

# Blockchain & Solidity Lab4 – Crowdfunding dApp Development

---

S2BC



---

## Lab 4: Run a dApp and Consider Next Steps

- BUILD / TEST / INTEGRATE / **RUN**

---

So far, you've followed the steps in Labs 1 to 3, gaining valuable insights into the core components of blockchain development. Now, in Lab 4, we will discuss crucial considerations for running a dApp in Morpheus.

### 1. Running the front-end

Follow these steps to run the front-end of your crowdfunding Dapp:

#### 1. Start the front-end Server:

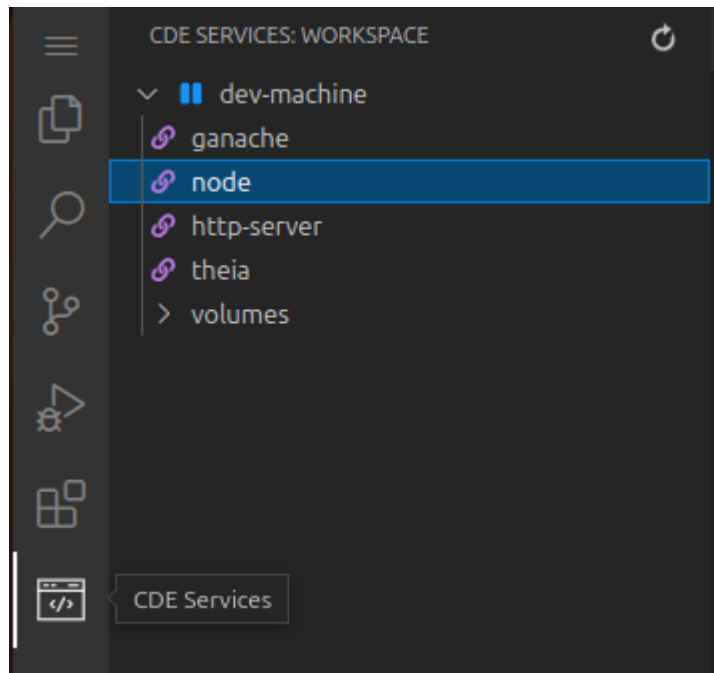
- Navigate to the `crowd-funding-update-2024/front-end` directory.
- Run the following command:

```
npm run dev
```

This will initiate the server for your Dapp's front-end.

#### 2. Open the Web App in Morpheus:

- In your Morpheus IDE interface, locate the CDE menu in the left menu bar.



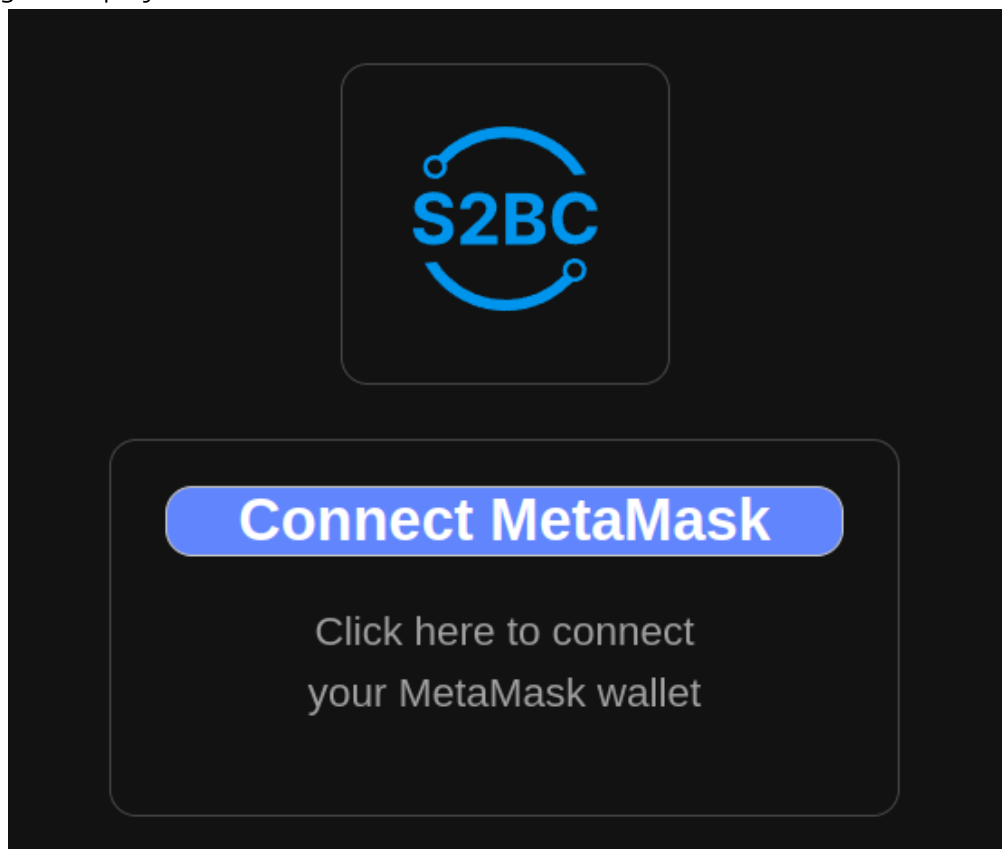
- Click on the node service to open your web app.

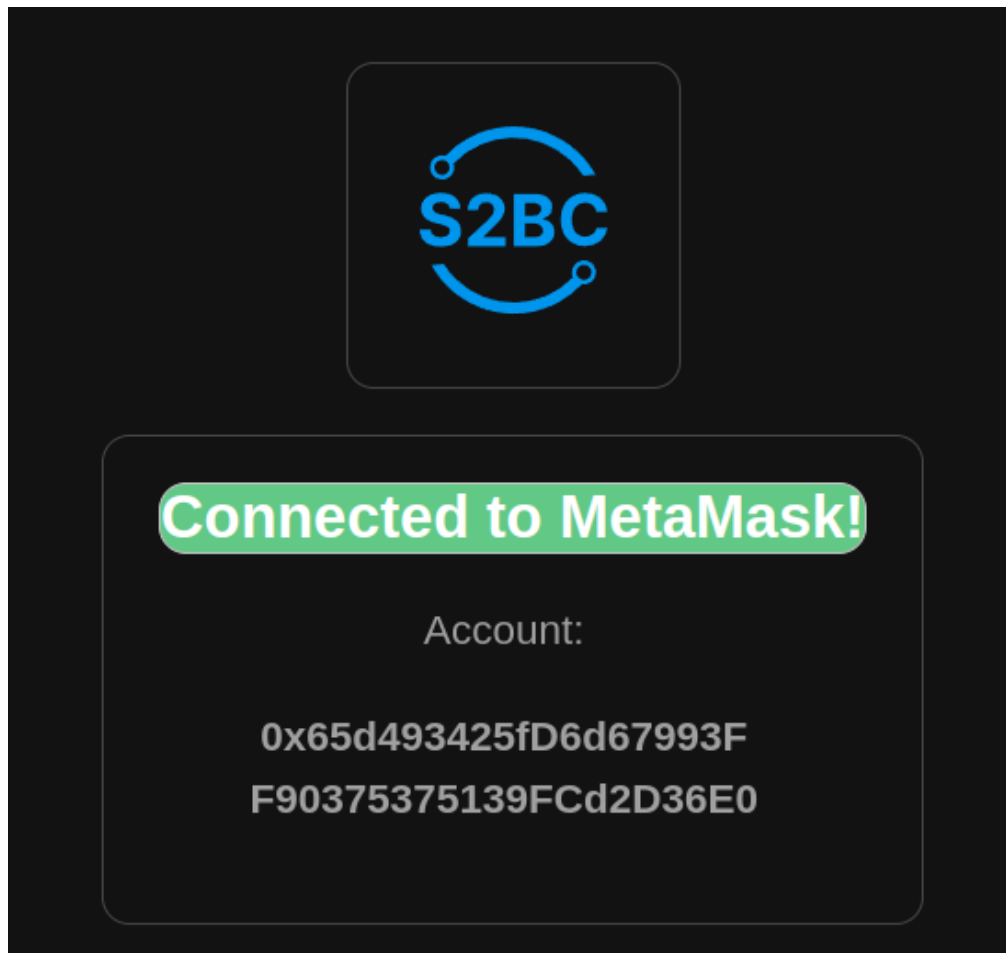
These steps ensure that your front-end server is up and running, and you can access your crowdfunding Dapp through Morpheus.

## 2. Trying the Crowdfunding Dapp

### 1. Connect to Metamask:

- Click the connect button. A Metamask popup will appear, asking if you want to connect. Accept using the deployer account.



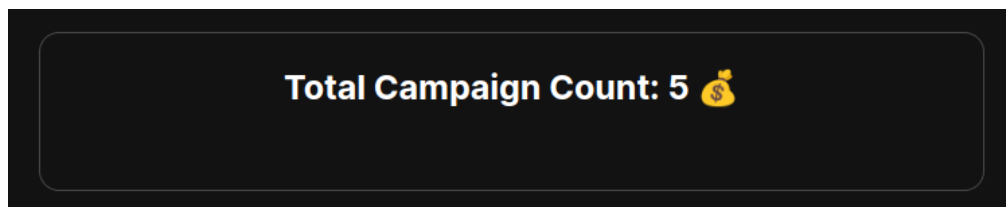


2. Create a campaign:

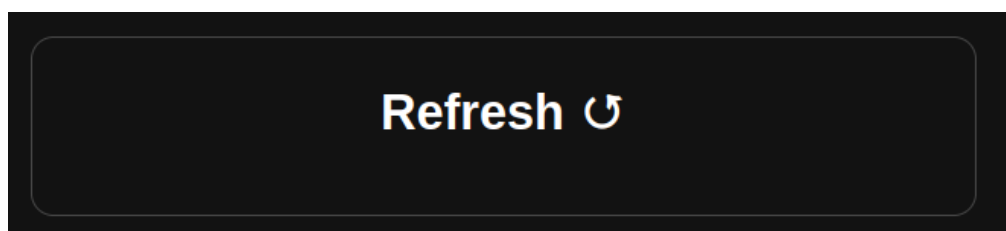
- Create a Campaign on the main dashboard:

A dark-themed form titled 'Campaign Creation:'. It contains three input fields: the first is labeled 'Description / title', the second contains the number '2', and the third is a button labeled 'Create Campaign' with a rocket icon.

3. Check count campaign number:



4. Refresh if needed:



- 2. Check the new campaign:

- The campaign's dashboard appeared at bottom:

## Tutorial Campaign

---

**id: 5**  
**0x0D3AE...1454**

---

Instance address: 0x0D3AE9E10d98  
B6eda12D645283387354b2801454

**Campaign details:**

Description: Tutorial Campaign

Manager: 0x65d493425fD6d67  
993FF90375375139FCd2D36E0

Minimum Contribution: 1  
0000000000000000


Contract Balance: 0

Number of Supporters: 0

Number of Requests: 0

---

**Support Campaign:**

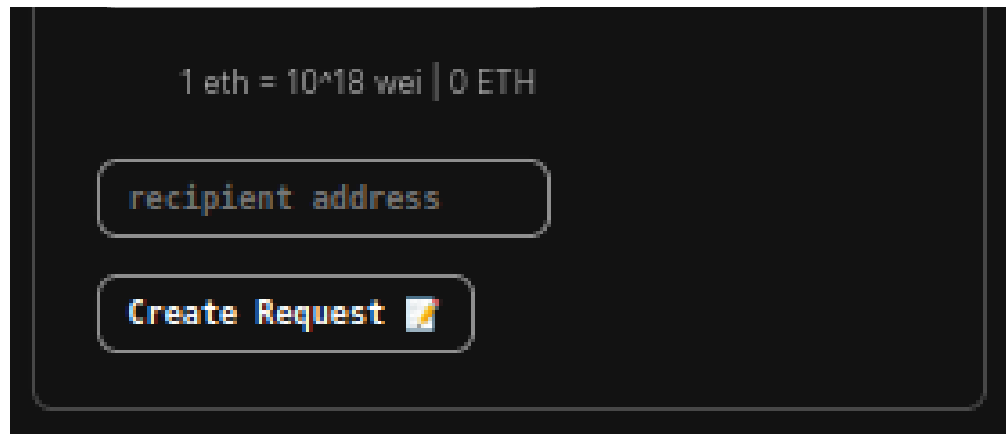
**Contribute** 

1 eth = 10<sup>18</sup> wei | Minimum  
contribution: 0 ETH

---


**Create release fund request:**

Campaign manager  
can propose donation.



1 eth = 10<sup>18</sup> wei | 0 ETH

recipient address

Create Request 

#### 5. Contribute:

- Participate in the campaign by contributing. (manager or/and supporter)



**Support Campaign:**

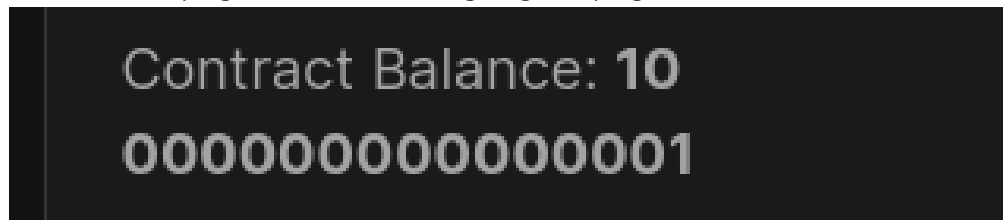
contribution (wei) 

Contribute 

1 eth = 10<sup>18</sup> wei | Minimum contribution: 0 ETH

#### 6. Check balance campaign:

- Return to the Campaign Panel to view ongoing campaign information.



Contract Balance: 10  
000000000000000001

#### 7. Create a request:

- Manager can create a request.

contribution: 0 ETH

---

### Create release fund request:


Campaign manager  
can propose donation.

request description

request amount

1 eth = 10<sup>18</sup> wei | 0 ETH

recipient address

Create Request 

#### 8. Review request:

- In Campaign Panel, review requests made by campaign manager.

---

### Request 1:


Description: **First request send money**

Amount: **1000000000000000001**

Recipient Address: **0x65d493425fD6  
d67993FF90375375139FCd2D36E0**

Finalized ? : **false**

Approve ✓

Finalize 

To finalize a request, the number  
of approvals must exceed  
half of the total supporters.

---

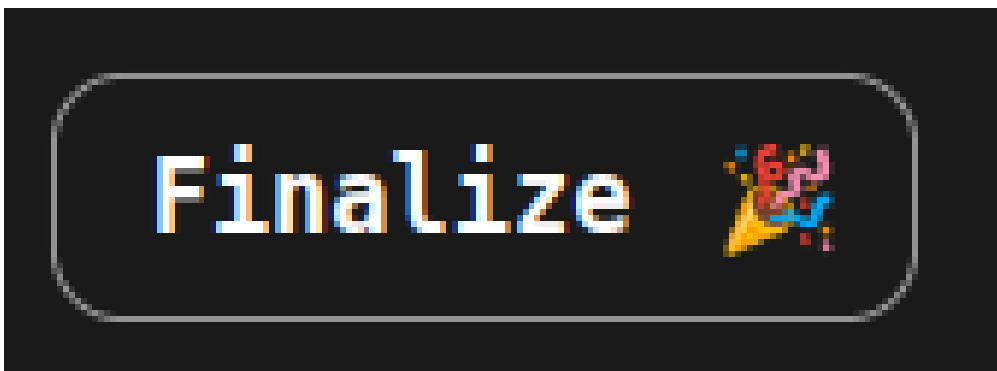
#### 9. Approve request:

- In Campaign Panel, approve requests made by campaign manager.



#### 10. Finalize request:

- Once the campaign is completed, finalize requests to release funds. The funds will be transferred to the recipient address.



Check the finalized status on the campaign panel:



### 3. Migrating to Sepolia Testnet and Utilizing Etherscan

To successfully migrate your dApp to the Sepolia Testnet and leverage Etherscan for enhanced visibility, follow the steps below:

#### Step 1: Obtain RPC\_URL and Etherscan API Key

- 1.1 Obtain the RPC\_URL for Sepolia from Morpheus, Alkemy's website, or Infura.
- 1.2 Obtain a free Etherscan API Key from the Etherscan website.

#### Step 2: Update Configuration Files

- 2.1 Open your **.env** file and modify the values as follows:

```
RPC_URL="https://eth-sepolia.g.alchemy.com/v2/APIKEY"  
PRIVATE_KEY="00000...000"
```

```
API_KEY="APIKEYFROMETHERSCAN"
```

Ensure the private key corresponds to the deployer account on Sepolia. You can use any account created with Metamask, and acquire testnet ETH from a faucet like Alkemy faucet.

2.2 Update the `chainID` in your `.env` file from 1303 to 11155111, then the `PRIVATE_KEY` and `RPC_URL`.

```
# This is the URL of the Ethereum RPC provider
RPC_URL="https://sepoliaaccess-20885.morpheuslabs.io/XXXXXXX"

# This is a private key for signing transactions
PRIVATE_KEY="your_private_key_here"

# This is the chain ID for the Sepolia Ethereum network
CHAIN_ID=11155111

# This is the address of a smart contract
CONTRACT_ADDRESS='0x1234567890abcdef'
```

2.3 Then change the network name "poa" to "sepolia" in your `hardhat.config.js`.

```
require("@nomicfoundation/hardhat-toolbox");
require("dotenv").config();

/** @type import('hardhat/config').HardhatUserConfig */
module.exports = {
  solidity: "0.8.22",
  networks: {
    // Add your network configuration here
    sepolia: {
      url: process.env.RPC_URL, // RPC URL of your network
      chainId: parseInt(process.env.CHAIN_ID), // Chain ID of your network
      accounts: [process.env.PRIVATE_KEY], // Array of private keys to use
      with this network
    },
  },
};
```

---

## Step 3: Redeploy the Contract on Sepolia

3.1 Navigate to the hardhat folder in your terminal.

3.2 Run the following command to redeploy the contract on Sepolia:

```
npx hardhat run scripts/deploy.js --network sepolia
```



## Step 4: Update Contract Address in front-end

4.1 Once the deployment is complete, locate the CampaignCreator contract address.

4.2 Copy the contract address and update the variable in `front-end/public/interact-contract.js` as follows:

```
const contractAddress = "campaignCreatorcontractaddress";
```

## Step 5: Restart the Server

5.1 Start or restart your server using the following command:

```
npm run dev
```

## Step 6: Verification on Etherscan

6.1 If you have chosen to verify your contract on Etherscan, you have two methods available:

### Method 1: Using Hardhat

To verify your contract using Hardhat, follow these steps:

1. Navigate to your Hardhat directory in the terminal.
2. Run the following command, replacing `<campaignCreatorcontractaddress>` with the actual address of your deployed contract:

```
npx hardhat verify <campaignCreatorcontractaddress> --network sepolia
```

3. Review the response in the terminal to confirm the success or any output related to the verification process.
4. Check Etherscan to verify if the contract has been successfully verified.

### Method 2: Using Etherscan Interface

An alternative method is to use the Etherscan interface directly. Provide the following information to Etherscan:

- Contract Code
- Compiler Version
- ABI (Application Binary Interface) of the contract

This method involves interacting with the Etherscan website to manually input the required details for verification.

Choose the method that best fits your workflow or preference. Successful verification ensures transparency and allows users to explore transactions and events within the voting contract on Etherscan, providing detailed insights at each step of the election.

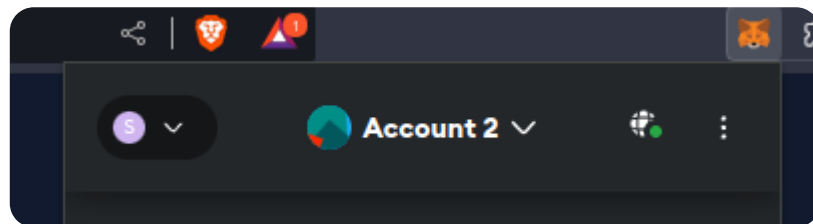
## Step 7: Test the New Setup

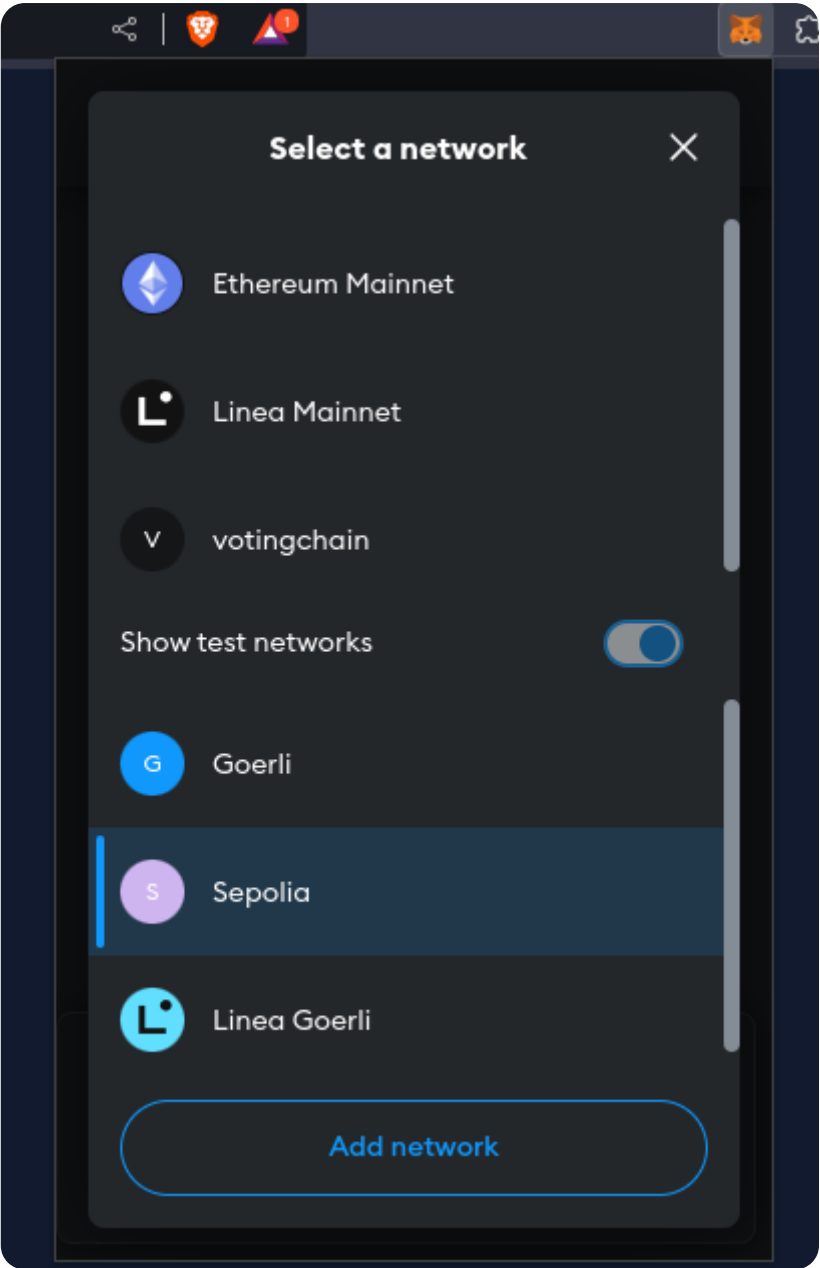
7.1 Retry launching a new campaign on this updated setup to ensure seamless functionality.

By following these steps, your dApp should now be successfully migrated to the Sepolia Testnet, utilizing the specified RPC\_URL and providing enhanced insights through Etherscan verification.

Explore the following Etherscan screenshots for a visual confirmation:

<https://sepolia.etherscan.io/>





Transactions								
Internal Transactions								
Token Transfers (ERC-20)								
Contract								
Events								
Latest 4 from a total of 4 transactions								
Download Page Data								
Transaction Hash	Method	Block	Age	From	To	Value	Txn Fee	
0xd60a2af5dec...	Finalize R...	5958975	41 mins ago	0x65d49342...FCd2D36E0	0x0D3AE9E1...4b2801454	0 ETH	0.00011221	
0x56e84b74d3...	Approve R...	5958973	41 mins ago	0x65d49342...FCd2D36E0	0x0D3AE9E1...4b2801454	0 ETH	0.00017689	
0x7e487376e2...	Create Re...	5958969	42 mins ago	0x65d49342...FCd2D36E0	0x0D3AE9E1...4b2801454	0 ETH	0.00029565	
0xc6fd1e5b06...	Contribute	5958963	43 mins ago	0x65d49342...FCd2D36E0	0x0D3AE9E1...4b2801454	0.01 ETH	0.00016976	

Transactions						
Internal Transactions						
Token Transfers (ERC-20)						
Contract						
Events						
Latest 2 internal transactions						
Download Page Data						
ADVANCED MODE: <input type="checkbox"/>						
Parent Transaction Hash	Block	Age	From	To	Value	
0xd60a2af5dec...	5958975	41 mins ago	0x0D3AE9E1...4b2801454	0x65d49342...FCd2D36E0	0.01 ETH	
0xcfc112c4c60...	5958946	47 mins ago	0xF1f2841B...6de268b7c	Contract Creation	0 ETH	

Transaction Action: **Call** **Finalize Request** Function by 0x65d49342...FCd2D36E0 on 0x0D3AE9E1...4b2801454

From: 0x65d493425fD6d67993FF90375375139FCd2D36E0

To: 0x0D3AE9E10d98B6eda12D645283387354b2801454

Transfer 0.010000000000000001 ETH From 0x0D3AE9E1...4b2801454 To 0x65d49342...FCd2D36E0

---

## 4. Uploading Your dApp on Morpheus app library to share with community

For detailed steps on uploading your dApp, refer to the [documentation](#).

<https://docs.morpheuslabs.io/docs/submit-app-to-the-app-store>

