FINAL EXAMINATION PROJECT

Course: Blockchain 1

Decentralized Charity Aid Platform (D-Help)

Stack: Solidity, JavaScript, MetaMask, Ethereum (Testnet)

https://github.com/jsowkw1/FINAL_BT.git

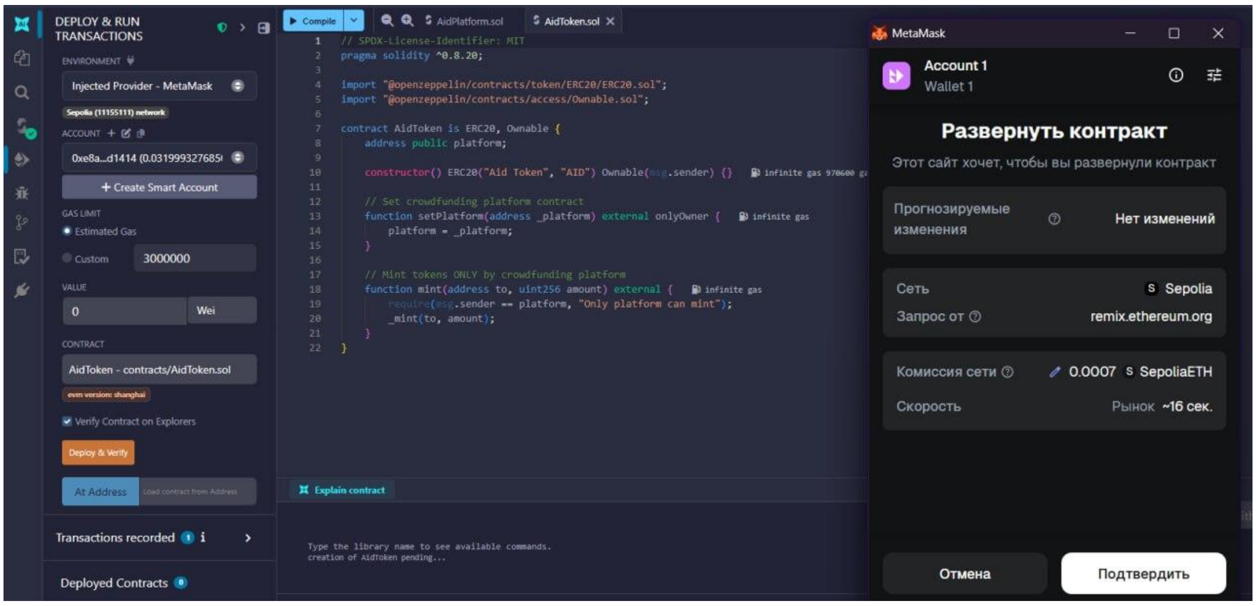
Group members:

Sagym Mukhammedzhan

Aliyev Miras

Shakizada Meiirzhan

Working with RemixIDE for deploy



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/access/Ownable.sol";

contract AidToken is ERC20, Ownable {
    address public platform;

    constructor() ERC20("Aid Token", "AID") Ownable(msg.sender) {}    infinite gas 970600 ga

    // Set crowdfunding platform contract
    function setPlatform(address _platform) external onlyOwner {    infinite gas
        platform = _platform;
    }

    // Mint tokens ONLY by crowdfunding platform
    function mint(address to, uint256 amount) external {    infinite gas
        require(msg.sender == platform, "Only platform can mint");
        _mint(to, amount);
    }
}
```

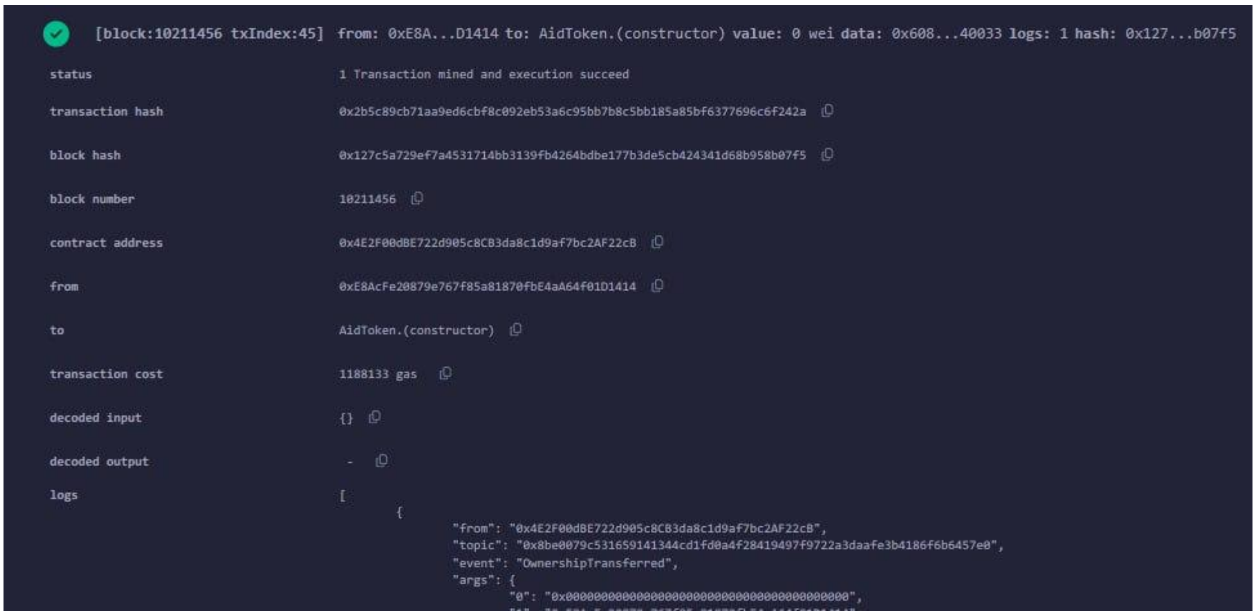
```
Type the library name to see available commands.
creation of AidToken pending...
```

Развернуть контракт

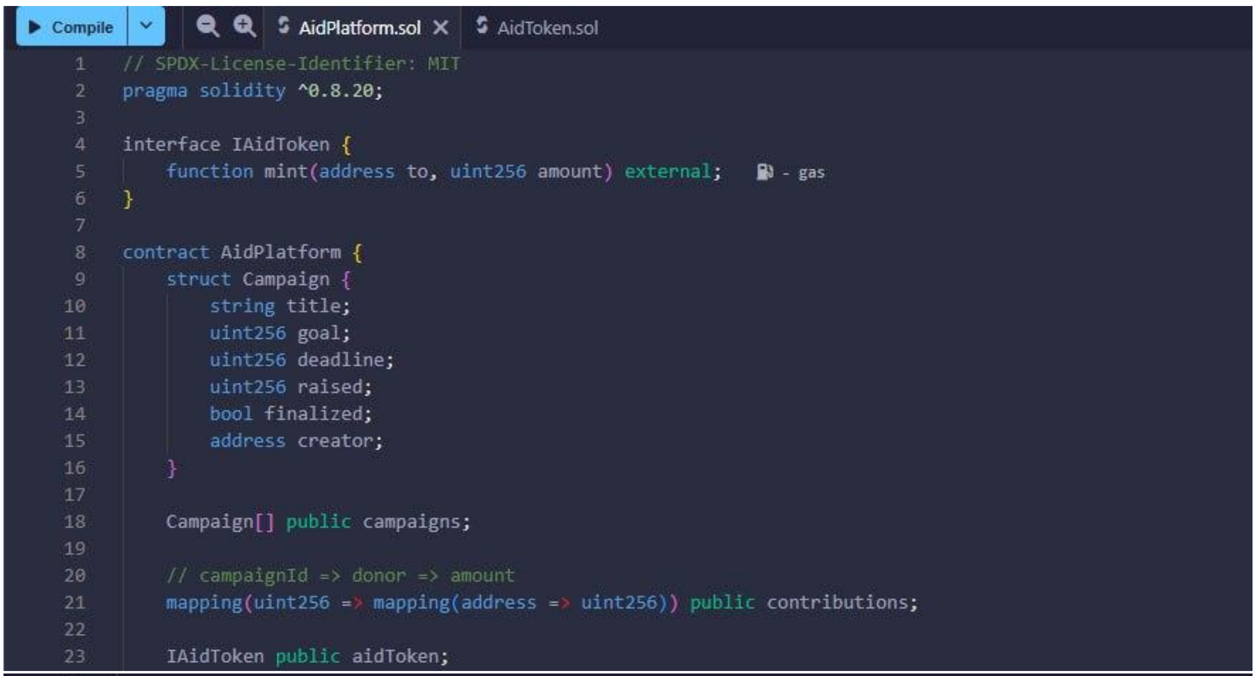
Этот сайт хочет, чтобы вы развернули контракт

Прогнозируемые изменения	Нет изменений
Сеть	Sepolia
Запрос от	remix.ethereum.org
Комиссия сети	0.0007 SepoliaETH
Скорость	Рынок ~16 сек.

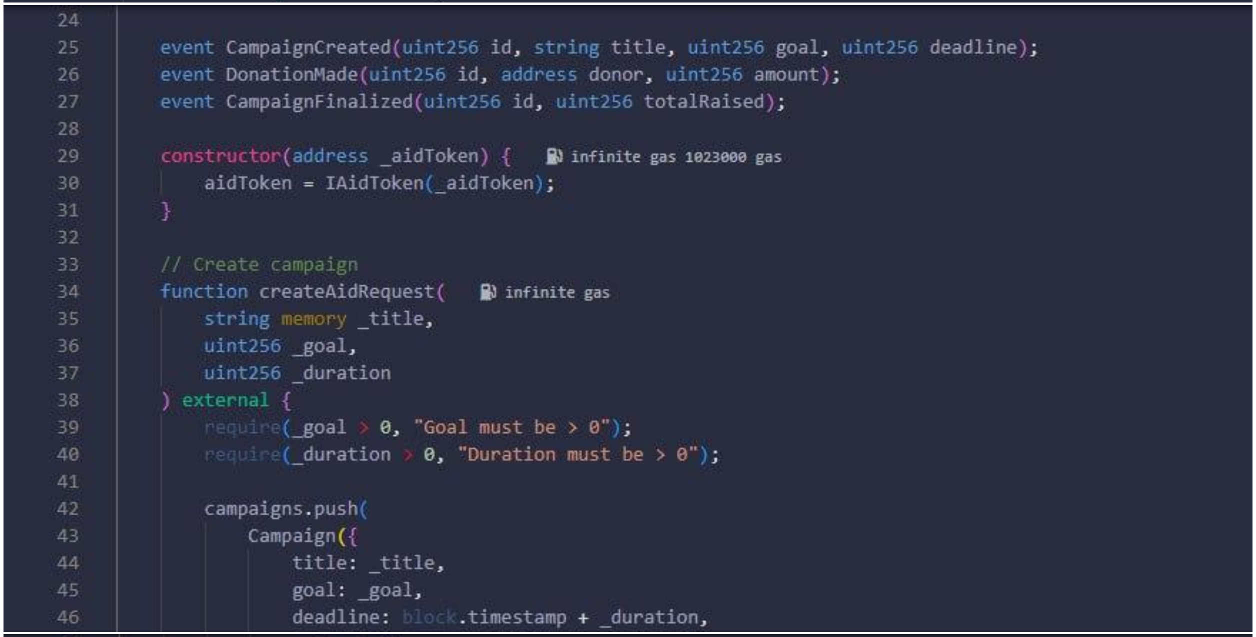
First, we deployed AidToken.sol



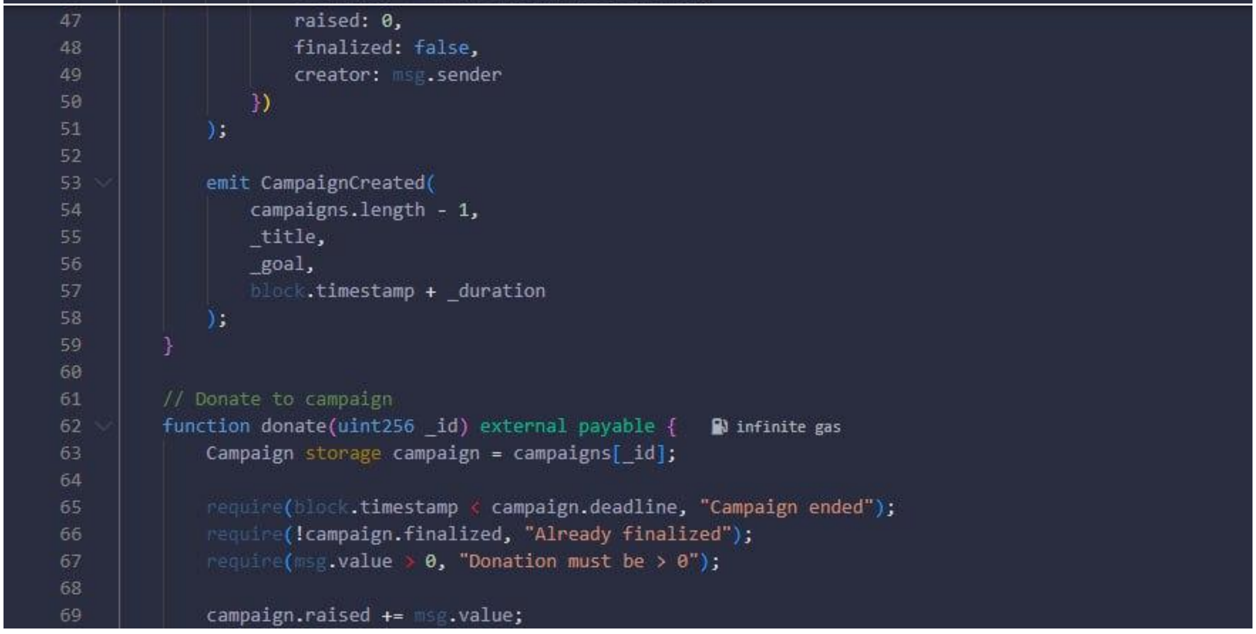
```
[block:10211456 txIndex:45] from: 0xE8A...D1414 to: AidToken.(constructor) value: 0 wei data: 0x608...40033 logs: 1 hash: 0x127...b07f5
status                1 Transaction mined and execution succeed
transaction hash      0x2b5c89cb71aa9ed6cbf8c092eb53a6c95bb7b8c5bb185a85bf6377696c6f242a
block hash            0x127c5a729ef7a4531714bb3139fb4264bdbe177b3de5cb424341d68b958b07f5
block number          10211456
contract address      0x4E2F00dBE722d905c8CB3da8c1d9af7bc2AF22cB
from                  0xE8AcFe20879e767f85a81870fbE4aA64f01D1414
to                    AidToken.(constructor)
transaction cost      1188133 gas
decoded input         {}
decoded output         -
logs                  [
                              {
                                      "from": "0x4E2F00dBE722d905c8CB3da8c1d9af7bc2AF22cB",
                                      "topic": "0x8be0079c531659141344cd1fd0a4f28419497f9722a3daafe3b4186f6b6457e0",
                                      "event": "OwnershipTransferred",
                                      "args": {
                                              "0": "0x0000000000000000000000000000000000000000",
```



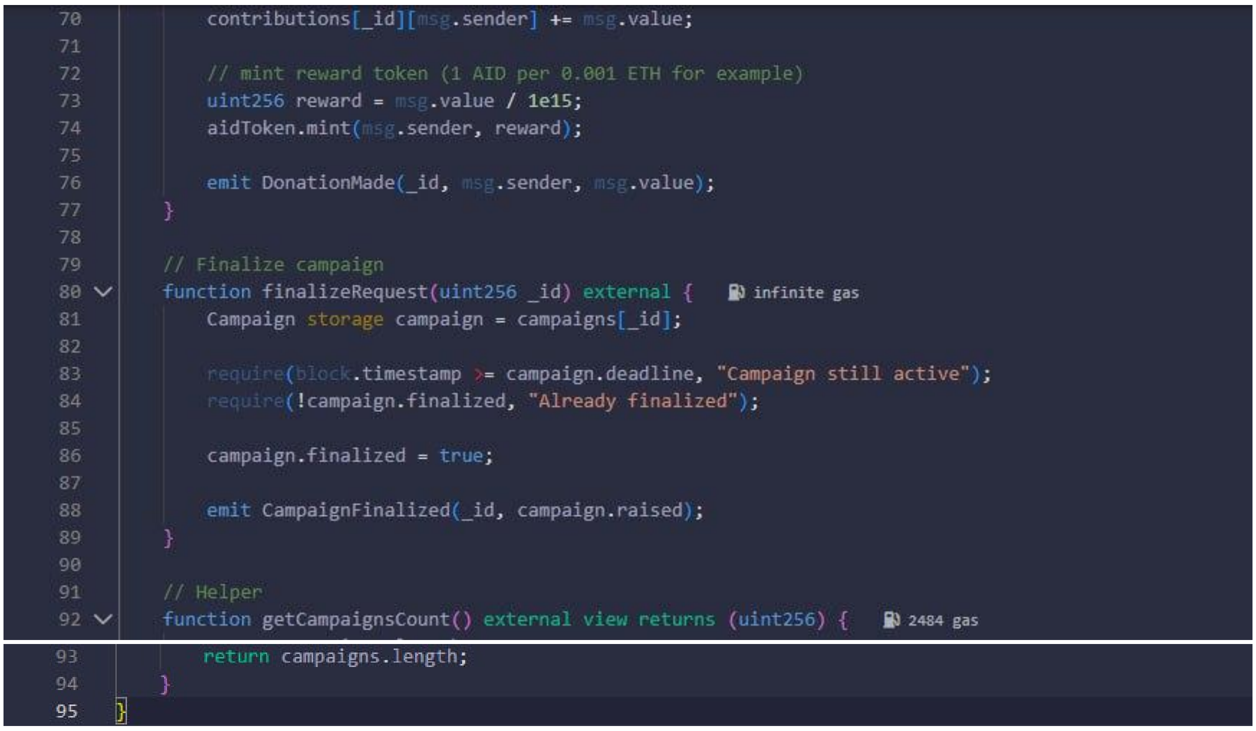
```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

interface IAidToken {
    function mint(address to, uint256 amount) external;
}

contract AidPlatform {
    struct Campaign {
        string title;
        uint256 goal;
        uint256 deadline;
        uint256 raised;
        bool finalized;
        address creator;
    }

    Campaign[] public campaigns;

    // campaignId => donor => amount
    mapping(uint256 => mapping(address => uint256)) public contributions;

    IAidToken public aidToken;
```



```solidity

    event CampaignCreated(uint256 id, string title, uint256 goal, uint256 deadline);
    event DonationMade(uint256 id, address donor, uint256 amount);
    event CampaignFinalized(uint256 id, uint256 totalRaised);

    constructor(address _aidToken) {
        aidToken = IAidToken(_aidToken);
    }

    // Create campaign
    function createAidRequest(
        string memory _title,
        uint256 _goal,
        uint256 _duration
    ) external {
        require(_goal > 0, "Goal must be > 0");
        require(_duration > 0, "Duration must be > 0");

        campaigns.push(
            Campaign({
                title: _title,
                goal: _goal,
                deadline: block.timestamp + _duration,
```

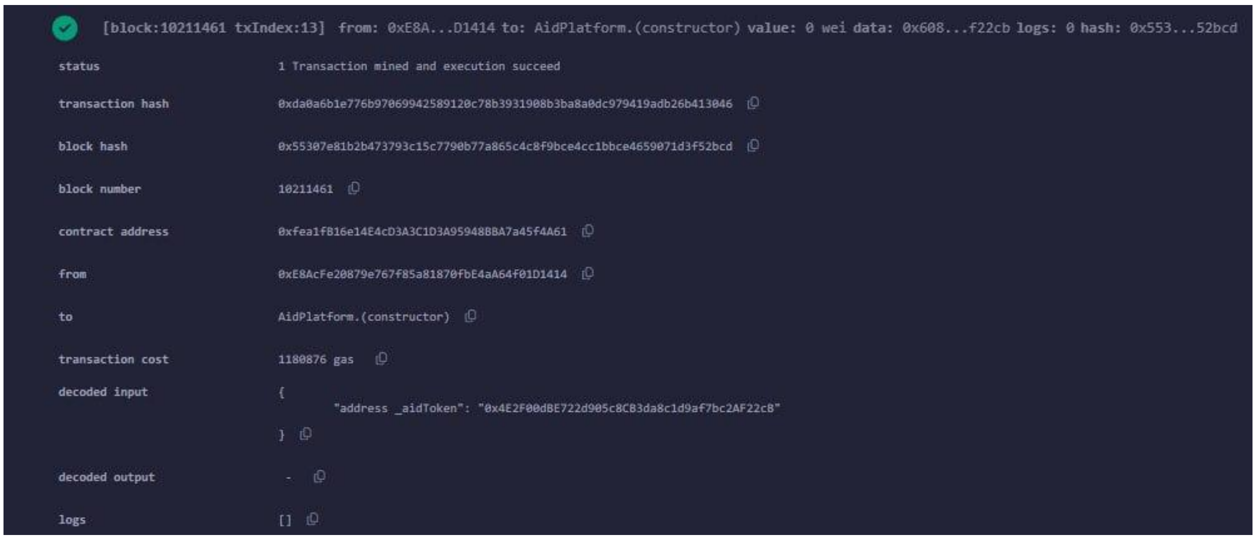


```solidity
                raised: 0,
                finalized: false,
                creator: msg.sender
            })
        );

        emit CampaignCreated(
            campaigns.length - 1,
            _title,
            _goal,
            block.timestamp + _duration
        );
    }

    // Donate to campaign
    function donate(uint256 _id) external payable {
        Campaign storage campaign = campaigns[_id];

        require(block.timestamp < campaign.deadline, "Campaign ended");
        require(!campaign.finalized, "Already finalized");
        require(msg.value > 0, "Donation must be > 0");

        campaign.raised += msg.value;
```



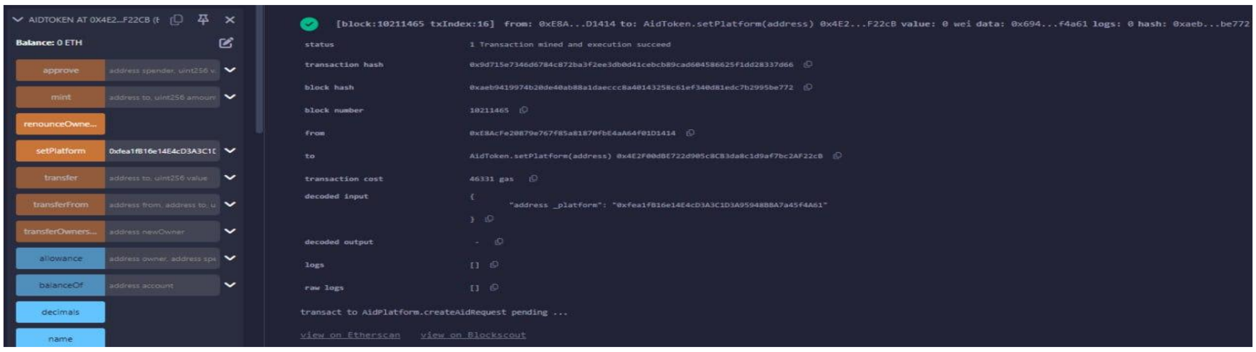
```
        contributions[_id][msg.sender] += msg.value;

        // mint reward token (1 AID per 0.001 ETH for example)
        uint256 reward = msg.value / 1e15;
        aidToken.mint(msg.sender, reward);

        emit DonationMade(_id, msg.sender, msg.value);
    }

    // Finalize campaign
    function finalizeRequest(uint256 _id) external {    infinite gas
        Campaign storage campaign = campaigns[_id];

        require(block.timestamp >= campaign.deadline, "Campaign still active");
        require(!campaign.finalized, "Already finalized");

        campaign.finalized = true;

        emit CampaignFinalized(_id, campaign.raised);
    }

    // Helper
    function getCampaignsCount() external view returns (uint256) {    2484 gas
        return campaigns.length;
    }
}
```

AidPlatform.sol



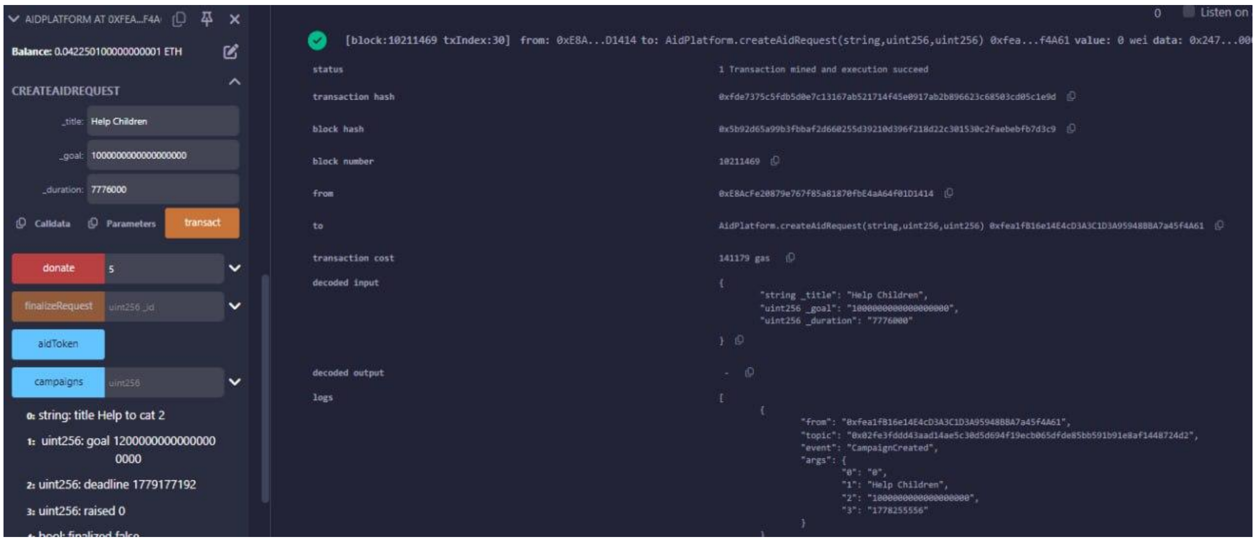
```
[block:10211461 txIndex:13]  from: 0xE8A...D1414 to: AidPlatform.(constructor) value: 0 wei data: 0x608...f22cb logs: 0 hash: 0x553...52bcd
status                 1 Transaction mined and execution succeed
transaction hash       0xda0a6b1e776b97069942589120c78b3931908b3ba8a0dc979419adb26b413046
block hash             0x55307e81b2b473793c15c7790b77a865c4c8f9bce4cc1bbce4659071d3f52bcd
block number           10211461
contract address       0xfea1f816e14E4cD3A3C1D3A95948BBA7a45f4A61
from                   0xE8AcFe20879e767f85a81870fbE4aA64f01D1414
to                     AidPlatform.(constructor)
transaction cost       1180876 gas
decoded input          {
                               "address _aidToken": "0x4E2F00dBE722d905c8CB3da8c1d9af7bc2AF22cB"
                       }
decoded output          -
logs                   []
```

and took his contract address



```
[block:10211465 txIndex:16]  from: 0xE8A...D1414 to: AidToken.setPlatform(address) 0x4E2...F22cB value: 0 wei data: 0x694...f4a61 logs: 0 hash: 0xaeb...be772
status                 1 Transaction mined and execution succeed
transaction hash       0x9d715e7346d6784c872ba3f2ee3db0d41cebcb89cad604586625f1dd28337d66
block hash             0xaeb9419974b20de40ab88a1daeccc8a40143258c61ef340d81edc7b2995be772
block number           10211465
from                   0xE8AcFe20879e767f85a81870fbE4aA64f01D1414
to                     AidToken.setPlatform(address) 0x4E2F00dBE722d905c8CB3da8c1d9af7bc2AF22cB
transaction cost       46331 gas
decoded input          {
                               "address _platform": "0xfea1f816e14E4cD3A3C1D3A95948BBA7a45f4A61"
                       }
decoded output          -
logs                   []
raw logs               []
transact to AidPlatform.createAidRequest pending ...
view on Etherscan   view on Blockscout
```

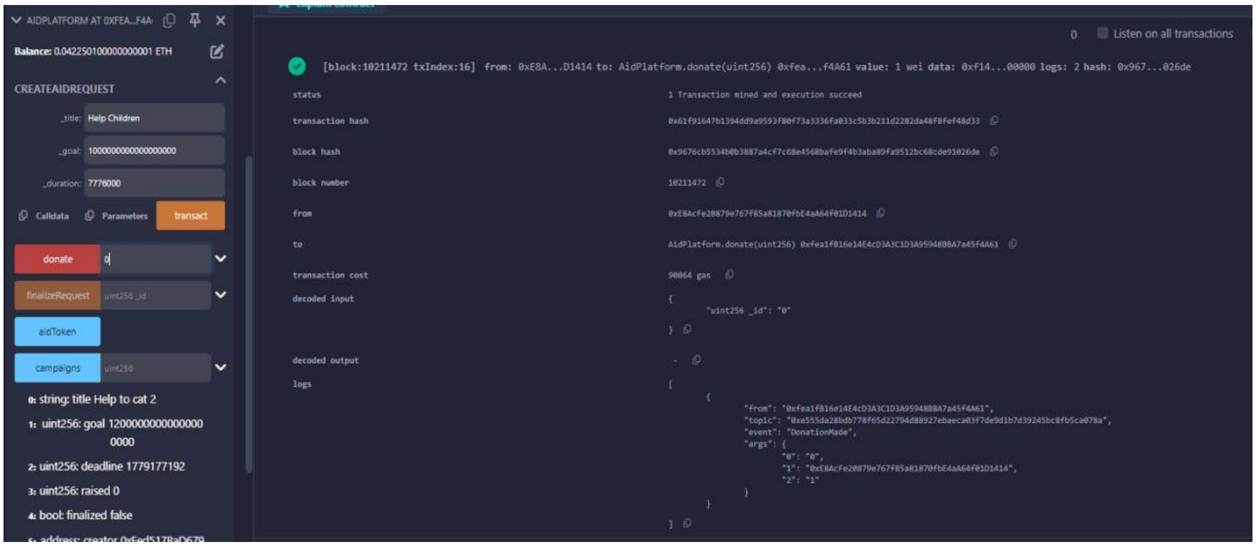
We took that contract address and put it into the setplatform function inside the AidToken.



```
AIDPLATFORM AT 0XFEA...F4A
Balance: 0.042250100000000001 ETH
CREATEAIDREQUEST
_title: Help Children
_goal: 1000000000000000000
_duration: 7776000
Calldata   Parameters   transact
donate  5
finalizeRequest  uint256 _id
aidToken
campaigns  uint256
0: string: title Help to cat 2
1: uint256: goal 12000000000000000000
2: uint256: deadline 1779177192
3: uint256: raised 0
```

```
[block:10211469 txIndex:30] from: 0xE8A...D1414 to: AidPlatform.createAidRequest(string,uint256,uint256) 0xfea...f4A61 value: 0 wei data: 0x247...00
status               1 Transaction mined and execution succeed
transaction hash     0xfde7375c5fdb5d0e7c13167ab521714f45e0917ab2b896623c68503cd05c1e9d
block hash           0x5b92d65a99b3fbbaf2d660255d39210d396f218d22c301530c2faebebfb7d3c9
block number         10211469
from                 0xE8AcFe20879e767f85a81870fbE4aA64f01D1414
to                   AidPlatform.createAidRequest(string,uint256,uint256) 0xfea1fB16e14E4cD3A3C1D3A95948BBA7a45f4A61
transaction cost     141179 gas
decoded input        {
                             "string _title": "Help Children",
                             "uint256 _goal": "1000000000000000000",
                             "uint256 _duration": "7776000"
                     }
decoded output       -
logs                 [
                             {
                                     "from": "0xfea1fB16e14E4cD3A3C1D3A95948BBA7a45f4A61",
                                     "topic": "0x02fe3fddd43aad14ae5c30d5d694f19ecb065dfde85bb591b91e8af1448724d2",
                                     "event": "CampaignCreated",
                                     "args": {
                                             "0": "0",
                                             "1": "Help Children",
                                             "2": "1000000000000000000",
                                             "3": "1778255556"
                                     }
```

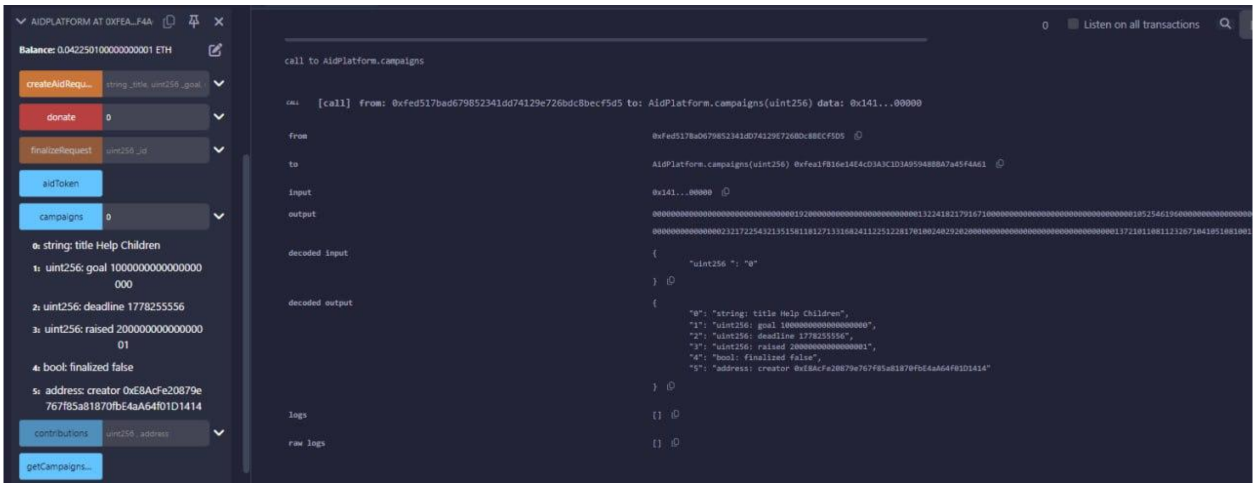
In aidplatform, we created a campaign or request with the name Help Children goal and duration, and it was assigned the index 0, as can be seen in the image.



```
AIDPLATFORM AT 0XFEA...F4A
Balance: 0.042250100000000001 ETH
CREATEAIDREQUEST
_title: Help Children
_goal: 1000000000000000000
_duration: 7776000
Calldata   Parameters   transact
donate  0
finalizeRequest  uint256 _id
aidToken
campaigns  uint256
0: string: title Help to cat 2
1: uint256: goal 12000000000000000000
2: uint256: deadline 1779177192
3: uint256: raised 0
4: bool: finalized false
```

```
[block:10211472 txIndex:16] from: 0xE8A...D1414 to: AidPlatform.donate(uint256) 0xfea...f4A61 value: 1 wei data: 0xf14...00000 logs: 2 hash: 0x967...026de
status               1 Transaction mined and execution succeed
transaction hash     0x61f91647b1394dd9a9593f80f73a3336fa033c5b3b211d2282da48f8fef48d33
block hash           0x9676cb5534b0b3887a4cf7c68e4568bafe9f4b3aba89fa9512bc68cde91026de
block number         10211472
from                 0xE8AcFe20879e767f85a81870fbE4aA64f01D1414
to                   AidPlatform.donate(uint256) 0xfea1fB16e14E4cD3A3C1D3A95948BBA7a45f4A61
transaction cost     90064 gas
decoded input        {
                             "uint256 _id": "0"
                     }
decoded output       -
logs                 [
                             {
                                     "from": "0xfea1fB16e14E4cD3A3C1D3A95948BBA7a45f4A61",
                                     "topic": "0xe555da28bdb778f65d22794d88927ebaeca03f7de9d1b7d39245bc8fb5ca078a",
                                     "event": "DonationMade",
                                     "args": {
                                             "0": "0",
                                             "1": "0xE8AcFe20879e767f85a81870fbE4aA64f01D1414",
                                             "2": "1"
                                     }
                             }
                     ]
```

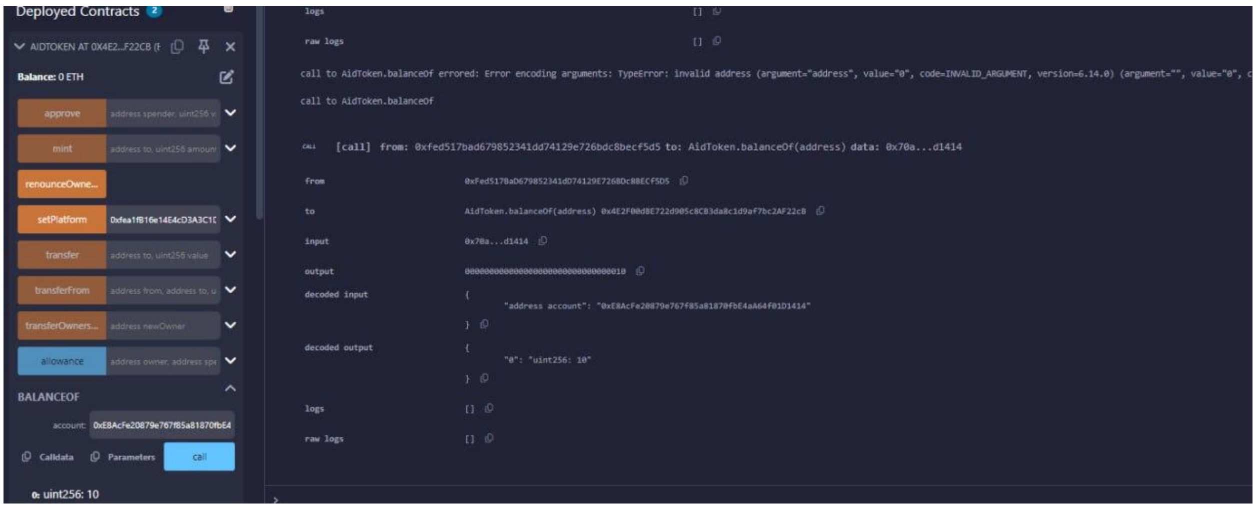
In the Aidplatform contract, after selecting the donation function, we wrote the ID of the newly created campaign, which is 0, and after writing how much we want to donate, we sent the money.



```
AIDPLATFORM AT 0XFEA...F4A
Balance: 0.042250100000000001 ETH
createAidRequ...  string _title, uint256 _goal,
donate  0
finalizeRequest  uint256 _id
aidToken
campaigns  0
0: string: title Help Children
1: uint256: goal 1000000000000000000
2: uint256: deadline 1778255556
3: uint256: raised 2000000000000000001
4: bool: finalized false
5: address: creator 0xE8AcFe20879e767f85a81870fbE4aA64f01D1414
contributions  uint256 , address
getCampaigns...
```

```
call to AidPlatform.campaigns

CALL  [call] from: 0xfed517bad679852341dd74129e726bdc8becf5d5 to: AidPlatform.campaigns(uint256) data: 0x141...00000
from                 0xFed517BaD679852341dD74129E726bDc8BECf5D5
to                   AidPlatform.campaigns(uint256) 0xfea1fB16e14E4cD3A3C1D3A95948BBA7a45f4A61
input                0x141...00000
output               0000000000000000000000000000000000192000000000000000000000000001322418217916710000000000000000000000000000000000010525461960000000000000000
                     0000000000000000232172254321351581181271331682411225122817010024029202000000000000000000000000000000000137210110811232671041051081001
decoded input        {
                             "uint256 ": "0"
                     }
decoded output       {
                             "0": "string: title Help Children",
                             "1": "uint256: goal 1000000000000000000",
                             "2": "uint256: deadline 1778255556",
                             "3": "uint256: raised 2000000000000000001",
                             "4": "bool: finalized false",
                             "5": "address: creator 0xE8AcFe20879e767f85a81870fbE4aA64f01D1414"
                     }
logs                 []
raw logs             []
```

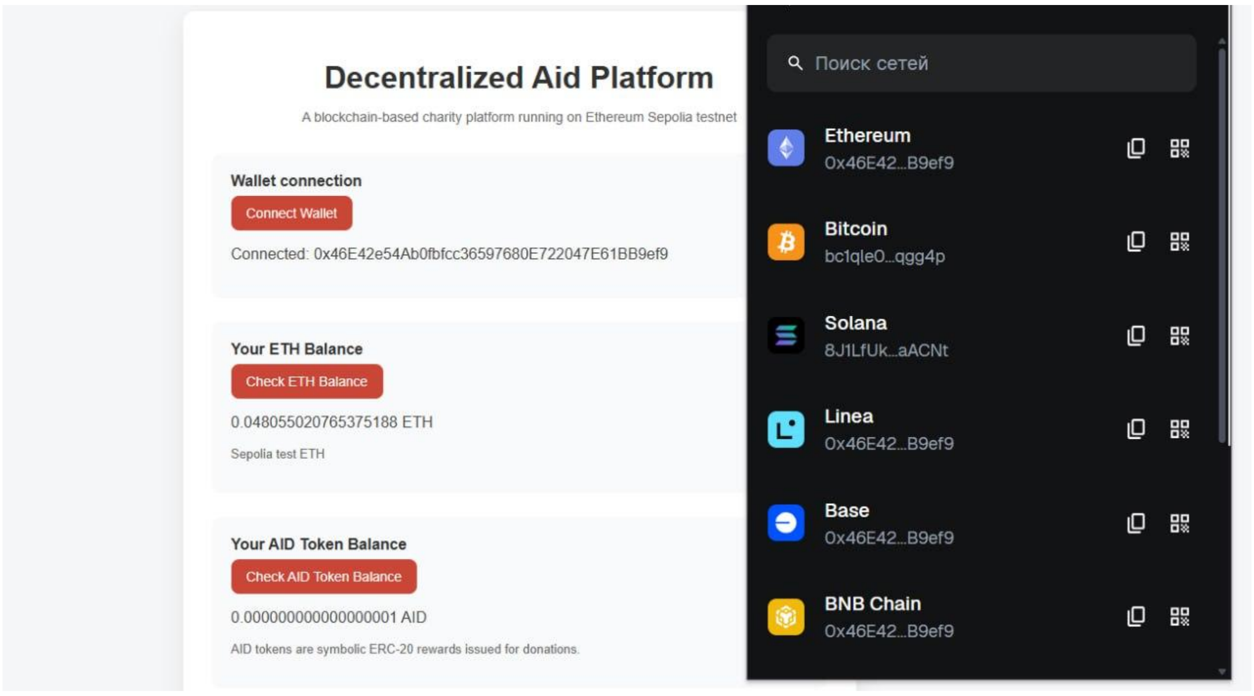
In the aidplatform contract, we selected the campaigns function, entered ID 0, and it displayed our campaign name, goal, raised funds, and deadline.



```
logs                                                        []
raw logs                                                    []

call to AidToken.balanceOf errored: Error encoding arguments: TypeError: invalid address (argument="address", value="0", code=INVALID_ARGUMENT, version=6.14.0) (argument="", value="0", c

call to AidToken.balanceOf

CALL   [call]  from: 0xfed517bad679852341dd74129e726bdc8becf5d5 to: AidToken.balanceOf(address) data: 0x70a...d1414

from                    0xFed517BaD679852341dD74129E726BDc8BECf5D5
to                      AidToken.balanceOf(address) 0x4E2F00d8E722d905c8C83da8c1d9af7bc2AF22cB
input                   0x70a...d1414
output                  0000000000000000000000000000000000010
decoded input           {
                                "address account": "0xE8AcFe20879e767f85a81870fbE4aA64f01D1414"
                        }
decoded output          {
                                "0": "uint256: 10"
                        }
logs                    []
raw logs                []
```

The donor's balance, or rather the reward, is 10 AID

Frontend(donate system)



Decentralized Aid Platform

A blockchain-based charity platform running on Ethereum Sepolia testnet

Wallet connection

Connect Wallet

Connected: 0x46E42e54Ab0fbfcc36597680E722047E61BB9ef9

Your ETH Balance

Check ETH Balance

0.048055020765375188 ETH

Sepolia test ETH

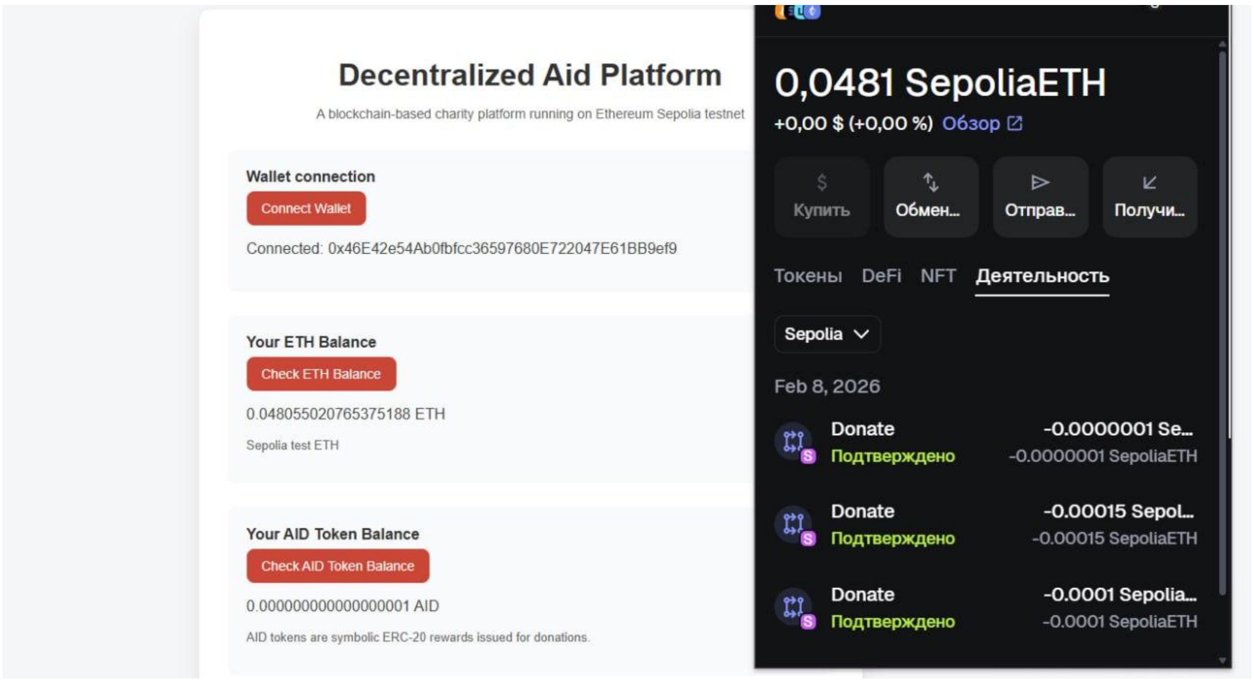
Your AID Token Balance

Check AID Token Balance

0.000000000000000001 AID

AID tokens are symbolic ERC-20 rewards issued for donations.

Ethereum	0x46E42...B9ef9
Bitcoin	bc1qle0...qgg4p
Solana	8J1LfUk...aACNt
Linea	0x46E42...B9ef9
Base	0x46E42...B9ef9
BNB Chain	0x46E42...B9ef9



Decentralized Aid Platform

A blockchain-based charity platform running on Ethereum Sepolia testnet

Wallet connection

Connect Wallet

Connected: 0x46E42e54Ab0fbfcc36597680E722047E61BB9ef9

Your ETH Balance

Check ETH Balance

0.048055020765375188 ETH

Sepolia test ETH

Your AID Token Balance

Check AID Token Balance

0.000000000000000001 AID

AID tokens are symbolic ERC-20 rewards issued for donations.

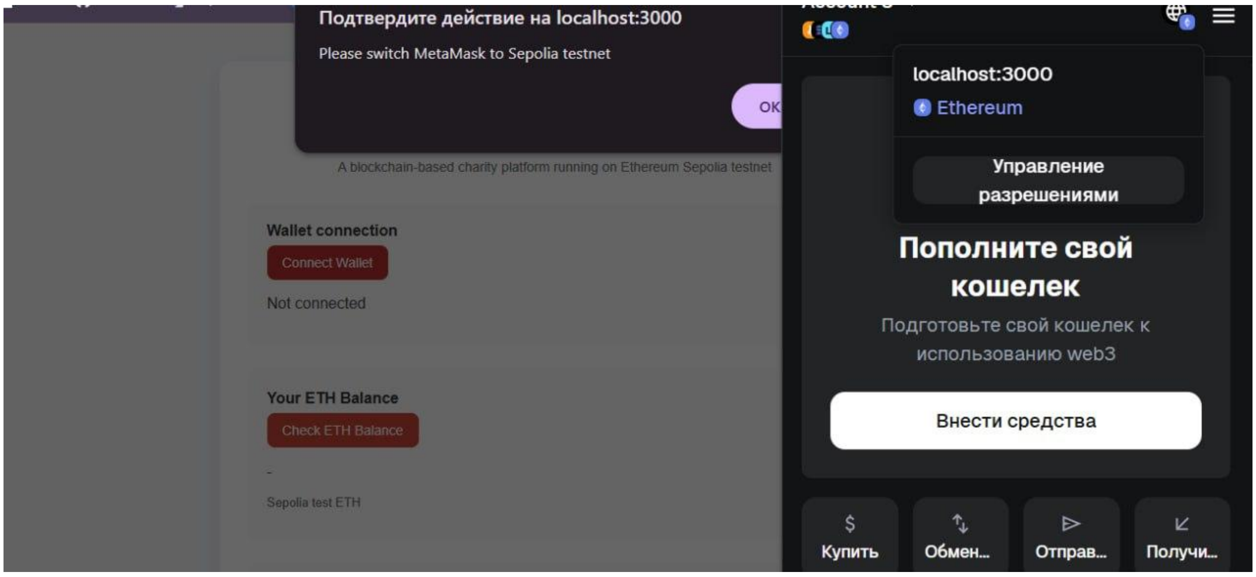
0,0481 SepoliaETH

+0,00 $ (+0,00 %) Обзор

Feb 8, 2026

Donate Подтверждено	-0.0000001 Se... -0.0000001 SepoliaETH
Donate Подтверждено	-0.00015 Sepol... -0.00015 SepoliaETH
Donate Подтверждено	-0.0001 Sepolia... -0.0001 SepoliaETH

Metamask has connected to our site, we see the same address as our wallet balance, and below are the AID tokens that are given as a reward.


Подтвердите действие на localhost:3000

Please switch MetaMask to Sepolia testnet

A blockchain-based charity platform running on Ethereum Sepolia testnet

Wallet connection

Not connected

Your ETH Balance

Sepolia test ETH

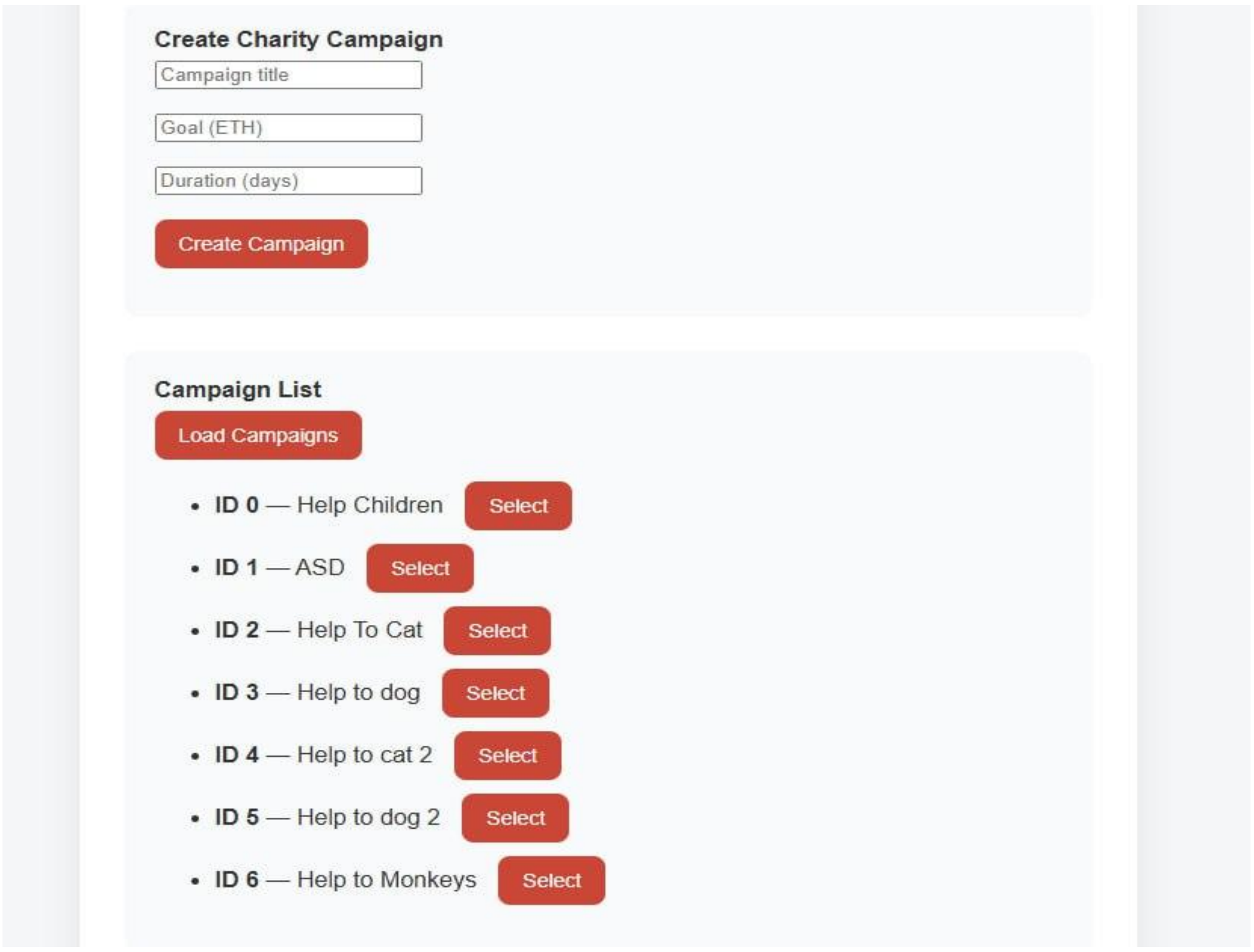
localhost:3000

Ethereum

Пополните свой кошелек

Подготовьте свой кошелек к использованию web3

Here we have a wallet check on Sepolia. If it is another wallet, in the Ethereum example, it will display an alert.


Create Charity Campaign

Campaign title

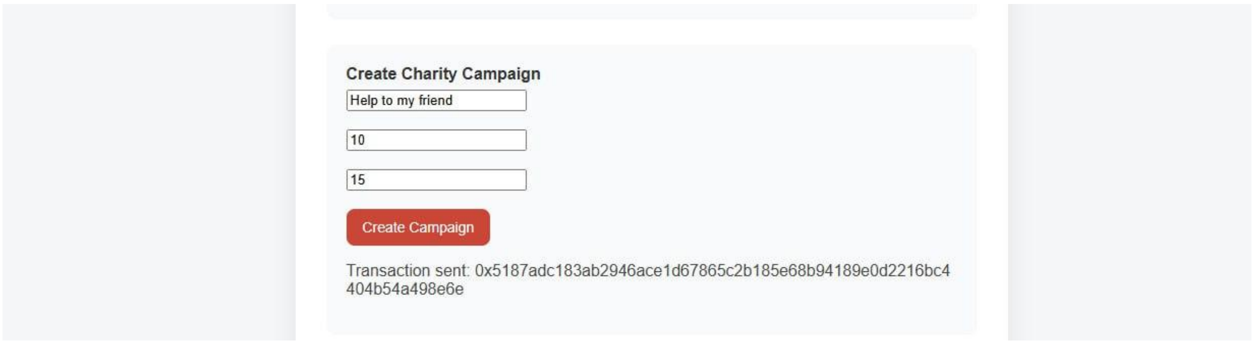
Goal (ETH)

Duration (days)

Campaign List

- **ID 0** — Help Children
- **ID 1** — ASD
- **ID 2** — Help To Cat
- **ID 3** — Help to dog
- **ID 4** — Help to cat 2
- **ID 5** — Help to dog 2
- **ID 6** — Help to Monkeys

Here we can create a campaign and below is a list of these campaigns sorted by ID.


Create Charity Campaign

Help to my friend

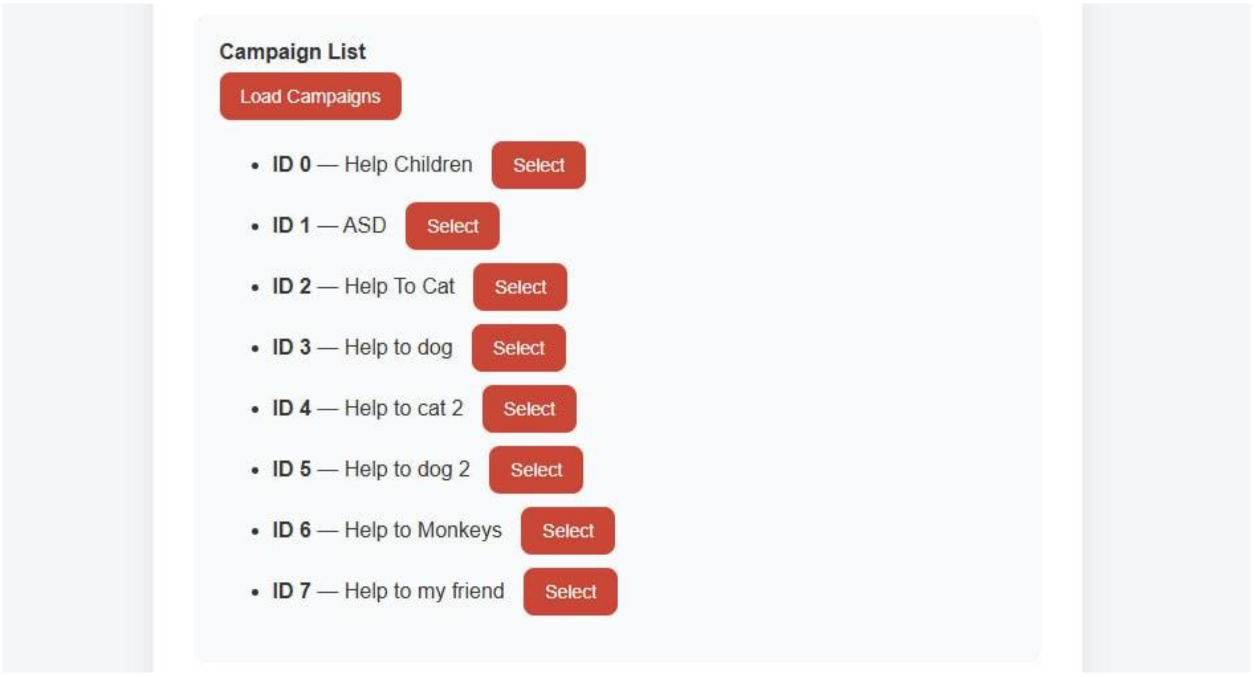
10

15

Create Campaign

Transaction sent: 0x5187adc183ab2946ace1d67865c2b185e68b94189e0d2216bc4404b54a498e6e

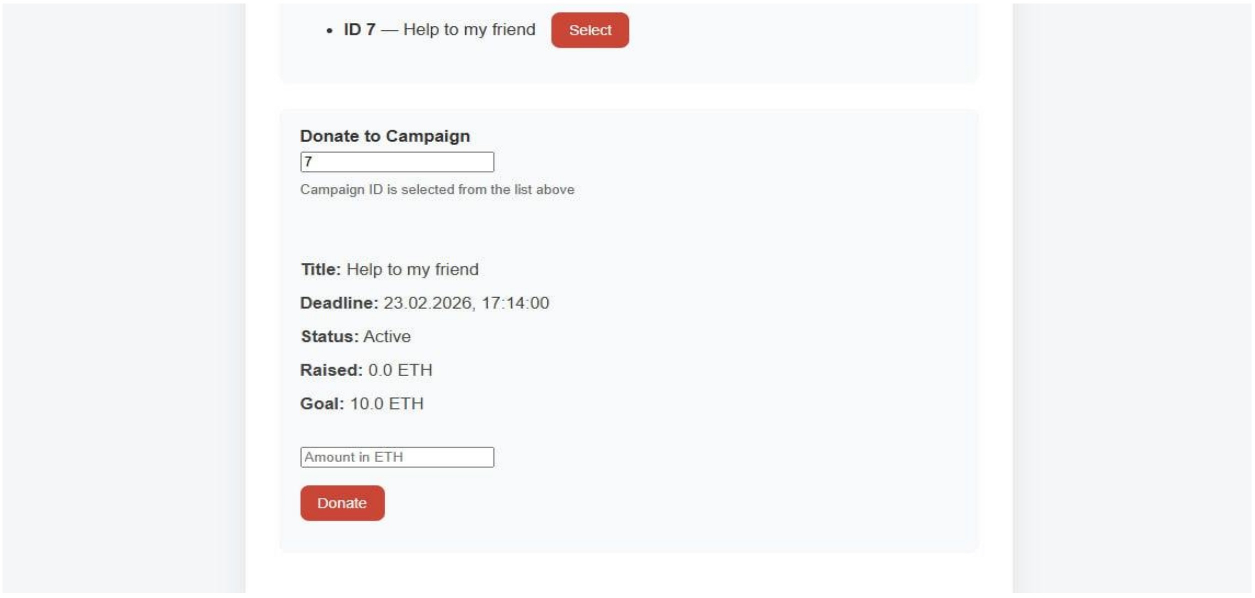
We created a new company and it gave us its ID.


Campaign List

Load Campaigns

- **ID 0** — Help Children Select
- **ID 1** — ASD Select
- **ID 2** — Help To Cat Select
- **ID 3** — Help to dog Select
- **ID 4** — Help to cat 2 Select
- **ID 5** — Help to dog 2 Select
- **ID 6** — Help to Monkeys Select
- **ID 7** — Help to my friend Select

The company we just added appeared on the list.


- **ID 7** — Help to my friend Select

Donate to Campaign

7

Campaign ID is selected from the list above

Title: Help to my friend

Deadline: 23.02.2026, 17:14:00

Status: Active

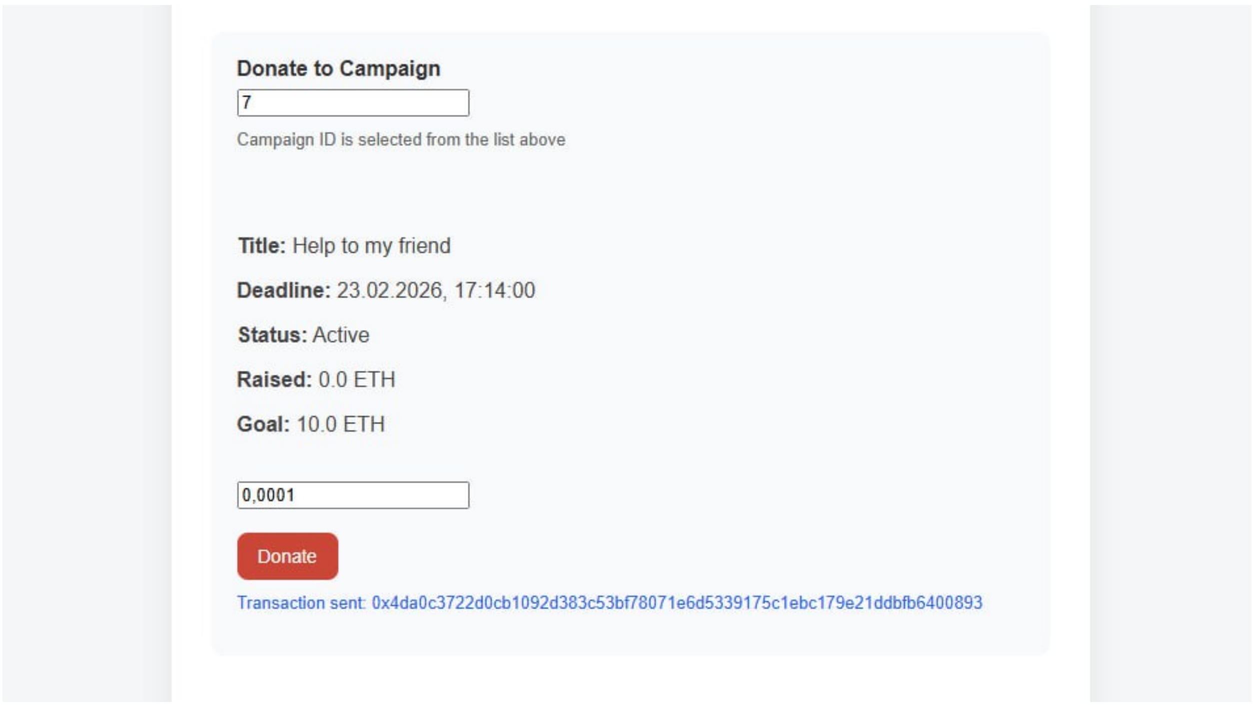
Raised: 0.0 ETH

Goal: 10.0 ETH

Amount in ETH

Donate

After clicking on the selection button, in our case by ID 7, it will display below the name of the deadline of this company, the status, how much needs to be collected, in our case 10 ETH, and how much has been collected since we opened this campaign, the money just collected is equal to 0



Donate to Campaign

7

Campaign ID is selected from the list above

Title: Help to my friend

Deadline: 23.02.2026, 17:14:00

Status: Active

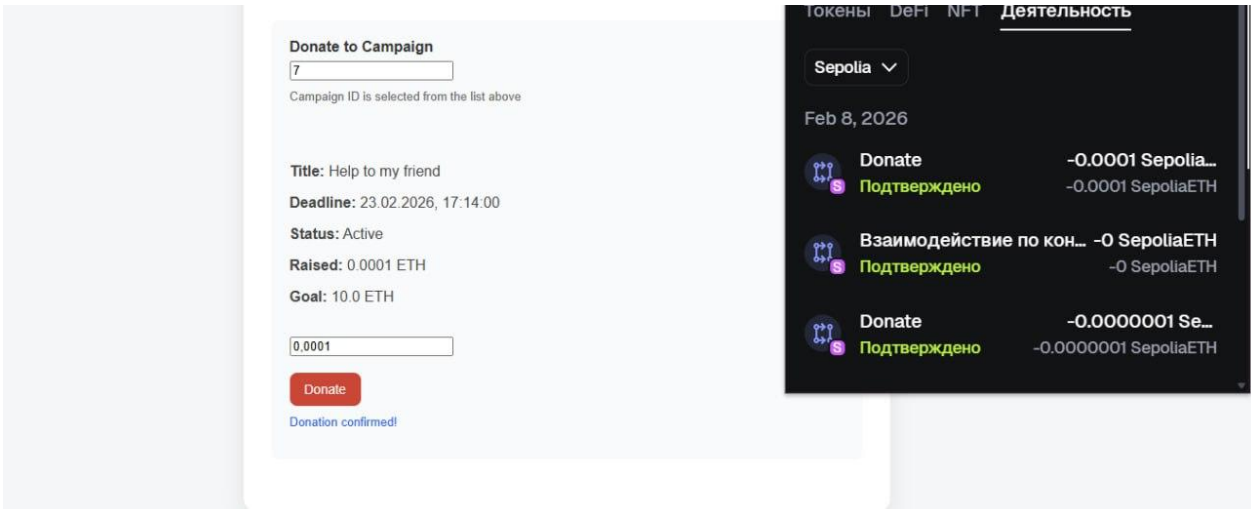
Raised: 0.0 ETH

Goal: 10.0 ETH

0,0001

Transaction sent: 0x4da0c3722d0cb1092d383c53bf78071e6d5339175c1ebc179e21ddbfb6400893

Donate 0.0001 ETH to this campaign. Transaction is being processed.



Donate to Campaign

7

Campaign ID is selected from the list above

Title: Help to my friend

Deadline: 23.02.2026, 17:14:00

Status: Active

Raised: 0.0001 ETH

Goal: 10.0 ETH

0,0001

Donation confirmed!

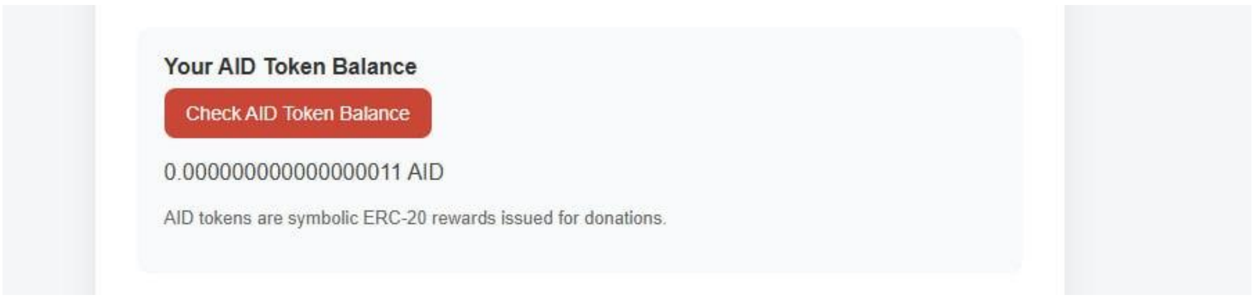
Feb 8, 2026

Donate -0.0001 Sepolia...
Подтверждено -0.0001 SepoliaETH

Взаимодействие по кон... -0 SepoliaETH
Подтверждено -0 SepoliaETH

Donate -0.0000001 Se...
Подтверждено -0.0000001 SepoliaETH

After processing, the balance was automatically updated to 0.001 ETH and a "Donation confirmed" message appeared! We can also see in MetaMask that the mint has been confirmed.

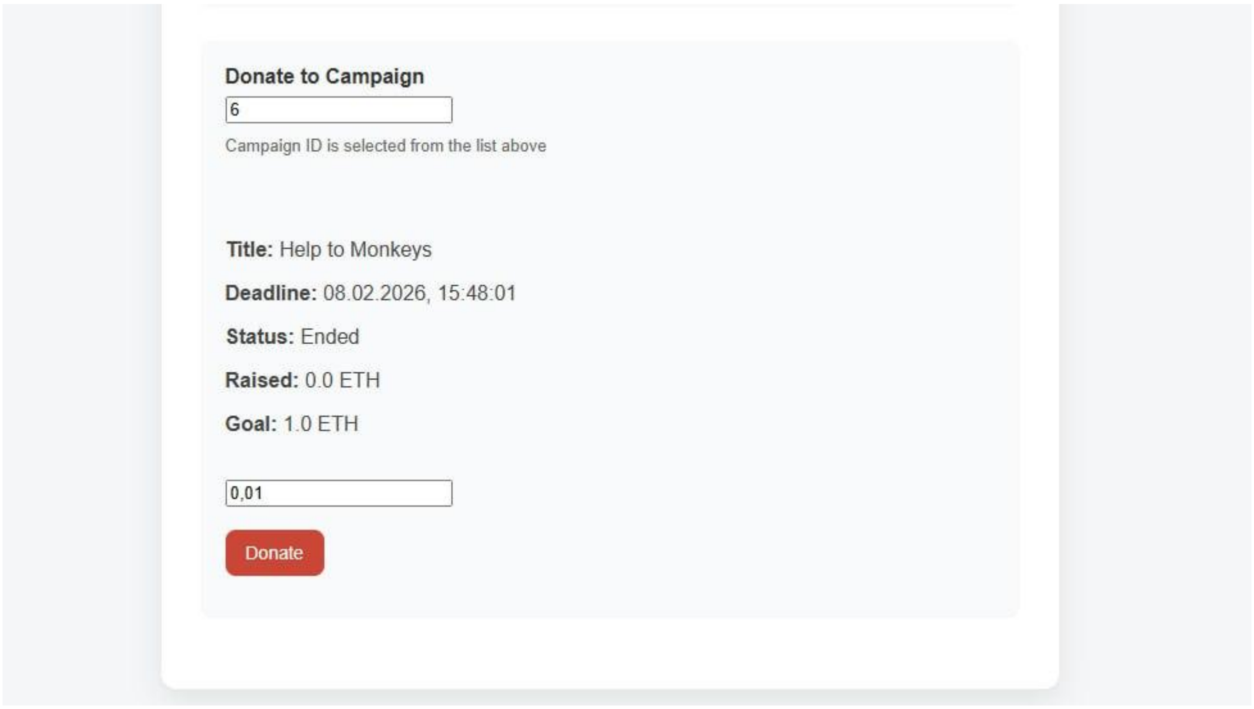


Your AID Token Balance

0.000000000000000011 AID

AID tokens are symbolic ERC-20 rewards issued for donations.

The AID token balance has also been updated.


Donate to Campaign

6

Campaign ID is selected from the list above

Title: Help to Monkeys

Deadline: 08.02.2026, 15:48:01

Status: Ended

Raised: 0.0 ETH

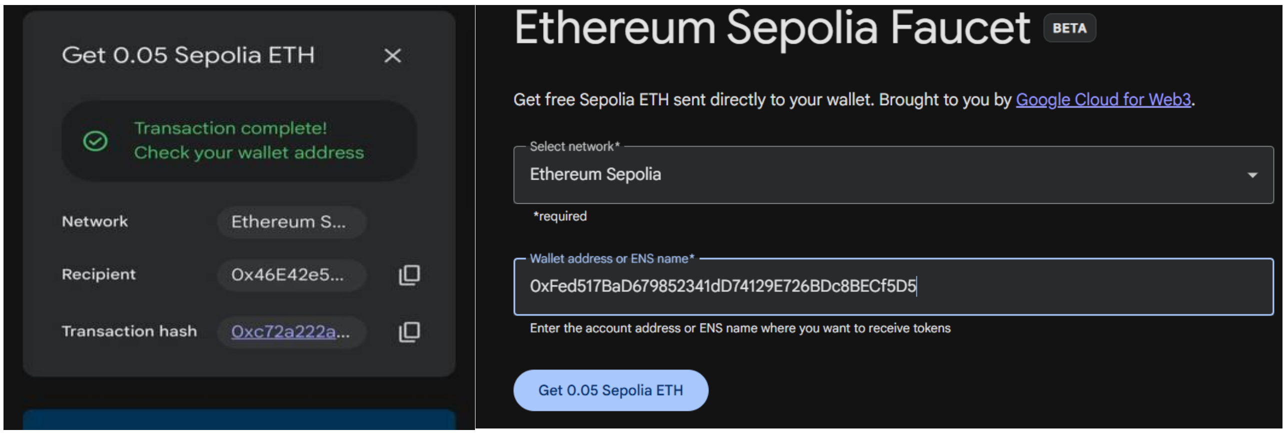
Goal: 1.0 ETH

0,01

We can't donate to a company with a completed deadline. After the campaign deadline is reached, the campaign can be finalized. If the collected amount is greater than or equal to the funding goal, the campaign is considered successful. In this case, the campaign creator is allowed to withdraw the collected funds. If the collected amount is less than the goal, the campaign is considered failed. In this case, contributors are able to refund their donations.

Network

The application operates exclusively on the Ethereum Sepolia test network. Deployment on Ethereum mainnet and usage of real cryptocurrency is strictly prohibited.


Get 0.05 Sepolia ETH

Transaction complete!
Check your wallet address

Network	Ethereum S...
Recipient	0x46E42e5...
Transaction hash	0xc72a222a...

Ethereum Sepolia Faucet BETA

Get free Sepolia ETH sent directly to your wallet. Brought to you by Google Cloud for Web3.

Select network*: Ethereum Sepolia

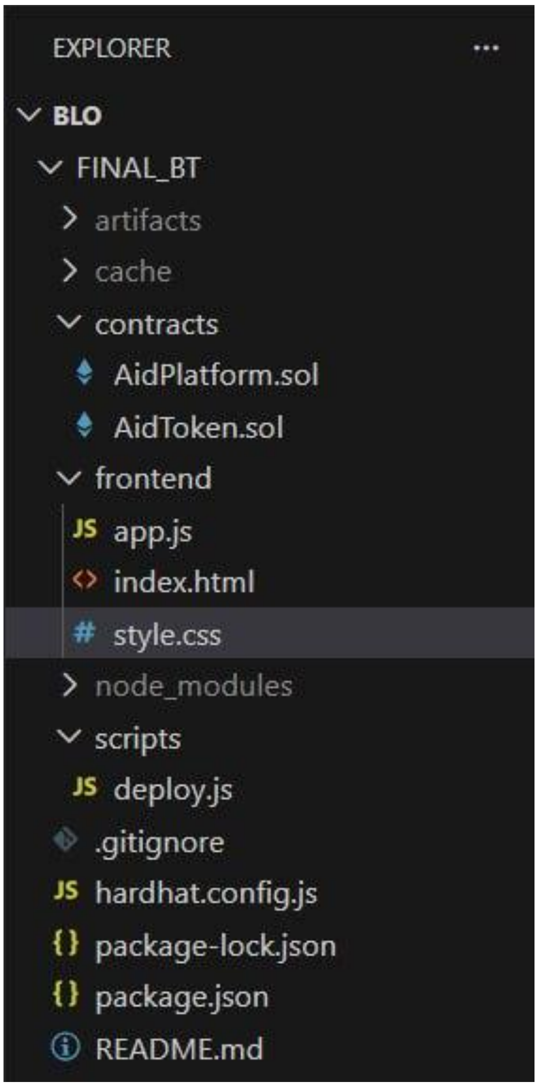
*required

Wallet address or ENS name*: 0xFed517BaD679852341dD74129E726BDc8BECf5D5

Enter the account address or ENS name where you want to receive tokens

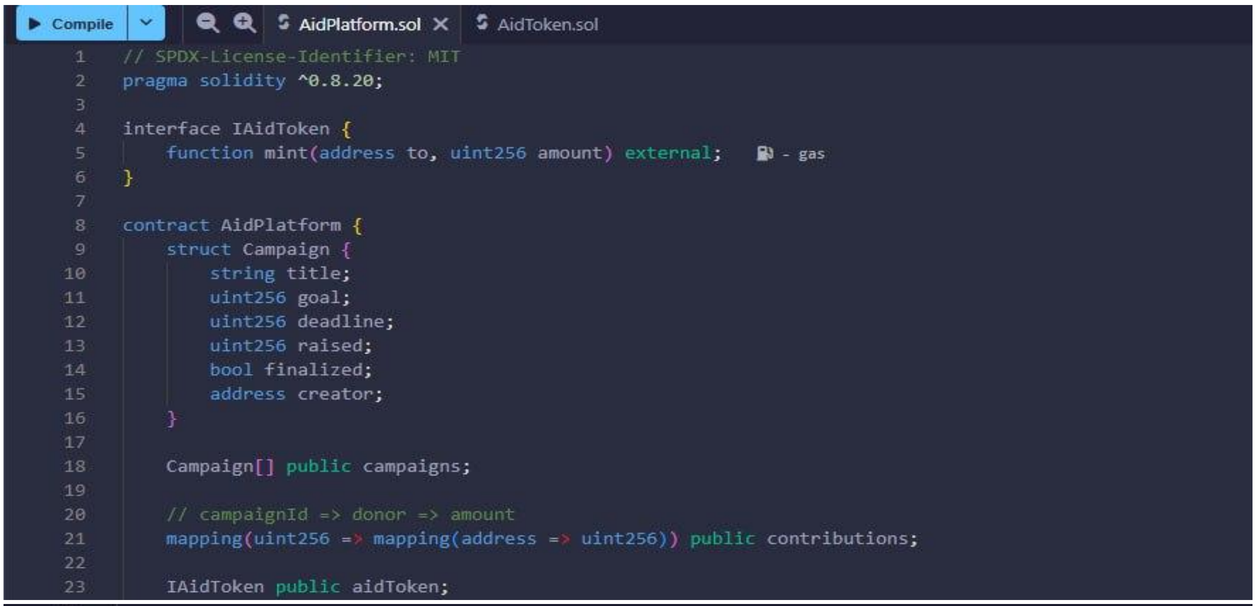
To obtain test ETH, we used the official Sepolia testnet faucet.

File Structure

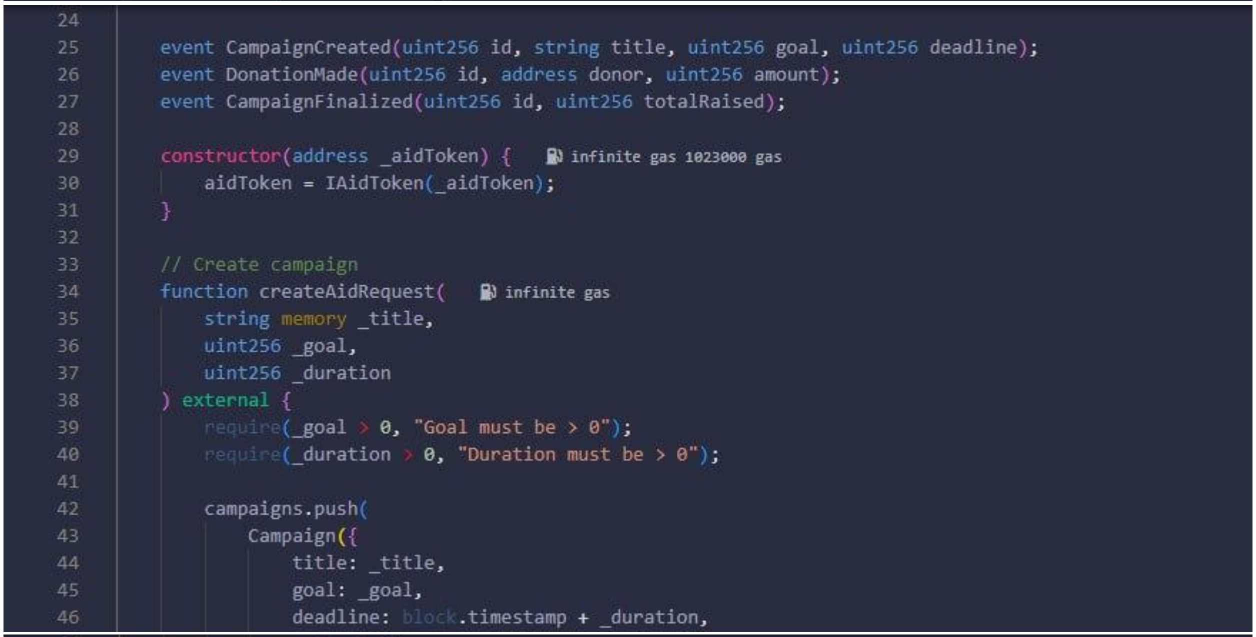


```
EXPLORER ...
∨ BLO
  ∨ FINAL_BT
    > artifacts
    > cache
    ∨ contracts
        AidPlatform.sol
        AidToken.sol
    ∨ frontend
        app.js
        index.html
        style.css
    > node_modules
    ∨ scripts
        deploy.js
    .gitignore
    hardhat.config.js
    package-lock.json
    package.json
    README.md
```

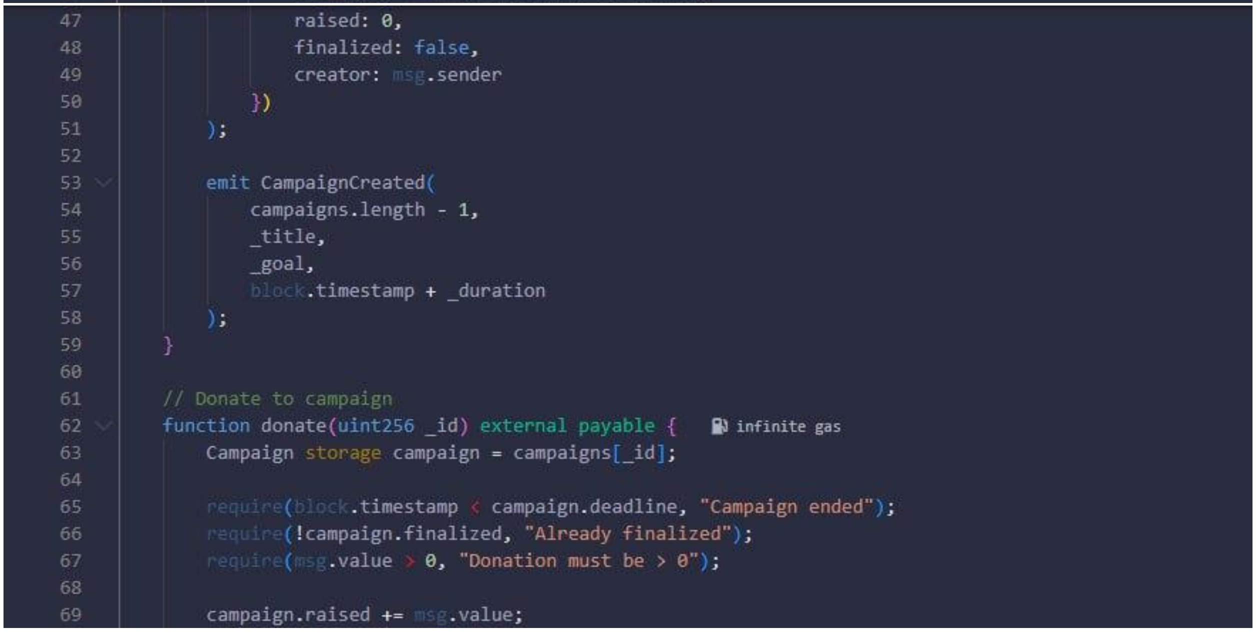
AidPlatform.sol



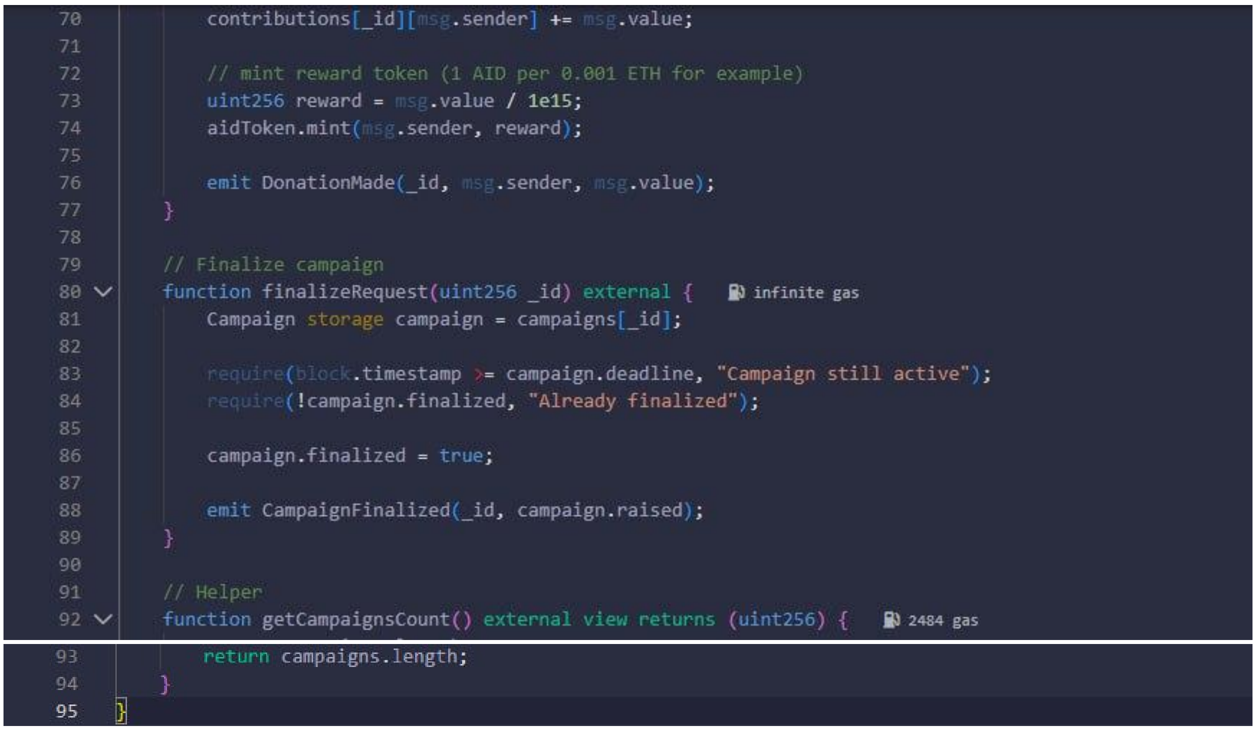
```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

interface IAidToken {
    function mint(address to, uint256 amount) external;
}

contract AidPlatform {
    struct Campaign {
        string title;
        uint256 goal;
        uint256 deadline;
        uint256 raised;
        bool finalized;
        address creator;
    }

    Campaign[] public campaigns;

    // campaignId => donor => amount
    mapping(uint256 => mapping(address => uint256)) public contributions;

    IAidToken public aidToken;
```



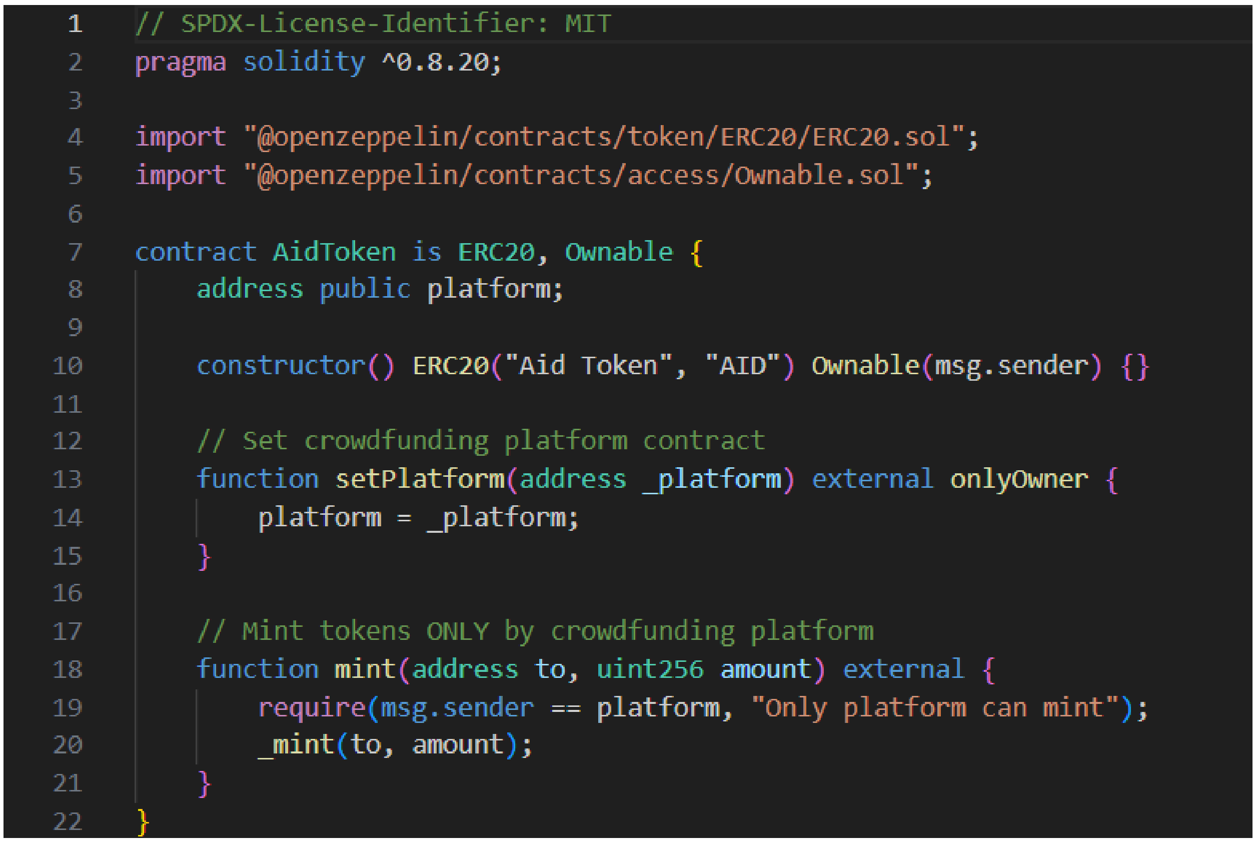
```solidity

    event CampaignCreated(uint256 id, string title, uint256 goal, uint256 deadline);
    event DonationMade(uint256 id, address donor, uint256 amount);
    event CampaignFinalized(uint256 id, uint256 totalRaised);

    constructor(address _aidToken) {
        aidToken = IAidToken(_aidToken);
    }

    // Create campaign
    function createAidRequest(
        string memory _title,
        uint256 _goal,
        uint256 _duration
    ) external {
        require(_goal > 0, "Goal must be > 0");
        require(_duration > 0, "Duration must be > 0");

        campaigns.push(
            Campaign({
                title: _title,
                goal: _goal,
                deadline: block.timestamp + _duration,
```



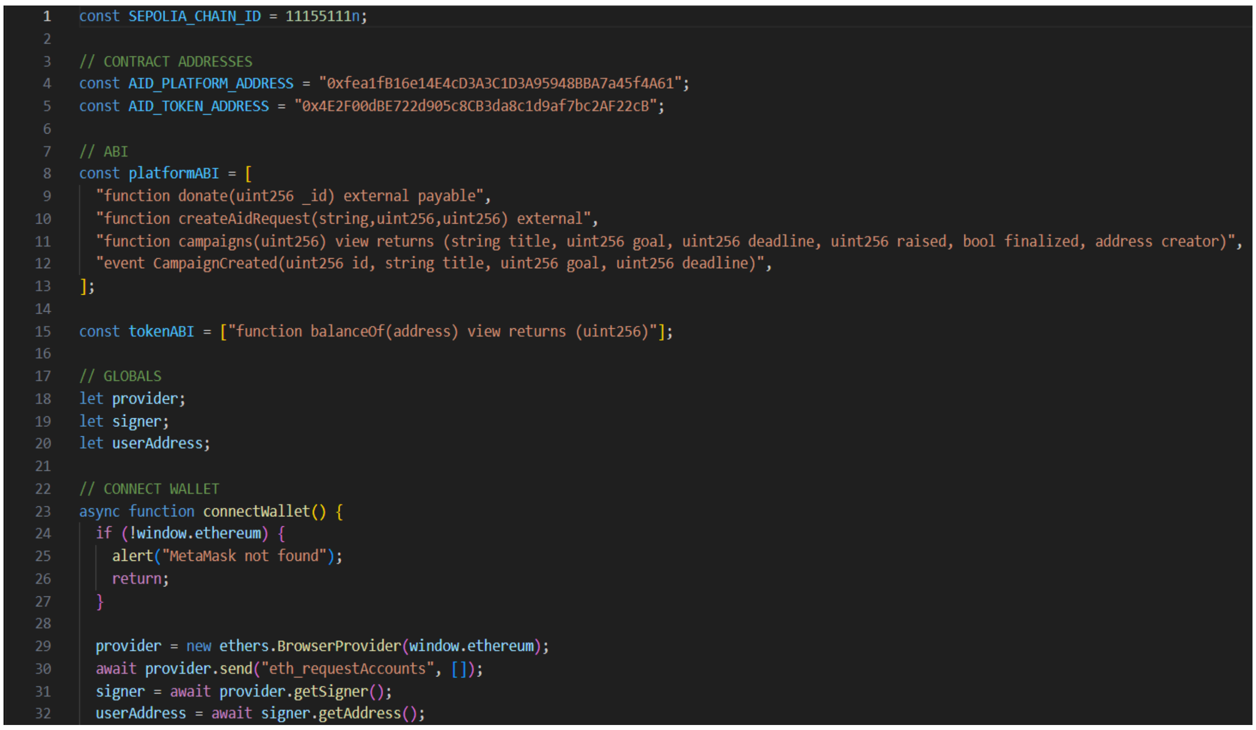
```solidity
                raised: 0,
                finalized: false,
                creator: msg.sender
            })
        );

        emit CampaignCreated(
            campaigns.length - 1,
            _title,
            _goal,
            block.timestamp + _duration
        );
    }

    // Donate to campaign
    function donate(uint256 _id) external payable {
        Campaign storage campaign = campaigns[_id];

        require(block.timestamp < campaign.deadline, "Campaign ended");
        require(!campaign.finalized, "Already finalized");
        require(msg.value > 0, "Donation must be > 0");

        campaign.raised += msg.value;
```


```
        contributions[_id][msg.sender] += msg.value;

        // mint reward token (1 AID per 0.001 ETH for example)
        uint256 reward = msg.value / 1e15;
        aidToken.mint(msg.sender, reward);

        emit DonationMade(_id, msg.sender, msg.value);
    }

    // Finalize campaign
    function finalizeRequest(uint256 _id) external {
        Campaign storage campaign = campaigns[_id];

        require(block.timestamp >= campaign.deadline, "Campaign still active");
        require(!campaign.finalized, "Already finalized");

        campaign.finalized = true;

        emit CampaignFinalized(_id, campaign.raised);
    }

    // Helper
    function getCampaignsCount() external view returns (uint256) {
        return campaigns.length;
    }
}
```

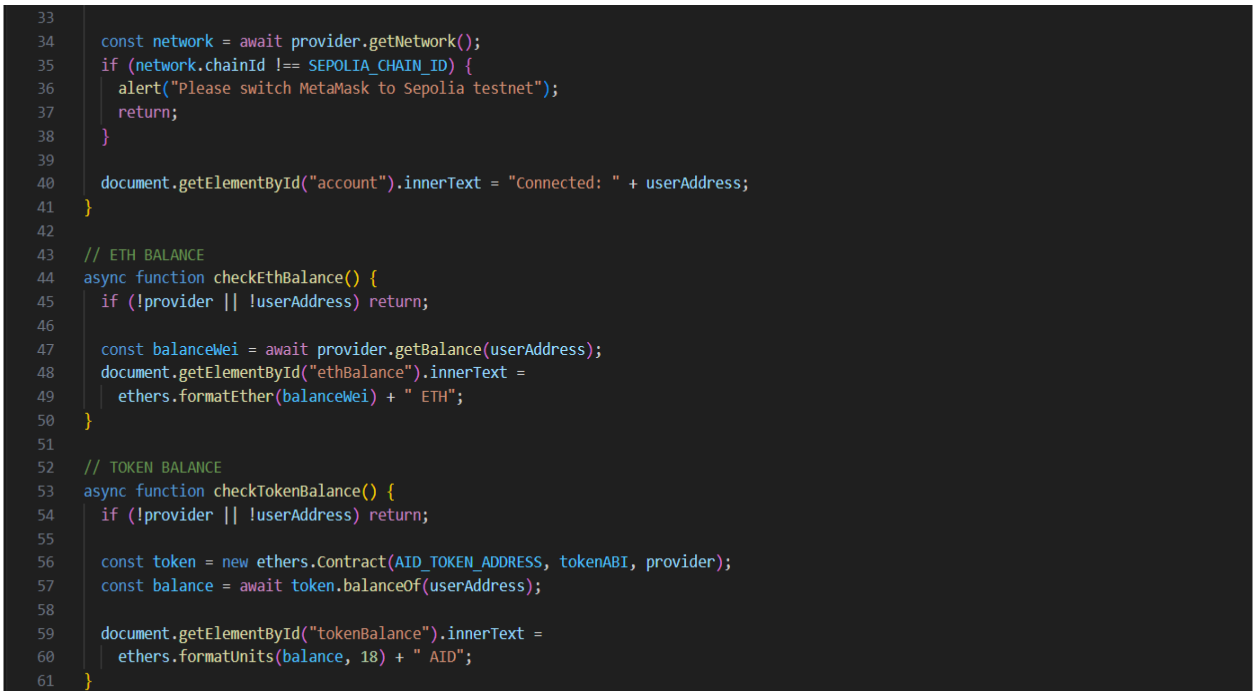
AidToken.sol


```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/access/Ownable.sol";

contract AidToken is ERC20, Ownable {
    address public platform;

    constructor() ERC20("Aid Token", "AID") Ownable(msg.sender) {}

    // Set crowdfunding platform contract
    function setPlatform(address _platform) external onlyOwner {
        platform = _platform;
    }

    // Mint tokens ONLY by crowdfunding platform
    function mint(address to, uint256 amount) external {
        require(msg.sender == platform, "Only platform can mint");
        _mint(to, amount);
    }
}
```

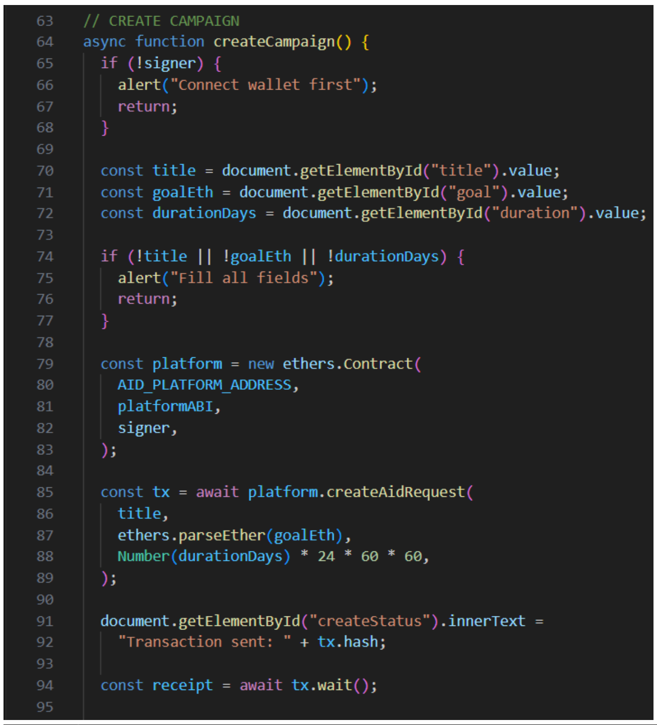
app.js



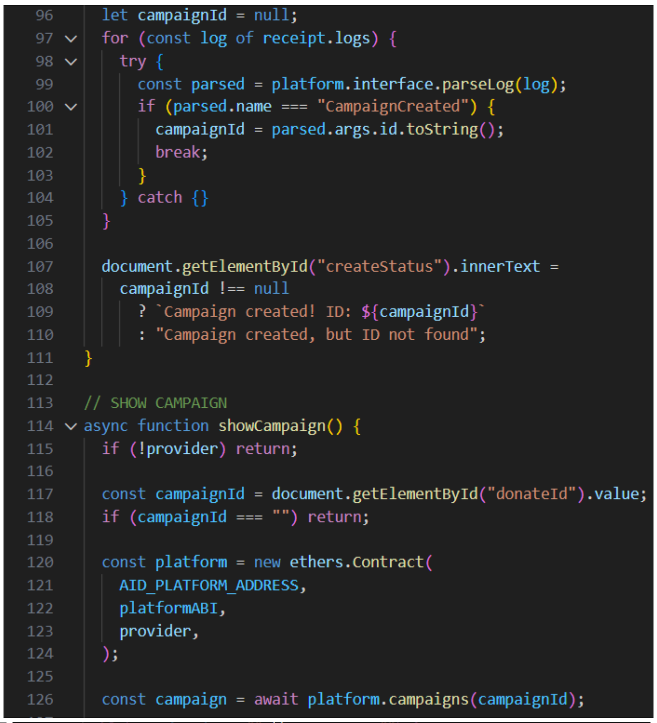
```javascript
const SEPOLIA_CHAIN_ID = 11155111n;

// CONTRACT ADDRESSES
const AID_PLATFORM_ADDRESS = "0xfea1fB16e14E4cD3A3C1D3A95948BBA7a45f4A61";
const AID_TOKEN_ADDRESS = "0x4E2F00dBE722d905c8CB3da8c1d9af7bc2AF22cB";

// ABI
const platformABI = [
  "function donate(uint256 _id) external payable",
  "function createAidRequest(string,uint256,uint256) external",
  "function campaigns(uint256) view returns (string title, uint256 goal, uint256 deadline, uint256 raised, bool finalized, address creator)",
  "event CampaignCreated(uint256 id, string title, uint256 goal, uint256 deadline)",
];

const tokenABI = ["function balanceOf(address) view returns (uint256)"];

// GLOBALS
let provider;
let signer;
let userAddress;

// CONNECT WALLET
async function connectWallet() {
  if (!window.ethereum) {
    alert("MetaMask not found");
    return;
  }

  provider = new ethers.BrowserProvider(window.ethereum);
  await provider.send("eth_requestAccounts", []);
  signer = await provider.getSigner();
  userAddress = await signer.getAddress();
```



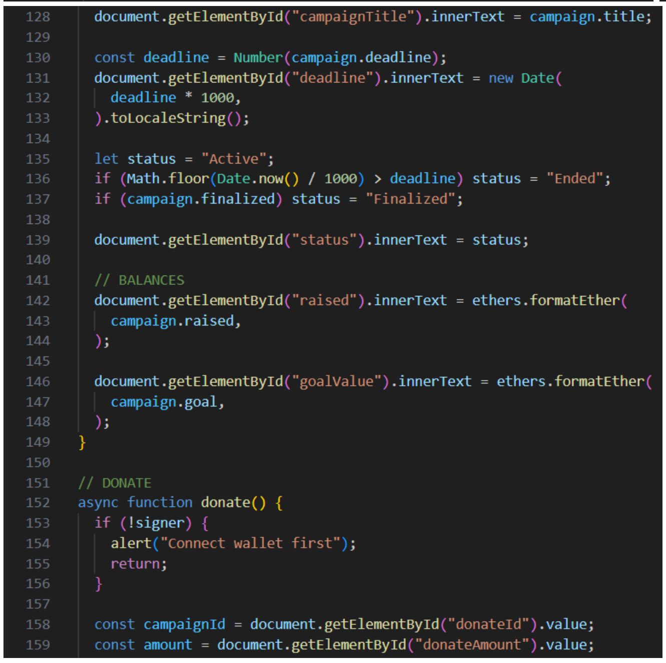
```javascript

  const network = await provider.getNetwork();
  if (network.chainId !== SEPOLIA_CHAIN_ID) {
    alert("Please switch MetaMask to Sepolia testnet");
    return;
  }

  document.getElementById("account").innerText = "Connected: " + userAddress;
}

// ETH BALANCE
async function checkEthBalance() {
  if (!provider || !userAddress) return;

  const balanceWei = await provider.getBalance(userAddress);
  document.getElementById("ethBalance").innerText =
    ethers.formatEther(balanceWei) + " ETH";
}

// TOKEN BALANCE
async function checkTokenBalance() {
  if (!provider || !userAddress) return;

  const token = new ethers.Contract(AID_TOKEN_ADDRESS, tokenABI, provider);
  const balance = await token.balanceOf(userAddress);

  document.getElementById("tokenBalance").innerText =
    ethers.formatUnits(balance, 18) + " AID";
}
```



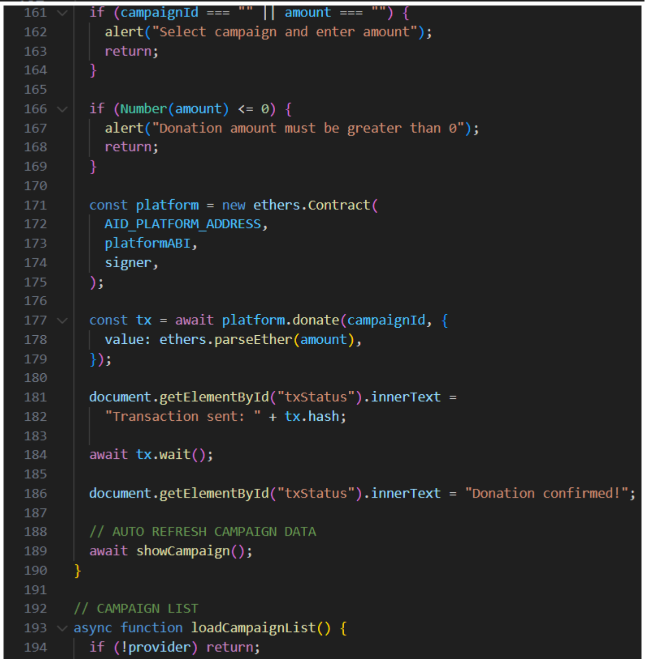
```
// CREATE CAMPAIGN
async function createCampaign() {
  if (!signer) {
    alert("Connect wallet first");
    return;
  }

  const title = document.getElementById("title").value;
  const goalEth = document.getElementById("goal").value;
  const durationDays = document.getElementById("duration").value;

  if (!title || !goalEth || !durationDays) {
    alert("Fill all fields");
    return;
  }

  const platform = new ethers.Contract(
    AID_PLATFORM_ADDRESS,
    platformABI,
    signer,
  );

  const tx = await platform.createAidRequest(
    title,
    ethers.parseEther(goalEth),
    Number(durationDays) * 24 * 60 * 60,
  );

  document.getElementById("createStatus").innerText =
    "Transaction sent: " + tx.hash;

  const receipt = await tx.wait();

```



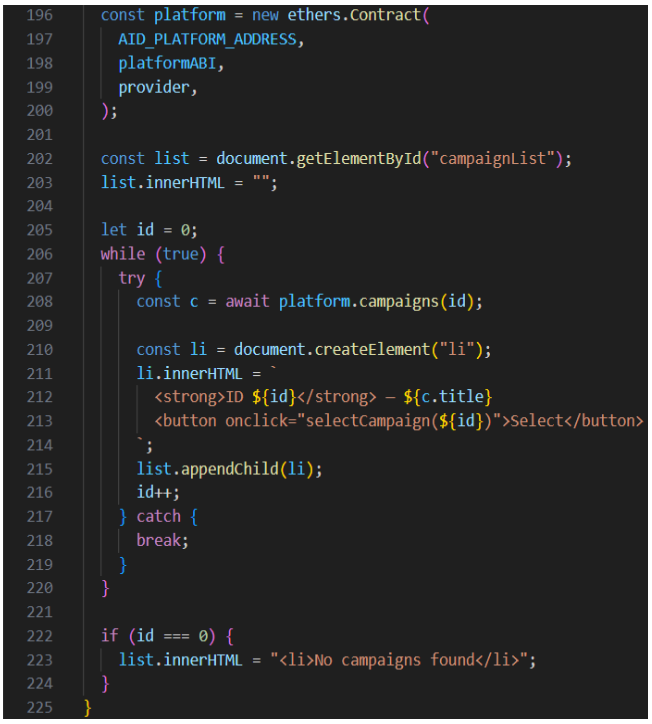
```
  let campaignId = null;
  for (const log of receipt.logs) {
    try {
      const parsed = platform.interface.parseLog(log);
      if (parsed.name === "CampaignCreated") {
        campaignId = parsed.args.id.toString();
        break;
      }
    } catch {}
  }

  document.getElementById("createStatus").innerText =
    campaignId !== null
      ? `Campaign created! ID: ${campaignId}`
      : "Campaign created, but ID not found";
}

// SHOW CAMPAIGN
async function showCampaign() {
  if (!provider) return;

  const campaignId = document.getElementById("donateId").value;
  if (campaignId === "") return;

  const platform = new ethers.Contract(
    AID_PLATFORM_ADDRESS,
    platformABI,
    provider,
  );

  const campaign = await platform.campaigns(campaignId);
```



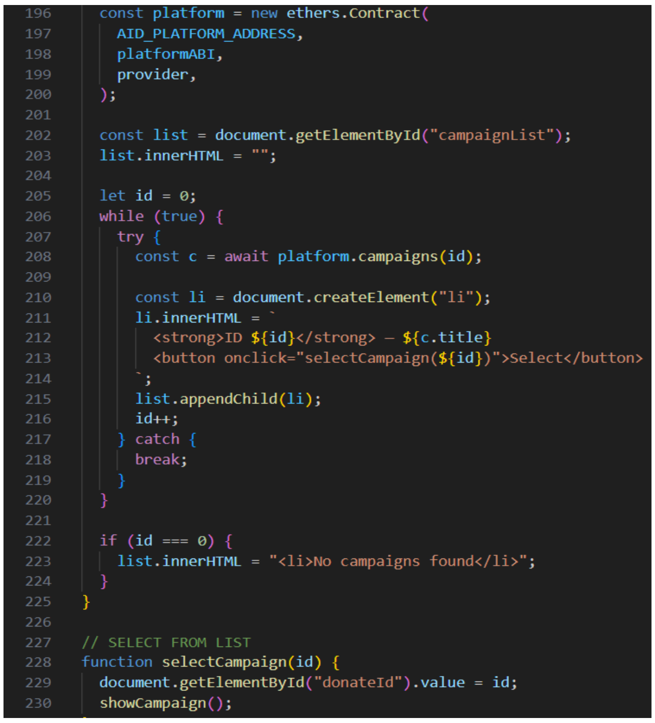
```
  document.getElementById("campaignTitle").innerText = campaign.title;

  const deadline = Number(campaign.deadline);
  document.getElementById("deadline").innerText = new Date(
    deadline * 1000,
  ).toLocaleString();

  let status = "Active";
  if (Math.floor(Date.now() / 1000) > deadline) status = "Ended";
  if (campaign.finalized) status = "Finalized";

  document.getElementById("status").innerText = status;

  // BALANCES
  document.getElementById("raised").innerText = ethers.formatEther(
    campaign.raised,
  );

  document.getElementById("goalValue").innerText = ethers.formatEther(
    campaign.goal,
  );
}

// DONATE
async function donate() {
  if (!signer) {
    alert("Connect wallet first");
    return;
  }

  const campaignId = document.getElementById("donateId").value;
  const amount = document.getElementById("donateAmount").value;
```



```
  if (campaignId === "" || amount === "") {
    alert("Select campaign and enter amount");
    return;
  }

  if (Number(amount) <= 0) {
    alert("Donation amount must be greater than 0");
    return;
  }

  const platform = new ethers.Contract(
    AID_PLATFORM_ADDRESS,
    platformABI,
    signer,
  );

  const tx = await platform.donate(campaignId, {
    value: ethers.parseEther(amount),
  });

  document.getElementById("txStatus").innerText =
    "Transaction sent: " + tx.hash;

  await tx.wait();

  document.getElementById("txStatus").innerText = "Donation confirmed!";

  // AUTO REFRESH CAMPAIGN DATA
  await showCampaign();
}

// CAMPAIGN LIST
async function loadCampaignList() {
  if (!provider) return;
```

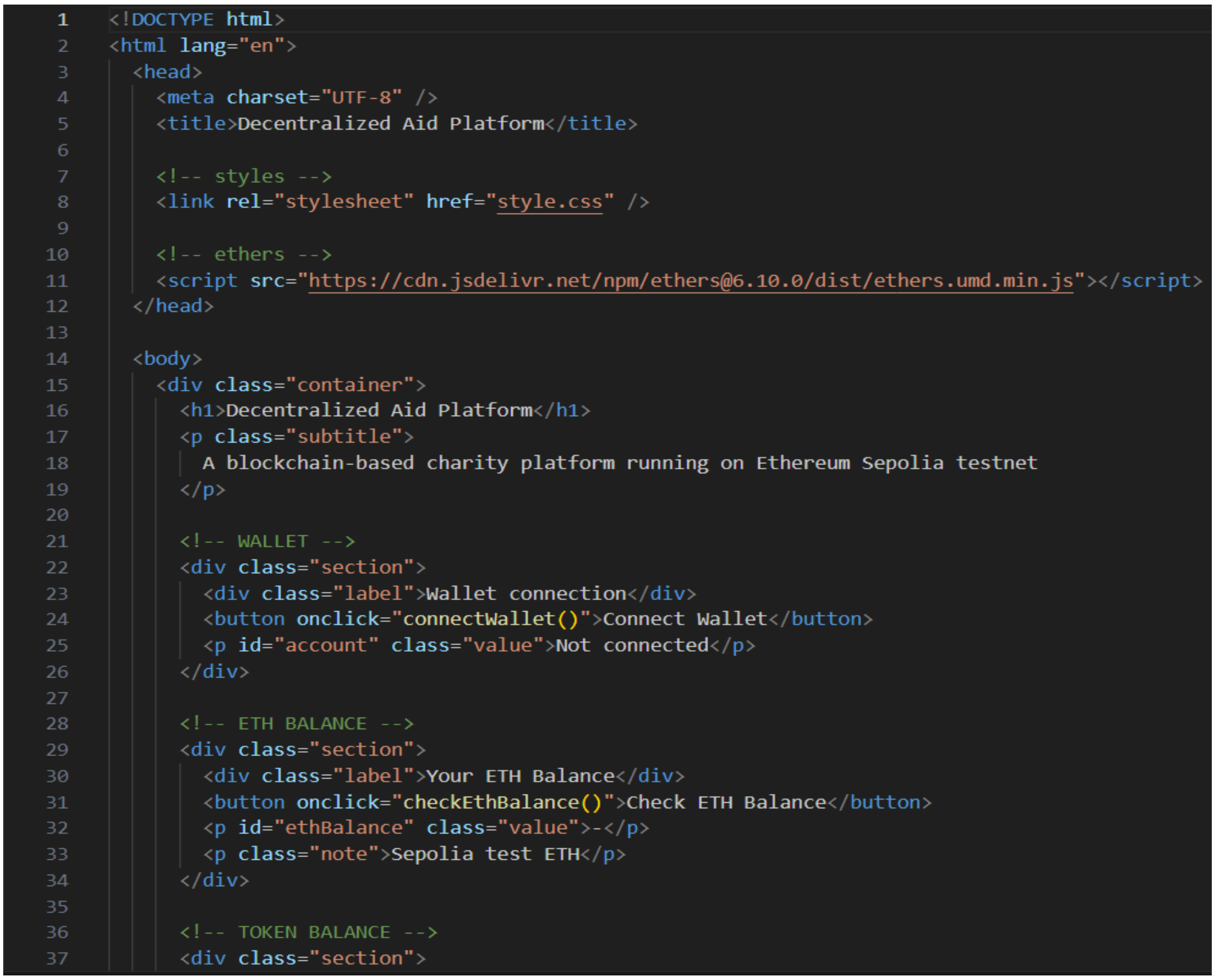


```javascript
  const platform = new ethers.Contract(
    AID_PLATFORM_ADDRESS,
    platformABI,
    provider,
  );

  const list = document.getElementById("campaignList");
  list.innerHTML = "";

  let id = 0;
  while (true) {
    try {
      const c = await platform.campaigns(id);

      const li = document.createElement("li");
      li.innerHTML = `
        <strong>ID ${id}</strong> — ${c.title}
        <button onclick="selectCampaign(${id})">Select</button>
      `;
      list.appendChild(li);
      id++;
    } catch {
      break;
    }
  }

  if (id === 0) {
    list.innerHTML = "<li>No campaigns found</li>";
  }
}
```

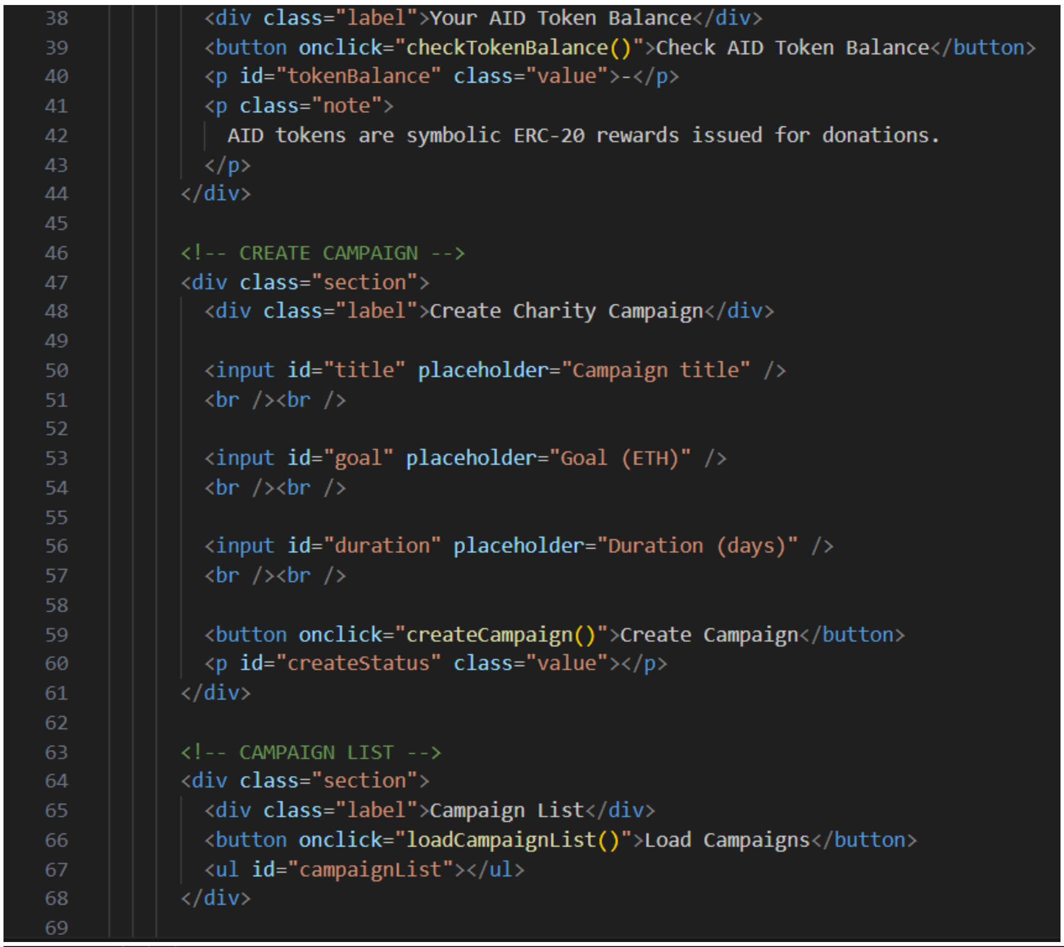


```javascript
  const platform = new ethers.Contract(
    AID_PLATFORM_ADDRESS,
    platformABI,
    provider,
  );

  const list = document.getElementById("campaignList");
  list.innerHTML = "";

  let id = 0;
  while (true) {
    try {
      const c = await platform.campaigns(id);

      const li = document.createElement("li");
      li.innerHTML = `
        <strong>ID ${id}</strong> — ${c.title}
        <button onclick="selectCampaign(${id})">Select</button>
      `;
      list.appendChild(li);
      id++;
    } catch {
      break;
    }
  }

  if (id === 0) {
    list.innerHTML = "<li>No campaigns found</li>";
  }
}

// SELECT FROM LIST
function selectCampaign(id) {
  document.getElementById("donateId").value = id;
  showCampaign();
```

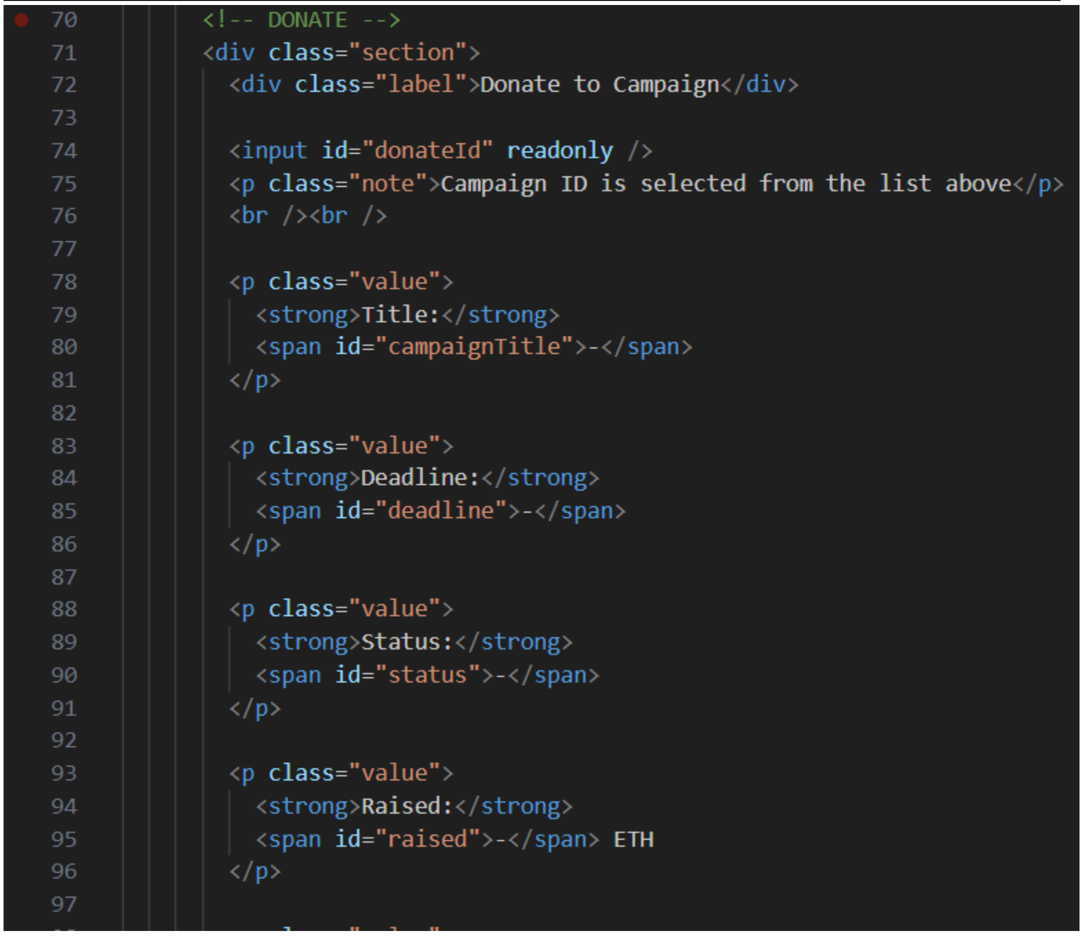
index.html



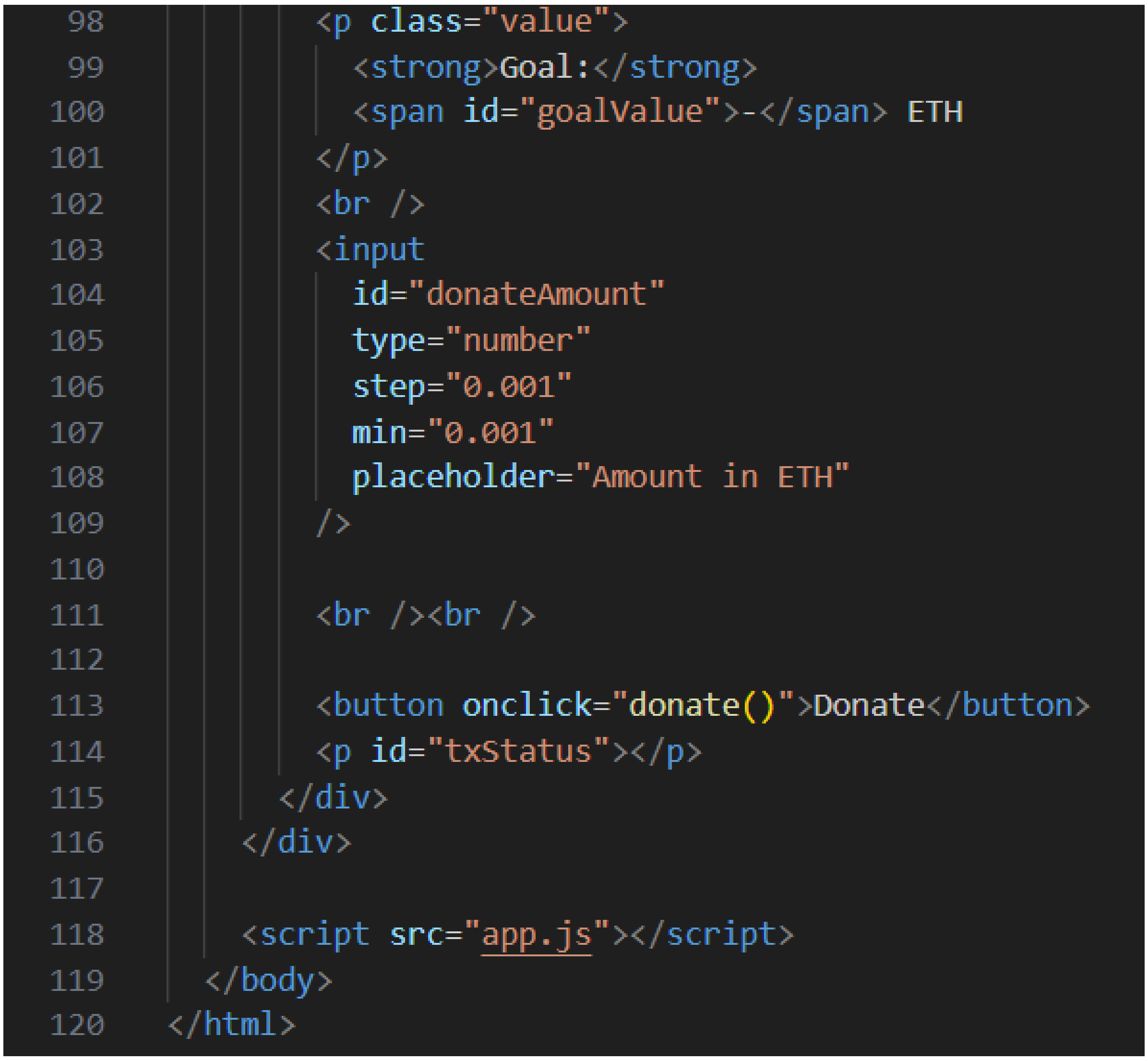
```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Decentralized Aid Platform</title>

    <!-- styles -->
    <link rel="stylesheet" href="style.css" />

    <!-- ethers -->
    <script src="https://cdn.jsdelivr.net/npm/ethers@6.10.0/dist/ethers.umd.min.js"></script>
  </head>

  <body>
    <div class="container">
      <h1>Decentralized Aid Platform</h1>
      <p class="subtitle">
        A blockchain-based charity platform running on Ethereum Sepolia testnet
      </p>

      <!-- WALLET -->
      <div class="section">
        <div class="label">Wallet connection</div>
        <button onclick="connectWallet()">Connect Wallet</button>
        <p id="account" class="value">Not connected</p>
      </div>

      <!-- ETH BALANCE -->
      <div class="section">
        <div class="label">Your ETH Balance</div>
        <button onclick="checkEthBalance()">Check ETH Balance</button>
        <p id="ethBalance" class="value">-</p>
        <p class="note">Sepolia test ETH</p>
      </div>

      <!-- TOKEN BALANCE -->
      <div class="section">
```



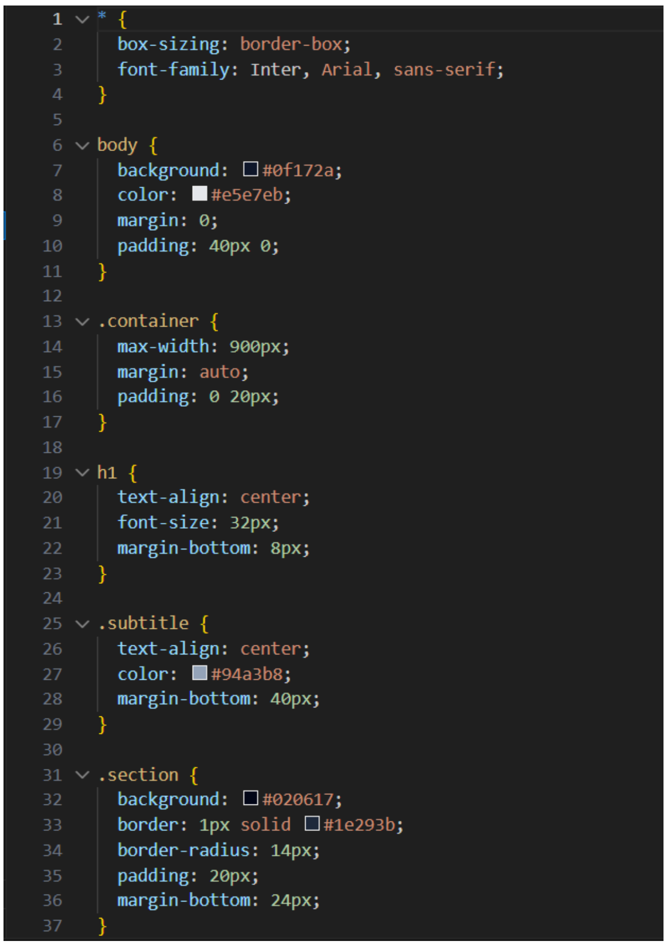
```html
          <div class="label">Your AID Token Balance</div>
          <button onclick="checkTokenBalance()">Check AID Token Balance</button>
          <p id="tokenBalance" class="value">-</p>
          <p class="note">
            AID tokens are symbolic ERC-20 rewards issued for donations.
          </p>
        </div>

        <!-- CREATE CAMPAIGN -->
        <div class="section">
          <div class="label">Create Charity Campaign</div>

          <input id="title" placeholder="Campaign title" />
          <br /><br />

          <input id="goal" placeholder="Goal (ETH)" />
          <br /><br />

          <input id="duration" placeholder="Duration (days)" />
          <br /><br />

          <button onclick="createCampaign()">Create Campaign</button>
          <p id="createStatus" class="value"></p>
        </div>

        <!-- CAMPAIGN LIST -->
        <div class="section">
          <div class="label">Campaign List</div>
          <button onclick="loadCampaignList()">Load Campaigns</button>
          <ul id="campaignList"></ul>
        </div>

```



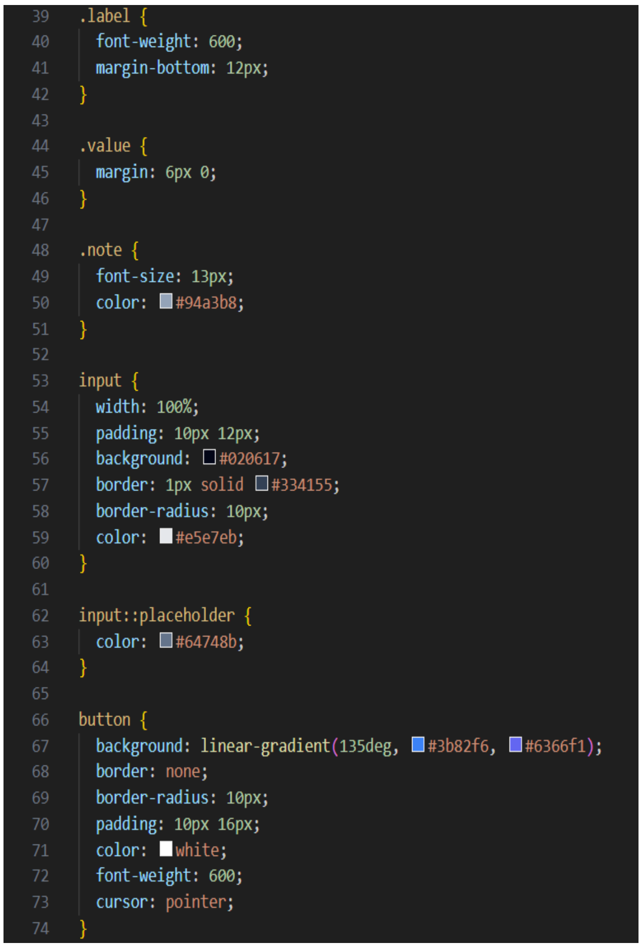
```html
        <!-- DONATE -->
        <div class="section">
          <div class="label">Donate to Campaign</div>

          <input id="donateId" readonly />
          <p class="note">Campaign ID is selected from the list above</p>
          <br /><br />

          <p class="value">
            <strong>Title:</strong>
            <span id="campaignTitle">-</span>
          </p>

          <p class="value">
            <strong>Deadline:</strong>
            <span id="deadline">-</span>
          </p>

          <p class="value">
            <strong>Status:</strong>
            <span id="status">-</span>
          </p>

          <p class="value">
            <strong>Raised:</strong>
            <span id="raised">-</span> ETH
          </p>

```


```html
            <p class="value">
              <strong>Goal:</strong>
              <span id="goalValue">-</span> ETH
            </p>
            <br />
            <input
              id="donateAmount"
              type="number"
              step="0.001"
              min="0.001"
              placeholder="Amount in ETH"
            />

            <br /><br />

            <button onclick="donate()">Donate</button>
            <p id="txStatus"></p>
          </div>
        </div>

        <script src="app.js"></script>
      </body>
    </html>
```

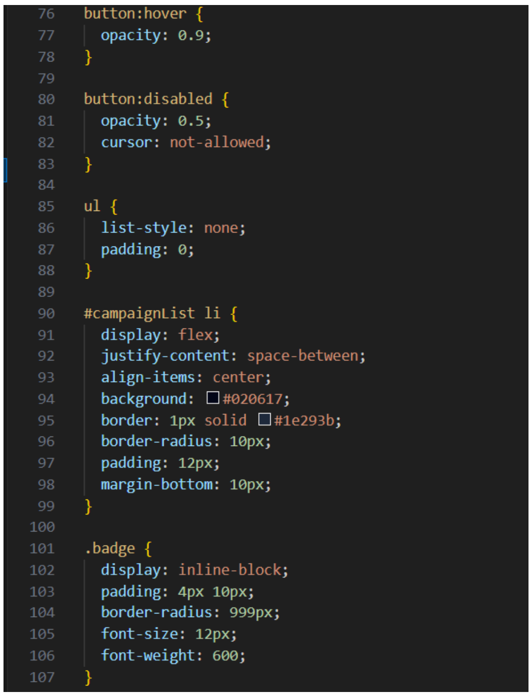
style.css



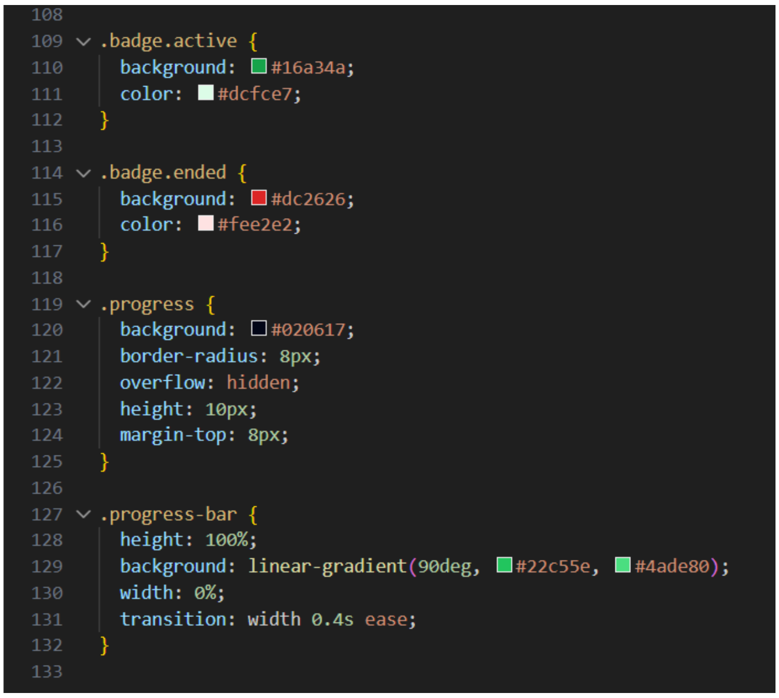
```css
* {
  box-sizing: border-box;
  font-family: Inter, Arial, sans-serif;
}

body {
  background: #0f172a;
  color: #e5e7eb;
  margin: 0;
  padding: 40px 0;
}

.container {
  max-width: 900px;
  margin: auto;
  padding: 0 20px;
}

h1 {
  text-align: center;
  font-size: 32px;
  margin-bottom: 8px;
}

.subtitle {
  text-align: center;
  color: #94a3b8;
  margin-bottom: 40px;
}

.section {
  background: #020617;
  border: 1px solid #1e293b;
  border-radius: 14px;
  padding: 20px;
  margin-bottom: 24px;
}
```



```css
.label {
  font-weight: 600;
  margin-bottom: 12px;
}

.value {
  margin: 6px 0;
}

.note {
  font-size: 13px;
  color: #94a3b8;
}

input {
  width: 100%;
  padding: 10px 12px;
  background: #020617;
  border: 1px solid #334155;
  border-radius: 10px;
  color: #e5e7eb;
}

input::placeholder {
  color: #64748b;
}

button {
  background: linear-gradient(135deg, #3b82f6, #6366f1);
  border: none;
  border-radius: 10px;
  padding: 10px 16px;
  color: white;
  font-weight: 600;
  cursor: pointer;
}
```

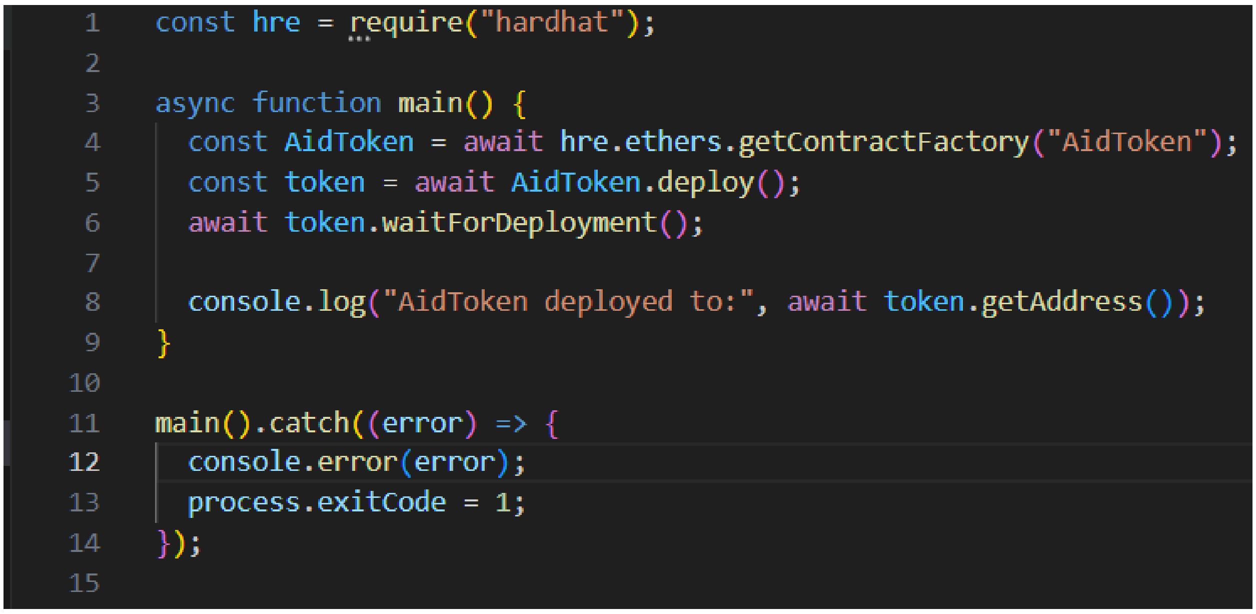


```css
button:hover {
  opacity: 0.9;
}

button:disabled {
  opacity: 0.5;
  cursor: not-allowed;
}

ul {
  list-style: none;
  padding: 0;
}

#campaignList li {
  display: flex;
  justify-content: space-between;
  align-items: center;
  background: #020617;
  border: 1px solid #1e293b;
  border-radius: 10px;
  padding: 12px;
  margin-bottom: 10px;
}

.badge {
  display: inline-block;
  padding: 4px 10px;
  border-radius: 999px;
  font-size: 12px;
  font-weight: 600;
}
```



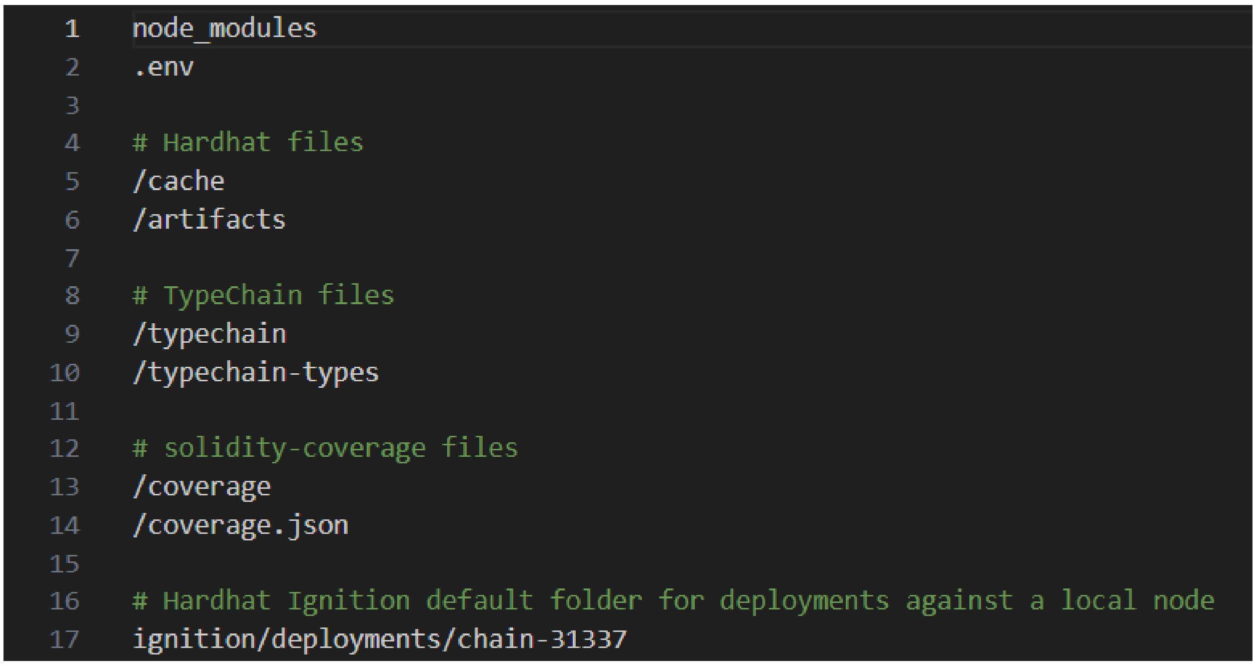
```css

.badge.active {
  background: #16a34a;
  color: #dcfce7;
}

.badge.ended {
  background: #dc2626;
  color: #fee2e2;
}

.progress {
  background: #020617;
  border-radius: 8px;
  overflow: hidden;
  height: 10px;
  margin-top: 8px;
}

.progress-bar {
  height: 100%;
  background: linear-gradient(90deg, #22c55e, #4ade80);
  width: 0%;
  transition: width 0.4s ease;
}

```

deploy.js



```javascript
const hre = require("hardhat");

async function main() {
  const AidToken = await hre.ethers.getContractFactory("AidToken");
  const token = await AidToken.deploy();
  await token.waitForDeployment();

  console.log("AidToken deployed to:", await token.getAddress());
}

main().catch((error) => {
  console.error(error);
  process.exitCode = 1;
});
```

.gitignore



```
node_modules
.env

# Hardhat files
/cache
/artifacts

# TypeChain files
/typechain
/typechain-types

# solidity-coverage files
/coverage
/coverage.json

# Hardhat Ignition default folder for deployments against a local node
ignition/deployments/chain-31337
```

Technical Documentation

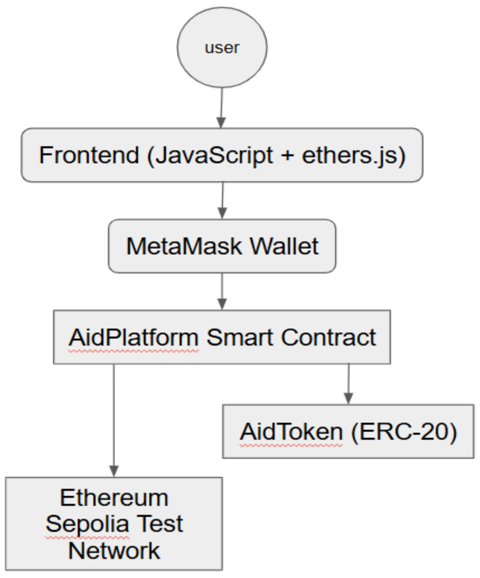
Ethereum Test Network Usage: This decentralized app was created strictly for use on the Ethereum Sepolia test network only. Deployment onto the Ethereum mainnet is not allowed, so there is no real crypto currency involved in transactions occurring within the app. All transactions with the blockchain are done using free test ETH.

Obtaining Test ETH: We obtained Test ETH from the official Sepolia testnet faucet: https://cloud.google.com/application/web3/faucet/ethereum/sepolia, which initially enabled us to deploy our smart contracts and also provide donations for our donation test transactions.

Finalizing Campaign Logic: The campaign can only be finalized after its expiration date. If the amount collected equals or exceeds the amount needed to fund the campaign, it was considered successful. The campaign creator can then withdraw funds raised from the donation's contributions. If the funding goal is not met, all contributors to the campaign will be able to receive a full refund of their donations.

MetaMask Integration: The application is integrated with MetaMask, which allows a user to connect their wallet to the application; when doing so, the application requests permission from the user to access their wallet accounts. Prior to executing a transaction, the application checks that the selected network is the Ethereum Sepolia test network; when MetaMask confirms a transaction, it means that the transaction has been successfully recorded on the blockchain.

Application Architecture: Frontend development was done using JavaScript and ethers.js, with Solidity-based smart contracts working on the Ethereum Sepolia test network as the application's foundational support. The frontend user experience is provided through MetaMask integration while communicating with the smart contracts.


user
Frontend (JavaScript + ethers.js)
MetaMask Wallet
AidPlatform Smart Contract
AidToken (ERC-20)
Ethereum Sepolia Test Network

Design and Technical Decisions

Smart Contract Design Decisions

The smart contract has been designed with an emphasis on simple and clear logic for both the purpose of crowdfunding and interaction with the Blockchain so that both core concepts can be conveyed effectively. The on-chain storage of campaign information (i.e., title, funding goal, deadline and amount raised) provides the necessary transparency and immutability of campaigning data. Contributions from individual users can be tracked using mappings, providing accurate accounting and the availability of refunds if needed.

To create better modularity and readability, the platform logic and token logic are separated into two (2) distinct smart contracts. Therefore, the crowdfunding and tokenization processes can operate independently while still being able to interact with one another securely.

Selection of the ERC20 Token as a Reward System

We selected the ERC-20 type of token because it is one of the most agreements in the Ethereum ecosystem. It allows for the demonstration of the fundamental concepts of tokenization (minting, balances, and access control) as well as having the capability to automate the minting of tokens at the time of donation, which facilitates a better experience for users, removing the necessity to manually distribute the rewards.

The token represents no intrinsic value and is for educational purposes only. The rights to mint tokens are restricted to the contract of the crowdfunding platform to prevent any unauthorized tokens from being produced, thereby increasing security.

Considerations of Security

A number of fundamental security measures were considered in the development process. Campaigns must wait until after the deadline to be finalized, which prevents premature closure of the campaigns. The funds may only be withdrawn by the creator of the campaign and only when that campaign has been successful. Refunds can only be issued to the original contributors, and only in the case of the campaign receiving no funding.

In addition, reentrancy protection methods were implemented in highly sensitive operations like donation, withdrawal, and refund functions to safeguard against performing a sequence of the same operation multiple times.

Decision-making on front-end design:

The front-end design was kept very simplistic on purpose so that the focus of the website will be on Blockchain interaction and not the visual complexity of different colour and/or style designs. In this case, we chose to use JavaScript as our programming language and ethers.js so that we can communicate with Smart Contracts on the Ethereum Blockchain due to their reliability, widespread use on Ethereum Based Applications, as well as their great library of tools for creating Ethereum-based smart contracts.

User interactions, including connecting a Wallet and making a donation or viewing balances, are handled completely through the use of MetaMask. This gives the user total control of their wallet, as well as the ability to approve or reject any type of transaction as the action occurs. This design decision further helps reinforce the decentralised concept of the application and also avoids the need to store any type of sensitive wallet data on the application server.

Development Environment

We decided to use Remix IDE and Hardhat for development and testing to improve smart contract debuggability and deployment. Using Remix, we were able to quickly prototype and test contracts, while Hardhat provided a structured environment to create both compilation and deployment scripts. This combination of tools ensured a highly efficient development process while maintaining full compatibility with Ethereum test networks

Scalability and educational purpose:

The objectives for this project were limited for educational purposes and were designed for clarity regarding functionality. The application's current design could be easily extended to include additional features such as categorising Campaigns or creating complex User Roles; however, the basic design focuses only on the primary Blockchain concepts of Smart Contracts, Tokenisation, and Decentralised User Interaction.

Conclusion

The conclusion drawn from the information above is that blockchain can be used in real-world situations and create decentralized crowdfunding applications. In creating the crowdfunding application, it used Solidity for developing smart contracts as well as JavaScript to interact with the front end, MetaMask to receive contributions and deploy and test the solution using the Ethereum Sepolia Test Network.

While working on the project, concepts in blockchain such as the development of smart contracts, a decentralized way to interact with users, a method of tokenizing using ERC-20 token standards, and using test networks were implemented and validated. The several steps that are involved in the crowdfunding process (campaign creating, handling contributions, issuing reward tokens, finalising campaigns, and refunding/withdrawing) show understanding of how a crowdfunding platform works in a decentralised way.

A clear emphasis on the security and transparency of an application's structure is shown by having data stored on the blockchain, controlled minting of tokens, and limited access to certain processes. Using MetaMask allowed users to authorise and record transactions directly onto the blockchain thus reinforcing the decentralisation of this application.

This project met the goals of the Blockchain course as it provided an example of how theoretical concepts apply to a real-world application. The knowledge and experience gained by developing this application provide an excellent basis for learning about decentralised applications and you will find ways of adding additional functionality to this application in the future; however, it must remain restricted to being developed and used strictly in an educational or testing environment.