

zkCarbonOffset @LambdaWeek.hackathon

zero-knowledge proofs of carbon offset

A dark blue diagonal gradient bar that starts from the bottom left and extends towards the top right, covering the lower half of the slide.

Prove the world your Carbon Footprint Offset

You computed your carbon footprint:

- Plane tickets to HackerHouses: 133.7 kg CO₂
- Tweeting about new projects: 420g CO₂
- Being BRC-20 wizard: >9000kg CO₂
- Breathing for one lambda week: 6.9 kg CO₂

You have a balance of Carbon Credits

The Build

Using Mina zero-knowledge framework:

- Contract privately mints you balance (assuming you paid for tokens)
- Create a zkApp implementing an ERC20-like token
- Additional method: offsetTokens
 - burn tokens
 - create a leaf in a MerkleMap

Future work

Right now: only **passing** testing

Next step:

- Deployed contract
- Front-end
- Back-end: getting Merkle Paths proofs

End goal

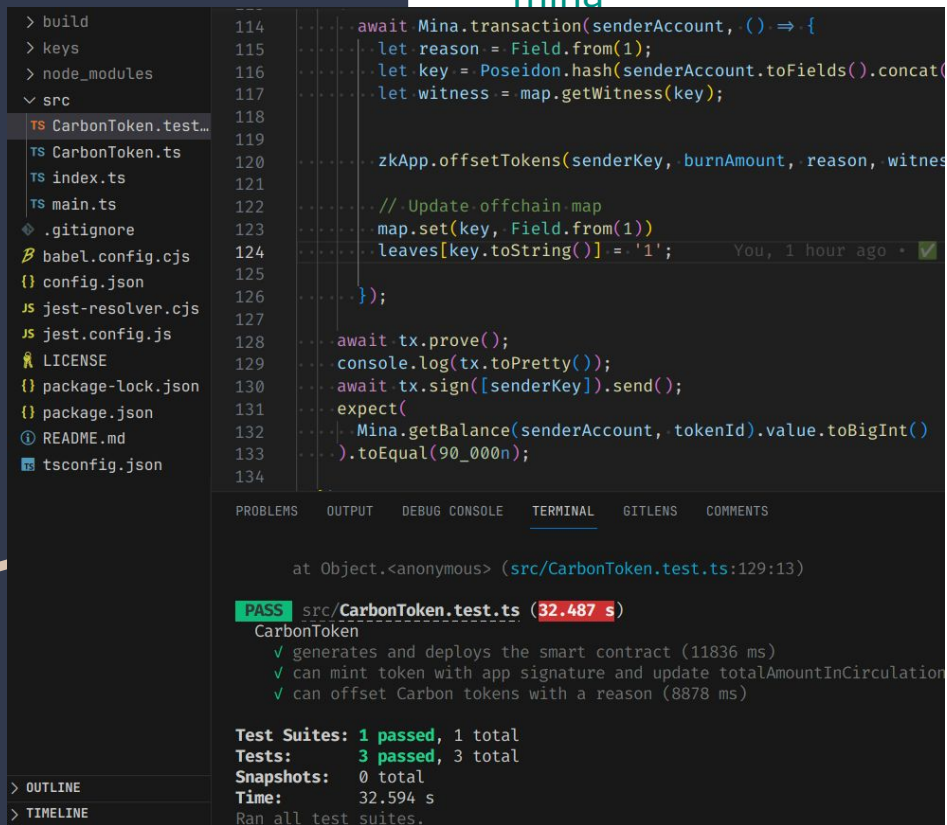
INPUT: User Footprint

- Plane tickets to HackerHouses: 133.7 kg CO2
- Tweeting about new projects: 420g CO2
- Being BRC-20 wizard: >9000kg CO2
- Breathing for one lambda week: 6.9 kg CO2

Output: QRcode link to Compensation Proofs checkable by anyone.

Code/Demo

<https://github.com/tekkac/zk-carbon-offset>
-mina



```
> build
> keys
> node_modules
v src
  TS CarbonToken.test.ts
  TS CarbonToken.ts
  TS index.ts
  TS main.ts
  .gitignore
  babel.config.cjs
  config.json
  jest-resolver.cjs
  jest.config.js
  LICENSE
  package-lock.json
  package.json
  README.md
  tsconfig.json
```

```
114     ...await Mina.transaction(senderAccount, () => {
115     ...     let reason = Field.from(1);
116     ...     let key = Poseidon.hash(senderAccount.toFields().concat(
117     ...     let witness = map.getWitness(key);
118
119
120     ...     zkApp.offsetTokens(senderKey, burnAmount, reason, witness);
121
122     ...     // Update offchain map
123     ...     map.set(key, Field.from(1))
124     ...     leaves[key.toString()] = '1';
125
126     ... });
127
128     ...await tx.prove();
129     ...console.log(tx.toPretty());
130     ...await tx.sign([senderKey]).send();
131     ...expect(
132     ...     Mina.getBalance(senderAccount, tokenId).value.toBigInt()
133     ...     ).toEqual(90_000n);
134
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL GITLENS COMMENTS

at Object.<anonymous> (src/CarbonToken.test.ts:129:13)

PASS src/CarbonToken.test.ts (32.487 s)

CarbonToken

- ✓ generates and deploys the smart contract (11836 ms)
- ✓ can mint token with app signature and update totalAmountInCirculation
- ✓ can offset Carbon tokens with a reason (8878 ms)

Test Suites: 1 passed, 1 total
Tests: 3 passed, 3 total
Snapshots: 0 total
Time: 32.594 s
Ran all test suites.

> OUTLINE
> TIMELINE