

**SPRING 2024-25 COMP 4541**

**Decentralized Crowdfunding  
Platform**

Name: HASAN, Dewan Saadman

SID: 20920414

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Contract Functionality</b>	<b>2</b>
2.1	Create Campaign . . . . .	2
2.2	Campaign Details . . . . .	2
2.3	Contribution . . . . .	3
2.4	Withdraw Funds before Goal . . . . .	3
2.5	Withdrawing Funds After Deadline . . . . .	4
2.6	Refunding Contributors . . . . .	4
2.7	Getting Campaign Details . . . . .	4
<b>3</b>	<b>Testing On Remix Virtual Machine</b>	<b>5</b>
3.1	Basic Functionalities . . . . .	5
3.2	Withdrawing Funds Before Goal . . . . .	6
3.3	Refunding Contributors . . . . .	7
3.4	Withdrawing Funds After Deadline . . . . .	8
3.5	Other Security Checks . . . . .	9
<b>4</b>	<b>User Interface</b>	<b>10</b>
<b>5</b>	<b>Conclusion</b>	<b>12</b>

# 1 Introduction

Crowdfunding is a method for individuals, startups, and organizations to raise funds for their projects or ventures by soliciting small contributions from online communities. It provides a platform for creators and entrepreneurs to showcase their ideas and turn them into reality without traditional funding methods like bank loans or venture capital, or to raise funds for charitable causes, personal projects, or community initiatives. Through crowdfunding, backers, or contributors, donate small amounts of money to support a project, often in exchange for rewards, equity, or other incentives.

Crowdfunding has gained immense popularity in recent years, with platforms like Kickstarter, Indiegogo, GoFundMe, and Patreon leading the way. These platforms, although different in their approaches, all share the common goal of connecting creators with potential backers. For example, Kickstarter focuses on funding creative projects like art, music, film, games through their all-or-nothing funding model, while GoFundMe is more geared towards personal causes and charitable fundraising. Patreon, on the other hand, allows creators to earn recurring income by offering exclusive content and perks to their subscribers.

In view of this idea, we build a decentralized crowdfunding platform on the Ethereum testnet. The platform allows users to create and fund projects by calling a smart contract. The smart contract is deployed on the Ethereum blockchain. Similar to Kickstarter, we employ an all-or-nothing funding model, where the project creator must reach their funding goal within a specified time frame. If the goal is not met, the funds are returned to the contributors. In addition, we propose a voting-based mechanism to allow the community to decide whether the creator would be able to withdraw the fund before the goal and time limit is reached.

## 2 Contract Functionality

In this section we will discuss the functionality of our smart contract. The contract is written in Solidity and is deployed on the Ethereum testnet. The contract has two types of users: the project or campaign creators and the contributors, who donate to the projects. The contract has the following functionalities:

### 2.1 Create Campaign

The project creator can create a campaign by calling the `createCampaign` function. It takes the following parameters:

- **title:** The title of the campaign.
- **description:** A brief description of the campaign.
- **goalAmount:** The funding goal for the campaign.
- **durationinDays:** The duration of the campaign in number of days.
- **image (optional):** The URL of the campaign image.

### 2.2 Campaign Details

Once created, a campaign is assigned a unique ID and is stored in the contract's mapping. A campaign stores the following additional information:

- **creator:** The address of the project creator.
- **fundsRaised:** The total amount of ETH raised for the campaign.
- **goalReached:** A boolean value indicating whether the amount raised met the goal or not.
- **isWithdrawn:** A boolean value indicating whether the funds have been withdrawn or not.
- **yesbeforegoal:** An int value indicating weighted vote in favour of early withdrawal.
- **totalvotes:** An int value indicating total votes for early withdrawal.
- **withdrawnbeforegoal:** A boolean value indicating early withdrawal.
- **afterwithdraw:** Contributor address mapping to boolean value indicating whether the contribution was made after the early withdrawal or not

- **contributors:** An array of addresses of contributors who have donated to the campaign.
- **contributions:** A mapping of contributor addresses to their respective contributions.

The campaign model is based on all-or-nothing funding model, where the project must reach its goal limit within the specified time frame. If the goal is not met, the contributors can call a function to get their funds back. However, we allow each campaign to withdraw the funds before the goal is reached at most one time. This process is voting based; when contributors donate to the campaign, they can choose to vote in favour of early withdrawal. The votes are weighted based on the amount contributed, which is indicated by **yesbeforegoal**. Early withdrawal is allowed if the number of yes votes is greater than half of the total votes, and possible only once throughout the entire duration of the campaign. Additionally, once the campaign deadline has passed, it will not be possible to withdraw partial funds in the event that the goal is not met.

## 2.3 Contribution

Contributors can donate to the campaign by calling the **contribute** function. It has a unique parameter **canwithdrawbefore**, which indicates whether the contributor wants to vote in favour of early withdrawal or not. If true, and the campaign has not been withdrawn early, the contributor's vote is counted towards the **yesbeforegoal** variable. The contributor's address is added to the campaign contributors array, and the contribution amount is stored in the contributions mapping.

## 2.4 Withdraw Funds before Goal

As mentioned earlier, the project creator can withdraw the funds before the goal is reached if the voting is in favour of early withdrawal. The creator can call the **withdrawFundsBeforeGoal** function, which checks if the early withdrawal is allowed. This goes through the following checks:

- The creator of the campaign must be the one calling the function.
- The campaign must be active and is not called after the deadline.
- Target amount is greater than the amount raised. Otherwise the creator can withdraw full amount after the deadline.
- Fund early withdrawal is allowed only once.
- The number of yes votes must be greater than half of the total votes.

In this case, the funds are transferred to the creator's address, `fundsRaised` is set to zero, and the target amount is reduced to the amount raised. The `withdrawnbeforegoal` variable is set to true, and the contribution array is updated. Any more contributions made after the early withdrawal will be marked as true in the `afterwithdraw` mapping, so that the contributors can get their funds back if the end goal is not reached.

## 2.5 Withdrawing Funds After Deadline

The project creator can withdraw the funds after the deadline by calling the `withdrawFunds` function. It can only be activated if

- The creator of the campaign must be the one calling the function.
- The campaign deadline must have passed.
- The campaign goal must be reached.
- Funds must not have been fully withdrawn before.

If the conditions are met, the funds are transferred to the creator's address, and the variables are changed to indicate that the funds have been withdrawn.

## 2.6 Refunding Contributors

Contributors can get their funds back by calling the `requestRefund` function. Refunds can be requested if the campaign deadline has passed and the goal has not been reached. The function also checks the `contributions` mapping to see if the contributor has made a contribution. If the conditions are met, the function checks if the campaign has been withdrawn early. If the campaign has been withdrawn early and the contribution was made before the early withdrawal, the contributor will not be able to get a refund as the funds have already been withdrawn. Otherwise, funds are transferred back to the contributor's address, and the contribution amount is set to zero.

## 2.7 Getting Campaign Details

Campaign details can be retrieved by calling the `getCampaign` function, which returns the state of the campaign so far.

## 3 Testing On Remix Virtual Machine

We test the contract on Remix VM, since test faucets were not adequate to conduct testing on the Ethereum testnet.

### 3.1 Basic Functionalities

The basic functionalities of the contract, namely campaign creation, making contributions, and campaign details are tested below:

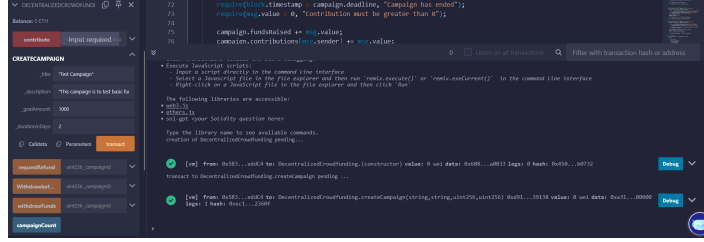


Figure 1: Campaign Creation



Figure 2: First transaction with yes vote

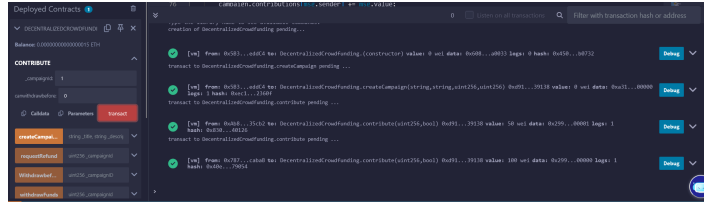


Figure 3: Second transaction with no vote

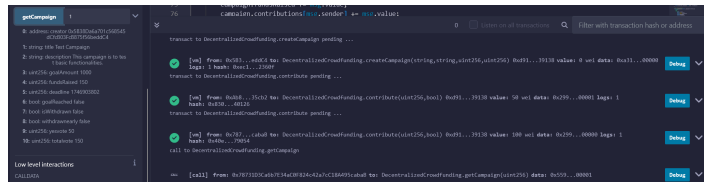


Figure 4: Campaign Details

## 3.2 Withdrawing Funds Before Goal

Since the campaign has not reached its deadline, the creator can call early withdrawal. Since it requires specific percent of vote, it will fail as following: Next, we will make



Figure 5: Early withdrawal failed

another transaction with a yes vote with sufficient weight, and then call the early withdrawal function again.

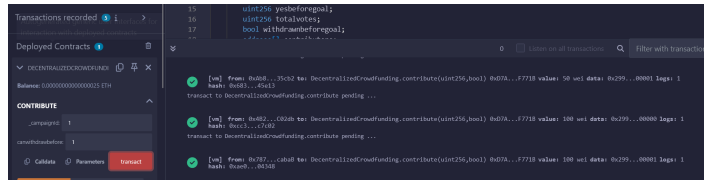


Figure 6: Third transaction with yes vote

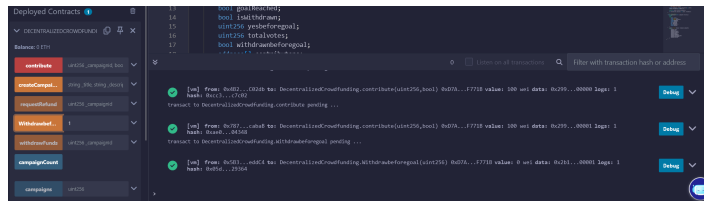


Figure 7: Successful early withdrawal

We can see the campaign details have been updated to show that the funds have been withdrawn, and the target amount has been reduced. Next, we try to see if we can call the early withdrawal function again.

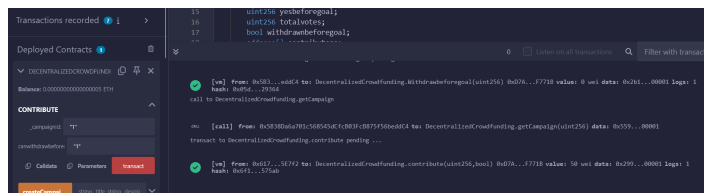


Figure 8: Fourth transaction with yes vote



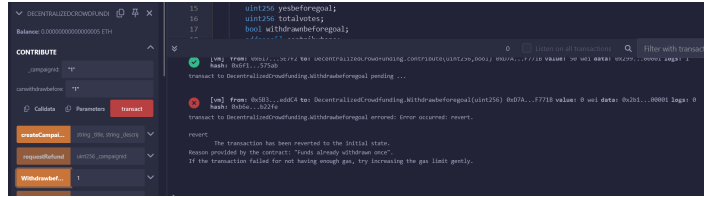


Figure 9: Second time early withdrawal failed

### 3.3 Refunding Contributors

Contributors can only get their funds back only after the campaign deadline has passed. Contributors do not get their funding back if the campaign goal is reached at any point of time.

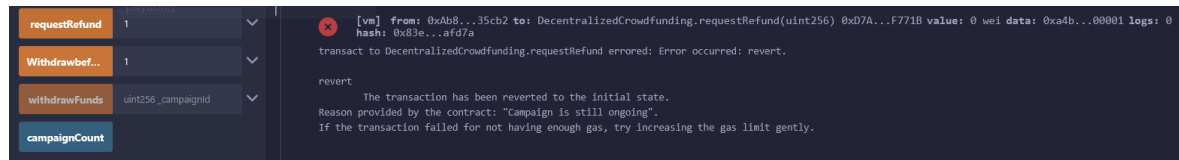


Figure 10: Campaign still active, refund failed

The process becomes a bit tricky if the campaign has been withdrawn early. In this case, only the contributors who made their contributions after the withdrawal can get their funds back, and can only be claimed after the deadline has passed and the goal has not been reached.



Figure 11: Refund failed because of early withdrawal

Figure 11 shows that the contributor can get their funds back even though the campaign has been withdrawn early, because the contribution was made after the early withdrawal.

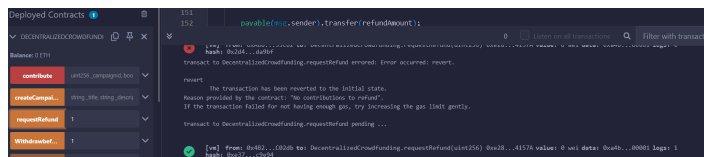


Figure 12: Successful refund after deadline and goal not reached

### 3.4 Withdrawing Funds After Deadline

Only the project creator can withdraw the funds after the deadline. In case the goal has not been reached, there is no way to withdraw the funds for the creator.

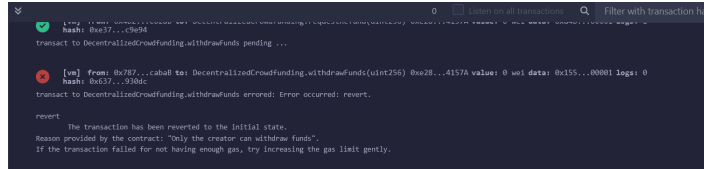


Figure 13: Only creators can withdraw funds after deadline

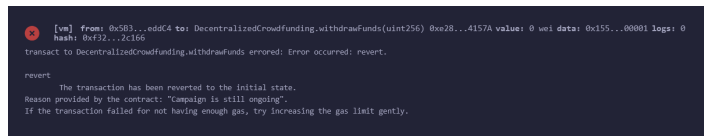


Figure 14: Cannot withdraw ongoing project

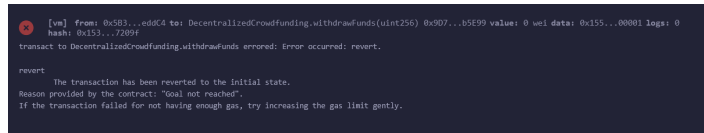


Figure 15: Cannot withdraw if goal has not been reached



Figure 16: Successful withdrawal with project condition

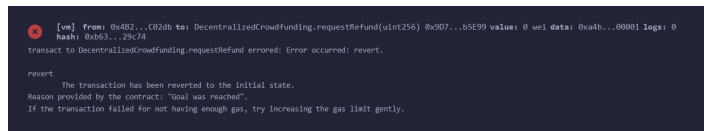


Figure 17: Unsuccessful refund request

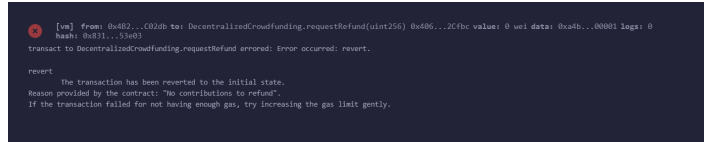


Figure 18: Cannot refund if no contribution made

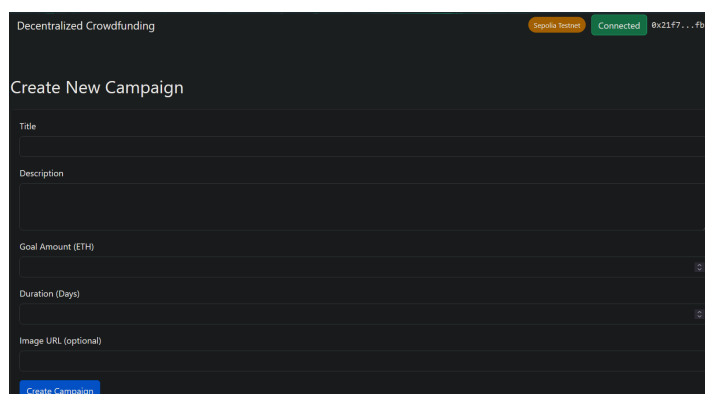
### 3.5 Other Security Checks

The contract is secured against re-entrancy attacks by ensuring states are updated before transferring funds. The contract also ensures that no token is stuck in the contract by validating who can donate and who can withdraw.

## 4 User Interface

We have built a simple user interface using html and javascript. The interface allows users to create campaigns, make contributions, withdraw funds (both before and after goal), and request refunds. Each campaign created is displayed on the interface, and potential contributors can see the details of the campaign, including goal amount, funds raised, time left for the campaign, vote percent for early withdrawal and so on.

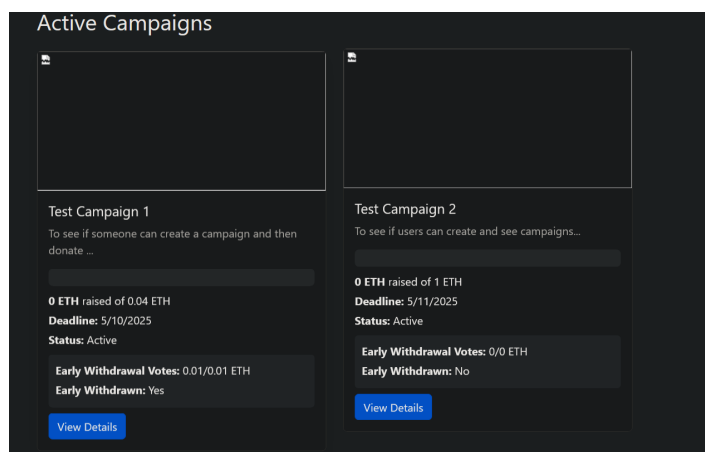
The interface requires the user to connect to their MetaMask wallet before using the functionalities. Once connected, users will see the following screen:



The screenshot shows a dark-themed web interface titled "Decentralized Crowdfunding". In the top right corner, there are two buttons: "Sign Up / Login" and "Connected", followed by a wallet address "0x21f7...fb84". The main heading is "Create New Campaign". Below this, there are several input fields: "Title", "Description", "Goal Amount (ETH)", "Duration (Days)", and "Image URL (optional)". Each field has a small icon to its right. At the bottom left, there is a blue button labeled "Create Campaign".

Figure 19: User Interface, campaign creation

Project creators can fill up the required fields and create their campaign. Once created, the campaign will be displayed on the interface as soon as the transaction is confirmed.



The screenshot shows a dark-themed web interface titled "Active Campaigns". It displays two campaign cards side-by-side. Each card has a placeholder image at the top. The first card is titled "Test Campaign 1" and contains the text "To see if someone can create a campaign and then donate ...". It shows "0 ETH raised of 0.04 ETH", a "Deadline: 5/10/2025", "Status: Active", "Early Withdrawal Votes: 0.01/0.01 ETH", and "Early Withdrawn: Yes". The second card is titled "Test Campaign 2" and contains the text "To see if users can create and see campaigns...". It shows "0 ETH raised of 1 ETH", a "Deadline: 5/11/2025", "Status: Active", "Early Withdrawal Votes: 0/0 ETH", and "Early Withdrawn: No". Both cards have a blue "View Details" button at the bottom.

Figure 20: Available Campaigns

To donate to a campaign, the user can click on the **View Details** button, which will show the details of the campaign and the option to donate. The user can enter their contribution amount and choose whether they want to vote in favour of early

withdrawal or not. Once the transaction is confirmed, updated campaign details will be displayed. In addition, before the deadline, the campaign create can withdraw the funds by clicking the **Withdraw Early** button.

The screenshot shows a dark-themed interface for 'Test Campaign 2'. On the left, there's a placeholder for a 'Campaign Image' and the text 'To see if users can create and see campaigns'. On the right, campaign details are listed: 'Creator: 0x21F7...f884', 'Goal: 1 ETH', 'Raised: 0 ETH', and 'Deadline: 5/11/2025, 4:17:48 PM'. Below this is a progress bar and the status 'Status: Active'. A section titled 'Early Withdrawal Voting' shows 'Yes Votes: 0 ETH', 'Total Votes: 0 ETH', and 'Early Withdrawal: No'. At the bottom, there's a 'Contribute to this campaign' section with a numeric input field set to '0.01', a checkbox labeled 'Vote to allow early withdrawal' which is checked, and a blue 'Donate' button.

Figure 21: Donating to a campaign

This screenshot shows the updated state of 'Test Campaign 2'. The progress bar is now partially filled with blue. The 'Raised' amount is updated to '0.01 ETH raised of 1 ETH'. The 'Status' remains 'Active'. The 'Early Withdrawal Voting' section now shows 'Early Withdrawal Votes: 0.01/0.01 ETH' and 'Early Withdrawn: No'. A blue 'View Details' button is located at the bottom.

Figure 22: Updated Campaign details

Once withdrawn early, the button will be disabled, and it will not be possible to call early withdrawal again.

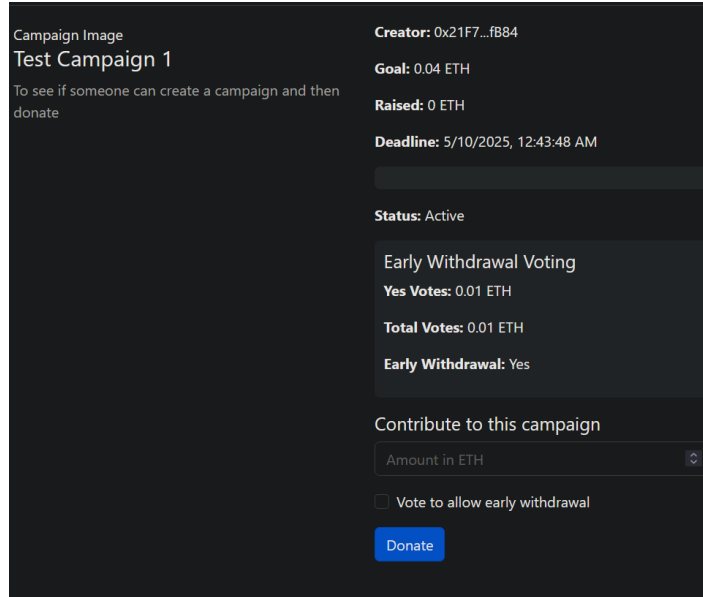


Figure 23: After early withdrawal

Once the campaign deadline has passed, there will be option for the creator to withdraw the funds, and for the contributors to request their funds back.

## 5 Conclusion

The Decentralized Crowdfunding with Early Withdrawal Voting smart contract represents a significant evolution in blockchain-based fundraising by introducing a flexible, trust-minimized system that balances the needs of both project creators and backers. With a fair voting mechanism, the contract empowers the community to make informed decisions about early fund withdrawals, ensuring that the interests of all parties are considered.

The early withdrawal system is designed to be transparent and secure, with clear rules and conditions for voting and fund withdrawal. This mechanism allows the project creator to access partial funds in the case of unmatched goals, while also penalizing greedy behavior. Since the early withdrawal call consumes more gas than the regular withdrawal, project creators are more incentivized to withdraw the funds only when necessary.

The user interface provides a seamless experience for both project creators and contributors, making it easy for anyone to call the contract functions. The integration with MetaMask and the Ethereum testnet allows users to interact with the contract without the need for complex setups or technical knowledge.

While this contract provides a robust foundation, potential improvements include reducing costs for refund processing in high-participation campaigns, enabling contributions in stablecoins or other ERC-20 tokens, and allowing backers to vote on fund releases at specific development stages.

Finally, This smart contract successfully addresses a critical limitation in traditional crowdfunding—rigidity in fund access—by introducing a decentralized, vote-based withdrawal system. By empowering backers with governance rights while protecting their refund eligibility, it fosters a more dynamic and equitable fundraising ecosystem. Future iterations could expand its use cases, but the current implementation already provides a secure, transparent, and flexible solution for blockchain-based crowdfunding.