BLOCKCHAIN Project Report On Crowd Funding

Department of

Computer Science and Engineering

Prepared by: Anurag(18CSU029) Harshit Goel(18CSU078)

Rubal(18CSU180) Raveesh(18CSU198)

Submitted to: Dr. Pooja Sapra

The NorthCap University

Gurugram, Haryana

CROWD FUNDING BASED ON BLOCKCHAIN

Approved by:

____ Dr. Pooja Sapra

____

Date

CROWD FUNDING BASED ON BLOCKCHAIN

I certify that *Anurag*, *Harshit Goel*, *Rubal* and *Raveesh* has met the requirements for format contained in the University format manual. The project embodies results of the original work, and studies as carried out by the students and the contents of the project do not form basis for the award of any degree to the candidate or to anybody else from this or any other University/Institution.

Dr. Pooja Sapra **Date**

Department of Computer Science

ACKNOWLEDGEMENTS

This project has helped us widen my technical expertise and would not have been possible without the assistance of our mentors and guides. It has been an excellent learning experience building this application. We would like to thank my project guide Dr. Pooja Sapra for giving us this opportunity to build this application and enhance my application developing skills required for gaining a competitive edge over other candidates in the job market. Dr. Pooja Sapra has been supportive and motivating to drive us towards completion of this project. All her valuable inputs have helped us to navigate through the project building phase and improve my approach and technologies. There are innumerable friends who helped me during this project, we are grateful to them.

Table of Contents

List of Figures

1. Introduction

Crowdfunding is the practice of funding a project or venture by raising small amounts of money from a large number of people, typically via the Internet. Crowdfunding is a form of crowdsourcing and alternative finance. With crowdfunding, it’s much easier for you to get your opportunity in front of more interested parties and give them more ways to help grow your business, from investing thousands in exchange for equity to contributing $20 in exchange for a first-run product or other reward.

This modern crowdfunding model is generally based on three types of actors: the project initiator who proposes the idea or project to be funded, individuals or groups who support the idea, and a moderating organization (the "platform") that brings the parties together to launch the idea.

2. Problem Statement

• Assume a person or team has started a project and wants some fund for its implementation.

• To fund the project funders can come and fund the project only if funding is in progress.

• If funding is either stopped or completed no one can fund the project anymore.

• All the funded money will be transacted to owner (person or team) only when funding is completed (i.e. project meets its funding target) until then all the amount transfers from funders account will be put on hold by contract.

• Once funding is completed no other user can further give funds for the project.

• If owner of campaign stops the funding all the funds given by others will be transacted back to them.

3. Design and Specifications

3.1 Aim of the project –

The main aim of this project is to collect fund from anyone who like to contribute in the project.

3.2 Project Features –

The main features of this project are –

a. Create your own campaign
b. Stop Funding
c. Funders
d. Project & Owner details.

3.3 User Classes and Characteristics –

a. Users

Owner – Responsible for Creating his/her campaign and receiving funds for the project and can stop funding also.

Funders – Can use the application for contributing some fund for the project.

3.4 Operating Environment –

Software Requirements:

- Solidity:

  Solidity is an object oriented high-level language used for writing smart contracts in the crowdfunding dapp. A statically typed language has features such as inheritance, libraries and complex user defined types. C++, Python and JavaScript, majorly influenced it. Solidity is compiled to bytecode that is executable by Ethereum virtual machine (EVM). Using solidity developers can write dapps that implement self-enforcing business logic contained in smart contracts, leaving an undeniable and permanent record of transactions.

- Metamask:

  Metamask is a chrome browser plugin that acts as a bridge between your browser and Ethereum blockchain by providing a secure identity vault, a user interface to manage multiple Ethereum wallets and sign blockchain transactions. It is one of the best ways to send transactions to Ethereum blockchain because it keeps a track of transaction execution and returns if any error occurs during mining or execution. It supports any ERC20 type token to be added to your wallet and trigger transaction on those ERC20 tokens. It is an Ethereum community open source project having more than million active users; hence, it is the most popular plugin to interact with blockchain.

3.5 Design Implementation and Constraints –

Input design:

a. Input to front end of system is designed.

b. User can fund the project or can create their own campaign.

Control design –

a. Users can fund the project only if funding is in progress. If funding is either stopped or completed no one can fund the project anymore.

b. All the funded money will be transacted to owner only when funding is completed i.e. funding is either equal to or greater than minimum fund he requires for project.

c. Once funding is completed no other user can further give funds for the project.

d. If owner of campaign stops the funding all the funds given by others will be transacted back to them.

4. Class Diagram

This explains the working of our project. It implies that whenever a project is created for getting funded, investment is open for all and there is an end date where investments would be stopped. If the amount required is reached, then funds are claimed else there is reclaiming of funds. This all get closed when the required amount of funds is achieved.

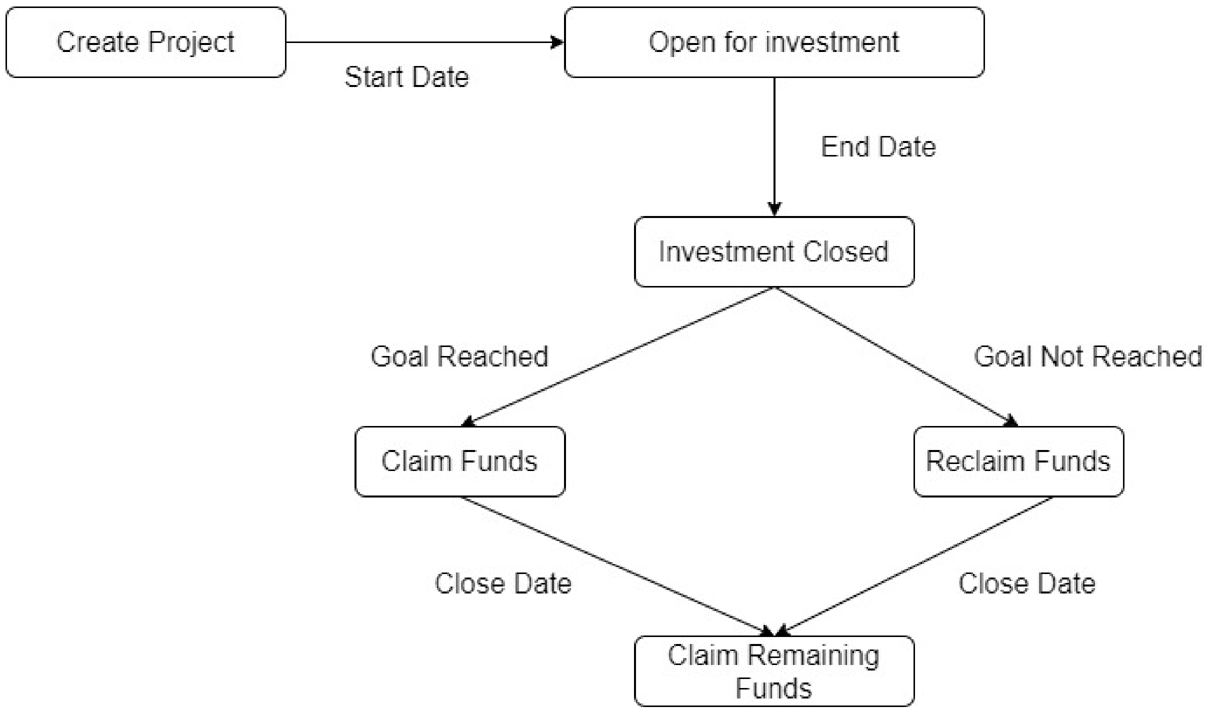


Figure 1 Class Diagram of project

5. Smart contract Specification

5.1 We need a way to save all the details –

CrowdFunding – This is the contract class.
FundingProject – A struct that hold project details as part of the contract.
Funder – A struct that holds funder details as part of the contract.

5.2 Instances for the structs –

Funder[] public funders; FundingProject public fundingproject;

5.3 Next, we created a modifier for some functions to control. It will check whether operation is performed by owner or not if not then raise an error.

require(msg.sender == fundingproject.owner,"Only owner can call this function.");

5.4 Creating Campaign The CrowdFunding() method is the constructor of our contract which would initialize the project details. It expects parameters that are used to initialize the FundingProject type.

fundingproject=FundingProject(_name,_email,_website,minimumfunds,amount raised,_owner,"F unding Started");

5.5 Funding Project Next code we will write is,

if (keccak256("Funding Stopped") == keccak256(fundingproject.status)) revert();

if (keccak256("Funding Completed") == keccak256(fundingproject.status)) revert();

This means we are comparing strings if they results in false then they will revert back. It will revert if funding is either completed or stopped by owner.
• keccak256 is a better version of the sha3 hashing algorithm. Find out more about hashing and these algorithms in our intro to blockchain. The function keccak256() accepts multiple arguments (parameters) separated by a comma, and then calculates a hash out of all of those put together. In this case it combines the blockhash before newest and the seed.

• revert function to generate exceptions to display an error and redo the current call.

Next, we initialize the Funder type and assign it to the funders array as part of the fundProject() method. The attribute amountraised of the FundingProject type will be updated to reflect the funder's amount. It means the funder has provided funding of that amount value. If the attribute amountraised is greater than minimumfunds i.e. if the project meets the funding target then the amount is sent to the beneficiary account i.e to owner's account.

This is done using the send() method of the FundingProject.owner attribute. Following shows the code snippet –

fundingproject.owner.send(fundingproject.amountraised)

The important thing to note is that the amount reaches the Owner's account only when minimum funding amount is raised. Until that duration, the amount transferred from Funder's account is put on hold by the contract.

fundProject() method uses the payable modifier to receive ethers from the application (i.e., from Funders account who is funding the project)

5.6 Stopping Funding Next, we define the stopFundRaising() method.

The said method returns all the money to the respective Funders (if owner stops the campaign or the funding target is not met) by iterating through the funders array and calling the send() method of Funder.fundedBy attribute for each funder object in the array. Following shows the code snippet –

funders[p].fundedby.send(funders[p].amount)

If funding is already completed(meets its target) then owner cannot stop the campaign. Following shows the code snippet–

if (keccak256(fundingproject.status)== keccak256("Funding Completed")) revert();

The stopFundRaising() method uses the payable modifier to send ethers from the contract to respective funders account. The stopFundRaising() method uses onlyOwner modifier so that no one can stop campaign except owner.

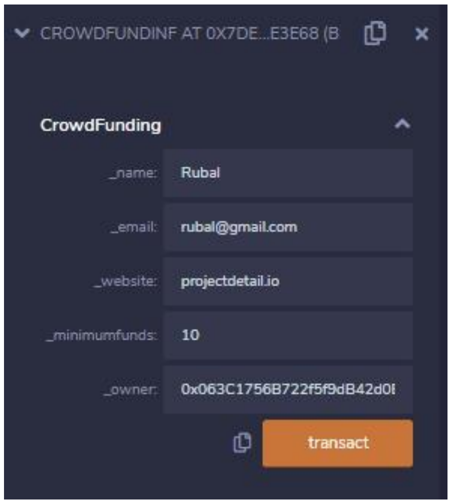
5.7 Get Project Status

Next, we define the getProjectStatus() constant method, that displays the status of the project at any interval.

6. Output specifications

6.1 Details Filling –

In this section we fill all the details of the project that is needed for funding.



CROWDFUNDINF AT 0X7DE...E3E68 (B

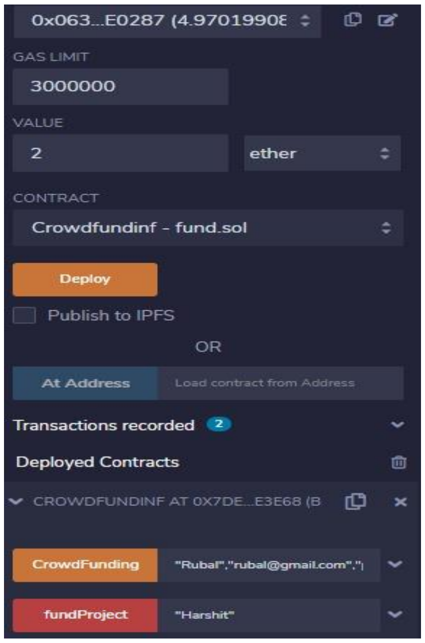
CrowdFunding

_name:	Rubal
_email:	rubal@gmail.com
_website:	projectdetail.io
_minimumfunds:	10
_owner:	0x063C1756B722f5f9dB42d0

Figure 2 Filling details of project

6.2 Funding Project –

A person can fund ethers in project



0x063...E0287 (4.9701990

GAS LIMIT

3000000

VALUE

2 ether

CONTRACT

Crowdfundinf - fund.sol

Transactions recorded 2

Deployed Contracts

CROWDFUNDINF AT 0X7DE...E3E68 (B

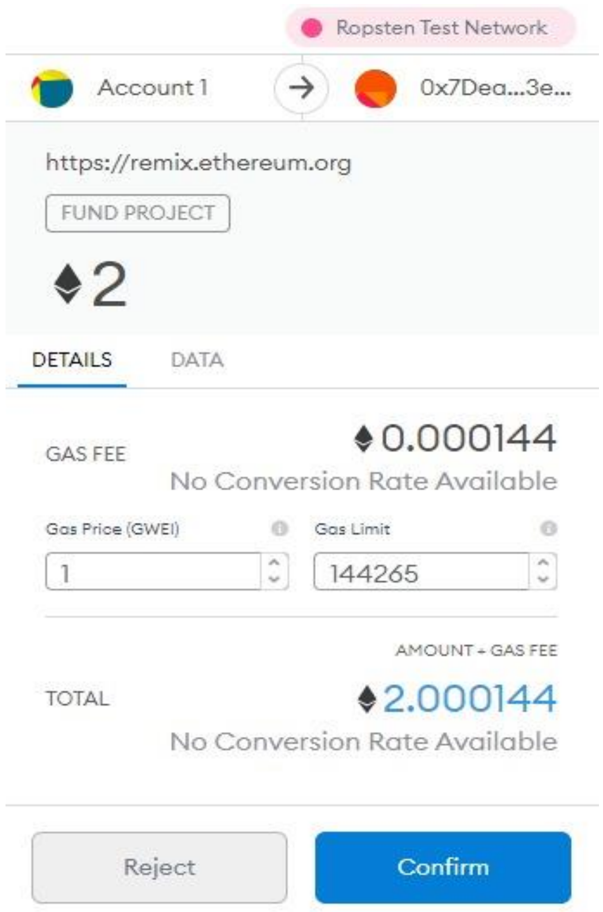
CrowdFunding "Rubal","rubal@gmail.com","

fundProject "Harshit"

Figure 3 funding 2ethers in project

6.3 Metamask –

Using metamask for transferring and making transaction of ethers.



Ropsten Test Network

Account 1 → 0x7Dea...3e...

https://remix.ethereum.org

FUND PROJECT

♦2

DETAILS DATA

GAS FEE ♦0.000144
No Conversion Rate Available

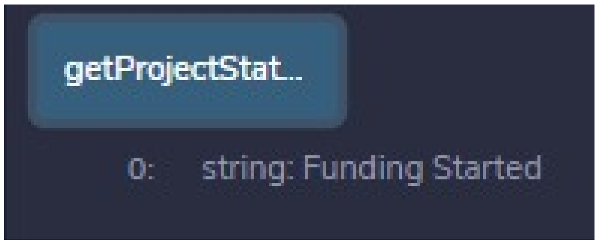
Gas Price (GWEI): 1
Gas Limit: 144265

AMOUNT + GAS FEE
TOTAL ♦2.000144
No Conversion Rate Available

Figure 4 Metamask transactions

6.4 Checking Project Status –

It can be checked from here weather the funding has started or not.

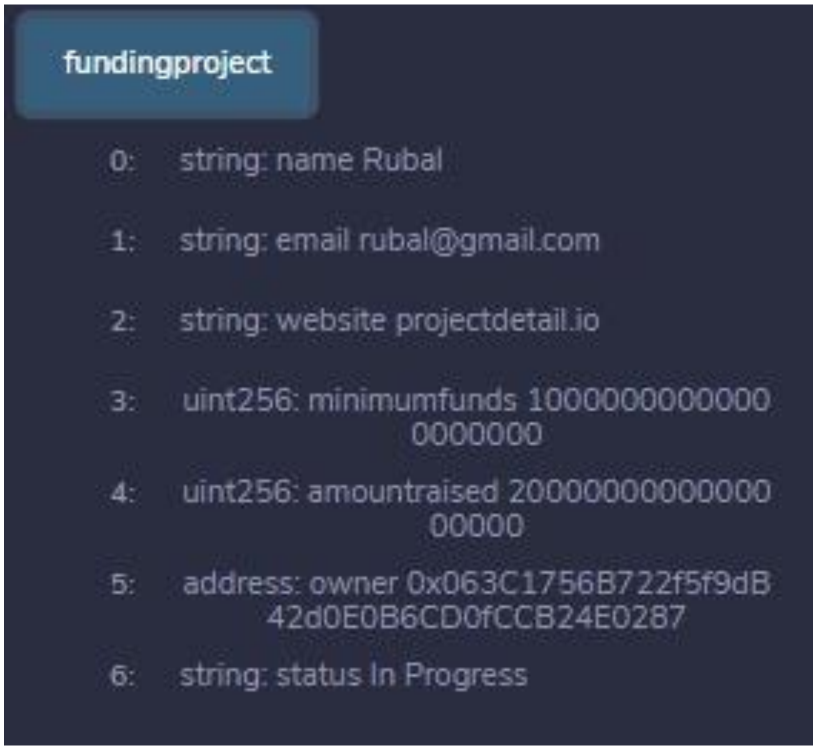


```
0:    string: Funding Started
```

Figure 5 Checking Project Status

6.5 Details check –

Here updated details will appear after funding has started.



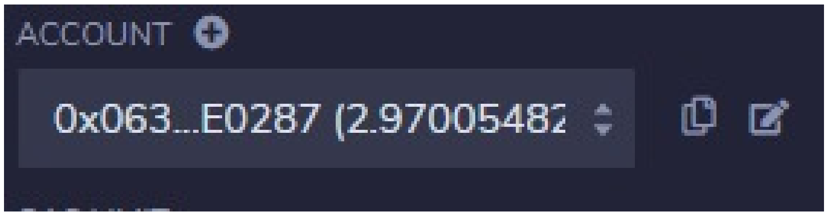
fundingproject

```
0:  string: name Rubal
1:  string: email rubal@gmail.com
2:  string: website projectdetail.io
3:  uint256: minimumfunds 1000000000000
                 0000000
4:  uint256: amountraised 20000000000000
                 00000
5:  address: owner 0x063C1756B722f5f9dB
        42d0E0B6CD0fCCB24E0287
6:  string: status In Progress
```

Figure 6 details updated after funded by a person

6.6 Amount Deducted –

After transaction has been ether will be reduced from person’s account.



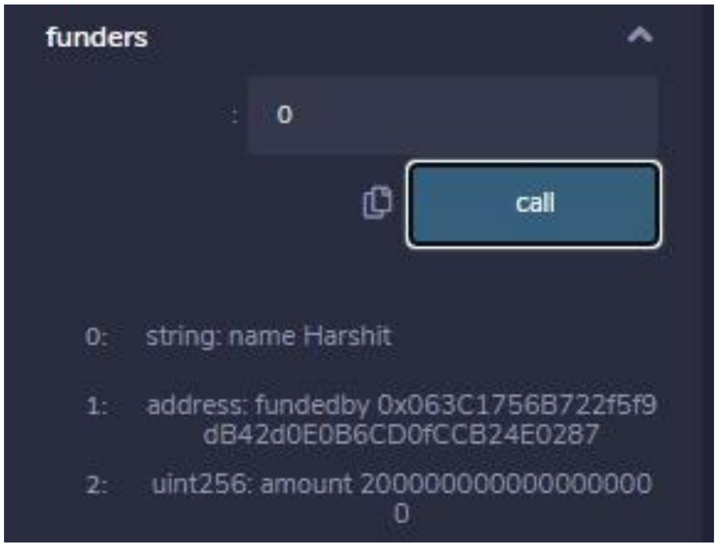
ACCOUNT

```
0x063...E0287 (2.97005482
```

Figure 7 amount deducted from his account

6.7 Funders details –

It will show all the details of funders.



```
0:   string: name Harshit
1:   address: fundedby 0x063C1756B722f5f9
       dB42d0E0B6CD0fCCB24E0287
2:   uint256: amount 20000000000000000
       0
```

Figure 8 details of funder

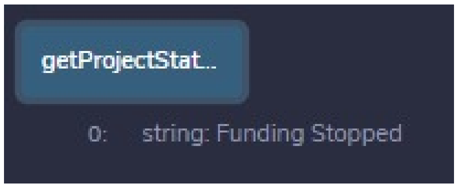
6.8 Stop Funding –

Funding gets stopped by clicking this icon.

Figure 9 Funding is stopped

6.9 Project status –

Status will be updated to "Funding Stopped" and no one can now able to fund the project and only owner can stop the funding



```
0:   string: Funding Stopped
```

Figure 10 Shows current status of project

6.10 Updated amount of account –

After stooping the funding ethers funded by funders will be transacted back to their account when funding is completed only then all the funded amount " amount raised" will be transacted to the owners account till then smart contract will hold the transaction.

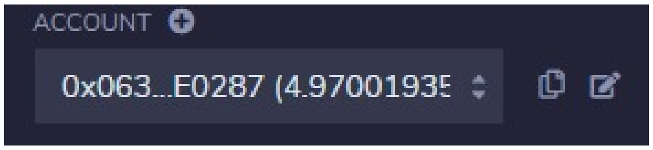
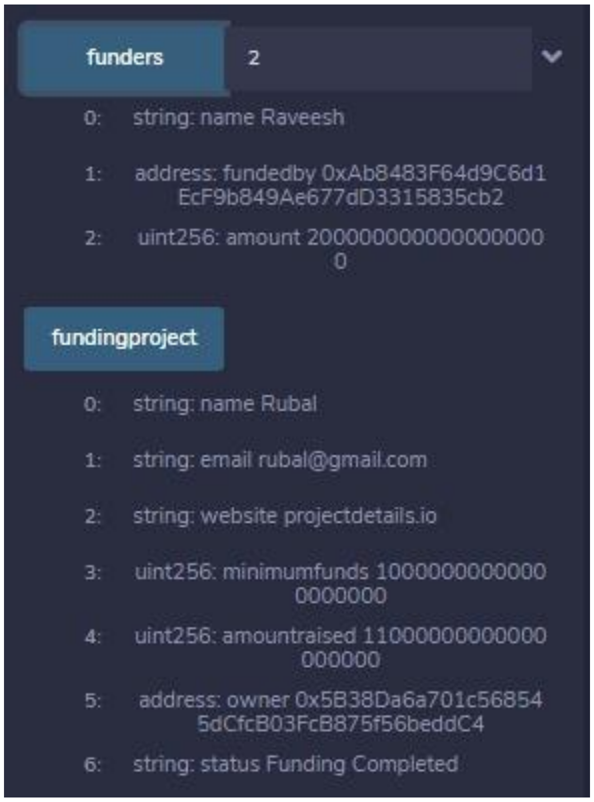


Figure 11 Increased ether in owners account

6.11 Funding completed –

This the time when funding is completed and no more funds are required. At time when funding is completed and no more funds are required at that time all the funded amount will be transacted to owner's account and after completion of funding owner can't stop the funding and no one can anymore fund to the project

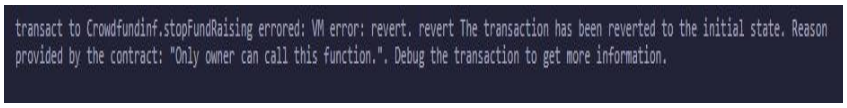


```
funders  2
0: string: name Raveesh
1: address: fundedby 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2
2: uint256: amount 2000000000000000000

fundingproject
0: string: name Rubal
1: string: email rubal@gmail.com
2: string: website projectdetails.io
3: uint256: minimumfunds 10000000000000000000
4: uint256: amountraised 11000000000000000000
5: address: owner 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
6: string: status Funding Completed
```

Figure 12 funding is completed

6.12 Error 1 –

When anyone other than owner tries to stop the funding it will raise the error.

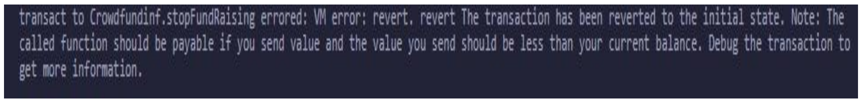


```
transact to Crowdfundinf.stopFundRaising errored: VM error: revert. revert The transaction has been reverted to the initial state. Reason
provided by the contract: "Only owner can call this function.". Debug the transaction to get more information.
```

Figure 13 Error 1

6.13 Error 2 –

After completion of funding even owner will not able to stop the funding.



```
transact to Crowdfundinf.stopFundRaising errored: VM error: revert. revert The transaction has been reverted to the initial state. Note: The
called function should be payable if you send value and the value you send should be less than your current balance. Debug the transaction to
get more information.
```

Figure 14 Error 2

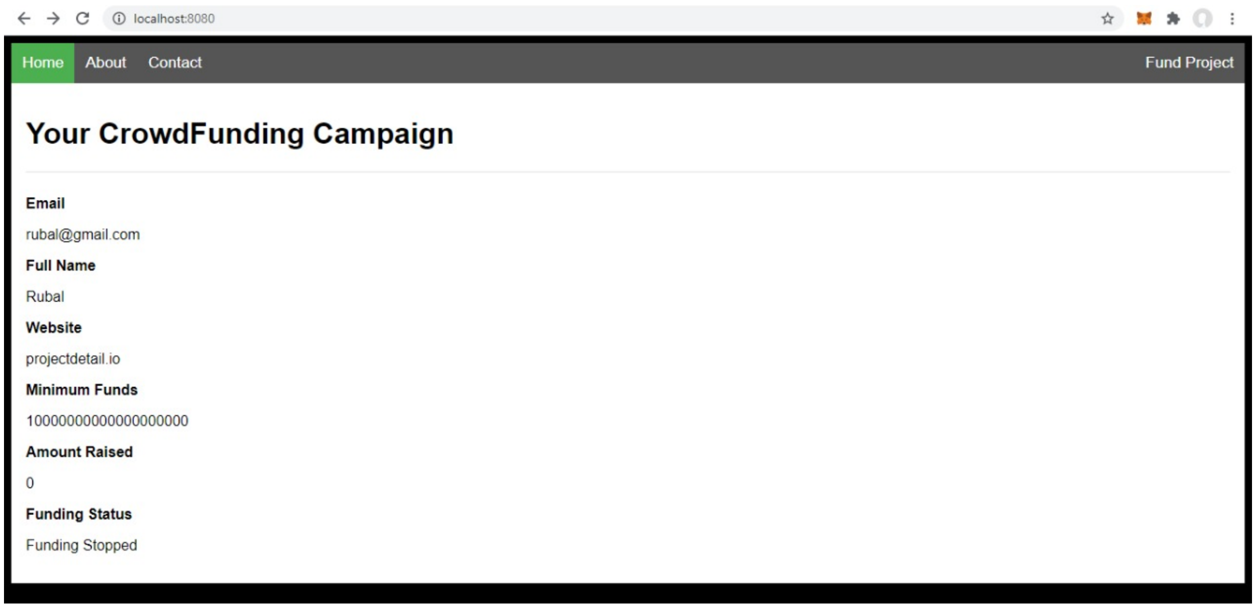
7. Smart Contract Code –

```
//Funding project details
pragma solidity ^0.4.26;
contract CrowdfundinfProject{
   struct FundingProject {
      string name;
      string email;
      string website;
      uint minimumfunds;
      uint amountraised;
      address owner;
      string status;
   }

   //Funder who funds project.
   struct Funder {
      string name;
      address fundedby;
      uint amount;
   }

   //Multiple funders can fund project
   Funder[] public funders;
   FundingProject public fundingproject;

   //modifier
   modifier onlyOwner() {
      require(msg.sender == fundingproject.owner,"Only owner can call this function.");
      _;
   }

   function CrowdFunding(string _name, string _email, string _website, uint
_minimumfunds, address _owner) public
   {
       //convert to ether
      uint minimumfunds = _minimumfunds * 1 ether;
      uint amountraised = 0;
      fundingproject =
FundingProject(_name,_email,_website,minimumfunds,amountraised,_owner,"Funding
Started");
   }
   function fundProject( string name) public payable {
      if (keccak256("Funding Stopped")== keccak256(fundingproject.status)) revert();
```

```
        if (keccak256("Funding Completed")== keccak256(fundingproject.status)) revert();
        funders.push(Funder({
            name: name,
            fundedby: msg.sender,
            amount: msg.value})
        );
        fundingproject.amountraised = fundingproject.amountraised + msg.value ;

        if (fundingproject.amountraised >= fundingproject.minimumfunds) {
            if(!fundingproject.owner.send(fundingproject.amountraised )) revert();

                fundingproject.status = "Funding Completed";
            }
            else {
                fundingproject.status = "In Progress";
            }
    }
    function stopFundRaising() public payable onlyOwner {
             if    (keccak256(fundingproject.status)==    keccak256("Funding    Completed"))
revert();
                fundingproject.status = "Funding Stopped";
             //return money to all funders
             for (uint p = 0; p < funders.length; p++) {
                        if(!funders[p].fundedby.send(funders[p].amount)) throw;
             }
             fundingproject.amountraised = 0;
    }
    function getProjectStatus() public constant            returns(string) {
                        return (fundingproject.status);
    }
}
```

8. Frontend Codes –

8.1 Index.html



Your CrowdFunding Campaign

Email

rubal@gmail.com

Full Name

Rubal

Website

projectdetail.io

Minimum Funds

1000000000000000000

Amount Raised

0

Funding Status

Funding Stopped

Figure 15 Homepage of Smart Contract

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body {
  font-family: Arial, Helvetica, sans-serif;
  background-color: black;
}

* {
  box-sizing: border-box;
}
```

```
.navbar {
  width: 100%;
  background-color: #555;
  overflow: auto;
}

/* Navbar links */
.navbar a {
  float: left;
  text-align: center;
  padding: 12px;
  color: white;
  text-decoration: none;
  font-size: 17px;
}

/* Navbar links on mouse-over */
.navbar a:hover {
  background-color: #000;
}

/* Current/active navbar link */
.active {
  background-color: #4CAF50;
}

.navbar .login-container {
  float: right;
```

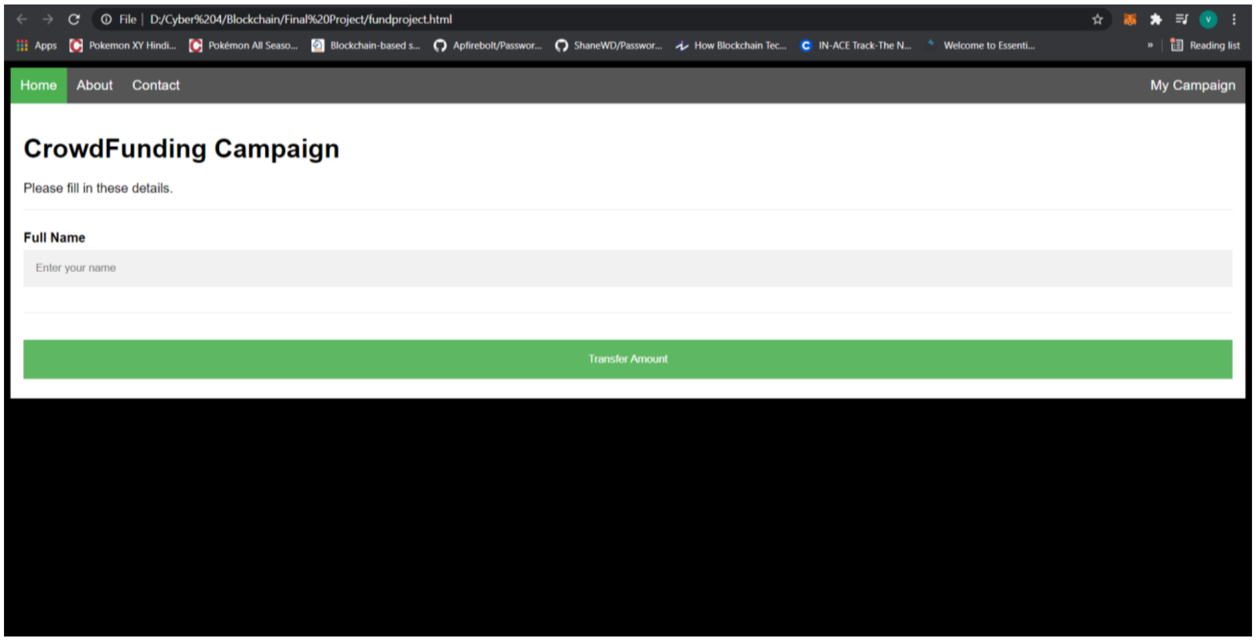
```
}

.topnav .login-container button {
  float: right;
  padding: 6px;
  margin-top: 8px;
  margin-right: 16px;
  background: #ddd;
  font-size: 17px;
  border: none;
  cursor: pointer;
}

.navbar .login-container button:hover {
  background: #ccc;
}

/* Add responsiveness - will automatically display the navbar vertically instead of
horizontally on screens less than 500 pixels */
@media screen and (max-width: 500px) {
  .navbar a {
    float: none;
    display: block;
  }
}

/* Add padding to containers */
.container {
  padding: 16px;
  background-color: white;
}

/* Full-width input fields */
input[type=text], input[type=password] {
  width: 100%;
  padding: 15px;
  margin: 5px 0 22px 0;
  display: inline-block;
  border: none;
  background: #f1f1f1;
}

input[type=number], input[type=password] {
  width: 100%;
  padding: 15px;
  margin: 5px 0 22px 0;
  display: inline-block;
  border: none;
  background: #f1f1f1;
}

input[type=address], input[type=password] {
  width: 100%;
  padding: 15px;
  margin: 5px 0 22px 0;
  display: inline-block;
  border: none;
  background: #f1f1f1;
}

input[type=text]:focus, input[type=password]:focus {
  background-color: #ddd;
  outline: none;
}

/* Overwrite default styles of hr */
hr {
  border: 1px solid #f1f1f1;
  margin-bottom: 25px;
}

/* Set a style for the submit button */
.registerbtn {
  background-color: #4CAF50;
  color: white;
  padding: 16px 20px;
  margin: 8px 0;
  border: none;
  cursor: pointer;
  width: 100%;
  opacity: 0.9;
}

.registerbtn:hover {
  opacity: 1;
```

```
}

/* Add a blue text color to links */
a {
  color: dodgerblue;
}

/* Set a grey background color and center the text of the "sign in" section */
.signin {
  background-color: #f1f1f1;
  text-align: center;
}

</style>
</head>
<body>

<form action="/action_page.php">
  <div class="navbar">
    <a class="active" href="index.html"><i class="fa fa-fw fa-home"></i> Home</a>
    <a href="#"><i class="fa fa-fw fa-search"></i> About</a>
    <a href="#"><i class="fa fa-fw fa-envelope"></i> Contact</a>
    <div class="login-container">
        <a href="fundproject.html"><i class="fa fa-fw fa-envelope"></i> Fund Project</a>
    </div>
  </div>
  <div class="container">
    <h1>Your CrowdFunding Campaign</h1>
    <hr>


    <label for="email"><b>Email</b></label>

    <p id="email"></p>


    <label for="name"><b>Full Name</b></label>

    <p id="names"></p>


    <label for="site"><b>Website</b></label>

    <p id="site"></p>


    <label for="fund"><b>Minimum Funds</b></label>

    <p id="min"></p>


    <label for="amount"><b>Amount Raised</b></label>

    <p id="amount"></p>


    <label for="fundstatus"><b>Funding Status</b></label>

    <p id="status"></p>


</form>


<script src="https://cdn.jsdelivr.net/gh/ethereum/web3.js@1.0.0-
beta.36/dist/web3.min.js"></script>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
crossorigin="anonymous"></script>


<script>

        var contract;
```

```
          var web3 = new Web3(web3.currentProvider);

$(document).ready(function()
{
  web3 = new Web3(web3.currentProvider);

  var address = "0x7Deafb9F5c8E064Ceb5eD4b7A30ef1f512CE3e68";
  var abi =[
          {
                  "constant": false,
                  "inputs": [
                          {
                                  "name": "_name",
                                  "type": "string"
                          },
                          {
                                  "name": "_email",
                                  "type": "string"
                          },
                          {
                                  "name": "_website",
                                  "type": "string"
                          },
                          {
                                  "name": "_minimumfunds",
                                  "type": "uint256"
                          },
                          {
```

```
				"name": "_owner",
				"type": "address"
			}
		],
		"name": "CrowdFunding",
		"outputs": [],
		"payable": false,
		"stateMutability": "nonpayable",
		"type": "function"
	},
	{
		"constant": false,
		"inputs": [
			{
				"name": "name",
				"type": "string"
			}
		],
		"name": "fundProject",
		"outputs": [],
		"payable": true,
		"stateMutability": "payable",
		"type": "function"
	},
	{
		"constant": false,
		"inputs": [],
		"name": "stopFundRaising",
        "outputs": [],
        "payable": true,
        "stateMutability": "payable",
        "type": "function"
    },
    {
        "constant": true,
        "inputs": [
            {
                "name": "",
                "type": "uint256"
            }
        ],
        "name": "funders",
        "outputs": [
            {
                "name": "name",
                "type": "string"
            },
            {
                "name": "fundedby",
                "type": "address"
            },
            {
                "name": "amount",
                "type": "uint256"
            }
        ],
		"payable": false,
		"stateMutability": "view",
		"type": "function"
	},
	{
		"constant": true,
		"inputs": [],
		"name": "fundingproject",
		"outputs": [
			{
				"name": "name",
				"type": "string"
			},
			{
				"name": "email",
				"type": "string"
			},
			{
				"name": "website",
				"type": "string"
			},
			{
				"name": "minimumfunds",
				"type": "uint256"
			},
			{
				"name": "amountraised",
				"type": "uint256"
				},
				{
					"name": "owner",
					"type": "address"
				},
				{
					"name": "status",
					"type": "string"
				}
			],
			"payable": false,
			"stateMutability": "view",
			"type": "function"
		},
		{
			"constant": true,
			"inputs": [],
			"name": "getProjectAmt",
			"outputs": [
				{
					"name": "",
					"type": "uint256"
				}
			],
			"payable": false,
			"stateMutability": "view",
			"type": "function"
		},
{
	"constant": true,
	"inputs": [],
	"name": "getProjectEmail",
	"outputs": [
		{
			"name": "",
			"type": "string"
		}
	],
	"payable": false,
	"stateMutability": "view",
	"type": "function"
},
{
	"constant": true,
	"inputs": [],
	"name": "getProjectFund",
	"outputs": [
		{
			"name": "",
			"type": "uint256"
		}
	],
	"payable": false,
	"stateMutability": "view",
	"type": "function"
},
{
	"constant": true,
	"inputs": [],
	"name": "getProjectName",
	"outputs": [
		{
			"name": "",
			"type": "string"
		}
	],
	"payable": false,
	"stateMutability": "view",
	"type": "function"
},
{
	"constant": true,
	"inputs": [],
	"name": "getProjectStatus",
	"outputs": [
		{
			"name": "",
			"type": "string"
		}
	],
	"payable": false,
	"stateMutability": "view",
	"type": "function"
},
        {
                "constant": true,
                "inputs": [],
                "name": "getProjectWebsite",
                "outputs": [
                        {
                                "name": "",
                                "type": "string"
                        }
                ],
                "payable": false,
                "stateMutability": "view",
                "type": "function"
        }
];

  contract = new web3.eth.Contract(abi, address);
  contract.methods.getProjectStatus().call().then(function(stat)
  {
    $('#status').html(stat);
  })
  contract.methods.getProjectName().call().then(function(name)
  {
    $('#names').html(name);
  })
      contract.methods.getProjectEmail().call().then(function(mail)
  {
    $('#email').html(mail);
    })
            contract.methods.getProjectWebsite().call().then(function(webs)
    {
        $('#site').html(webs);
    })
            contract.methods.getProjectFund().call().then(function(min)
    {
        $('#min').html(min);
    })
            contract.methods.getProjectAmt().call().then(function(amt)
    {
        $('#amount').html(amt);
    })
  })
</script>
</body>
</html>
```

8.2 Fundproject.html


File | D:/Cyber%204/Blockchain/Final%20Project/fundproject.html
Home
About
Contact
My Campaign

CrowdFunding Campaign

Please fill in these details.

Full Name

Enter your name

Transfer Amount

Figure 16 fundproject.html

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body {
  font-family: Arial, Helvetica, sans-serif;
  background-color: black;
}

* {
  box-sizing: border-box;
}

.navbar {
  width: 100%;
  background-color: #555;
  overflow: auto;
}

/* Navbar links */
.navbar a {
  float: left;
  text-align: center;
  padding: 12px;
  color: white;
  text-decoration: none;
  font-size: 17px;
}

/* Navbar links on mouse-over */
.navbar a:hover {
  background-color: #000;
}

/* Current/active navbar link */
.active {
  background-color: #4CAF50;
}

.navbar .login-container {
  float: right;
}

.topnav .login-container button {
  float: right;
  padding: 6px;
  margin-top: 8px;
  margin-right: 16px;
  background: #ddd;
  font-size: 17px;
  border: none;
  cursor: pointer;
}

.navbar .login-container button:hover {
  background: #ccc;
}

/* Add responsiveness - will automatically display the navbar vertically instead of
horizontally on screens less than 500 pixels */
@media screen and (max-width: 500px) {
  .navbar a {
    float: none;
    display: block;
  }
}
```

```
/* Add padding to containers */
.container {
  padding: 16px;
  background-color: white;
}

/* Full-width input fields */
input[type=text], input[type=password] {
  width: 100%;
  padding: 15px;
  margin: 5px 0 22px 0;
  display: inline-block;
  border: none;
  background: #f1f1f1;
}

input[type=number], input[type=password] {
  width: 100%;
  padding: 15px;
  margin: 5px 0 22px 0;
  display: inline-block;
  border: none;
  background: #f1f1f1;
}

input[type=address], input[type=password] {
  width: 100%;
  padding: 15px;
  margin: 5px 0 22px 0;
  display: inline-block;
  border: none;
  background: #f1f1f1;
}

input[type=text]:focus, input[type=password]:focus {
  background-color: #ddd;
  outline: none;
}

/* Overwrite default styles of hr */
hr {
  border: 1px solid #f1f1f1;
  margin-bottom: 25px;
}

/* Set a style for the submit button */
.registerbtn {
  background-color: #4CAF50;
  color: white;
  padding: 16px 20px;
  margin: 8px 0;
  border: none;
  cursor: pointer;
  width: 100%;
  opacity: 0.9;
}

.registerbtn:hover {
  opacity: 1;
}

/* Add a blue text color to links */
a {
  color: dodgerblue;
}

/* Set a grey background color and center the text of the "sign in" section */
.signin {
  background-color: #f1f1f1;
  text-align: center;
}

</style>
</head>
<body>
<script src="https://cdn.jsdelivr.net/gh/ethereum/web3.js@1.0.0-
beta.36/dist/web3.min.js"></script>
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
crossorigin="anonymous"></script>
<form action="">
  <div class="navbar">
    <a class="active" href="index.html"><i class="fa fa-fw fa-home"></i> Home</a>
    <a href="#"><i class="fa fa-fw fa-search"></i> About</a>
    <a href="#"><i class="fa fa-fw fa-envelope"></i> Contact</a>
    <div class="login-container">
        <a href="#"><i class="fa fa-fw fa-user"></i> My Campaign</a>
    </div>
  </div>
  <div class="container">
    <h1>CrowdFunding Campaign</h1>
    <p>Please fill in these details.</p>
    <hr>

    <label for="name"><b>Full Name</b></label>
    <input type="text" placeholder="Enter your name" name="name" id="name"
required>

    <!--<label for="fund"><b>Minimum Fund Giving</b></label>
    <input type="number" placeholder="Enter the fund you want to transfer"
name="fund" id="fund" required>

    <label for="add"><b>Address</b></label>
    <input type="address" placeholder="your address" name="add" id="add"
required> -->

    <hr>
    <button id="button" type="submit" class="registerbtn">Transfer
Amount</button>
  </div>

</form>
<script>
     var contract;
     var web3;

  $(document).ready(function()
  {
    web3 = new Web3(web3.currentProvider);

    var address = "0x7Deafb9F5c8E064Ceb5eD4b7A30ef1f512CE3e68";
    var abi =[
     {
                "constant": false,
                "inputs": [
                        {
                                "name": "_name",
                                "type": "string"
                        },
                        {
```

```
                "name": "_email",
                "type": "string"
            },
            {
                "name": "_website",
                "type": "string"
            },
            {
                "name": "_minimumfunds",
                "type": "uint256"
            },
            {
                "name": "_owner",
                "type": "address"
            }
        ],
        "name": "CrowdFunding",
        "outputs": [],
        "payable": false,
        "stateMutability": "nonpayable",
        "type": "function"
    },
    {
        "constant": false,
        "inputs": [
            {
                "name": "name",
                "type": "string"
            }
        ],
        "name": "fundProject",
        "outputs": [],
        "payable": true,
        "stateMutability": "payable",
        "type": "function"
    },
    {
        "constant": false,
        "inputs": [],
        "name": "stopFundRaising",
        "outputs": [],
        "payable": true,
        "stateMutability": "payable",
		"type": "function"
	},
	{
		"constant": true,
		"inputs": [
			{
				"name": "",
				"type": "uint256"
			}
		],
		"name": "funders",
		"outputs": [
			{
				"name": "name",
				"type": "string"
			},
			{
				"name": "fundedby",
				"type": "address"
			},
			{
				"name": "amount",
				"type": "uint256"
			}
		],
		"payable": false,
		"stateMutability": "view",
		"type": "function"
	},
	{
		"constant": true,
		"inputs": [],
		"name": "fundingproject",
		"outputs": [
			{
				"name": "name",
				"type": "string"
			},
			{
				"name": "email",
				"type": "string"
			},
			{
				"name": "website",
				"type": "string"
			},
			{
				"name": "minimumfunds",
				"type": "uint256"
			},
			{
				"name": "amountraised",
				"type": "uint256"
			},
			{
				"name": "owner",
				"type": "address"
			},
			{
				"name": "status",
				"type": "string"
			}
		],
		"payable": false,
		"stateMutability": "view",
		"type": "function"
	},
	{
		"constant": true,
		"inputs": [],
		"name": "getProjectAmt",
		"outputs": [
			{
				"name": "",
				"type": "uint256"
			}
		],
		"payable": false,
		"stateMutability": "view",
		"type": "function"
	},
	{
		"constant": true,
		"inputs": [],
		"name": "getProjectEmail",
		"outputs": [
				{
					"name": "",
					"type": "string"
				}
			],
			"payable": false,
			"stateMutability": "view",
			"type": "function"
		},
		{
			"constant": true,
			"inputs": [],
			"name": "getProjectFund",
			"outputs": [
				{
					"name": "",
					"type": "uint256"
				}
			],
			"payable": false,
			"stateMutability": "view",
			"type": "function"
		},
		{
			"constant": true,
			"inputs": [],
			"name": "getProjectName",
			"outputs": [
				{
					"name": "",
					"type": "string"
				}
			],
			"payable": false,
			"stateMutability": "view",
			"type": "function"
		},
		{
			"constant": true,
			"inputs": [],
			"name": "getProjectStatus",
			"outputs": [
				{
```

```
                    "name": "",
                    "type": "string"
                }
            ],
            "payable": false,
            "stateMutability": "view",
            "type": "function"
    },
    {
            "constant": true,
            "inputs": [],
            "name": "getProjectWebsite",
            "outputs": [
                {
                    "name": "",
                    "type": "string"
                }
            ],
            "payable": false,
            "stateMutability": "view",
            "type": "function"
    }
];

    contract = new web3.eth.Contract(abi, address);
  })
  $('#button').click(function()
  {
    var n= '';
    n= $('#name').val();
    web3.eth.getAccounts().then(function(accounts)
    {
      var acc= accounts[0];
      return contract.methods.fundProject(n).send({from: acc});
    }).then(function(tx)
    {
      console.log(tx);
    }).catch(function(tx)
    {
      console.log(tx);
    })
  })
</script>
```

```
</body>
</html>
```