

Blockchain final project

InvoiceFund -Decentralized workin capital financing

Team:

Name: Daniyal Adilbekov

Group: SE-2435

Name: Bibifatima Bisesheva

Group: SE-2437

Name: Ataniyaz Mutigolla

Group: SE-2437

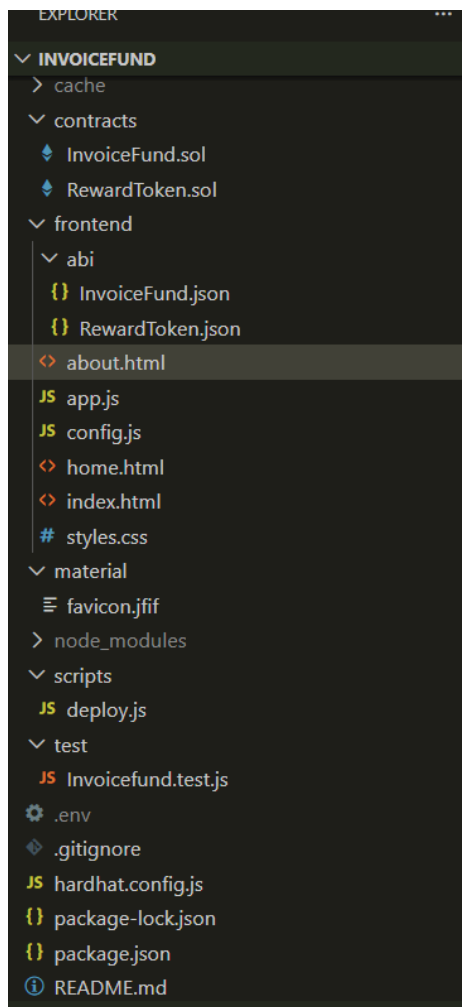
Project Overview

So, basically, InvoiceFund is a decentralized crowdfunding platform designed to provide early liquidity to businesses awaiting invoice payments. Instead of relying on banks or intermediaries, funding is raised through smart contracts. Participants receive ERC-20 reward tokens, which demonstrate proportional participation and accountability on the blockchain. The system operates entirely on the Ethereum Sepolia testnet using MetaMask.

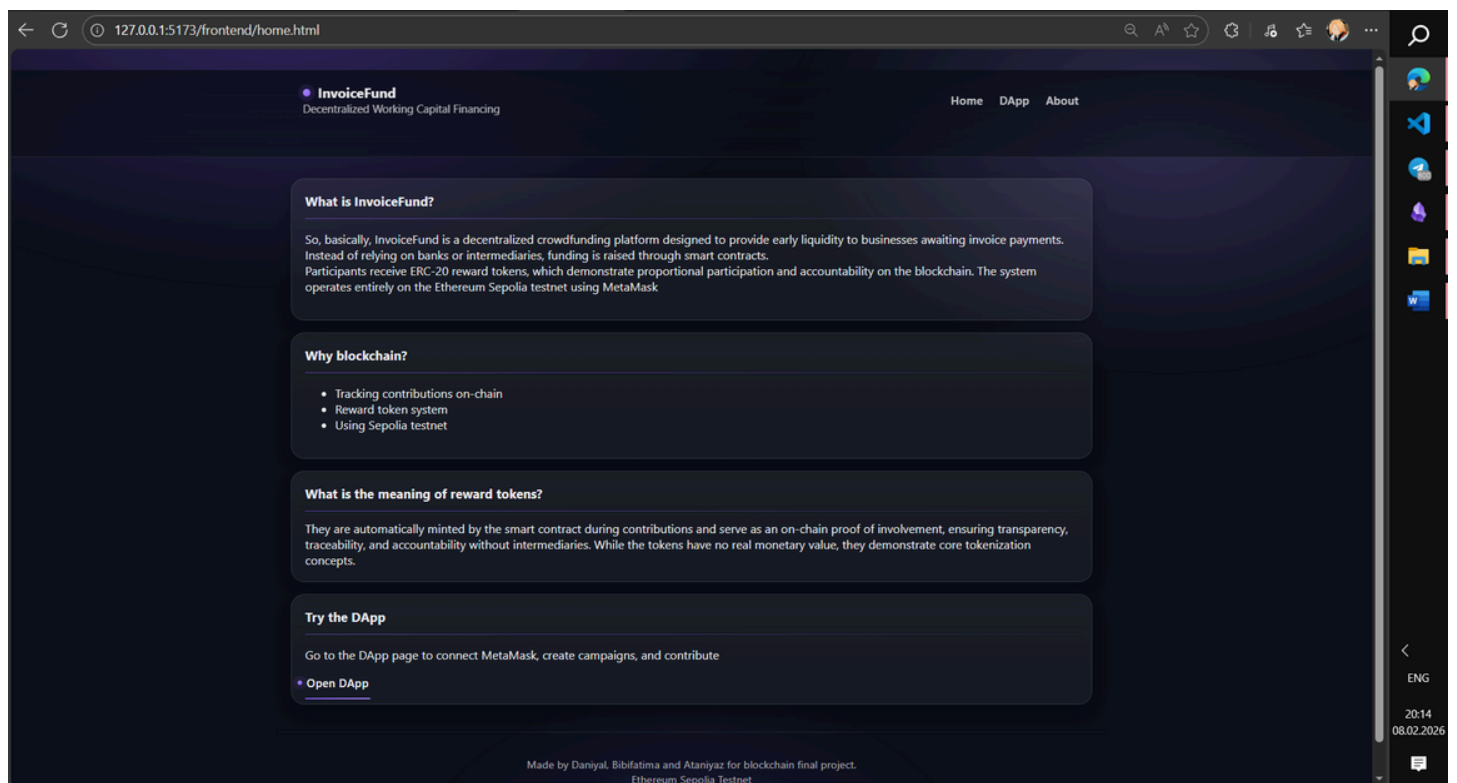
Aand This project was developed as a final project and operates exclusively on an Ethereum test network using free test tokens.

This project was developed as a blockchain final project

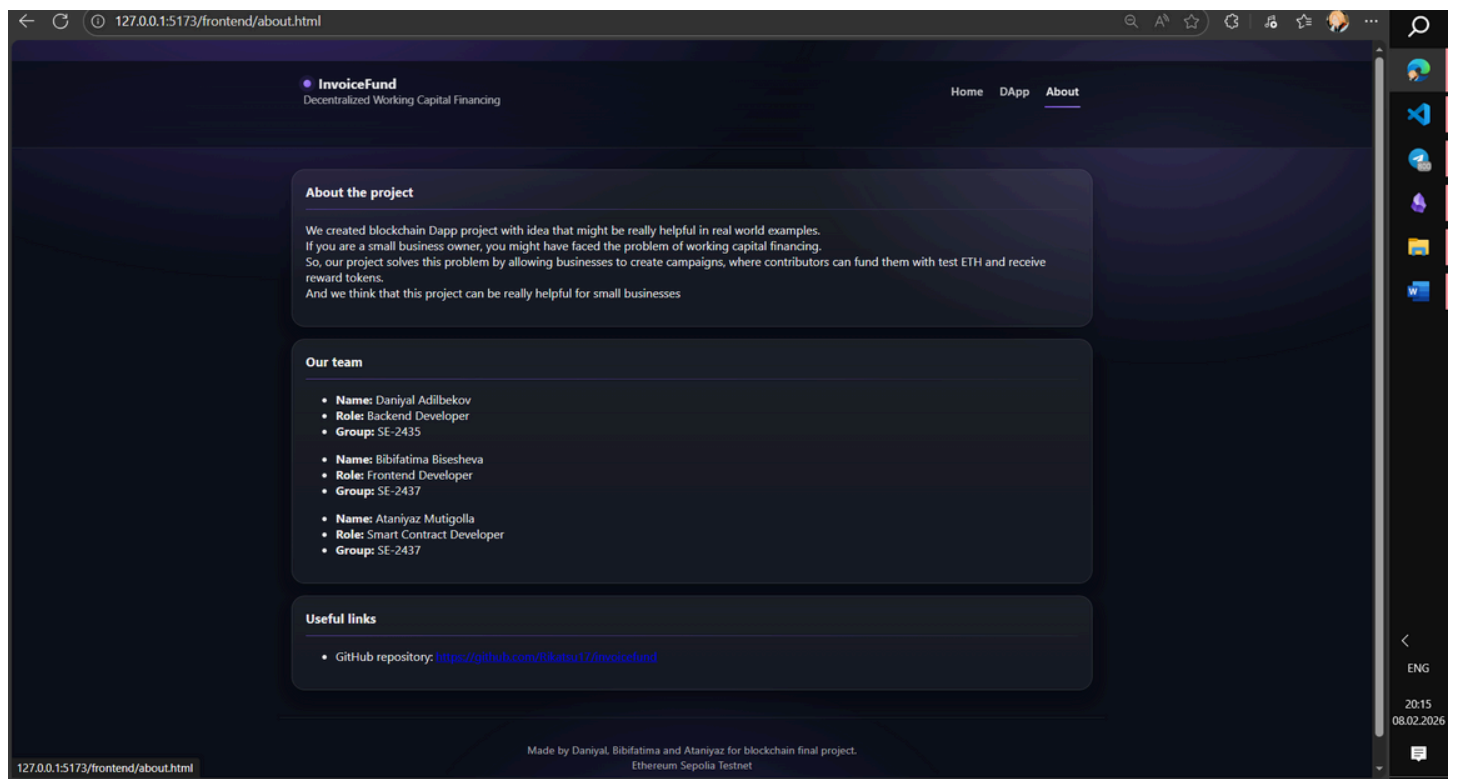
Project architecture



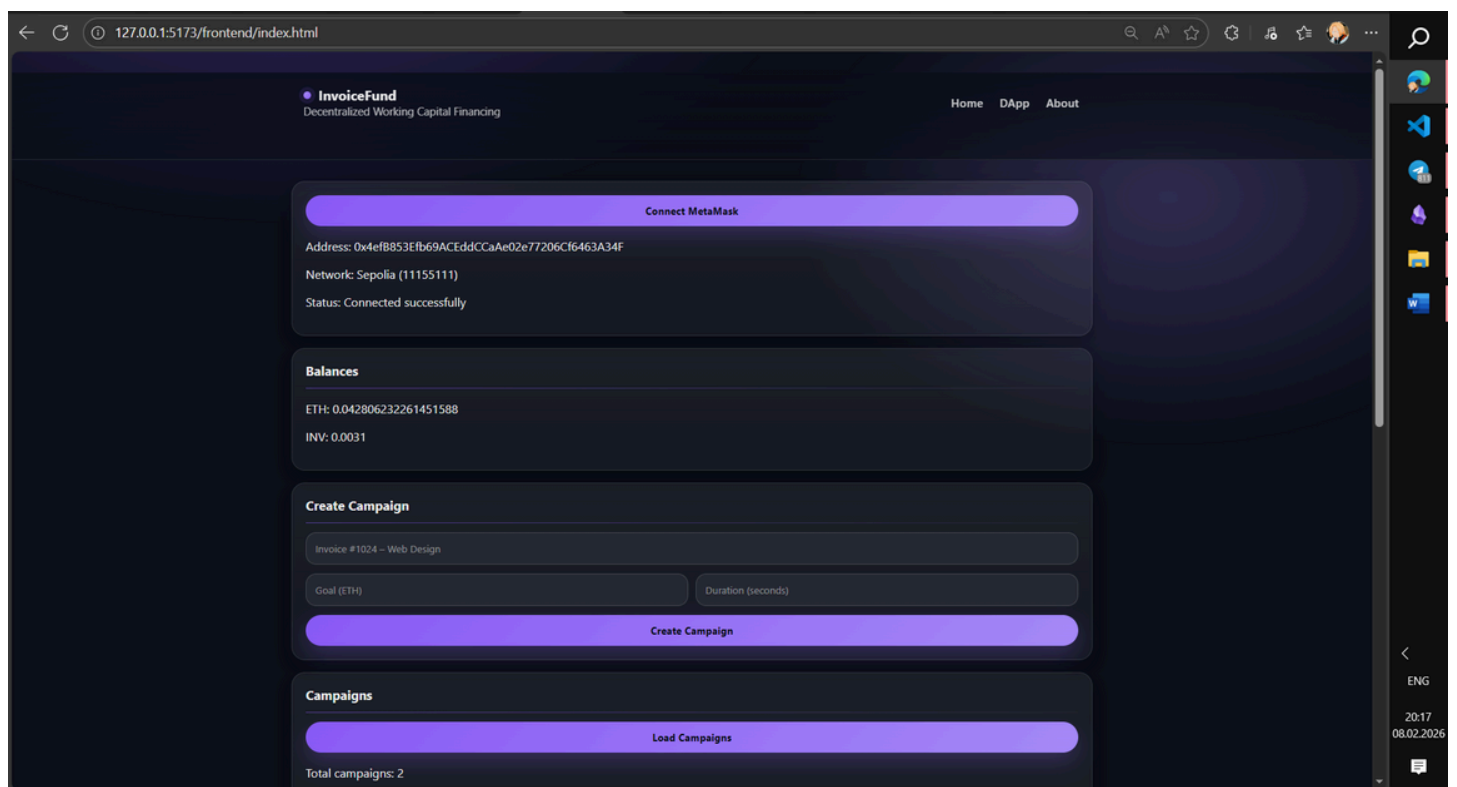
Home.html where is main information about project



About.html where you can see information about creators and links



And there is main file index.html where are functions like create campaign, list of campaigns, contribute function, and finalizing any campaign.



Campaigns

Load Campaigns

Total campaigns: 2

ID 0

Title: invoice 1025

Owner: 0x4efB853Efb69ACEdCCaAe02e77206Cf6463A34F

Goal: 0.0003 ETH

Raised: 0.00003 ETH

Deadline: 08.02.2026, 14:58:36

Finalized: true

ID 1

Title: invoice 666

Owner: 0x4efB853Efb69ACEdCCaAe02e77206Cf6463A34F

Goal: 0.0003 ETH

Raised: 0.000001 ETH

Deadline: 08.02.2026, 16:15:24

Finalized: false

Contribute

Contribute

Campaign ID

Amount (ETH)

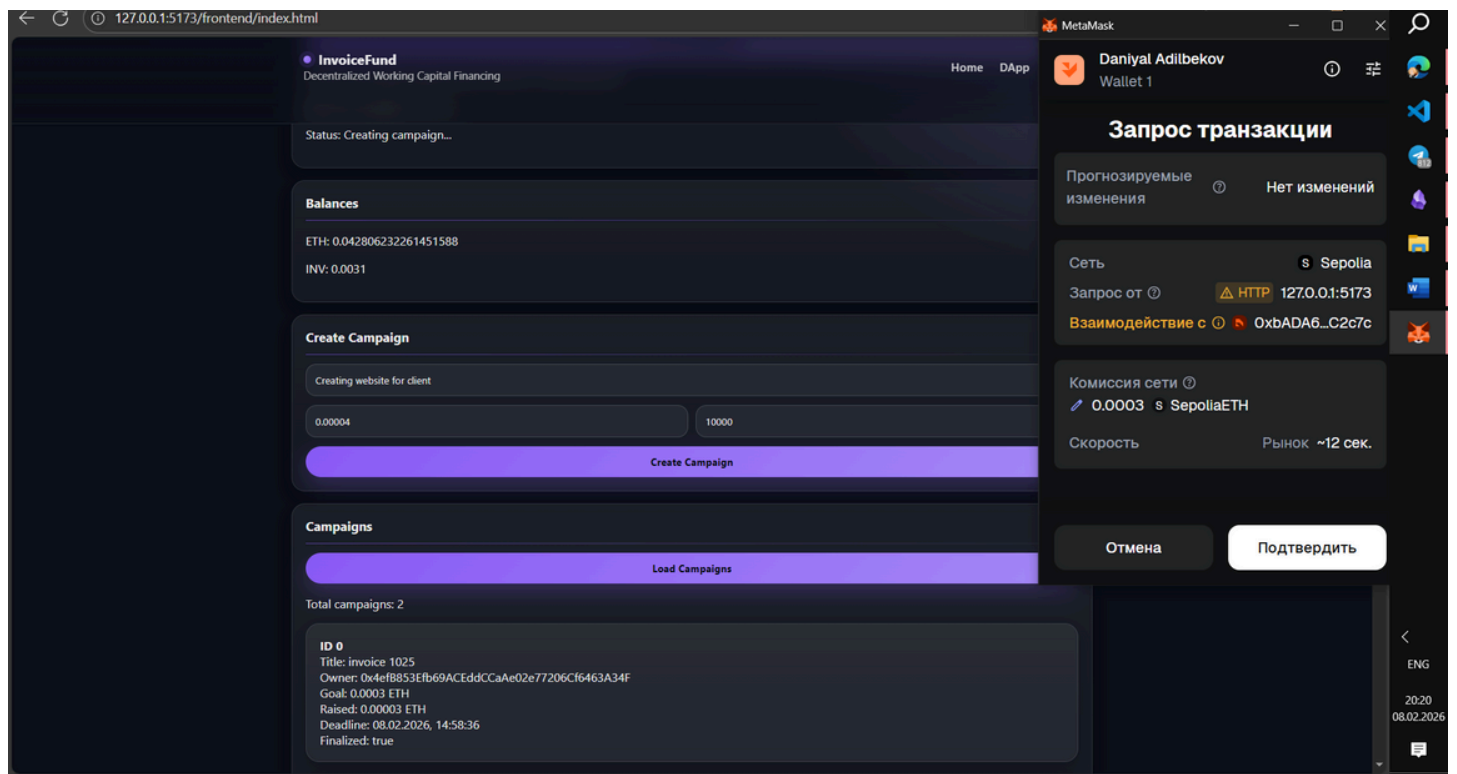
Contribute

Finalize

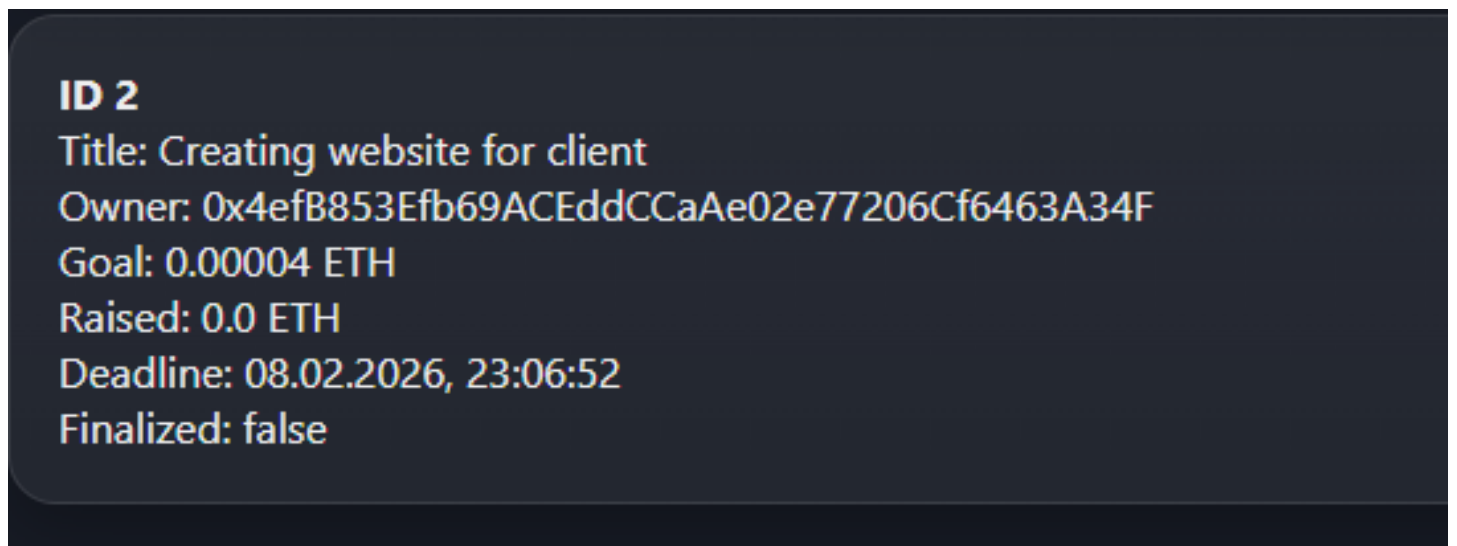
Campaign ID

Finalize Campaign

How it works



There is created campaign with full information



There is creating campaing in metamask

Daniyal Adilbekov ▾



Say hello to Bitcoin
Trade, manage, and more

Токены DeFi NFT Деятельность

Sepolia ▾

Feb 8, 2026



Create Campaign

Подтверждено



Finalize

Подтверждено



Contribute

Подтверждено



Create Campaign

Подтверждено



Contribute

Подтверждено



Create Campaign

Подтверждено

Create Campaign



Статус

Просмотр в проводнике блоков

Подтверждено

Скопировать ID транзакции

Из

Место назначения



Account 1



0xbADA6...C2c7c

Транзакция

Одноразовый код 13

Сумма -0 SepoliaETH

Лимит Газа (Единицы) 127337

Использовано Газа (Единицы) 126253

Базовая комиссия (Гвей) 1.001767412

Плата за приоритет (Гвей) 1.5

Итого платы за газ 0.000316 SepoliaETH

Макс. комиссия на газ 0.000000003 SepoliaETH

Итого 0.00031586 SepoliaETH

+ Журнал активности

-0 SepoliaETH

-0 SepoliaETH

-0 SepoliaETH

-0 SepoliaETH

-0.000001 SepoliaETH

-0.000001 SepoliaETH

-0 SepoliaETH

-0 SepoliaETH

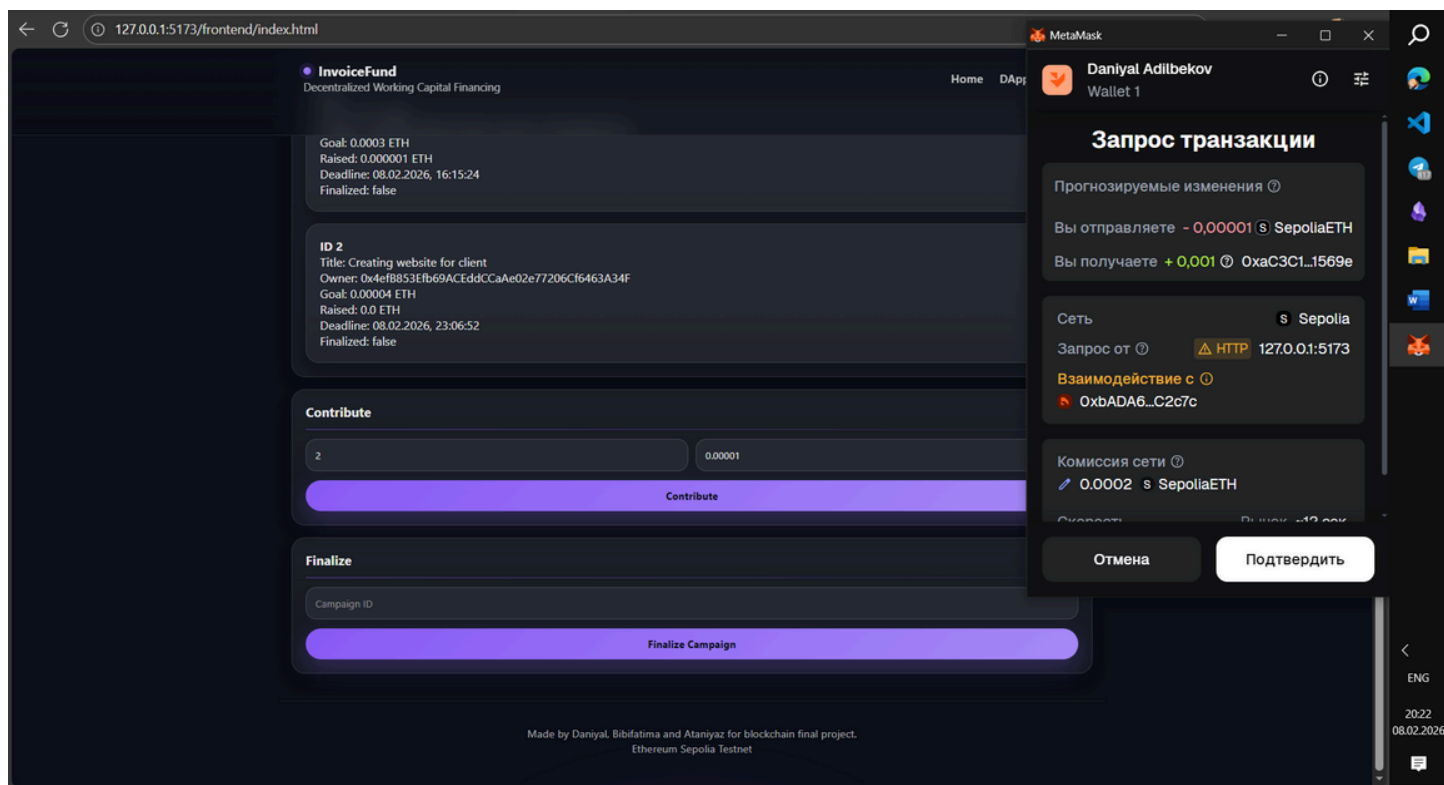
-0.00003 SepoliaETH

-0.00003 SepoliaETH

-0 SepoliaETH

-0 SepoliaETH

Now if people want to contribute to this campaign



Changes total of INV



And in campaign itself shows how much raised

ID 2

Title: Creating website for client

Owner: 0x4efB853Efb69ACEdDCCaAe02e77206Cf6463A34F

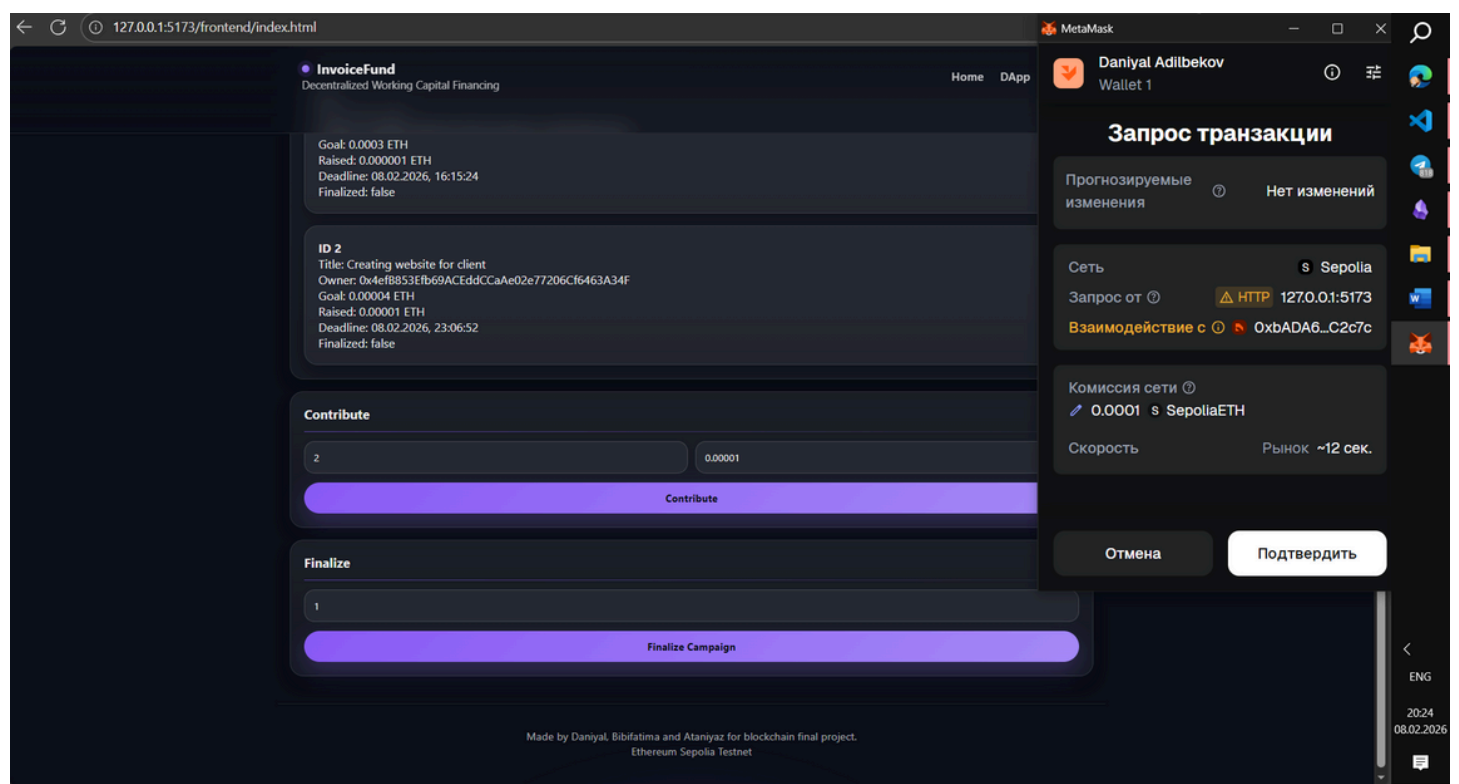
Goal: 0.00004 ETH

Raised: 0.00001 ETH

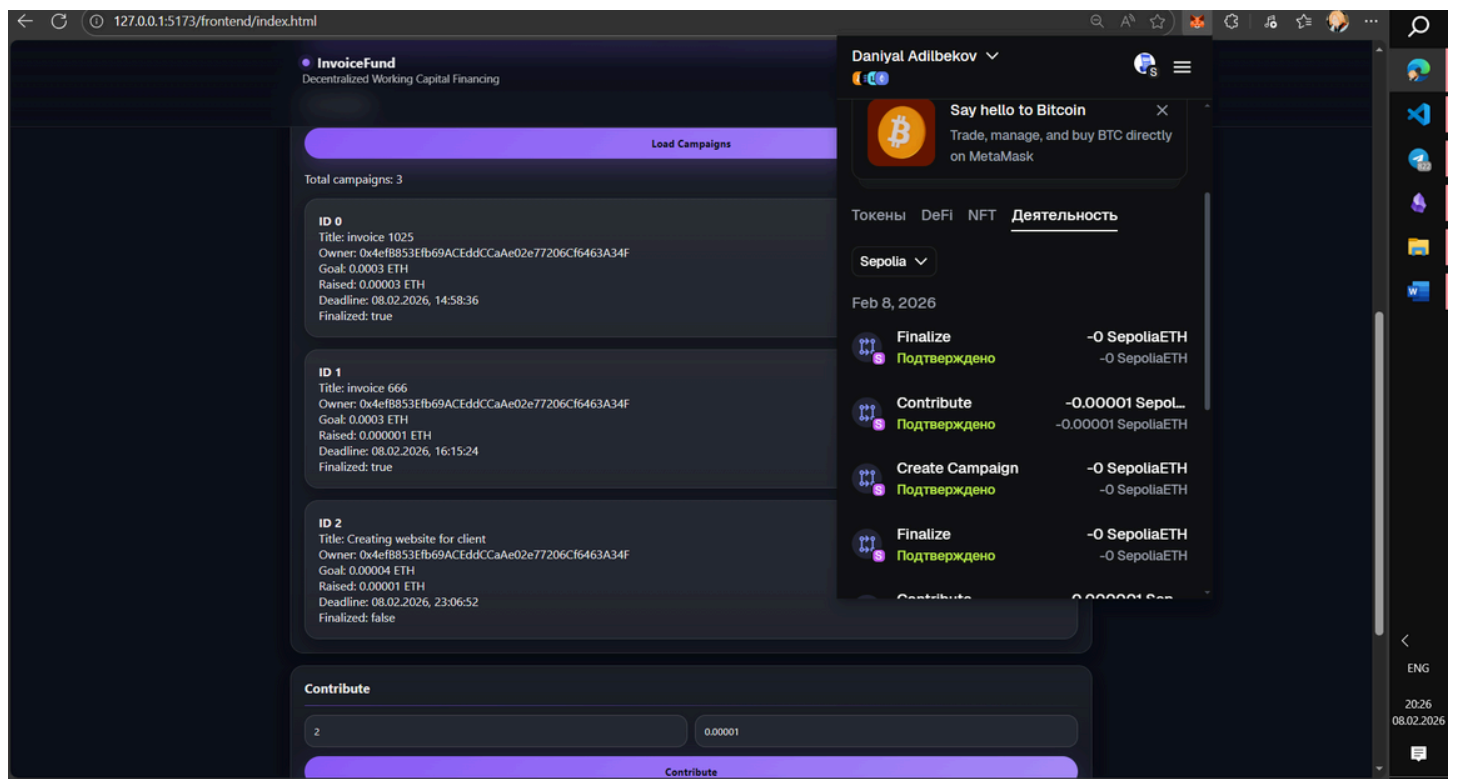
Deadline: 08.02.2026, 23:06:52

Finalized: false

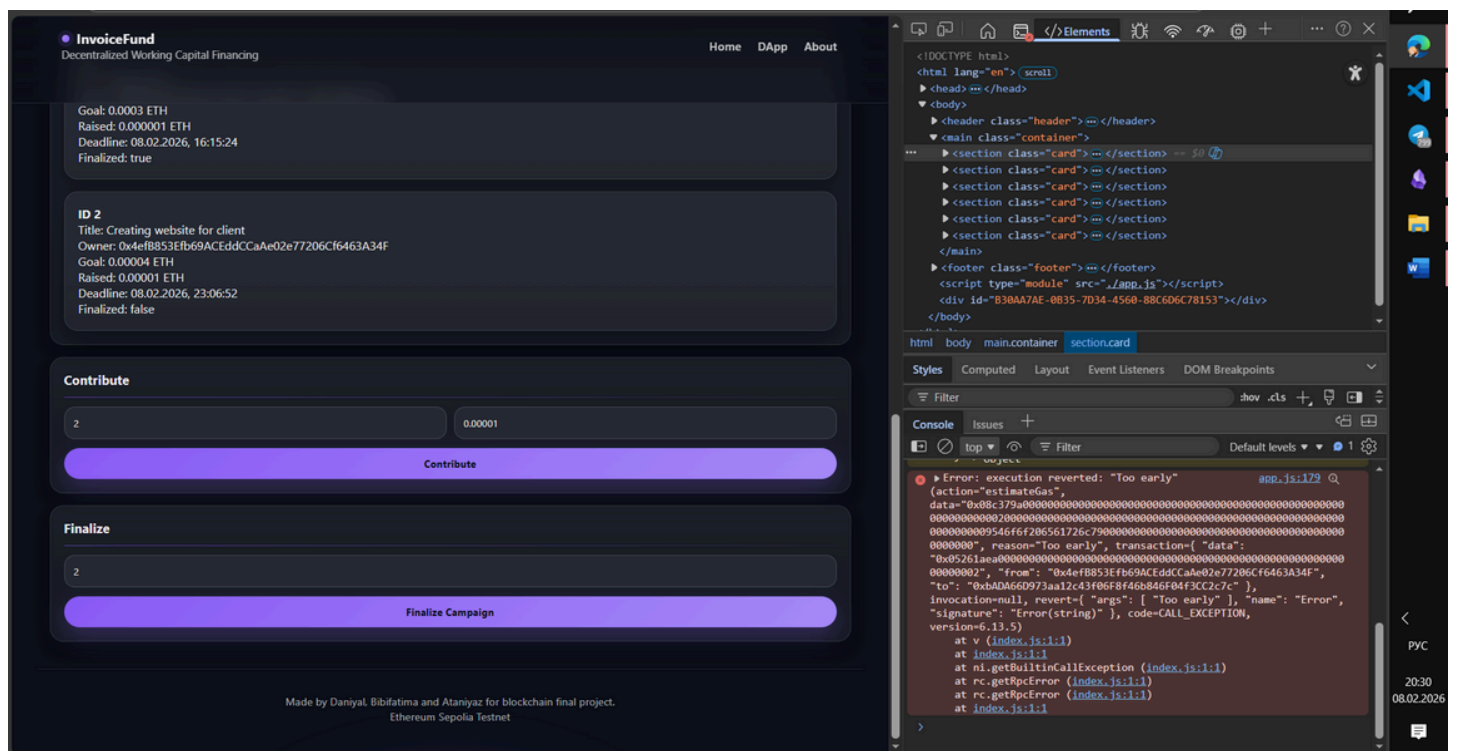
About finalizing. I can finalize the campaign only after deadline



There it shows in metamask and campaign with id finalized changes to true



if i try to finalize before deadline, it will show as error



Purpose of the Project

The main objective of this project is to demonstrate practical knowledge of:

Using Solidity for smart contracts

ERC-20 tokens integration

Client-side blockchain interaction using JavaScript

MetaMask wallet integration

Interaction with sepolia test networks

Basic dapp architecture

The application addresses the problem of delayed invoice payments by simulating decentralized invoice crowdfunding.

Technology Stack Solidity

Solidity

Hardhat Ethereum Sepolia Test Network

ERC-20 Token Standard

JavaScript

MetaMask

HTML & CSS

Smart Contracts

The project consists of two smart contracts:

1. InvoiceFund

Responsible for:

Creating crowdfunding campaigns

Accepting ETH contributions

Tracking individual and total contributions

Finalizing campaigns after the deadline

Minting reward tokens for contributors

2. RewardToken (ERC-20)

Custom ERC-20 token(inv)

Minted automatically during campaign participation

Has no real monetary value

Used strictly for educational demonstration

Reward formula

Reward = contributionWei × REWARD_RATE / 1e18

Frontend Features

The client-side application allows users to:

Connect a MetaMask wallet

Validate the active blockchain network (Sepolia)

View wallet address and balances (ETH & INV)

Create crowdfunding campaigns

Browse existing campaigns

Contribute test ETH to campaigns

Automatically receive ERC-20 reward tokens

Finalizing campaigns

All blockchain transactions are executed securely through MetaMask.

Frontend to blockchain interaction

The frontend interacts with the Ethereum Sepolia test network using app.js and the MetaMask wallet. MetaMask provides the connection between the web application and the blockchain.

After the user connects MetaMask, the application initializes an app BrowserProvider and obtains a signer representing the active wallet account. Smart contracts are accessed using their deployed addresses and ABI files.

Read-only operations (loading campaigns, checking balances) are performed via the provider, while state-changing operations (creating campaigns, contributing ETH, finalizing campaigns) are executed as blockchain transactions and require user confirmation through MetaMask.

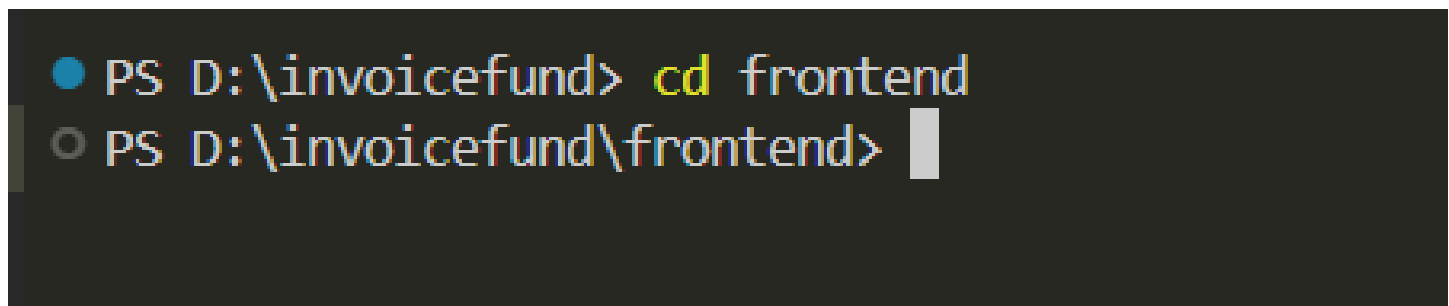
The frontend validates that the user is connected to the Sepolia test network and listens for MetaMask account or network changes to keep the application state consistent.

How to Run the Project

Steps

1)Navigate to the frontend directory:

```
cd frontend
```



```
PS D:\invoicefund> cd frontend
PS D:\invoicefund\frontend>
```

2)Start a local static server:

```
http-server -p 5173 -c-1
```

```

RewardToken ownership transferred to InvoiceFund success
PS D:\invoicefund> http-server -p 5173 -c-1
>>
Starting up http-server, serving ./

http-server version: 14.1.1

http-server settings:
CORS: disabled
Cache: -1 seconds
Connection Timeout: 120 seconds
Directory Listings: visible
AutoIndex: visible
Serve GZIP Files: false
Serve Brotli Files: false
Default File Extension: none

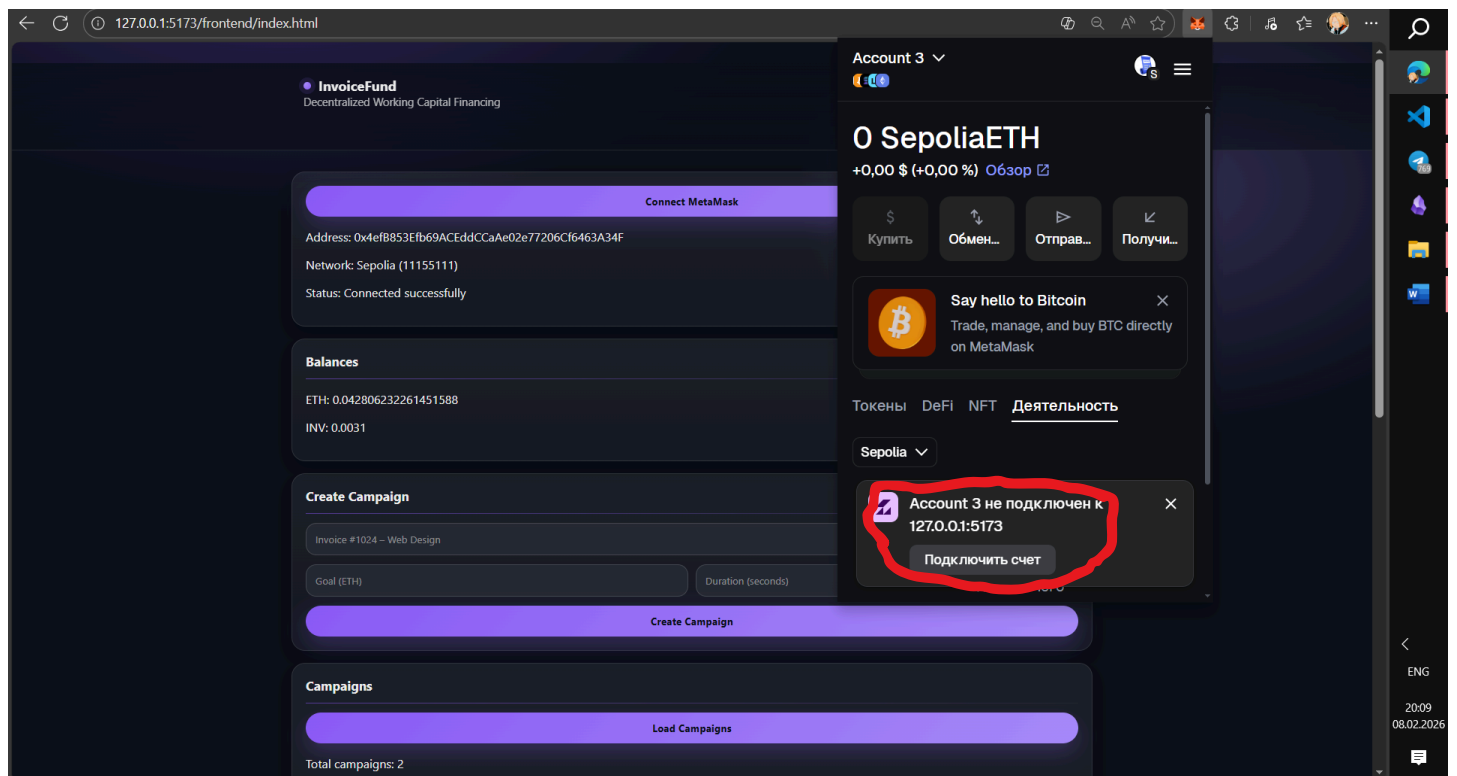
Available on:
  http://26.0.74.107:5173
  http://192.168.56.1:5173
  http://192.168.8.4:5173
  http://127.0.0.1:5173
Hit CTRL-C to stop the server

```

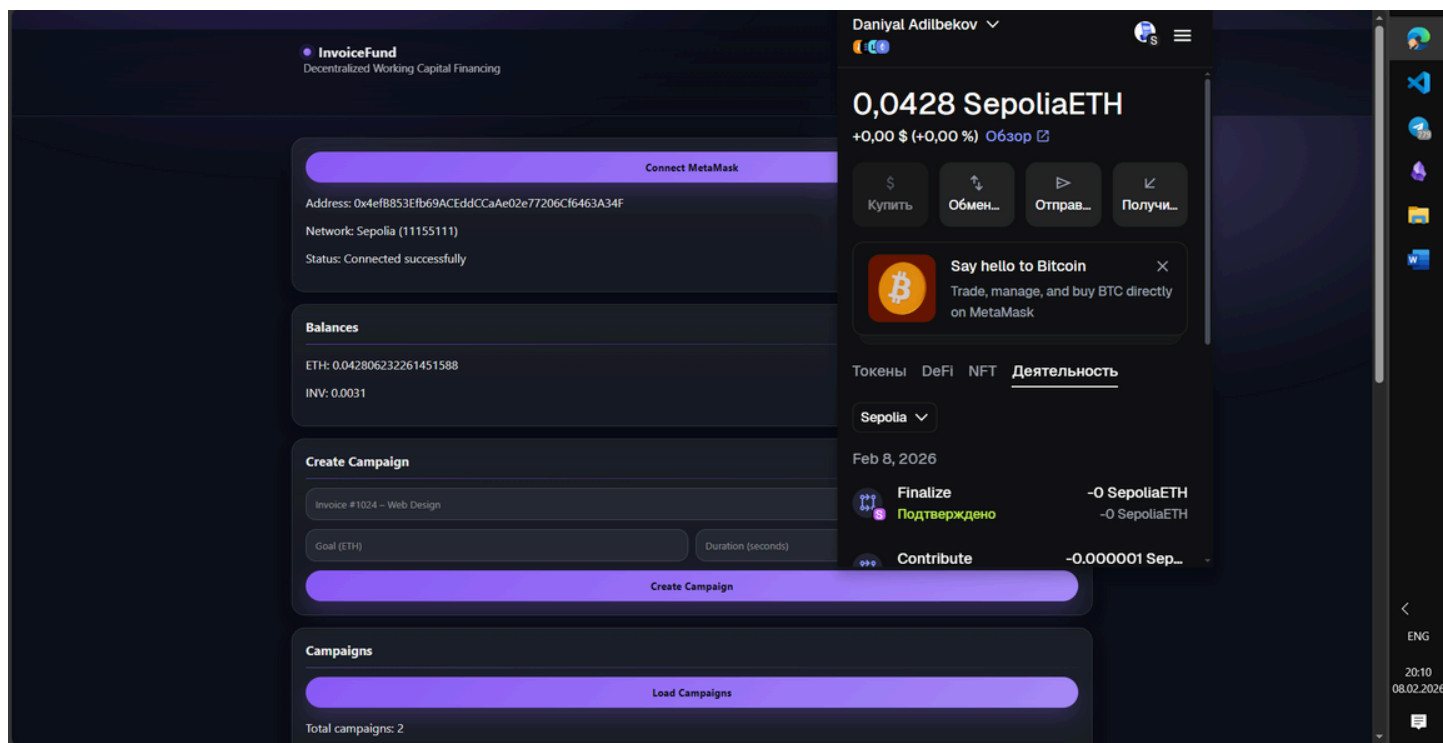
3) Open the application in a browser:

<http://127.0.0.1:5173>

4) Switch MetaMask to the Sepolia Test Network



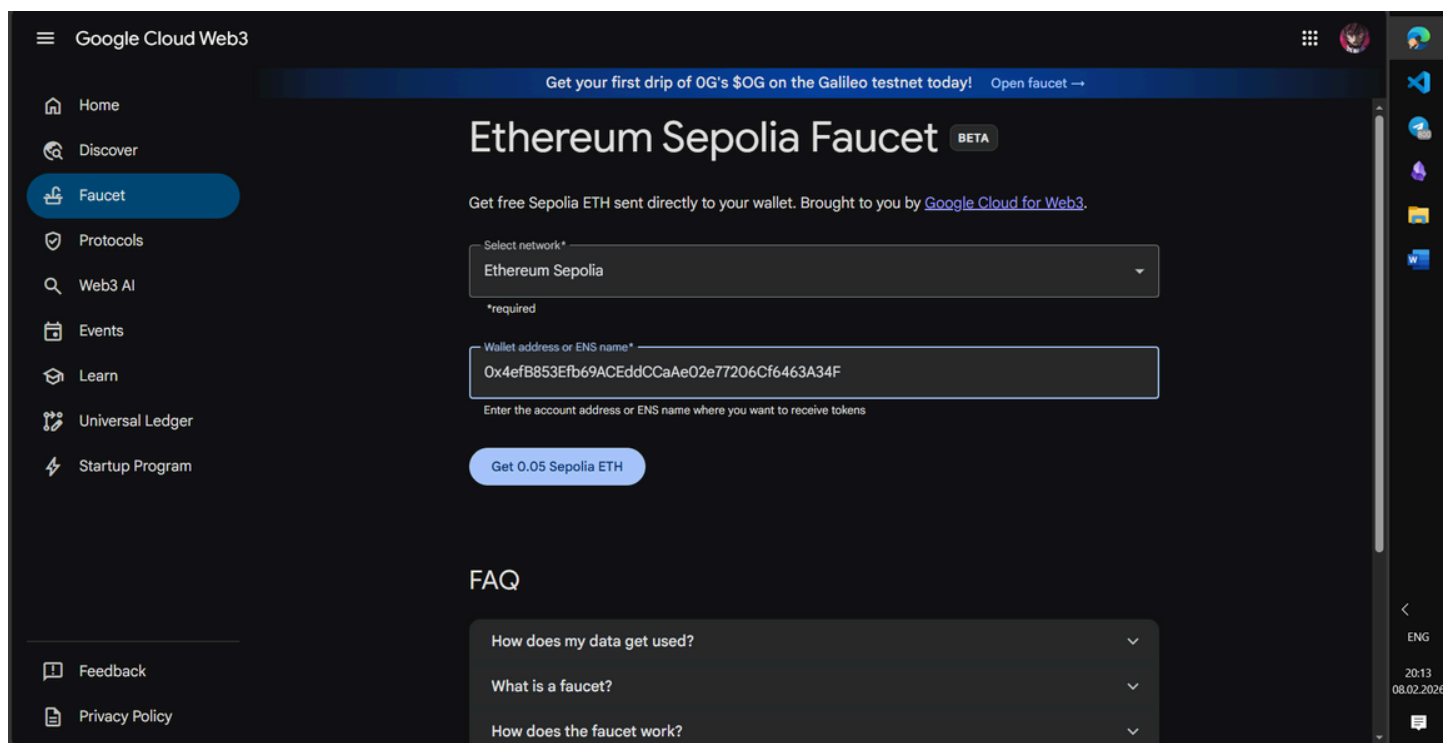
5) Connect MetaMask and interact with the DApp



How to get Sepolia test ethereum


Firstly u need to get your account adress, then go to websites where you can get test sepoliaETH

For example: <https://cloud.google.com/application/web3/faucet/ethereum/sepolia> deploy to sepolia



Testing

Automated unit tests are implemented using Hardhat to verify contract deployment, creating compaigns, contribution logic, reward token minting and etc. To run it write in terminal:
npx hardhat test

```
PS D:\invoicefund> npx hardhat test  
[dotenv@17.2.3] injecting env (2) from .env -- tip:  audit secrets and track compliance: https://dotenvx.com/ops
```

InvoiceFund DApp

- ✓ Should deploy and transfer RewardToken ownership to InvoiceFund (2575ms)
- ✓ Should create a campaign with correct parameters (62ms)
- ✓ Should accept contributions and track raised amount (46ms)
- ✓ Should mint reward tokens proportional to contribution (49ms)
- ✓ Should not allow contribute to non-existing campaign (213ms)
- ✓ Should not allow finalize before deadline (54ms)
- ✓ Should allow finalize after deadline (52ms)
- ✓ Should not allow finalize twice
- ✓ Should not allow contribute after finalize (45ms)

9 passing (3s)

```
PS D:\invoicefund> 
```

Conclusion

InvoiceFund demonstrates a complete decentralized crowdfunding workflow using Ethereum smart contracts, MetaMask integration, and a client-side DApp interface. The project highlights key blockchain concepts such as decentralization, transparency, tokenization, and secure user interaction on a test network.