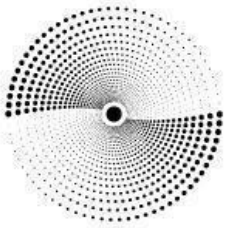
IPFS Design Document

Version 0.3

June 21, 2022

UNIVERSALDOT FOUNDATION



UNIVERSAL.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

UNIVERSAL.

Revision History

Name	Revision comment	Date
Igor Stojanov	Initial draft	21/06/2022
Igor Stojanov	Adding references and addendums v.02	28/06/2022
Igor Stojanov	Architectural design and reference v.03	06/07/2022
Igor Stojanov	Finalized Data Models and requirements	09/08/2022
Igor Stojanov	Added License information v.04	14/09/2020

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

UNIVERSAL.

Summary

Decentralized storage is a necessity for any decentralized application. Since **on-chain data** storage tends to be non-viable for many use cases, a good way to ensure information integrity is to store on-chain only pointers to data, while the actual files are stored **off-chain**. All large files such as media and documents are stored through separate decentralized storage and the access is managed through business logic that comes from the blockchain.

Currently, on the market, there are a plethora of open-sourced and closed-source solutions for decentralized data storage. **IPFS** is one of the leading open-source peer-2-peer decentralized storage solutions. As such, we have chosen to integrate with the most suitable and mature technology stack that solves the problem of storing large files in a decentralized system.

The rest of this document outlines how we intend to use **IPFS** within the application, its functional and non-functional requirements as well as the architectural design of the solution.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

UNIVERSAL.

IPFS

Overview

IPFS is a **peer-to-peer (p2p)** storage network. Content is accessible through peers located anywhere in the world who might relay information, store it, or do both. IPFS knows how to find what you ask for using its content address rather than its location.

There are **three fundamental principles** to understanding IPFS:

- Unique identification via content addressing
- Content linking via directed acyclic graphs (DAGs)
- Content discovery via distributed hash tables (DHTs)

These three principles build upon each other to enable the IPFS ecosystem.

IPFS is a **natural fit for blockchain** use cases. The common state of the chain is distributed on-chain among participants, and specific data is stored on IPFS. Thanks to content addressing, the blockchain only needs to store the IPFS multi hash, and users are sure to fetch correct data from any of their peers. This architecture is becoming the de facto standard for blockchain applications.

Merkle DAG

At the heart of IPFS is the MerkleDAG, a directed acyclic graph whose links are hashes. This gives all objects in IPFS useful properties:

- **authenticated**: content can be hashed and verified against the link
- **permanent**: once fetched, objects can be cached forever
- **universal**: any data structure can be represented as a merkledag
- **decentralized**: objects can be created by anyone, without centralized writers

In turn, these yield properties for the system as a whole:

- links are content addressed
- objects can be served by untrusted agents
- objects can be cached permanently
- objects can be created and used offline
- networks can be partitioned and merged
- any data structure can be modeled and distributed

IPFS is a stack of network protocols that organize agent networks to create, publish, distribute, serve, and download merkledags. It is the authenticated, decentralized, permanent web.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

Nodes and Network model

The IPFS network uses a PKI-based identity. An "ipfs node" is a program that can find, publish, and replicate merkledag objects. Its identity is defined by a private key. Specifically:

```
privateKey, publicKey := keygen()

nodeID := multihash(publicKey)
```

All hashes in ipfs are encoded with multi hash, a self-describing hash format. The actual hash function used depends on security requirements. The cryptosystem of IPFS is upgradeable, meaning that as hash functions are broken, networks can shift to stronger hashes. There is no free lunch, as objects may need to be rehashed, or links duplicated. But ensuring that tools built do not assume a pre-defined length of hash digest means tools that work with today's hash functions will also work with tomorrow's longer hash functions too.

As of this writing, IPFS nodes *must* support:

- sha2-256
- sha2-512
- sha3

The Stack

IPFS has a stack of modular protocols. Each layer may have multiple implementations, all in different modules. This spec will only address the interfaces between the layers and briefly mention possible implementations. Details are left to the other specs.

IPFS has five layers:

- naming - a self-certifying PKI namespace (IPNS)
- merkledag - data structure format (thin waist)
- exchange - block transport and replication
- routing - locating peers and objects
- network - establishing connections between peers

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

These are briefly described bottom-up.

3.1 Network

The network provides point-to-point transports (reliable and unreliable) between any two IPFS nodes in the network. It handles:

- NAT traversal - hole punching, port mapping, and relay
- supports multiple types of transport - TCP, SCTP, UTP, ...
- supports encryption, signing, or clear communications
- multi-multiplexes -multiplexes connections, streams, protocols, peers, ...

3.2 Routing -- finding peers and data

The IPFS Routing layer serves two important purposes:

- peer routing -- to find other nodes
- content routing -- to find data published to ipfs

The Routing System is an interface that is satisfied by various kinds of implementations. For example

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

- DHTs: perhaps the most common, DHTs can be used to create a semi-persistent routing record distributed cache in the network.
- mdns: used to find services advertised locally. mdns (or dnssd) is a local discovery service. We will be using it.
- snr: supernode routing is a delegated routing system: it delegates to one of a set of supernodes. This is roughly like federated routing.
- dns: ipfs routing could even happen over dns.

3.3 Block Exchange -- transferring content-addressed data

The IPFS Block Exchange takes care of negotiating bulk data transfers. Once nodes know each other -- and are connected -- the exchange protocols govern how the transfer of content-addressed blocks occurs.

The Block Exchange is an interface that is satisfied by various kinds of implementations. For example:

- Bitswap: our main protocol for exchanging data. It is a generalization of BitTorrent to work with arbitrary (and not known apriori) DAGs.
- HTTP: a simple exchange can be implemented with HTTP clients and servers.

3.4. Merkledag -- making sense of data

As discussed above, the IPFS merkledag (also known as IPLD - InterPlanetary Linked Data) is the data structure at the heart of IPFS. It is an acyclic directed graph whose edges are hashes. Another name for it is the merkleweb.

The merkledag data structure is:

```
message MDagLink {
  bytes Hash = 1;    // multihash of the target object
  string Name = 2;   // utf string name. should be unique per object
  uint64 Tsize = 3;  // cumulative size of target object
}

message MDagNode {
  MDagLink Links = 2;  // refs to other objects
  bytes Data = 1;      // opaque user data
}
```

The merkledag is the "thin waist" of authenticated data structures. It is a minimal set of information needed to represent + transfer arbitrary authenticated data structures. More complex data structures are implemented on top of the merkledag, such as:

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

- git and other version control systems
- bitcoin and other blockchains
- unixfs, a content-addressed unix filesystem

See more in the IPLD spec.

3.4.1 Merkledag Paths

The merkledag is enough to resolve paths:

`/ipfs/QmdpMvUptHuGysVn6mj69K53EhitFd2LzeHCmHrHasHjVX/test/foo`

- (a) Would first fetch + resolve QmdpMvUptHuGysVn6mj69K53EhitFd2LzeHCmHrHasHjVX
- (b) Then look into the links of (a), find the hash for test, and resolve it
- (c) Then look into the links of (b), find the hash for foo, and resolve it

See more in the path resolution spec.

Let's resolve this path

/ipfs/QmdpMvUptHuGysVn6mj69K53EhitFd2LzeHCmHrHasHjVX/test/foo

IPFS

3.5 Naming -- PKI namespace and mutable pointers

IPFS is mostly concerned with content-addressed data, which by nature is immutable: changing an object would change its hash -- and thus its address, making it a *different* object altogether. (Think of it as a copy-on-write filesystem).

The IPFS naming layer -- or IPNS -- handles the creation of:

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

- mutable pointers to objects
- human-readable names

IPNS is based on SFS. It is a PKI namespace -- a name is simply the hash of a public key. Whoever controls the private key controls the name. Records are signed by the private key and distributed anywhere (in IPFS, via the routing system). This is an egalitarian way to assign mutable names in the internet at large, without any centralization whatsoever, or certificate authorities.

See more in the IPNS spec.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

Requirements

The integration with IPFS has some requirements related to its functionality. These abstract functional requirements are outlined in the table below.

Functional

Requirement #	Description	Details
Req 1.	The front end of the application should be decentralized	
Req 2.	Large files should be stored in IPFS, and their Hash should be stored on substrate on-chain	
Req 3.	Organizations should be able to create a private cluster of IPFS nodes and should be able to share this information with other organizational members privately	
Req 4.	The following documents should be stored in IPFS by default: Profile: - Additional Information Task: - Specifications - Attachments Dao: - Vision	See Data Models for more details
Req 5.	There should be dockerized deployment through docker-compose.	There should be IPFS configuration options in the front-end that allow users to configure to which node, or peers they would have to connect
Req 6.	There should be a front-end user interface that allows the management of personal files.	This can be inspired from https://github.com/ipfs/ipfs-webui

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

Non-functional

Quality Attribute	Sub-attribute	Description
Security	Direct communication	Operations have to be submitted directly to the blockchain nodes
	SSL, TLS, WSS	All communication happens through Secure Transport Layer
Usability	User-friendliness	The IPFS user interface should provide a convenient, non-technical UI & UX
	Human-readable errors	The web app should provide any errors/warnings in non-technical explanation to the user
	User interface aesthetics	The UI elements (shapes, font, colors, sizes) should align with each other to provide a pleasing and satisfying experience for a user
	Known terminology	The IPFS user interface will have similar terminology as other web-based projects.
	Accessibility, language	The IPFS user interface should be available to be used in different languages & region support (e.g. locale currency format)
	Accessibility, visual impairment	Large font size-mode
Performance	Speed	The IPFS user interface should be able to process information fast by making API calls through secure web socket
	Optimized networking	API calls should be focused on specific features, avoiding generalized calls.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

Reliability	100 % uptime	The IPFS user interface should be available at all times
	Fault tolerance	The IPFS user interface should be developed & tested to avoid crashes
	Recoverability	If operation failed the IPFS user interface should provide an opportunity to re-send the operation again, without a need to re-construct & re-sign everything from scratch.
Updatability	Deployment	The Deployment of the application should be automated and should be straightforward. Shall support Docker-based deployment
	Runtime updates	The IPFS user interface should be able to fetch & adapt to runtime updates
Modularity	Modularity	The deployment should be modular based on individual computing preferences
Accessibility	Easy of use	The web app should be available on all major web browsers.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

Architecture Design

A fully decentralized application requires a large amount of computing power to execute transactions and existing business logic. Due to the large compute power needed, hereby we provide two different deployment versions of the application.

- **Full Node** deployment contains the software complete stack.
- **Light Node** contains only the necessary elements.

Full Node

A full node allows for all parts of the application to be run locally. Thus a full node contains all necessary storage and processing power on an individual machine, which includes the following:

- Substrate Node
- Front End application
- Tensorflow Serving
- IPFS Node and IPFS Cluster

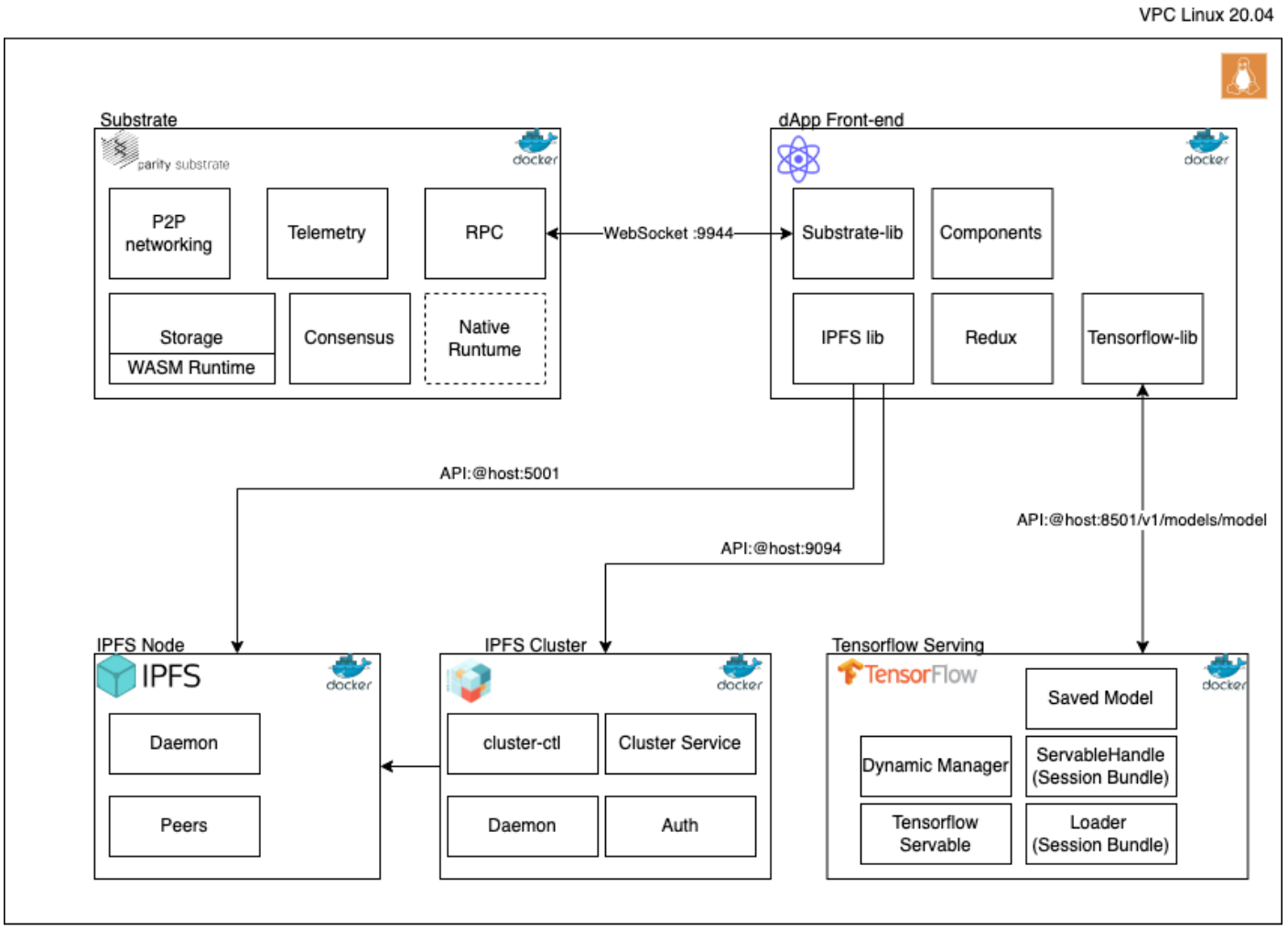
Participants that are running the whole technology stack locally, will be financially incentivized to do so.

A Full Substrate Node can serve as a Validator and as such will be able to earn staking rewards that are issued each Era.

Tensorflow Serving will be able to be used as a Service where users will be able to connect to Tensforlow Nodes that will serve their requests. Tensorflow Sering Nodes can choose their revenue models such as charging per request or monthly bundles. It is essential that there are community-offered TensorFlow Serving Servers that will provide Services for free. The Universaldot Foundation will run several such community-offered Tensorflow services.

IPFS Nodes can also provide a gateway as a service to other participants that do not want to pin files to their local computing device. Different revenue services can be established to support this. The Universaldot Foundation will also run several free community-offered IPFS Gateways.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation



Full Node Architecture diagram

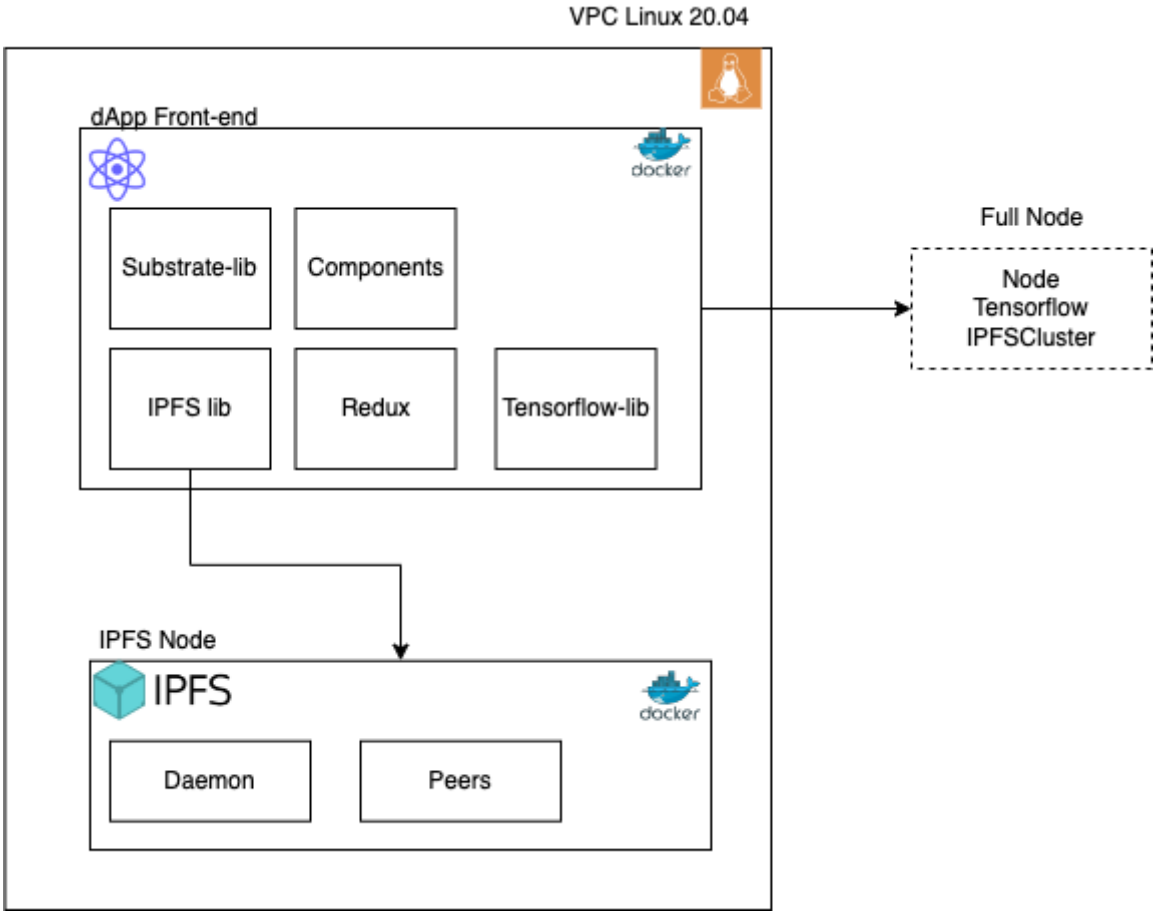
KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

Node

A Light node contains a smaller subset of the application. As such it is intended for regular users who can access the application locally, but delegate processing power and trust to external Full Nodes. A light node contains the following components:

- Front End Application
- (IPFS Node) - optional

A Light Node will allow regular users to access the application while delegating compute power to other participants on the network that will run Full Nodes. In addition to entrusting computer power, they will also entrust other participants with the validity of the data they are providing.



Light Node Architecture Diagram

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

UNIVERSAL.

Data Model

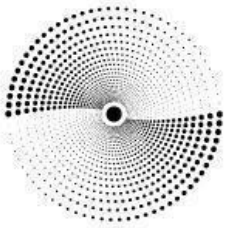
The application can store global static resources in IPFS which are referenced in the front-end application. These resources could be images, fonts, styles, and other static resources which can be accessed by the application itself.

Profile

- The data below is stored in IPFS in JSON format.
- The metadata is associated with on-chain data through AccountID.
- The Metadata is allowed to mutate. Its mutation has to be based on an agreed interface (API).
- The AccountID is the only piece of immutable data, that is used as a key to link the rest of the metadata.

Characteristic	Description	Type	Entry
AccountID (On Chain)	Primary ID for a profile. One profile per AccountID	Pub key	Automatic
Location	Location of the user	String (Geolocation)	Manual, Optional
JobTitle	Self assigned job description	String	Manual, Optional
Email	personal email of user	String	Manual, Optional
Education	Educational institutions the user has attended	String or String array	Manual, Optional
Image	Profile picture of user	file	Manual, Optional
Language	Languages that the user speaks	String array	Manual, Mandatory

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation



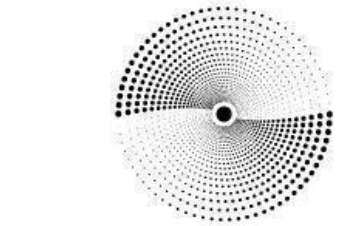
UNIVERSAL.

Task

- The data below is stored in IPFS in JSON format.
- The TaskID is the key ID through which the rest of the data gets associated with.
- In cases when data is stored off-chain, its associated IPFS hash can be stored on-chain.

Characteristic	Description	Type	Entry
TaskID (On Chain)	Primary ID for a Task.	HashID	Automatic
Deliverable	Files to be delivered as part of deliverable	File	Manual
Review	Feedback after work is completed	String	Manual
Attachments (refactor from Substrate)	Keep all attachments related to order on IPFS	File(s)	Manual
Feedback (refactor from substrate)	Feedback should be kept off-chain	String	Manual
Specification (refactor from substrate)	Specification should be kept off-chain	String	Manual

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation



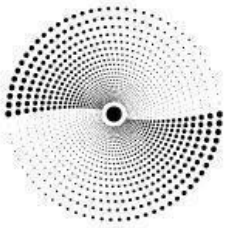
UNIVERSAL.

Dao

- The data below is stored in IPFS in JSON format.
- The OrgID is the key ID with which the rest of the data gets associated with.
- In cases when data is stored off-chain, its associated IPFS hash can be stored on-chain.

Characteristic	Description	Type	Entry
OrgID (On-Chain)	Hash of Dao and primary ID for an organization	HashID	Automatic
Vision Document	Store vision document in IPFS	File	Manual
Description	Store all relevant descriptions regarding the organization	String	Manual
Media	Public Media files associated with the organization	Files	Manual
Similar Organizations	Links to similar organizations	Array	Automatic

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation



UNIVERSAL.

References

[1] IPFS API - http://docs.ipfs.io.ipns.localhost:8080/reference/http/api/

[2] IPFS Cluster API - http://cluster.ipfs.io.ipns.localhost:8080/documentation/reference/api/

[3] IPFS Docs - http://docs.ipfs.io.ipns.localhost:8080/concepts/how-ipfs-works/#how-ipfs-works

[4] IPFS Architecture - https://github.com/ipfs/specs/blob/main/ARCHITECTURE.md

[5] IPFS Github - https://github.com/ipfs

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS

Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation