Tools to Help *You* Rewire Music

OMI & COALA IP | IPDB | BigchainDB

OMI, New York, March 10

Trent McConaghy

How do you collect digital art?

= How can digital artists get compensated?

Jonathan Monaghan
Escape Pod, 2015


r/btc
MAGIC
Internet
Money
Join us

How do you collect digital art?

= How can digital artists get compensated?

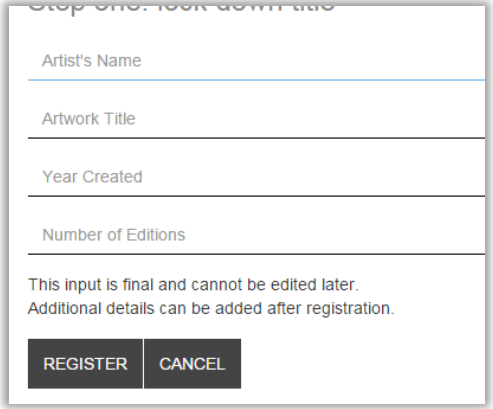
What if you could own digital art the way you own Bitcoin?

Jonathan Monaghan
Escape Pod, 2015

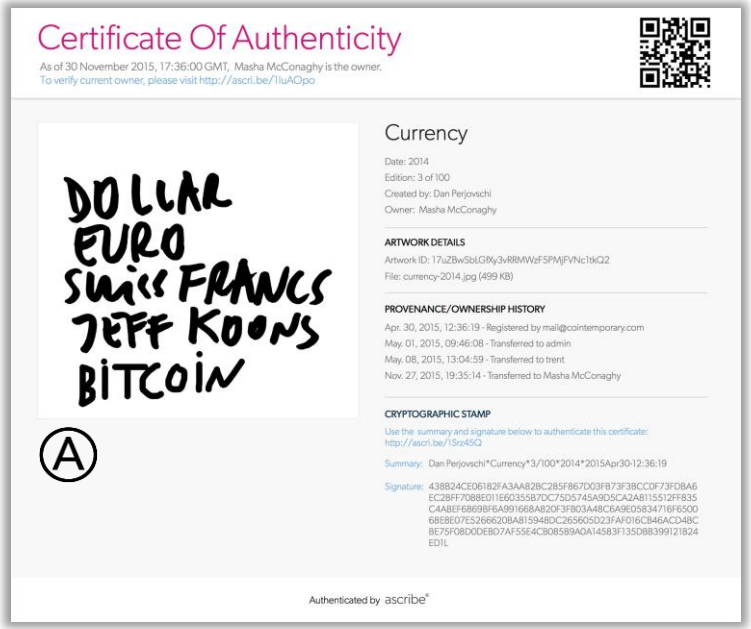
Register / license IP on shared substrate

How: ascribe.io, via blockchain & contract law

work



Register:
bind work + composition



License

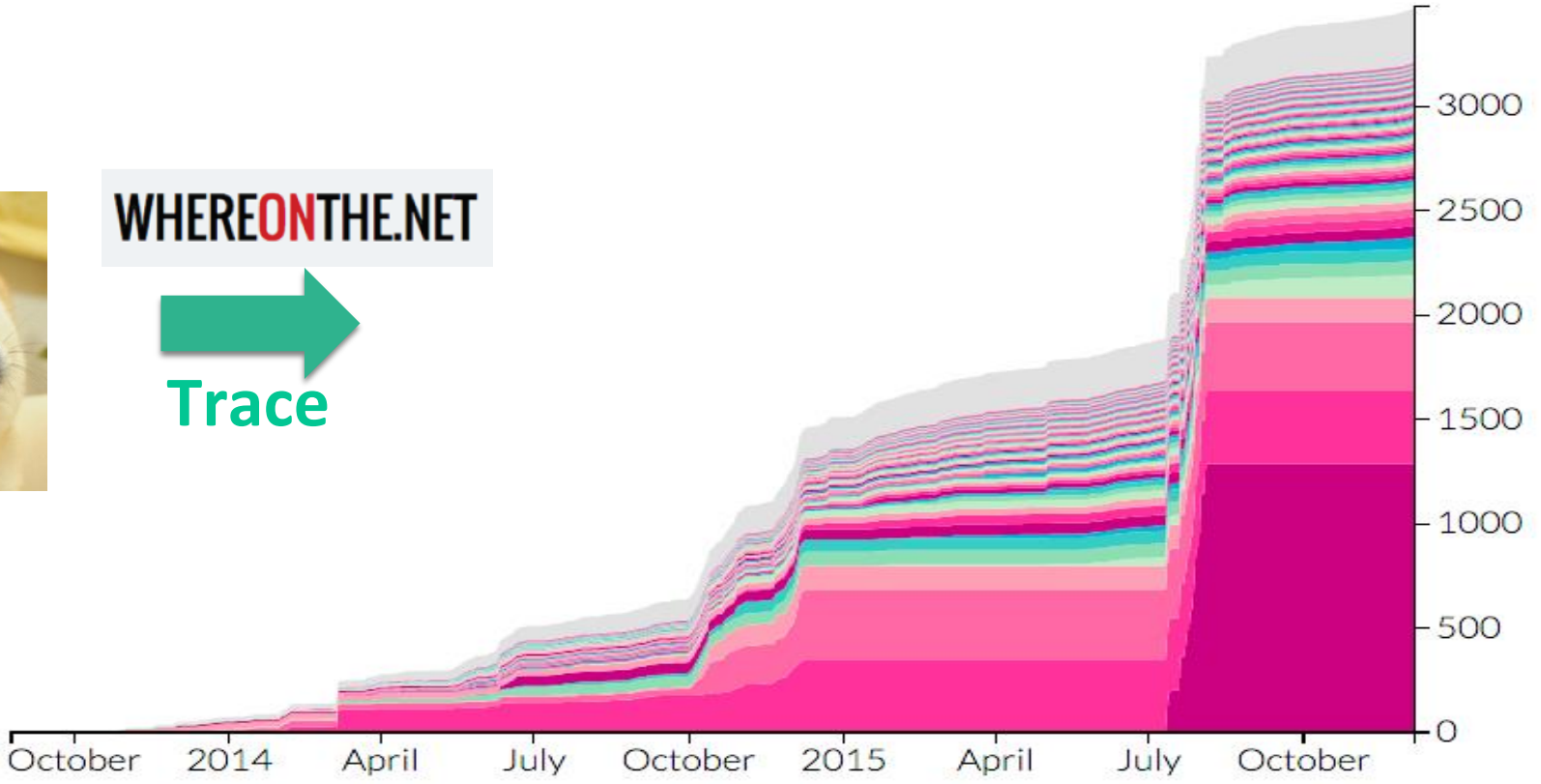
Trace when/where IP is used

How: whereonthe.net, via web crawl & AI

work

WHEREONTHE.NET

Trace

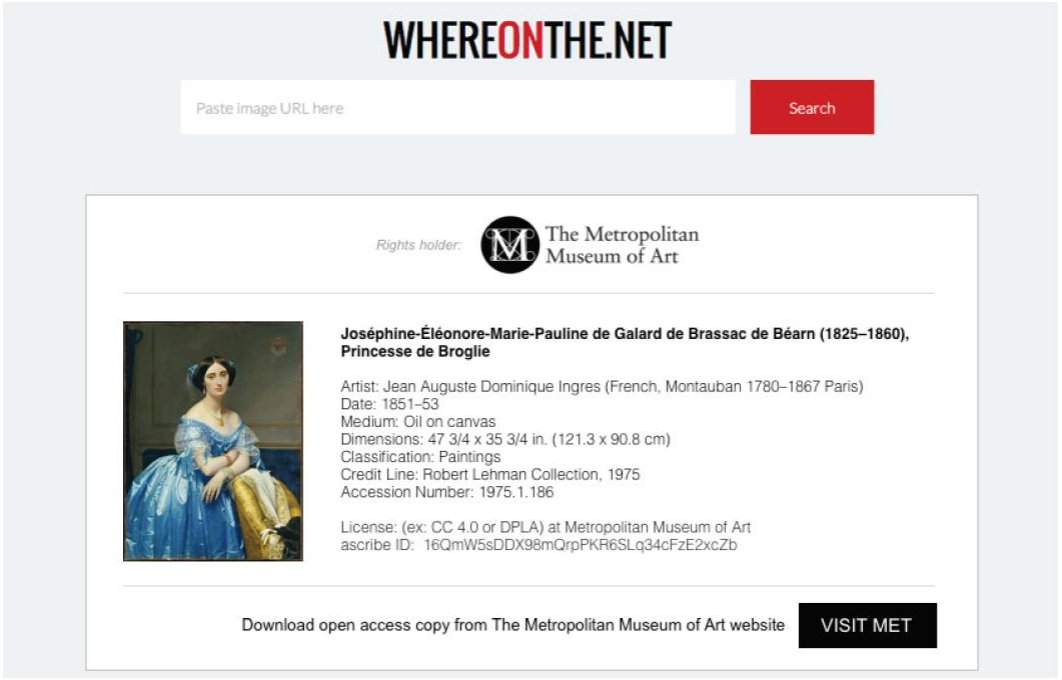


Recover metadata

How: ascribe IP registry x whereonthe.net ML

work

Check registry



metadata

Problems

1. **IP Licensing Flexibility**

2. **Scale**

Problems

1. IP Licensing Flexibility

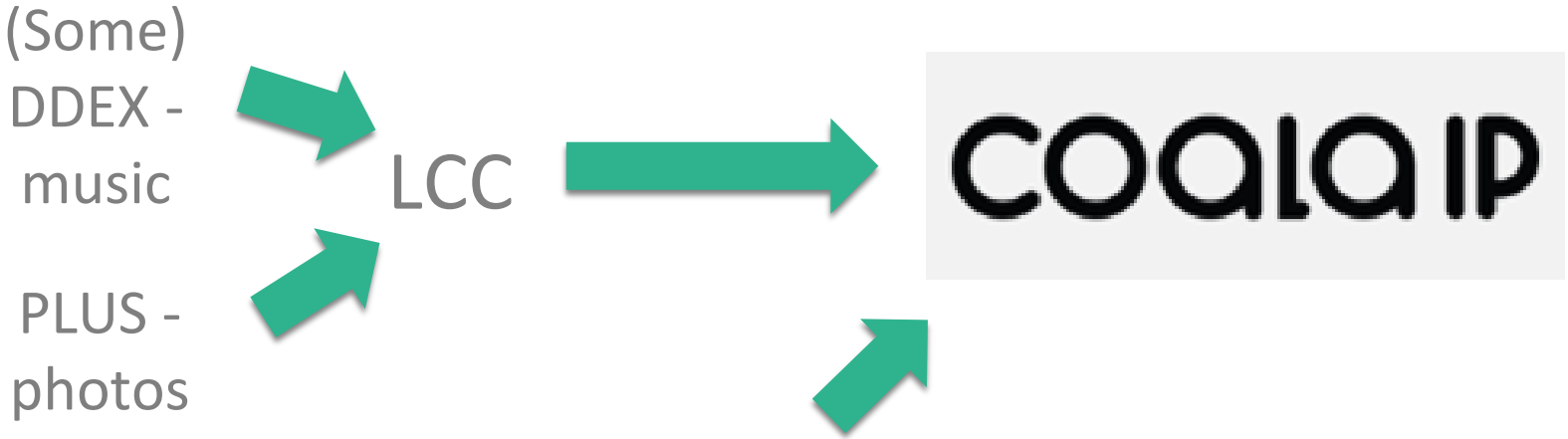
Tedious to add new IP use cases.

E.g. what about stills in a video? Fractional ownership?

2. Scale

Problem 1: Blockchain-ready flexible IP Licensing

Solution: COALA IP (A piece of MVI?)



Engineer in modern web & blockchain tech:

- JSON, IPLD, schema.org
- “own” = have private key

Community effort: COALA, ascribe/BDB, Ujo, IPFS, Mycelia, Resonate, jaak, ..

Iterations w LCC,DDEX,PLUS

COALA IP Example

Campbell's Soup Can (Tomato), 1964
Andy Warhol

Credit
© The Andy Warhol Foundation for the Visual Arts, Inc. / DACS

Medium
Synthetic polymer paint and silkscreen ink on canvas

Dimensions
36 x 24 inches

Cryptographic ID
2dd4aa50b557b99867c36fb107a28a82b...

Is this match incorrect? Please tell us!

License


```
{
  "id": "2dd4aa50b557b99867c36fb107a28a82be2fbf91ed6fad13d1e73cd9089c5fd4",
  "transaction": {
    "conditions": [
      {
        "cid": 0,
        "condition": {
          "details": {
            "bitmask": 32,
            "public_key": "J84XFuzgHeK3TmZs1Ls7YkjnzgT2UD6pgdbq31chJSVB",
            "signature": null,
            "type": "fulfillment",
            "type_id": 4
          },
          "uri": "cc:4:20:_mW5sp8SRZS_aqNBhD5ufgutrt2nctlVvScGh2dku-4:96"
        },
        "owners_after": [
          "J84XFuzgHeK3TmZs1Ls7YkjnzgT2UD6pgdbq31chJSVB"
        ]
      }
    ],
    "fulfillments": [
      {
        "fid": 0,
        "fulfillment": "cf:4:bJ6dDrSpaWNJ8JsscAwA_PLTV9iIHLStYwexn58clG0M",
        "owners_before": [
          "8K1JyfKKaxoYtwmLMQKJcbXAAC4hqcSHWryqNLBVPvih"
        ],
        "input": null
      }
    ],
    "operation": "CREATE",
    "timestamp": "1475077440",
    "version": 1,
    "data": {
      "payload": {
        "name": "Campbell's Soup Can (Tomato), 1964",
        "@context": [
          "<coalaip placeholder>",
          "http://schema.org/"
        ],
        "@id": "",
        "@type": "CreativeWork",
        "isManifestation": true,
        "manifestationOfWork": "57d60075431...377697d6a295fa8b",
        "url": "https://d1e2kq1i6pwuuf.cloudfront.net/production/6/1/6157-842.jpg"
      },
      "uuid": "d4fa5ce2-bdbb-40ce-8e0f-4e0a2e995bd7"
    }
  }
}
```

Problems

1. IP Licensing Flexibility

2. Scale

Bitcoin limited in throughput, capacity, latency.

No other blockchains scaled either.

But… big data scales…

Problem 2: scalable blockchain storage

Solution: BigchainDB

Start with a traditional

“big data” database

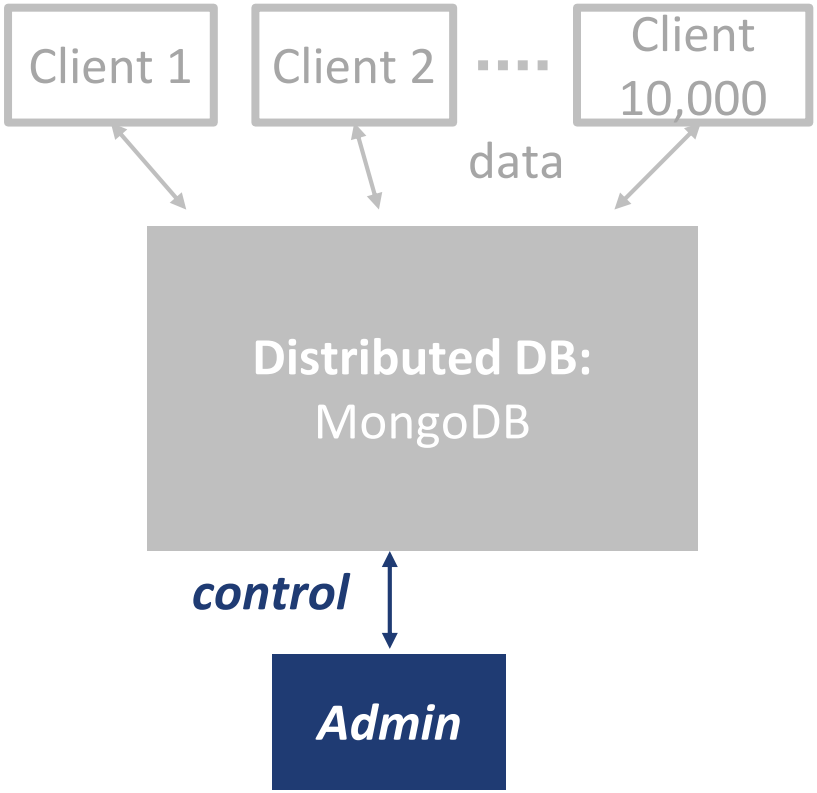
- Scale
- Querying

BIGCHAIN DB

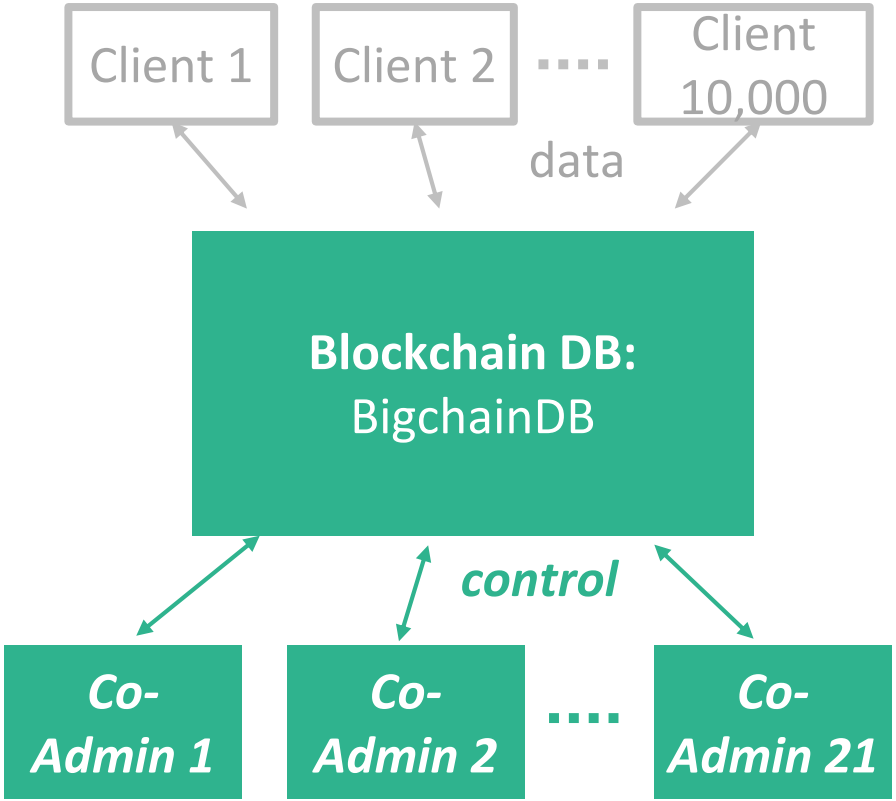
Best of both worlds

Engineer in **blockchain** characteristics:

- Decentralized control (co-admin)
- Immutable (audit trails)
- Assets (“own” = have private key)



A traditional database has
one administrator
(centralized)



A blockchain database has
many administrators
(decentralized)

Problem 2a: need a shared *public* database

Solution: IPDB network running BDB software

- Anyone can write to & read from
- With privacy controls

- Governed as a non-profit foundation with *caretakers*: Internet Archive, Open Media Foundation, COALA, …

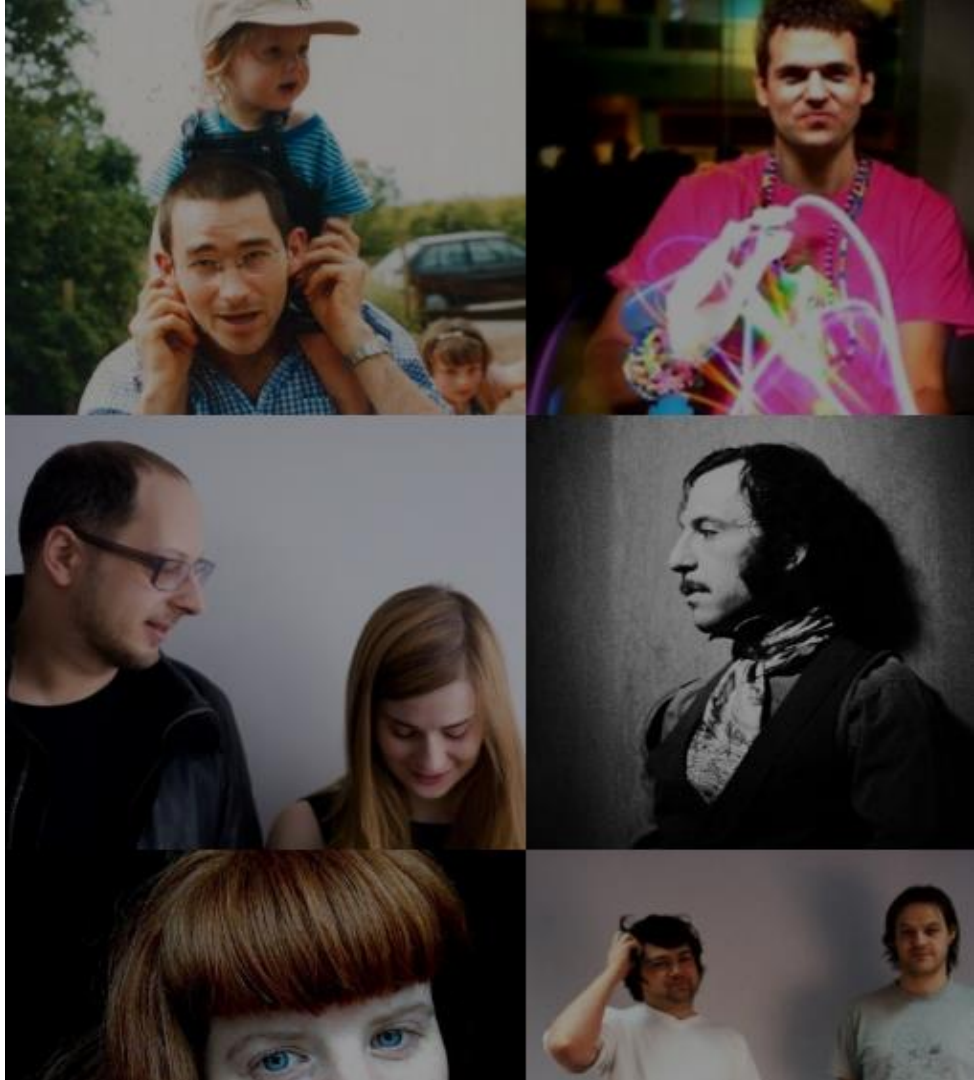
IPDB Interplanetary Database

Example BigchainDB & IPDB Usage

res( )nate

Vertical:
IP – Music rights

Value proposition:
A streaming service owned by all



Authenteq

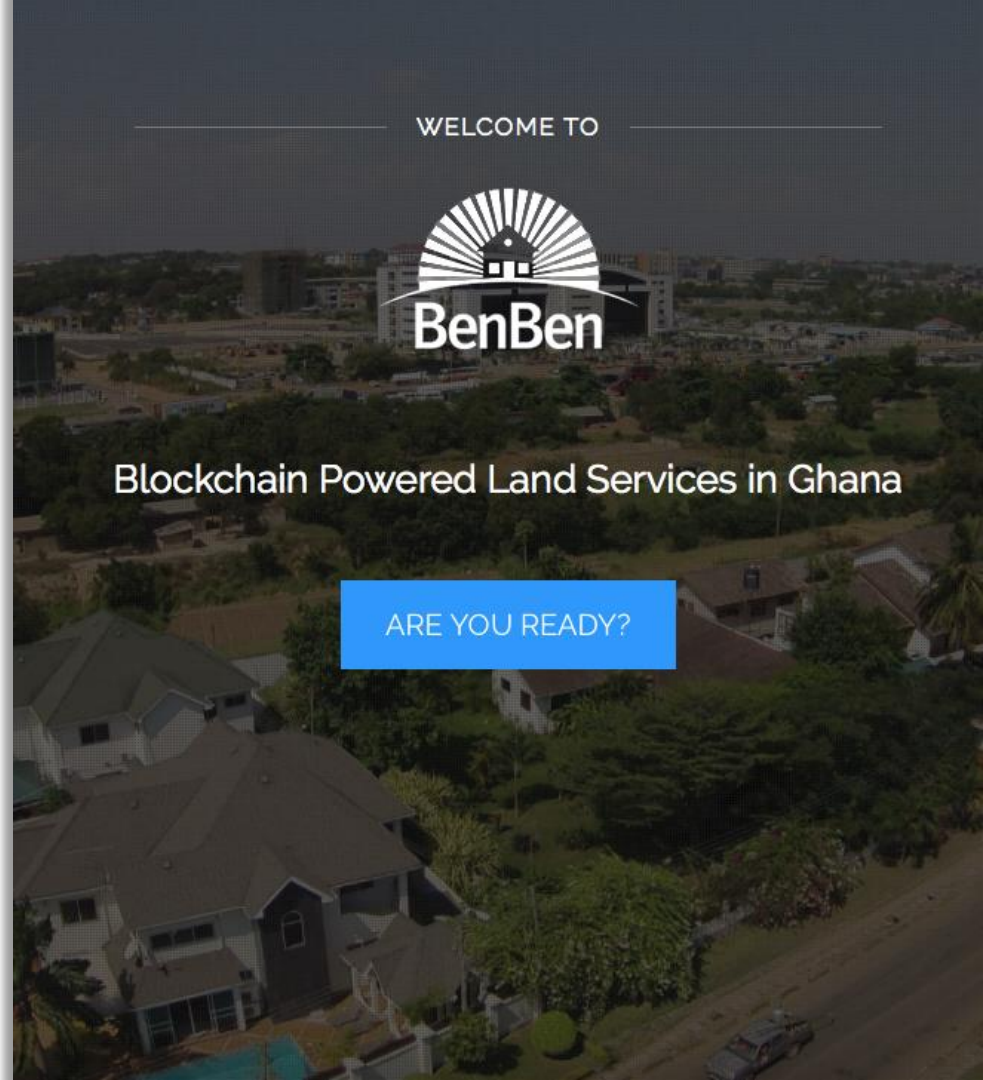
Vertical:
Identity

Value proposition:
Low-friction assurance, sovereign personal dat

BenBen

Vertical:
Government – Land Registry

Value proposition:
Low-cost registry, less risk of corruption



Recruit

Vertical:
ID - Education Credentials

Value proposition:
reduce fraudulent degrees, lower HR friction

RWE

Vertical:
Energy

Value proposition:
manage $ flow in energy deregulation

Tangent90

Vertical:
Supply Chain / Health

Value proposition:
government-mandated
transparent $ flow

A few more (with COALA IP)

Backend music platform - Envoke

Interoperability among networks – Interledger / Ripple

Supply chain / IoT – Innogy and R&C

French govt creative archives - BOKK

Luxury goods provenance - <coming soon>

Financial Infrastructure – 17 POCs with one partner

National identity system (>10M people) - <coming soon>

Loyalty and Reward System – CapGemini

Voting – SettleMint

Fix social media filter bubble – <coming soon>

Personal data consent – <coming soon>

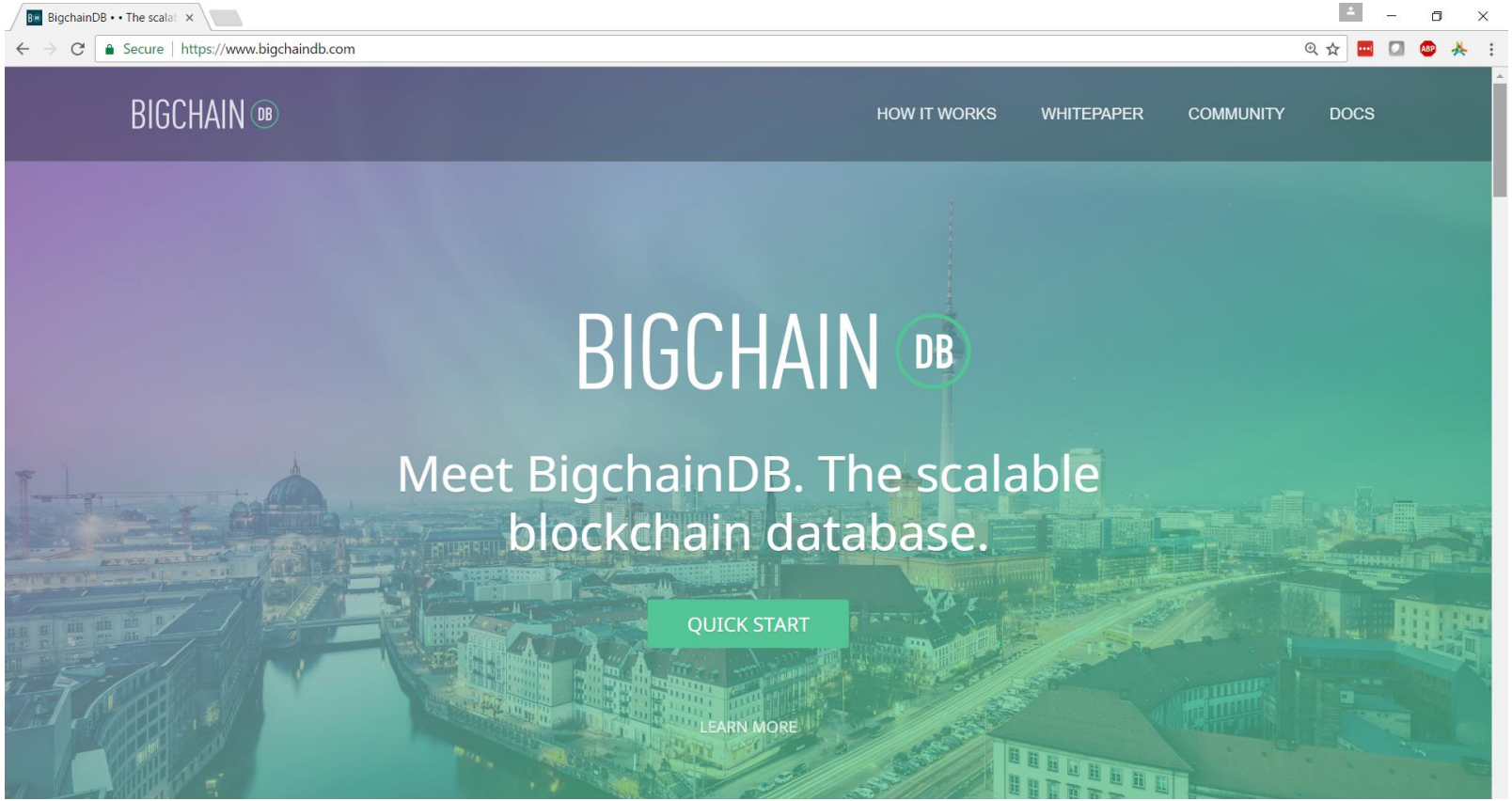
<<waiting list on IPDB – 200+ of orgs>>

Using BigchainDB / IPDB

On Deploying

No need to re-do the whole stack,
Just add one more (special) database.

To start: bigchaindb.com -> Quickstart



5.2. Create a Digital Asset

```
from bigchaindb import crypto

# Create a test user
testuser1_priv, testuser1_pub = crypto.generate_key_pair()

# Define a digital asset data payload
digital_asset_payload = {'msg': 'Hello BigchainDB!'}

# A create transaction uses the operation `CREATE` and has no inputs
tx = b.create_transaction(b.me, testuser1_pub, None, 'CREATE', payload=digital_

# All transactions need to be signed by the user creating the transaction
tx_signed = b.sign_transaction(tx, b.me_private)

# Write the transaction to the bigchain.
# The transaction will be stored in a backlog where it will be validated,
# included in a block, and written to the bigchain
b.write_transaction(tx_signed)
```

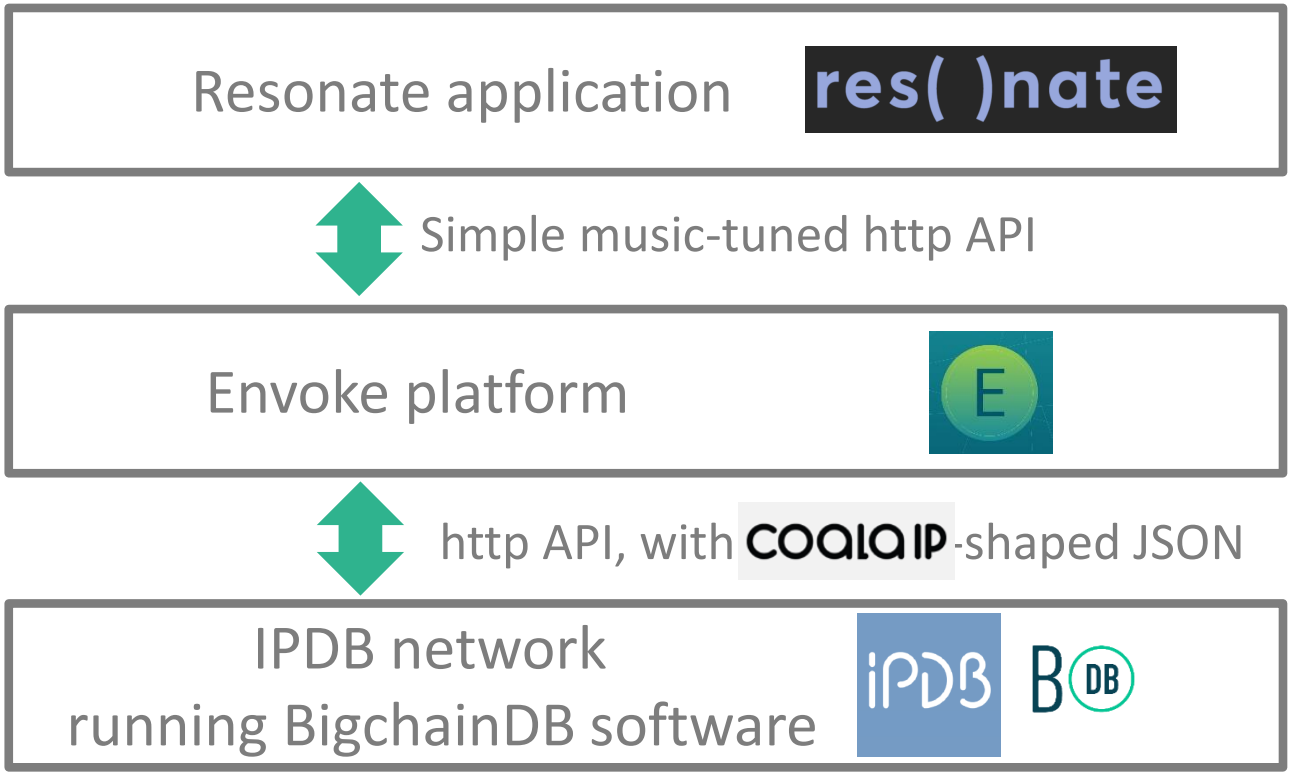
5.3. Read the Creation Transaction from the DB

```
# Retrieve a transaction from the bigchain
tx_retrieved = b.get_transaction(tx_signed['id'])
tx_retrieved
```

```
{
  // Content Addressable identifier
  "id": "811f13e...ec6f46729",

  // One of "CREATE" or "TRANSFER"
  "operation": "CREATE",

  // Description of asset being created
  "asset": {
    "data": {
      "definition": "Asset definition"
    }
  },

  // Each input contains a fulfillment to a previous output
  "inputs": [
    {
      "fulfillment": "cf:4:__Y_Um6H7...",
      "fulfills": null,
      "owners_before": [
        "JEAkEJqLbbgDRAtMm8YAjGp759Aq2qTn9eaEHUj2XePE"
      ]
    }
  ],

  // Each output defines an amount of an asset, and cryptographic
  // conditions to be able to transfer it
  "outputs": [
```

Did you notice…
this was all it took to use blockchain technology!

Example Music Stack: Resonate + Envoke



Partly decentralized stack

…

APPLICATION

PROCESSING
e.g. EC2

FILE SYSTEM
e.g. S3, HDFS

DATABASE
e.g. MySQL, MongoDB,
BigchainDB/IPDB

Fully decentralized stack

…

APPLICATION

PROCESSING
e.g. EC2, **Ethereum, Sawtooth Lake**

FILE SYSTEM
e.g. S3, HDFS, **IPFS**

DATABASE
e.g. MySQL, MongoDB
BigchainDB/IPDB

E-GOLD / E-CASH
Bitcoin, zcash, Eth

Towards Shipping…

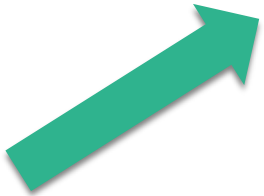
Q: How to efficiently make a target use case demo?

A: Hackfests

"POC"
- Targeted scope
- Targeted team
- Long time: 1-3 mos

"Hackfest"
- Short time: 1-3 d
- Targeted scope
- Targeted team (2-4 co's)

"Hackathon"
- Short time: 1-3 d
- Open-ended team
- Undirected scope
- High org. effort



Example output
- Biz case: 10 slides
- Frontend: mockup or code
- Backend v0.1: code w. API
- Medium post

BigchainDB + IPDB + COALA IP

are tools

to help you rewire the music industry

for the benefit of everyone

Let's hack! Let's ship!

coalaip.org

ipdb.foundation

bigchaindb.com

Trent McConaghy
trent@bigchaindb.com