dadi

White paper

Decentralized web services.

A new era of cloud computing services,
powered by blockchain technology.

Q4 2017

Website – dadi.cloud

Contents

1. Preface

The Internet was conceived as a global, democratic communication network - a mesh of computers where information and power were equally distributed.

This principle is under attack from all sides. Companies like *Google*, *Facebook* and *Amazon* are centralizing control, while powerful lobbying groups are working to undermine net neutrality.

We are facing a future where you are the customer – and the product – of a network controlled by only a handful of global corporations.

But what if we reversed this relationship? What if we collectively owned the network, while also being the beneficiaries of the revenue that comes from its use?

Introducing DADI, a decentralized platform for web services, built on the Ethereum blockchain and committed to upholding the founding principles of the Internet by democratizing computational power.

DADI represents a fundamental shift in cloud computing services by enabling peer-to-peer collaboration on a massive scale. Our technology finds the closest and most appropriate hub of computational power and distributes tasks accordingly.

Any device - from a smartphone to a supercomputer cluster - can join the network and sell computational power as a miner.

On the opposite side of this transaction, platform users benefit from significant reductions in cost, enhanced security, rate flexibility and cutting edge performance.

In stark contrast to other ICO events, the technology of DADI is already here. The microservices at the heart of the platform have been in development for four years and are in production today, powering products like *Empire Online*, *Monocle* and *Virgin Limited Edition*.

DADI is to the cloud what the cloud was to owned infrastructure. It's tackling a $250-billion-dollar industry[1] head on.

[1] http://www.gartner.com/newsroom/id/3616417

2. Introduction

2.1. What is DADI?

DADI is a global, decentralized cloud platform, focused on the provision of web services to help you build, scale and grow your digital products.

With DADI there is no single authority that regulates computing resource distribution. The platform uses cost-efficient fog computing[2] organized by a **Decentralized Autonomous Organization (DAO)** rather than a centralized cloud structure, and DADI's web services are organized around a microservices architecture that provides a series of intelligent apps for building digital products.

To setup an API in *AWS*, you would first have to write an API wrapper in your language of choice, then you would need to rent an instance, install the required supporting software, push your code and move into a maintenance cycle. Things get even more complex if you need to scale to meet growing demand.

With DADI, your only concern is your data. DADI API - one of DADI's core web services - is available on demand, and will scale to meet your requirements seamlessly and automatically.

Every digital product is powered by web services. Every business using the web can already leverage DADI technology for improved efficiency and performance, but soon the DADI network will be able to support these web services at a price point unimaginable today – some 90% cheaper than traditional cloud services such as AWS. Plus, anyone with a laptop, games console, mobile phone or any connected device will be able to earn income by providing spare computation capacity for rent.

DADI represents a radical overhaul of the cloud computing sector. Its mission is to uphold the founding principles of the Web by democratizing computational power.

DADI: **D**ecentralized **A**rchitecture for a **D**emocratic **I**nternet

[2] https://en.wikipedia.org/wiki/Fog_computing

2.2. Business overview

DADI is a little over four years old and has spent the majority of that time in R&D, building out a series of interconnected web services for individuals, companies and governments. DADI's web services are in production with some of the most recognizable global brands, including *Virgin Limited Edition*, *Empire* and *Monocle*, proving the core technology at scale.

The DADI team is 18 strong, majority engineers, and all senior in their own fields. The company operates an all-remote setup, with a focus on asynchronous working designed to facilitate a healthy life/work balance.

Further reading: How we work

DADI is financed through cash reinvested by the business and short-term debt. It has a current MRR of c.$165,000, built on a handful of early stage Enterprise engagements. MRR is expected to raise to c.$210,000 through Q1 2018 with a strong pipeline pushing through the rest of the year. The company's current position demonstrates the huge potential in the platform.

Further reading: DADI Business Overview

2.2.1. Commitment to R&D

DADI's founders have invested c.$2 million in direct R&D to date, building out the web services at the heart of the platform. At the time of writing there have been **5,503 commits** from **38 contributors** to the platform, along with **273 releases**.

This effort will increase post Crowdsale through the expansion of DADI roadmaps backed by additional resource within the core engineering team.

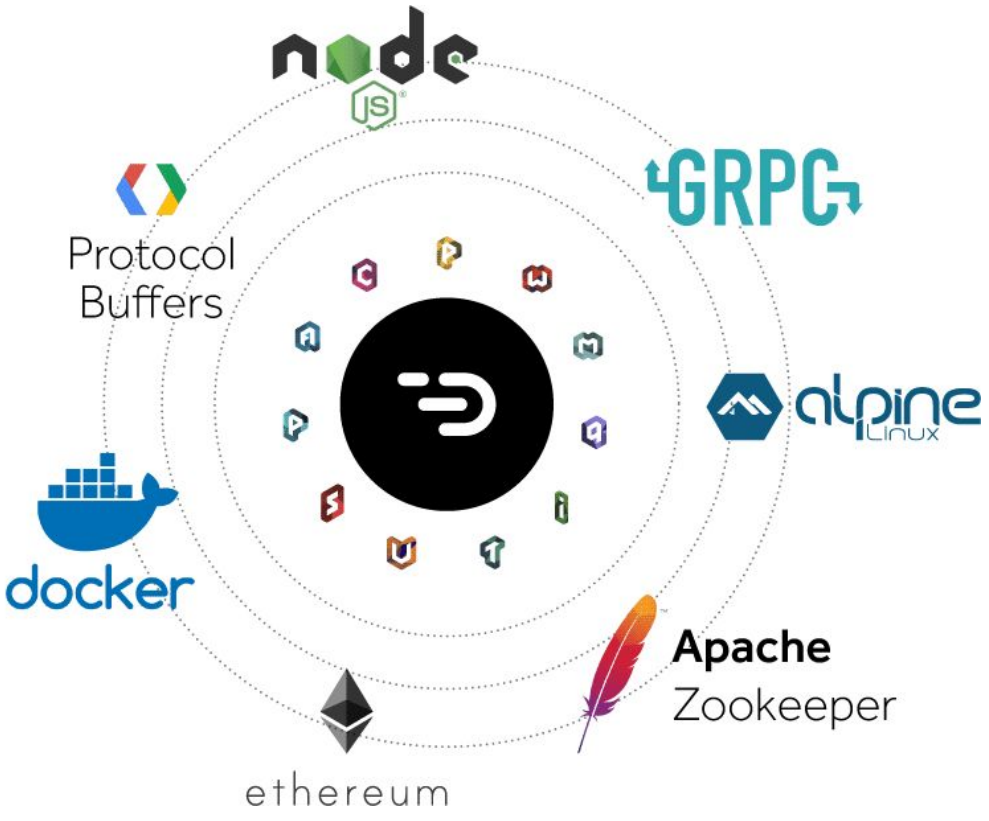
2.3. Technology overview

DADI technology has been in development for four years. The web services at the heart of the platform are currently in production, delivering solutions for leading brands worldwide.

You can think of DADI as a containerized layer on top of established P2P technologies - Protocol Buffer for performant language neutral data transfer and GRPC for distributing procedures across multiple hosts.

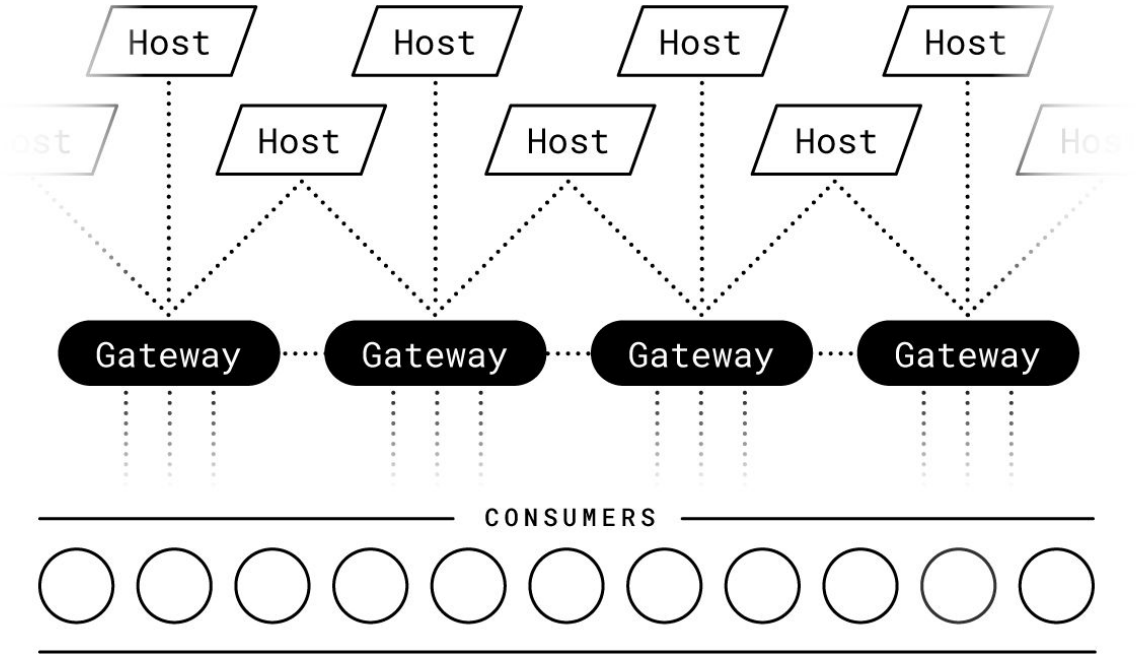
DADI's hybrid architecture layers a series of Open Source web services on top of **container services** and **gateway layers** that are distributed throughout the network. Working in this way removes the potential of performance impact form congestion on the Ethereum network.

2.3.1. High level platform architecture



2.3.2. Network players

There are three key groups within the DADI Network: Hosts, Gateways (including Stargates), and Consumers. **Hosts** are network node owners who contribute computational power. DADI Web Services run in a container service within Host environments. **Gateways** are network node owners who contribute bandwidth. They are the entry point to the network, acting as an aggregate point for Host node capacity and providing the domain name system that makes Host resources addressable. **Consumers** are the users of the network – individuals, businesses, governments and their customers.



Note: Gateways and Hosts can be thought of as Miners

2.3.3. Hosts

Hosts run the DADI container service, which is implemented using VMs and Docker. Each node runs the host application as well as the client application within a single encrypted VM. Essentially, a Host node is a collection of services and a service locator.

DADI Web Services run within the container service on the Host. More information can be found in section 3.3.1 below.

2.3.3.1. Web Services

DADI Web Services provide streamlined, elegant solutions to common tasks to help you build, scale and grow your digital products.

There are ten key services in development (five of which are functionally complete and in production), with more planned for future roadmapped development:

- DADI API*
- DADI CDN*
- DADI Identity
- DADI Match
- DADI Predict
- DADI Publish*
- DADI Queue*
- DADI Track
- DADI Visualize
- DADI Web*

** Functionally complete and available for production use today*

DADI Web Services are interconnecting Node.js applications, with shared modules and supporting tools, including a command line interface (CLI) designed to simplify the setup and deployment of the technology.

Source code for existing web services can be found here: https://github.com/dadi

2.3.4. Gateways

You can think of a DADI Gateway as the master node in an ad hoc cluster comprising one Gateway and many Hosts. The Gateway is the entry point for Consumer requests. It handles Host authentication, translates requests to Jobs, manages a Job request pool, maintains a Pub/Sub messaging system and verifies responses from Hosts.

Stargates are identical to Gateways, but come with the addition of a DNS layer - an authoritative nameserver - that manages domain mapping into the DADI Network; a Zoopkeeper installation that maintains the VPC; a CLI interface that manages container deployment to Hosts; and a contract management layer for Consumer-DADI negotiation.

2.4. The DADI token

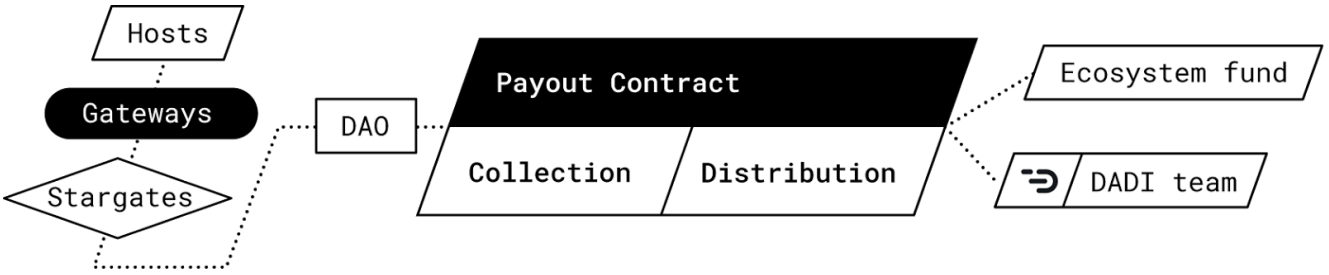
2.4.1. The DADI token

DADI uses its own ERC20 token (DADI), providing distributed value for Crowdsale participants.

Further reading: DADI Business Overview

The creation of DADI tokens will be a one time event. The Token Creation event is the only time that these tokens can be created, and therefore the total supply of DADI tokens is fixed.

DADI tokens are an integral part of the DADI Platform. Consumers will be charged tokens for their usage of DADI Web Services. The biggest slice of those tokens will go to Hosts, with a percentage of every Host's tokens paid to the Gateways that they are attached to. A smaller percentage is retained and dedicated to supporting the underlying connectivity of the network. A similar percentage is paid to the DADI team for the ongoing development and iteration of the platform.



A small portion of DADI tokens will be continually burnt, tied to platform usage. The domain name system within DADI takes a micropayment per DNS request satisfied through the Stargates, which is assumed to be the equivalent of a Wei in the first instance (the smallest unit of Eth at 0.000000000000000001).

Platform fees for Consumers will initially be denominated in DADI. An exchange will be built into front end interfaces, allowing Consumers to purchase services in their currency of choice.

2.4.2. Hosts

DADI Hosts - network node owners who contribute computational power - are incentivized for their participation in the network. You can think of Hosts as miners, with rewards provided in the form of DADI tokens.

The availability of Host resources is managed through a proof of work system that periodically reports activity against specific service contracts, updating a reputation system that is used to favour the most performant Hosts.

Hosts announce themselves to the network by time locking a small volume of DADI tokens. This also feeds the reputation system and helps to mitigate the risk of Sybil attacks, as valuable locks are not trivial to create.

If you have spare computational power DADI allows you to generate tokens while delivering real world services for individuals, businesses and governments.

2.4.3. Gateways

DADI Gateways are network node owners who predominantly contribute bandwidth. Gateways handle network negotiation and provide edge caching for Host-processed content. Gateways are rewarded with a percentage of the tokens from all of the Hosts connected to them, with rewards provided in the form of DADI tokens.

The availability of Gateways is managed through a proof of bandwidth system that periodically reports activity against specific service contracts and Host connectivity, updating a reputation system that is used to favour the most performant Gateways.

Gateways announce themselves to the network by time locking a large volume of DADI tokens (an order of magnitude larger than that required of Hosts). As with Hosts, this feeds the reputation system and helps to mitigate the risk of Sybil attacks.

2.4.3.1. Stargates

Stargates provide vital functions in the network in the form of domain name mapping and Consumer-DADI contract negotiation. They are rewarded with a percentage of all Consumer tokens.

Stargates announce themselves by fully committing a very large volume of DADI tokens – **a buy-in system that ensures a high level of capacity and consistency across the network**.

Stargates are intended for very high bandwidth environments, such as data center owners/operators. Their initial deployment will be handled directly by DADI, who will install Stargates in key regions ahead of the network launch.

Application for status as a Stargate is manual, but like Gateways, Stargates are assessed under the *reputation system*, which grades and prefers nodes based on performance.

2.5. Use cases

2.5.1. Content and publishing platforms

Content management is a primary use case for DADI. It combines flexible editorial interfaces with an API-first/COPE (Create Once, Publish Everywhere) philosophy to future-proof content storage and facilitate rapid product development.

DADI also significantly reduces operating costs and provides for a data-driven strategy – both in terms of personalizing content or commercial opportunity and in delivering detailed analytics from which to make informed product development decisions.

2.5.1.1. Live example: Empire

Bauer Xcel Media chose DADI Web Services for its refresh of Empire in Summer 2015, and delivered a fully rebuilt site eight weeks later.

The new product is an API-first development, built in support of lean editorial workflow, clutter-free user experience across multiple devices and enhanced SEO functionality.

Bauer is now working on a release to include the latest version of DADI Publish, and is rolling DADI out across other titles in their international portfolio.

“The DADI platform provides unique and market leading functionality.”
Director of Search, Bauer Xcel Media

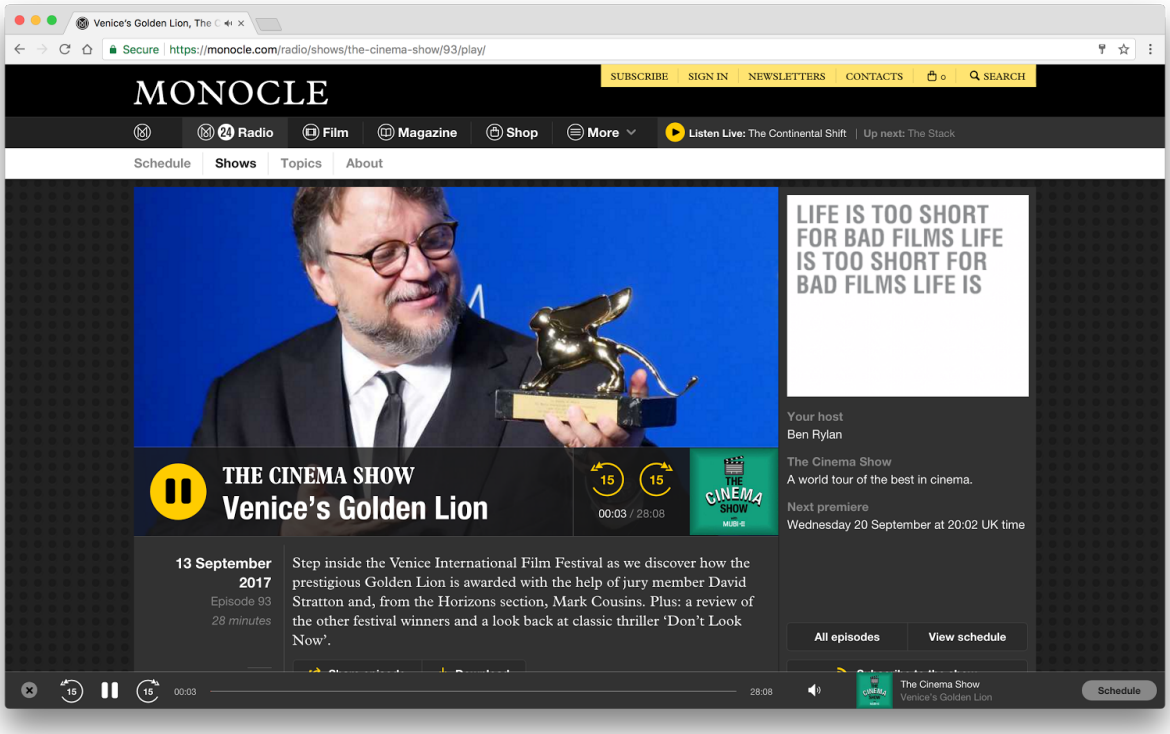
2.5.2. Content manipulation and distribution

DADI facilitates the seamless delivery of image, audio and video assets for digital products accessed across a range of devices in multiple contexts.

DADI CDN serves the right asset at the right moment to save users waiting for content to appear. DADI CDN can be used with other DADI Web Services or alongside other technology to improve user experience and reduce the costs associated with asset management and delivery.

2.5.2.1. Live example: Monocle CDN

Monocle uses DADI CDN for the delivery of image, audio and video assets across all aspects of its digital business, from editorial to ecommerce.



The technology serves content assets for users on mobile, tablet or desktop and takes into account variables such as connection speed to deliver the right balance between performance and quality. Features such as DADI CDN's ability to auto-crop for different aspect ratios while retaining the focal point of the image are especially effective.

"We used CDN to optimise user experience and improve overall performance. The cost saving was the cherry on top."
Head of Digital, Monocle

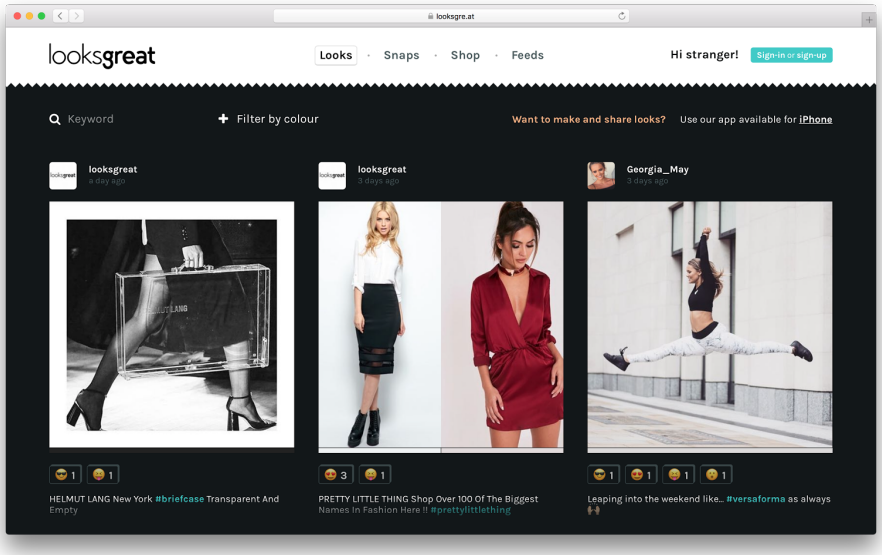
2.5.3. Apps: tablet and mobile

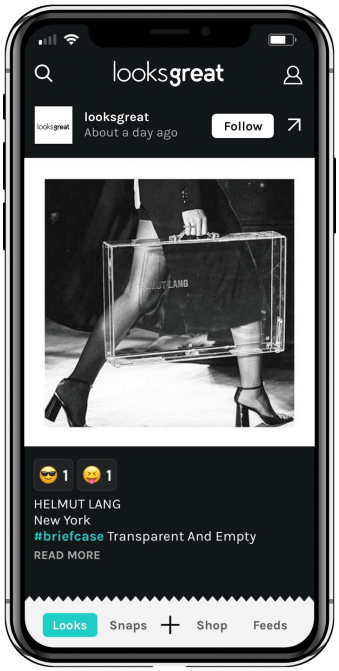
DADI's content management technology – Publish, API and CDN – is a hand-in-glove fit for mobile app development, with an Android/IOS SDK providing a leg up in integration.

The inherent flexibility in these tools makes DADI the ideal platform for editorial or ecommerce projects working across mobile and tablet devices.

2.5.3.1. Live example: Looksgreat

Looksgreat is conceived as the social hub for fashion in the UK – a social media product for fashionistas, new designers and brands to share and rate clothing, accessories and 'looks'.





The product works across responsive web and IOS app builds (with Android coming soon), using a single user view – and is building an audience of engaged users sharing and reacting to different fashion styles.

It was built using DADI's IOS SDK and combines the content management elements of the DADI product suite. It also offers an ecommerce component via affiliate partners for clothing purchase.

2.5.4. ‘Always-on’ products

As a pioneer in decentralized cloud services, DADI’s infrastructure is uniquely performant for products such as news services which need to be ‘always-on’.

Unlike traditional networks there is no way to interrupt delivery of services for large chunks of audience by targeting individual servers. The DADI network is made up of myriad hubs of computational power, making it next to impossible to shut down.

2.5.5. Subscription services

DADI Web Services are as proficient in managing user data as they are content, and can support even complex subscription services as part of a wider CRM capability.

The platform can be used to manage flexible paywalls - or data walls - for access to content or services, and accurately measure users’ access to these services. DADI can also help tune subscription products for maximum conversion.

2.5.6. Cost-sensitive products

A key benefit to the DADI platform is cost-efficiency – it’s decentralized architecture means users can save over 90% compared with traditional cloud providers.

For start-ups sensitive to cost or charities that want to divert more of their budgets away from overheads, DADI technology represents a real step forward in the provision of scale digital services without enterprise bills.

2.5.7. Content classification

DADI has the power to automate the classification and tagging of editorial content using its unique and highly accurate taxonomic framework.

DADI Match works alongside Publish and can be used to retrospectively classify content for improved SEO, search, or to enhance content recommendation tools. It can also function as an assistant to editors while they are creating content.

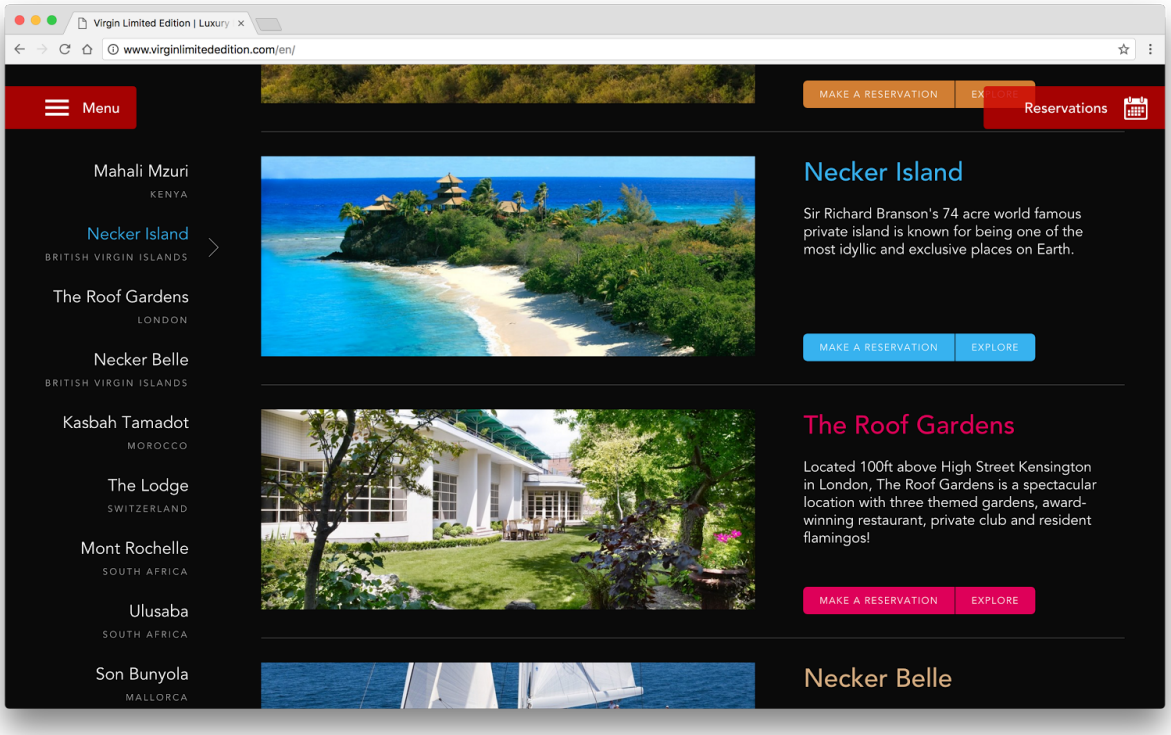
2.5.8. Customer relationship management

DADI Identity provides a ‘single customer view’ for relationship management in digital products and beyond.

A typical application tracks activity to enrich user profiles and support targeted messaging (email or push notifications, for example). User records can also be updated using flexible templates should communication happen face-to-face or over the phone.

2.5.8.1. Live example: Virgin Limited Edition

Virgin Limited Edition uses DADI to manage relationships with its customers as part of a suite of digital services powered by the technology – including its website.



A user data store provides a single version of the truth – vital for being able to tune a customer contact strategy for a demanding demographic.

“DADI has become an essential part of our growth strategy, building not only our ‘shop-front window’ site, but also helping us manage our customer data more intelligently”
Director of Marketing, Virgin Limited Edition

2.5.9. Ecommerce

DADI brings the flexibility of modern content management to retail by facilitating seamless control of product detail and inventory – plus enhanced analytics for forensic measurement and optimization against business targets.

DADI comes into its own when products need to be presented in a more ‘content-alike’ fashion – such as for premium retailers. It makes light work of asset and inventory management and connects with myriad third party providers from payment gateways to in-store POS technology.

2.5.10. Unique user experiences with machine learning

DADI provides for more than a ‘one-size fits all’ approach to user experience. It can be configured to track user activity and make real-time ‘decisions’ about content or commercial opportunity most relevant to users.

This has practical applications in audience retention – giving customers what they want to make them stay – but also in serving opportunities such as native or display advertising, products or services at the right moment to maximize conversion.

2.5.11. Big data

DADI has the capability to act as an umbrella store for user data across multiple products in a digital portfolio. It can also take feeds from third-party providers.

A likely use case for this would be a publisher with multiple titles looking for single user view – DADI Track can provide this by pooling user data across products and, for example, facilitating a single display inventory or allowing for single sign on using DADI Identity.

2.6. Cost efficiency for Consumers

DADI drives huge cost savings in product effort by easing the development and deployment process. Products that typically take months to build can now be tackled in weeks.

"Products built with DADI are delivered in days and weeks, not weeks and months"
Chief Technology Officer, Bauer Media

And running costs when live are a fraction of the cost of a traditional cloud setup. Online automotive magazine *What Car?* saved 65% on their month-to-month infrastructure bill when they moved to DADI Web Services.

Typical cost savings will be 90% vs. *Amazon AWS*, when the full network comes online.

2.7. Key links

Key links in support of the DADI platform:

- Supporting site for the DADI Crowdsale
- Supporting site for DADI Web Services
- Documentation
- Forum
- Discord
- Github
- Twitter
- Reddit

3. DADI technology

3.1. Decentralizing web services

DADI comprises a Gateway VPC made up of ad hoc clusters of contributing Gateways and contributing Hosts. Gateways provide a real-time domain name system, a messaging service and a request pool while Hosts provide a containerization layer for Consumer apps built using DADI Web Services.

This hybrid architecture is designed to answer the key challenges that the blockchain presents when considering its use for web services (sometimes called cloud services or remote computing services).

The largest technical hurdles to decentralizing web services are consumer firewalls and DNS. Firewalls on home networks prevent inbound requests, presenting a hard stop for any system looking to implement a traditional HTTP over TCP client-server framework. DNS currently provides no method of addressing the blockchain, making the mapping of domains from existing TLDs to the blockchain impossible.

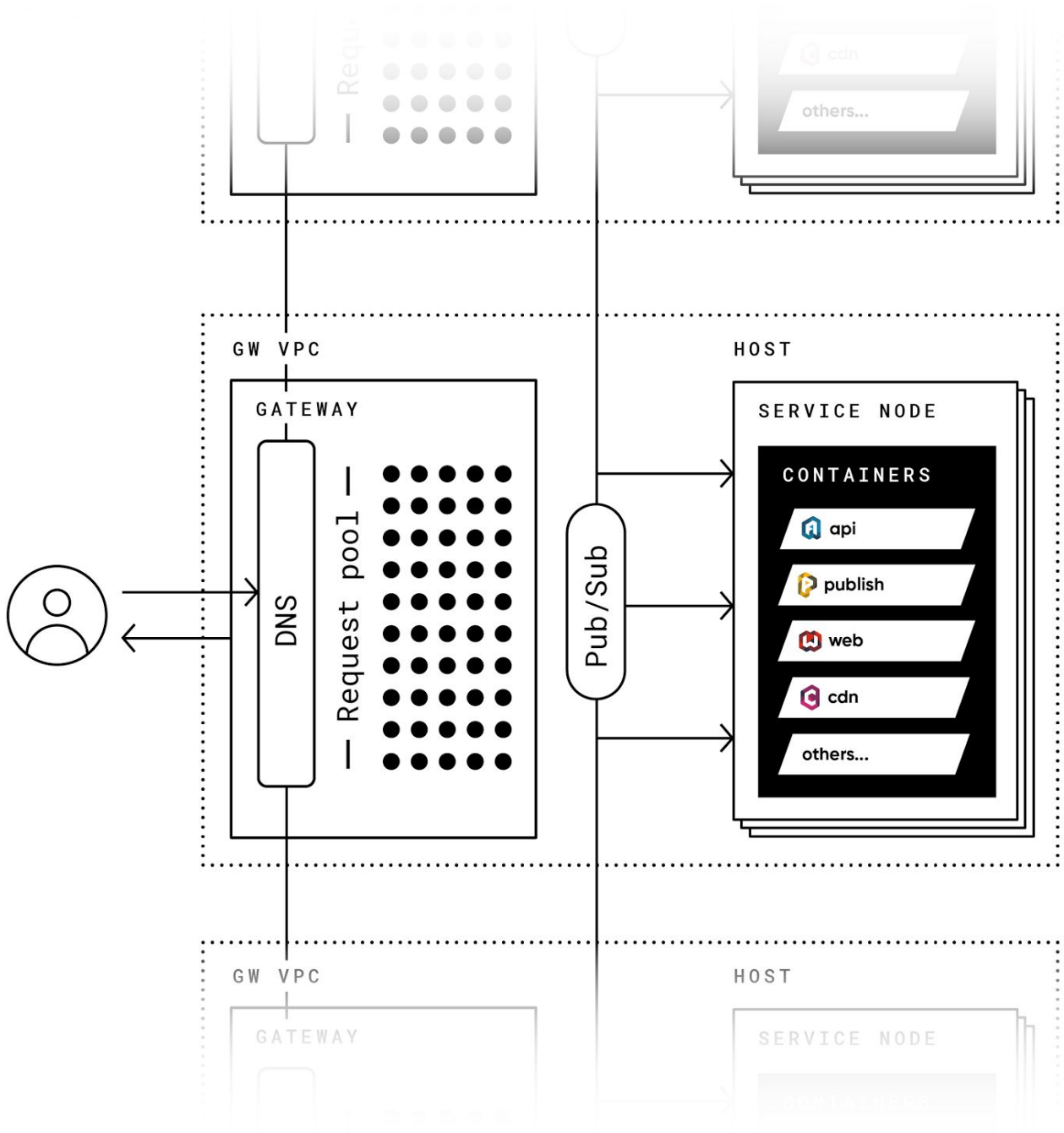
DADI overcomes these hurdles by employing two key node types in the network: Gateways and Hosts.

The relationship between these constructs presents a radical overhaul of traditional client>server architectures whereby a client sends an HTTP request to the web server. DADI's architecture instead presents a system whereby a client sends an HTTP request to a Job request pool, from which workers fetch Jobs, complete them and hand back a response.

In short, DADI takes the concept of a traditional application server - for example a web server such as *Apache* or *NGINX* - and turns it into a client.

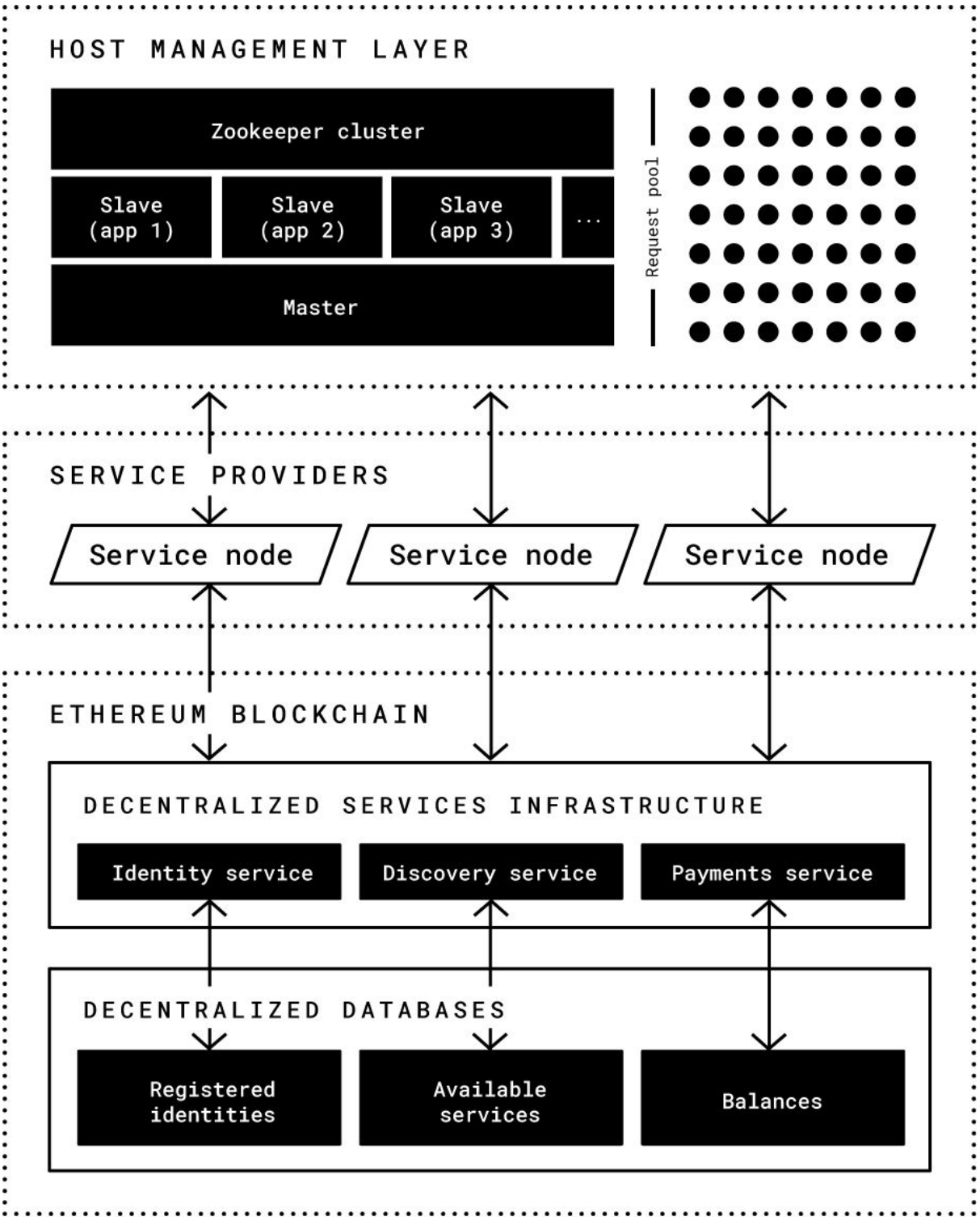
Domain mapping is handled through the bundling of an authoritative nameserver within a special version of a DADI Gateway called a Stargate. Stargates have fixed IP ranges available and run the DNS for dadi.cloud. This enables the mapping of Consumer subdomains - including www. - to *.dadi.cloud subdomains using the CName system, while root Consumer domains can be mapped directly into the VPC on a regional basis.

3.1.3. Platform architecture



3.2. Gateways

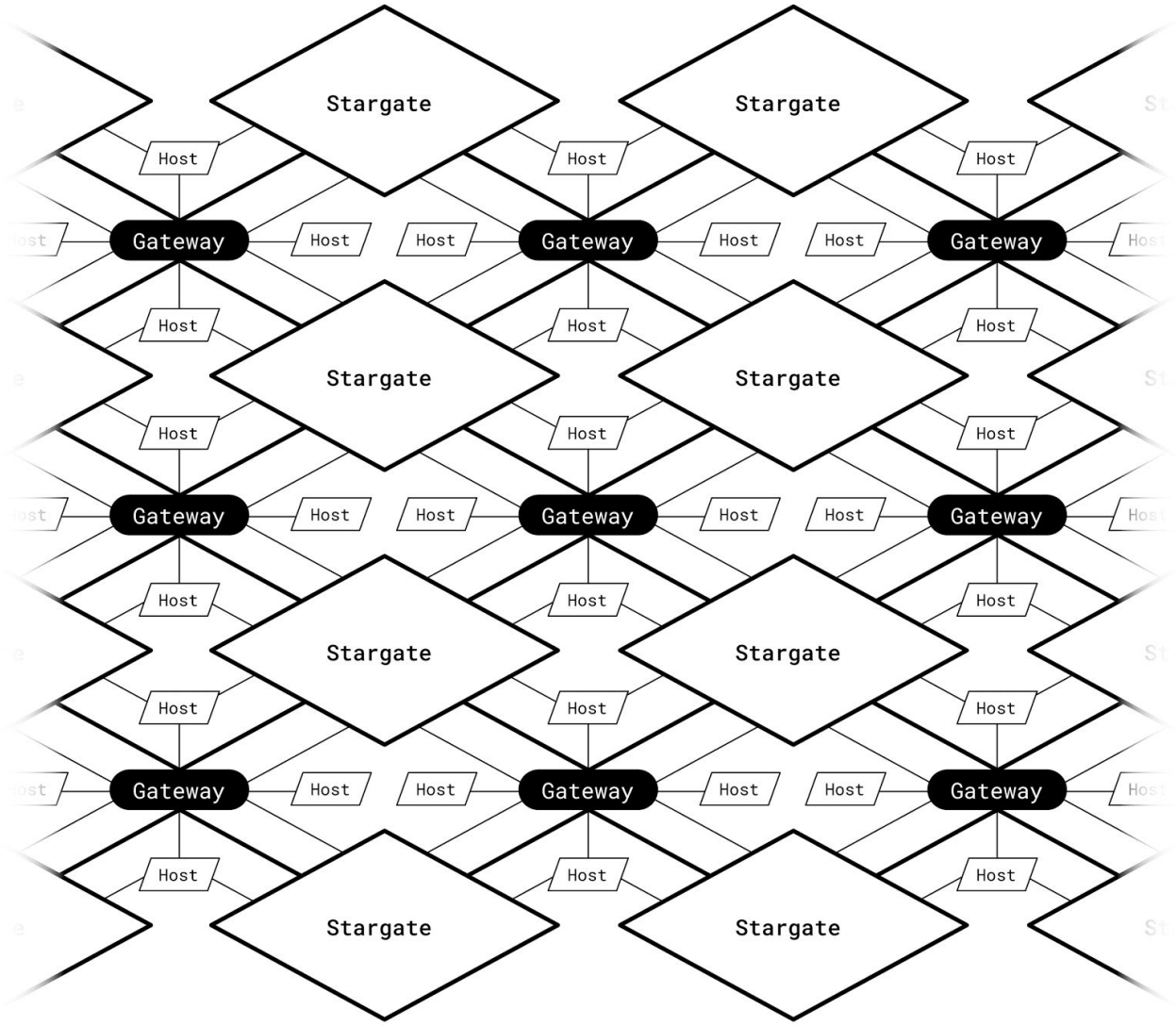
You can think of a DADI Gateway as the master node in an ad hoc cluster comprising one Gateway and many Hosts. The Gateway is the entry point for Consumer requests. It handles Host authentication, translates requests to Jobs, manages a Job request pool, maintains a Pub/Sub messaging system and verifies responses from Hosts.



3.2.1. Stargates

Stargates are identical to Gateways in most respects, but they include the authoritative nameservers for the DADI Network and the CLI interface that manages container deployment to the network. Stargates will be implemented directly by the DADI team in key regions ahead of the launch of the network and will also be available to those who can commit an appropriate level of resources (a high bandwidth environment and the right level of hardware). For example it is expected that Stargates will be operated by key infrastructure providers – several of which the DADI team are in early stage conversation with.

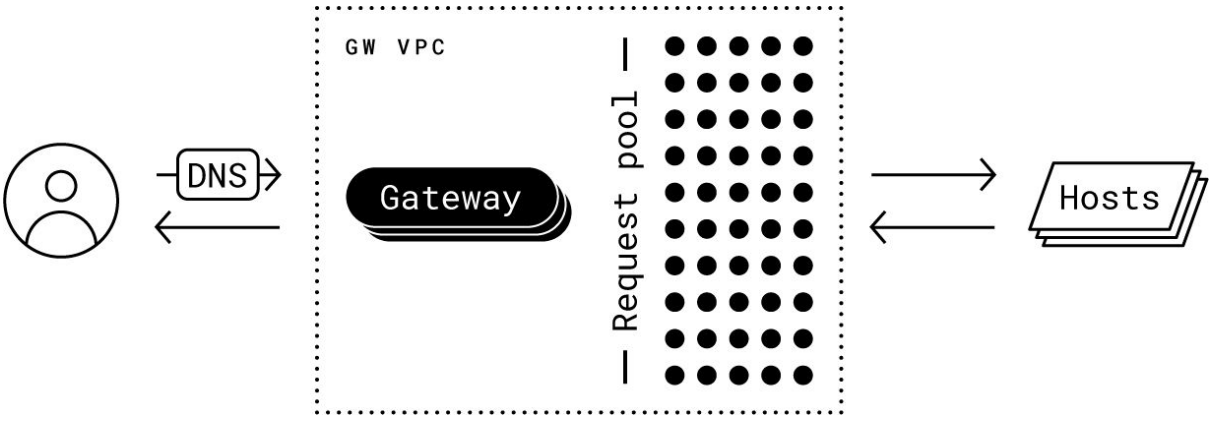
Separating DNS functionality and directly handling the rollout of early nodes enables DADI to guarantee the availability of a network backbone from day one, as well as ensuring the stability of its domain routing system.



3.2.2. The request pool

Requests at this level in the platform are called *Jobs*. *Jobs per second* then is an expression of the scale of the DADI Network at any given time. *Job satisfaction* is used as a measure of performance for Hosts, built into the reputation system and used for request prioritization over time.

All requests to the Gateway VPC enter into a request pool residing in a pub/sub messaging queue. Hosts establish a persistent connection with the pool, filtering jobs based on the Consumer apps that they hold and pushing the response back to the Gateway once processing has completed.



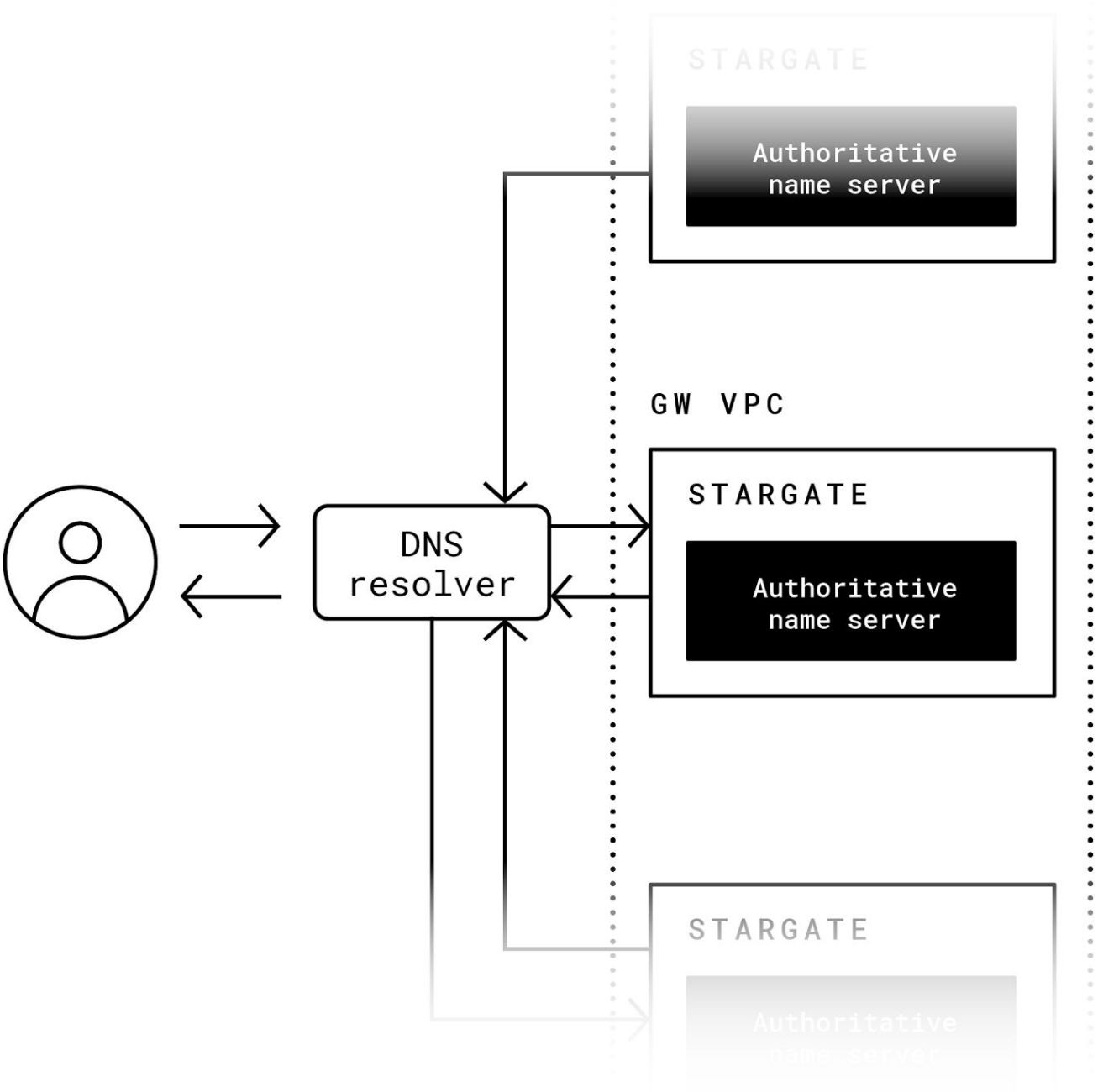
Each job can be processed by any number of the Hosts participating in the Job contract, but reward is paid on a first-past-the-post basis. Hosts that attempted the job but failed to deliver first are forced to exit and return to the pool for another Job.

Hosts must inform the Gateway when they engage in processing a job. This along with their ability to process a job fastest is used to determine which Gateway accepts the Host's authentication request. In this way we match the performance conditions of Gateways and Hosts in order to optimize overall network performance.

Jobs queued in the request pool have a *priority status* based on metrics including mean frequency of request and mean response time. Outperformed Hosts are used less frequently for high priority jobs.

3.2.3. DNS

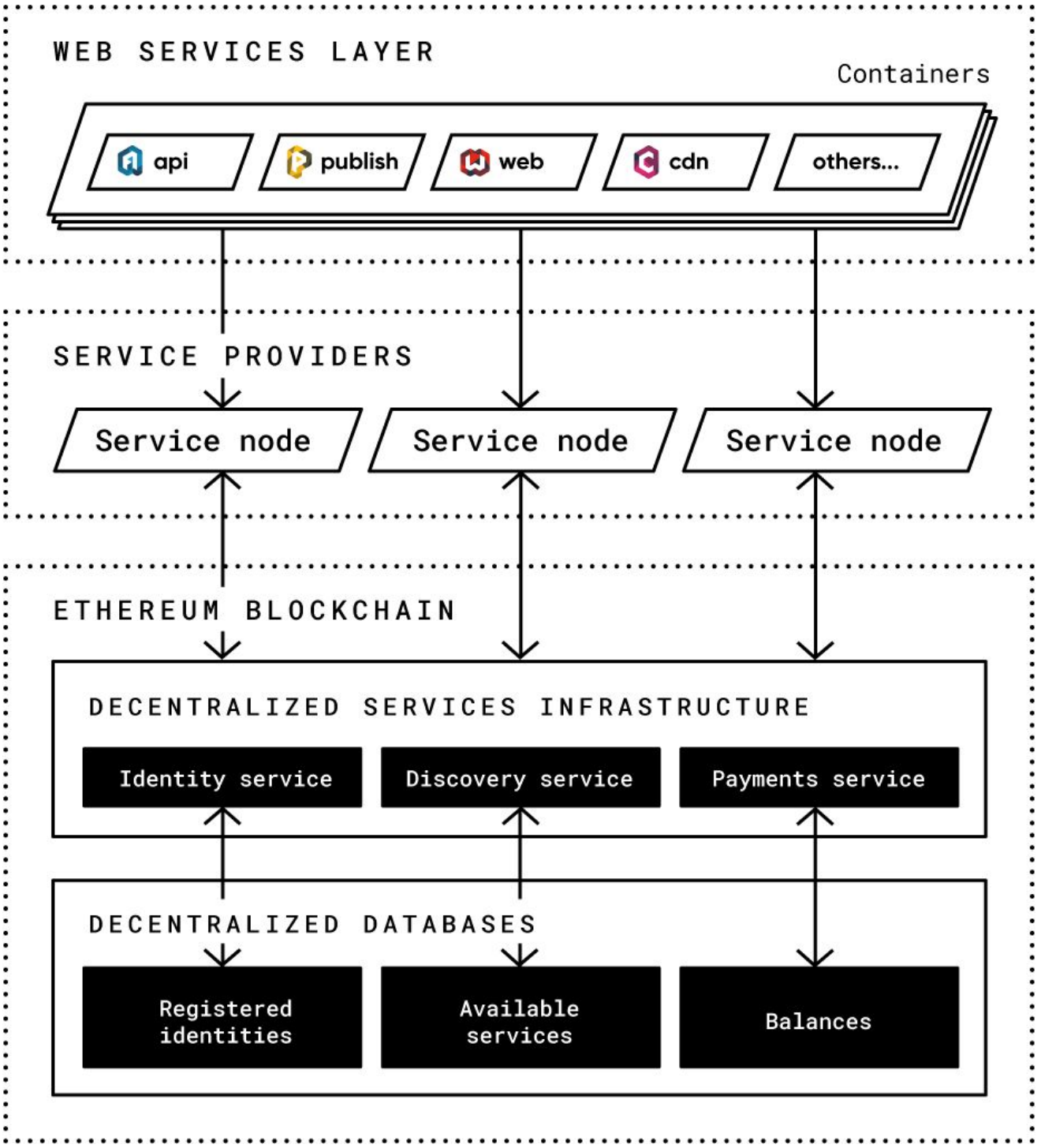
The authoritative nameservers for the DADI Network reside in Stargates. They provide functionality similar to traditional DNS, but with the lowest TTL possible - 5 seconds - and an attached geographical mapping system designed to route requests by region in order to reduce topological distance between the Consumer and the Gateway/Host cluster.



It is expected that existing GO libraries will be utilised for the DNS layer, for example: https://github.com/miekg/dns

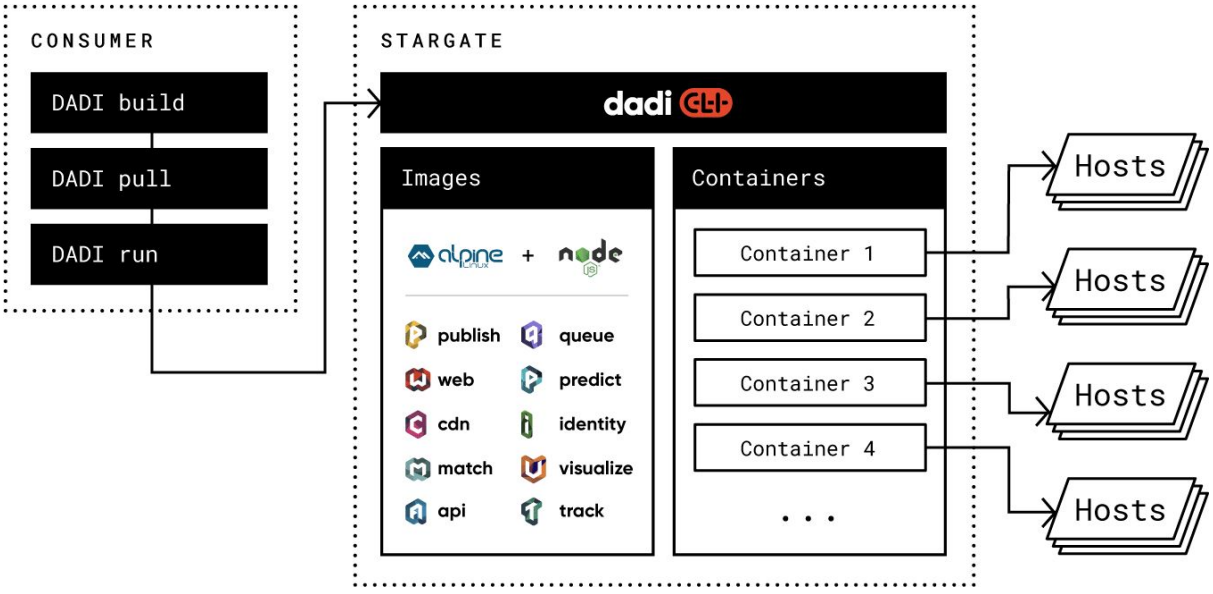
3.3. Hosts

You can think of DADI Hosts as the slave nodes in an ad hoc cluster comprised of one Gateway and many Hosts. The Host provides the computational power for the running of DADI Web Services. Web Services are containerised, both as shared layers (for the provision of a grouped CDN service for example) and individually as bundles of Consumer apps built using the technology.



3.3.1. Containerization: DADI Web Services

DADI makes use of Docker for containerization within Hosts. It is managed behind a CLI interface that resides in Stargates, drawing on centrally maintained images for DADI Web Services to build containers with common components and handle deployments to the Host network on a per contract basis.



3.3.2 Benefits of containerization

Digital Ocean have a great article breaking down the benefits of containerization. You can [read it here](#).

3.4. DADI Web Services

DADI Web Services are organized around a microservices architecture that provides a series of intelligent applications for building digital products. They provide ready-made solutions to common requirements: a platform for virtually every use case.

Further reading: web services deep dive

3.4.1. API

A high-performance RESTful API layer designed in support of API-first development and the principles of COPE.

DADI API is a rapid set-up data layer ready to act as the backbone to your digital platform. It is designed to be plugged into a templating layer, a mobile application or to be used with any other data consumer – and can be used alongside other DADI Web Services.

It can contain your business/domain logic (the part of a platform that encodes the real-world business rules that determine how data is created, displayed, stored and changed). DADI API allows you to get a complete data layer up and running in minutes.

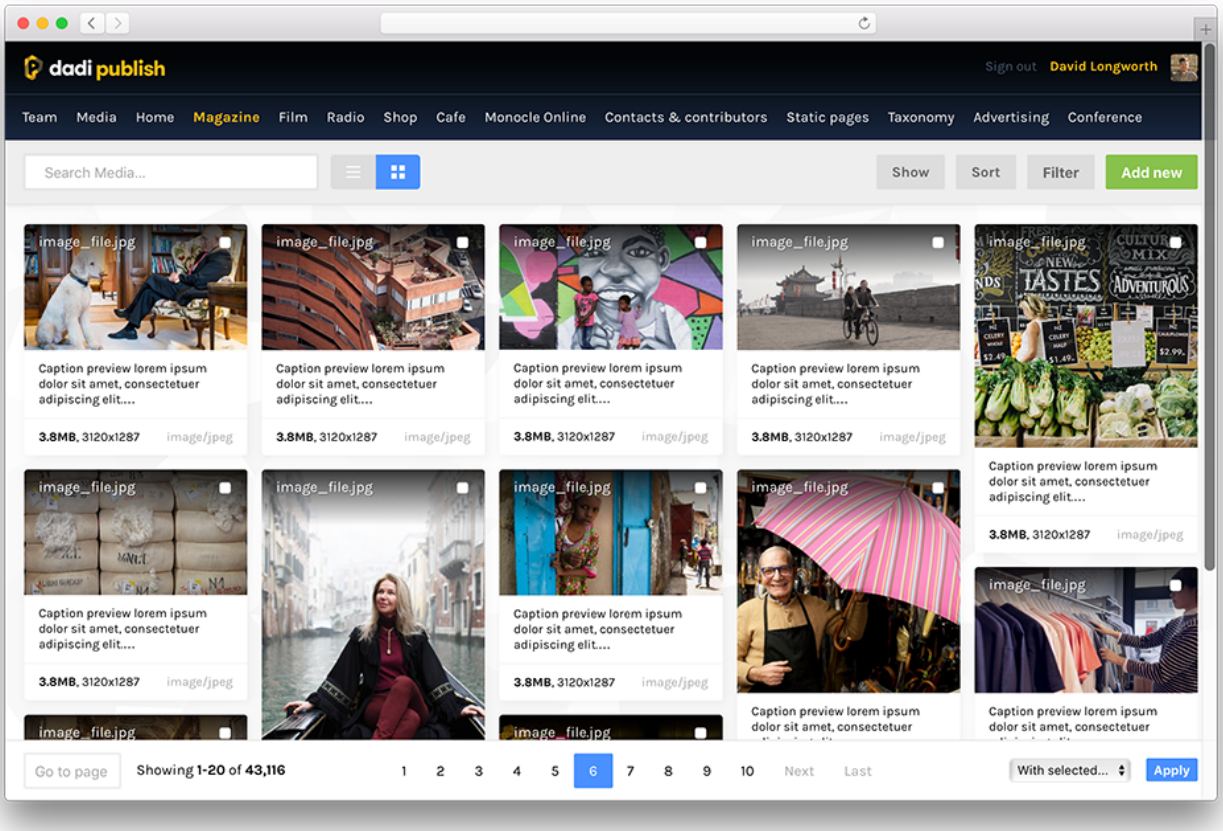
Find out more: https://dadi.tech/en/api/

View source on Github
Install with NPM
Documentation

3.4.2. Publish

dadi publish

A writer's window to the world of content creation. Flexible interfaces designed to optimize editorial workflow.



DADI Publish is developed to optimize editorial workflow with flexible interfaces that adapt to your requirements, meaning more time can be spent making content than wasting time managing it. It is built to support the principles of COPE (Create Once Publish Everywhere).

Journalists and editors can seamlessly manage output across multiple products from one place, while engineers can make light work of even complex content collections. Publish currently works with DADI API. Connectivity with other data stores will follow.

Find out more: https://dadi.tech/en/publish/

View source on Github

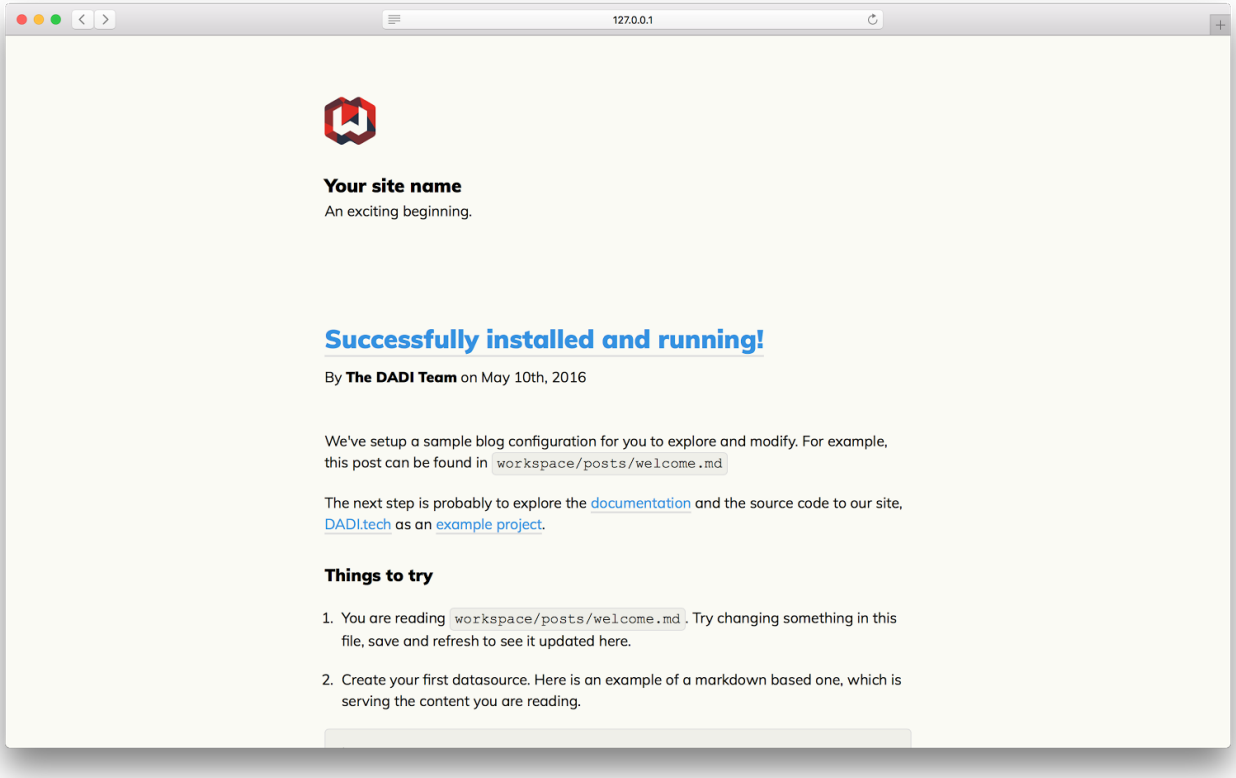
Documentation & NPM release coming soon

3.4.3. Web

dadi web

A schemaless templating layer that can operate as a standalone platform or with API as a full stack web application.

DADI Web makes it easy to build custom enterprise-grade web applications. Easily create static pages or connect to APIs to generate data-driven apps giving you the power to search, paginate, sort and filter your data, on the server or in-browser.



DADI Web supports a variety of templating languages, providing a simple yet powerful template layer for displaying your data. These include LinkedIn's Dust.js, Pug.js or Handlebars – or you can easily craft your own.

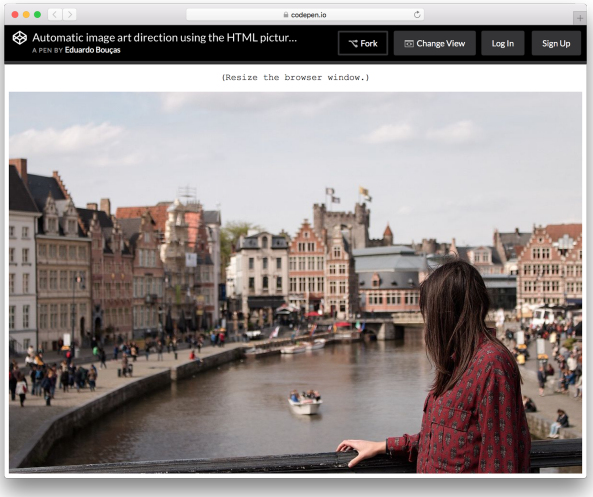
Find out more: https://dadi.tech/en/web/

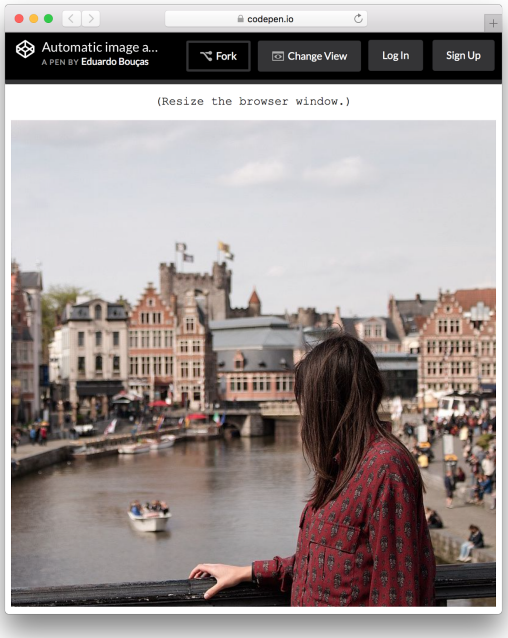
View source on Github
Install with NPM
Documentation

3.4.4. CDN

dadi cdn

A just-in-time asset manipulation and delivery layer designed as a modern content distribution solution.





CDN can dynamically apply art direction to images for responsive layouts. See https://codepen.io/eduardoboucas/full/ZOApOK for more info.

DADI CDN is designed to carry the processing and delivery load associated with image manipulation and asset delivery (CSS/JS/fonts/audio/video).

CDN has support for caching, header control, image manipulation, image compression, image format conversion and advanced manipulation techniques such as entropy analysis. It has an authenticated API for fined grained cache control and provides a JSON endpoint for enhanced file data, including colour and focal point analysis.

Find out more: https://dadi.tech/en/cdn/

[Test variables in the CDN sandbox]
[View source on Github]
[Install with NPM]
[Documentation]

3.4.5. Store

A cloud storage solution for all types of data, with built-in security, privacy and redundancy.

DADI Store is designed to store and retrieve data files - such as images, PDFs and Word documents - in a highly efficient, distributed manner within a network.

It is highly scalable, distributing data among available networked nodes to provide a high level of privacy and redundancy.

3.4.6. Queue

A lightweight queue processing system powered by Redis, featuring simple task routing and throttling.

DADI Queue can be used for any tasks or functions in a digital product that are critical and potentially high-volume.

It ensures that messages are received and marked for processing, and that they can be processed only once. It is capable of delivering messages to workers on a defined schedule, or throttling workers on a messages-per-duration basis.

View source on Github
Install with NPM
Documentation coming soon

3.4.7. Identity

Guarantees uniqueness of individuals – and powers segmentation – for anonymous and known users.

DADI Identity collects and stores information on both anonymous and known users to drive a dynamic CRM strategy well beyond basic email segmentation.

It powers personalization and can be used with DADI Track to allow for data-driven experiences in digital products – presenting content and commercial opportunities relevant to the individual, rather than one-size-fits-all.

3.4.8. Track

A real-time, streaming data layer providing accurate metrics at individual and product level.

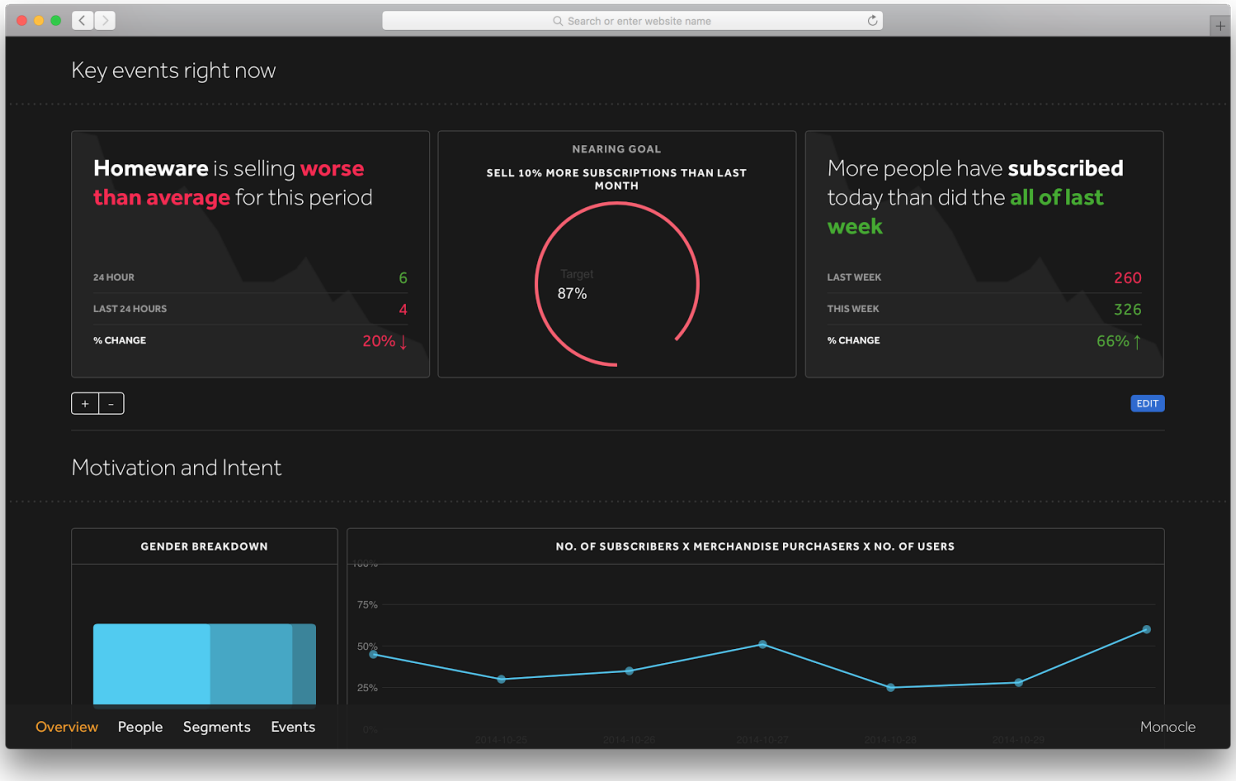
DADI Track is an enterprise-strength web, mobile and event analytics platform. It enables you to build audience segments using data collected across all platforms and channels which can then be used to personalize your communication and content.

Practical applications include enriching user records stored in DADI Identity, piping real-time usage stats into DADI Visualize, or feeding an existing data warehouse.

3.4.9. Visualize

dadi visualize

A data visualization interface for Identity and Track, capable of taking data feeds from virtually any source.



DADI Visualize is a flexible set of interfaces that allow you to rapidly configure dashboards to display real-time data and analytics from your digital product.

It has the ability to mix data on content and users, allowing you to drill down all the way to the actions of an individual – and can be used to measure commercial goals and KPIs.

3.4.10. Match

A taxonomic framework for automated content classification through machine learning which plugs into Publish.

DADI Match draws on an exclusive and highly accurate taxonomic framework - developed by DADI in conjunction with Canopy Insight - to automatically tag content fragments with relevant key words.

It can be used to retrospectively classify content, make recommendations for editors at the point of content creation and drive content recommendations.

3.4.11. Predict

A machine learning layer that predicts user behaviour at an individual level based on past interactions.

DADI Predict has the power to identify like audience types based on (for example) demographic information and serve content or commercial opportunity at the right moment to maximize audience retention or propensity to convert.

It works in real time to segment customers during a single session and predict preference based on the actions of similar customers that have previously completed successful user journeys. Practical applications can include presentation of related content, display advertising or retail products.

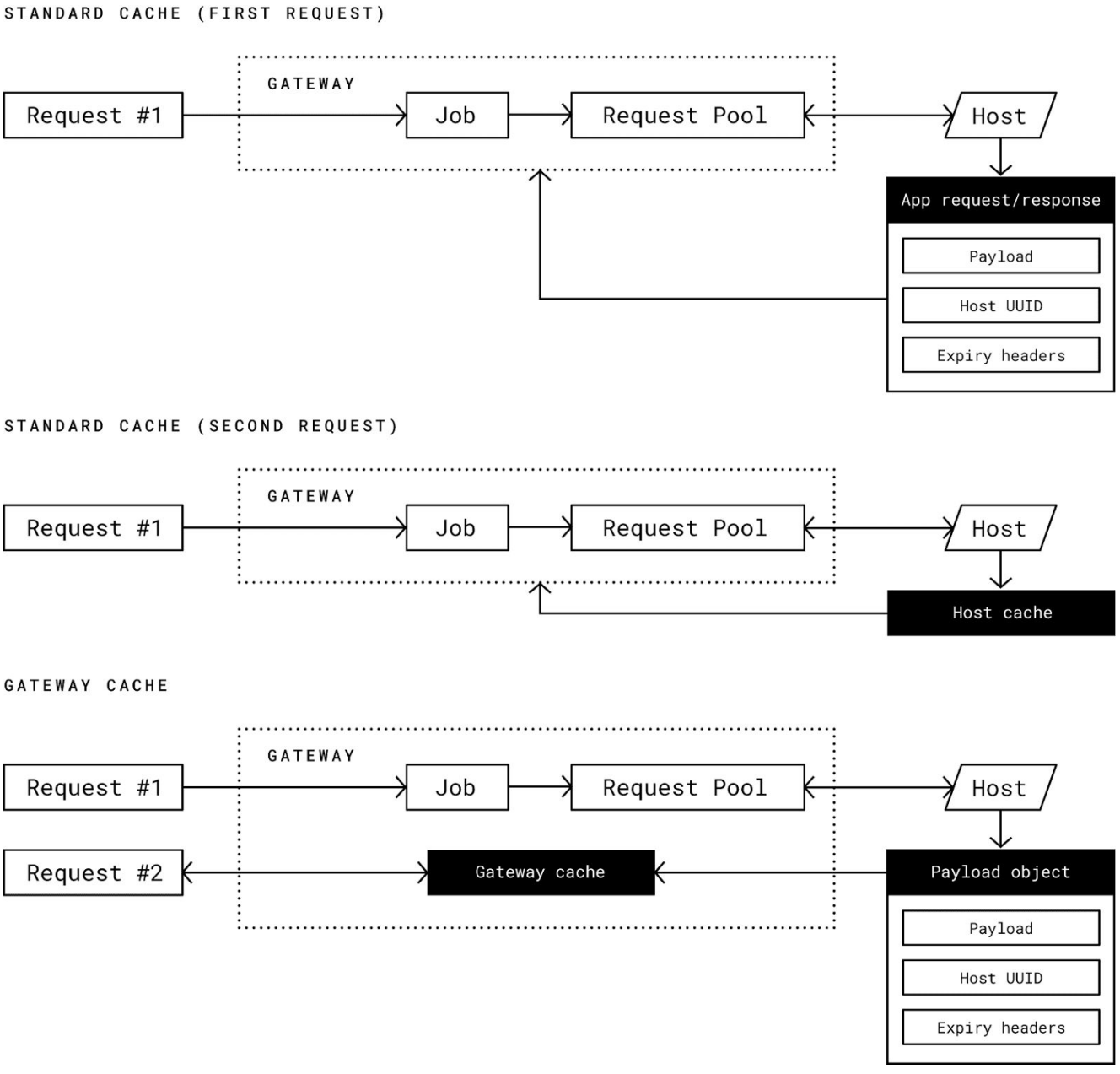
3.4.12. DADI Web Services documentation

Documentation for existing DADI web services can be found at https://docs.dadi.tech/.

3.5. Caching

Caching is built into DADI at both Consumer app (Host) and Gateway level. Host level caching is facilitated through individual DADI Web Services, which share a caching module that supports both Redis and filesystem caching. Gateway level caching uses the cache headers returned in the Host payload response to determine whether to cache, and for how long. Gateway caching carries a premium in service pricing reflective of the value of the role that they play in the network.

On Git: https://github.com/dadi/cache



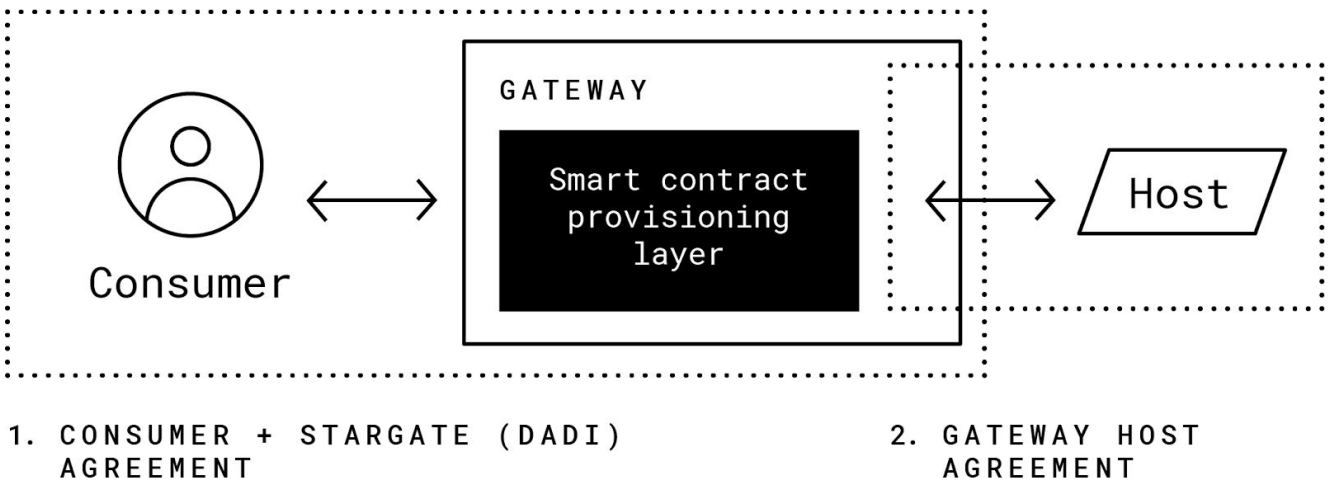
3.6. The smart contract system

3.6.1. Web services contracts

DADI enables the formation of web services contracts between peers. Contracts are agreements between providers of computational power and their clients, defining which services will be provided, at what scale and at what price.

They require all network players to prove that they are delivering their contracted services at regular intervals. Contracts are stored on the blockchain, making them publicly auditable.

Contracts with Consumers are handled in Stargates, which establish forward contracts with Gateways. Contracts with Hosts are handled between the Gateway VPC and individual Hosts automatically, at the point that a Host authenticates with a Gateway.



This approach is called *collective negotiation*, and it allows for ad hoc clusters of computational power to be formed and maintained extremely quickly – essential given the nature of the services being provided.

3.6.2. Simple contract flow

Stargates control the flow of tokens in exchange for services, receiving payments from Consumers and establishing forward contracts with Gateways, which in turn establish forward contracts with Hosts.

Stargates, Gateways and Hosts have their own wallets — simple contracts containing a proof of stake. This goes a long way to ensuring the commitment of the contributor to the network. The proof of stake can also be increased by the Miner, adding to their earning potential.

3.6.3. Whitelisting

Stargates host a network manifest, storing realtime attributes of all connected Gateways and Hosts. The manifest regularly updates each host record with performance and trust analytics. After the initial onboarding phase into the network, Gateways and Hosts are regularly placed on a peer review process by more trusted nodes, which guarantee dequeued requests receive the correct response from the hosted application. Any detected inconsistencies with the response body are stored against the nodes manifest record, and quickly lead to blacklisting.

3.6.4. Contract logic

Once enrolled in the network, each node takes on one of the following states:

1. Initialized
2. Verified
3. Blacklisted

An initialized host is created when the contract first receives a wallet address. Options for real time payments will be explored throughout the first phase of the testnet build. The initial payout mechanism will allow hosts to define a payout period in either time, or tokens.

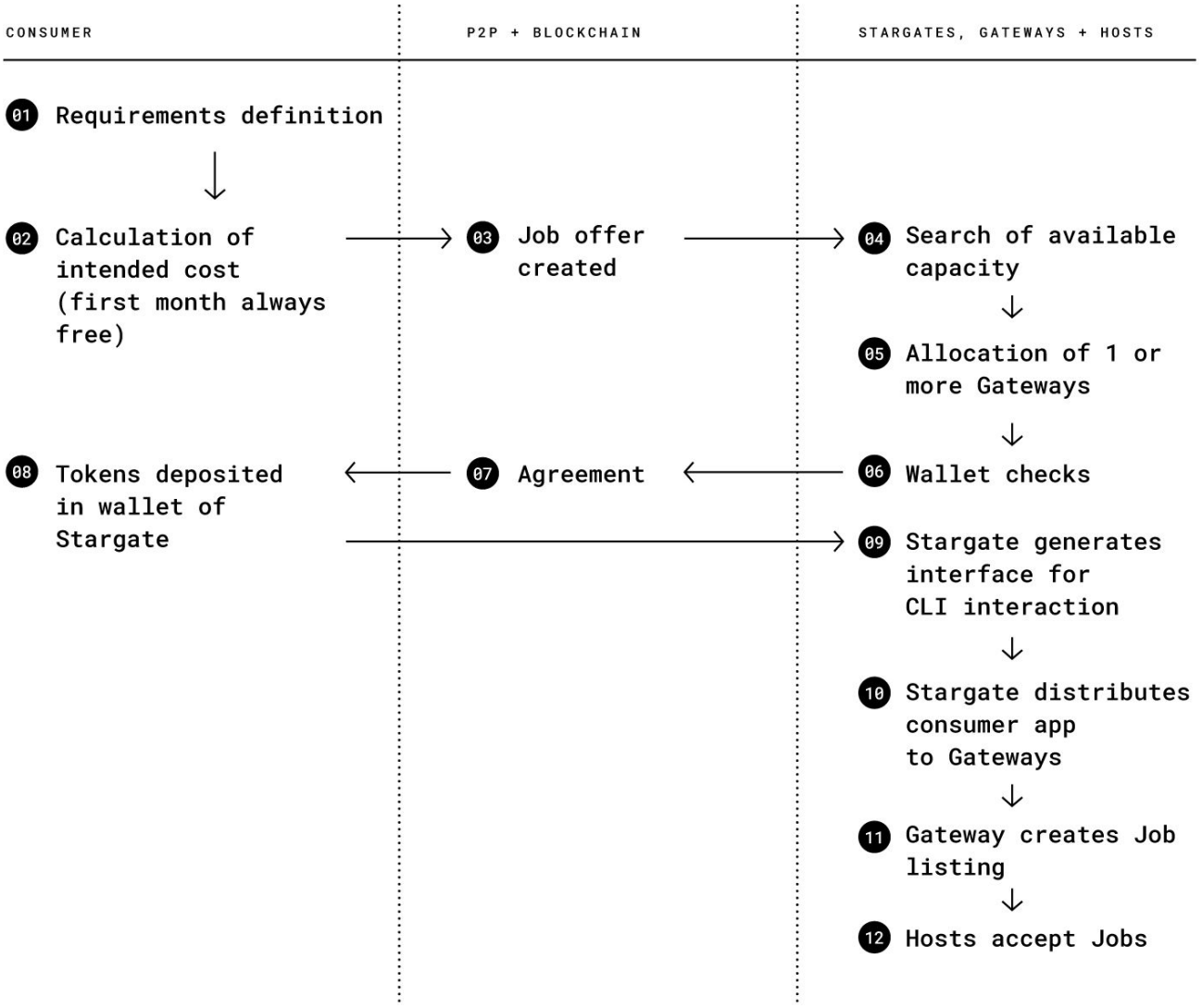
The process of Stargate, Gateway or Host verification consists of a peer review of hardware and network capabilities, requiring peers within the network to perform a system analysis, before updating the contract. These peer reviews are performed by idle and trusted peers, and the subject of review pays for this process. Payments can only be processed after verification is successful.

Blacklisting occurs when the whitelist logic demonstrates that a Stargate, Gateway or Host is not acting in compliance with the network trust guidelines. In such an instance a Miners POS may be frozen.

3.6.5. Consumer-Gateway interaction

Consumers interact with the Gateway VPC through DADI's interfaces. Interactions are run through Stargates, which manage the calculation of intended cost for services, generate Job offers, allocate resource at Gateway level based on available capacity, perform wallet checks for all parties and enter into an agreement with the Consumer on behalf of the network.

Once an agreement is in place the Stargate provides a CLI interface for Consumer interaction with the network, accepts Consumer app bundles and distributes them to the Gateways attached to the contract.



3.6.6. Stargate-Gateway-Host interaction

The process of cooperation between Stargates, Gateways and Hosts starts with the setup of an Ethereum smart contract in a Stargate, containing the DADI tokens to be used to pay for the processing and delivery of Consumer apps.

The Ethereum address of this smart contract, the address of the issuing Stargate and the Gateway itself are recorded in a smart contract generated and maintained by individual Stargates.

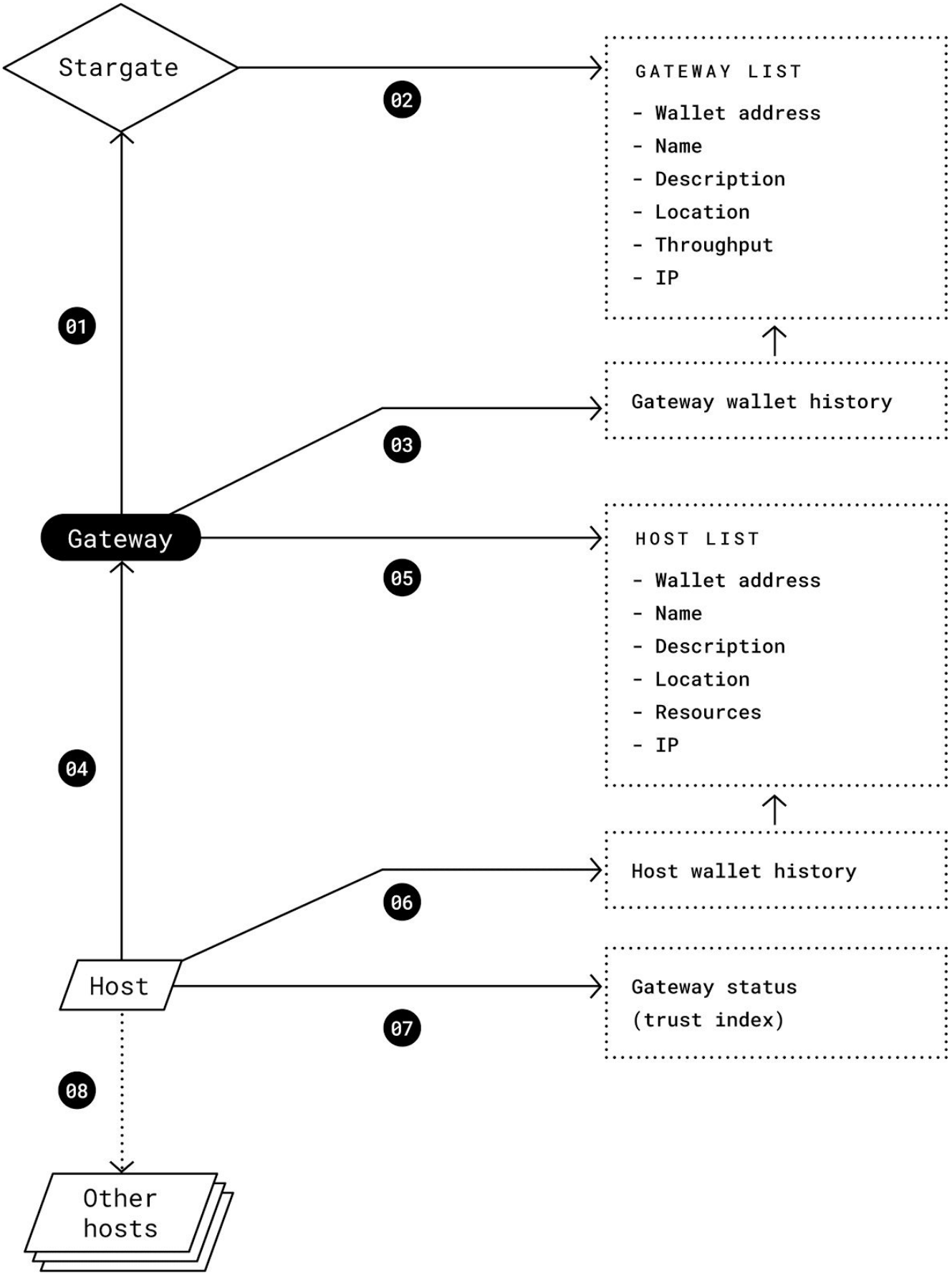
The Gateway List includes both unverified Gateways and verified Gateways. Gateway information within smart contracts include the address of the Gateway owner, the address of the Gateway wallet and the Gateway's IP.

A second smart contract is generated and maintained by individual Gateways. This includes both unverified Hosts and verified Hosts. Host information within DADI smart contracts include the address of the Host owner, the address of the Host wallet and the Host's IP.

The process flow for contract negotiation is as follows:

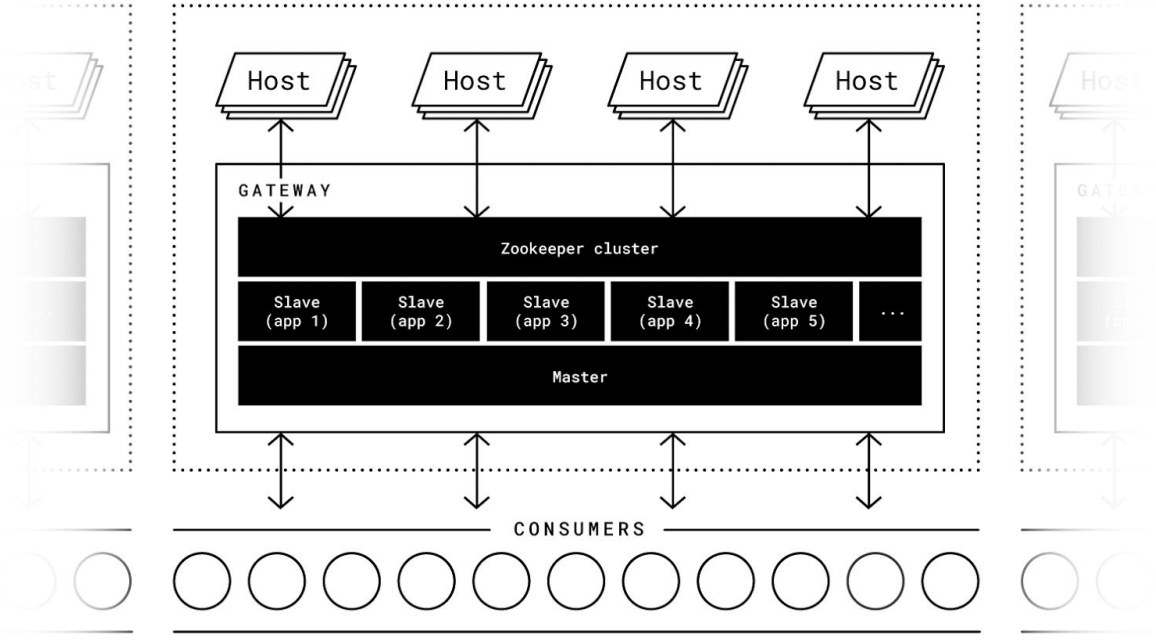
1. Gateways authenticate with a single Stargate
2. The Stargate updates its Gateway List
3. The Gateway maintains its listing and wallet history
4. Hosts authenticate with a single Gateway
5. The Gateway updates its Host List
6. The Host maintains its listing and wallet history
7. The Host monitors the status of the Gateway in a trust index
8. The Host shares Gateway status information with other Hosts

In addition to this public lists containing Stargate and Gateway information will be maintained. These are used at the head of the authentication process for Gateways>Stargates and for Hosts>Gateways.


Stargate
02
GATEWAY LIST
- Wallet address
- Name
- Description
- Location
- Throughput
- IP
01
Gateway wallet history
03
Gateway
05
HOST LIST
- Wallet address
- Name
- Description
- Location
- Resources
- IP
04
Host wallet history
06
Host
Gateway status
(trust index)
07
08
Other
hosts

3.7. Messaging service

To keep a consistent list of connected Gateways and Hosts, the application layer utilizes Apache Zookeeper to diminish the possibility of race conditions and deadlock errors, and to leverage its capabilities as an ephemeral cluster of pub/sub message brokers.



3.8. Security

3.8.1. For Consumers: results verification

Guaranteeing resources are as expected is a key part of the distributed delivery mechanism.

Detection of poor behavior is achieved with a 'spot-check' challenge which, at ad hoc intervals, mirrors queued *Jobs* into the *Line Manager* queue, appending a hash of the payload.

Line Manager jobs are picked up by Hosts that were outperformed on the original job. Line Manager Jobs include the execution timestamp headers that were returned with the original payload so as to ensure that datetime-specific values are rendered in the same way.

Rather than returning the entire payload, the Host handling the Line Manager Job returns a hash of the response to reduce bandwidth.

Under normal conditions, Hosts are informed when a job they are processing has been resolved by another Host. When a Job is selected for spot-check and other Hosts have attempted to process the same job, the Host used to process the spot-check will be selected from this list to leverage the already part-processed response and reduce the requirement to involve other nodes.

3.8.1.1 Process flow

1. A request hits the Gateway
2. The Gateway adds the request to the Job request pool
3. The Gateway duplicates the job into the Line Manager pool
4. Host “A” completes the pending job and returns the payload to the Gateway
5. The Gateway hashes the payload and stores it in the corresponding entry in the Line Manager pool
6. Host “B” picks up the Line Manager job
7. Host “B” sends a hash of the payload to the Gateway
8. The Gateway compares the hash against the existing hash on file
9. If the hash does not match, the poor behaviour flag against Host “A” is incremented

3.8.2. Gateway>Host communication security

When a Host informs the Gateway that it intends to process a Job, a set of security measures are invoked:

1. The Host requests a Job, passing a JSON web token (JWT) to the Gateway
2. The Gateway returns a signed authentication request, including permissions
3. The Host verifies the requests authenticity, using permissions to identify resources
4. The Host creates payload and signs with challenge
5. The Host sends the Gateway the original authentication request along with the Host-produced signed authentication response
6. The Gateway verifies the validity of both request and response

3.8.2.1. Sample auth request


```
{
  "header": {
    "typ": "JWT",
    "alg": "ES256"
  },
  "payload": {
    "issuedAt":"1440624435.28",
    "challenge":"8befe9e5-db3a-408a-aaae-c41c1c8eee55",
    "permissions":["blockchainid"],
    "issuer": {
      "publicKey":"0231e4873b5569c5811...",
      "username":"host_user"
    }
  },
  "signature": "MEUCIQDzUaSrgTR_tTpNSVci..."
}
```

3.8.2.2. Sample response


```
{
  "header": {
    "typ": "JWT",
    "alg": "ES256"
  },
  "payload": {
    "issuer": {
      "publicKey": "027d28f9951ce46538951e36...",
      "username": "host_user"
    },
    "issuedAt": 1444258935251,
    "permissions": [
      {
        "action": "sign",
        "data": "0b42722b-e781-434a-805d-c09c476e86b9"
      },
      {
        "action": "disclose",
        "method": "GET",
        "resource": "/app/resource"
      }
    ]
  },
  "signature": "4sMvmUQ6q5DuAEXYaVIwVSe1nzd4Kj..."
}
```


3.8.3. Protecting Hosts from hostile workloads

Please refer to section 2.3.3 Hosts for more information regarding container isolation in the DADI network and for more detail documentation relating to the attack surface for containers, please see Docker security.

3.8.4. Managing reputation

The reputation of Hosts is largely a measure of performance against contract expectations and against the performance of other Hosts fulfilling the same contract. A poorly written or erroring Consumer app would impact all of the Hosts responding to the Job, meaning no reputational damage would result from attempting to execute the erroneous package.

In such a scenario it is the reputation of the Consumer that would be impacted.

3.8.5 Encryption

Docker recognise the need for encryption and have spoken about it extensively. They have developed solutions that we will be making use of within the DADI node applications.

"A common request that we've heard from the Docker community is the need to have strong cryptographic guarantees over what code and what versions of software are being run in your infrastructure. This is an absolute necessity for secure and auditable production deployments."

DADI makes use of Docker Content Trust, a system which integrates The Update Framework (TUF) into Docker using Notary, an open source tool that provides trust over any content. More information can be found here.

3.8.6 A note on SSL

DADI includes automatic and free SSL certificate generation at the point of contract agreement, generating certificates for attached Consumer domains using Let's Encrypt within the negotiating Stargate.

On Git: https://github.com/dadi/ssl

3.9. Command Line Interface (CLI)

DADI CLI is a command-line tool to help with the installation and customization of the various products of the DADI platform.

The beta version of CLI is available now, providing guided setup for DADI API, DADI Web and DADI CDN. Additional services are being added ahead of the first full release.

Example: installing web using DADI CLI

The development roadmap for CLI focuses on tight integration with the decentralized network, specifically at enabling a frictionless experience for service use to Consumers as well as supporting DADI Hosts and DADI Gateway providers.

Git: https://github.com/dadi/cli

3.10. Deployment tools

Currently, DADI CLI downloads and installs the various DADI applications locally (to the Consumer's machine or server). This behaviour will be kept and referred to as *unauthenticated mode*.

An *authenticated mode* will be introduced, allowing users to associate their CLI session to their DADI account. To authenticate, users simply indicate the email address associated with their account, where they will receive a security code and a link:

```
$ dadi authenticate eb@dadi.co
> We sent an email to 'eb@dadi.co'. Ensure the security code at the top
matches 1763819 and click on the link provided to authenticate.
```

Once they click the link, CLI will automatically become associated with their account and go into authenticated mode.

```
> Thanks, Eduardo! You're now authenticated.
```

3.10.1. Deploying apps to the network

When in *authenticated mode*, users can easily deploy an app to the network, either at the point of creation or at a later stage.

```
# Creates a new app and pushes it to the network
$ dadi api my-new-api --deploy

# Pushes an existing app to the network
$ cd my-new-cdn && dadi deploy
```

In both cases, the user will see a message with a temporary live URL pointing to their app, generated by concatenating names from a dictionary with a random number.

```
> Your app has been deployed. See it live at
https://charles-darwin-39.dadi.cloud.
```

This can then be changed to a custom domain using the CLI.

```
$ dadi domain:set cdn.somedomain.tech
> Your app is now set to use the domain ‘cdn.somedomain.tech’.
```

Alternatively this can be done through the control panel.

3.10.2. Continuous integration

3.10.2.1. Association with version control repositories

Consumers can associate a version control repository (on GitHub, GitLab or Bitbucket) with an app to enable continuous integration workflows.

When running the `deploy` command, CLI checks for the presence of a `.git` directory and, if found, automatically associates the app with the repository defined in the remote.

Alternatively, the association can be made manually.

```
$ dadi config:ci link https://github.com/johndoe/cdn.git
```

3.10.2.2. Bundles and environments

When an app is pushed to the DADI Network all of its dependencies are resolved and downloaded, resulting in a bundle. This bundle can then be promoted across several environments – defined by the user – all the way to production.

The bundle is immutable – meaning that all of its dependencies will never be recalculated – ensuring that the build that was tested and approved is the exact same one that will be pushed to the live environment.

Each environment can have its own set of environment variables, allowing apps to adopt different configurations based on which environment they are currently running under (e.g. API can use different databases as it transitions between environments).

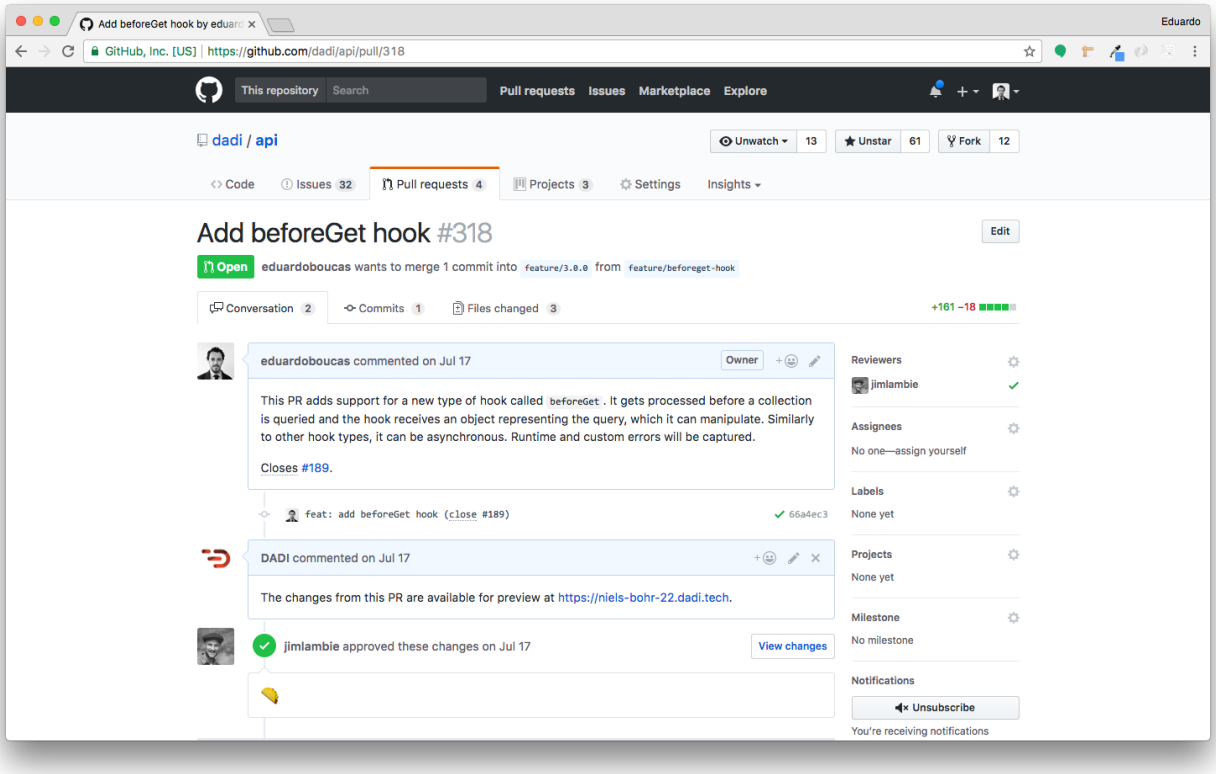
Environment variables can be defined in the control panel or using the CLI.

```
$ dadi config:variables set DATABASE_HOST
      my-new-db-host.somedomain.tech --env=stage
```

Creating or updating environment variables automatically reboots the app so it can inherit the new values.

3.10.2.3. Automatic deployments

When apps are associated with a repository, any proposed changes to the codebase can be automatically pushed to a live URL for preview/QA. This works by creating a bundle of the app with the new state and deploying it to a node in the network.



If the pull request is closed, it is removed from the node. If the pull request is merged, then the bundled will be promoted to the next environment. Based on the structure defined for each organization, this could be a staging or QA environment, UAT, or even to production.

3.11. Front end interfaces

DADI will be primarily available under https://dadi.tech, which will be retooled as the entrance point for the network and it's technologies, supporting both Consumers and those who want to contribute their resources to the network as Miners (Gateways and Hosts). There will also be a supporting native application, available for all major operating systems.

Early design work for this interface set can be found later in this document, in the User Experience & Design section.

3.11.1. DADI's interfaces, from the viewpoint of a Consumer

DADI presents as a standard cloud services interface, introducing DADI Web Services and providing direct routes into platform use. Individual or groups of services can be deployed, with an indication of cost provided on the basis of usage and preference. Consumers will be able to specify their deployment type, setting preferences such as region, redundancy and power.

The ability to pay for resources in fiat - all major currencies - will be supported through the seamless inclusion of an exchange within the interface set. This is seen as critical for the platform to be able to gain critical mass.

3.11.2. DADI's interfaces, from the viewpoint of a Miner

Miners will be able to monitor the performance of their Gateways and Hosts through an interface that aggregates activity (active contracts, usage), and reports on earnings (confirmed, projected). The DADI Gateway and Host apps will be available for direct download.

3.11.2.1. Gateway application

Available via the command line or as a desktop application for OSX, Linux and Windows, the Gateway app provides an interface for attaching the Gateway to a wallet, naming the Gateway and configuring the parameters within which the Gateway will operate (stake size, price per TB of bandwidth, host capacity, bandwidth limits etc.). The application will guide the user through the setup process to the point of announcement on the network (the point at which the node is open to accepting contracts).

3.11.2.2. Host application

Available via the command line or as a desktop application for OSX, Linux and Windows, the Host app provides an interface for attaching the Host to a wallet, naming the Host and configuring the parameters within which the Host will operate (stake size, power utilization, bandwidth limits etc.). The application will guide the user through the setup process to the point of announcement on the network (the point at which the node is available for syncing with a Gateway).

3.11.3. The DADI Marketplace

A marketplace for the buying and selling of both pre-configured and packaged products will be built as part of the front end interfaces to the DADI Network. Marketplace will be open to businesses and individual developers to distribute and monetize products created using DADI Web Services. Products will be categorized within Marketplace and made available to Consumers for one-click deployment, simplifying SaaS procurement.

Examples of products within Marketplace include:

- Pre-packaged/themed websites (ecommerce, content sites, blogs)
- Single Sign On (SSO) for web and mobile applications
- Developer tools (issue & bug tracking, monitoring, testing etc.)
- Bespoke tracking tools and analytics interfaces
- Custom interfaces and collection types for DADI Publish
- Data Providers for DADI Web (to load data from third parties, such as GitHub, Dropbox, Instagram, etc.)
- Data Connectors for DADI API (couchDB, JSON, Redis etc.)
- Templating Engines for DADI Web

3.12. DADI GitHub repositories

DADI can be found at https://github.com/dadi. Key repositories:

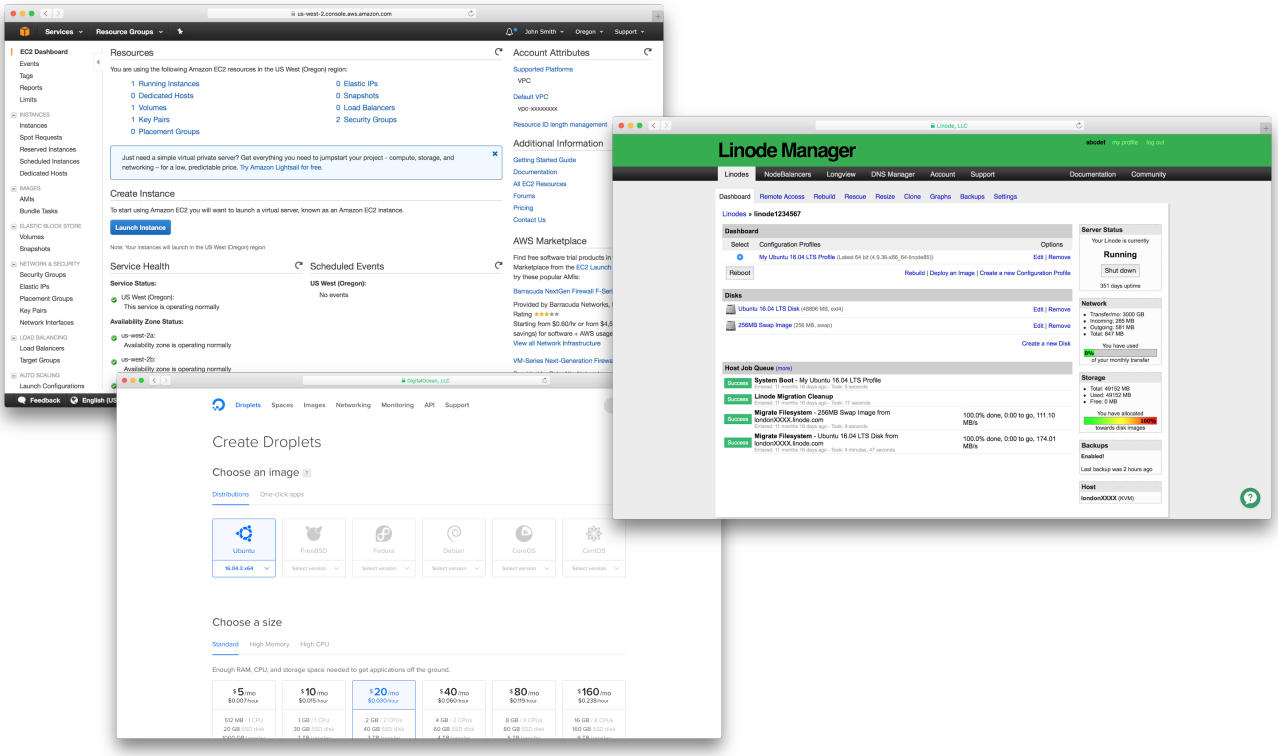
- https://github.com/dadi/api
- https://github.com/dadi/publish
- https://github.com/dadi/web
- https://github.com/dadi/cdn
- https://github.com/dadi/queue
- https://github.com/dadi/cli

4. User experience & design

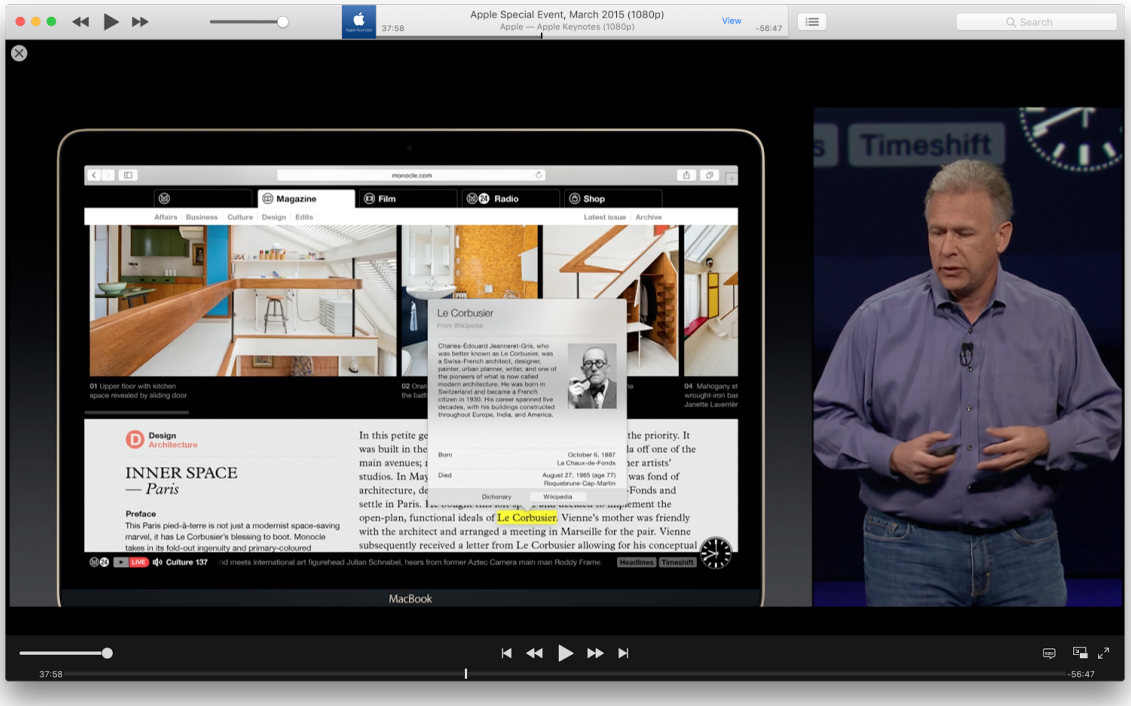
DADI firmly believes that user experience is key to the success of technology. Design is – and has always been – as important to DADI as the technology itself.

DADI is designed to be accessible, simple and easy-to-use. The concepts may be complex, but the interfaces are not – the DADI team follows a fully tested, iterative approach to UX for all templates and the goal is not to over-design. The trick is to tame the complexity and deliver it simply. Because if people can do something effortlessly, they will.

Many comparable services fail in this regard. *Amazon Web Services*, *DigitalOcean*, *Linode* and *Heroku* often overwhelm with the number of configurations and options. It is important to DADI that people of all skill levels can use the technology effectively – from individual developers to enterprise system admins – and not feel as though they are picking at the internet with a pin.



DADI has a rich history of applying design thinking to make its web services more usable – for example DADI CLI helps make complex development tasks trivial – and to solve the problems of clients such as *Monocle*, *Bauer* and *Virgin Limited Edition*, where DADI has assisted with the consumer-facing experience. As a measure of success, DADI was honoured to have its *Monocle* work featured in multiple *Apple* keynotes.



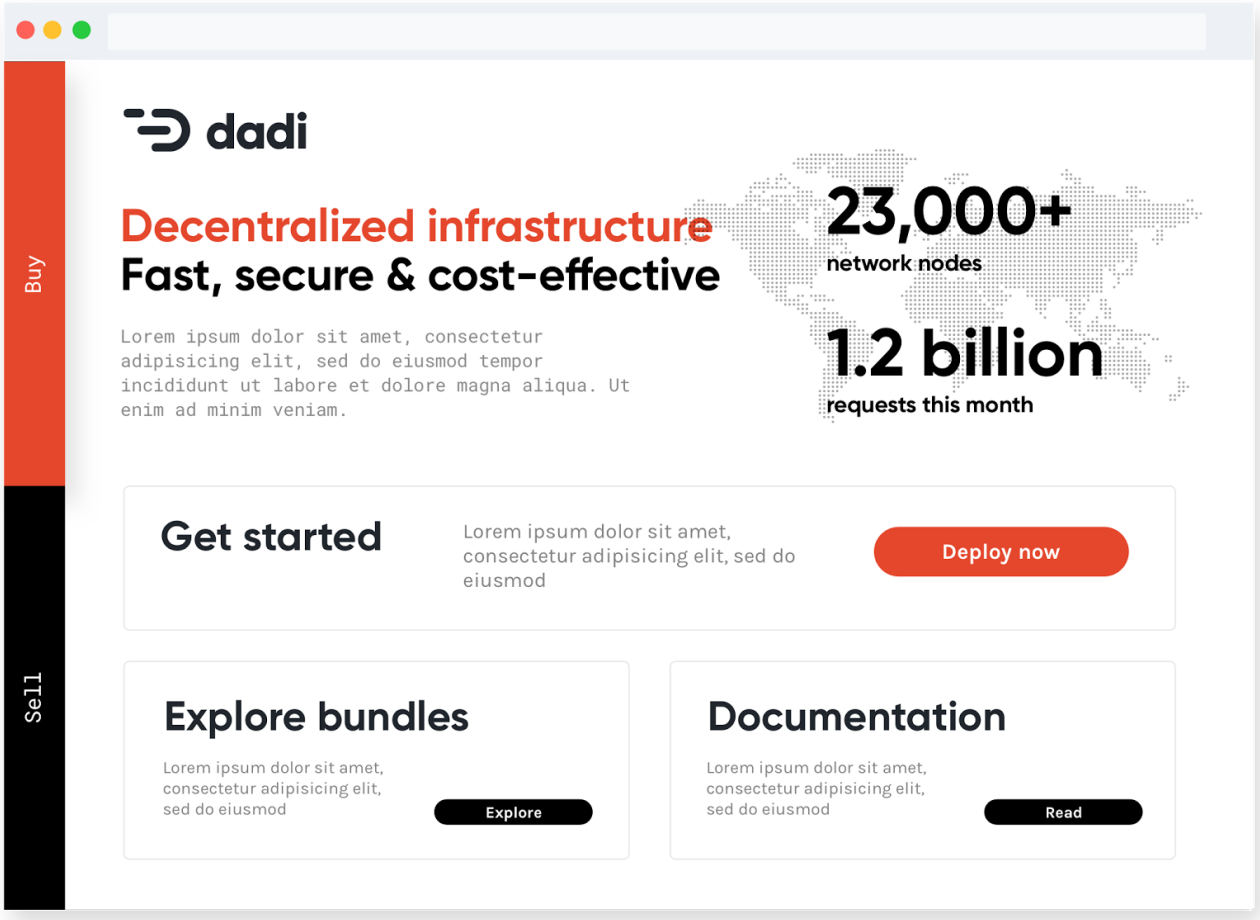
DADI is also taking steps into content management interfaces with DADI Publish, currently in active development.

4.1. Front end interfaces

Early work has begun on the frontend user experience for Consumers and Miners within DADI. These screens should be considered directional. Further work is roadmapped, and development will happen in the open, with code directly available via Github.

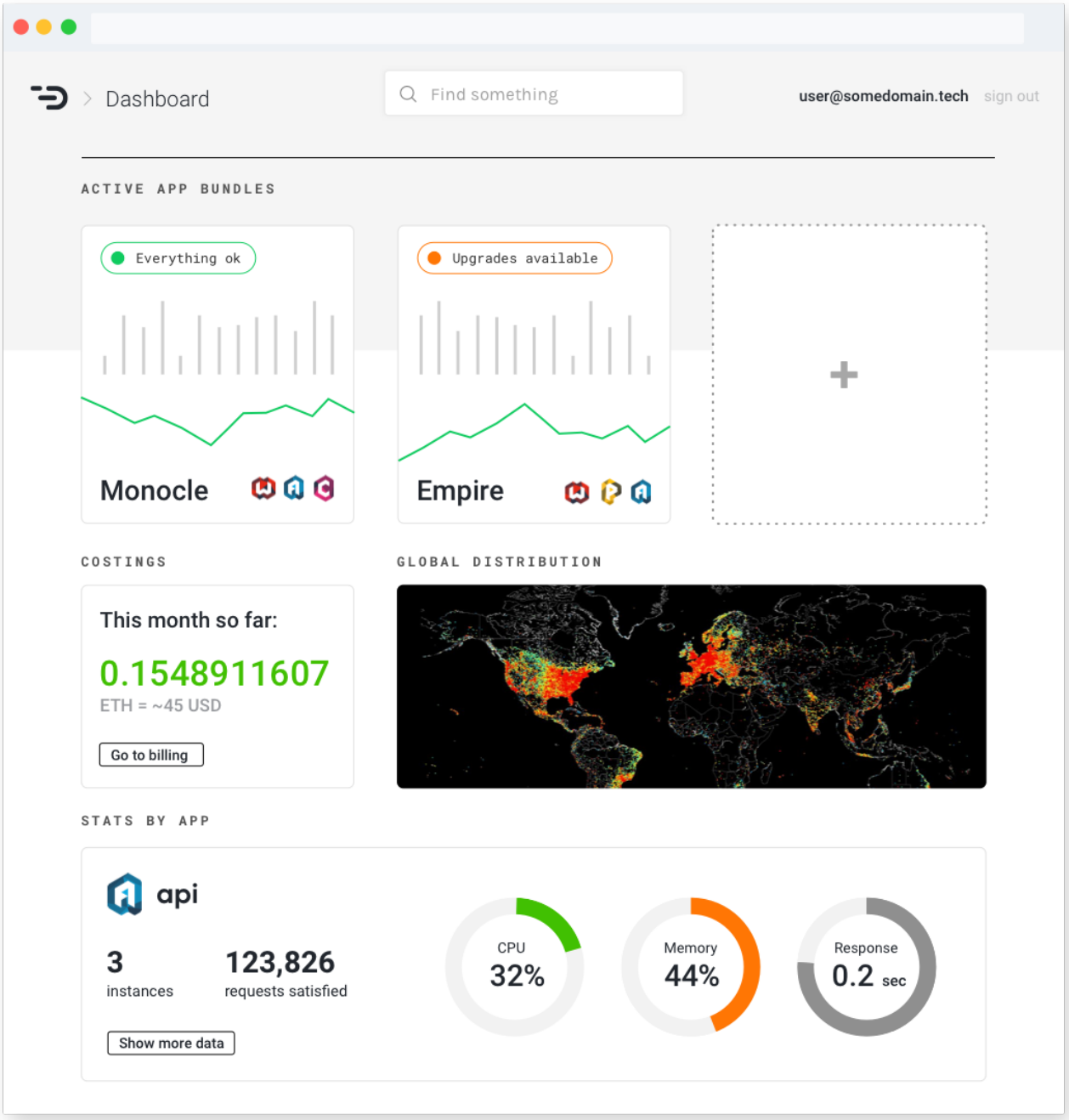
4.1.1. Consumer entry point

Consumer entry points will be focused on lowering the barrier to entry to network use, using clear language and focused calls to action.



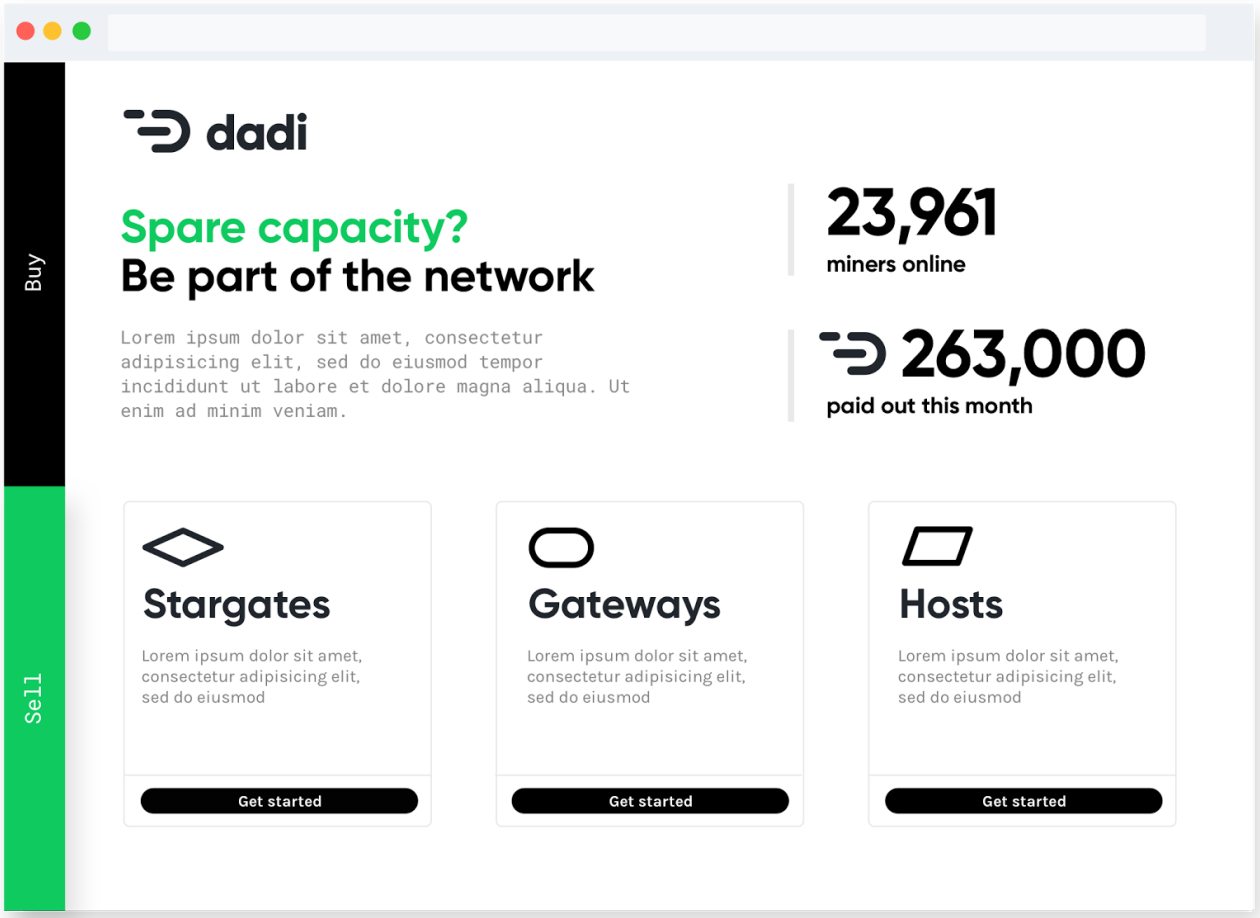
4.1.2. Consumer usage monitoring

Once a consumer is active in the network they will have access to an interface detailing network usage, and providing the tools necessary to enable them to manage their app bundles and effectively control their spend.



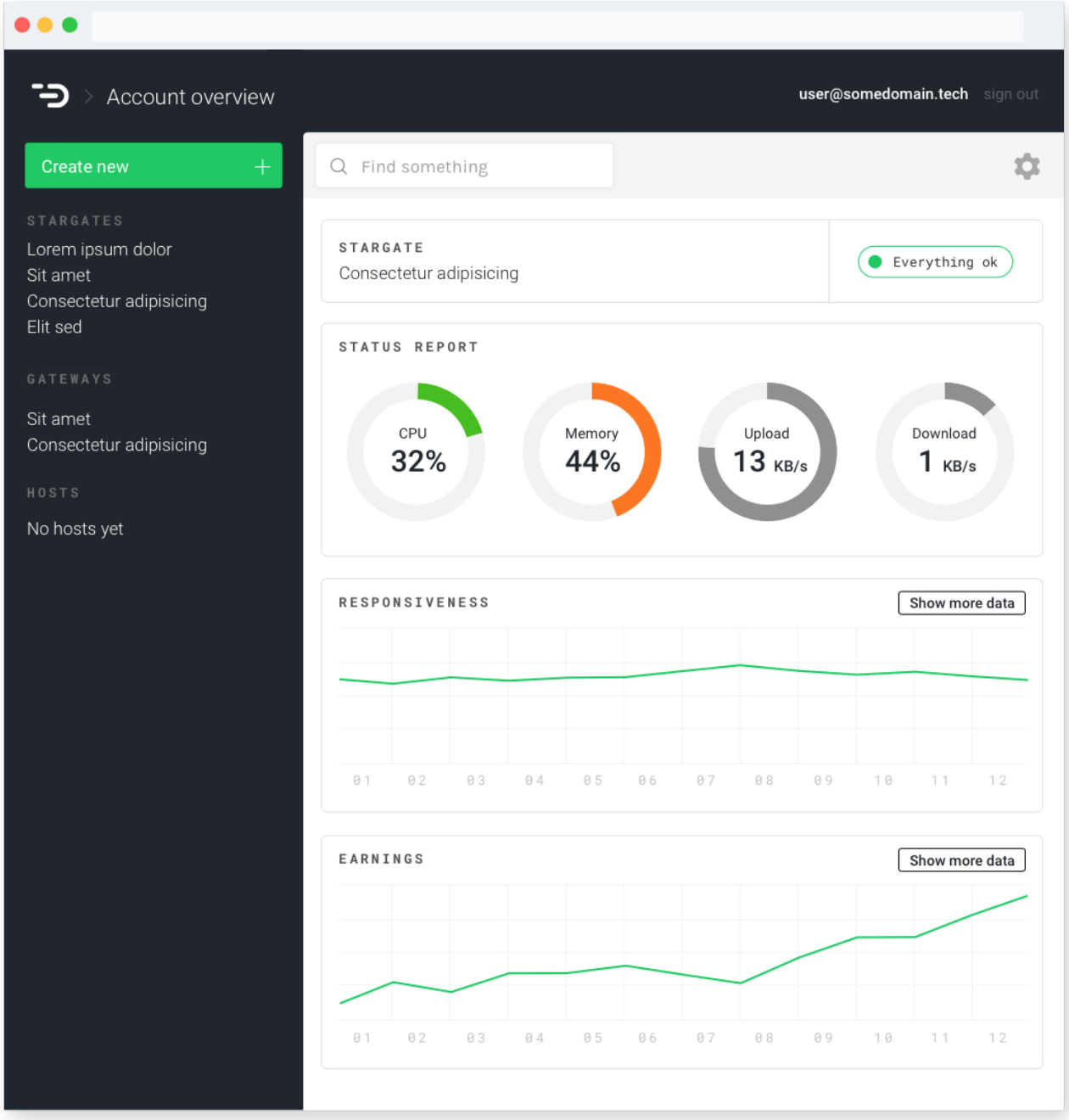
4.1.3. Miner entry point

Miner entry points will be focused on getting those interested in joining the network up and running as quickly as possible, with direct downloads for pre-compiled builds of DADI's Miner software, easy access setup guides and direct links to support.



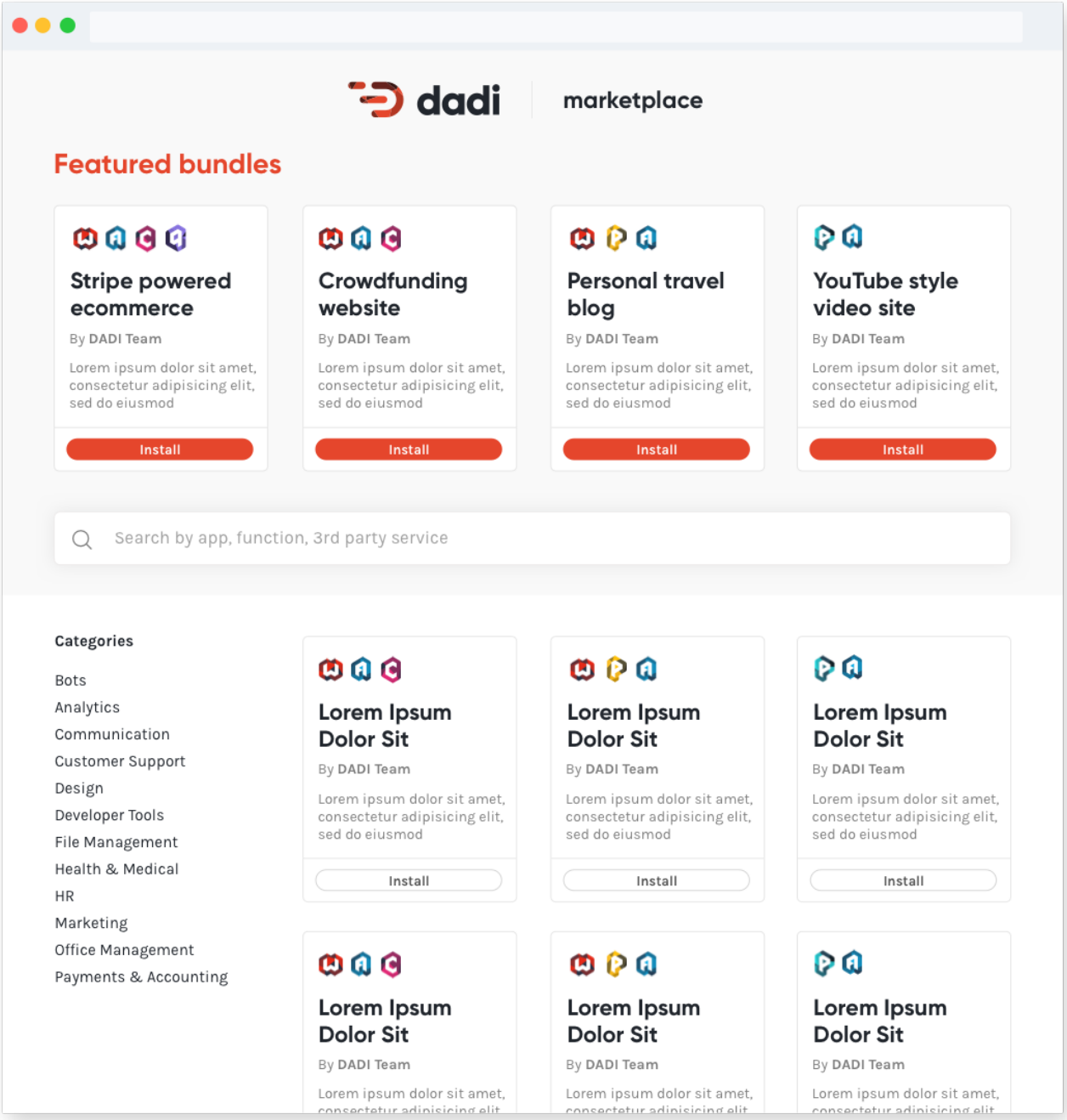
4.1.4. Miner performance monitoring

Miners will have a simple interface that reports usage and performance statistics for their Stargates, Gateways and Hosts. In addition to detailing contract status, Miners will be able to see granular detail of system resource usage as well as their earnings, enabling them to make fully informed business decisions.



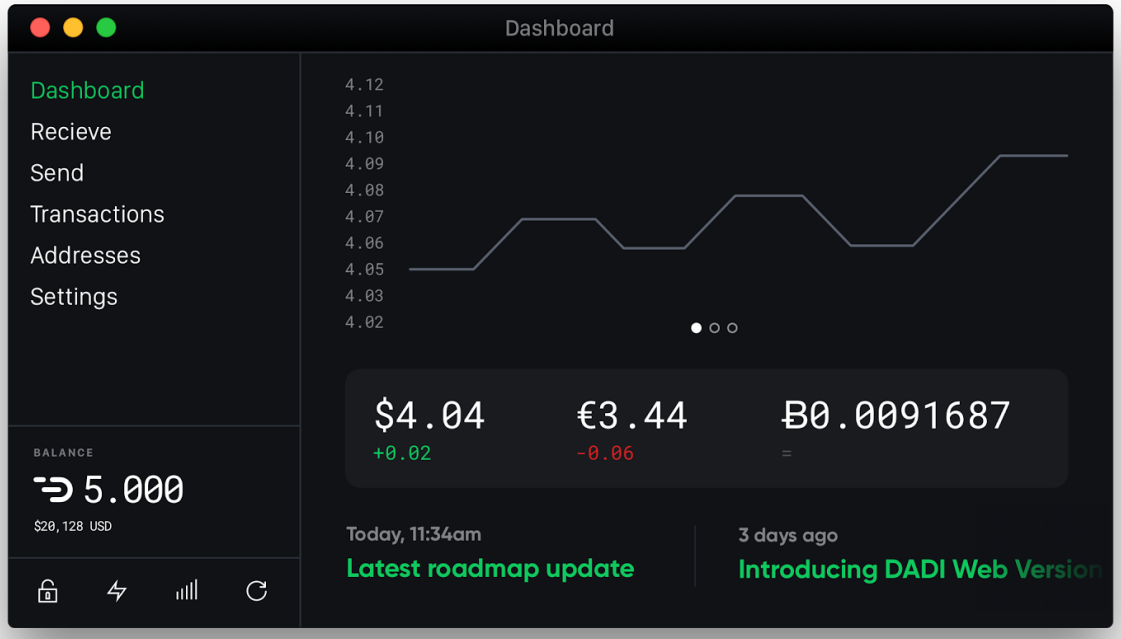
4.1.5. The DADI Marketplace

The DADI Marketplace will be a forum for platform users and developers to share and build common (and obscure!) configurations for the platform – everything from feature plugins to full front-end solutions that can be deployed quickly and with minimal configuration.



4.1.6. The DADI wallet

DADI is developing its own wallet – a secure cross-platform solution for holding and exchanging tokens, monitoring performance and reading updates from the team.



5. Development roadmap

5.1. Current status

Several core DADI Web Services are complete and in production, with others at various stages in the development lifecycle. (Further reading: web services deep dive). DADI Gateway and DADI Host are in active early stage development. Hardware test beds for the first DADI Stargate and the first DADI Host have been installed with Netwise in London Bridge (Netwise House, 24 Old Jamaica Road London SE16 4AW UK). Our testnet is also live and in production testing.

5.2. Milestones

This is a high level roadmap for the twelve months post-Crowdsale. Timings are dependent on the scale of success of the Crowdsale and reflect the DADI team's ambition: the more funds raised, the faster and further the team will be able to travel.

5.2.1. Q4 2017

Release	Description
DADI Publish Beta	Full public release of DADI publishing interfaces: production ready build, complete with baseline test coverage. *More information: https://dadi.tech/en/publish/* *View source on Github* *Documentation & NPM release coming soon*
Hardware test bed	Custom build server-class hardware installed with Netwise in London Bridge as a testbed for DADI Stargate, Gateway and Host, specifically designed to provide benchmark data for latency between platform core layers. The setup will also be used to trial key components in a live context, using a containerized build of DADI CDN.
DADI API milestone release: v3.0.0	Version 3.0 of DADI API will widen the choice of data connectors available and allow engineers to build their own. *More information: https://dadi.tech/en/api/* *View source on Github* *Install with NPM* *Documentation*

5.2.1. Q4 2017 cont.

Release	Description
DADI Web milestone release: v4.0.0	Version 4.0 of DADI Web brings several performance enhancements and a simplified codebase for easier maintenance. *More information:* *https://dadi.tech/en/web/* *View source on Github* *Install with NPM* *Documentation*

5.2.2. Q1 2018

Release	Description
Beta release of DADI Store	Full public release of DADI cloud storage solution: production ready build, complete with baseline test coverage. *More information: Details and links available soon*
DADI CDN milestone release	Significant feature release of DADI CDN, details to be confirmed. *More information:* *https://dadi.tech/en/cdn/* *Test variables in the CDN sandbox* *View source on Github* *Install with NPM* *Documentation*
Publish milestone release	Significant feature release of DADI Publish, details to be confirmed *More information:* *https://dadi.tech/en/publish/* *View source on Github* *Documentation & NPM release coming soon*
Alpha release of Track	First release of DADI's enterprise-strength web, mobile and event analytics platform. Private Alpha for testing and development. *More information: Details and links available soon*
Alpha release of Identity	First release of DADI's core CRM tool that collects and stores data on anonymous and known users. Private Alpha for testing and development. *More information: Details and links available soon*

Release	Description
Alpha release of Gateway (Linux)	First release of DADI Gateway, containing MVP VPC functionality, messaging (Apache Zookeeper) and a custom Job queue. Note: Alpha builds at this level in the platform are designed to test the performance of key components (latency, throughput and resource usage) and are not production ready. *More information: Details and links available soon*
Alpha release of Host (Linux)	First release of DADI Host, containing MVP Docker layer, Gateway integration and support for DADI CDN. *More information: Details and links available soon*
Alpha release of DADI Wallet (Linux, OSX and Windows)	DADI Wallet is a ERC20 capable wallet built specifically in support of DADI. The first release will provide basic support for DADI contract management alongside Ether wallets, with a clear focus on user experience and lower the barrier to entry for participation in virtual currencies. Later builds will include direct fiat conversion and Consumer facing management functionality for DADI Web Services. *More information: Details and links available soon*
Preview of network ready DADI.tech	HTML/CSS/JS interfaces for the evolution of DADI.tech, including account management interfaces, Marketplace functionality and Miner support pages. *More information: Details and links available soon*

5.2.3. Q2 2018

Release	Description
Alpha release of DADI Visualize	First release of DADI's data visualization interface for Identity and Track. Private Alpha for testing and development. *More information: Details and links available soon*
Beta release of DADI Track	Full public release of DADI's web, mobile and event analytics platform: production ready build, complete with baseline test coverage. *More information: Details and links available soon*
DADI API milestone release	Significant feature release of DADI API, details to be confirmed. *More information: Details and links available soon*
DADI Web milestone release	Significant feature release of DADI Web, details to be confirmed *More information: Details and links available soon*
Alpha release of Stargate (Linux)	First full release of DADI Stargate for testing with trusted partners. *More information: Details and links available soon*
Alpha release of Gateway (Windows)	Multi platform support for DADI Gateway. *More information: Details and links available soon*
Alpha release of Host (OSX and Windows)	Multi platform support for DADI Host. *More information: Details and links available soon*
Alpha release of network ready DADI.tech	First full build of DADI.tech in support of the full network. *More information: Details and links available soon*
Alpha release of the DADI wallet	First full stable release of the DADI wallet, integrated with DADI.tech's APIs for service provisioning (note: not fully live until Q4) *More information: Details and links available soon*

5.2.4. Q3 2018

Release	Description
Network goes live	DADI network up and running with “backbone” Stargate, Gateway and Host set in key locations. Network open for testing by early adopters. *More information: Details and links available soon*
Container bundles of DADI Web Services	Containerised builds of Publish, API, Web, CDN and Store, ready for testing in the decentralised network. *More information: Details and links available soon*
Alpha release of DADI Match	First release of DADI’s taxonomic framework for automated content classification through machine learning. Private Alpha for testing and development. *More information: Details and links available soon*
Alpha release of DADI Predict	First release of DADI’s machine learning engine for predictive analysis based on individuals, actions and objects. Private Alpha for testing and development. *More information: Details and links available soon*
Beta release of DADI Visualize	Full public release of DADI’s data visualization interface for Identity and Track: production ready build, complete with baseline test coverage. *More information: Details and links available soon*
Beta release of DADI Identity	Full public release of DADI’s anonymous and known user data store: production ready build, complete with baseline test coverage. *More information: Details and links available soon*
Beta release of Stargate	Beta release for Linux *More information: Details and links available soon*
Beta release of Gateway	Beta release for Linux, OSX and Windows *More information: Details and links available soon*

5.2.4. Q3 2018 cont.

Release	Description
Beta release of Host	Beta release for Linux, OSX and Windows. *More information: Details and links available soon*
CDN milestone release	Significant feature release of DADI CDN, details to be confirmed. *More information: Details and links available soon*
Publish milestone release	Significant feature release of DADI Publish, details to be confirmed. *More information: Details and links available soon*

5.2.5. Q4 2018

Release	Description
Beta release of Match	Full public release of DADI's taxonomic framework for automated content classification through machine learning: production ready build, complete with baseline test coverage. *More information: Details and links available soon*
Beta release of Predict	Full public release of DADI's taxonomic framework for automated content classification through machine learning: production ready build, complete with baseline test coverage. *More information: Details and links available soon*
DADI first full release	**Code name "Constellation"** – the first full public release of DADI. *More information: Details and links available soon*

5.3. Release cycles

The core applications and services in the DADI platform follow a sixty day release cycle, during which a number of new features are worked on along with any fixes required (as determined by the issue tracker in each repository).

Having a schedule for the release of new features ensures that those already using the services in production environments can plan their own deployment schedules around core updates.

5.4. Development standards and contributing guidelines

5.4.1. Coding Standards

To ensure consistency throughout the source code we follow the style rules defined by StandardJS.

5.4.2. Test Coverage

All new features, changes or bug fixes must be covered by one or more tests. We maintain a baseline level of test coverage of 80% for each of the applications, which should increase or stay the same as development continues, never decrease.

5.4.3. Git Commit Guidelines

We follow the commit guidelines from the Angular.js project and require that commit messages are formatted in a particular way. This gives us a more readable project history, and in some case we make use of the commit messages to generate release notes.

5.4.4. Releasable Master Branch

The ”master” branch of each application repository is always in line with the latest published version of the application. Development work is based off a main "develop" branch which contains all features and bug fixes that are ready for release. New features are developed in isolation on "feature" branches which are derived from the main "develop" branch and will eventually be merged back into it.

5.4.5. Contributing guidelines

Contributing guidelines can be found in the route all DADI repositories, for example: https://github.com/dadi/api/blob/master/CONTRIBUTING.md

6. In summary

DADI is tackling a $250-billion-dollar industry head on. It represents a radical overhaul of the cloud computing sector, upholding the founding principles of the Web by democratizing computational power.

DADI's approach focuses on simplicity - taming hugely complex systems to provide the services that underpin digital products in a manner accessible to all. It is a global network of ad hoc clusters comprising of one Gateway and multiple Hosts, organized by a DAO.

DADI technology has been in active R&D for four years and its core Web Services are already live, powering some of the world's most recognizable businesses.

The 18 strong DADI team is two-thirds engineering. The senior team have been working together for 18 years.

DADI's founders have invested c.$2 million in direct R&D to date. The business is financed through cash reinvested by the business and short-term debt.

Development roadmaps for the DADI platform push out 18 months post-Crowdsale, building on a proven track record to deliver and iterate a unique proposition built on blockchain technology.

6.1. Crowdsale contacts

Name	Role	Email
Joseph Denne	*CEO*	jd@dadi.co
Christopher Mair	*Marketing, PR & outreach*	cm@dadi.co
James Lambie	*CTO*	jl@dadi.co

6.2. Engage with us

Questions? Ideas? Just want to chew the fat? Reach out:

- Telegram
- Forum / Reddit
- Discord
- Twitter