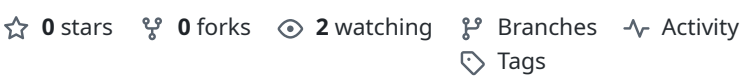
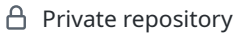

qwoqwop /
BidenBuck
Code
Issues
Pull requests
Actions
Projects
Security
Insights


0 stars
0 forks
2 watching
Branches
Activity
Tags


Private repository

1 Branch 0 Tags Go to file t Go to file Add file Code

qwoqwop added action card to frontend 49e8ea6 · 16 hours ago

.deps	"Add mint function to contract"	last month
.states/vm-cancun	"Add mint function to contract"	last month
artifacts	"Add mint function to contract"	last month
frontend	added action card to frontend	16 hours ago
Architecture.jpg	Revert "yo"	last month
FinalArchitecture.png	Add files via upload	16 hours ago
README.md	Update README.md	16 hours ago
blch.sol	"Add mint function to contract"	last month

README

BidenBuck

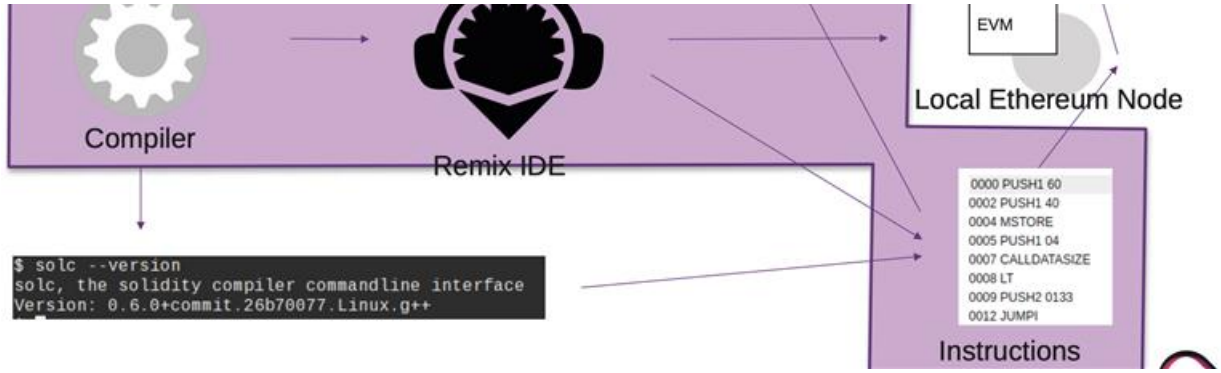
Frontend:

We plan to do our frontend with the framework react because it is part of the WebEngineering 3 module and we would like to get more experience with this framework.

Blockchain:

We include Ethereum as our blockchain, because it has robust smart contract functionality. We will use Ethereum for issuing and redeeming the BidenBuck with smart contracts and will include the existing DEX MetaMask because we do not want to run our own Ethereum node. We will try to keep the Gas Fees as low as possible by minimalizing actions like storage. The prototype is supposed to be functional on an Ethereum testnet. To rebalance our coin we plan to orient ourselves on the price of uranium.

Our architecture will be generally based on the image below as described in the lecture.

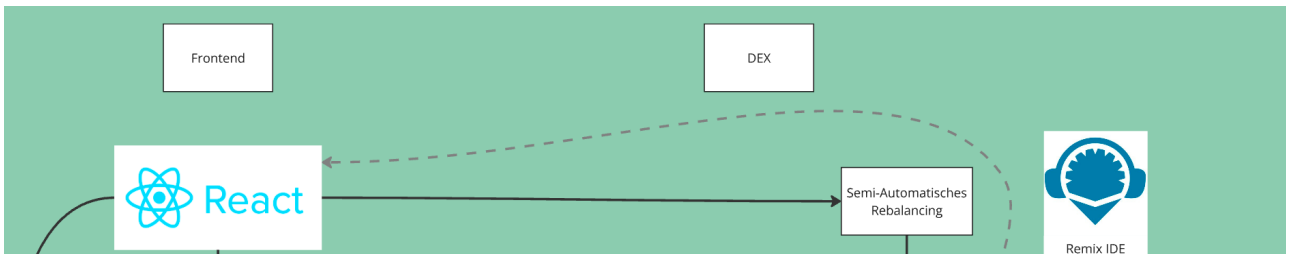


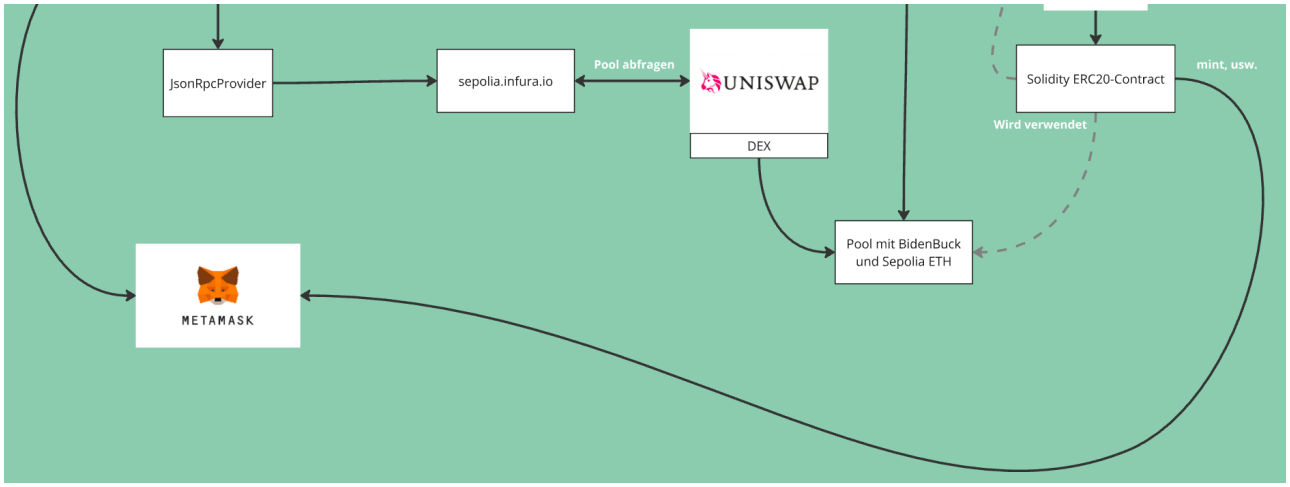
Second hand-in:

We have decided to move from gitlab to github because gitlab is not supported by the remixIDE so it is much more convenient for us to use github. Also we have made our decision to use uniswap as our DEX since sushiswap does not work with the sepholia testnet. There we started exploring liquidity pools to give our coin some "value". Our current plan is to give our coin its value with Sepholia Ethereum and adjusting this price based on the price of uranium. For our frontend we have created a simple react app. It is already able to connect to a contract that was created. All we currently need for that connection is to put the ABI (which can be found in the Solidity Compiler tab) and contract ID into the source code where we can run functions that were defined in the contract. For testing purposes we have created the function get and set which allows us to store and retrieve a number from the contract with our frontend via MetaMask.

Final Product

The BidenBuck is a standard stablecoin that runs on the sepolia testnet and is based on the ERC-20 contract standard. Once the contract is deployed we create a liquidity pool on Uniswap, where the BidenBuck can be traded. Our frontend is used to administer the BidenBuck. It shows us various informations like details about the liquidity pool and even allows us to mint more tokens. We have also implemented a rebalancing function, which detects if the exchange rate deriviates more than 10% from our dedicated exchange rate. If that happens the rebalancing process will be started automatically. If the exchange rate for BidenBucks in ETH is too low, we swap the necessary amount of ETH to BidenBucks. If the rate is too hight, we mint more BidenBucks and swap them for ETH. We have tried to fully automatize this process, however minting tokens directly to the pool had some really bad consequences. The balance of the BidenBuck did not change in the pool, however the balance that we recieved in the frontent changed and the exchange rate changed as well. We also tired to swap BidenBucks for ETH in the pool via a swap router in our react app. However we had to give up on this as we could not resolve the errors we got and found no informations about this anywhere. So we came up with a solution to have our application calculate and mint the necessary amounts. Then it tells us exactly what to do to balance the exchange rate. Our frontend is made with react and is accessible via localhost:3000 when the application is running.





Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 2

TobiKistler

qwoqwop

Languages

Solidity 83.1% **JavaScript** 14.1% **CSS** 1.8% **HTML** 1.0%

Suggested workflows

Based on your tech stack

Publish Node.js Package to GitHub Packages Configure

Publishes a Node.js package to GitHub Packages.

Webpack Configure

Build a NodeJS project with npm and webpack.

Gulp Configure

Build a NodeJS project with npm and gulp.

More workflows Dismiss suggestions