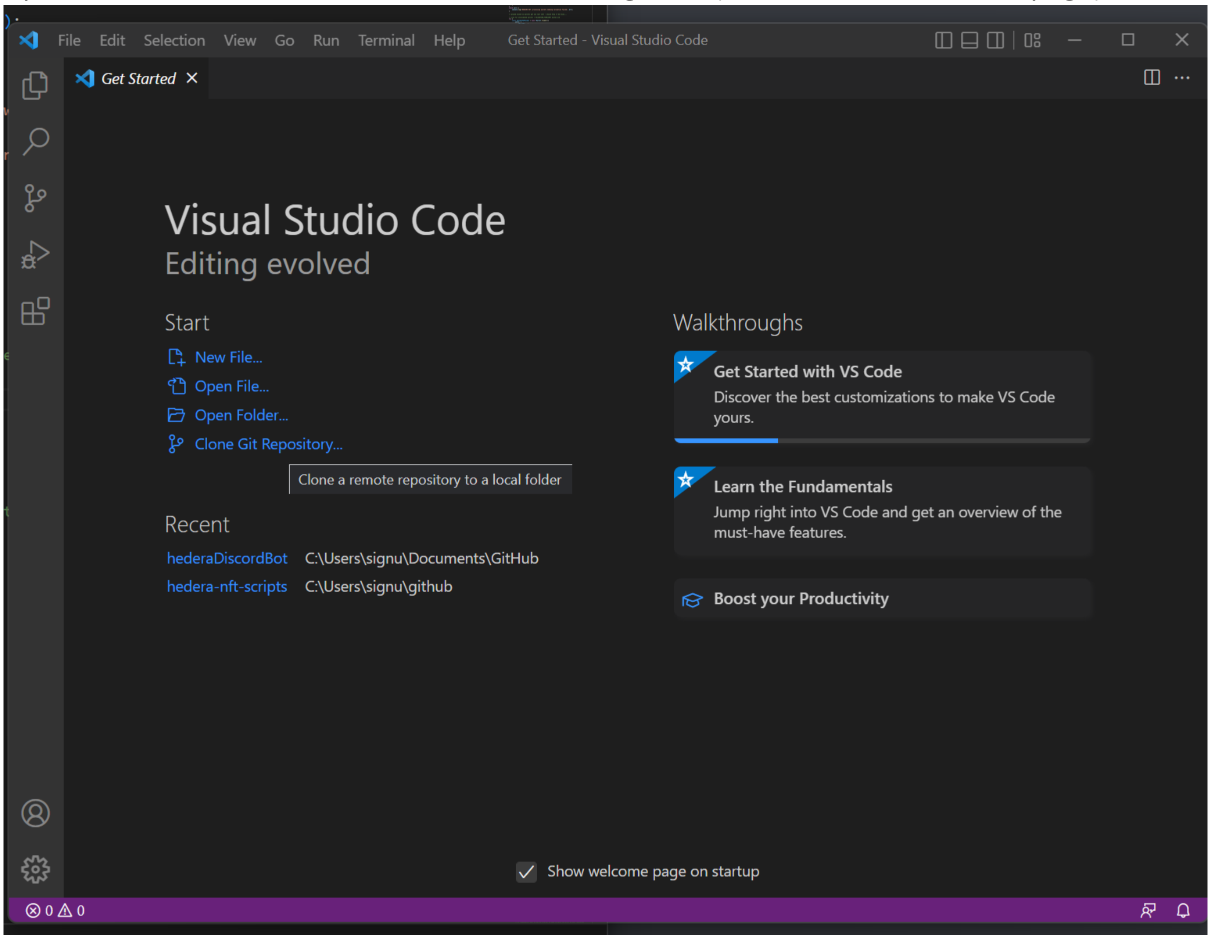
Install:

1) Visual Studio Code - https://code.visualstudio.com/Download
2) Node JS - https://nodejs.org/en/
3) NFT STORAGE API KEY

   https://nft.storage/

   Signup get the API KEY

4) Open VS Code and clone the latest source from github (middle of the welcome page)



Visual Studio Code

Editing evolved

Start

New File...

Open File...

Open Folder...

Clone Git Repository...

Clone a remote repository to a local folder

Recent

hederaDiscordBot C:\Users\signu\Documents\GitHub

hedera-nft-scripts C:\Users\signu\github

Walkthroughs

Get Started with VS Code

Discover the best customizations to make VS Code yours.

Learn the Fundamentals

Jump right into VS Code and get an overview of the must-have features.

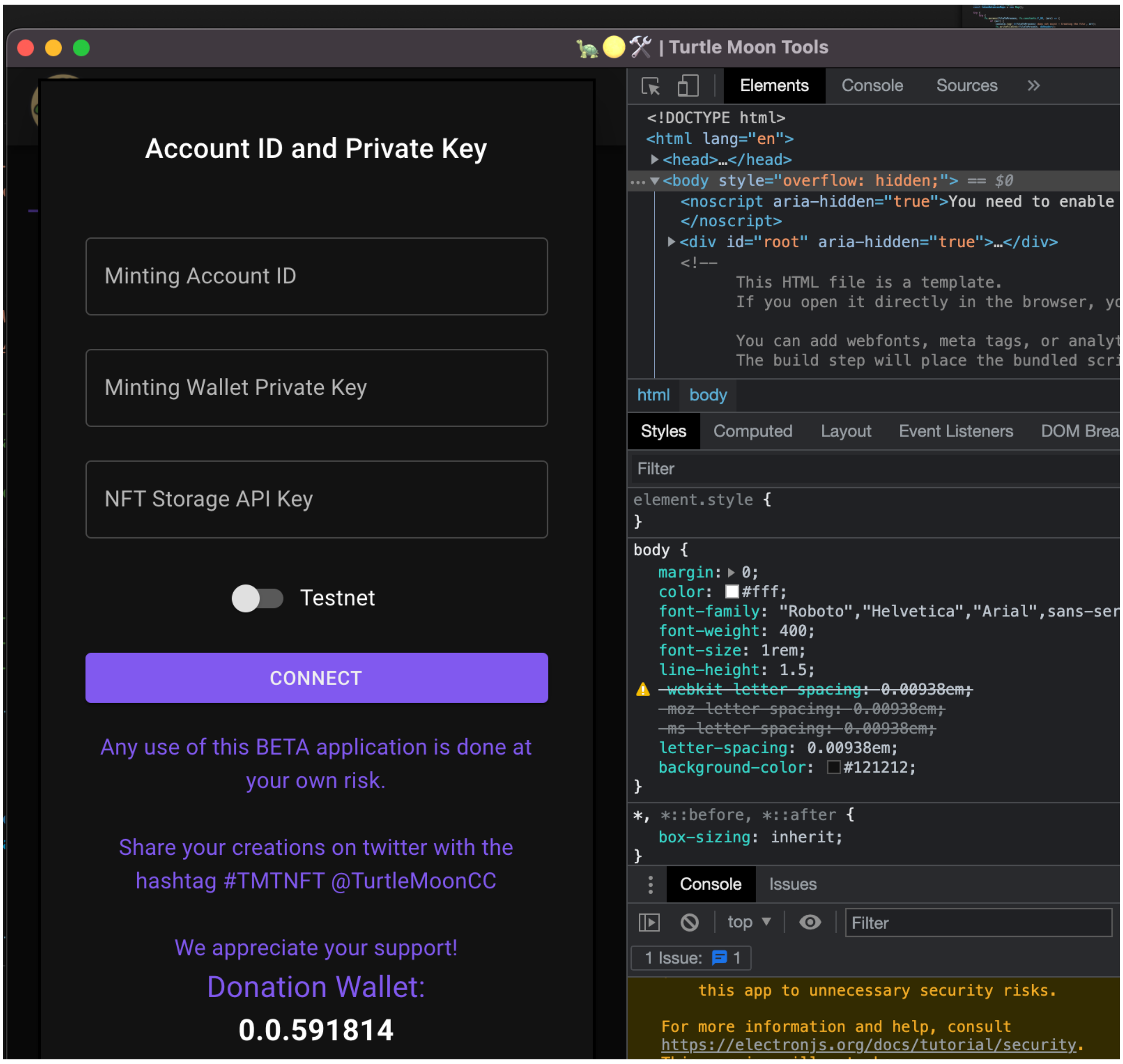
Boost your Productivity

Show welcome page on startup

Repository-> https://github.com/HGraphPunks/turtle-moon-tools-minting

5) VS Studio open a terminal window -> terminal -> new terminal
6) Type: npm install
7) Type: npm run electron:start

If you see something like this… you are golden



Account ID and Private Key

Minting Account ID

Minting Wallet Private Key

NFT Storage API Key

Testnet

CONNECT

Any use of this BETA application is done at your own risk.

Share your creations on twitter with the hashtag #TMTNFT @TurtleMoonCC

We appreciate your support!

Donation Wallet:

0.0.591814

```
<!DOCTYPE html>
<html lang="en">
 ▸<head>…</head>
…▾<body style="overflow: hidden;"> == $0
    <noscript aria-hidden="true">You need to enable
    </noscript>
   ▸<div id="root" aria-hidden="true">…</div>
    <!--
          This HTML file is a template.
          If you open it directly in the browser, yo

          You can add webfonts, meta tags, or analyt
          The build step will place the bundled scri
html  body
```

```
element.style {
}
body {
   margin: ▸ 0;
   color: #fff;
   font-family: "Roboto","Helvetica","Arial",sans-ser
   font-weight: 400;
   font-size: 1rem;
   line-height: 1.5;
   ~~-webkit-letter-spacing: 0.00938em;~~
   ~~-moz-letter-spacing: 0.00938em;~~
   ~~-ms-letter-spacing: 0.00938em;~~
   letter-spacing: 0.00938em;
   background-color: #121212;
}
*, *::before, *::after {
   box-sizing: inherit;
}
```

```
        this app to unnecessary security risks.

    For more information and help, consult
    https://electronjs.org/docs/tutorial/security.
```

Errors:

If you get this:

```
[0]   opensslErrorStack: [ 'error:03000086:digital envelope routines::initialization error' ],
[0]   library: 'digital envelope routines',
[0]   reason: 'unsupported',
[0]   code: 'ERR_OSSL_EVP_UNSUPPORTED'
[0] }
[0]
[0] Node.js v17.3.1
```

It is due to deprecation of SSL libraries in Node v17.X. Either downgrade to Node 16.X or edit your package.json:

from -> "start": "craco start",

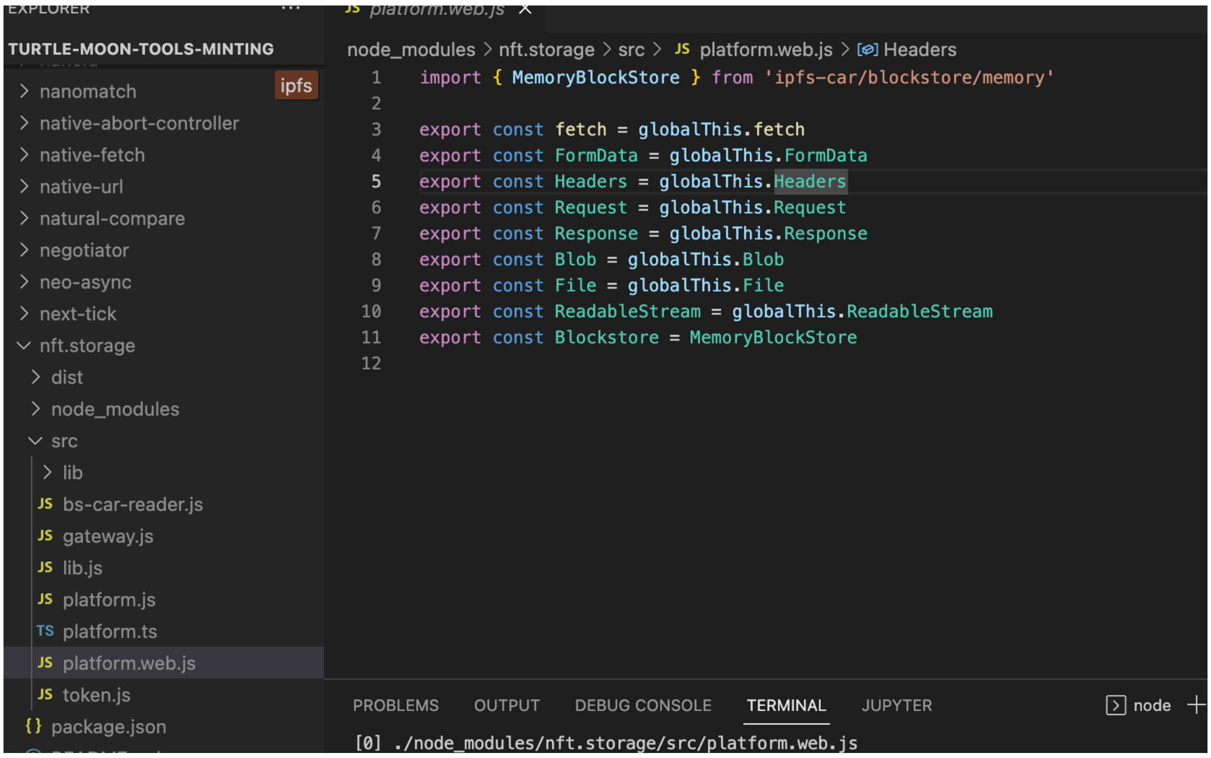
change to:

"start": "craco --openssl-legacy-provider start",

If you get this:

```
[1] (Use Electron --trace-warnings ... to show where the warning was created)
[0] Failed to compile.
[0]
[0] ./node_modules/nft.storage/src/platform.web.js
[0] Module not found: Can't resolve 'ipfs-car/blockstore/memory' in
'/Users/fsoumagnas/Desktop/turtle-moon-tools-minting/node_modules/nft.storage/src'
```

Then you need this dirty hack….



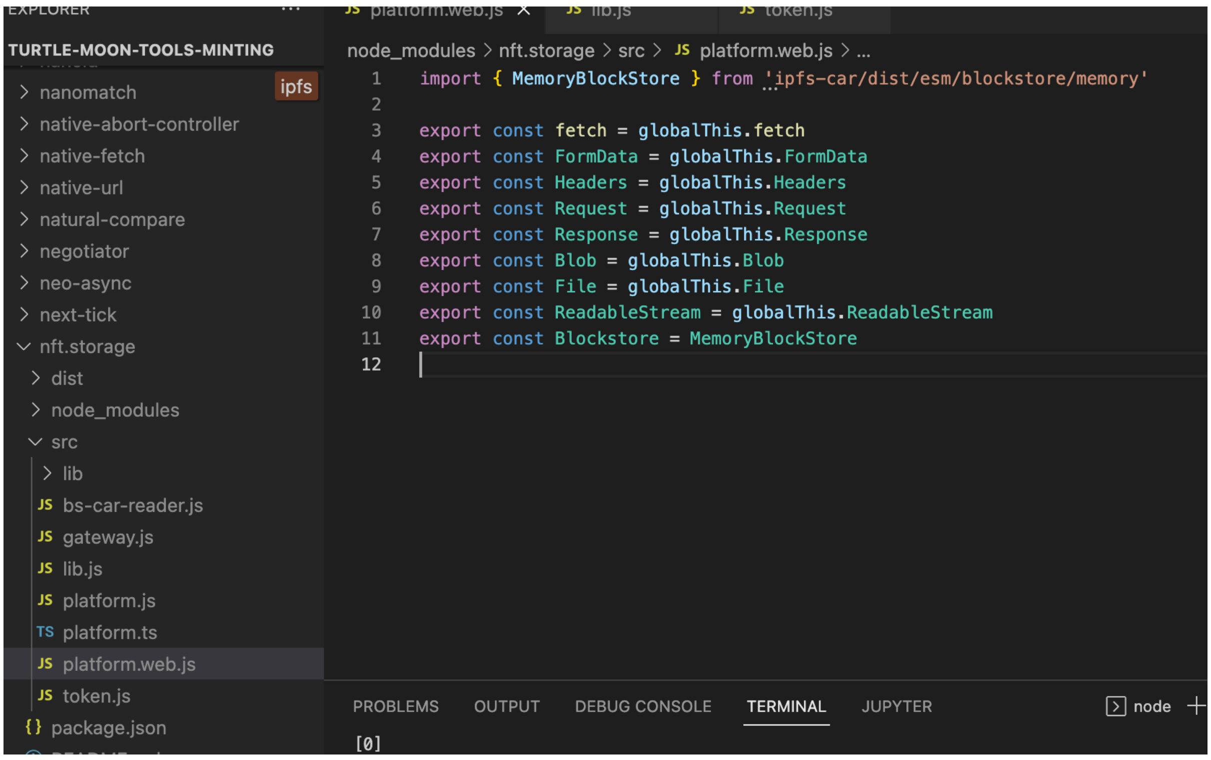
node_modules > nft.storage > src > platform.web.js > Headers

```js
import { MemoryBlockStore } from 'ipfs-car/blockstore/memory'

export const fetch = globalThis.fetch
export const FormData = globalThis.FormData
export const Headers = globalThis.Headers
export const Request = globalThis.Request
export const Response = globalThis.Response
export const Blob = globalThis.Blob
export const File = globalThis.File
export const ReadableStream = globalThis.ReadableStream
export const Blockstore = MemoryBlockStore
```

```
[0] ./node_modules/nft.storage/src/platform.web.js
```

Inside platform.web.js, update to this: import { MemoryBlockStore } from 'ipfs-car/dist/esm/blockstore/memory'



```
import { MemoryBlockStore } from 'ipfs-car/dist/esm/blockstore/memory'

export const fetch = globalThis.fetch
export const FormData = globalThis.FormData
export const Headers = globalThis.Headers
export const Request = globalThis.Request
export const Response = globalThis.Response
export const Blob = globalThis.Blob
export const File = globalThis.File
export const ReadableStream = globalThis.ReadableStream
export const Blockstore = MemoryBlockStore

```

```
[0]
```

Inside lib.js, update to this: import { pack } from 'ipfs-car/dist/esm/pack'

Inside token.js, update to this: import { pack } from 'ipfs-car/dist/esm/pack'

Usage:

Enter the minting wallet ID / Private key and the NFT Storage API key. Click the slider if you want to mint on Mainnet (use mainnet credentials of course!) and click connect.

Bulk Minting the more complicated so we will tackle that generalised case.

Here is the (beautified) JSON from the mirror node as an example (highlighted fields are those you set when minting and described below):

http://testnet.mirrornode.hedera.com/api/v1/tokens/0.0.47540431

```
{
  "admin_key": null,
  "auto_renew_account": "0.0.2777997",
  "auto_renew_period": 7776000,
  "created_timestamp": "1656939337.160839003",
  "custom_fees": {
    "created_timestamp": "1656939337.160839003",
    "fixed_fees": [],
    "royalty_fees": [
      {
        "amount": {
          "numerator": 100,
          "denominator": 10000
        },
        "fallback_fee": {
          "amount": 300000000,
          "denominating_token_id": null
        },
        "collector_account_id": "0.0.26102195"
      }
    ]
  },
  "decimals": "0",
  "deleted": false,
  "expiry_timestamp": null,
  "fee_schedule_key": null,
  "freeze_default": false,
  "freeze_key": null,
  "initial_supply": "0",
  "kyc_key": null,
  "max_supply": "100",
  "memo": "",
  "modified_timestamp": "1656939347.670488582",
  "name": "Airdrop Test Token",
  "pause_key": null,
  "pause_status": "NOT_APPLICABLE",
  "supply_key": {
    "_type": "ED25519",
    "key": "4e173f18ac4a2f9f99fafd6c82f6f77d4821e7ef4074962e0d396df45932239f"
  },
  "supply_type": "FINITE",
  "symbol": "ADT",
  "token_id": "0.0.47540431",
  "total_supply": "36",
  "treasury_account_id": "0.0.2777997",
  "type": "NON_FUNGIBLE_UNIQUE",
  "wipe_key": null
}
```

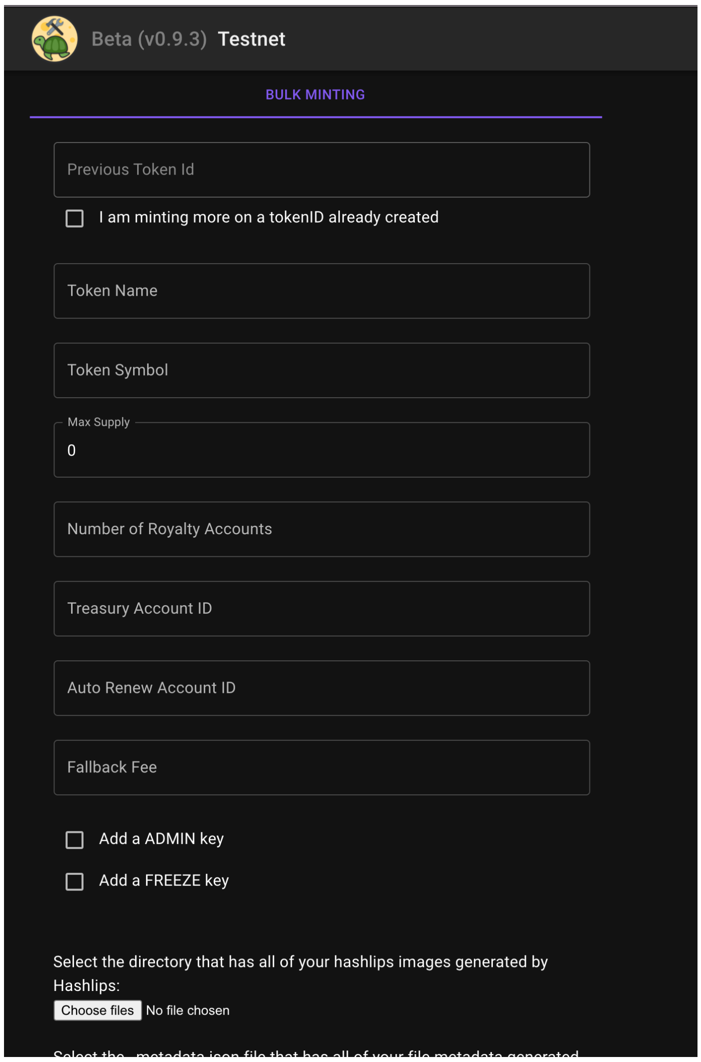
Previous token Id: if you are minting additional NFTs on an existing token (you will need to remember the supply key from the initial mint to achieve this) then tick the box and enter the token ID. If this is a virgin mint then ignore this.

Token Name/Symbol: Name of token / shorthand symbol

Max Supply: The maximum number of NFTs you want on this token ID ***[you do not need to mint them all right now, but you can't change this without an admin key so best to get it right upfront! Per the above example I minted 36 but have a max supply of 100].***

Number of royalty accounts: increment for the accounts required. N.B. royalties can be paid to other accounts apart from the treasury with no issue (as in example above). For each you will have to set the %, the account and the fallback. If you add as fallback fee the NFT will need to be signed by both sides of a trade (or a hashpack secure trade) unless one side of the trade is the treasury / royalty receiving account.

Treasury Account / Auto Renew Account: Typically the minting wallet you specified before connecting.


Beta (v0.9.3) Testnet

BULK MINTING

Previous Token Id

☐ I am minting more on a tokenID already created

Token Name

Token Symbol

Max Supply
0

Number of Royalty Accounts

Treasury Account ID

Auto Renew Account ID

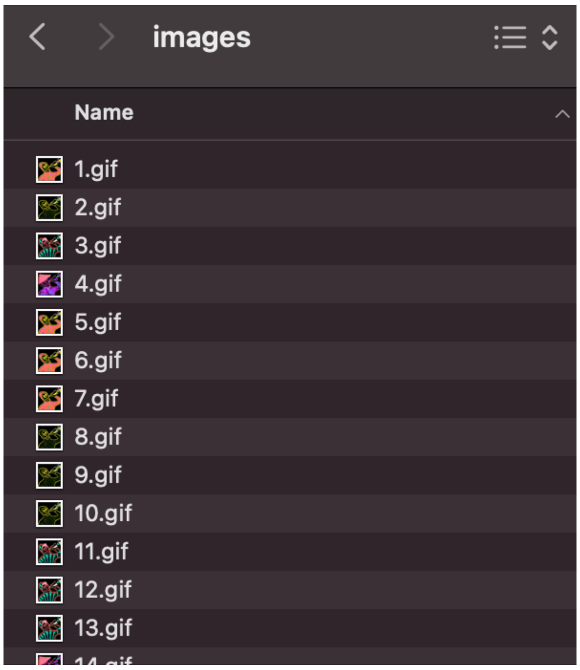
Fallback Fee

☐ Add a ADMIN key

☐ Add a FREEZE key

Select the directory that has all of your hashlips images generated by Hashlips:
Choose files No file chosen

Now choose the files for the mint. Each token to be minted should have a name matching the name set in the metadata file. As an example, 1.gif -> 36.gif below.



1.gif
2.gif
3.gif
4.gif
5.gif
6.gif
7.gif
8.gif
9.gif
10.gif
11.gif
12.gif
13.gif

Now create your metadata. I called it _metadata.json to match the name shown in the app. I have reproduced an example with just 2 JSON objects representing tokens but the file I used had 36 (one for each NFT):

I have not tested which fields can be removed but I know the 'name' is 100% required.

Be careful to ensure the edition and the image file name align.

The date is an epoch date. I suspect unnecessary but not debugged to prove. Go to https://www.epochconverter.com click 'human date to timestamp' then copy/paste tge timestamp in milliseconds for this field.

Attributes -> metadata goes here if you are adding any.

- Please be consistent in spelling / capitalisation / names.
- Please ensure if you use a trait_type in one NFT use it for all.
- Separate JSON objects inside the [] with commas
- https://jsonbeautifier.org is handy to check for errors / layout or VS Code (editor of your choice) will help.

This metadata format is not HIP-412 compliant multi-file NFT format (I am simply giving a guide to bootstrap TMT) however it is entirely functional for the ecosystem.

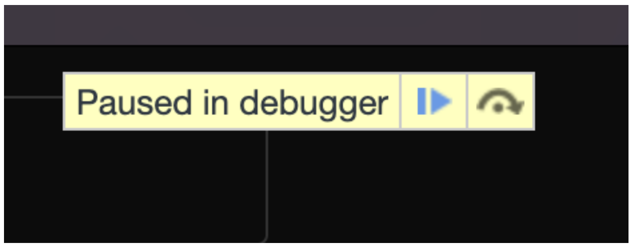
```
[
 {
  "creator": "Stowerling",
  "category": "Digital Art",
  "name": "Airdrop Test Token",
  "description": "Dummy token for testing the airdrop script",
  "image": "ipfs://LinkIsRepleacedWhenUploadingWithTurtleMoonTools/1.gif",
  "date": 1656916771000,
  "edition": 1,
  "properties": {
    "extras": []
  },
  "attributes": [
    {
      "trait_type": "type",
      "value": "RektGuy"
    },
    {
      "trait_type": "month",
      "value": "July"
    },
    {
      "trait_type": "year",
      "value": "2022"
    }
  ],
  "compiler": "Turtle Moon Tools"
},
{
  "creator": "Stowerling",
  "category": "Digital Art",
  "name": "Airdrop Test Token",
  "description": "Airdrop script test token",
  "image": "ipfs://LinkIsRepleacedWhenUploadingWithTurtleMoonTools/2.gif",
  "date": 1656916771000,
  "edition": 2,
  "properties": {
    "extras": []
  },
  "attributes": [
    {
      "trait_type": "type",
      "value": "RektGuy"
    },
    {
      "trait_type": "month",
      "value": "July"
    },
    {
      "trait_type": "year",
      "value": "2022"
    }
  ],
  "compiler": "Turtle Moon Tools"
  }
]
```

Mint

Double check your work for spelling mistakes!

‘Mint Collection’ button.

Currently there is a debugger set to kick in so you may see this:



If so, press the play button.

Sit back and watch…fingers crossed it will complete and give you back the token ID *AND* the supply key.

Protect that supply key as if it were your private key. This key will allow you to mint additional NFTs (up to max supply you set) and burn NFTs on this token.