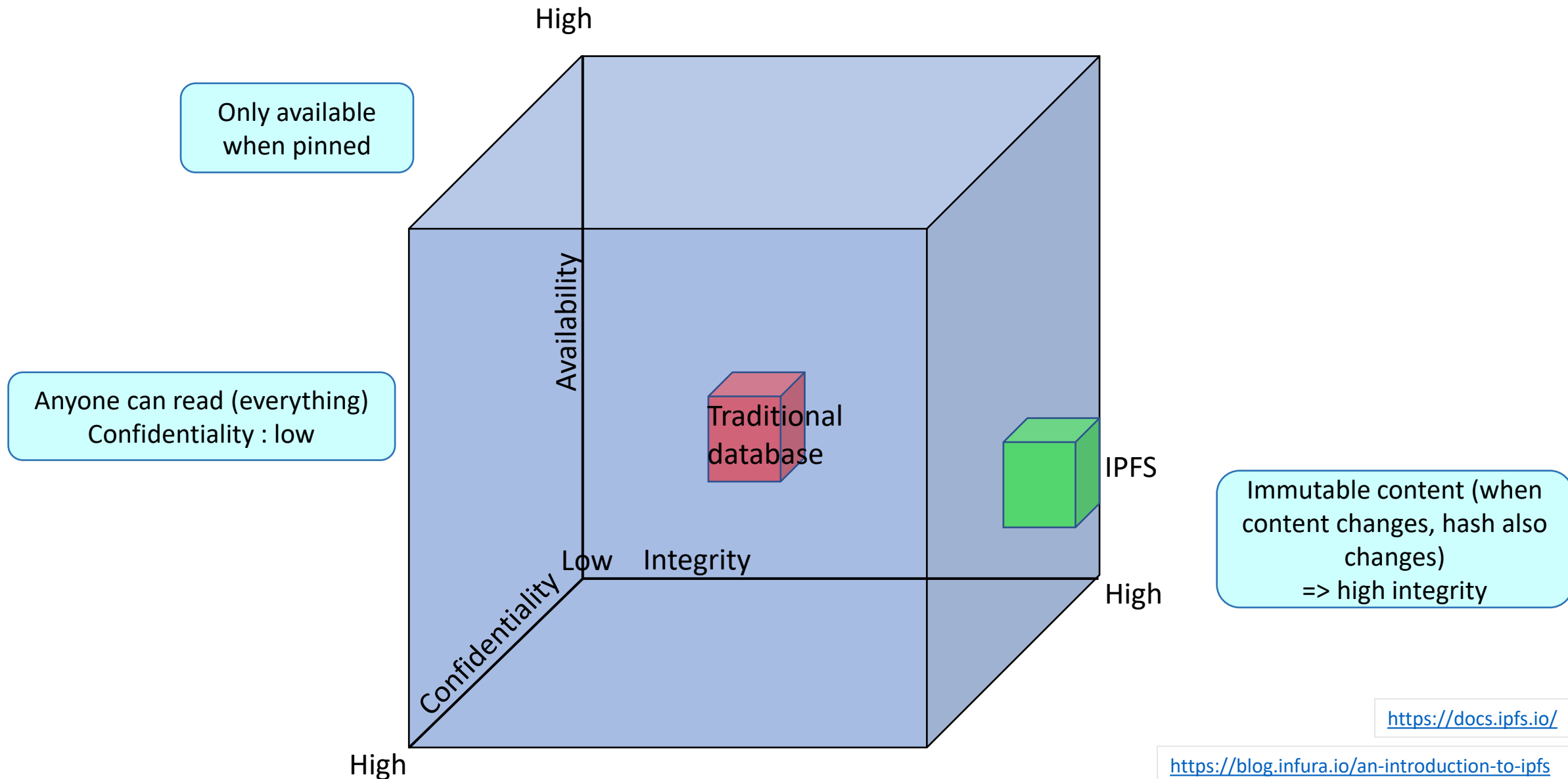


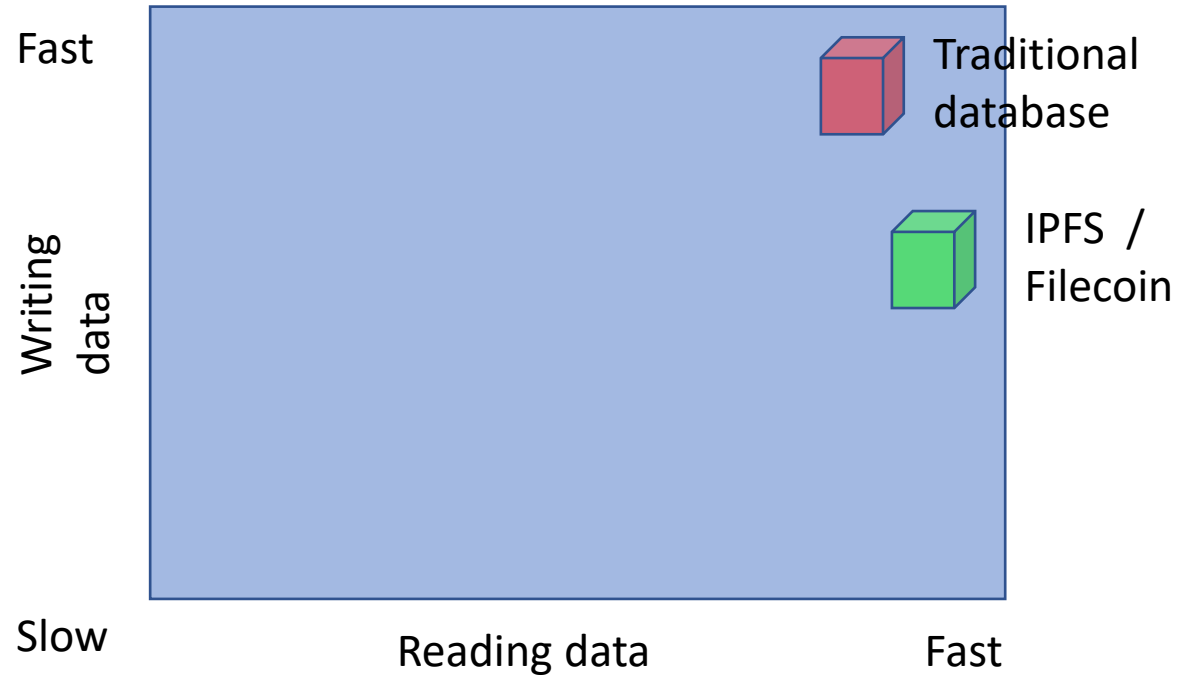
PD-11 IPFS



PD-11.1 IPFS Intro: Characteristics IPFS



PD-11.1 IPFS Intro: Performance

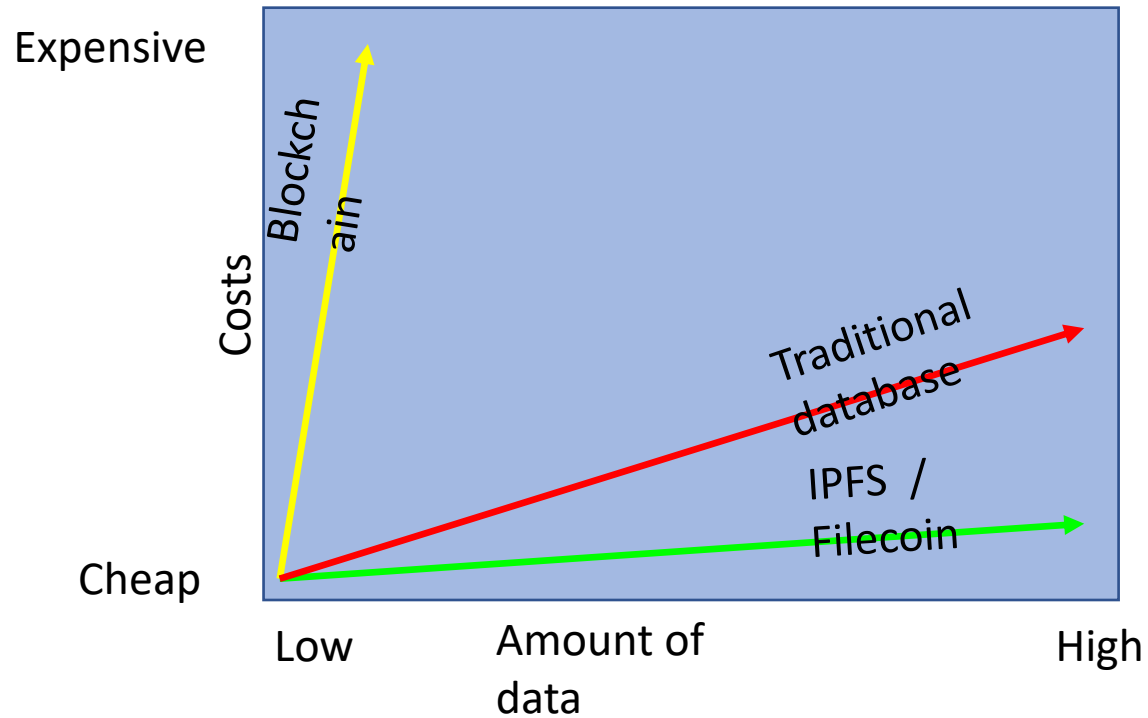


Caching possible (isn't changing)
Fetch from everyone

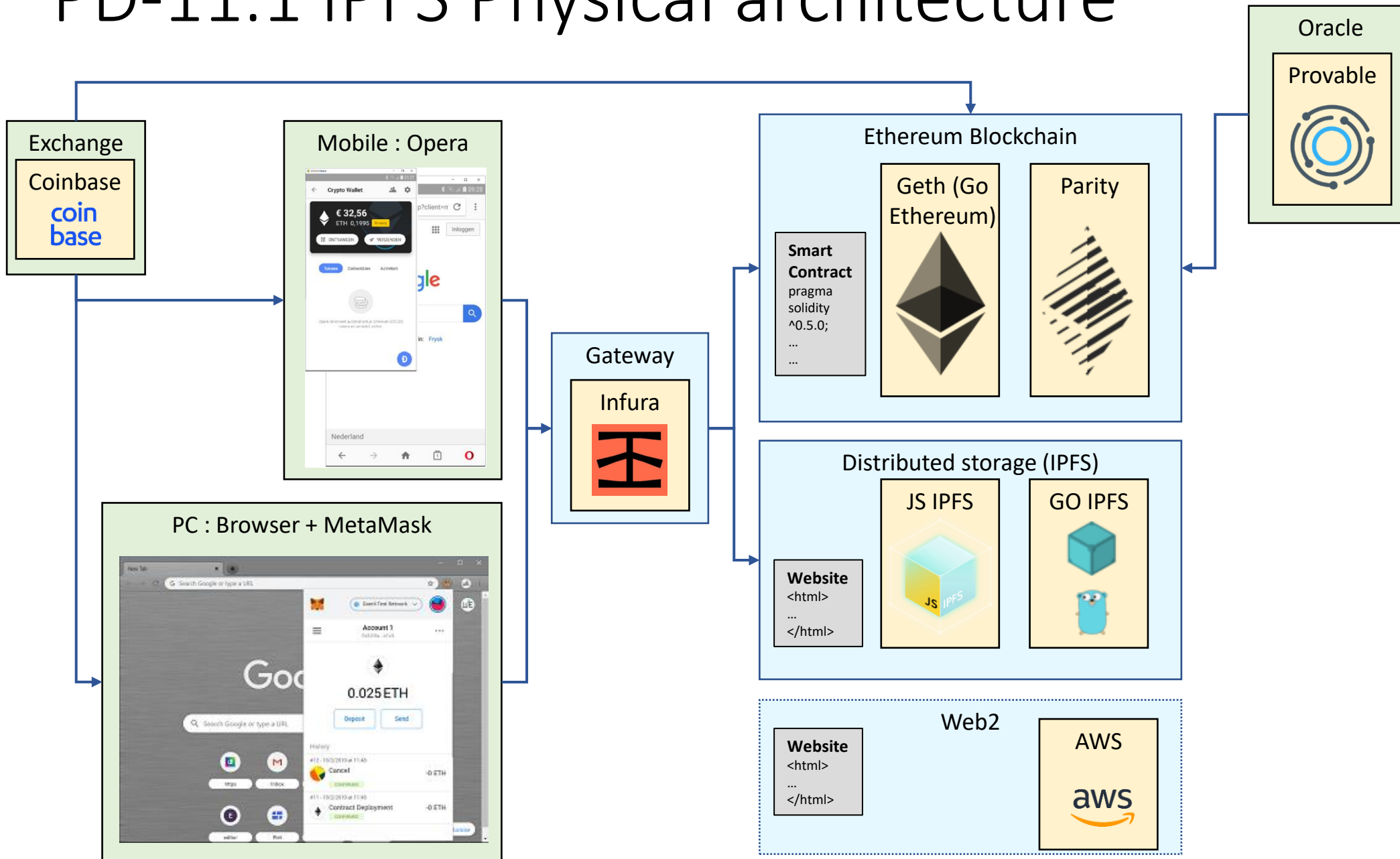
De-duplicate possible

Don't have to download first to verify hash

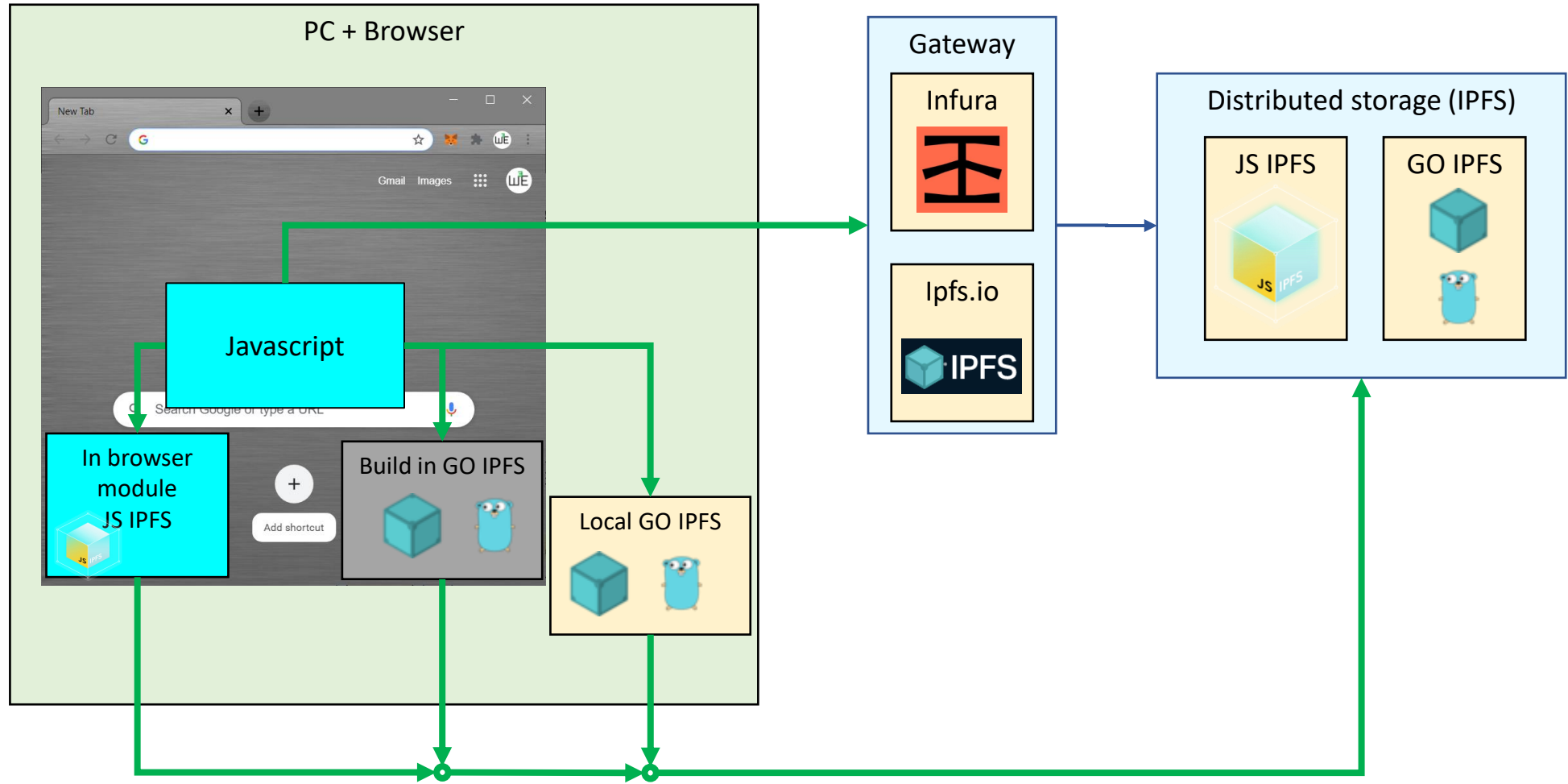
PD-11.1 IPFS Intro: Amounts of data vs Costs



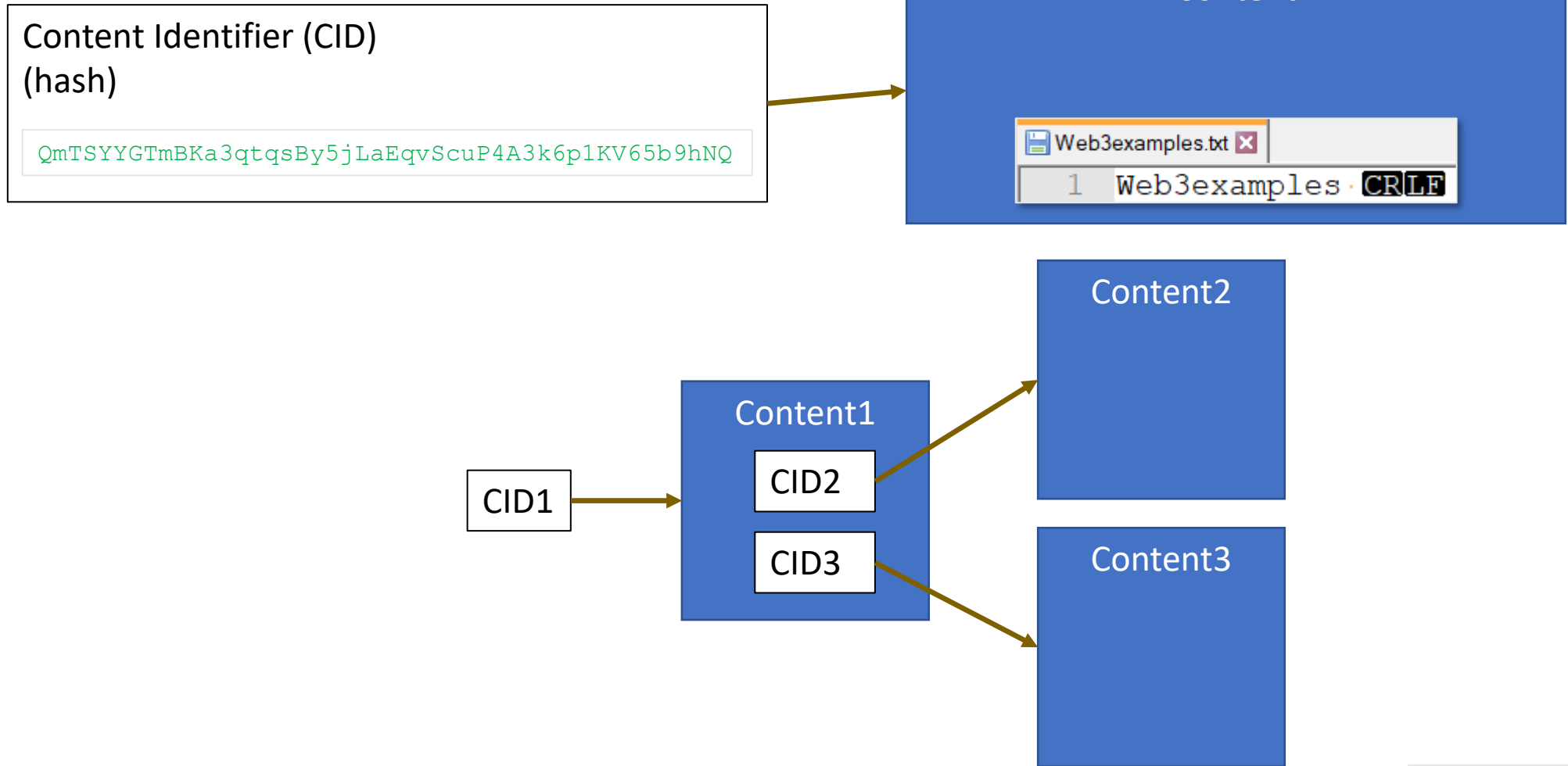
PD-11.1 IPFS Physical architecture



PD-11.1 IPFS Options



PD-11.2 Content Identifiers (CIDs)

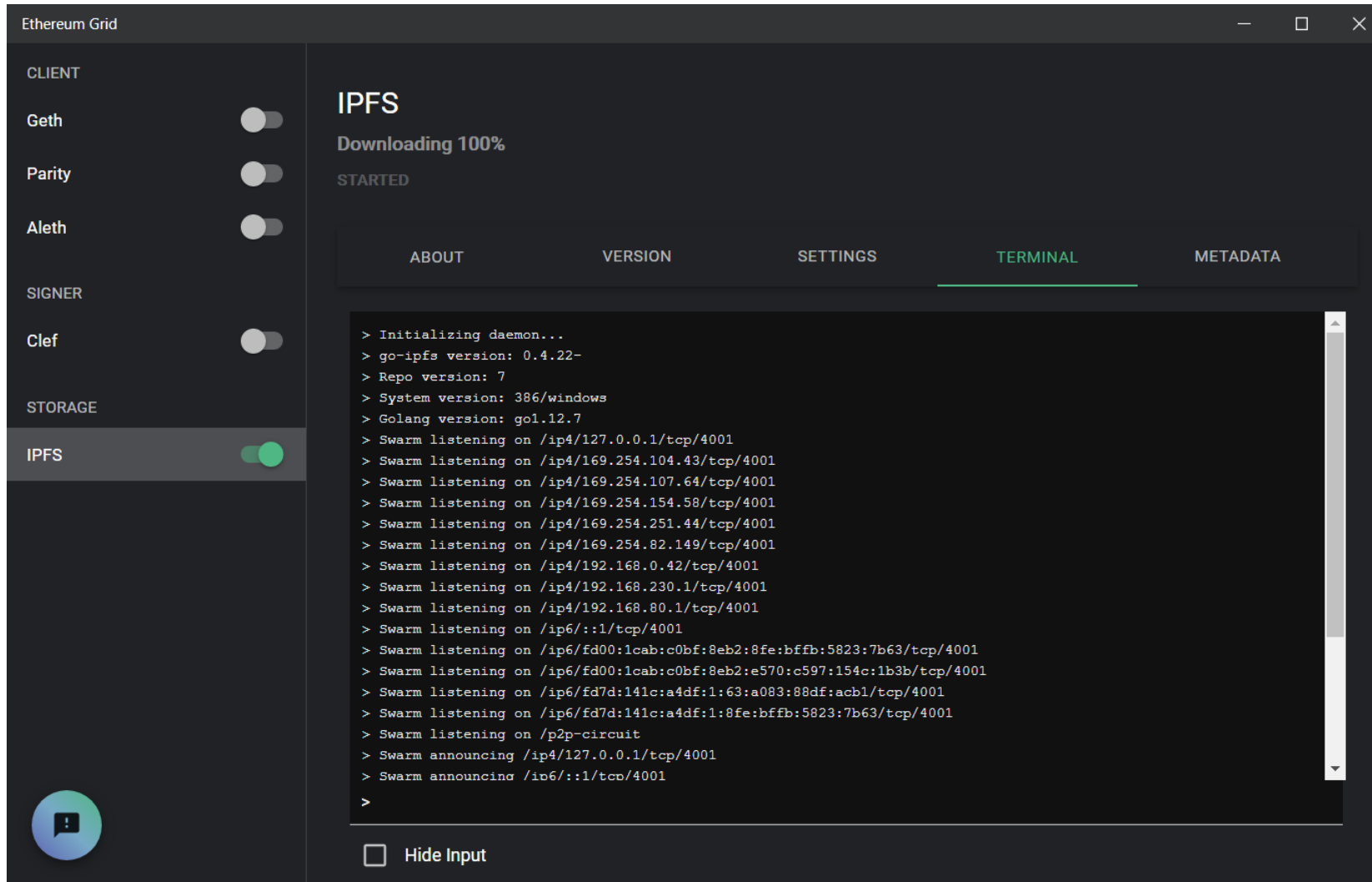


<https://docs.ipfs.io/>

<https://blog.infura.io/an-introduction-to-ipfs>

<https://ipfs.io/ipfs/QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ>

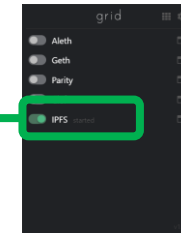
PD-11.3 IPFS Commmmands - Grid – GO-IPFS



The screenshot shows the Ethereum Grid application interface. On the left, there is a sidebar with sections for CLIENT, SIGNER, and STORAGE. Under CLIENT, Geth, Parity, and Aleth are listed with toggle switches. Under SIGNER, Clef is listed with a toggle switch. Under STORAGE, IPFS is listed with a toggle switch that is currently turned on. The main area is titled 'IPFS' and shows 'Downloading 100%' and 'STARTED'. Below this, there are tabs for ABOUT, VERSION, SETTINGS, TERMINAL, and METADATA. The TERMINAL tab is active, displaying the following output:

```
> Initializing daemon...
> go-ipfs version: 0.4.22-
> Repo version: 7
> System version: 386/windows
> Golang version: go1.12.7
> Swarm listening on /ip4/127.0.0.1/tcp/4001
> Swarm listening on /ip4/169.254.104.43/tcp/4001
> Swarm listening on /ip4/169.254.107.64/tcp/4001
> Swarm listening on /ip4/169.254.154.58/tcp/4001
> Swarm listening on /ip4/169.254.251.44/tcp/4001
> Swarm listening on /ip4/169.254.82.149/tcp/4001
> Swarm listening on /ip4/192.168.0.42/tcp/4001
> Swarm listening on /ip4/192.168.230.1/tcp/4001
> Swarm listening on /ip4/192.168.80.1/tcp/4001
> Swarm listening on /ip6:::1/tcp/4001
> Swarm listening on /ip6/fd00:1cab:c0bf:8eb2:8fe:bffb:5823:7b63/tcp/4001
> Swarm listening on /ip6/fd00:1cab:c0bf:8eb2:e570:c597:154c:1b3b/tcp/4001
> Swarm listening on /ip6/fd7d:141c:a4df:1:63:a083:88df:acb1/tcp/4001
> Swarm listening on /ip6/fd7d:141c:a4df:1:8fe:bffb:5823:7b63/tcp/4001
> Swarm listening on /p2p-circuit
> Swarm announcing /ip4/127.0.0.1/tcp/4001
> Swarm announcing /ip6:::1/tcp/4001
>
```

At the bottom of the terminal, there is a checkbox labeled 'Hide Input' which is currently unchecked.



IPFS.exe Location on windows:
%appdata%\grid\app_cache\bin\bin_ipfs

<https://grid.ethereum.org>

<https://dist.ipfs.io/#go-ipfs>

PD-11.3 RPC API via port 5001 - version

```
>curl -X POST "http://127.0.0.1:5001/api/v0/version"
```

```
{"Version":"0.7.0","Commit":"","Repo":"10","System":"386/windows","Golang":"go1.14.4"}
```

```
>curl -X POST "http://127.0.0.1:5001/api/v0/stats/repo"
```

```
{"RepoSize":52123001,"StorageMax":10000000000,"NumObjects":2691,"RepoPath":"C:\\...\\.ipfs","Version":"fs-repo@10"}
```

```
> curl -X POST "http://127.0.0.1:5001/api/v0/add?pin=true" -H "Content-Type: multipart/form-data" -F  
file=@Web3examples.txt
```

```
{"Name":"Web3examples.txt","Hash":"QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ","Size":"23"}
```

PD-11.3 Gateway

```
>curl http://127.0.0.1:8080/ipfs/QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ  
Web3examples
```

<http://127.0.0.1:8080/ipfs/QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ>

PD-11.3 IPFS add & cat

```
> echo Web3examples > Web3examples.txt
```

```
> ipfs add Web3examples.txt
```

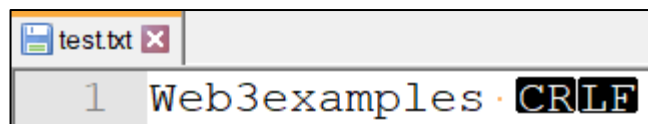
```
15 B / 15 B [=====] 100.00% added
```

```
QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ Web3examples.txt
```

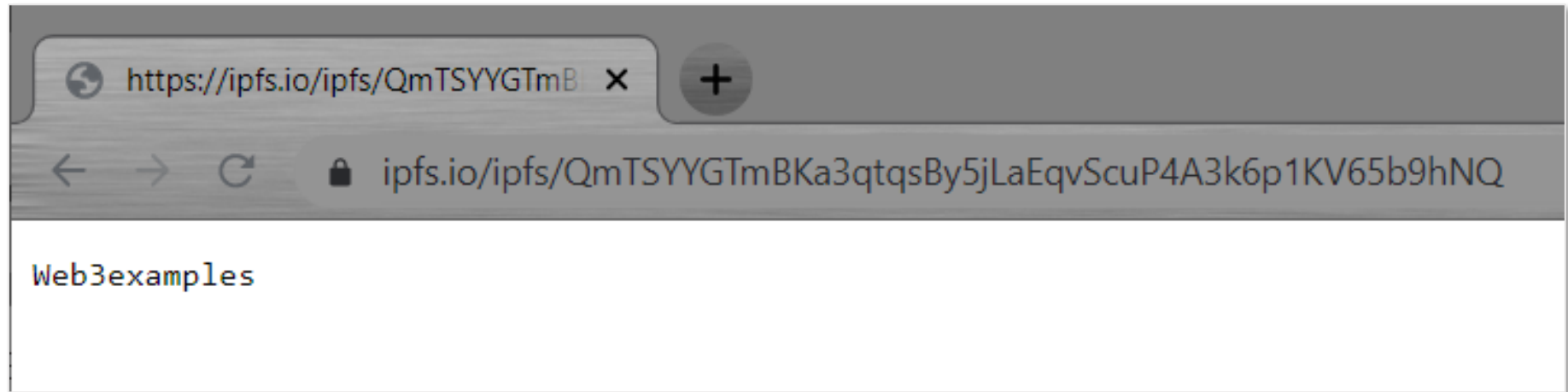
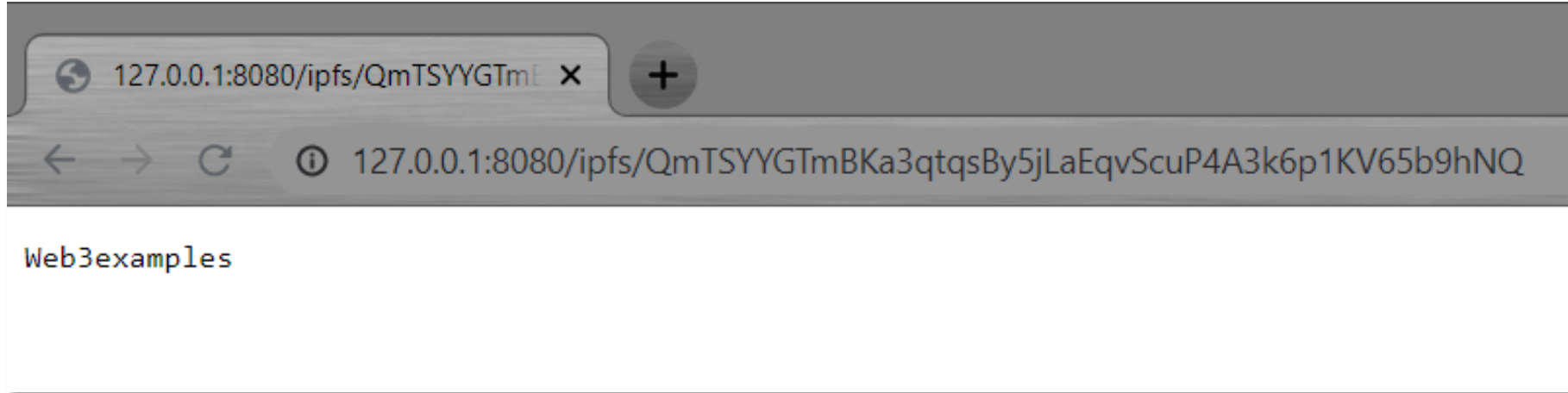
```
15 B / 15 B [=====] 100.00%
```

```
> ipfs cat QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ
```

```
Web3examples
```



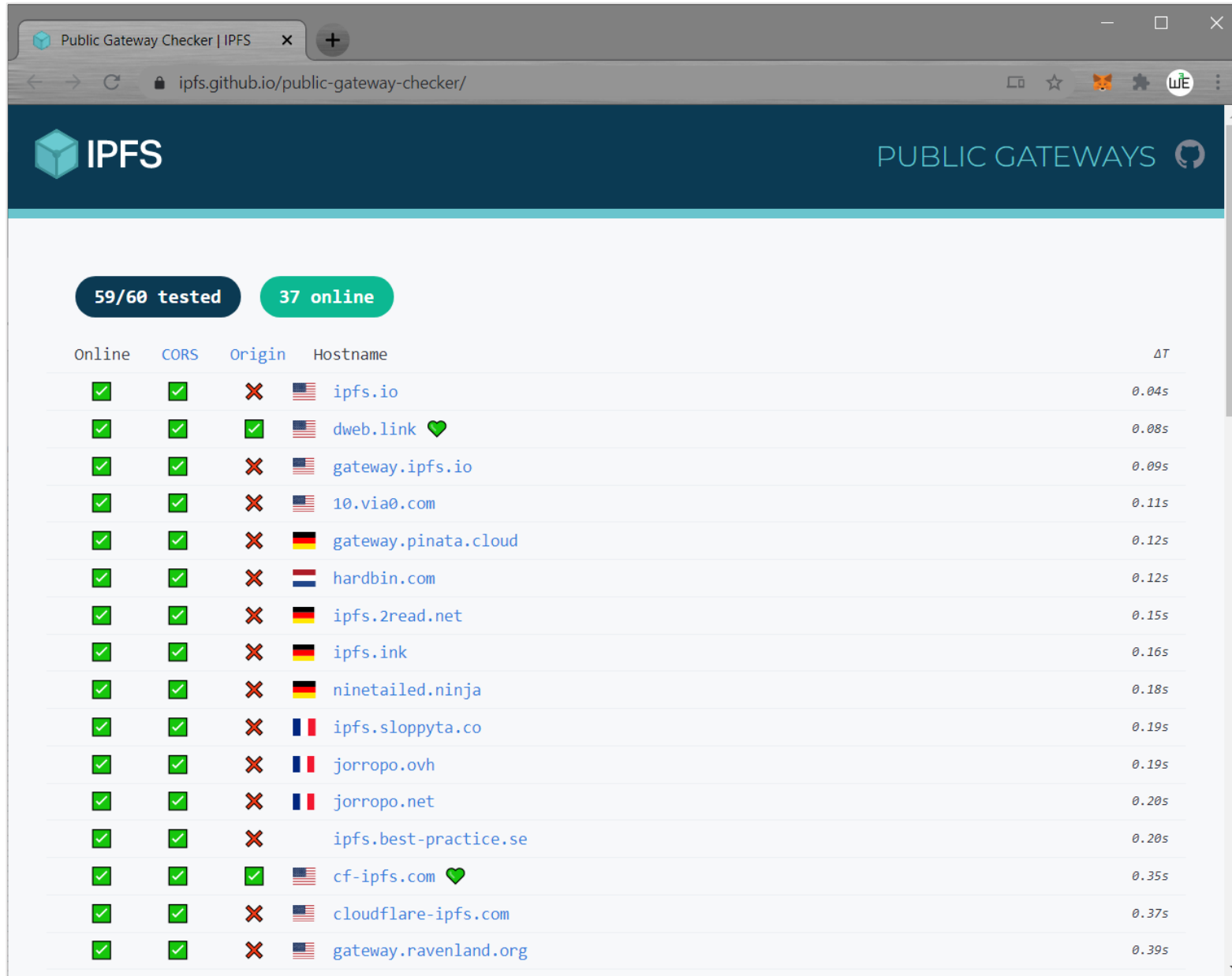
PD-11.4 Gateways - access via gateway




















<http://127.0.0.1:8080/ipfs/QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ>

<https://ipfs.io/ipfs/QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ>

PD-11.4 List of IPFS Gateways



The screenshot shows a web browser window with the URL `ipfs.github.io/public-gateway-checker/`. The page features the IPFS logo and the title "PUBLIC GATEWAYS". Below the header, there are two status indicators: "59/60 tested" and "37 online". The main content is a table listing various IPFS gateways. Each row includes a green checkmark for "Online" status, a green checkmark for "CORS", a red 'X' for "Origin", a hostname with a country flag, and a response time in seconds under the column "ΔT".

Online	CORS	Origin	Hostname	ΔT
✓	✓	✗	 ipfs.io	0.04s
✓	✓	✓	 dweb.link 	0.08s
✓	✓	✗	 gateway.ipfs.io	0.09s
✓	✓	✗	 10.via0.com	0.11s
✓	✓	✗	 gateway.pinata.cloud	0.12s
✓	✓	✗	 hardbin.com	0.12s
✓	✓	✗	 ipfs.2read.net	0.15s
✓	✓	✗	 ipfs.ink	0.16s
✓	✓	✗	 ninetailed.ninja	0.18s
✓	✓	✗	 ipfs.sloppyta.co	0.19s
✓	✓	✗	 jorropo.ovh	0.19s
✓	✓	✗	 jorropo.net	0.20s
✓	✓	✗	ipfs.best-practice.se	0.20s
✓	✓	✓	 cf-ipfs.com 	0.35s
✓	✓	✗	 cloudflare-ipfs.com	0.37s
✓	✓	✗	 gateway.ravenland.org	0.39s

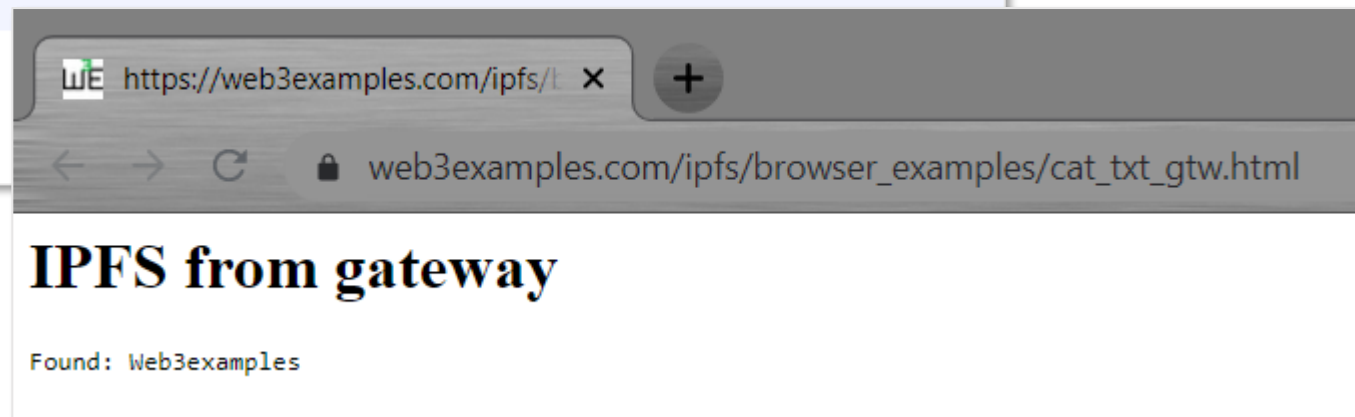
<https://ipfs.github.io/public-gateway-checker/>

PD-11.4 Access via gateway in website

```
cat_txt_gtw.html x
1 <html>
2   <body>
3     <h1>IPFS from gateway</h1>
4     <pre id="log" style="width:100%;height:200px"></pre>
5     <script type="text/javascript">
6       function log(logstr) {
7         document.getElementById("log").innerHTML += logstr + "\n";
8       }
9       async function f() {
10        const hash = "QmTSYYGTmBKa3qtasBy5iLaEqvScuP4A3k6p1KV65b9hNQ";
11        var response = await fetch("https://ipfs.io/ipfs/" + hash);
12        var str = await response.text();
13        log(`Found: ${str}`);
14      }
15      f();
16    </script>
17  </body>
18 </html>
```

https://github.com/web3examples/ipfs/tree/master/browser_examples/cat_txt_gtw.html

https://web3examples.com/ipfs/browser_examples/cat_txt_gtw.html



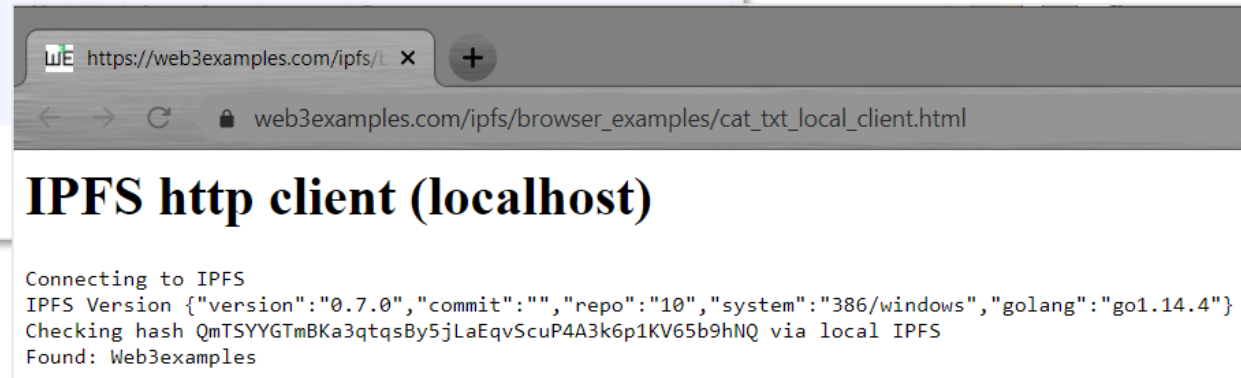
PD-11.5 Javascript library: ipfs-http-client local

Local running IPFS
(via Grid)

```
cat_bt_local_client.html x
1 <html>
2   <head>
3     <script src="https://unpkg.com/ipfs-http-client/dist/index.min.js"></script>
4   </head>
5   <body>
6     <h1>IPFS http client (localhost)</h1>
7     <pre id="log" style="width:100%;height:200px"></pre>
8     <script type="text/javascript">
9       function log(logstr) {
10        document.getElementById("log").innerHTML += logstr + "\n";
11      }
12      async function f () {
13        const hash = "QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ";
14        log(`Connecting to IPFS`);
15        const ipfs = window.IpfsHttpClient('http://localhost:5001');
16        const version = await ipfs.version().catch(x => log(`Error: ${x}`));
17        log(`IPFS Version ${JSON.stringify(version)}`);
18        log(`Checking hash ${hash} via local IPFS`);
19        var str = ""
20        for await (const result of ipfs.cat(hash)) {
21          str += String.fromCharCode.apply(null, result); // convert uint8array to string
22        }
23        log(`Found: ${str}`);
24      }
25      f();
26    </script>
27  </body>
28 </html>
```

https://github.com/web3examples/ipfs/tree/master/browser_examples/cat_txt_local_client.html

https://web3examples.com/ipfs/browser_examples/cat_txt_local_client.html



<https://github.com/ipfs/js-ipfs/tree/master/packages/ipfs-http-client>

PD-11.5 Javascript library: ipfs-http-client Infura

```
cat_txt_infura_client.html
1 <html>
2 .....<head>.....
3 .....<script src="https://unpkg.com/ipfs-http-client/dist/index.min.js"></script>...
4 </head>
5 .....<body>
6 .....<h1>IPFS http client (infura)</h1>
7 .....<pre id="log" style="width:100%;height:200px"></pre>
8 .....<script type="text/javascript">
9 .....function log(logstr) { .....
10 .....document.getElementById("log").innerHTML +=logstr+"\n";
11 .....}
12 .....async function f() {
13 .....const hash="QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ";
14 .....log(`Connecting to IPFS`);
15 .....const ipfs = window.IpfsHttpClient('https://ipfs.infura.io:5001');
16 .....const version = await ipfs.version().catch(x=>log(`Error: ${x}`))
17 .....log(`IPFS Version ${JSON.stringify(version)}`)
18 .....log(`Checking hash ${hash} via IPFS on Infura`)
19 .....var str=""
20 .....for await (const result of ipfs.cat(hash)) {
21 .....str += String.fromCharCode.apply(null, result); //convert uint8array to string
22 .....}
23 .....log(`Found: ${str}`);
24 .....}
25 .....f();
26 .....</script>
27 .....</body>
28 </html>
```

https://github.com/web3examples/ipfs/tree/master/browser_examples/cat_txt_infura_client.html

https://web3examples.com/ipfs/browser_examples/cat_txt_infura_client.html



<https://github.com/ipfs/js-ipfs/tree/master/packages/ipfs-http-client>

PD-11.6 IPFS running in browser

```
cat_txt_ipfs_lib.html
1 <html>
2 ..... <head> .....
3 ..... <script src="https://unpkg.com ipfs/dist/index.min.js"></script>
4 ..... </head>
5 ..... <body>
6 ..... <h1>IPFS in browser</h1>
7 ..... <pre id="log" style="width:100%;height:200px"></pre>
8 ..... <script type="text/javascript">
9 ..... function log(logstr) {
10 ..... document.getElementById("log").innerHTML +=logstr+"\n";
11 ..... }
12 ..... async function f () {
13 ..... const hash="QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ";
14 ..... log(`Connecting to IPFS`);
15 ..... const ipfs = await window.Ipfs.create();
16 ..... log(`Status = ${ipfs.isOnline() ? 'online' : 'offline'}`);
17 ..... const version = await ipfs.version().catch(x=>log(`Error: ${x}`));
18 ..... log(`IPFS Version: ${JSON.stringify(version)}`);
19 ..... log(`Checking hash ${hash} via IPFS in browser`);
20 ..... var str=""
21 ..... for await (const result of ipfs.cat(hash)) {
22 ..... str += String.fromCharCode.apply(null, result); // convert uint8array to string
23 ..... }
24 ..... log(`Found: ${str}`);
25 ..... }
26 ..... f();
27 ..... </script>
28 ..... </body>
29 </html>
```

https://web3examples.com/ipfs/browser_examples/cat_txt_ipfs_lib.html

https://github.com/web3examples/ipfs/tree/master/browser_examples/cat_txt_ipfs_lib.html



<https://github.com/ipfs/js-ipfs>

PD-11.6 IPFS running in browser

IPFS in browser

Connecting to IPFS
Status = online
IPFS Version {"versi
Checking hash QmTSYY
Found: Web3examples

Application

- Manifest
- Service Workers
- Storage

Storage

- Local Storage
- Session Storage
- IndexedDB
 - ipfs - https://web3examples.com
 - ipfs
 - ipfs/blocks - https://web3examples.com
 - ipfs/blocks
 - ipfs/datastore - https://web3examples.com
 - ipfs/datastore
 - ipfs/keys - https://web3examples.com
 - ipfs/keys
 - ipfs/pins - https://web3examples.com
 - Web SQL
 - Cookies

ipfs

Security origin https://web3examples.com

Version 2

Object stores 1

Delete database Refresh database

IPFS Storage within browser
Can be deleted and will
rebuild when page is
refreshed

PD-11.7 RPC Add & pin via Infura

```
>curl "https://ipfs.infura.io:5001/api/v0/add?pin=true" -X POST -H "Content-Type: multipart/form-data" -F file=@Web3examples.txt
```

```
{"Name": "Web3examples.txt", "Hash": "QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ", "Size": "23"}
```

https://github.com/web3examples/ipfs/blob/master/rpc/curl_add_infura.cmd

<https://infura.io/docs/ipfs/post/add>

PD-11.7 Add & Pin via website on Infura

The image shows a code editor on the left and a browser window on the right. The code editor displays the HTML and JavaScript code for a web page that uses the IPFS HTTP client to add a file to the IPFS network via Infura. The browser window shows the rendered page with the title "IPFS add (Infura)" and the output of the add command.

```
1 <html>
2   <head>
3     <script src="https://unpkg.com/ipfs-http-client/dist/index.min.js"></script>
4   </head>
5   <body>
6     <h1>IPFS add (Infura)</h1>
7     <pre id="log" style="width:100%;height:200px"></pre>
8     <script type="text/javascript">
9       function log(logstr) {
10        document.getElementById("log").innerHTML += logstr + "\n";
11      }
12      async function f () {
13        const str = "Web3examples\n"; // note: trailing space, CR, LF
14        const ipfs = window.IpfsHttpClient('https://ipfs.infura.io:5001');
15        log(`Adding ${str} to ipfs via Infura`);
16        const result = await ipfs.add(str).catch(log);
17        log(`cid=${result.path}`);
18      }
19      f();
20    </script>
21  </body>
22 </html>
```

Browser window: https://web3examples.com/ipfs/browser_examples/add_txt_infura.html

Browser window: https://github.com/web3examples/ipfs/tree/master/browser_examples/add_txt_infura.html

Browser window: <https://web3examples.com/ipfs/>

Browser window: web3examples.com/ipfs/browser_examples/add_txt_infura.html

Browser window: **IPFS add (Infura)**

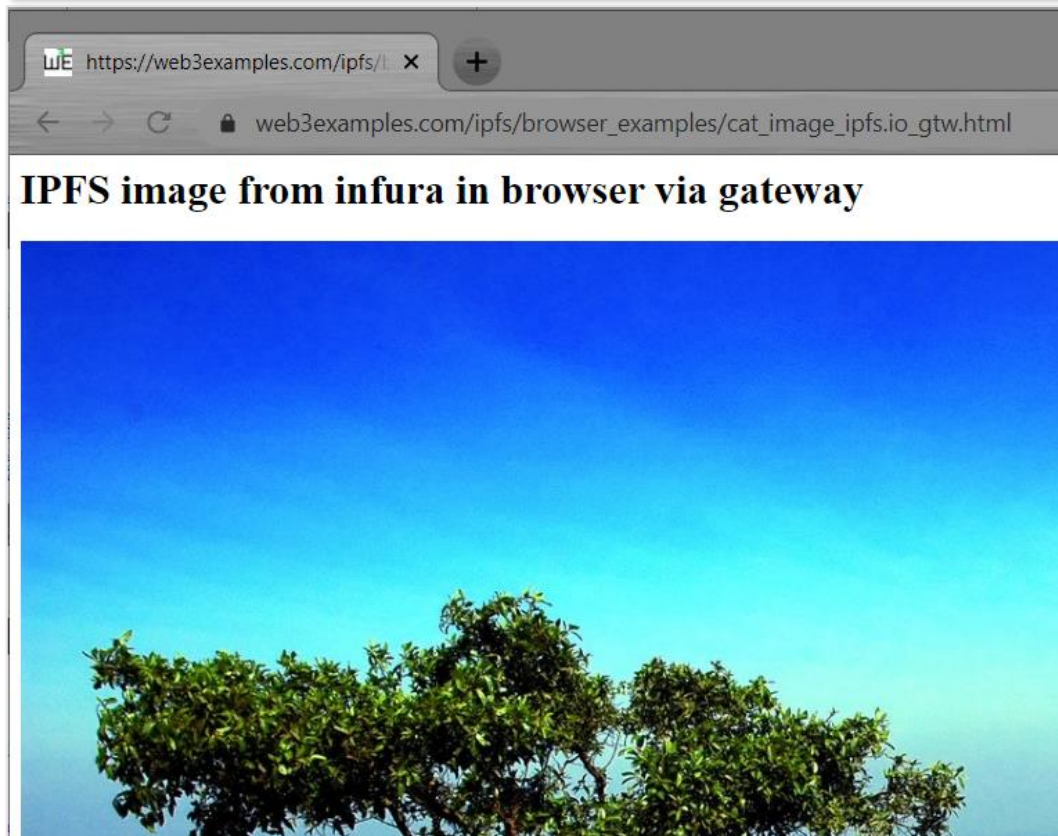
Browser window: Adding Web3examples to ipfs via Infura
cid=QmTSYYGTmBKa3qtqsBy5jLaEqVScuP4A3k6p1KV65b9hNq

Note: free, but rate limited

<https://github.com/ipfs/js-ipfs/tree/master/packages/ipfs-http-client>

PD-11.8 Get image via Gateway

```
cat_image_ipfs.io_gtw.html x
1 <html>
2   <head>
3     <script src="https://unpkg.com/ipfs-http-client/dist/index.min.js"></script>
4   </head>
5   <body>
6     <h1>IPFS image from infura in browser via gateway</h1>
7     
8   </body>
9 </html>
```



<https://ipfs.io/ipfs/QmcT9k932Hq4WN7K2TBAmHRhnLDVVfh7uCxQHZWJ3dZaDt>

https://web3examples.com/ipfs/browser_examples/cat_image_ipfs.io_gtw.html

https://github.com/web3examples/ipfs/tree/master/browser_examples/cat_image_ipfs.io_gtw.html

PD-11.8 Get image via infura & javascript library

```
1 <html>
2   <head>.....
3   <script src="https://unpkg.com/ipfs-http-client/dist/index.min.js"></script>
4 </head>
5   <body>
6     <h1>IPFS image from infura in browser via library</h1>.....
7     <pre id="log" style="width:100%;height:100px"></pre>
8     <img id="myimage"></img>
9
10    <script type="text/javascript">
11      function log(logstr) {
12        document.getElementById("log").innerHTML +=logstr+"\n";
13      }
14      async function f () {
15        var ui8arr=[]
16        const hash="QmcT9k932Hq4WN7K2TBAmHRhnLDVVfh7uCxQHZWJ3dZaDt";
17        log(`Connecting to IPFS`);
18        const ipfs = window.IpfsHttpClient('https://ipfs.infura.io:5001');
19        const version = await ipfs.version().catch(x=>log(`Error: ${x}`))
20        log(`IPFS Version ${JSON.stringify(version)}`)
21        log(`Checking hash ${hash} via IPFS on Infura`)
22        for await (const result of ipfs.cat(hash)) {
23          ui8arr.push(result) //get all parts of the image
24          var blob = new Blob(ui8arr, {type: "image/jpeg"}); //convert to blob
25          var url=URL.createObjectURL(blob) //make usable in img
26          document.getElementById("myimage").src=url;
27        }
28      f();
29    </script>
30  </body>
31 </html>
```

https://web3examples.com/ipfs/...
web3examples.com/ipfs/browser_examples/cat_image_infura_client.html

IPFS image from infura in browser via library

Connecting to IPFS
IPFS Version {"version":"0.7.0","commit":"db62f0e34-dirty","repo":"10","system":"amd64/linux","goLang":"go1.14.9"}
Checking hash QmcT9k932Hq4WN7K2TBAmHRhnLDVVfh7uCxQHZWJ3dZaDt via IPFS on Infura



https://web3examples.com/ipfs/browser_examples/cat_image_infura_client.html

https://github.com/web3examples/ipfs/tree/master/browser_examples/cat_image_infura_client.html

PD-11.9 Host website on IPFS

```
>ipfs add -r mini/
```

```
806 B / ? added Qmd582iEf1NbGstZ5adY41na6KYPztGd8pZ4EfCzdcLQCj mini/images/logo.png
1007 B / ? added QmX7PzcFKq8Gd5jSbdvs2THeWR3p91dBPfDZUxAf5hKPFs mini/index.html
1007 B / ? added Qmb86U2bQXy7ytfnVZ1XoC8Z6LRp6XGi1AWYM6JCjztCyd mini/images
1007 B / 1007 B [=] 100.00%
      added QmYwjdfP3AMHSxqT6RT3QWxy7Jw1j2mKxFfCweQAfMJ1wS mini
1007 B / 1007 B [==] 100.00%
```

```
>ipfs dag get QmYwjdfP3AMHSxqT6RT3QWxy7Jw1j2mKxFfCweQAfMJ1wS
```

```
{ "data": "CAE=",
  "links": [
    { "Cid": {"/": "Qmb86U2bQXy7ytfnVZ1XoC8Z6LRp6XGi1AWYM6JCjztCyd"},
      "Name": "images",
      "Size": 872
    },
    { "Cid": {"/": "QmX7PzcFKq8Gd5jSbdvs2THeWR3p91dBPfDZUxAf5hKPFs"},
      "Name": "index.html",
      "Size": 212
    }
  ]
}
```

```
> ipfs dag get Qmb86U2bQXy7ytfnVZ1XoC8Z6LRp6XGi1AWYM6JCjztCyd
```

```
{ "data": "CAE=",
  "links": [
    { "Cid": {"/": "Qmd582iEf1NbGstZ5adY41na6KYPztGd8pZ4EfCzdcLQCj"},
      "Name": "logo.png",
      "Size": 817
    }
  ]
}
```



https://web3examples.com/ipfs/site_on_ipfs/mini

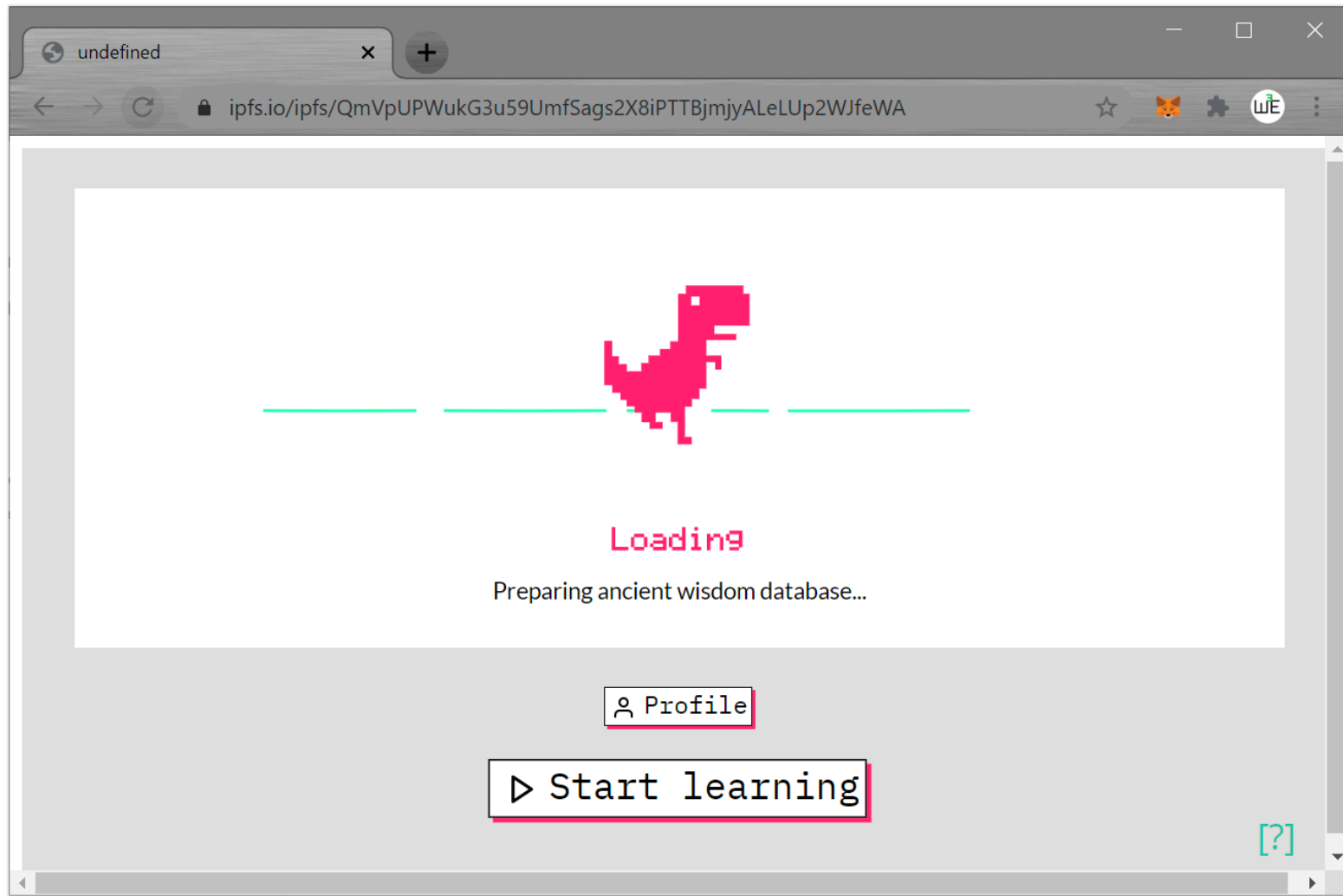
<https://github.com/ipfs-shipyard/ipfs-deploy>

<https://simpleaswater.com/hosting-website-using-ipfs/>

<http://localhost:8080/ipfs/QmYwjdfP3AMHSxqT6RT3QWxy7Jw1j2mKxFfCweQAfMJ1wS/>

<https://ipfs.io/ipfs/QmYwjdfP3AMHSxqT6RT3QWxy7Jw1j2mKxFfCweQAfMJ1wS/>

PD-11.9 Host website on IPFS: Koios site



PD-11.10 References: IPNS

```
> ipfs name publish QmYwjdfP3AMHSxqT6RT3QWxy7Jw1j2mKxFfCweQAfMJ1wS
```

```
Published to k2k4r8pkzbkqce4lsafe7qlpylhzmkzari9xs2lryb5v3dl1e16ugf4q:  
/ipfs/QmYwjdfP3AMHSxqT6RT3QWxy7Jw1j2mKxFfCweQAfMJ1wS
```



Notes:

slow ~ 1 minute for first load
Have to republish every 12 hours

<https://discuss.ipfs.io/t/ipns-how-much-is-the-lifetime/6705>

<https://ipfs.io/ipfs/QmYwjdfP3AMHSxqT6RT3QWxy7Jw1j2mKxFfCweQAfMJ1wS/>

<https://ipfs.io/ipns/k2k4r8pkzbkqce4lsafe7qlpylhzmkzari9xs2lryb5v3dl1e16ugf4q>

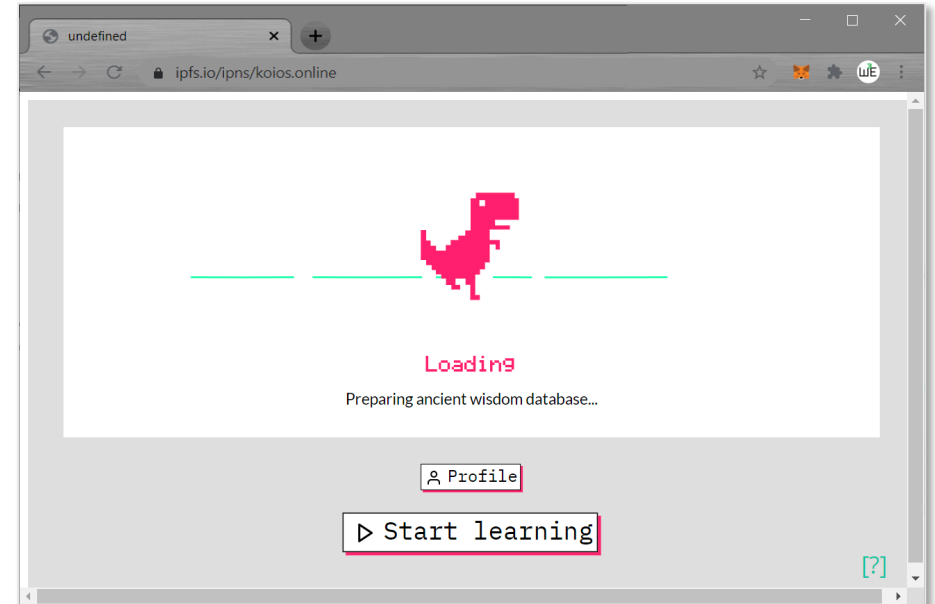
PD-11.10 References: IPNS + _dnslink

```
> dig txt _dnslink.koios.online
```

```
; <<>> DiG 9.9.5-W1 <<>> txt _dnslink.koios.online  
;; global options: +cmd  
;; Got answer:  
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 44983  
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0,  
ADDITIONAL: 1  
  
;; OPT PSEUDOSECTION:  
; EDNS: version: 0, flags:; udp: 512  
;; QUESTION SECTION:  
;_dnslink.koios.online.      IN      TXT  
  
;; ANSWER SECTION:  
_dnslink.koios.online.  866     IN      TXT  
"dnslink=/ipfs/QmVpUPWukG3u59UmfSags2X8iPTTBjmjyALeLUp2WJfeWA"  
  
;; Query time: 28 msec  
;; SERVER: 192.168.1.1#53(192.168.1.1)  
;; WHEN: Mon Dec 14 15:15:53 W. Europe Standard Time 2020  
;; MSG SIZE rcvd: 123
```

Your custom DNS records for koios.online:

RECORD (KOIOS.ONLINE ZONE)	TYPE	VALUE	ACTIONS
_dnslink	TXT	dnslink=/ipfs/QmVpUPWukG3u59UmfSags2X8iPTTBjmjyALeLUp2WJfeWA	Edit Delete



<https://github.com/rgl/dig-setup>

<https://ipfs.io/ipns/koios.online>

<https://ipfs.io/ipfs/QmVpUPWukG3u59UmfSags2X8iPTTBjmjyALeLUp2WJfeWA>

PD-11.11 Paid services: Pinning services



VS



eternum

<https://ipfs.github.io/public-gateway-checker/>

<https://pinata.cloud>

<https://temporal.cloud>

<https://www.eternum.io>

<https://docs.ipfs.io/concepts/persistence>

<https://medium.com/temporal-cloud/comparing-ipfs-pinning-services-pricing-functionality-temporal-eternum-pinata-d38b87a279d8>

PD-11.11 Paid services: Incentivized storage



<https://docs.filecoin.io/build/filecoin-pinning-services>

<https://filecoin.io>

<https://www.datprotocol.com>

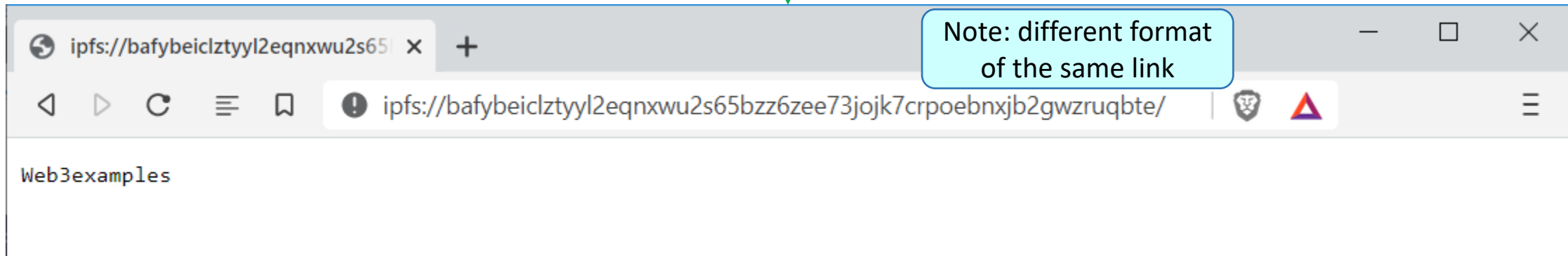
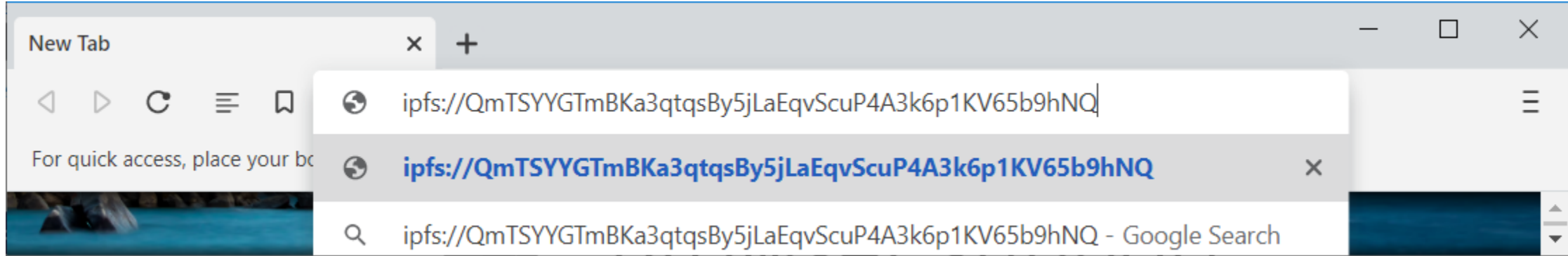
<https://liquidapps.io/>

<https://storj.io>

<https://sia.tech>

<https://swarm.ethereum.org>

PD-11.12 Browser support



<ipfs://QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ>

https://www.reddit.com/r/BATProject/comments/k5izey/ipfs_support_lands_in_brave_browser_on_the/

PD-11.13 Encryption for IPFS

Currently in rapid development

<https://peergos.org/>

<https://pypi.org/project/nucypher-ipfs/>

<https://docs.textile.io/buckets/#encryption>

<https://www.memoryandthought.me/golang,/ipfs/2020/09/04/dag-jose-project.html>

<https://github.com/ceramicnetwork/js-dag-jose>

PD-11.14 Access IPFS via Oracle

```
provable_ipfs.sol x
1  pragma solidity >= 0.5.0 < 0.6.0;
2  import "github.com/provable-things/ethereum-api/provableAPI.sol";
3  contract RandomExample is usingProvable {
4      string public result;
5      bytes32 public queryId;
6
7      function __callback(bytes32 _queryId,
8          string memory _result, bytes memory _proof ) public {
9          require(msg.sender == provable_cbAddress());
10         result = _result;
11     }
12     function GetIPFS() public {
13         queryId=provable_query("IPFS",
14             "QmTSYYGTmBKa3qtqsBy5jLaEqvScuP4A3k6p1KV65b9hNQ");
15     }
16 }
```

result

0: string: Web3examples