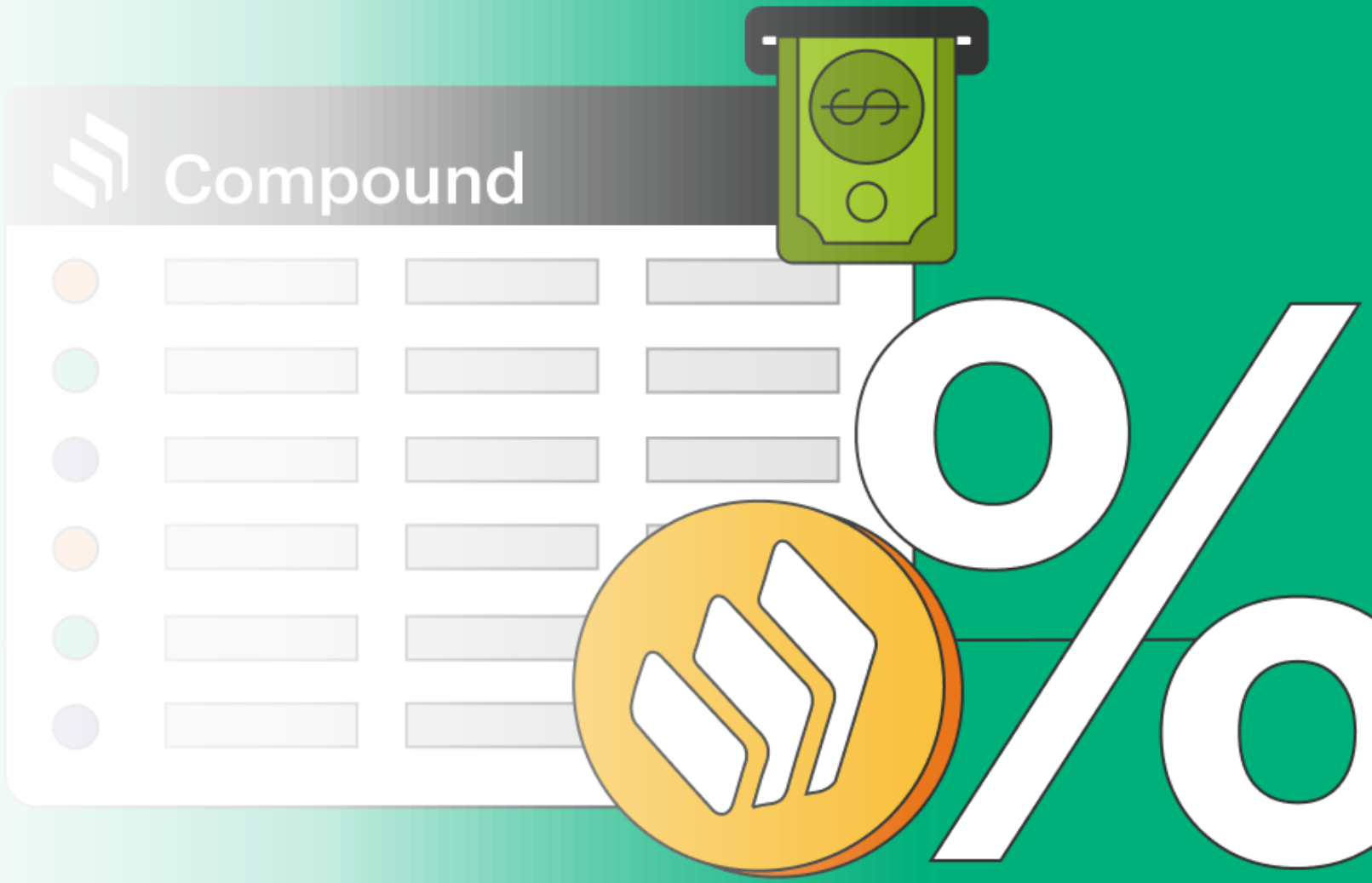
Deploy Compound Protocol



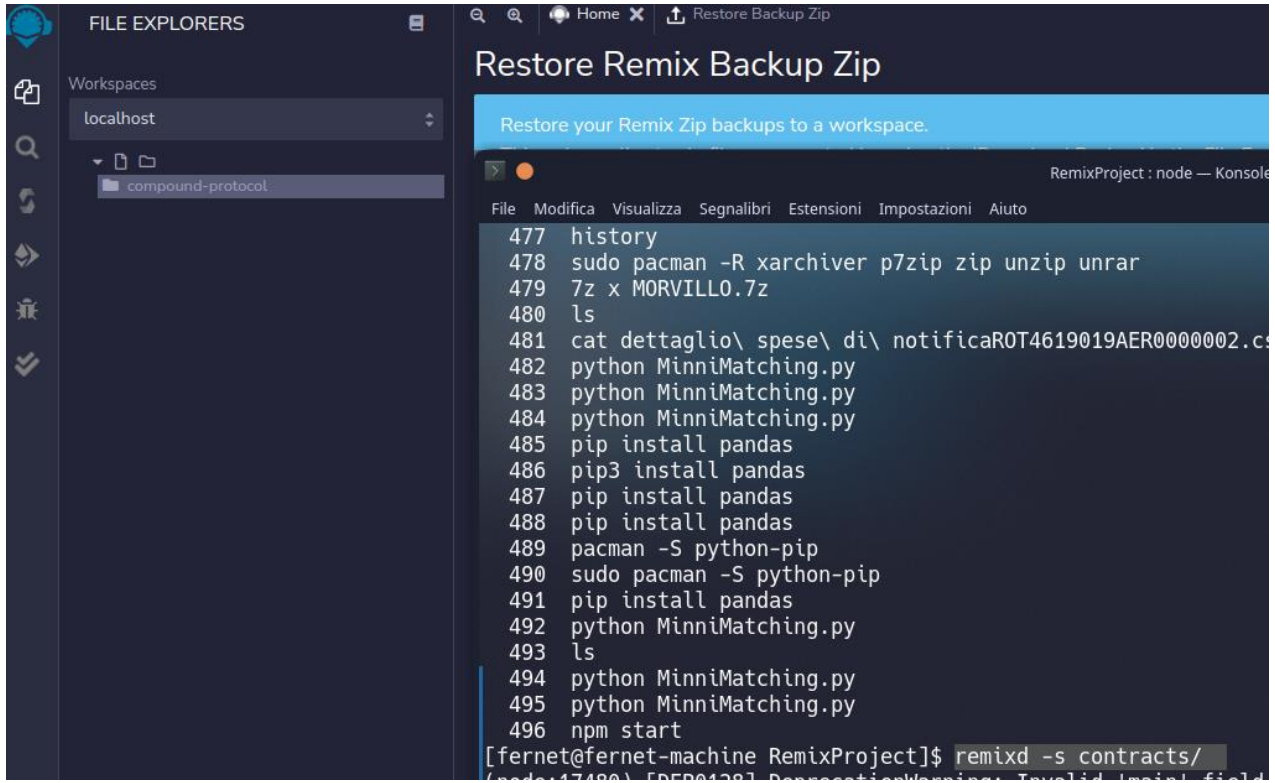
Corso di

FINANCIAL COMPUTING E TECNOLOGIE PER L'HIGH FREQUENCY TRADING

PIERLUIGI LIGUORI
FABIANO PRIORE
MARIKA PIA SALVATO

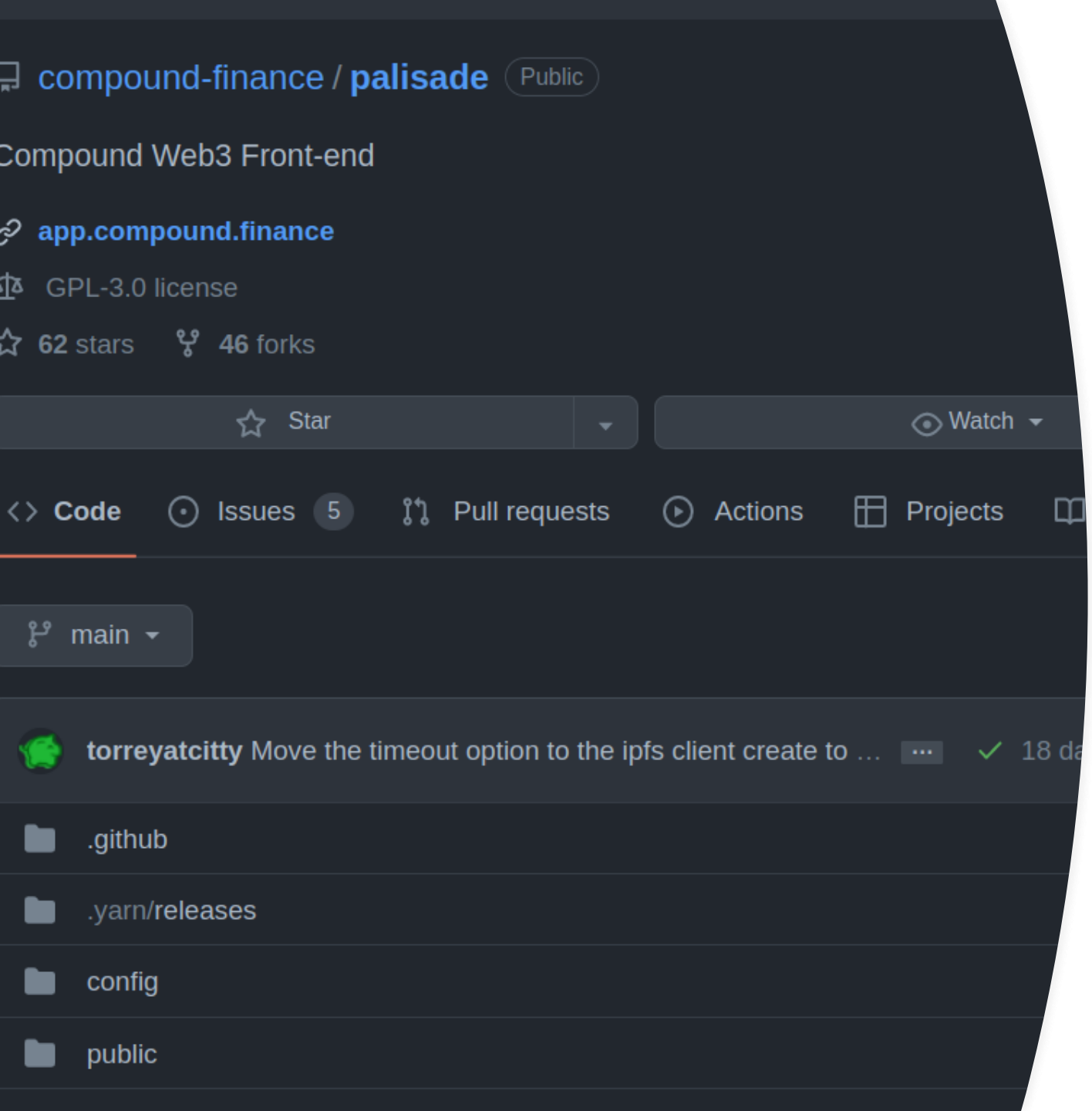
Repository Github del Protocollo

Collegamento in locale con Remix



Dopo aver scaricato il codice del protocollo, lanciare da terminale:

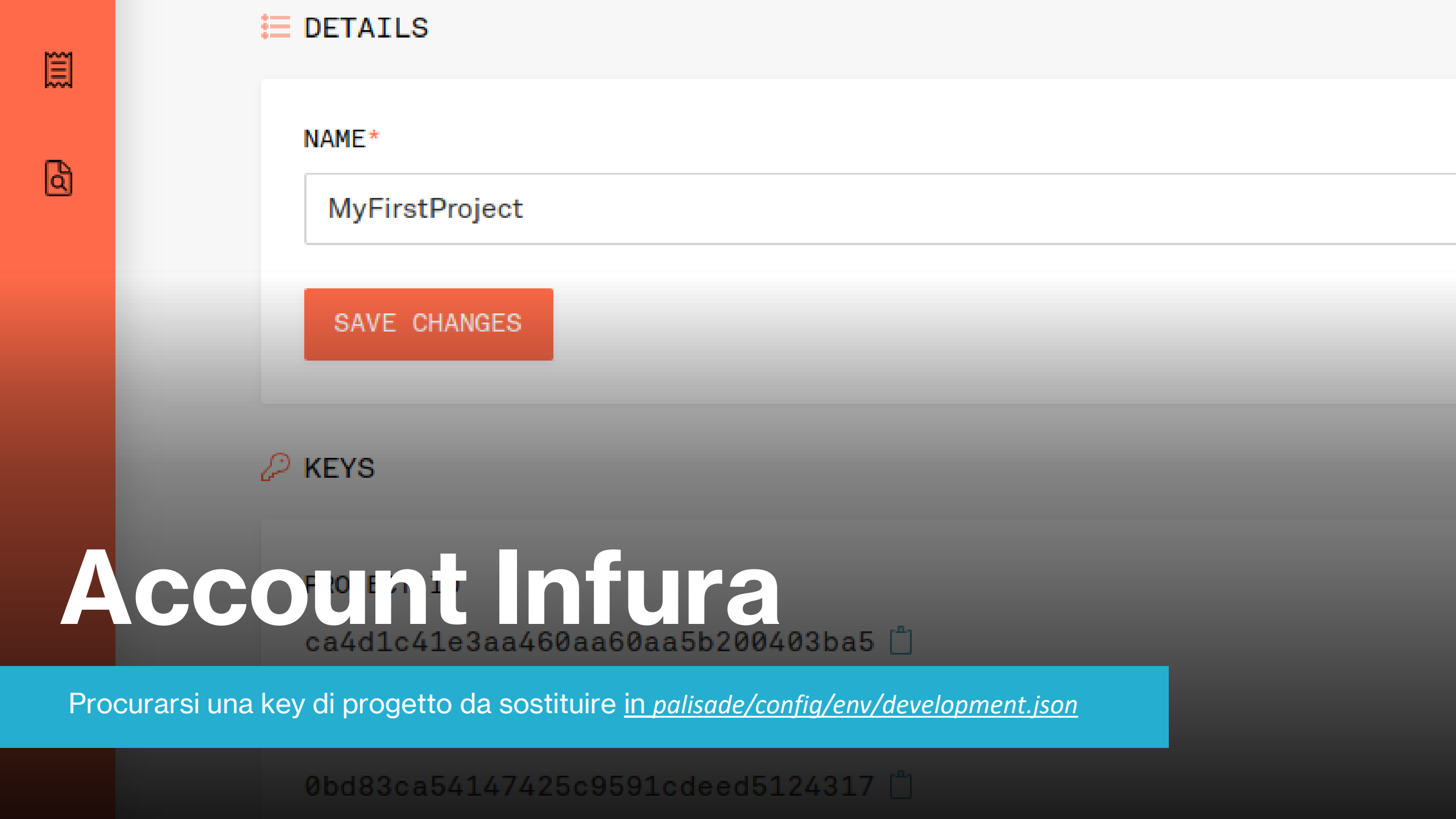
```
npm install -g remixd

remixd –s contracts/
```



Repository GitHub Frontend

Clonazione del repository in locale:

- git clone https://github.com/compound-finance/palisade.git
- cd palisade/
- yarn install
- yarn install --ignore-platform
- yarn watch-i18n
- yarn watch-css
- yarn start



Account Infura

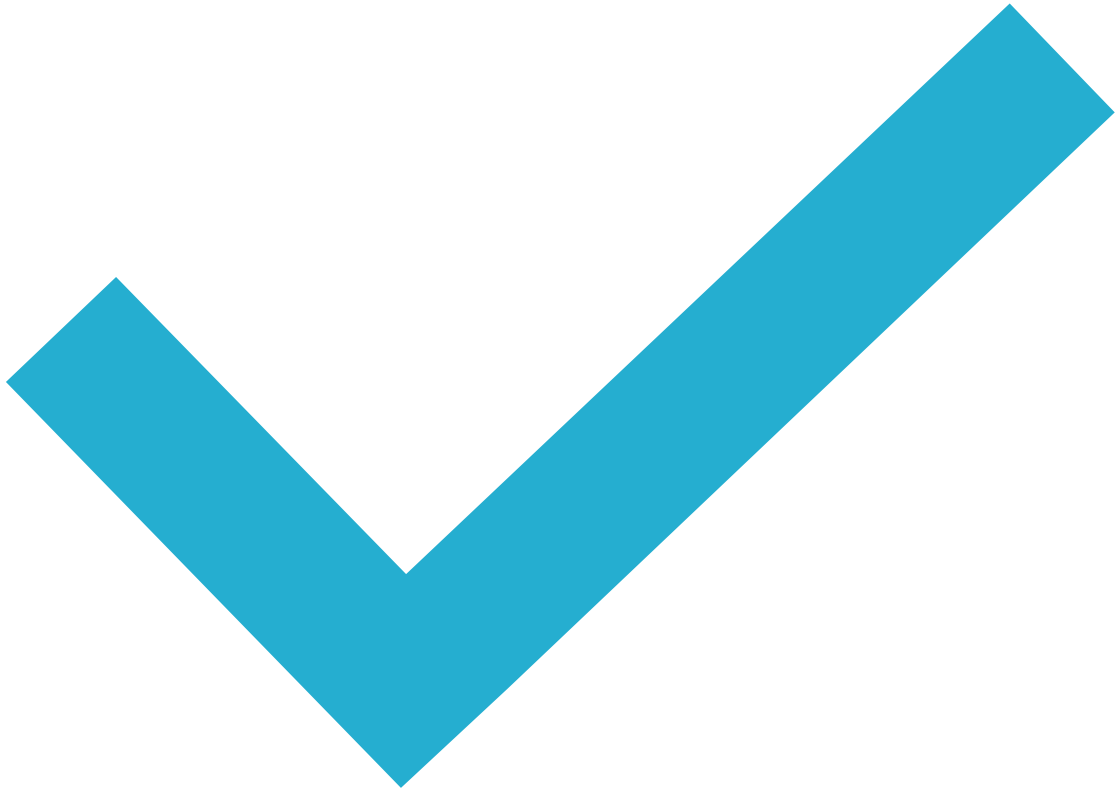
Procurarsi una key di progetto da sostituire in *palisade/config/env/development.json*

Note

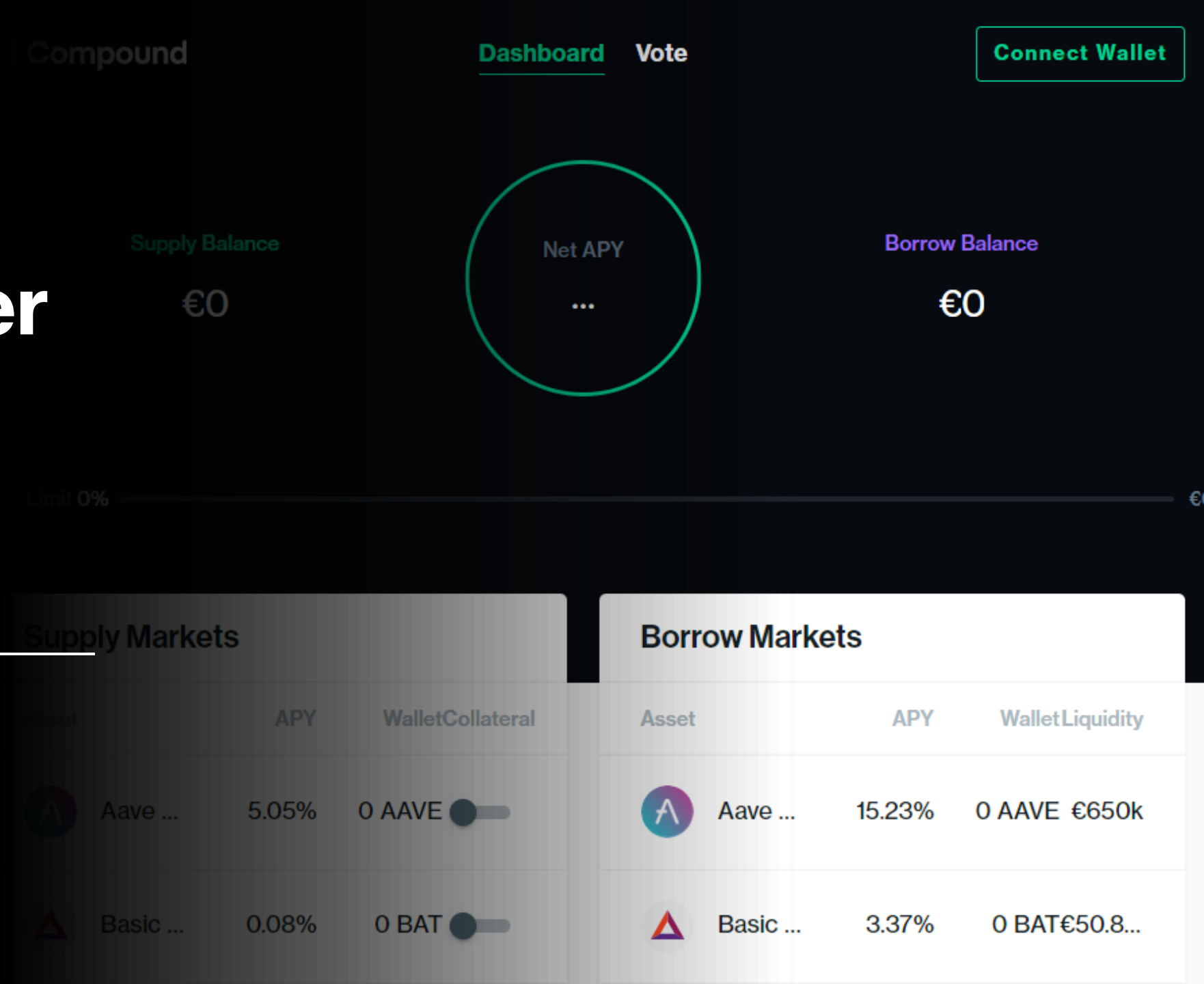
- Vengono rideployate le funzionalità nella dashboard di https://app.compound.finance/
- Non vengono trattati i contratti relativi alla governance
- La TestNet usata è Kovan
- Alcuni contratti che riguardano oracoli e tokens già presenti non vengono rimpiazzati
- Ogni contratto rideployato deve essere rimpiazzato nel file palisade/node_modules/compound-config/networks/kovan.json

Come avviare la nostra implementazione

- Sostituire il file kovan.json fornito in allegato
- Avviare il frontend con *npm start*



Istruzioni per deployare il compound protocol

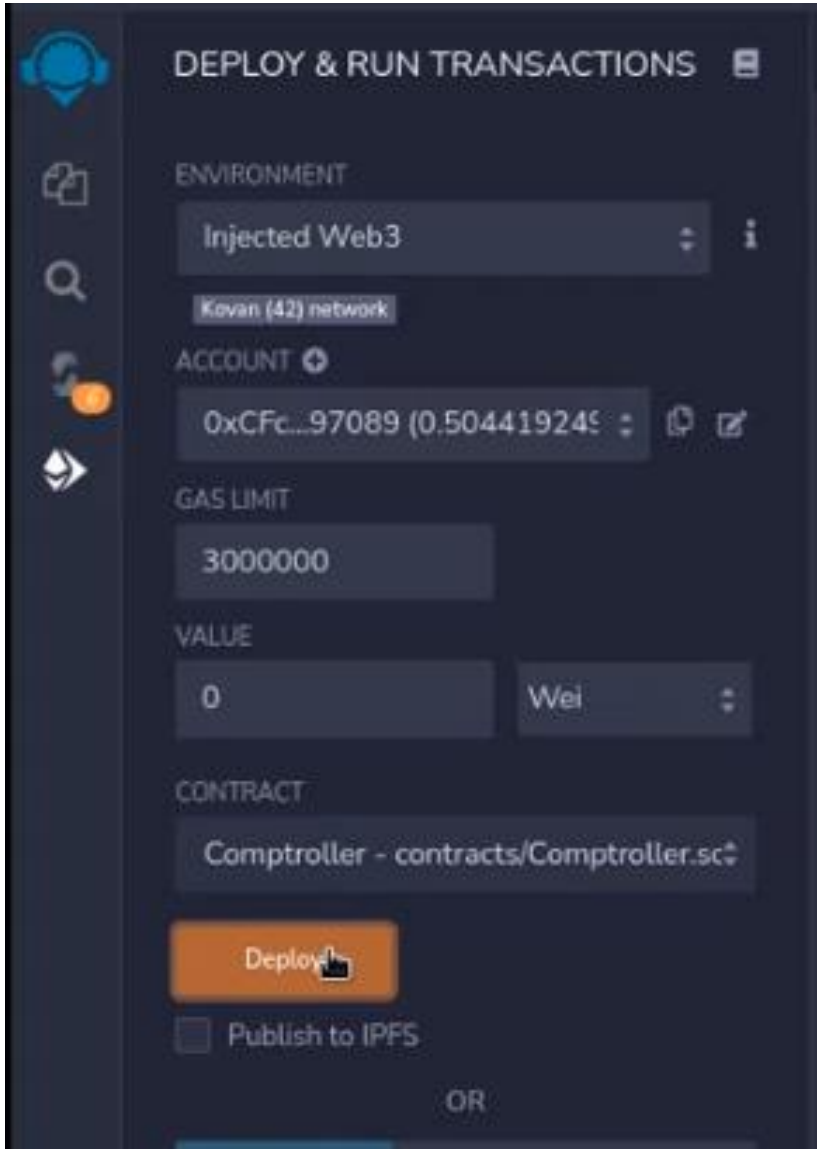


```
    /**
     * @notice Return the address of the COMP token
     * @return The address of COMP
     */
    function getCompAddress() virtual public view returns (addres
        return 0xc00e94Cb662C3520282E6f5717214004A7f26888;
    }
}
```

- Copiare l' indirizzo del Contratto **Comp** (ricavabile dal file *kovan.json corrente*)
- Incollarlo nella funzione getComAddress() nel codice di Comptroller.sol

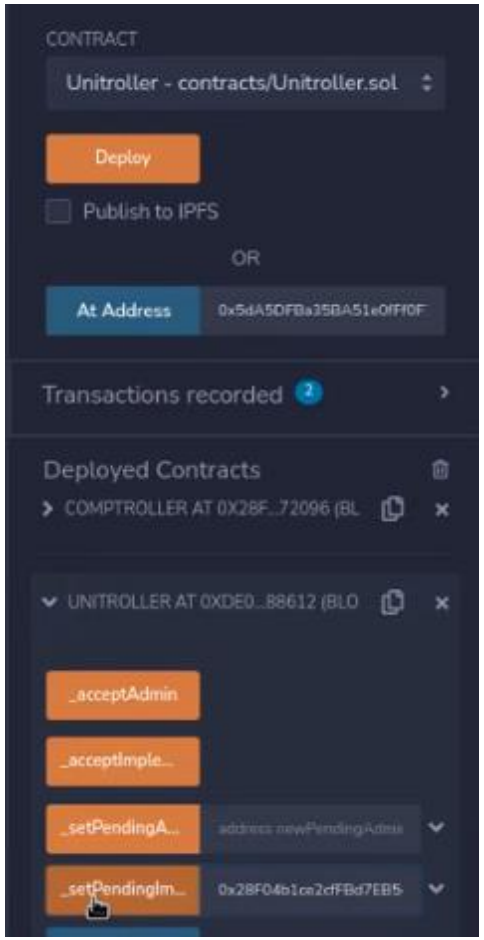
Deploy di Comptroller.sol

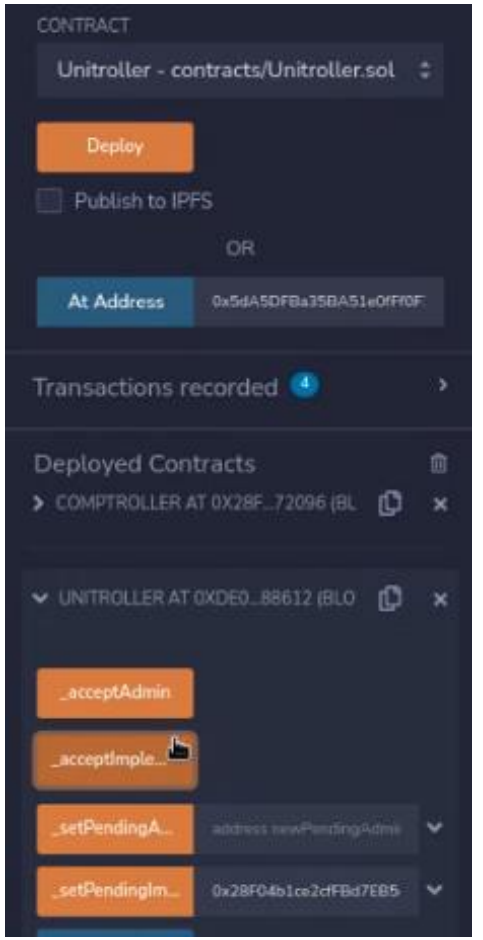
Da Remix, compilare Comptroller.sol e, dopo aver collegato *MetaMask*, deployare sulla TestNet pubblica

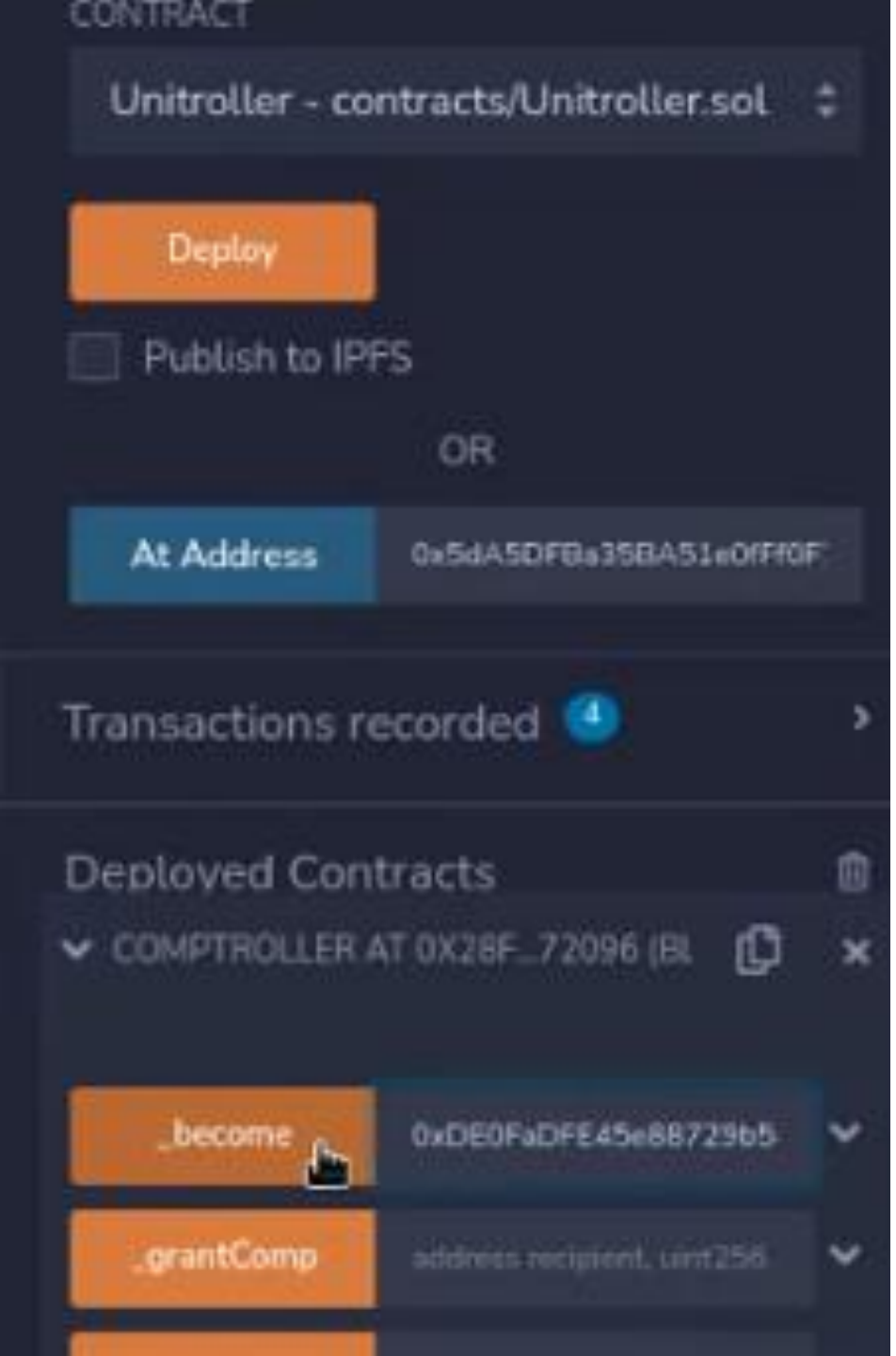


Unitroller.sol

- Fare il deploy di Unitroller.sol
- Chiamare la funzione _setPendingImplementation() inserendo come parametro l' indirizzo del Comptroller precedentemente creato
- Chiamare la funzione _acceptImplementation()

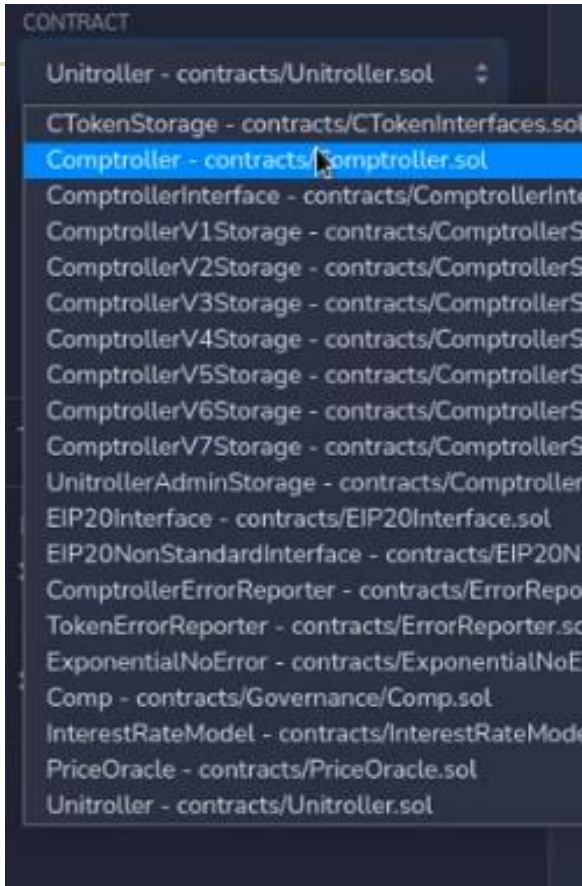






Unitroller.sol (II)

- Copiare l' indirizzo di Unitroller e darlo in input alla funzione _become() del Comptroller
- Reimportare l' Unitroller copiando il suo indirizzo sulla la voce 'At Address' ma questa volta mascherandolo con l' implementazione di un Comptroller

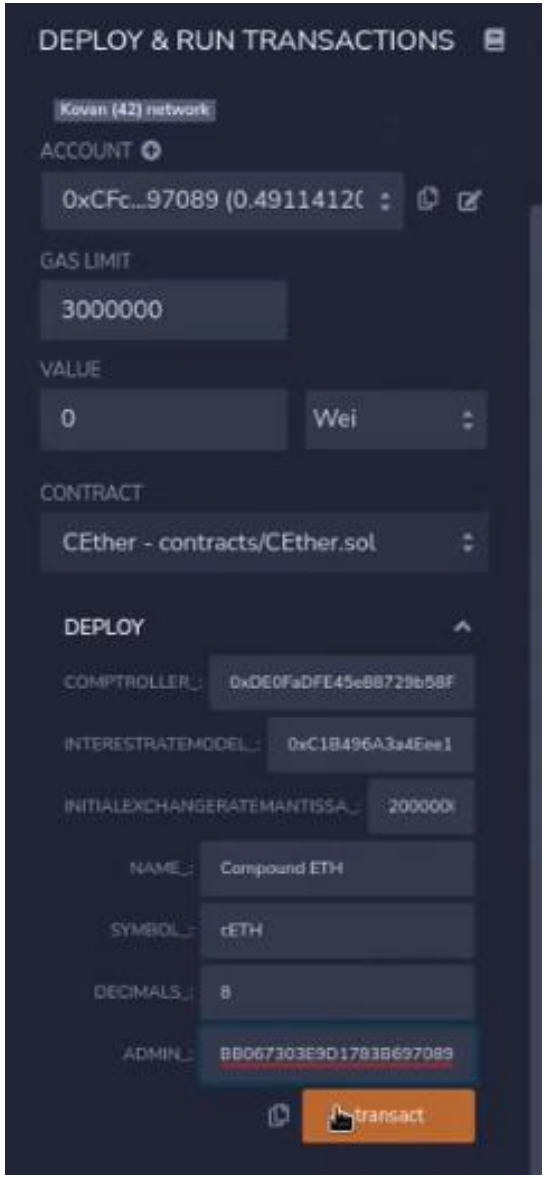


Oracolo

aggiungere oracolo (l' indirizzo si trova nel file *kovan.json*) a Unitroller (Comptroller) tramite _setPriceOracle()

Deploy CEther

- Inserisci l’ indirizzo del Comptroller creato
- Inserisci l’ indirizzo dell’ InterestRateModel esistente
- Specificare come Admin l’ indirizzo del proprio Wallet
- Riempire gli altri campi come in figura



Aggiungi cETH all’ Unitroller

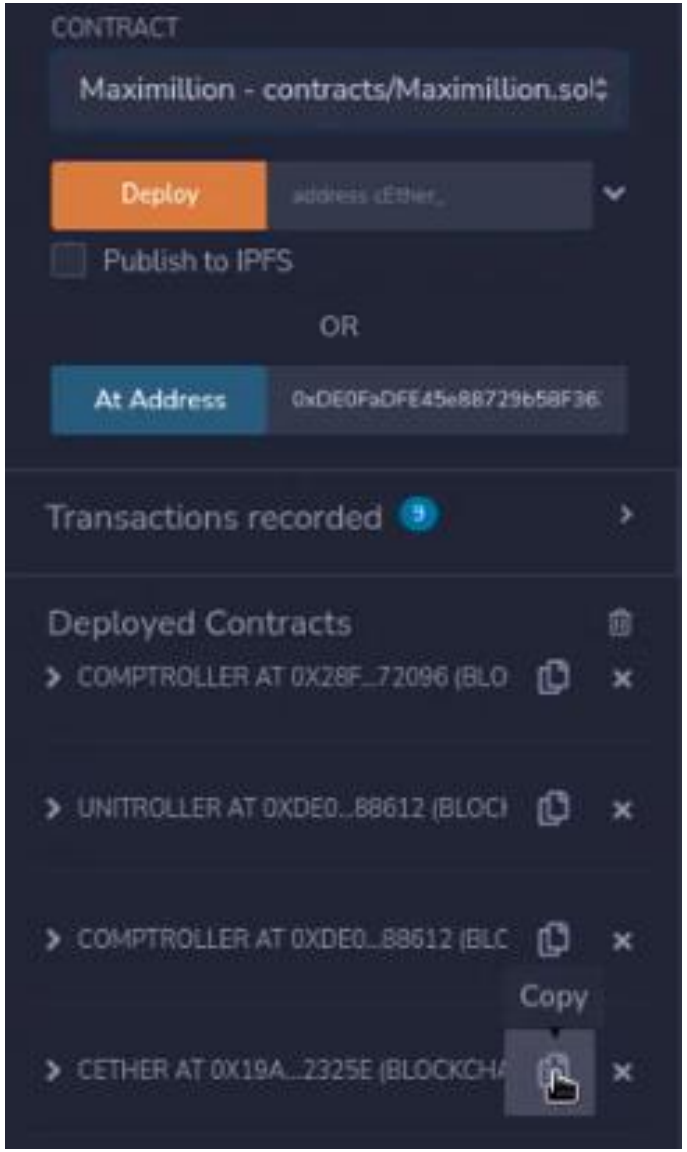
da Unitroller (Comptroller), chiama _supportMarket() per **cETH** (appena creato)

_setCollateralFactor()

- da Unitroller (Comptroller), chiama _setCollateralFactor() per cETH con valore 75*10^18

Maximillion

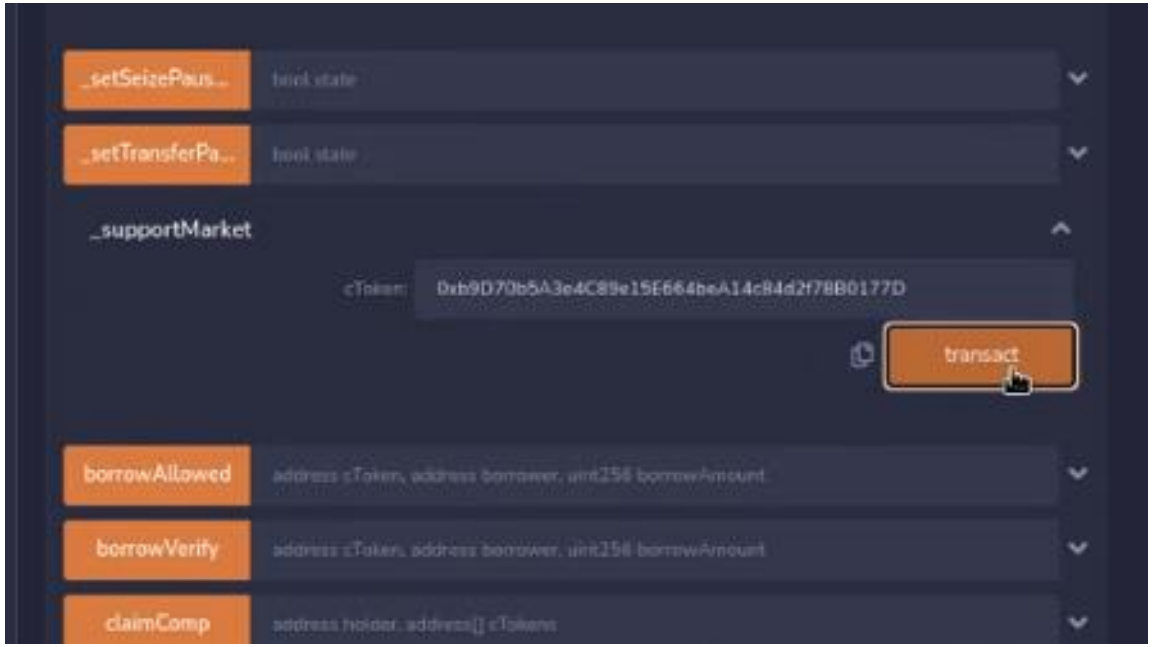
- Deployare un nuovo contratto da Maximillion.sol
- Come parametro, inserire l' indirizzo di CEther
- Maximillion è essenziale per la parte di repay e borrow

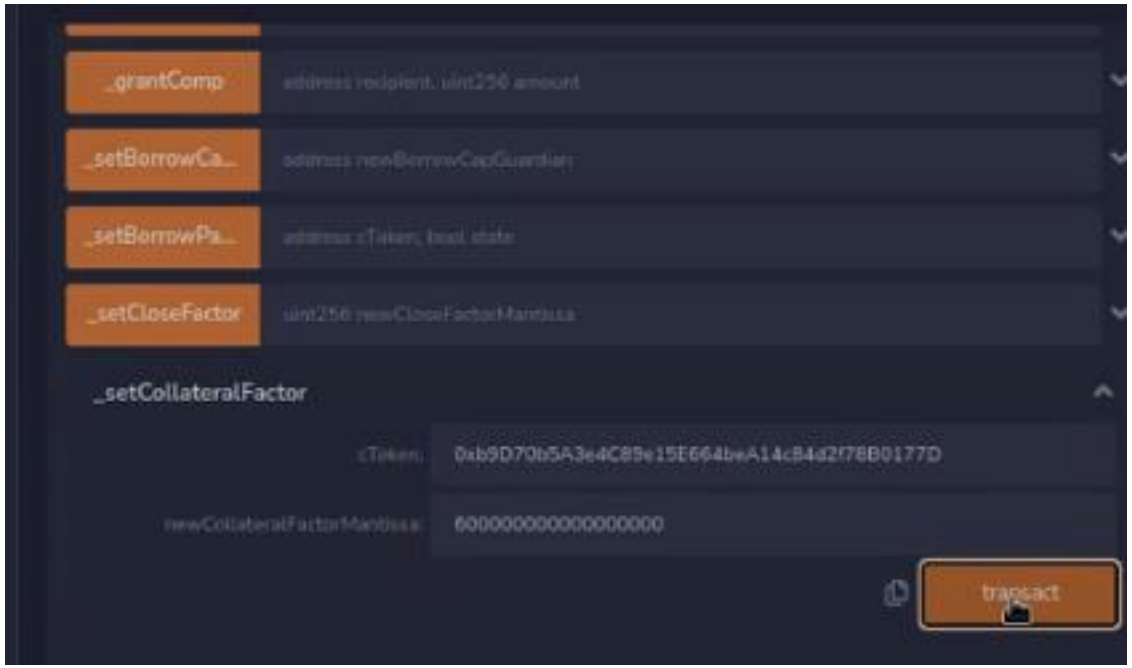


Aggiunta dei Token

CToken

- Copiare indirizzo di un Token già esistente (o crearne uno nuovo) da aggiungere al protocollo
- Accertarsi che provenga dall' implementazione CErc20Delegator o CErc20Immutable
- In figura viene mostrato l'esempio di deploy del cToken relativo a **BAT**
- **UNDERLYING_**: bisogna specificare l' indirizzo del token da aggiungere
- Altri valori sono prelevabili da **kovan.json**



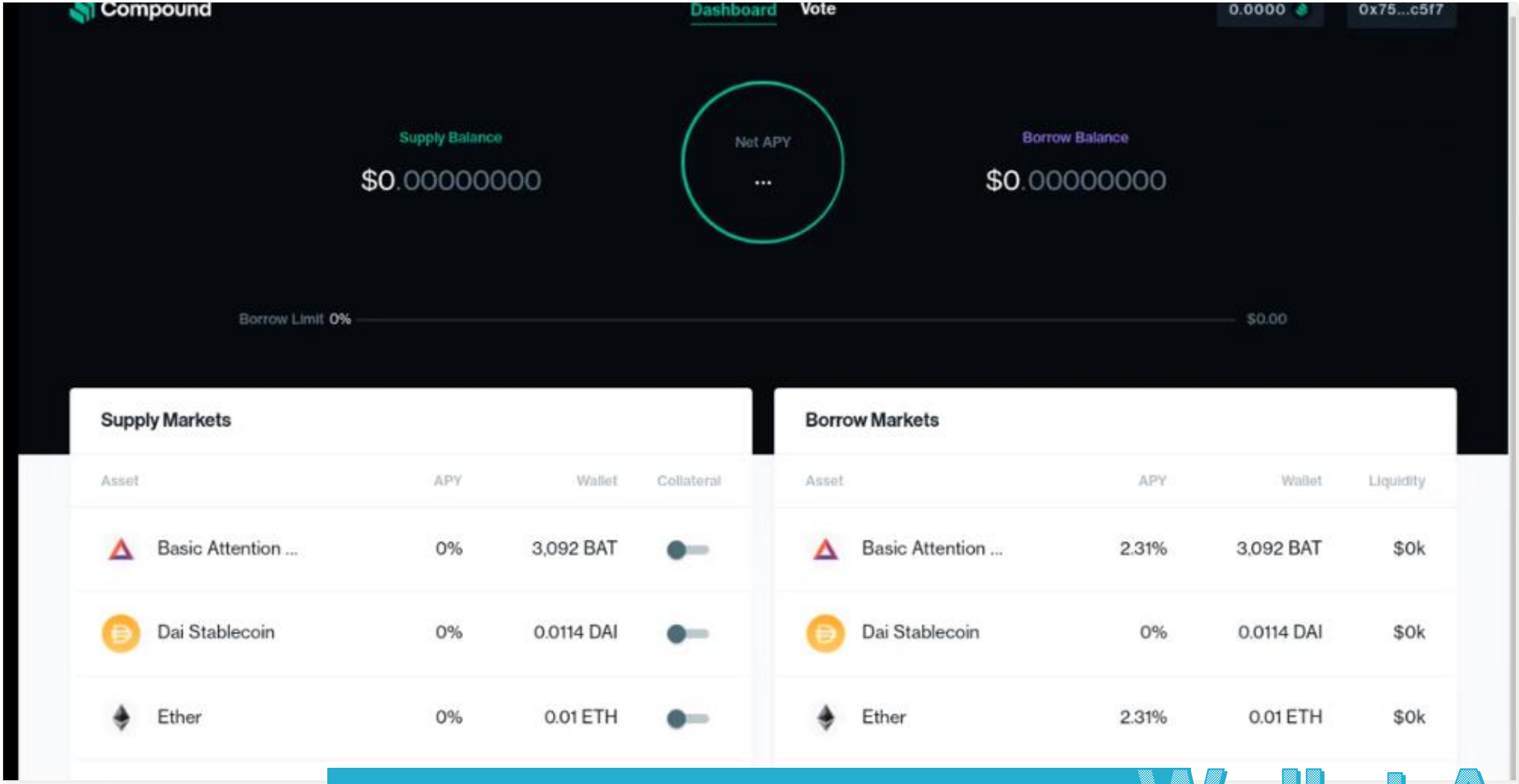


da **Unitroller** (Comptroller):

chiama **_supportMarket** per il cToken appena creato

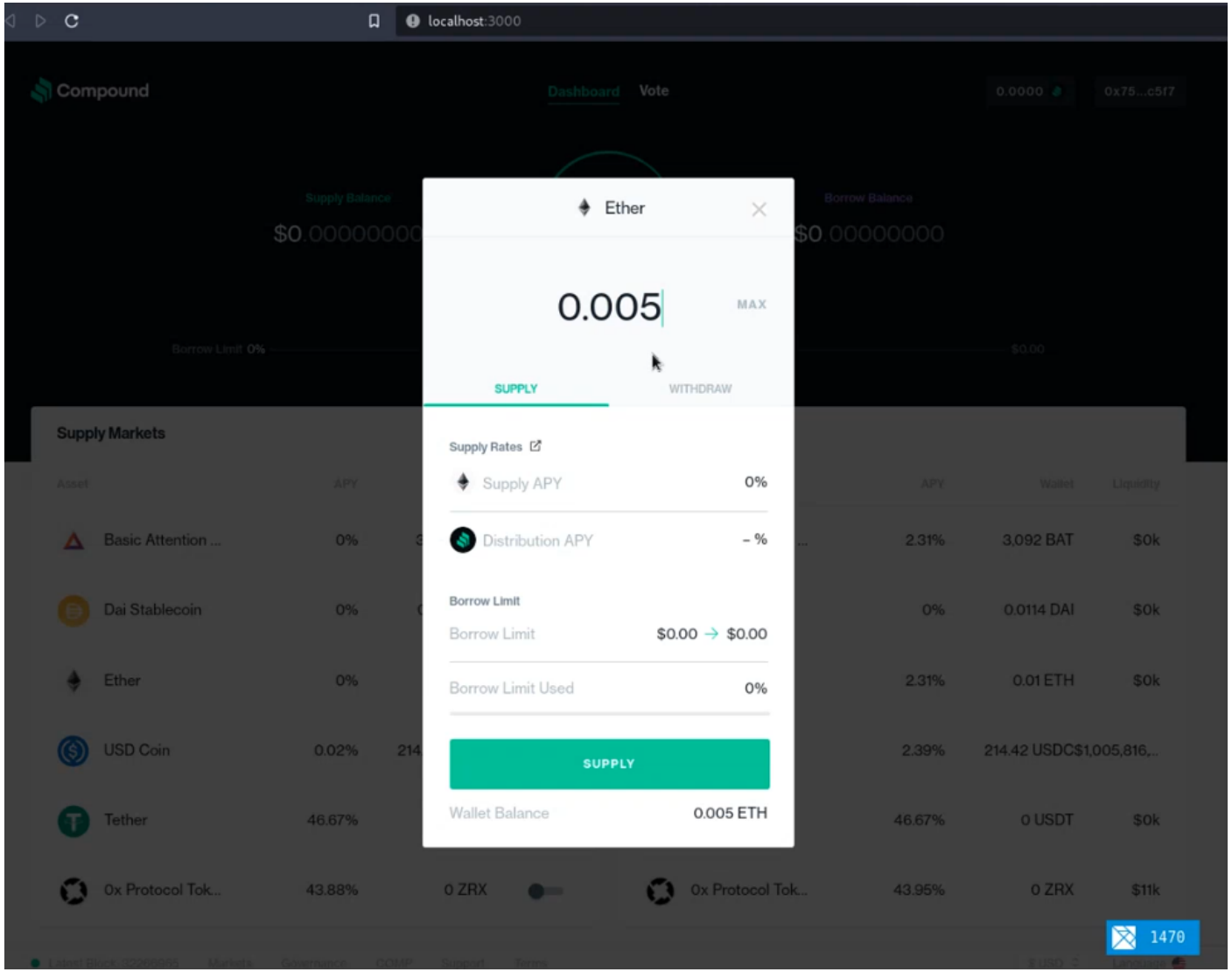
chiama **_setCollateralFactor** per il cToken

Esecuzione


Compound
Dashboard
Vote
0.0000
0x75...c5f7
Supply Balance
$0.00000000
Net APY
...
Borrow Balance
$0.00000000
Borrow Limit 0%
$0.00
Supply Markets
Asset
APY
Wallet
Collateral
Basic Attention ...
0%
3,092 BAT
Dai Stablecoin
0%
0.0114 DAI
Ether
0%
0.01 ETH
Borrow Markets
Asset
APY
Wallet
Liquidity
Basic Attention ...
2.31%
3,092 BAT
$0k
Dai Stablecoin
0%
0.0114 DAI
$0k
Ether
2.31%
0.01 ETH
$0k

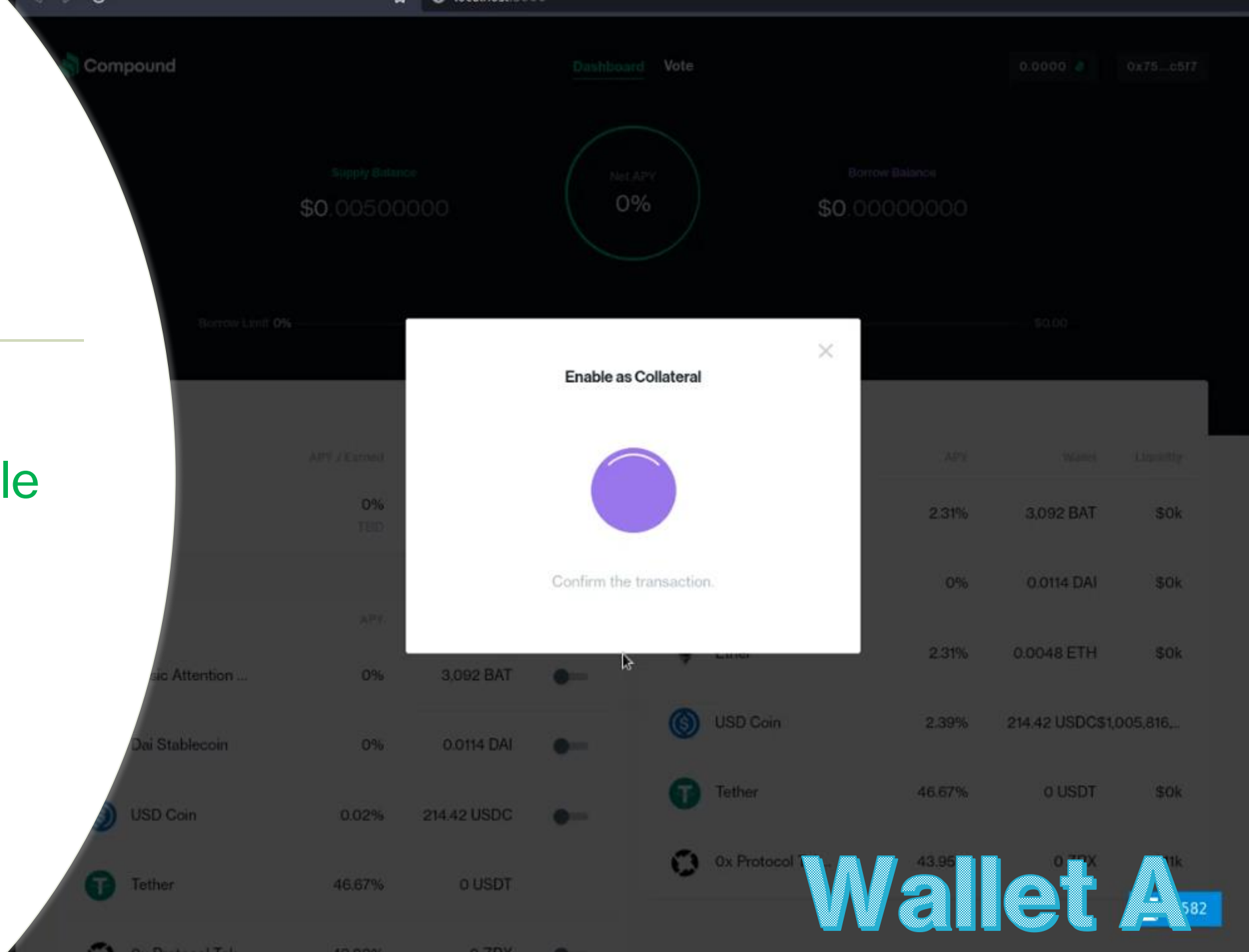
Wallet A

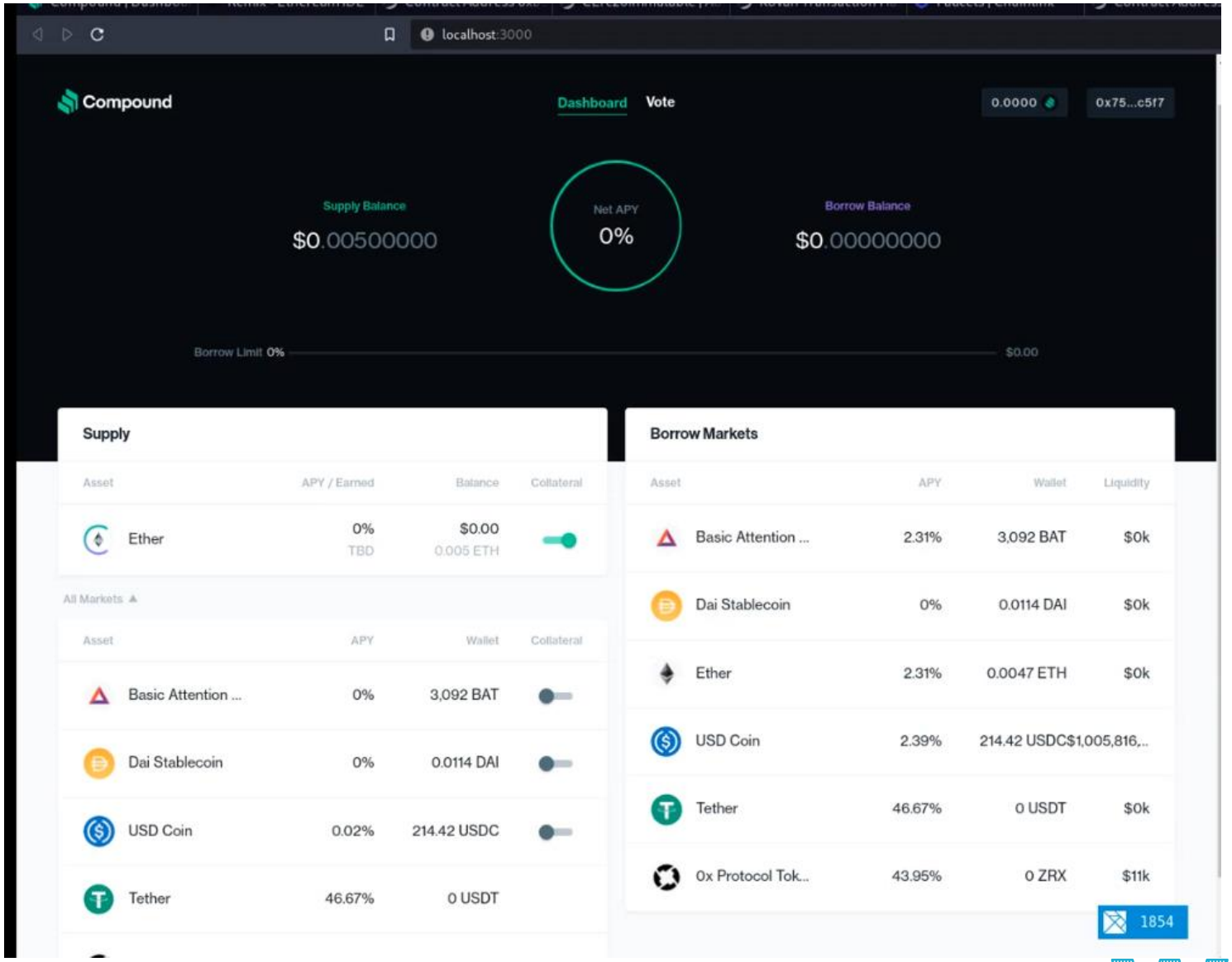
- Supply di Ether



Wallet A

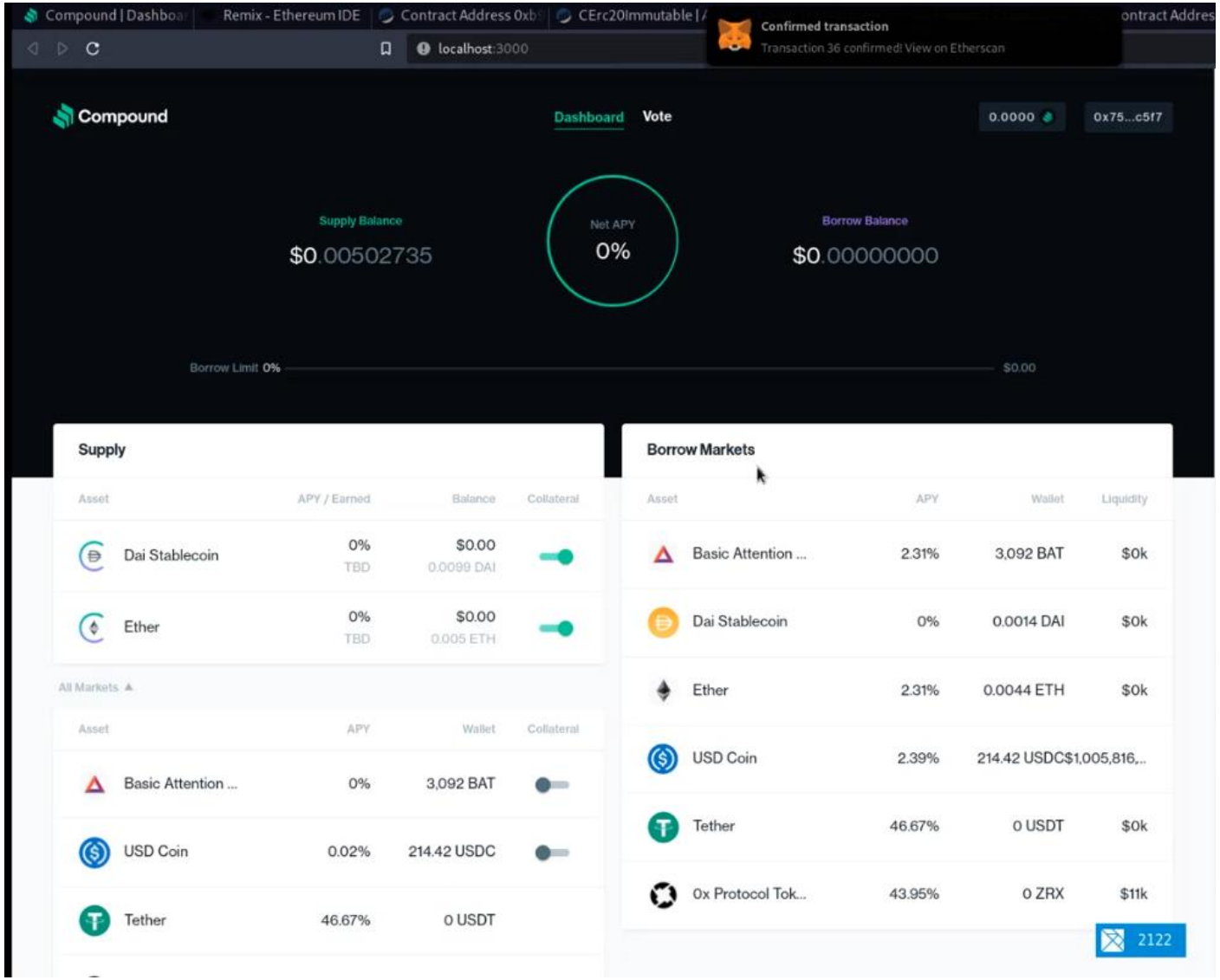
- Abilita Ether come collaterale




localhost:3000
Compound
Dashboard
Vote
0.0000
0x75...c5f7
Supply Balance
$0.00500000
Net APY
0%
Borrow Balance
$0.00000000
Borrow Limit 0%
$0.00
Supply
Asset
APY / Earned
Balance
Collateral
Ether
0%
TBD
$0.00
0.005 ETH
All Markets
Asset
APY
Wallet
Collateral
Basic Attention ...
0%
3,092 BAT
Dai Stablecoin
0%
0.0114 DAI
USD Coin
0.02%
214.42 USDC
Tether
46.67%
0 USDT
Borrow Markets
Asset
APY
Wallet
Liquidity
Basic Attention ...
2.31%
3,092 BAT
$0k
Dai Stablecoin
0%
0.0114 DAI
$0k
Ether
2.31%
0.0047 ETH
$0k
USD Coin
2.39%
214.42 USDC$1,005,816,...
Tether
46.67%
0 USDT
$0k
0x Protocol Tok...
43.95%
0 ZRX
$11k
1854

Wallet A

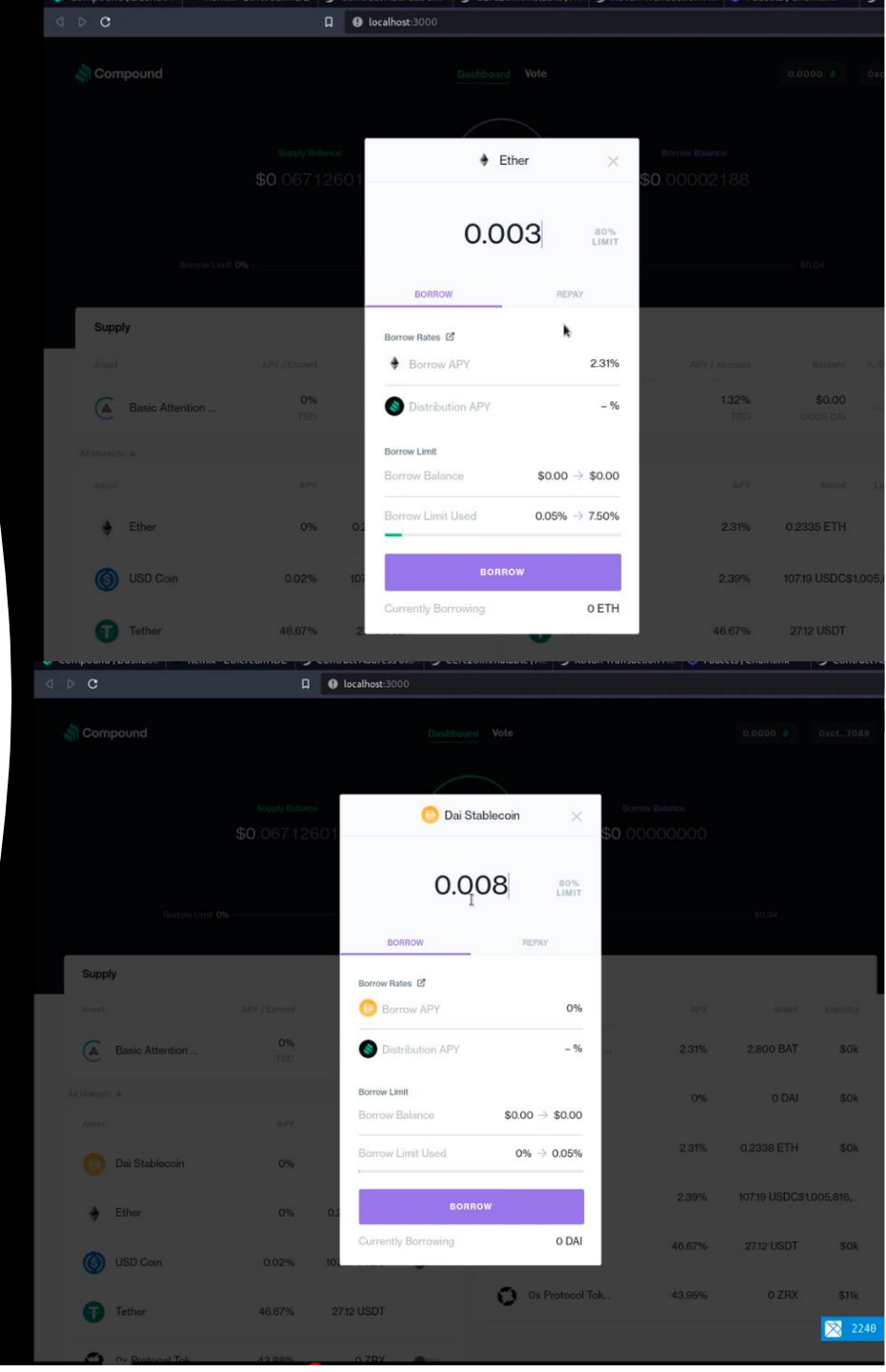
- Supply di DAI
- Abilita DAI come collaterale



Wallet A

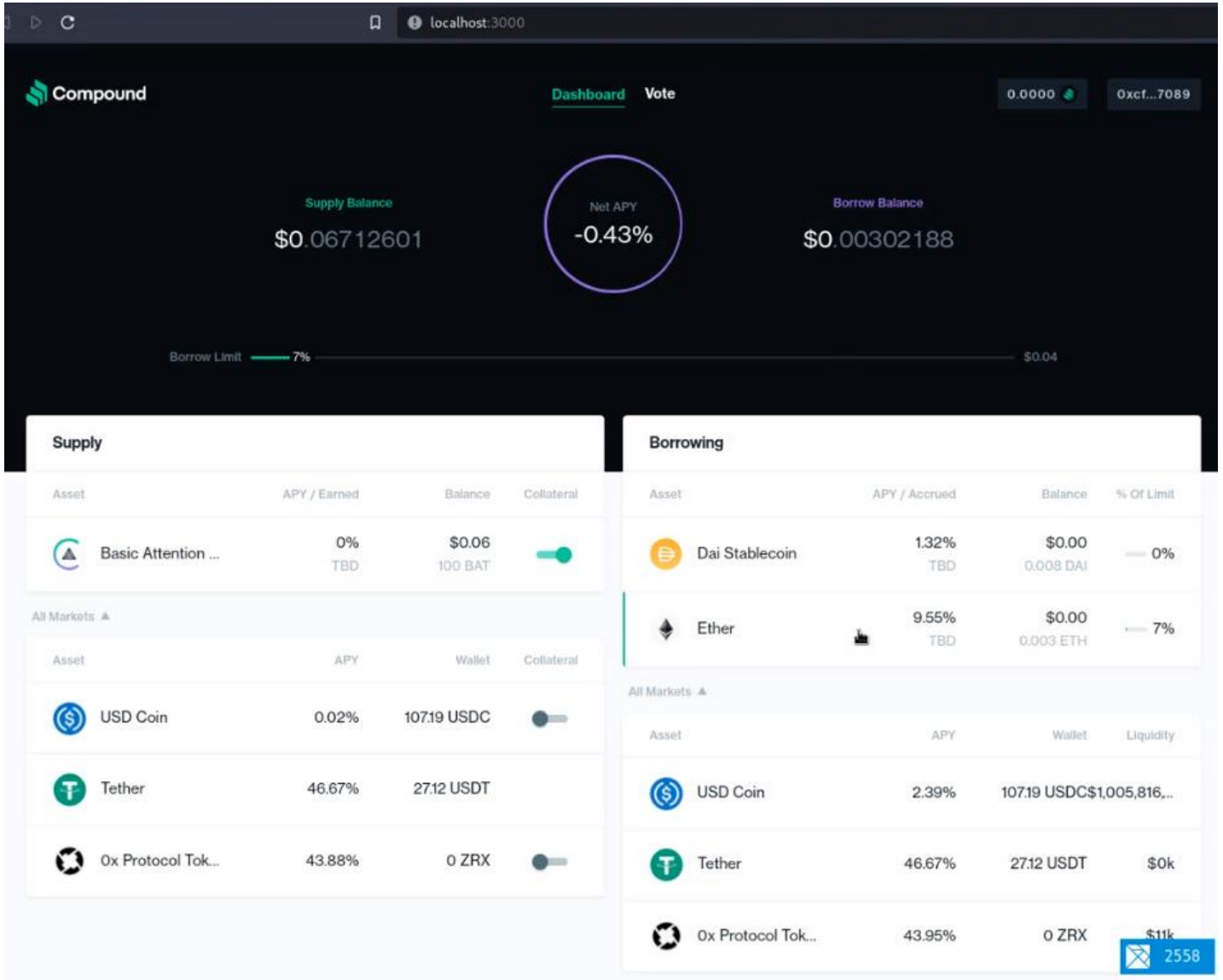
NB: prima di fare operazioni di Borrow, Repay, e withdraw assicurarsi che ci sia liquidità

- Borrow di DAI
- Borrow di Ether



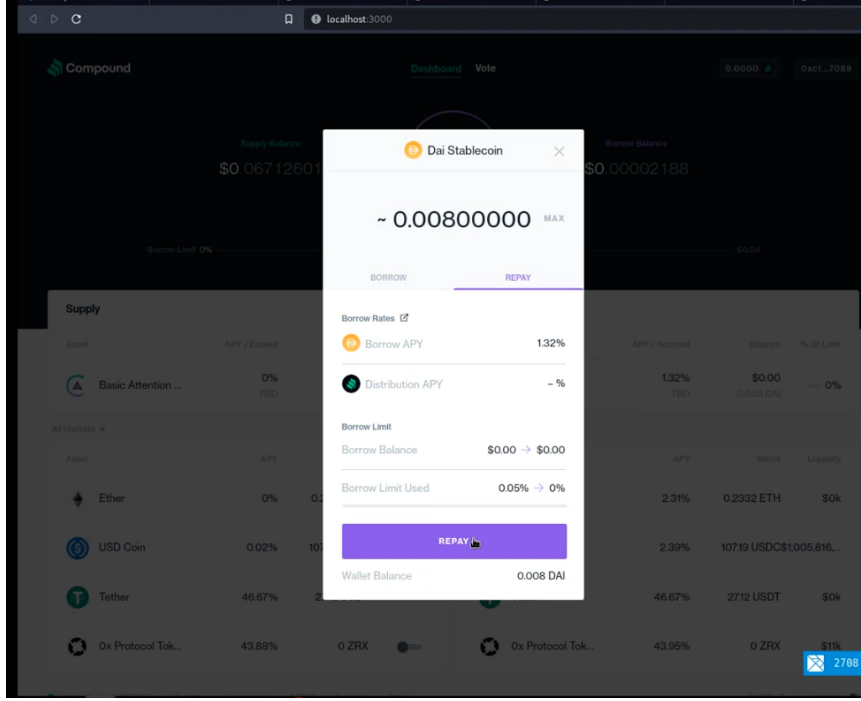
Wallet B

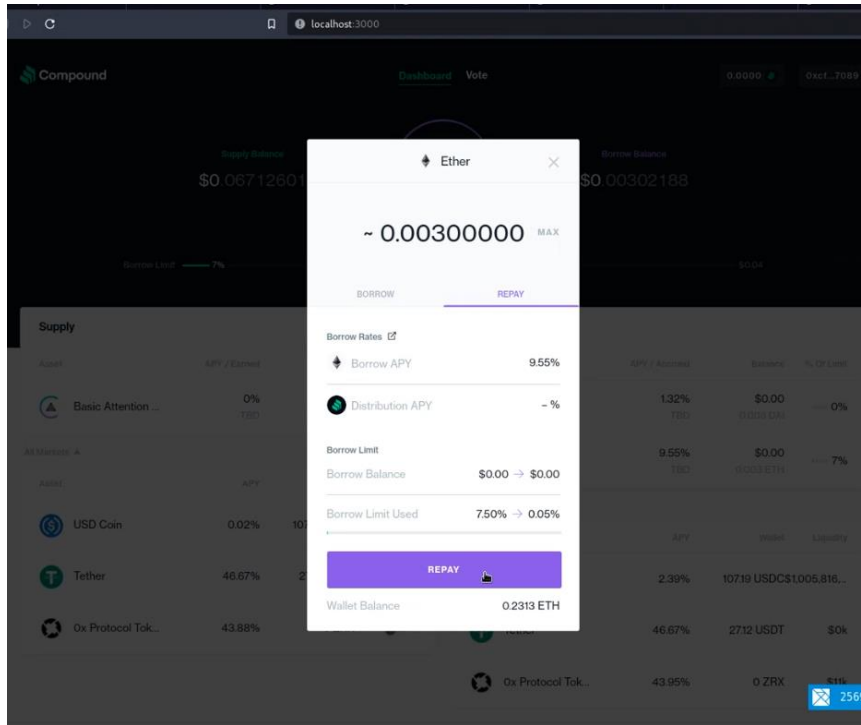
Wallet B


localhost:3000
Compound
Dashboard
Vote
0.0000
0xcf...7089
Supply Balance
$0.06712601
Net APY
-0.43%
Borrow Balance
$0.00302188
Borrow Limit
7%
$0.04
Supply
Asset
APY / Earned
Balance
Collateral
Basic Attention ...
0%
TBD
$0.06
100 BAT
All Markets
Asset
APY
Wallet
Collateral
USD Coin
0.02%
107.19 USDC
Tether
46.67%
27.12 USDT
0x Protocol Tok...
43.88%
0 ZRX
Borrowing
Asset
APY / Accrued
Balance
% Of Limit
Dai Stablecoin
1.32%
TBD
$0.00
0.008 DAI
0%
Ether
9.55%
TBD
$0.00
0.003 ETH
7%
All Markets
Asset
APY
Wallet
Liquidity
USD Coin
2.39%
107.19 USDC
$1,005,816,...
Tether
46.67%
27.12 USDT
$0k
0x Protocol Tok...
43.95%
0 ZRX
$11k
2558

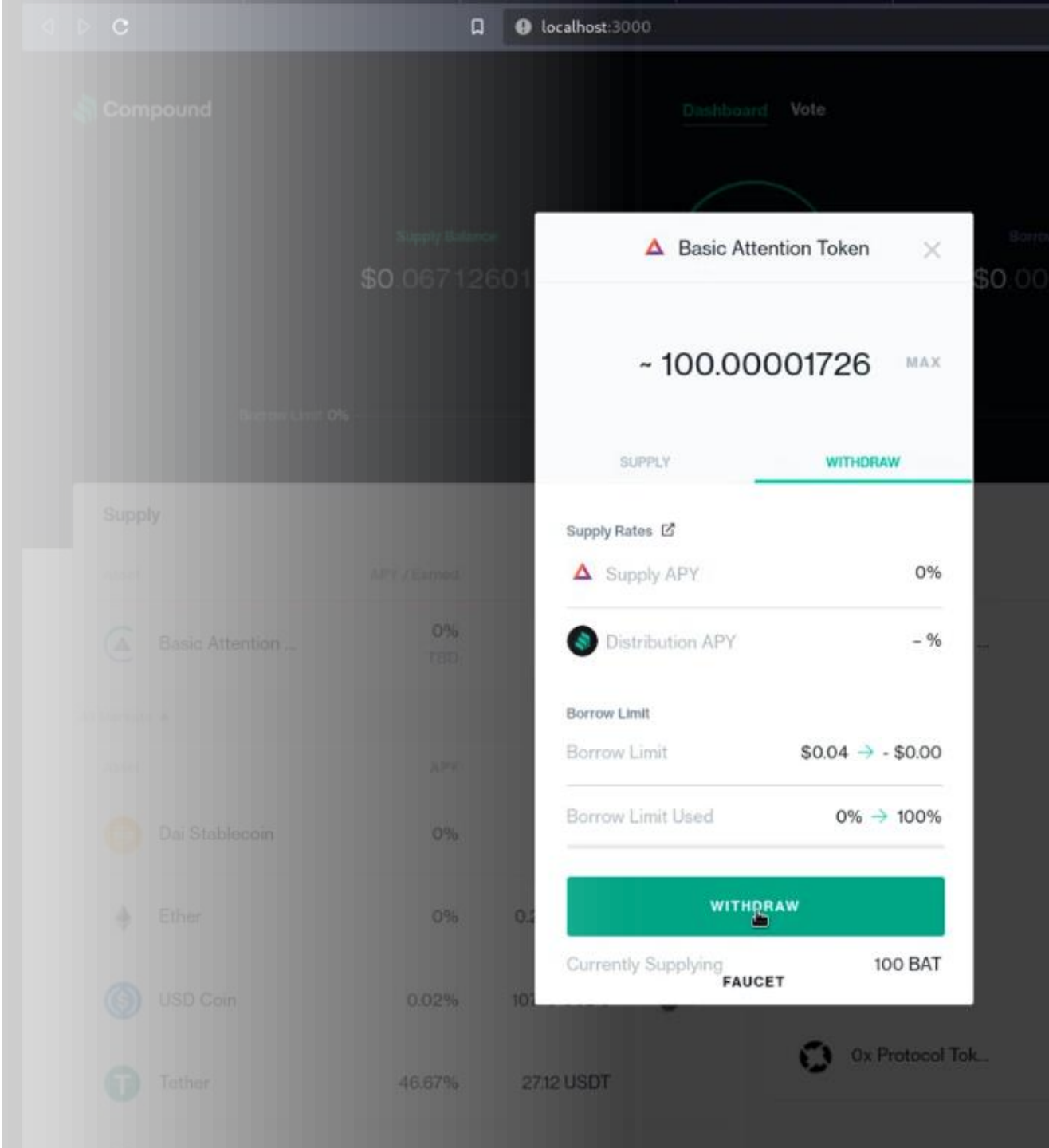
- Repay di Ether
- Repay di DAI

Wallet B



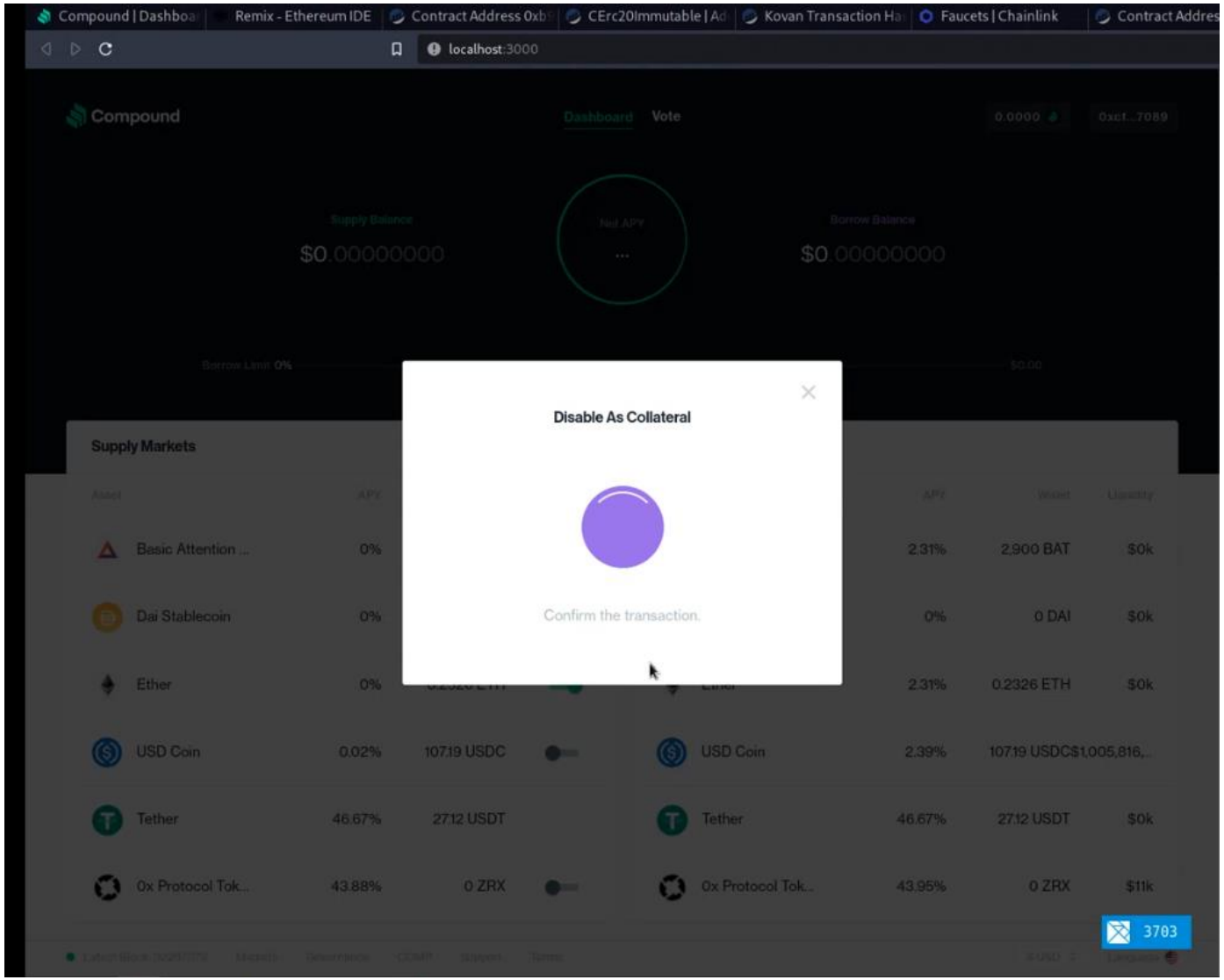


- Withdraw di BAT



Wallet B

- Disattiva BAT come collaterale



Wallet B