "CROWDFUNDING AT 0x9B0...D24BA" is expanded, showing a "DEPLOY & RUN TRANSACTIONS" section. Under this, a "createRequest" button is visible. Below it, there are fields for "\_description" ("Please help me raise money"), "\_recipient" ("d54dEBCaA616b6151A4703"), and "\_value" (20). To the right of these fields is a "transact" button. Further down, there are buttons for "makePayment" (with a dropdown "uint256 \_requestNo"), "refund", "voteRequest" (with a dropdown "uint256 \_requestNo"), and "admin". At the bottom of the transaction section, it says "0: address: 0x39E5813fa939Dc6aB1F824653 d64DcA62599031c". On the far right, a sidebar titled "Account 2" shows a balance of "99.937 ETH" with buttons for "Buy", "Send", and "Swap". Below this, sections for "Assets" and "Activity" show "Contract Interaction" logs. At the bottom of the interface, there's a search bar and a "Debug" button.

This screenshot is similar to the one above but shows the transaction details after it has been submitted. The "Contract Interaction" sidebar is now visible and displays the following information:

Details	
From:	0x39E5813fa939Dc6aB1F824653
To:	0x9B00ea4c3882237...
Nonce:	2
Amount:	-0 ETH
Gas Limit (Units):	113941
Gas Used (Units):	113941
Gas price:	20
Total:	0.002279 ETH

Below the details, the "Activity Log" section shows three entries:

- Transaction created with a value of 0 at 01:19 on 10/14/2021.
- Transaction submitted with estimated gas fee of 0.002 ETH at 01:19 on 10/14/2021.
- Transaction confirmed at 01:19 on 10/14/2021.

voteRequests by the two accounts so that makePayment can be approved.

The screenshot shows the Remix Ethereum IDE interface. On the left, there's a sidebar with various icons. The main area has tabs for "CryptoFlight.sol" and "CrowdFunding.sol". The "CrowdFunding.sol" tab is active, displaying the Solidity code for the contract. In the center, there's a "DEPLOY & RUN TRANSACTIONS" section with several buttons: "createRequest", "makePayment", "refund", and "voteRequest". The "voteRequest" button is highlighted. Below it, there's a dropdown menu set to "contributors" and a field where "\_requestNo:" is set to "0". An orange "transact" button is next to it. To the right of this, there's a "Ganache Local" interface showing "Account 3" with the address 0x5264...4703 and 79.9969 ETH. It has "Buy", "Send", and "Swap" buttons. Below that is an "Assets" section and an "Activity" section which lists a "Vote Request" and a "Contribute" transaction. At the bottom, there's a "Debug" button and a log window showing a transaction being sent to the contract.

```
76 contributors[msg.sender] = 0;
77
78 //admin is the one who deployed the fundingCreator contract
79 modifier onlyAdmin() {
80     require(msg.sender == admin, "Only Admin can call this function");
81 }
82
83
84
85
86 function createRequest(string memory _description, address payable _recipient, uint
87 Request storage newRequest = requests[numRequests];
88 numRequests++;
89
90 newRequest.description = _description;
91 newRequest.recipient = _recipient;
92 newRequest.value = _value;
93 newRequest.completed = false;
94 newRequest.noOfVoters = 0;
95
96 emit CreateRequestEvent(_description, _recipient, _value);
97
98
99 function voteRequest(uint _requestNo) public{
100     require(contributors[msg.sender] > 0, "You must be a contributor to vote!");
101     Request storage thisRequest = requests[_requestNo];
102
103     require(thisRequest.voters[msg.sender] == false, "You have already voted!");
104     thisRequest.voters[msg.sender] = true;
105     thisRequest.noOfVoters++;
106
107     function makePayment(uint _requestNo) public onlyAdmin{
108         require(raisedAmount >= goalAmount);
109         Request storage thisRequest = requests[_requestNo];
110         require(thisRequest.completed == false, "This request has been completed");
111         // majority of voters voted 'yes' (more than 50%)
112         require(thisRequest.noOfVoters > noOfContributors / 2);
113
114         thisRequest.recipient.transfer(thisRequest.value);
115         thisRequest.completed = true;
116         emit MakePaymentEvent(thisRequest.recipient, thisRequest.value);
117     }
118 }
```

This screenshot is similar to the previous one but shows "Account 4" with the address 0x4E6C...2Ce2 and 89.9978 ETH. The "Activity" section shows a "Vote Request" and a "Contribute" transaction. The "Debug" log window at the bottom shows a transaction being sent to the contract.

```
76 contributors[msg.sender] = 0;
77
78 //admin is the one who deployed the fundingCreator contract
79 modifier onlyAdmin() {
80     require(msg.sender == admin, "Only Admin can call this function");
81 }
82
83
84
85
86 function createRequest(string memory _description, address payable _recipient, uint
87 Request storage newRequest = requests[numRequests];
88 numRequests++;
89
90 newRequest.description = _description;
91 newRequest.recipient = _recipient;
92 newRequest.value = _value;
93 newRequest.completed = false;
94 newRequest.noOfVoters = 0;
95
96 emit CreateRequestEvent(_description, _recipient, _value);
97
98
99 function voteRequest(uint _requestNo) public{
100     require(contributors[msg.sender] > 0, "You must be a contributor to vote!");
101     Request storage thisRequest = requests[_requestNo];
102
103     require(thisRequest.voters[msg.sender] == false, "You have already voted!");
104     thisRequest.voters[msg.sender] = true;
105     thisRequest.noOfVoters++;
106
107     function makePayment(uint _requestNo) public onlyAdmin{
108         require(raisedAmount >= goalAmount);
109         Request storage thisRequest = requests[_requestNo];
110         require(thisRequest.completed == false, "This request has been completed");
111         // majority of voters voted 'yes' (more than 50%)
112         require(thisRequest.noOfVoters > noOfContributors / 2);
113
114         thisRequest.recipient.transfer(thisRequest.value);
115         thisRequest.completed = true;
116         emit MakePaymentEvent(thisRequest.recipient, thisRequest.value);
117     }
118 }
```

## Details after creating request by admin.

The screenshot shows the Remix Ethereum IDE interface. On the left, there's a sidebar with various icons. The main area has tabs for "CryptoFlight.sol" and "CrowdFunding.sol". The "CrowdFunding.sol" tab is active, displaying the Solidity code for the contract. In the center, there's a "DEPLOY & RUN TRANSACTIONS" section. Under "variables", it shows:

- noOfContribut...: 0
- numRequests: 0
- raisedAmount: 0 uint256: 30000000000000000000
- requests: 0
  - 0: string: description Please help me raise money
  - 1: address: recipient 0x5264A661233E3e9A31d54dEBCa616b6151A4703
  - 2: uint256: value 20
  - 3: bool: completed false
  - 4: uint256: noOfVoters 2

Under "Low level interactions", there's a "CALLDATA" section with a "Transact" button. The "call" log shows:

```
[call] from: 0x4E6cC791AaCE5eE0Bb2C9674aC73dcA2E8872Ce2 to: CrowdFunding.requests(uint256) data: 0x81d...00000
```

Contributors take the address and return the amount they contributed.

The screenshot shows the Remix Ethereum IDE interface, similar to the previous one but with updated variable values. The "variables" section now shows:

- admin: 0: address: 0x39E5813fa939Dc6aB1F824653d64Dca62599031c
- contributors: 4dEBCa616b6151A4703
  - 0: uint256: 20000000000000000000
- deadline: 0: uint256: 1634164702
- getBalance: 0: uint256: 30000000000000000000
- goalAmount: 0: uint256: 30000000000000000000
- minContribution: 0: uint256: 100
- noOfContribut...: 0: uint256: 2

The "call" log at the bottom remains the same as in the previous screenshot.

DEPLOY & RUN TRANSACTIONS

CROWDFUNDING AT 0x9B0...D24BA

contributors: 0x39E5813fe939Dc6aB1F824653d64Dca62599031c

admin: 0x39E5813fe939Dc6aB1F824653d64Dca62599031c

contributors: 4dEBCaA616b6151A4703

deadline: 0:uint256: 20000000000000000000

getBalance: 0:uint256: 20000000000000000000

goalAmount: 0:uint256: 30000000000000000000

minContribution: 0:uint256: 0

makePayment: 0

refund: 0

voteRequest: \_requestNo: "0"

createRequest: "Please help me raise mon..."

contribute: 0

transact

```

76     contributors[msg.sender] = 0;
77 }
78
79 //admin is the one who deployed the fundingCreator contract
80 modifier onlyAdmin() {
81     require(msg.sender == admin, "Only Admin can call this function");
82 }
83
84
85
86 function createRequest(string memory _description, address payable _recipient, uint _value) public {
87     Request storage newRequest = requests[numRequests];
88     numRequests++;
89
90     newRequest.description = _description;
91     newRequest.recipient = _recipient;
92     newRequest.value = _value;
93     newRequest.completed = false;
94     newRequest.noOfVoters = 0;
95
96     emit CreateRequestEvent(_description, _recipient, _value);
97 }
98
99 function voteRequest(uint _requestNo) public {
100    require(contributors[msg.sender] > 0, "You must be a contributor to vote!");
101    Request storage thisRequest = requests[_requestNo];
102
103    require(thisRequest.voters[msg.sender] == false, "You have already voted!");
104    thisRequest.voters[msg.sender] = true;
105    thisRequest.noOfVoters++;
106
107    function makePayment(uint _requestNo) public onlyAdmin{
108        require(raisedAmount >= goalAmount);
109        Request storage thisRequest = requests[_requestNo];
110        require(thisRequest.completed == false, "This request has been completed");
111        // majority of voters voted 'yes' (more than 50%)
112        require(thisRequest.noOfVoters > noOfContributors / 2);
113
114        thisRequest.recipient.transfer(thisRequest.value);
115        thisRequest.completed = true;
116        emit MakePaymentEvent(thisRequest.recipient, thisRequest.value);
117    }
118 }
```

Activity tab: Make Payment - Oct 14 - remix.ethereum.org -0 ETH -0 ETH

Activity tab: Contract Interaction - Oct 14 - remix.ethereum.org -0 ETH -0 ETH

Balance: 99.9358 ETH

Balance of Account 3 after making payment with 20 wei.

DEPLOY & RUN TRANSACTIONS

refund: 0

voteRequest: \_requestNo: "0"

transact

admin: 0x39E5813fe939Dc6aB1F824653d64Dca62599031c

contributors: 4dEBCaA616b6151A4703

deadline: 0:uint256: 1634164702

getBalance: 0:uint256: 299999999999999980

goalAmount: 0:uint256: 30000000000000000000

minContribution: 0:uint256: 0

```

76     contributors[msg.sender] = 0;
77 }
78
79 //admin is the one who deployed the fundingCreator contract
80 modifier onlyAdmin() {
81     require(msg.sender == admin, "Only Admin can call this function");
82 }
83
84
85
86 function createRequest(string memory _description, address payable _recipient, uint _value) public {
87     Request storage newRequest = requests[numRequests];
88     numRequests++;
89
90     newRequest.description = _description;
91     newRequest.recipient = _recipient;
92     newRequest.value = _value;
93     newRequest.completed = false;
94     newRequest.noOfVoters = 0;
95
96     emit CreateRequestEvent(_description, _recipient, _value);
97 }
98
99 function voteRequest(uint _requestNo) public {
100    require(contributors[msg.sender] > 0, "You must be a contributor to vote!");
101    Request storage thisRequest = requests[_requestNo];
102
103    require(thisRequest.voters[msg.sender] == false, "You have already voted!");
104    thisRequest.voters[msg.sender] = true;
105    thisRequest.noOfVoters++;
106
107    function makePayment(uint _requestNo) public onlyAdmin{
108        require(raisedAmount >= goalAmount);
109        Request storage thisRequest = requests[_requestNo];
110        require(thisRequest.completed == false, "This request has been completed");
111        // majority of voters voted 'yes' (more than 50%)
112        require(thisRequest.noOfVoters > noOfContributors / 2);
113
114        thisRequest.recipient.transfer(thisRequest.value);
115        thisRequest.completed = true;
116        emit MakePaymentEvent(thisRequest.recipient, thisRequest.value);
117    }
118 }
```

Activity tab: Vote Request - Oct 14 - remix.ethereum.org -0 ETH -0 ETH

Activity tab: Contribute - Oct 14 - remix.ethereum.org -20 ETH -20 ETH

Balance: 79.9969 ETH

Request 1 makes a payment of 20 ethers. The completed value has been set to true in requests. The number of requests is 2.

Activities Firefox Web Browser

Remix - Ethereum IDE x 181IT113\_IT465\_Lab\_2 x Epoch Converter - Unix Ti x 01:36 14 اکتوبر • 133% ☆

Getting Started CI/CD Pipeline: Implement How To Install Go on ... How To Install and Us... Install Kubernetes Clu... Build Your Own Multi...

Home CryptoFlight.sol CrowdFunding.sol

DEPLOY & RUN TRANSACTIONS

minContribution  
0: uint256: 100

noOfContribut...  
0: uint256: 2

numRequests  
0: uint256: 2

raisedAmount  
0: uint256: 30000000000000000000000000000000

requests  
1

0: string: description Please help me raise mon  
ey

1: address: recipient 0x5264A66123BE3e9A31  
d54eBCaA616b6151A4703

2: uint256: value 20000000000000000000

3: bool: completed true

4: uint256: noOfVoters 2

contributors[msg.sender] = 0;

modifier onlyAdmin() {  
 require(msg.sender == admin, "Only Admin can call this function");  
}

function createRequest(string memory \_description, address payable \_recipient, uint \_value) public {  
 Request storage newRequest = requests[numRequests];  
 numRequests++;  
  
 newRequest.description = \_description;  
 newRequest.recipient = \_recipient;  
 newRequest.value = \_value;  
 newRequest.completed = false;  
 newRequest.noOfVoters = 0;  
  
 emit CreateRequestEvent(\_description, \_recipient, \_value);  
}

function voteRequest(uint \_requestNo) public {  
 require(contributors[msg.sender] > 0, "You must be a contributor to vote!");  
 Request storage thisRequest = requests[\_requestNo];  
  
 require(thisRequest.voters[msg.sender] == false, "You have already voted!");  
 thisRequest.voters[msg.sender] = true;  
 thisRequest.noOfVoters++;  
}

function makePayment(uint \_requestNo) public onlyAdmin{  
 require(raisedAmount >= goalAmount);  
 Request storage thisRequest = requests[\_requestNo];  
 require(thisRequest.completed == false, "This request has been completed");  
 // majority of voters voted 'yes' (more than 50%)  
 require(thisRequest.noOfVoters > noOfContributors / 2);  
  
 thisRequest.recipient.transfer(thisRequest.value);  
 thisRequest.completed = true;  
 emit MakePaymentEvent(thisRequest.recipient, thisRequest.value);  
}

CALL [call] from: 0x39E5813fa9390c6aB1F824653d64DcA62599031c to: CrowdFunding.numRequests()  
data: 0x9bf...42bf

Low level interactions

CALL DATA

Ganache Local Account 2 0x39E5...03ic Connected 99.9326 ETH Assets Activity

Buy Send Swap

Make Payment -0 ETH Oct 14 - remix.ethereum.org

Contract Interaction -0 ETH Oct 14 - remix.ethereum.org

Debug

Activities Firefox Web Browser 01:36 14 أكتوبر ●

Remix - Ethereum IDE 181IT113\_IT465\_Lab\_2 Epoch Converter - Unix Ti +

Getting Started CI/CD Pipeline: Implement How To Install Go on ... How To Install and Use... Install Kubernetes Clu... Build Your Own Multi...

Home CryptoFlight.sol CrowdFunding.sol

DEPLOY & RUN TRANSACTIONS

minContribution 0: uint256: 100

noOfContribut... 0: uint256: 2

numRequests 0: uint256: 2

raisedAmount 0: uint256: 30000000000000000000000000000000

requests 1

0: string: description Please help me raise mon...ey

1: address: recipient 0x5264A661233E3e9A31d54dEBCaA616b6151A4703

2: uint256: value 20000000000000000000000000000000

3: bool: completed true

4: uint256: noOfVoters 2

Low level interactions

contributors[msg.sender] = 0;

//admin is the one who deployed the fundingCreator contract

modifier onlyAdmin() { require(msg.sender == admin, "Only Admin can call this function"); };

function createRequest(string memory \_description, address payable \_recipient, uint numRequests) public { Request storage newRequest = requests[numRequests]; numRequests++;

newRequest.description = \_description;

newRequest.recipient = \_recipient;

newRequest.value = \_value;

newRequest.completed = false;

newRequest.noOfVoters = 0;

emit CreateRequestEvent(\_description, \_recipient, \_value); }

function voteRequest(uint \_requestNo) public { require(contributors[msg.sender] > 0, "You must be a contributor to vote!"); Request storage thisRequest = requests[\_requestNo];

require(thisRequest.voters[msg.sender] == false, "You have already voted!");

thisRequest.voters[msg.sender] = true;

thisRequest.noOfVoters++;

}

function makePayment(uint \_requestNo) public onlyAdmin { require(raisedAmount > goalAmount); Request storage thisRequest = requests[\_requestNo];

require(thisRequest.completed == false, "This request has been completed");

// majority of voters voted 'yes' (more than 50%)

require(thisRequest.noOfVoters > noOfContributors / 2);

thisRequest.recipient.transfer(thisRequest.value);

thisRequest.completed = true;

emit MakePaymentEvent(thisRequest.recipient, thisRequest.value); }

call [call] from: 0x39E5813fa939Dc6aB1F824653d64DcA62599031c to: CrowdFunding.numRequests() data: 0x9fb...42b1f

Debug

Account 2 Connected 0x39E5813fa939Dc6aB1F824653d64DcA62599031c

99.9326 ETH

Buy Send Swap

Assets Activity

Make Payment Oct 14 - remix.ethereum.org -0 ETH -0 ETH

Contract Interaction Oct 14 - remix.ethereum.org -0 ETH -0 ETH

Activities Firefox Web Browser 01:37 14 أكتوبر ●

Remix - Ethereum IDE 181IT113\_IT465\_Lab\_2 Epoch Converter - Unix Ti +

Getting Started CI/CD Pipeline: Implement How To Install Go on ... How To Install and Use... Install Kubernetes Clu... Build Your Own Multi...

Home CryptoFlight.sol CrowdFunding.sol

DEPLOY & RUN TRANSACTIONS

\_requestNo: 1

transact

admin

0: address: recipient 0x39E5813fa939Dc6aB1F824653d64DcA62599031c

contributors 4dEBCaA616b6151A4703

0: uint256: 20000000000000000000000000000000

deadline

0: uint256: 1634164702

getBalance

0: uint256: 99999999999999999999999999999999

goalAmount

0: uint256: 30000000000000000000000000000000

minContribution

0: uint256: 100

contributors[msg.sender] = 0;

//admin is the one who deployed the fundingCreator contract

modifier onlyAdmin() { require(msg.sender == admin, "Only Admin can call this function"); };

function createRequest(string memory \_description, address payable \_recipient, uint numRequests) public { Request storage newRequest = requests[numRequests]; numRequests++;

newRequest.description = \_description;

newRequest.recipient = \_recipient;

newRequest.value = \_value;

newRequest.completed = false;

newRequest.noOfVoters = 0;

emit CreateRequestEvent(\_description, \_recipient, \_value); }

function voteRequest(uint \_requestNo) public { require(contributors[msg.sender] > 0, "You must be a contributor to vote!"); Request storage thisRequest = requests[\_requestNo];

require(thisRequest.voters[msg.sender] == false, "You have already voted!");

thisRequest.voters[msg.sender] = true;

thisRequest.noOfVoters++;

}

function makePayment(uint \_requestNo) public onlyAdmin { require(raisedAmount > goalAmount); Request storage thisRequest = requests[\_requestNo];

require(thisRequest.completed == false, "This request has been completed");

// majority of voters voted 'yes' (more than 50%)

require(thisRequest.noOfVoters > noOfContributors / 2);

thisRequest.recipient.transfer(thisRequest.value);

thisRequest.completed = true;

emit MakePaymentEvent(thisRequest.recipient, thisRequest.value); }

call [call] from: 0x39E5813fa939Dc6aB1F824653d64DcA62599031c to: CrowdFunding.numRequests() data: 0x9fb...42b1f

Debug

Account 3 Connected 0x39E5813fa939Dc6aB1F824653d64DcA62599031c

99.9956 ETH

Buy Send Swap

Assets Activity

Vote Request Oct 14 - remix.ethereum.org -0 ETH -0 ETH

Vote Request Oct 14 - remix.ethereum.org -0 ETH -0 ETH

## New deployment to check the functionality of refund.

The screenshot shows the Remix Ethereum IDE interface. On the left, there's a sidebar with various icons. The main area has tabs for "DEPLOY & RUN TRANSACTIONS" and "CrowdFunding.sol". The Solidity code for CrowdFunding.sol is displayed:

```
pragma solidity >= 0.6.0 < 0.9.0;
contract creationFunding{
    CrowdFunding[] public fundings;
    function createFunding(uint inputGoalAmount, uint inputDeadline) public {
        CrowdFunding newFunding = new CrowdFunding(inputGoalAmount, inputDeadline);
        fundings.push(newFunding);
    }
}
contract CrowdFunding{
    mapping(address => uint) public contributors;
    address public admin;
    uint public noOfContributors;
    uint public minContribution;
    uint public deadline;
    uint public goalAmount;
    uint public raisedAmount;
    struct Request{
        string description;
        address payable recipient;
        uint value;
        bool completed;
        uint noOfVoters;
        mapping(address => bool) voters;
    }
    event ContributeEvent(address _sender, uint _value);
    event CreateRequestEvent(string _description, address _recipient, uint _value);
    event MakePaymentEvent(address _recipient, uint _value);
    mapping(uint =>Request) public requests;
    uint public numRequests;
    constructor(uint _goalAmount, uint _deadline, address _eoA){
        goalAmount = _goalAmount * (1 ether);
        deadline = block.timestamp + _deadline;
        admin = _eoA;
    }
}
```

The "DEPLOY & RUN TRANSACTIONS" panel shows a "Deploy" button and a "Transactions recorded" section. The "Transactions recorded" section lists "CREATIONFUNDING AT 0XBEA...3C45" with a "createFunding" entry. The "createFunding" entry shows inputs: "inputGoalAmount: 20" and "inputDeadline: 600". Below it is a "funding" dropdown set to "uint256". There's also a "Low level interactions" section with a "CALLDATA" tab and a "Transact" button.

The right side of the interface shows a Ganache Local account with 99.9392 ETH. It includes sections for "Assets" (Buy, Send, Swap) and "Activity" (Contract Interaction and Contract Deployment logs). Transaction details for a recent interaction are shown:

[block:2 txIndex:0] from: 0x0CD6...dF21 to: CrowdFunding.(fallback) 0x8Ea...3C453  
value: 0 wei data: 0x795...00258 logs: 0 hash: 0xac...ad117

A "Debug" button is at the bottom right.

This screenshot shows a second deployment of the CrowdFunding contract. The interface is similar to the first one, with the Solidity code for CrowdFunding.sol:

```
pragma solidity >= 0.6.0 < 0.9.0;
contract creationFunding{
    CrowdFunding[] public fundings;
    function createFunding(uint inputGoalAmount, uint inputDeadline) public {
        CrowdFunding newFunding = new CrowdFunding(inputGoalAmount, inputDeadline);
        fundings.push(newFunding);
    }
}
contract CrowdFunding{
    mapping(address => uint) public contributors;
    address public admin;
    uint public noOfContributors;
    uint public minContribution;
    uint public deadline;
    uint public goalAmount;
    uint public raisedAmount;
    struct Request{
        string description;
        address payable recipient;
        uint value;
        bool completed;
        uint noOfVoters;
        mapping(address => bool) voters;
    }
    event ContributeEvent(address _sender, uint _value);
    event CreateRequestEvent(string _description, address _recipient, uint _value);
    event MakePaymentEvent(address _recipient, uint _value);
    mapping(uint =>Request) public requests;
    uint public numRequests;
    constructor(uint _goalAmount, uint _deadline, address _eoA){
        goalAmount = _goalAmount * (1 ether);
        deadline = block.timestamp + _deadline;
        admin = _eoA;
    }
}
```

The "DEPLOY & RUN TRANSACTIONS" panel shows a "getBalance" entry with value 0, and several other variables like "goalAmount", "minContribution", "noOfContribut...", "numRequests", "raisedAmount", and "requests" with their respective values. The "Low level interactions" section is present.

The right side shows a Ganache Local account with 94.9982 ETH. The "Activity" section shows a "Contribute" entry:

[call] from: 0x353932e6950B70A966bbd4d65Be89f77d79F6AB0 to: CrowdFunding.raisedAmount()  
data: 0xc59...ee1dc

A "Need help? Contact MetaMask Support" message is visible in the bottom right.

Activities Firefox Web Browser • 02:11 14 آکتوبر ٢٠٢٣

Remix - Ethereum IDE x 181IT113\_IT465\_Lab\_2 - x + https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js 133% ⚡

Getting Started CI/CD Pipeline: Implementing a Crowdfunding Contract How To Install Go on... How To Install and Use... Install Kubernetes Clu... Build Your Own Multi...

DEPLOY & RUN TRANSACTIONS

Home CrowdFunding.sol

```
pragma solidity >= 0.6.0 < 0.9.0;

contract creationfunding{
    CrowdFunding[] public fundings;
    function createFunding(uint inputGoalAmount, uint inputDeadline) public {
        CrowdFunding newFunding = new CrowdFunding(inputGoalAmount, inputDeadline);
        fundings.push(newFunding);
    }
}

contract CrowdFunding{
    mapping(address => uint) public contributors;
    address public admin;
    uint public noOfContributors;
    uint public minContribution;
    uint public deadline;
    uint public goalAmount;
    uint public raisedAmount;

    struct Request{
        string description;
        address payable recipient;
        uint value;
        bool completed;
        uint noOfVoters;
        mapping(address => bool) voters;
    }

    event ContributeEvent(address _sender, uint _value);
    event CreateRequestEvent(string _description, address _recipient, uint _value);
    event MakePaymentEvent(address _recipient, uint _value);

    mapping(uint => Request) public requests;
    uint public numRequests;

    constructor(uint _goalAmount, uint _deadline, address _eoA){
        goalAmount = _goalAmount * (1 ether);
        deadline = block.timestamp + _deadline;
        admin = _eoA;
    }
}
```

getBalance 0: uint256: 0

goalAmount 0: uint256: 20000000000000000000000000000000

minContribution 0: uint256: 100

noOfContributors 0: uint256: 2

numRequests 0: uint256: 0

raisedAmount 0: uint256: 12000000000000000000000000000000

requests uint256

Low level interactions

CALL [call] from: 0x852F0C979ecD7B2cbF7e518B60420cF7559f11B to: CrowdFunding.raisedAmount() data: 0xC59...ee1dc

Ganache Local Connected Account 4 0x852F0C979ecD7B2cbF7e518B60420cF7559f11B

92.9988 ETH

Buy Send Swap

Assets Activity

Contribute Oct 14 · remix.ethereum.org -7 ETH -7 ETH

Need help? Contact MetaMask Support

Raised Amount is: 12ethers (goal Amount is 20 ethers)

```
pragma solidity >= 0.6.0 < 0.9.0;

contract creationFunding{
    CrowdFunding[] public fundings;
    function createFunding(uint inputGoalAmount, uint inputDeadline) public {
        CrowdFunding newFunding = new CrowdFunding(inputGoalAmount, inputDeadline, msg.sender);
        fundings.push(newFunding);
    }
}

contract CrowdFunding{
    mapping(address => uint) public contributors;
    address public admin;
    uint public noOfContributors;
    uint public minContribution;
    uint public deadline;
    uint public goalAmount;
    uint public raisedAmount;
    struct Request{
        string description;
        address payable recipient;
        uint value;
        bool completed;
        uint noOfVoters;
        mapping(address => bool) voters;
    }
    event ContributeEvent(address _sender, uint _value);
    event CreateRequestEvent(string _description, address _recipient, uint _value);
    event MakePaymentEvent(address _recipient, uint _value);

    mapping(uint =>Request) public requests;
    uint public numRequests;
    constructor(uint _goalAmount, uint _deadline, address _eoA){
        goalAmount = _goalAmount * (1 ether);
        deadline = block.timestamp + _deadline;
        admin = _eoA;
    }
}
```

Low level interactions

CALLDATA

Transact

0 listen on network Search with transaction hash or address

[block:5 txIndex:0] from: 0x353...F6AB0 to: CrowdFunding.refund() 0x86c...dFc32 value: 0 wei data: 0x590...e1a3 logs: 0 hash: 0x646...2596e

Debug

## Refund requested by Account 3 after deadline.

The screenshot shows the Remix Ethereum IDE interface. On the left, there's a sidebar with various icons. In the center, the code editor displays the `CrowdFunding.sol` contract. The transaction history panel on the right shows a refund transaction from account 3 to account 4. The refund amount is 0 wei, and the log message indicates a successful refund.

```
pragma solidity >= 0.6.0 < 0.9.0;

contract creationFunding{
    CrowdFunding[] public fundings;
    function createFunding(uint inputGoalAmount, uint inputDeadline) public {
        CrowdFunding newFunding = new CrowdFunding(inputGoalAmount, inputDeadline, msg.sender);
        fundings.push(newFunding);
    }
}

contract CrowdFunding{
    mapping(address => uint) public contributors;
    address public admin;
    uint public noOfContributors;
    uint public minContribution;
    uint public deadline;
    uint public goalAmount;
    uint public raisedAmount;
    struct Request{
        string description;
        address payable recipient;
        uint value;
        bool completed;
        uint noOfVoters;
        mapping(address => bool) voters;
    }
    event ContributeEvent(address _sender, uint _value);
    event CreateRequestEvent(string _description, address _recipient, uint _value);
    event MakePaymentEvent(address _recipient, uint _value);
    mapping(uint =>Request) public requests;
    uint public numRequests;
    constructor(uint _goalAmount, uint _deadline, address _ea){
        goalAmount = _goalAmount * (1 ether);
        deadline = block.timestamp + _deadline;
        admin = _ea;
    }
}
```

Call details:  
[call] from: 0x852f0c9f79ec07b2cbF7e518B60420cF7559f11B  
to: CrowdFunding.contributors(address)  
data: 0x1f6...9f11b

## Refund to Account 4 after deadline.

This screenshot shows the same setup as the previous one, but with a different account selected. Account 4 is connected and has a balance of 99.9984 ETH. A refund transaction is shown in the history, indicating a transfer of 0 ETH from account 3 to account 4.

Call details:  
[block:6 txIndex:0] from: 0x852...9f11b to: CrowdFunding.refund()  
value: 0 wei data: 0x590...1ae3 logs: 0 hash: 0x7ea...9fee5

## 2) CryptoFlight

The screenshot shows the Remix Ethereum IDE interface. On the left is a sidebar with various icons. The main area has tabs for "Home", "CrowdFunding.sol", and "CryptoFlight.sol". The "CryptoFlight.sol" tab is active, displaying the following Solidity code:

```
pragma solidity >=0.5.0 <0.6.0;

contract CryptoFlightFactory {
    CryptoFlight[] public flightsDeployed;

    function createFlight(uint256 minBid, string memory departure, string memory destination) public {
        CryptoFlight flight = new CryptoFlight(minBid, departure, destination, msg.sender);
        flightsDeployed.push(flight);
    }

    function getFlightDetailsDeployed() public view returns (CryptoFlight[] memory) {
        return flightsDeployed;
    }
}

contract CryptoFlight {
    struct Traveller {
        uint256 bid;
        address payable user;
        bool confirmed;
    }

    Traveller[] public travellers;
    address payable adminFlight;
    uint256 minBid;
    string departure;
    string destination;
    bool booked;

    modifier onlyAdmin() {
        require(msg.sender == adminFlight);
    }

    constructor(uint256 _minBid, string memory _departure, string memory _destination, address payable _adminFlight) public payable {
        adminFlight = _adminFlight;
        minBid = _minBid;
        departure = _departure;
        destination = _destination;
        booked = false;
    }
}
```

Below the code editor, there's a "flightsDeployed" dropdown set to 2, and a "getFlightDetail..." button. The "Low level interactions" section shows a "Transact" button. The right side of the interface shows a Ganache Local account summary for "Account 2" with 99.9062 ETH, and a transaction history section.

### Add Traveller 1

The screenshot shows the Remix Ethereum IDE interface. The "CryptoFlight.sol" tab is active, displaying the same Solidity code as the previous screenshot. The "Low level interactions" section now includes a "addTraveller" button. The right side of the interface shows a Ganache Local account summary for "Account 3" with 99.9961 ETH, and a transaction history section.

## Add Traveller 2.

The screenshot shows the Remix Ethereum IDE interface. On the left, there's a sidebar with various icons. In the center, the Solidity code for the `CryptoFlight` contract is displayed. A transaction is being prepared to call the `confirmFlight` function, with the parameter `_seats` set to "3". The right side of the interface shows the Ganache Local blockchain, which has Account 4 (0x852F0C9F79ecD7B2cbF7e518B60420cF7559f11B) connected and containing 99.997 ETH. The Activity tab shows a recent "Contract Interaction" and a "Refund". At the bottom, a transaction is shown with the call data: `[call] from: 0x852F0C9F79ecD7B2cbF7e518B60420cF7559f11B to: CryptoFlight.travellers(uint256) data: 0x65b...00001`.

```
pragma solidity >=0.5.0 <0.6.0;
contract CryptoFlightFactory {
    CryptoFlight[] public flightsDeployed;
    function createFlight(uint256 minBid, string memory departure, string memory destination) public returns (CryptoFlight flight) {
        flight = new CryptoFlight(minBid, departure, destination, msg.sender);
        flightsDeployed.push(flight);
    }
    function getFlightDetailsDeployed() public view returns (CryptoFlight[] memo) {
        return flightsDeployed;
    }
}
contract CryptoFlight {
    struct Traveller {
        uint256 bid;
        address payable user;
        bool confirmed;
    }
    Traveller[] public travellers;
    address payable adminFlight;
    uint256 minBid;
    string departure;
    string destination;
    bool booked;
    modifier onlyAdmin() {
        require(msg.sender == adminFlight);
    }
    constructor(uint256 _minBid, string memory _departure, string memory _destination, address payable _adminFlight) public payable {
        adminFlight = _adminFlight;
        minBid = _minBid;
        departure = _departure;
        destination = _destination;
        booked = false;
    }
    function confirmFlight(uint256 _bid, address payable _user) public onlyAdmin {
        Traveller storage traveller = travellers[_bid];
        traveller.user = _user;
        traveller.confirmed = true;
    }
    function getFlightDetails() public view returns (Traveller[] memo) {
        return travellers;
    }
}
```

Confirmed is set to true after flight is confirmed by the admin.

This screenshot shows the same Remix Ethereum IDE setup as the previous one, but with different transaction details. A transaction is being prepared to call the `getFlightDetails` function, with the parameter `_seats` set to "3". The Ganache Local blockchain now shows Account 2 (0xCD63B...7F21) connected and containing 99.9041 ETH. The Activity tab shows two recent "Contract Interaction" events. At the bottom, a transaction is shown with the call data: `[call] from: 0xCD63B...7F21 to: CryptoFlight.travellers(uint256) data: 0x65b...00000`.

```
pragma solidity >=0.5.0 <0.6.0;
contract CryptoFlightFactory {
    CryptoFlight[] public flightsDeployed;
    function createFlight(uint256 minBid, string memory departure, string memory destination) public returns (CryptoFlight flight) {
        flight = new CryptoFlight(minBid, departure, destination, msg.sender);
        flightsDeployed.push(flight);
    }
    function getFlightDetailsDeployed() public view returns (CryptoFlight[] memo) {
        return flightsDeployed;
    }
}
contract CryptoFlight {
    struct Traveller {
        uint256 bid;
        address payable user;
        bool confirmed;
    }
    Traveller[] public travellers;
    address payable adminFlight;
    uint256 minBid;
    string departure;
    string destination;
    bool booked;
    modifier onlyAdmin() {
        require(msg.sender == adminFlight);
    }
    constructor(uint256 _minBid, string memory _departure, string memory _destination, address payable _adminFlight) public payable {
        adminFlight = _adminFlight;
        minBid = _minBid;
        departure = _departure;
        destination = _destination;
        booked = false;
    }
    function confirmFlight(uint256 _bid, address payable _user) public onlyAdmin {
        Traveller storage traveller = travellers[_bid];
        traveller.user = _user;
        traveller.confirmed = true;
    }
    function getFlightDetails() public view returns (Traveller[] memo) {
        return travellers;
    }
}
```