Project Details

Files:

InitialFormV3 shows how the final user interface should be shown. This mockup shows the main components that are part of this project.

Info: Structure of the third-party API

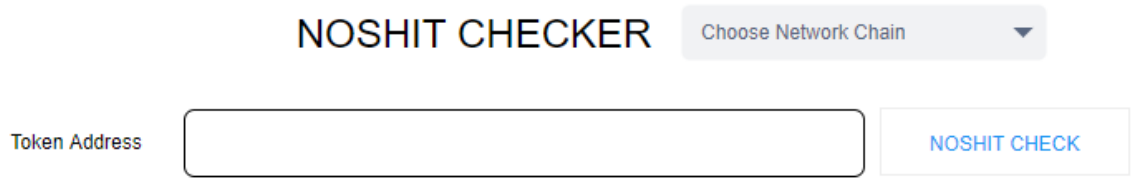
https://api.bscscan.com/api?module=accountaction=balance
&address=Token Address&apikey=YourApiKeyToken

API KEY: 4XKBU6XVDMFW61YVHURHYJ5489BEPJFZF9

Replace YourApiKeyToken with 4XKBU6XVDMFW61YVHURHYJ5489BEPJFZF9 in all mentioned API

Token Address = is the address provided by a user of the platform to see if a toke is a honeypot or not, see example below.

Choose Network Chain: Indicates which network you want to make this search. This should be invisible in the final application. The user just gives the token address, and once the application does not find nothing in BSC scan then it searches in Ethereum.



Section “HoneyPot”

Git: In the following git repo you find the current development of the project for honeypot checker. This already works and caught existing honeypots in BSC and ETHEREUM. You can test it by following instructions on PostmanTestExample.
You will see how we use BSC API to get the ABI of smart contracts and to read the interface of the smart contracts.

Git repo: //or sent project folder

Your task:

a) To show the results from this application:

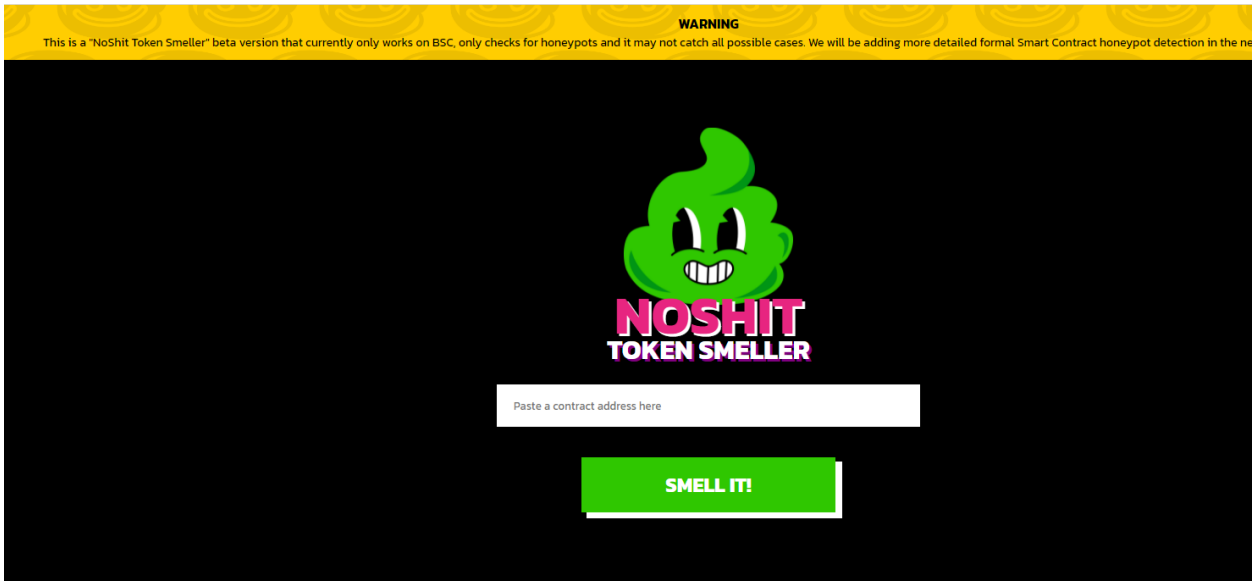
- If the application catches the toke address as honeypot, then show the results **M1** “Show the error message”

- If not a honeypot then show message **M2:** that the token contract is not a honeypot.

Not that **M1 and M2** should be customizable (stored in a variable/constant…), so as admin we can adapt them as we need.

b) If the application catches the token contract as honeypot, then you save that address into a text file or database.

In terms of the front end, the application should look like this: https://noshitchecker.xyz/



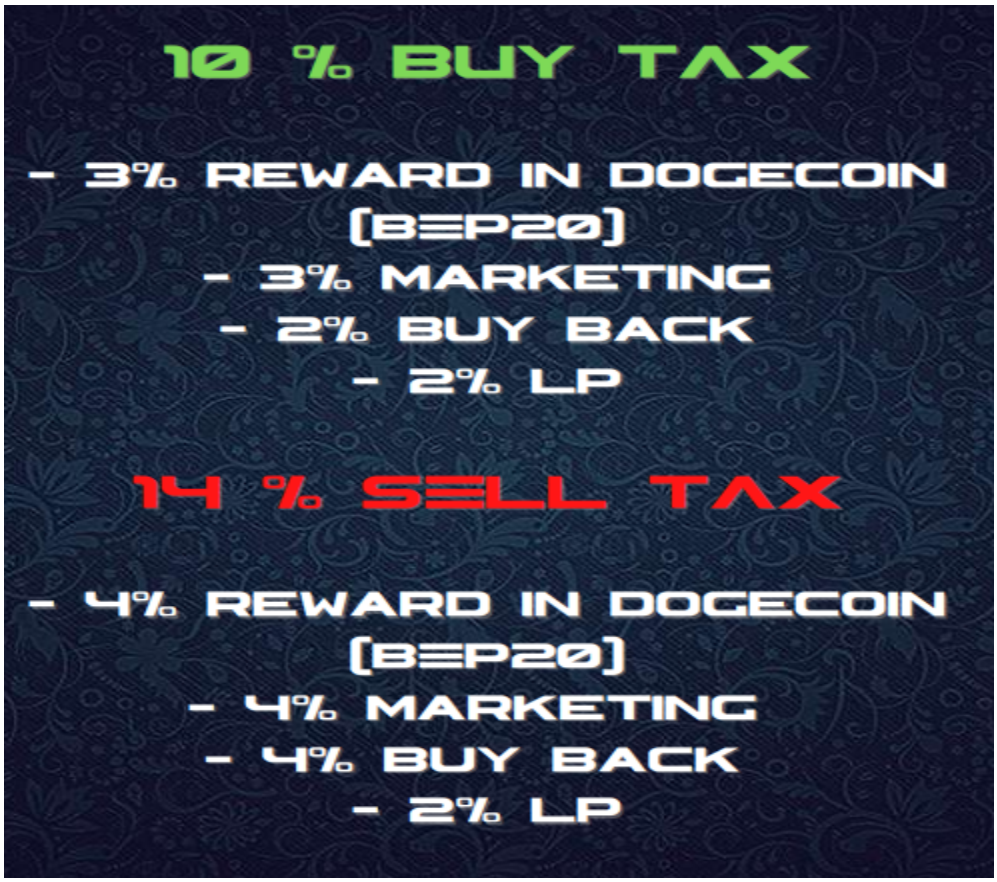
Section “Tokenomiks”

This section presents information about the selling and buying tax of the token, see the example below. Their results should be shown for any token. The example below is one of them but might be some others.

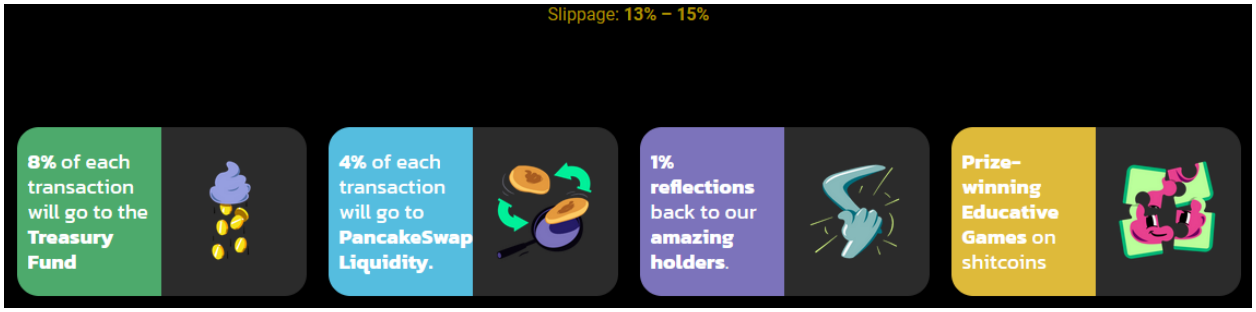
One label that shows if the token has “Liquidity Locked”, should be shown. To manage that you can use the following link
https://dxsale.app/app/v3/dxlplocksearch?id=0&add=0x53F042f3e809d2DcC9492dE2DbF05d1DA0EF5fbb&type=lpdefi&chain=BSC

And you replace the 0x53F042f3e809d2DcC9492dE2DbF05d1DA0EF5fbb with Token Address (for any search).

Here you can show only respond yes/no, if yes then provide label with link (or just show (link))



Another example:



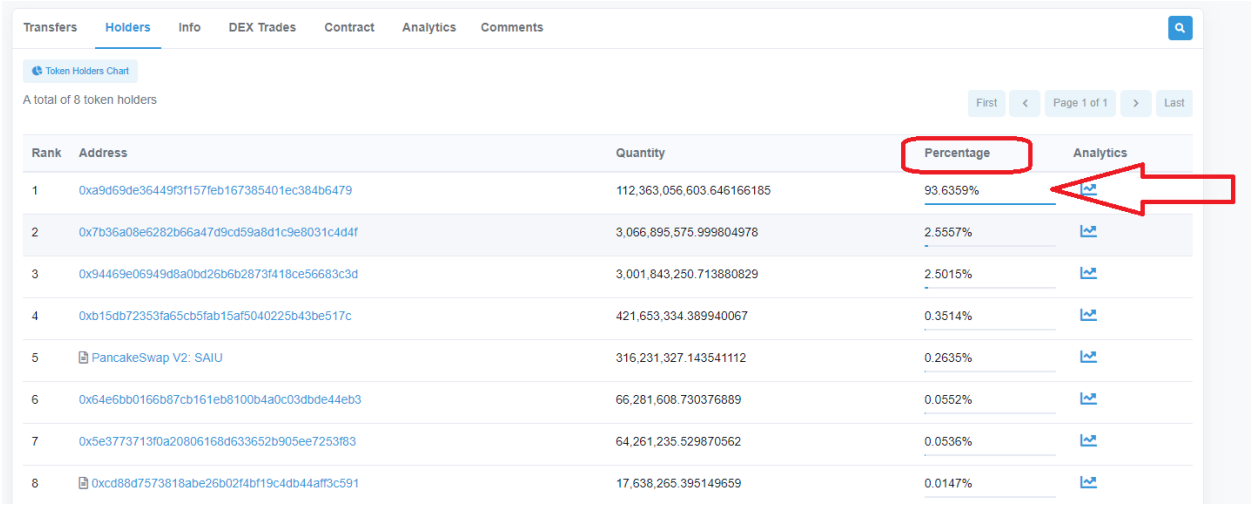
Section “Rughpull”

In this section we check results to fulfill this information.

For the rug pull, we can have the list of “holders” and if one of them or a combination of several of them (excluding exchange wallets such as pancake or unwrap, or addresses such as dead address or null address) holds more than 35% and if the address is not “0x000000000000000000000000000000000000dead” or “0x0dcb9863b54f3150b7d749f7d5b68d8542263edb” then is considered “rughpull”.
The message if positive is “ Yes, it contains address which has X% which can drain liquidity”

Here is an example as below:

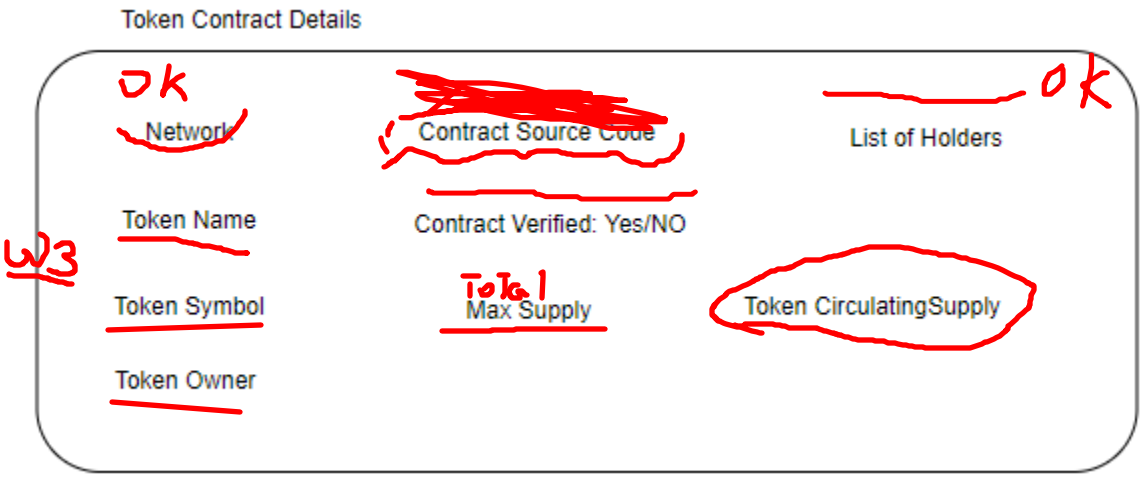
1) You can get the list of holders as explained below in “**List of Token Holders**”
2) For example, we have this:https://bscscan.com/token/0xeeced416862b3aa7a2047ccf7440d04572c4f5d9#balances
3) Here you see that one person has more than 98% and this is rugphull case



To realize this perhaps you need to do web scraping or maybe you can use other techniques, as fits your expertise

Section “Token Contract Details”

Consider the labels shown in the mockup:



To fill their results you can use ABI of the smart contract, which you have already for the honeypot section. **Using web3 functions is the recommended option here, when possible.** Still, I provided some API to understand the context.

You can use different web3 methods to get these results.

```
//1 Toke name
contract.methods.name().call((err, name) => {console.log(name)})

//2 Token Symbol of token
contract.methods.symbol().call((err, symbolTK) => {console.log(symbolTK)})

//3 Token Owner
contract.methods.owner().call((err, ownerResult) =>
{console.log(ownerResult)})
```

Contract Sourc Code: https://api.bscscan.com/api
?module=contract
&action=getsourcecode
&address=0x0e09fabb73bd3ade0a17ecc321fd13a19e81ce82
&apikey=YourApiKeyToken

Once click on **Contract Source Code,** it should open one page which shows the source code in a structured way.

Contract Verified: Yes/NO, this should return results based on API results. See “Check Source Code Verification Submission Status” in the following link
https://docs.bscscan.com/api-endpoints/contracts

Token CirculatingSupply (can be retrieved with web3 of API):
https://api.bscscan.com/api
?module=stats
&action=tokenCsupply
&contractaddress=0xe9e7cea3dedca5984780bafc599bd69add087d56
&apikey=YourApiKeyToken

Token Account Balance (can be retrieve with web3 or api): https://api.bscscan.com/api
?module=account
&action=tokenbalance
&contractaddress=0xe9e7cea3dedca5984780bafc599bd69add087d56
&address=0x89e73303049ee32919903c09e8de5629b84f59eb
&tag=latest
&apikey=YourApiKeyToken

List of Token Holders: Can be taken by this link: List of Holders

https://bscscan.com/token/TokenAddress#balances,

Here you can find all holders for the Token Contract:
https://bscscan.com/token/0x53F042f3e809d2DcC9492dE2DbF05d1DA0EF5fbb#balances

Section “Graphs and Charts”

Provide link button which open specific web page. Once clicked it will redirect to the intend web page based on input from “Token Address” e.g., 0x53f042f3e809d2dcc9492de2dbf05d1da0ef5fbb

PooCoin -- https://poocoin.app/tokens/0x53f042f3e809d2dcc9492de2dbf05d1da0ef5fbb
BscScan -- https://bscscan.com/token/0x53F042f3e809d2DcC9492dE2DbF05d1DA0EF5fbb
Blogged -- https://charts.bogged.finance/?c=bsc&t=0x53F042f3e809d2DcC9492dE2DbF05d1DA0EF5fbb
GecoTerminal -- https://geckoterminal.com/bsc/pools
List of Transactions (Open new webform) --
https://api.bscscan.com/api?module=account&action=txlist&address=0xF426a8d0A94bf039A35CEE66dBf0227A7a12D11e&startblock=0&endblock=99999999&page=1&offset=10&sort=asc&apikey=YourApiKeyToken

Section “Check List of Transfer Events Specific SC and account”

See this link, point “ Get a list of 'BEP-20 Token Transfer Events' by Address” for better understanding.

In the first text box, the user can provide the token address (this can get default from Token Address) and if he wants he can add another address

https://api.bscscan.com/api?module=account
&action=tokentx&contractaddress=0xc9849e6fdb743d08faee3e34dd2d1bc69ea11a51&address=0x7bb89460599dbf32ee3aa50798bbceae2a5f7f6a&page=1&offset=5&startblock=0&endblock=999999999&sort=asc&apikey=YourApiKeyToken

Contract address is Token Address

You can use like this:

https://api.bscscan.com/api?module=account&action=tokentx&contractaddress=0x53F042f3e809d2DcC9492dE2DbF05d1DA0EF5fbb&offset=5&startblock=0&endblock=999999999&sort=asc&apikey=YourApiKeyToken

Token Circulating Supply for given Address: https://api.bscscan.com/api
?module=stats
&action=tokenCsupply
&contractaddress=0xe9e7cea3dedca5984780bafc599bd69add087d56
&apikey=YourApiKeyToken