

# A step-by-step guide on how to create a cryptocurrency:

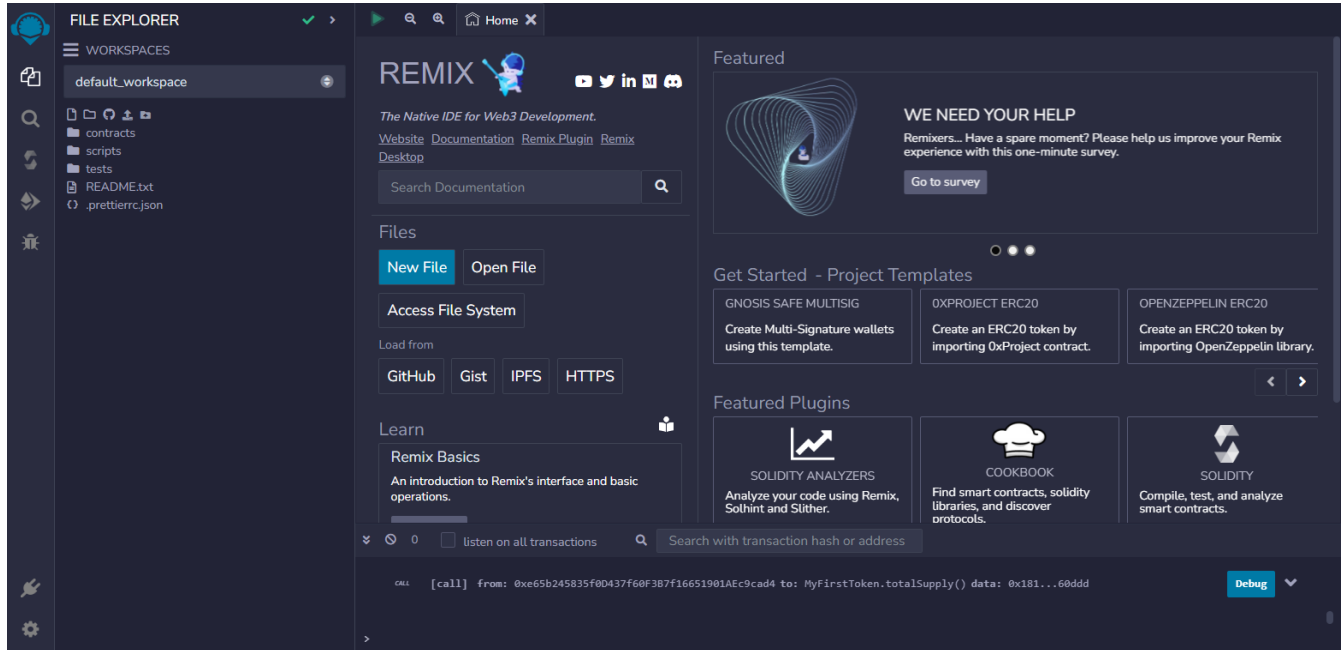
*Guide by Regel Cabrales*

## Prerequisites:

- ☒ Familiarity with Ethereum
- ☒ A Metamask wallet and some test tokens in the wallet (e.g. MATIC)

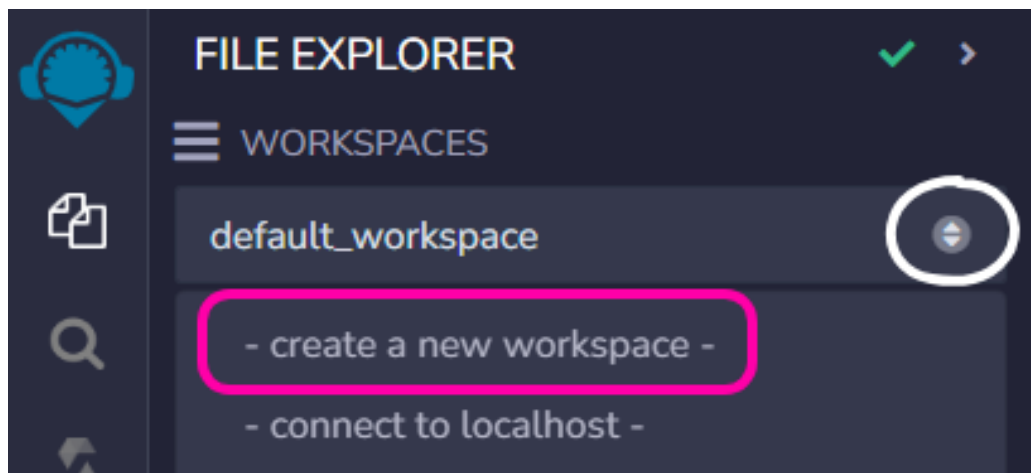
## Step 1: Set up Remix

- Go to the **Remix** website: <https://remix.ethereum.org/>

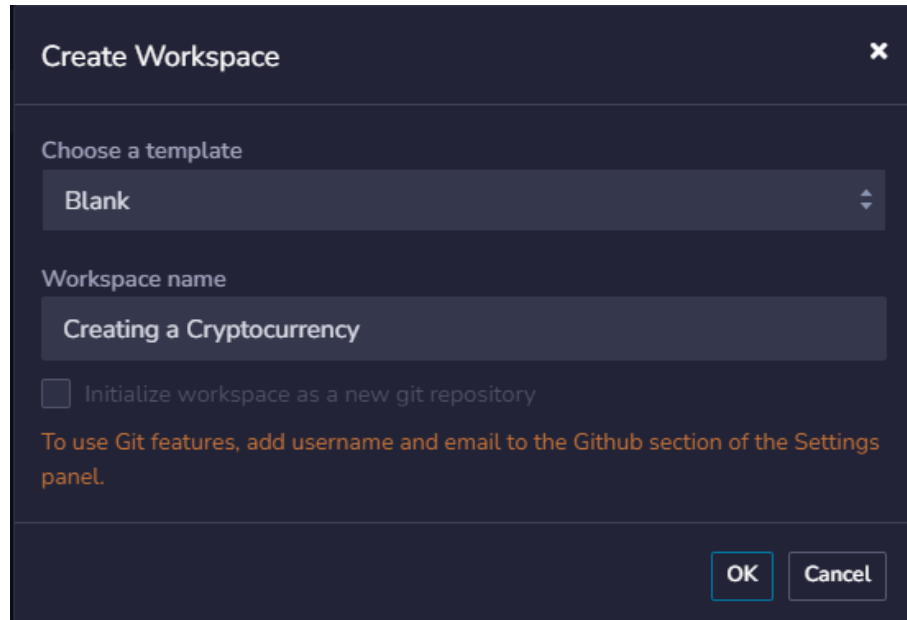


*\*Remix is a Solidity IDE that allows you to write, deploy, and test smart contracts. You can use it in your web browser.*

- Create a new workspace by clicking on the dropdown button (inside the white circle) then clicking on “ - create a new workspace - ” (inside the pink box) or use the default workspace.

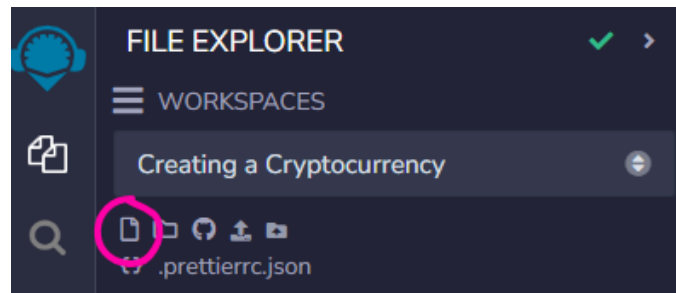


3. A window will pop up for creating a new workspace. Set the template as “**Blank**” and give your workspace a name (e.g. “Creating a Cryptocurrency”). Then click “**OK**” to create your new workspace.



## Step 2: Write the Smart Contract

1. Create a new Solidity file by clicking the new file icon in the File Explorer (inside the pink circle) and name your new file (e.g., MyToken.sol). Remember to add the .sol extension to your file.



2. Start with a boiler plate. Declare the smart contract license and Solidity version we'll be working with by typing this in:

```
// SPDX-License-Identifier: MIT  
pragma solidity ^0.8.0;
```

*\*MIT License is an open-source license that allows you to use, modify, and distribute the code with minimal restrictions.*

3. Import the **ERC20 contract** and **Ownable contract** from the OpenZeppelin library by typing this in:

```
import "@openzeppelin/contracts/token/ERC20/ERC20.sol";  
import "@openzeppelin/contracts/access/Ownable.sol";
```

*\*ERC20 token contract keeps track of fungible tokens.*

*\*Ownable is for implementing ownership in your contracts.*

*\*Learn more about OpenZeppelin and its contracts by visiting this website:*

<https://docs.openzeppelin.com/contracts/4.x/>

4. Declare a new smart contract and inherit the ERC20 and Ownable contract by typing:

```
contract MyToken is ERC20, Ownable {}
```

- Inside the contract you declared type this constructor in to initialize your token with two parameters, the name of your token and the symbol of your token:

```
constructor() ERC20("Neko Token", "CATO") {}
```

- Inside the constructor, use the “**\_mint**” function inherited from the ERC20 contract.

```
_mint(msg.sender, 10 * (10 ** 18));
```

*\***msg.sender** is the Ethereum address of the account that deploys the contract. In this case, it's the address of the person creating the token.*

*\***10 \* (10 \*\* 18)** calculates the initial supply of your token, creating a supply of 10 quintillion tokens.*

- Create a function called “**mint**” that allows the owner of the contract to create and add more tokens to the total supply.

```
function mint(uint256 amount) public onlyOwner {  
    _mint(msg.sender, amount);  
}
```

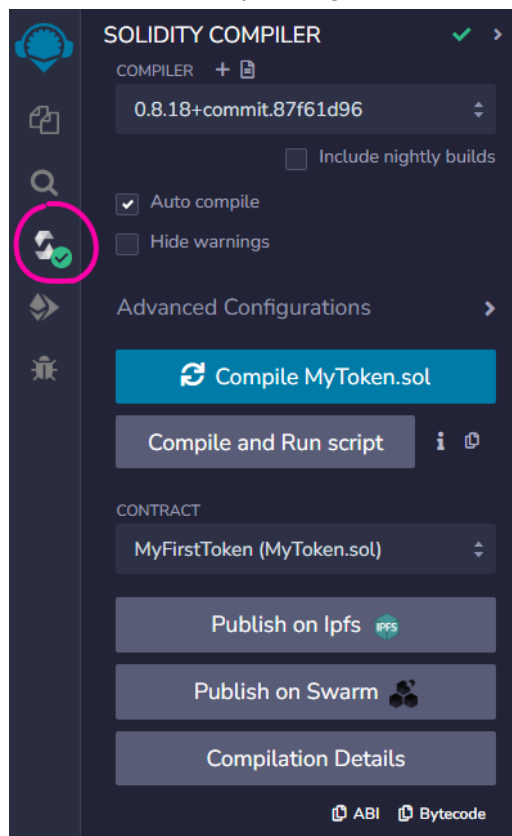
*\***onlyOwner** ensures that only the owner can execute this function.*

- Create another function called “**burn**” that allows the contract owner to “burn” (destroy) a specified amount of tokens from the total supply.

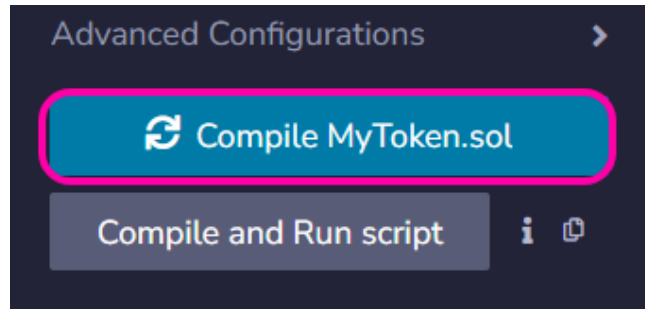
```
function burn(uint256 amount) public onlyOwner {  
    _burn(msg.sender, amount);  
}
```

### Step 3: Compile and Deploy the Contract

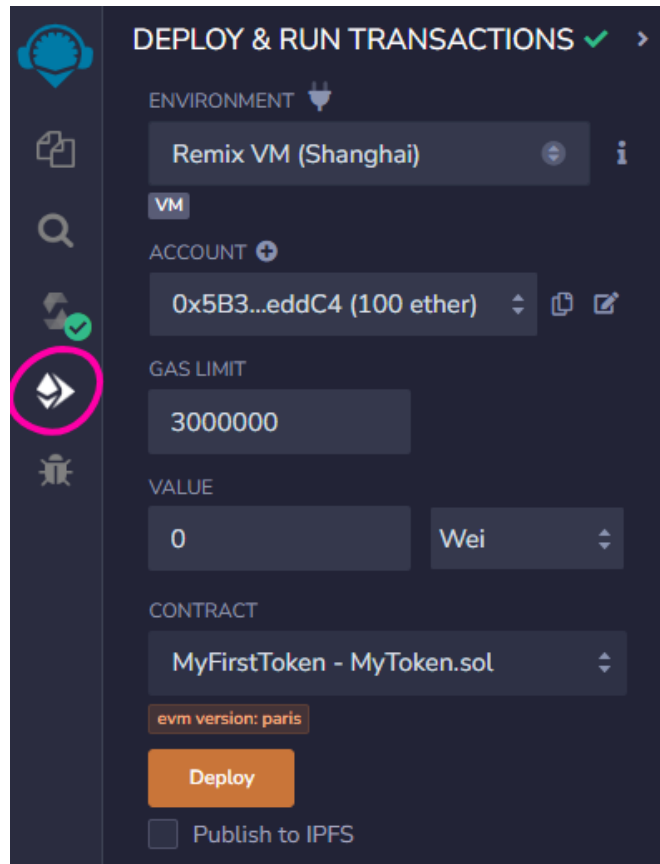
- Go to the “**Solidity Compiler**” tab (inside the pink circle). You can tick the “Auto compile” checkbox to automatically compile your contract if there are any changes.



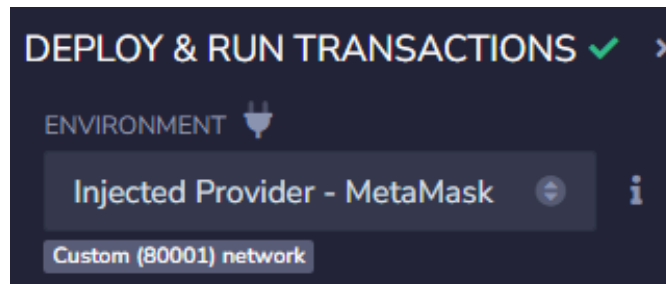
- Click the “**Compile (file name)**” button *(inside the pink box)* to compile your contract. Make sure there are no errors.



- Switch to the “**Deploy & Run Transactions**” tab *(inside the pink circle)*.

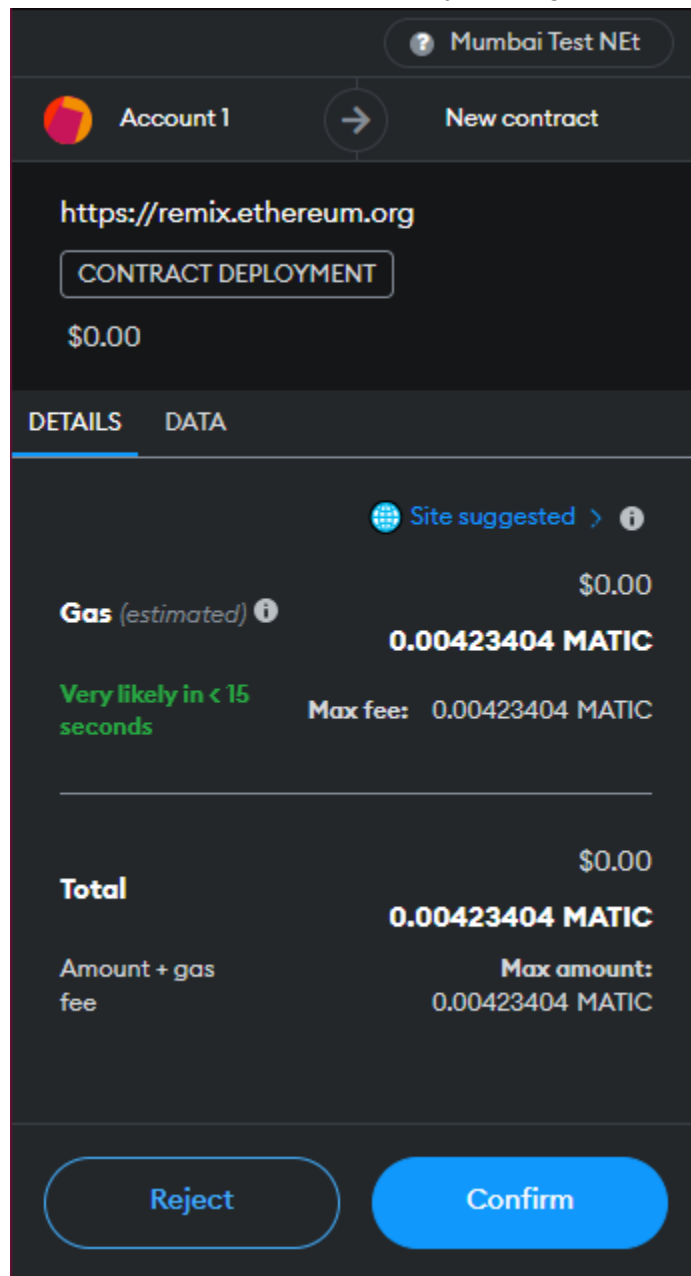


- Make sure that your environment is set to “**Injected Provider - Metamask**”.



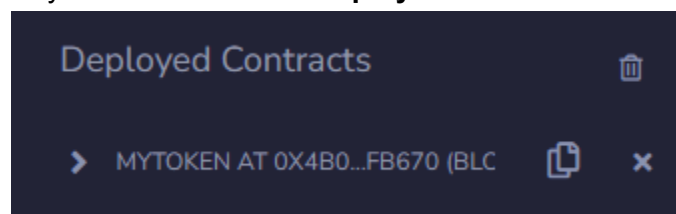
- Click the orange “**Deploy**” Button.

6. A Metamask window will pop up. Confirm the transaction by clicking “**Confirm**”.



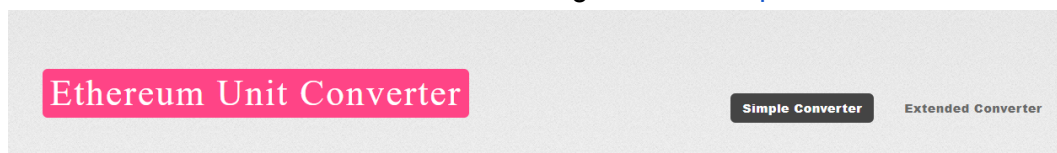
#### Step 4: Create and Add Tokens to Your Token Supply

1. After deployment, you'll see your contract in the “**Deployed Contracts**” section.



2. Expand your deployed contract, and you'll find the functions defined in your contract ('*mint*' and '*burn*') together with other functions from the ERC20 and Ownable contracts (e.g. '*totalSupply*', '*owner*', etc). You can interact with your token contract by using these functions.

3. Go to the the **Ethereum Unit Converter** website through this link: <https://eth-converter.com>



## Simple Unit Converter

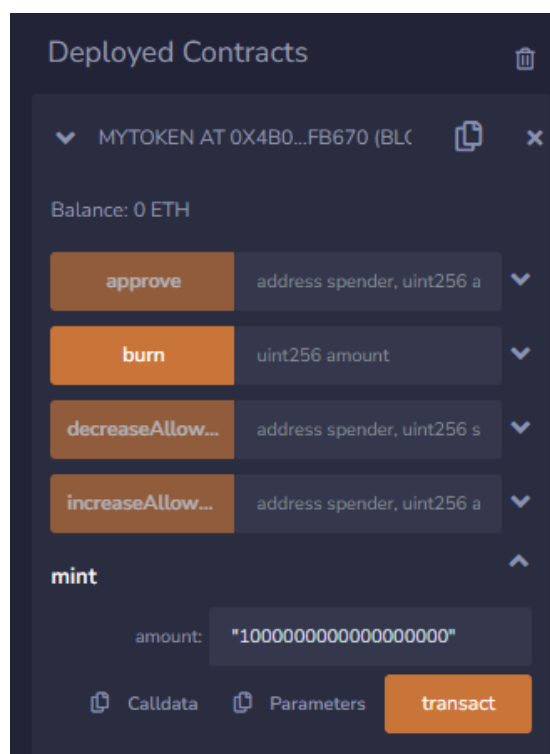
After constantly using a calculator to convert between Ether units i thought it would neat to implement the [EthereumJS-Units](#) library and [BigNumber](#) library on a website for everyone to use. There is a [simple](#) and [full](#) converter page, since the majority wouldn't bother for any other units than Ether, Gwei and Wei.

Wei	<input type="text" value="Wei"/>
Gwei	<input type="text" value="Gwei"/>
Ether	<input type="text" value="Ether"/>

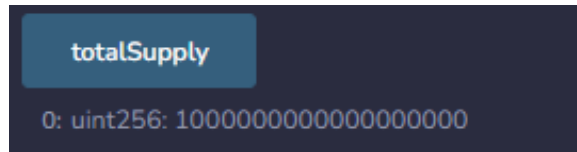
4. The token you created is fungible which means its equivalent to Ether (1 Your Token = 1 Ether). Enter how many tokens you want in the Ether input field to convert it into Wei, which is the amount we will be entering for the mint function.

Wei	<input type="text" value="1000000000000000000"/>
Gwei	<input type="text" value="1000000000"/>
Ether	<input type="text" value="1"/>

5. Copy the converted unit in Wei and paste it in the mint function in your deployed contract. Then click **“transact”**.

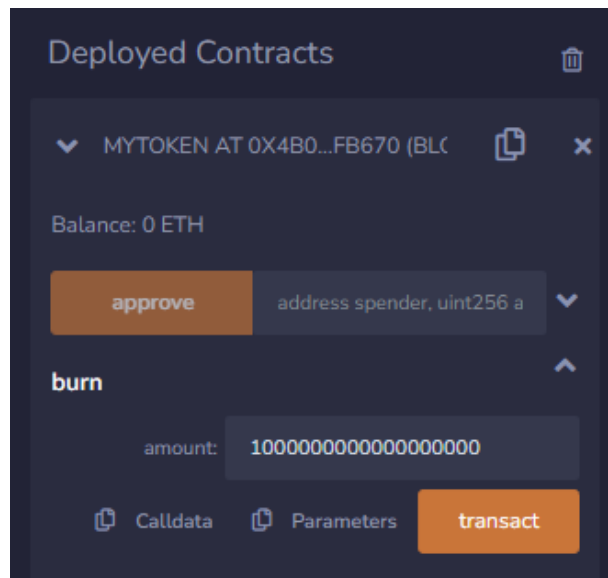


6. A Metamask window will pop up. Confirm the transaction by clicking “**Confirm**”.
7. Once the transaction is complete, you can check the total supply of your tokens by simply clicking on the “**totalSupply**” function in your deployed contract.

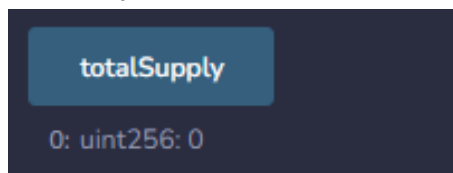


### Step 5: Burn/Destroy Tokens from Your Token Supply

1. You could also burn/destroy tokens from your token supply by using the “**burn**” function in your deployed contract. Use the Ethereum Unit Converter to calculate the amount you want to burn. Put in that amount in the “**burn**” function then click “**transact**”.

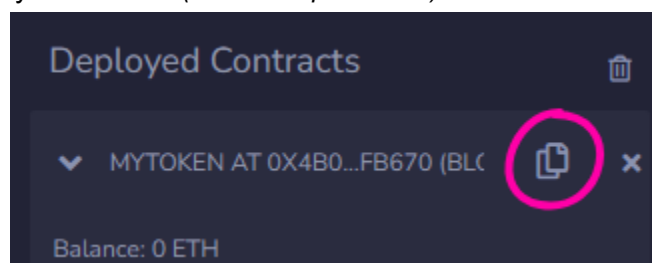


2. A Metamask window will pop up. Confirm the transaction by clicking “**Confirm**”.
3. Once the transaction is complete, you can once again check the total supply of your tokens by clicking on the “**totalSupply**” function in your deployed contract. .



### Step 6: Check the Total Supply of Your Created Token

1. To see how much of your created tokens are in your wallet, copy your contract address by clicking on the copy icon in your deployed contract (*inside the pink circle*).



2. After copying, go to this website: <https://mumbai.polygonscan.com>

Polygon PoS Chain Testnet Explorer

All Filters Search by Address / Txn Hash / Block / Token / Domain Name

Latest Blocks

Bk	Validated By	MATIC
39840521 25 secs ago	0x5082f249cdb2f2c1ee0... 8 txns in 2 secs	0.04332 MATIC
39840520 27 secs ago	0x5082f249cdb2f2c1ee0... 37 txns in 2 secs	0.01767 MATIC
39840519 29 secs ago	0x5082f249cdb2f2c1ee0... 10 txns in 2 secs	0.0099 MATIC
39840518 31 secs ago	0x5082f249cdb2f2c1ee0... 154 txns in 2 secs	0.0032 MATIC
39840517 33 secs ago	0x5082f249cdb2f2c1ee0... 40 txns in 2 secs	0.0057 MATIC

Latest Transactions

Tx	From	To	MATIC
0xe1eed5a5f7a3... 25 secs ago	0xd36143540aab8a347...	0xbec2a34060da4f9841...	0 MATIC
0x8fccdb1067cf7... 25 secs ago	0x02cfd02397c65c32fa...	0x870e95e11a49f56b32...	0 MATIC
0x67f2206ae06b... 25 secs ago	0xf58119ab19dbaf2ef59...	0x16c328285befd46c4a...	1 MATIC
0xe2274326f5bc... 25 secs ago	0x3ea4bf681bdc3a1d7e...	0x16c328285befd46c4a...	1 MATIC
0x49e568fc4e8bf... 25 secs ago	0x8ec971284a41bb0e6e...	0xc820dd023bf7ac0feb...	10 MATIC

3. Paste your copied contract address in the search bar and hit enter. You'll see details about your contract in the next page such as the balance, transactions etc.

Contract 0x4B0171D7775fb7da1338F7b30ECF4CB527fb670

Contract Overview

Balance: 0 MATIC

More Info

My Name Tag: Not Available

Contract Creator: 0xe65b245835f0d437f60... at txn 0x7f9d607350950b8128...

Token Tracker: Neko Token (CATO)

Transactions ERC-20 Token Txns Contract Events

4. Click on your token by the “Token Tracker” (inside the pink box).

More Info




My Name Tag: Not Available

Contract Creator: 0xe65b245835f0d437f60... at txn 0x7f9d607350950b8128...

Token Tracker: Neko Token (CATO)



- After clicking, you'll see details about your token such as the total supply, holders, the transfers that have been done to your token, and other information.

Overview <span>ERC-20</span>	
Total Supply:	 0 CATO 
Holders:	0 addresses
Transfers:	7 

- You have now successfully created your token, as well as minted and burned from your token's total supply.

# A step-by-step guide on how to add liquidity to your token

## Prerequisites:

- ☒ Familiarity with Ethereum and Solidity
- ☒ A Metamask wallet and your tokens in the wallet (e.g. CATO)

## Step 1: Connect BNB Smart Chain Testnet to Your Wallet

1. Open the **Chainlist** website through this link: <https://chainlist.org>.



Helping users connect to EVM powered networks

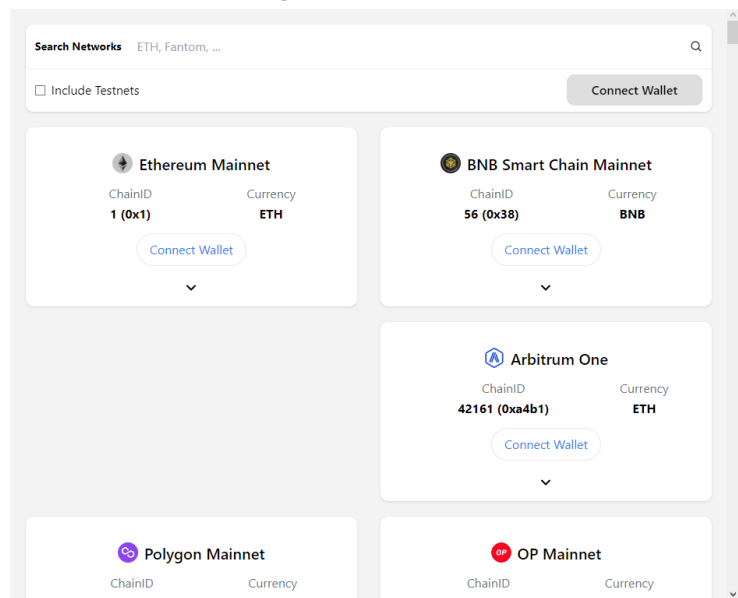
Chainlist is a list of EVM networks. Users can use the information to connect their wallets and Web3 middleware providers to the appropriate Chain ID and Network ID to connect to the correct chain.

Add Your Network +

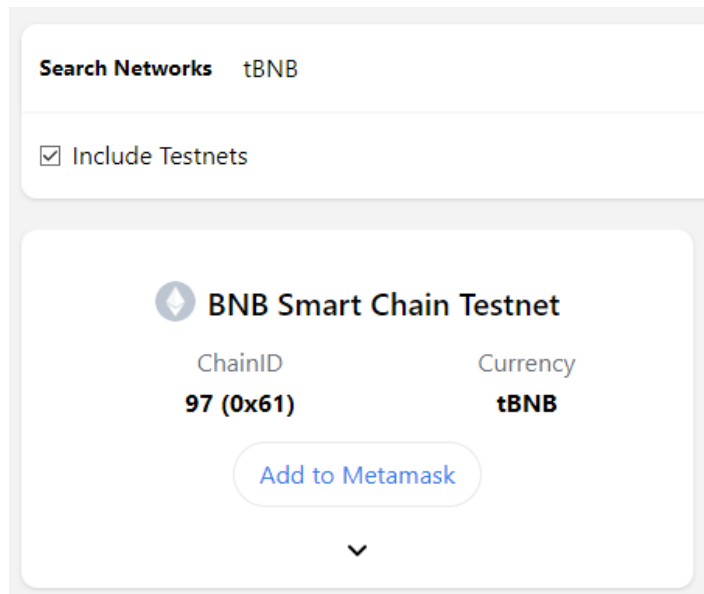
Add Your RPC +

View Code

Toggle Theme



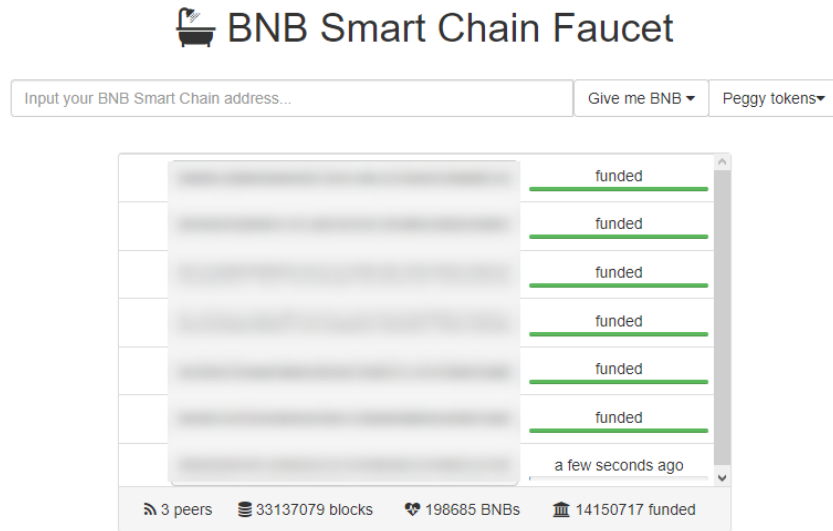
2. Search "**tBNB**" in the search bar and make sure to include testnets.



3. Find the **BNB Smart Chain Testnet Network** with the **ChainID of 97** and click on "**Add to MetaMask**" or a similar button to connect the network to your MetaMask wallet.
4. A MetaMask window will pop up asking for permission to connect. Click "**Connect**." Switch your network to the Binance Smart Chain Testnet before you proceed in this guide.

## Step 2: Add tBNB Tokens to Your Wallet

1. Open the **BNB Smart Chain: Faucet** website through this link: <https://testnet.bnbchain.org/faucet-smart>. You'll be doing a Captcha to prevent bots from accessing the website.



How does this work?

[BTC,ETH,XRP,BUSD,USDT,USDC,DAI](#) are issued as BEP20 token.

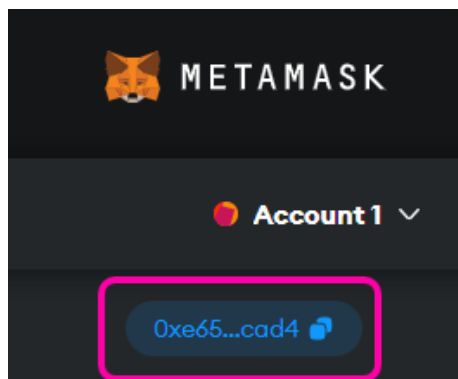
[Click to get detail about BEP20.](#)

Support Discord: [discord.gg/bnbchain](https://discord.gg/bnbchain)

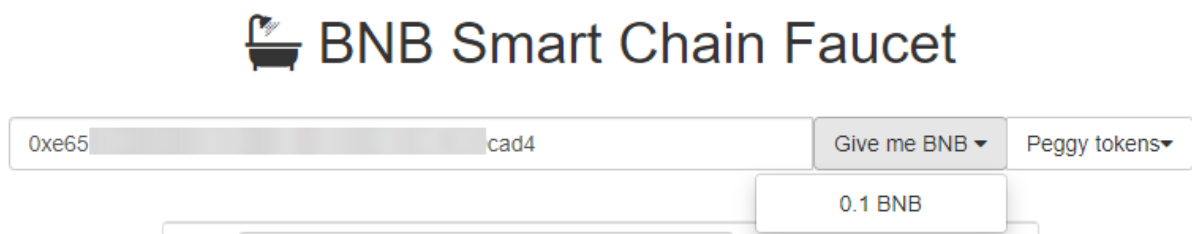
[The faucet is powered by OpenSea's testnet faucet bot.](#)

***\*tBNB** stands for "Testnet Binance Coin." It's a cryptocurrency related to the Binance blockchain used for testing and development purposes on their testnet network.*

2. Copy your MetaMask wallet address (inside the pink box) and paste it on to the BNB Smart Chain Faucet input field.

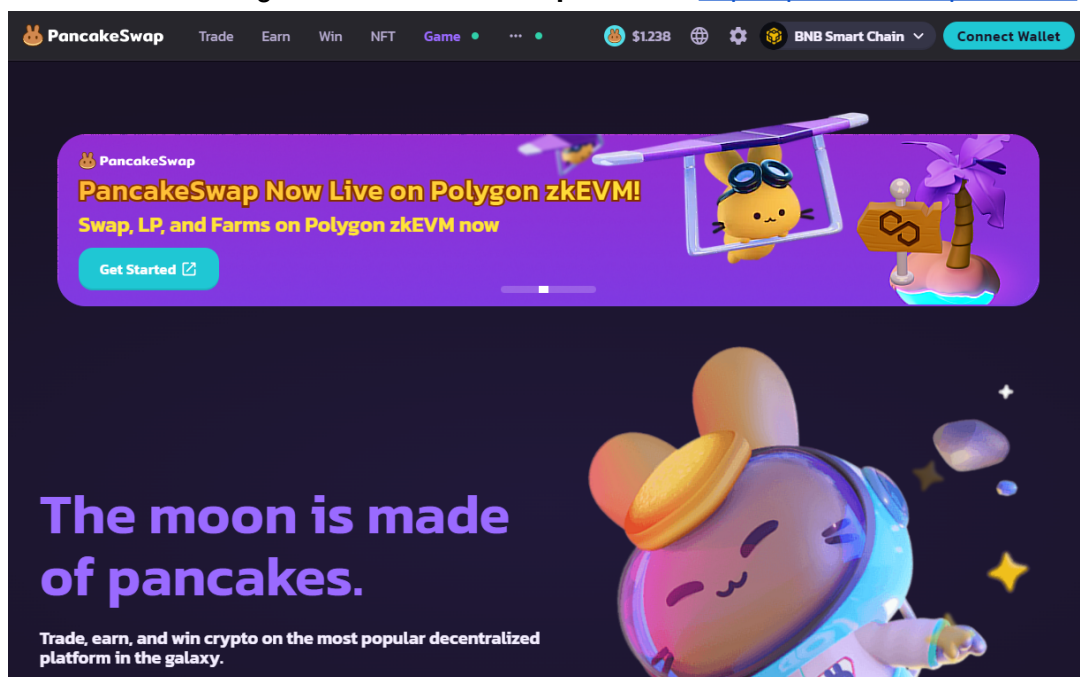


3. Click on the **"Give me BNB"** dropdown button and select **"0.1 BNB"**. Once you do, the faucet will process your request, and you should receive the test BNB tokens in your wallet shortly.

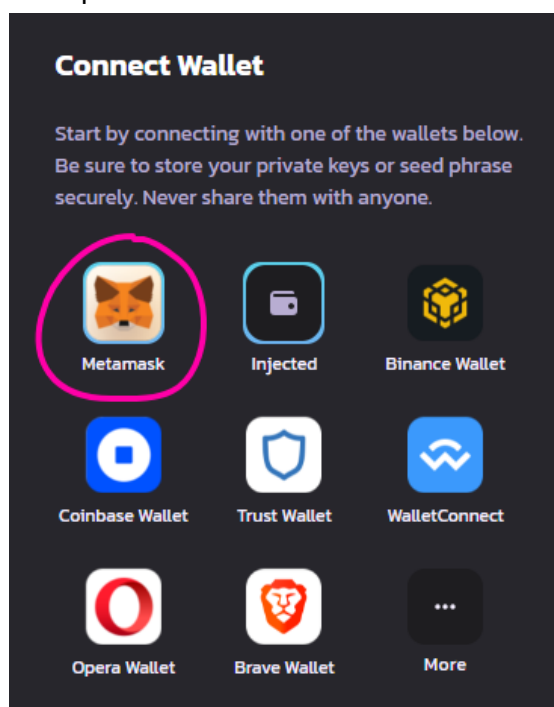


### Step 3: Access PancakeSwap and Connect Your MetaMask Wallet

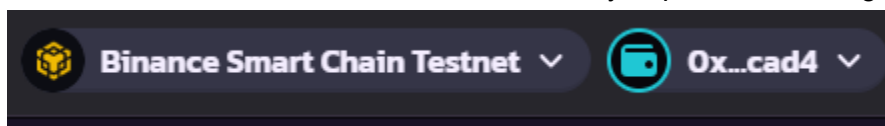
1. Open your web browser and go to the **PancakeSwap** website: <https://pancakeswap.finance/>.



2. Click on **"Connect Wallet"** on the PancakeSwap interface.
3. Choose **"MetaMask"** as your wallet provider.

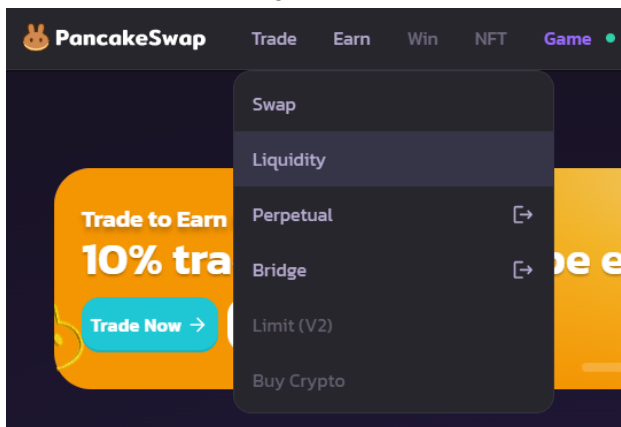


4. A MetaMask window will pop up asking for permission to connect. Click **"Connect."** Don't forget to switch your network to the Binance Smart Chain Testnet before you proceed in this guide.



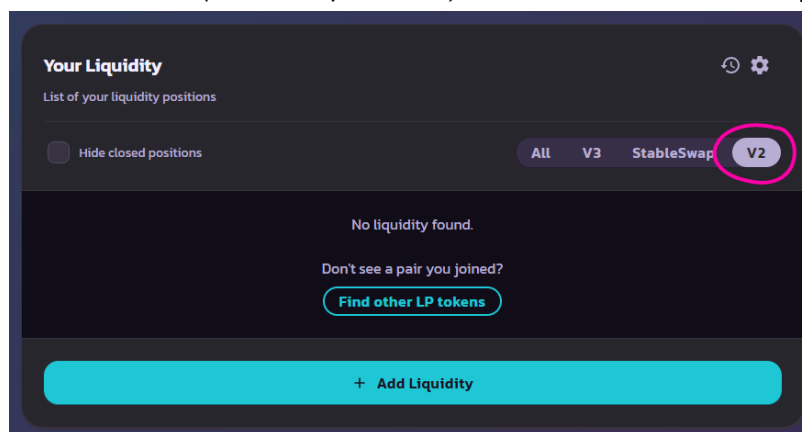
## Step 4: Adding Liquidity to Your Token

1. After connecting your MetaMask Wallet, hover over the "Trade" tab located in the top menu on the PancakeSwap interface. Then, click on "Liquidity."

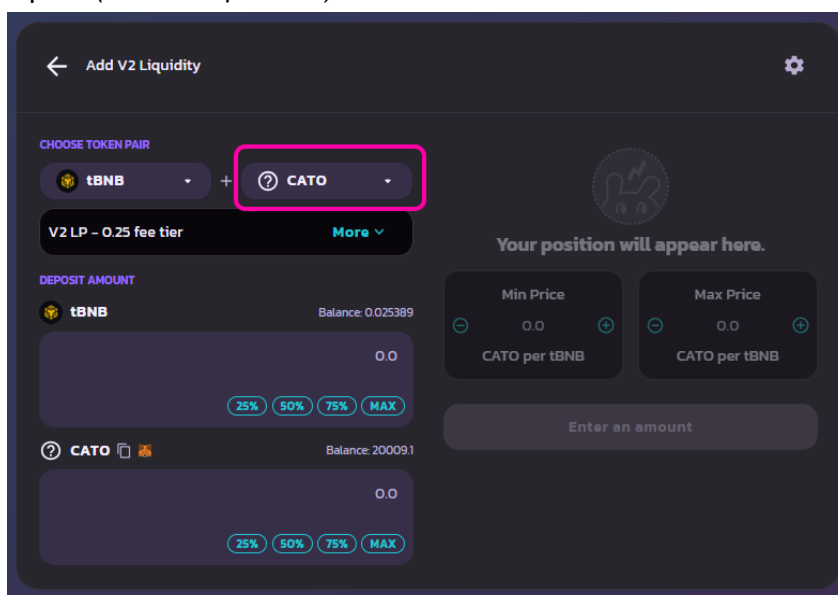


*\***Liquidity** is all about how easily you can trade a token without causing significant price movements. High liquidity is considered a positive characteristic for tokens because it provides traders with better execution and a more stable market.*

2. On the Liquidity page, click on "V2" (inside the pink circle) then click on the "Add Liquidity" Button.



3. In the "Add V2 Liquidity" interface, choose your token pair by selecting tBNB for the first part then your token on the second part. (inside the pink box)



4. Input the amount of your token you want to provide as liquidity. PancakeSwap will automatically calculate and display the equivalent amount of tBNB required, if not input the tBNB amount you wish to deposit. Make sure that you don't max out the tBNB in your wallet because it takes 24 hours to receive tBNB tokens again from the smart faucet.

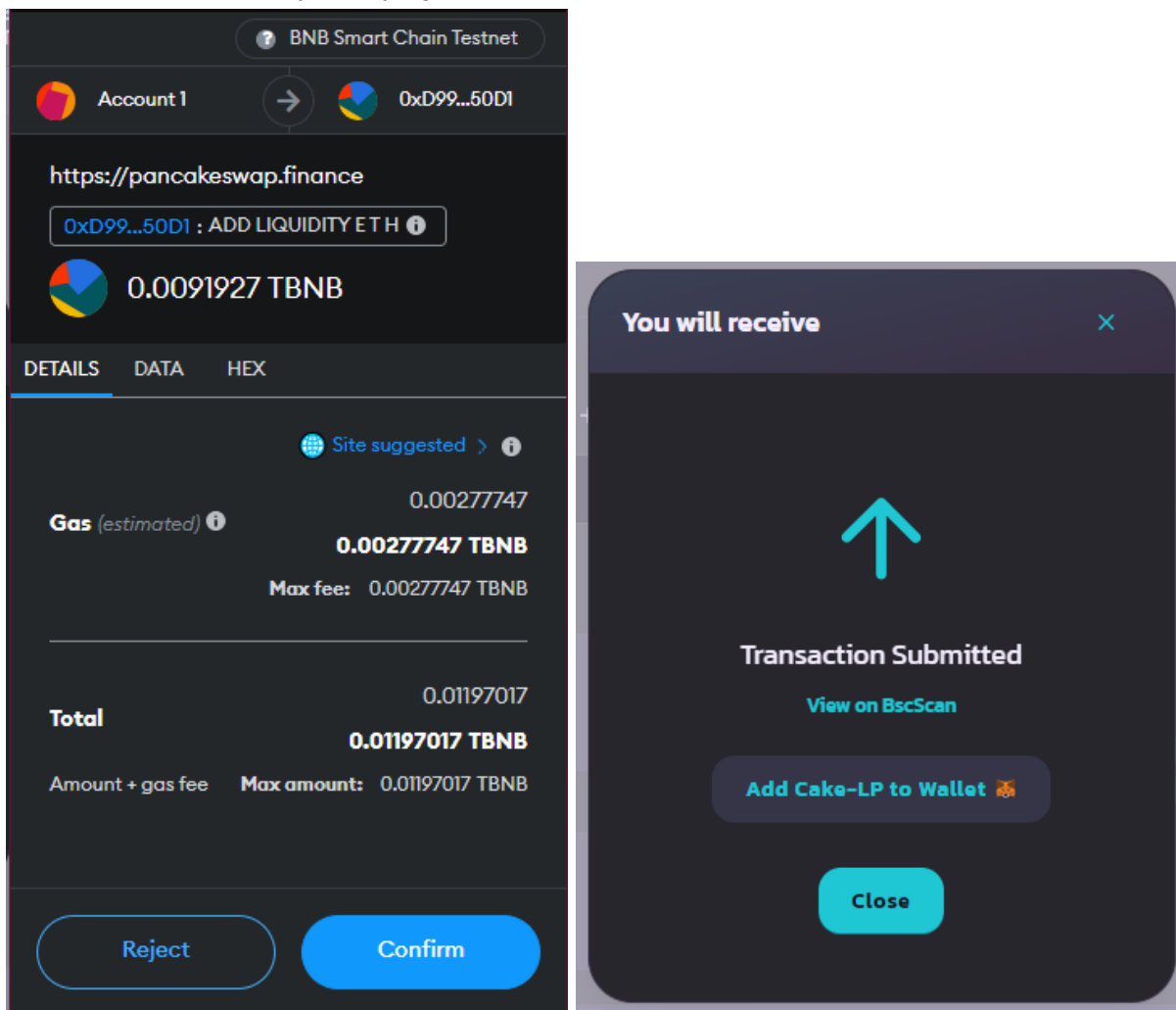
The image shows the 'DEPOSIT AMOUNT' screen on PancakeSwap. It has two sections. The top section is for tBNB, with a balance of 0.025389. The input field shows 0.0091927, and there are buttons for 25%, 50%, 75%, and MAX. The bottom section is for CATO, with a balance of 20009.1. The input field shows 1500, with a note '~141,276,534,990.85 USD' and the same percentage buttons.

5. Click “Add” and a window will pop up. This will show the details of the liquidity you're trying to add.

The image shows a confirmation window titled 'You will receive'. It contains the following information:

- YOU WILL RECEIVE:** tBNB-CATO LP, 3.71199
- YOUR SHARE IN THE PAIR:** 99.94%
- INPUT:** A pie chart showing two inputs: tBNB (0.0091927) and CATO (1500).
- RATES:** 1 tBNB = 163200 CATO, 1 CATO = 0.000006128 tBNB
- SLIPPAGE TOLERANCE:** 0.5%
- Confirm Supply:** A large red button at the bottom.

6. A MetaMask window will pop up asking for confirmation. Click “**Confirm**” to proceed. A window in PancakeSwap is then displayed saying that the transaction has been submitted, click “**Close**” on this.



7. You will see the updated balance of the tBNB and your tokens in your wallet. This means you have successfully added liquidity to your token. **Happy Minting!**

