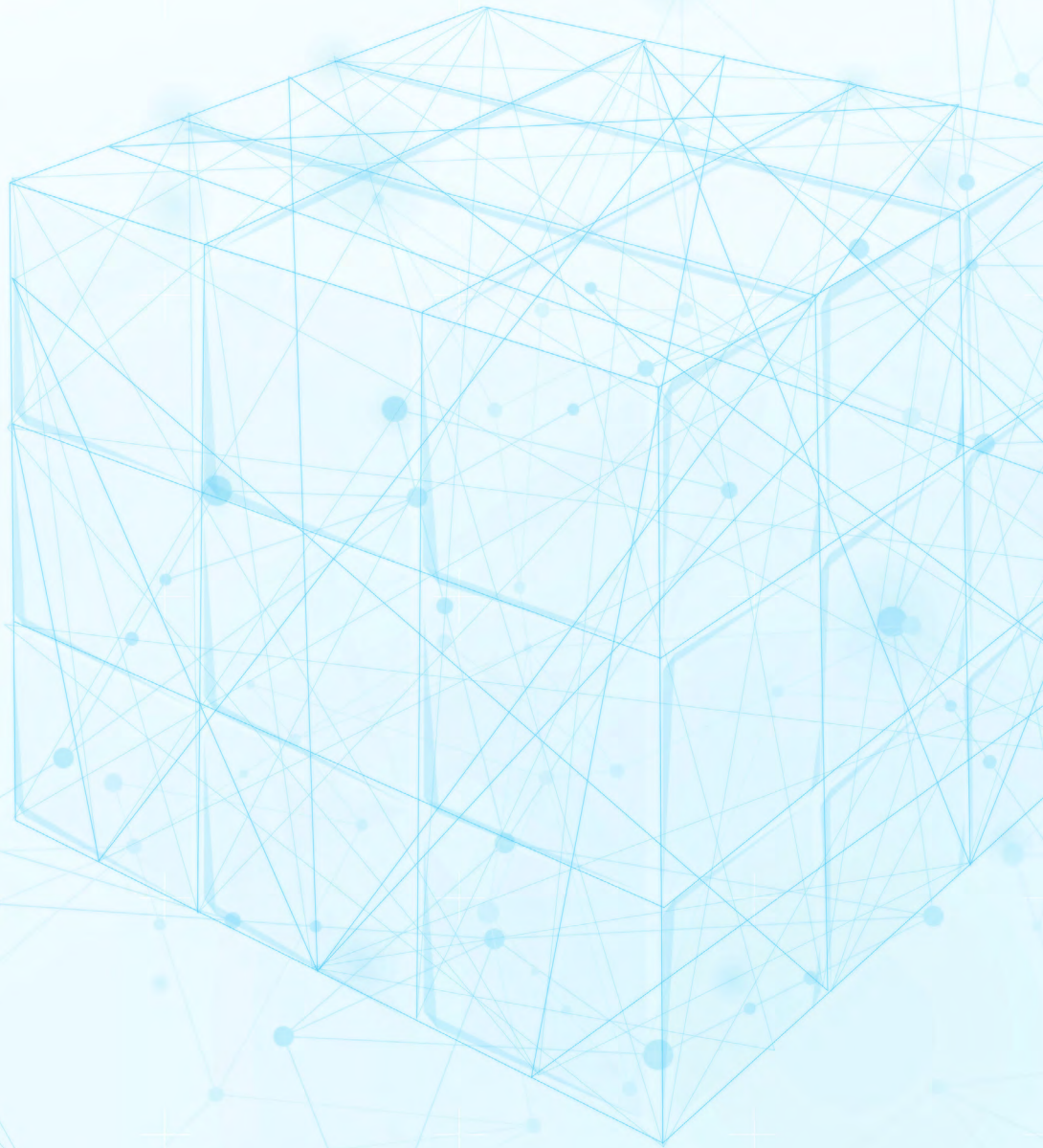
Offshore Energy Digital Architecture

NZTTP Programme

# Technical Feasibility

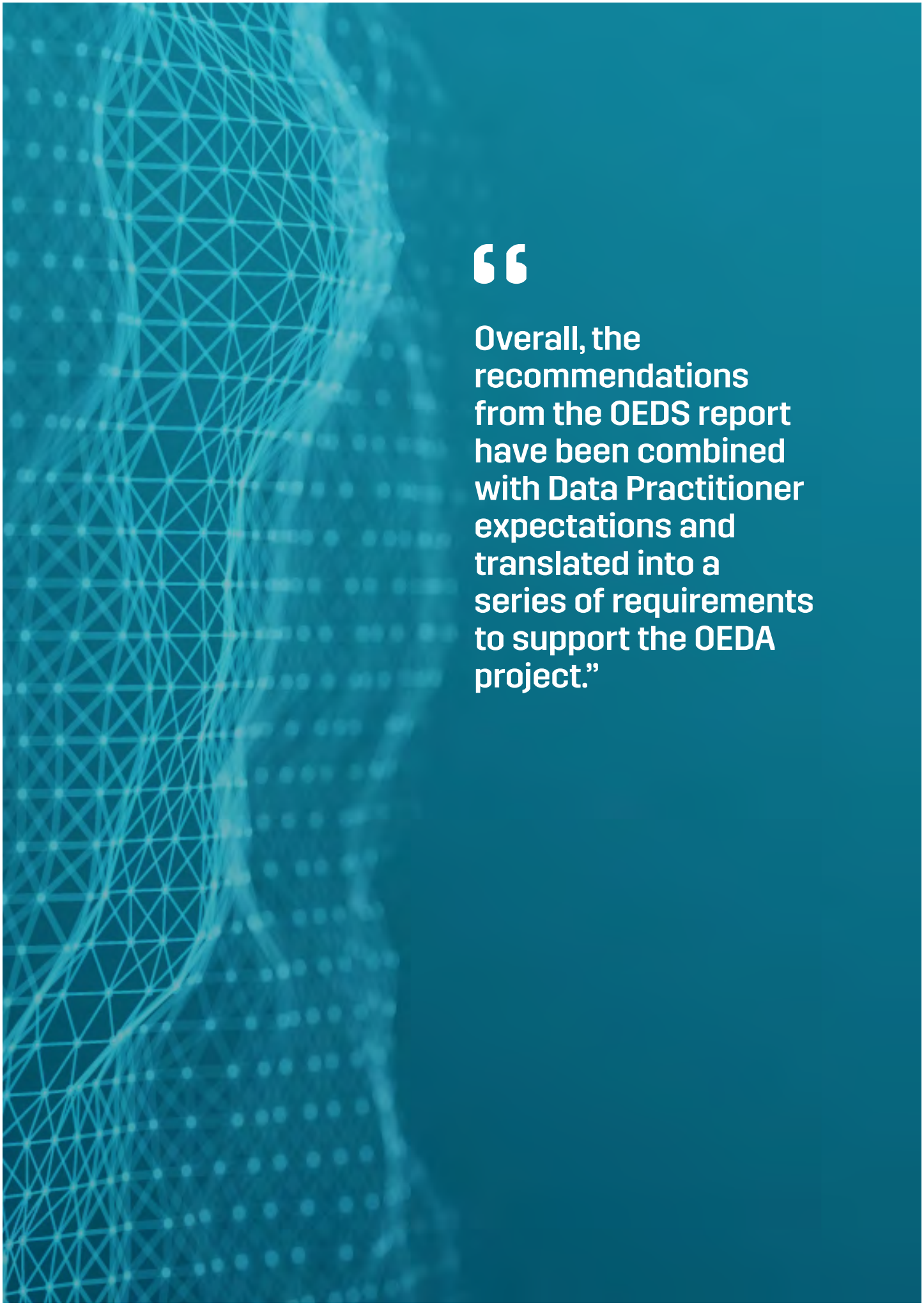
OEDA



Net Zero Technology Transition Programme

Net Zero Technology Centre

Technology Driving Transition



> “Overall, the recommendations from the OEDS report have been combined with Data Practitioner expectations and translated into a series of requirements to support the OEDA project.”

# Contents

# 1.0

# Summary

The Offshore Energy Digital Architecture (OEDA) project supports the integration of the data and digital infrastructure that is required to deliver the future offshore energy system and demonstrate that we can secure, capture and make available critical industry data, in a manner which is as open as possible."

The project aims to create "a digital energy technology ecosystem which will maximise the UKCS-related digital activity"[1]. OEDA is fundamentally a pilot data sharing platform that enables awareness and access to relevant datasets, to enable shared analytics and increased use of data across the sector to support decision making, increased use of automation, remote control technologies and improved operational efficiency.

The Net Zero Technology Centre has partnered with InDHu, a start-up that has the principal members responsible for driving the digital transformation at Airbus, to provide a literature review and configure Foundry for the pilot in phase 2 of the OEDA project.

OEDA Report 1 - Data Sharing Landscape captured the output of an extensive literature review that defined the OEDA Requirements from a consolidated set of recommendations, best practices and lessons learned from existing implementations across a chain of eight reports in the wider energy sector, both onshore and offshore from June 2019 to June 2022.

The purpose of OEDA Report 2 – Technical Feasibility is to demonstrate that the OEDA requirements can be met with a high confidence similar to the UK Government definition of Technology Readiness Level 6 (TRL6) to deliver on the Offshore Energy Data Strategy (OEDS) Data Catalogue and Data Fabric recommendations. The assessment was conducted using an example architecture based on open-source software using the Make implementation strategy - in essence taking (multiple) existing applications, combining, extending and configuring them to meet the design goals. It showed that not only is OEDA technically feasible, but the technology base is mature, has a range of potential suppliers and investment in this area would facilitate key skills in the wider digital economy.

A number of permutations are recognised in deployment environments (On-Prem, Cloud & Hybrid), approaches (Bare Metal, Virtualized, Containerised & Serverless) and implementation strategies (Make, Build & Buy) and considered in concluding that the associated confidence level was high.

The importance of OEDA Report 2 – Technical Feasibility is to demonstrate the range of options that are currently available to achieve the OEDA requirements for the OEDS recommendations and to show industry that the technology exists with a focus on how it is deployed.

[1] The Oil & Gas Technology Centre (2020) - Net Zero Technology Transition Programme - Appendix VII Offshore Energy Digital Architecture (OEDA) Business Case.

# 2.0

# Offshore Energy Digital Architecture (OEDA)

There are five planned reports in establishing a sector wide OEDA:

1

OEDA
Data Sharing Landscape

2

OEDA
Technical Feasibility

3

OEDA
Pilot Architecture and Ontology Design

4

OEDA
Potential Business & Cost Model based on Pilot

5

OEDA
Review

This report is the second in the series, and it determines whether an OEDA Data Sharing Platform is technically feasible using an example Open Source based architecture to perform the evaluation. The third and fourth report is based on the evaluation of the pilot data sharing platform and associated business and cost model. The final report documents the OEDA project and provides recommendations to establish next steps.

To help determine requirements for a sector wide data sharing capability, the OEDA project will use Palantir Technologies' Foundry[2] platform along with InDHu[3] as partners for a pilot. This was primarily due the success of Foundry in the aviation sector with the implementation in Skywise[2]. Airbus was able to create an ecosystem aimed at accelerating and expanding the exploitation of aviation data across multiple parties from customers, suppliers and even competitors in the field of aircraft maintenance.

The foundation for their digital platform was Foundry and many of the key personnel who supported the Airbus Digital Transformation are now part of the InDHu start-up. In the best traditions of the NZTC in trialling new technologies for the offshore energy sector, the OEDA project evaluates Foundry as a pilot for the OEDA Data Sharing Platform with the expertise of InDHu in its deployment and configuration.

The purpose of this report series is therefore not to substantiate retrospectively the pilot selection. The scope is to gather existing implementations, recommendations and best practices from the wider energy sector into a preliminary set of requirements to evaluate the pilot and inform subsequent platform evaluations from other providers. Experience from the pilot will help determine and refine the proposed OEDA Requirements to support subsequent phases that will eventually lead to a tender for a data sharing platform.

[2] Palantir Technologies (2023) - Palantir Foundry
[3] InDHu (2023) - Industrial Data Hub
[4] Airbus (2023) - Skywise | Enhance | Services

# 3.0

# Scope

The purpose of the report is to determine to what extent OEDA is technically feasible by evaluating an example architecture based on data industry standard technologies that meet the OEDA Requirements[5]. The intent is to provide sufficient fidelity to determine the confidence in achieving each of the requirements, but without the extensive effort of generating a detailed design of a data sharing platform. The assessment will also consider the associated deployment considerations as a function of people, technical risk, cost and organisation focus. As there is a large matrix of combinations for deployment environments (On-Prem, Cloud & Hybrid), approaches (Bare Metal, Virtualized, Containerised & Serverless) and implementation strategies (Make, Build and Buy), the report focuses on a single approach for the purposes of technical feasibility and considers the relative merits of the others based on InDHu experience.

[5] NZTC (2023) - OEDA Report 1 - Data Sharing Landscape

# 4.0

# Background

In 2020, the Business Case for OEDA (included in Appendix VII of the Net Zero Technology Transition Programme report) identified "the complexity and the scale of the challenge to integrate the data from multiple organisations, sectors, technologies, and solutions is substantial. There is a significant risk that meeting the 2045 net zero target will be impossible without investment in deploying key digital technologies in support of this target. Transformation will be excessively costly if these technologies are not deployed in a co-ordinated, collaborative way to avoid a slower more expensive transformation"[6].

OEDA is fundamentally a data sharing platform that enables awareness and access to relevant datasets, demonstrates "shared analytics platforms that are as open as possible" and promotes "increased use of data across the sector to support decision making, increased use of automation, remote control technologies and improved operational efficiency".

In August 2021, the Scottish Government awarded NZTC a £16.5million investment programme[7] into accelerating a range of energy transition projects to help deliver Scotland's net-zero economy. The Net Zero Technology Transition Programme is expected to enable £403billion for the economy and 21,000 jobs by 2050; it covers seven projects that have matched funding from industry.

Many of the stakeholders for OEDA include participants in the Offshore Energy Data Strategy (OEDS) Taskforce, which made two key strategic recommendations with regards to a data sharing platform. OEDA is not an isolated initiative but forms part of significant movement within the wider energy sector that has produced multiple projects and at least eight related reports, both onshore and offshore, over a three-year period between June 2019 to June 2022.






[6] The Oil & Gas Technology Centre (2020) - Net Zero Technology Transition Programme - Appendix VII Offshore Energy Digital Architecture Business Case.
[7] Scottish Government. (2021) - Investing in net-zero technology - gov.scot

# 5.0

# Requirements and Technical Approach

The technical OEDA Requirements are centred on the Offshore Energy Data Strategy[8] recommendations for an Offshore Energy Data Catalogue (OEDC) and Data Sharing Fabric (DSF). To better understand the context around these, an extensive literature review was conducted to derive a set of requirements based on a consolidated set of recommendations, best practices and lessons learned from existing implementations across a chain of eight reports in the wider energy sector, both onshore and offshore from June 2019 to June 2022. These were captured in OEDA Report 1 - Data Sharing Landscape[9] and presented in two tables.

The primary or OEDA Requirements are based on the wider energy sector (and prefixed with "E"), however based on InDHu experience, a second set of requirements was also proposed reflecting the expectations of Data Practitioners (prefixed with "D") and are both presented in Appendix A. As the Offshore sector has yet to accept the proposed OEDA Requirements, both sets of requirements will be evaluated to determine whether the OEDA Data Sharing Platform is technically feasible.

To determine technical feasibility, the relevant technologies were examined using an example architecture to provide a realistic environment and configuration. The assessment was not restricted to just the capability but also considered how the technology is likely to be configured, (if necessary) modified, deployed, maintained and supported to determine the confidence level. This was assessed using a simplified three-tier rating system of low, medium and high, where high is similar to a Technology Readiness Level (TRL) of 6, based on the definition used throughout the UK government.[10]

The following sections introduce the different deployment approaches as well as the different implementation strategies.

**To determine technical feasibility, the relevant technologies were examined using an example architecture to provide a realistic environment and configuration.**

8 Energy Systems Catapult (2022) - Delivering a Digitalised Energy System
9 NZTC (2023) - OEDA Report 1 - Data Sharing Landscape
10 DSTL (2023) - DASA Standard Terms and Conditions

# 5.1 Technical Deployment

A key input in considering technical feasibility is the deployment environment and computational resources required to support the technology. For example, an application that requires 32 processor cores, 64 GiB of memory and only works when installed directly on Microsoft Windows 7 would be difficult for most IT teams to deploy and potentially costly. The compute resource requirements for Q4 2023 are in the top 10%, the support for a single legacy operating system installed on a Bare Metal server would also have licensing and supplier support challenges. In contrast, an application which had the same compute requirements (processor cores and memory) but supported recent editions of Windows, MacOS, Linux as well as native support for Virtualised and Containerised deployment would impose minimal constraints on IT teams.

Therefore, the assessment considered the most common deployment environments (On-Prem, Cloud & Hybrid) and approaches (Bare Metal, Virtualized, Containerised & Serverless). Traditionally, applications were deployed on servers on company premises (On-Prem) but due to the introduction of hyperscale cloud providers, there has been a trend to host in the Cloud with its improved connectivity, potential for greater availability and some reduced cost by only using the compute resources needed. To optimise for costs, some businesses also adopt a hybrid approach between On-Prem and Cloud, where a service is deployed locally until it grows to a scale where additional resources are required and then scaled up in the Cloud. Another design pattern is to have a service running on both On-Prem and the Cloud simultaneously for resilience.

With regards to deployment approaches, traditionally applications were deployed on Bare Metal servers, that is directly on machines with just an operating system (such as Microsoft Windows). The move to virtualization through popular technologies from VMware, Xen and more recently Hyper-V from Microsoft allowed multiple operating systems and therefore complete isolation of multiple applications on the same hardware. Increasing application density reduced the hardware costs per application and encouraged greater utilisation of resources with a positive impact on the environment. The drive for containerisation popularised by Docker as a tool, format and ecosystem was in effect for application isolation without having to resort to operating system isolation as well. The containers were, therefore, much lighter, faster to start (and therefore easier to horizontally scale by having multiple copies) than traditional virtual machines. This layer of abstraction permits users to focus on just the application and in effect allow a service provider to manage the compute resources and operating system maintenance.

A further level of abstraction is the concept of Serverless or Cloud functions; rather than pay a service provider the costs of hosting a virtual machine or container whether it is being used or not, the concept is to pay for only the resources used by the application or program when actually run. Consider a service that removes the background from a customer provisioned image - the server version of the application will be hosted somewhere either as a VM or container to ensure it is available when the customer uploads an image and therefore, costs are involved even if the service is idle. In contrast, the serverless equivalent would see the application only being initiated when an image has been uploaded, and therefore the costs are entirely dependent on processor, memory and network utilisation for only the duration of the program and not how long the application has been available. The costs associated with keeping the application in a ready state are covered by the Cloud provider.

The key advantage of this approach for the developer is the complete abstraction of the infrastructure (no servers nor network to manage and maintain), potential for unlimited scale and if the application is compatible with this design, significantly cheaper at a certain scale. It is a design pattern common with large scale mobile gaming applications, but is not suitable for all types of use cases, such as where tracking state is important (e.g. Databases). If a program is idempotent, self-contained and in general has a single input and output, then it may be suitable for a serverless approach. In practice, some applications are purely serverless but many use a hybrid approach, where the bulk of the on-demand computation is serverless.

There are other deployment approaches which use a combination of the five that have been discussed such as micro VMs or the use of co-location in a data centre but these are the main approaches, and will be used as part of the assessment process in determining the confidence level of the various technologies.

## 5.2 Implementation Strategies

It is proposed to simplify the implementation strategies by grouping them into the three categories of Make, Build and Buy; the definitions are not precise and there are degrees of overlap but Make is in essence taking (multiple) existing applications, combining, extending and configuring them to meet the design goals, then deploying and maintaining the subsequent services with minimal external (commercial support). This reflects the free to use element of most open-source software but places the heaviest burden in maintaining, running and support whilst in parallel addressing the challenges of integrating multiple disparate services together. This approach is applicable to any application whether commercial, free for commercial use or open source.

The concept of Build is to take ready-made deployed services and combine them much like in the Make approach to deliver the design intent. This is the Software-As-A-Service (SaaS) paradigm, common with open-source projects, where a commercial entity is set up to deploy and support the application. By removing the burden of running the application itself, the host organisation can focus on more value-added elements of the overall service. This offloads the majority of the maintenance, operational and support activities for each individual application but with the same technical challenge in integrating multiple applications and the associated maintenance and support.

It should be noted that not all SaaS applications offer the same level of service for the same open source product. For example, Microsoft's Azure offers the PostgreSQL database as a service[11] but only manages the underlying compute and running of the application. Maintenance or other database related activities are not included so an organisation consuming such a service will need to provide its own database administrators. In contrast, a provider like Crunchy Data[12] offers to manage the database in its entirety and also provides technical guidance on how to use it effectively. The same approach is also applicable to commercial applications such as an Oracle Database[13], which is available in standalone form and as a Database-As-A-Service (DaaS).

The final concept is called Buy, which could be buying the platform from a third party that itself Makes or Builds it, or an integrated data platform provider such as Dataiku or Palantir's Foundry. In either case, the responsibility of the organisation is predominantly to configure it to their needs, with some potential adaptations. Relative to Build (which reduces the operating, maintenance and support burden), the Buy approach also removes the majority of the integration burden since a single combined service is being provisioned. In short, for Make the host organisation is responsible for deployment, maintenance and integration, for Build primarily integration (and its maintenance) and for Buy, potentially neither.

[11] Microsoft (2023) - Azure Database for PostgreSQL
[12] Crunchy Data (2023) - Crunchy Data
[13] Oracle (2023) - Database | Oracle

# 6.0

# Data Provider Assumptions

The Offshore Energy Data Catalogue and Data Sharing Fabric are part of the same ecosystem with data providers and therefore subject to their technical limitations, with at least two sets of integrations required:

**Automated metadata transfer**

**Authentication and authorization between the fabric and the data provider**

The novel component in the OEDS-defined data catalogue is the inclusion of a metadata aggregator. The technical burden in the integration between the two systems can potentially have two extremes; the first being that there are no changes made to the data providers. In this instance, the aggregator needs to support a variety of interfaces to extract the metadata from a data provider provisioned API, to accessing a portal securely and ingesting an XML file, to web scraping using a bot and performing complex post processing akin to web crawlers used by search engines.

The other extreme (adopted by the Ice Breaker One[14] and advocated by the EDTF approach) is to put the burden on the data provider, in either constructing an API to a set standard or deploying a metadata depositor - an automated means of translating the data provider's dataset format and metadata into a format compatible with the catalogue at the data provider's technical expense.

The latter may dampen data providers engagement, where it remains unclear which standard of data catalogue within the wider energy and offshore industries will prevail. As defining the technical and metadata standards of the data provider are out of scope for this phase of the OEDA project, the following assumptions are proposed for this activity. For machine unfriendly formats (such as irregular or poorly constructed data), it is proposed either the data provider hosts a compatible API with known schema or uses other automated means to deposit the data. For tabular data, the metadata aggregator shall access either the whole dataset or sufficiently representative samples.

The approach for the Data Sharing Fabric integration with upstream data providers is the same, although easier technically. If the data providers use common website based authentication (e.g. HTTP Basic Auth or Digest Auth), then they can relatively easily be supported but to meet cybersecurity requirements, the use of an Identity Provider (IdP) is strongly recommended. In practice, most organisations have migrated to video conferencing technologies such as Microsoft Teams, which require authentication with a corporate IdP anyway and therefore integration to the DSF will be relatively straightforward.

There is a more fundamental role from data providers to support industry wide collaboration albeit outside the scope of OEDA and that is the provision of data in machine readable formats. Much of the energy sector, both onshore and offshore, assumes the role of growing capabilities such as automation and application of Artificial Intelligence to bring significant benefits to the sector and support the goal of Net Zero. However, that is predicated on the datasets being machine readable, which is subject for data provider standards (and out of scope of this report) but it has a key technical impact on data practitioner engagement. For the purposes of this activity, the technical choices have been made to accommodate the majority of the uncertainty in data provider interfaces, ultimately it will manifest itself as additional time, cost and technical complexity in the integration of the metadata aggregator.

[14] Icebreaker One (2023)

# 7.0

# Technical Feasibility

The purpose of this section is to identify a range of applications and services, when combined with the correct configuration and if necessary, any development activity will deliver the OEDS Data Catalogue and Data Sharing Fabric by meeting the OEDA Requirements. At a high level, all of them can be met with largely open-source software, potentially self-hosted with the exception of the identity provider, which is best managed by a specialist or dedicated organisation to meet cybersecurity best practice.

The assessment conducted is based on the Make approach using an example architecture based on open-source software to assess the technical feasibility of meeting the OEDA Requirements and the data practitioner requirements; the intention is not to provide a detailed design or attempt to optimise the deployment approach. As has been highlighted, there are many permutations to Deployment Environments (On-Prem, Cloud & Hybrid) and approaches (Bare Metal, Virtualized, Containerised & Serverless) and Implementation Strategies (Make, Build & Buy) and covering every permutation is unrealistic. It is recommended prior to the tender phase for OEDA that all relevant combinations are considered but from a technical feasibility perspective only a single example is required.

**The example architecture is based on open-source software and the Make approach for the following reasons:**

- A key theme and concept throughout the range of energy sector reports is Openness and the recommendation to use open source software as captured in the OEDA Requirements[15] (Req ID. E3) and recorded in Appendix A.
- As there is no single product or service that meets both sets of requirements, the Make approach provides the most flexibility and therefore the most likely to meet the requirements if it is technically feasible.
- Majority of the Build options are themselves predicated on open source products with the notable exception of databases such as Snowflake[16] or Amazon's DynamoDB[17]. The exception also demonstrates that in other areas, even Amazon Web Services (the world's largest cloud provider[18]) offers equivalent SaaS products of popular open source software (including databases despite its own proprietary product).
- The same open-source components also form the key components of fully integrated platforms such as that from Databricks[19] (e.g. Apache Spark, Delta Lake and MLflow), Datalku[20] and Foundry from Palantir. For example, Foundry uses the PostgreSQL[21] (referred internally as Postgate) databases where fast computation of tabular data is required, Elasticsearch[22] (referred internally as Phonograph) for so called objects and like major Big Data platforms, Apache Spark[23] for all other datasets.

As there are many permutations in meeting the OEDA and data practitioner requirements, generally one or two examples will be provided to demonstrate that it should be technically feasible. As the technologies utilised are mature and popular within the data industry, there are potentially hundreds of related implementations and associated suppliers that could meet the individual needs identified for the Make and Build approach. For the Buy approach, a data catalogue and the concepts underpinned by the DSF are key components of most data analytics platforms such as Datalku, Palantir Foundry, Data Robot etc.

All of the example technologies are open source (Req ID. E3) and free for commercial use unless otherwise stated.

---

[15] NZTC (2023) - OEDA Report 1 - Data Sharing Landscape
[16] Snowflake (2023) - Snowflake Data Cloud
[17] Amazon Web Services (2023) - Amazon DynamoDB
[18] Statista (2023) - Global Cloud Infrastructure Market Share
[19] Databricks (2023) - The Data Lakehouse Platform
[20] Datalku (2018) - All About Open Source
[21] The PostgreSQL Global Development Group (2023) - PostgreSQL
[22] Elasticsearch B. V. (2023) - What is Elasticsearch?
[23] The Apache Foundation (2023) - Apache Spark

# 7.1 Data Sharing Fabric

There is no single open-source application that satisfies the requirements of the OEDS definition of the Data Sharing Fabric. Instead, the aims can be achieved by the adoption of an identity provider (IdP) coupled with an API Gateway (Req ID. E2). For the IdP, a popular example is Keycloak[24] sponsored by Red Hat or Jannsen[25] with its associated commercial SaaS called Gluu[26]. In all cases they support all of the industry protocols (Req ID. E10), provide key features like Single Sign On (SSO), connect to a variety of existing authentication systems as well as provide secure modern access controls (e.g. multi-factor authentication and passwordless access etc).

In addition to providing basic Authentication (AuthN) and Authorization (AuthZ) services, both support a Zero Trust model with time limited tokens (Req ID. E10). Furthermore, the support for the OAuth 2.0 and the opinionated OIDC protocols enable a User (human) to authorise a machine to obtain its own individual access to a protected resource (Req ID. D2). Alternatively, support is also available for client and server-side Transport Layer Security (TLS) certificates (similar to the Icebreaker One implementation) but not recommended due to the burden of maintaining one's own Certificate Authority and securing the associated secrets (Req ID. D2). It should be stated that all communication will be encrypted using TLS and is addressed in the Technical Deployment section.

Although there is enterprise standard open source IdP, the practice of running one's own service is highly discouraged as the impact from a minor misconfiguration can be disproportionate. Furthermore, additional steps to provide defence in depth such as monitoring and hardening of access points are difficult to realise and best left to specialist teams. A similar argument is applied with developing and deploying one's own encryption library - it is a highly specialised area and difficult to get right. For this reason, to meet cybersecurity best practices, it is better use a trusted service provider instead i.e. opt for a Build approach with an open source provider or an industry leading IdP such as Okta[27].

Example, open-source API Gateway providers are Tyk[28] and Kong[29] (Req ID. E2); they have similar features in that they support authentication and authorization through OAuth 2.0 and OIDC and, therefore, integrate seamlessly with the selected identity providers. A Gateway API provides a range of services to simplify access to multiple APIs (e.g. from different data providers) or HTTP resources (Req ID. E5), permits common security policies - the enactment of the Governance - (e.g. tag APIs as Ooen or Shared) and a range of security measures from rate limiting (to protect against accidental and deliberate abuse) and monitoring capabilities (Req ID. E6). The use of time limited token-based authentication from OIDC supports the Zero Trust model (Req ID. E10).

The configuration can be controlled and a version history maintained based on the git version control system (Req ID. D1). In the simplest setup, users on the IdP can be characterised on their level of open and shared data source access. Each API added to the API Gateway can also have a security policy that mimics the same categorisation. When a user then attempts to access an API, the gateway checks if they are authenticated first and then checks whether they are permitted to access that resource (authorized). This separates the two concerns between the systems. The focus of the IdP is on the users and what resources they can access, whereas the gateway is used to tag and control access to the resources (such as data providers and their content) appropriately.

Both products have equivalent SaaS options that provide additional enterprise friendly features, the use of dashboards and customer support for the Build implementation approach. This demonstrates that OEDA Requirements Req ID. E2, E3, E5, E6, E10, D1 & D2 can be met.

[24] Red Hat (2023) - Keycloak
[25] The Janssen Project (2022) - Janssen Documentation
[26] Gluu Inc (2023) - Gluu Server
[27] Okta (2023) - Okta UK
[28] Tyk Technologies (2023) - Tyk
[29] Kong Inc (2023) - The Cloud-Native API Gateway

## 7.2 Data Catalogue

The majority of the OEDS Data Catalogue functionality could be met by the Icebreaker One implementation used for Open Net Zero based on the open source data management platform CKAN[30] if the data provider adheres to the Icebreaker One Trust framework, as discussed in OEDA Report 1: Data Sharing Landscape. Whether the architecture can be scaled to deliver a sector wide Data Sharing Platform should be investigated in the latter phases of the OEDA programme but there are three design choices which may impact data provider participation and the ability to meet OEDA Requirements.

The Icebreaker One team uses a Financial API (as used in the open banking industry) for good levels of security but that imposes a technical burden on data providers to become compatible by adopting their data provider API. Secondly, the Icebreaker One equivalent of a metadata aggregator is therefore predicated on a HTTP based API, which, as discussed in OEDA Report 1, will be unsuitable for the much wider range of datatypes expected in the offshore industry relative to the onshore energy industry. The final design choice around user accounts and subsequent use of an authorization server (referred to in the documentation) is unclear and therefore it is not possible to assess if it meets cybersecurity expectations.

### To determine technical feasibility the following open-source components were used in the example architecture:

**Core Data Catalogue functionality**
e.g. Open Metadata

**Hosted git Version Control System (VCS)**
e.g. Gitlab or Gitea

**API management using API Gateway**
e.g. Tyk or Kong

**Authentication and Authorization using Identity Providers**
e.g. Keycloak or Jannsen

**Computational framework to support data exploration**
e.g. JupyterHub or Binder

**Workflow Orchestrator for metadata aggregator**
e.g. Apache Airflow or Dagster

**Custom Application Programming Interface (API)**
e.g. Spring (Java) or Fast API (Python)

**Web User Interface - any combination of JavaScript and Cascading Style Sheets (CCS)**
e.g. React and Tailwind CSS

[10] Open Knowledge Foundation (2023) - CKAN

As stated previously, there are many permutations of technology stacks that achieve the aims of OEDA but to demonstrate the technical feasibility an example architecture will be introduced in three parts to reflect broadly the three types of workflows: a user browsing the data catalogue, user access of a data provider and the use of APIs for automation to perform both tasks. The first figure shows a user browsing the data catalogue:

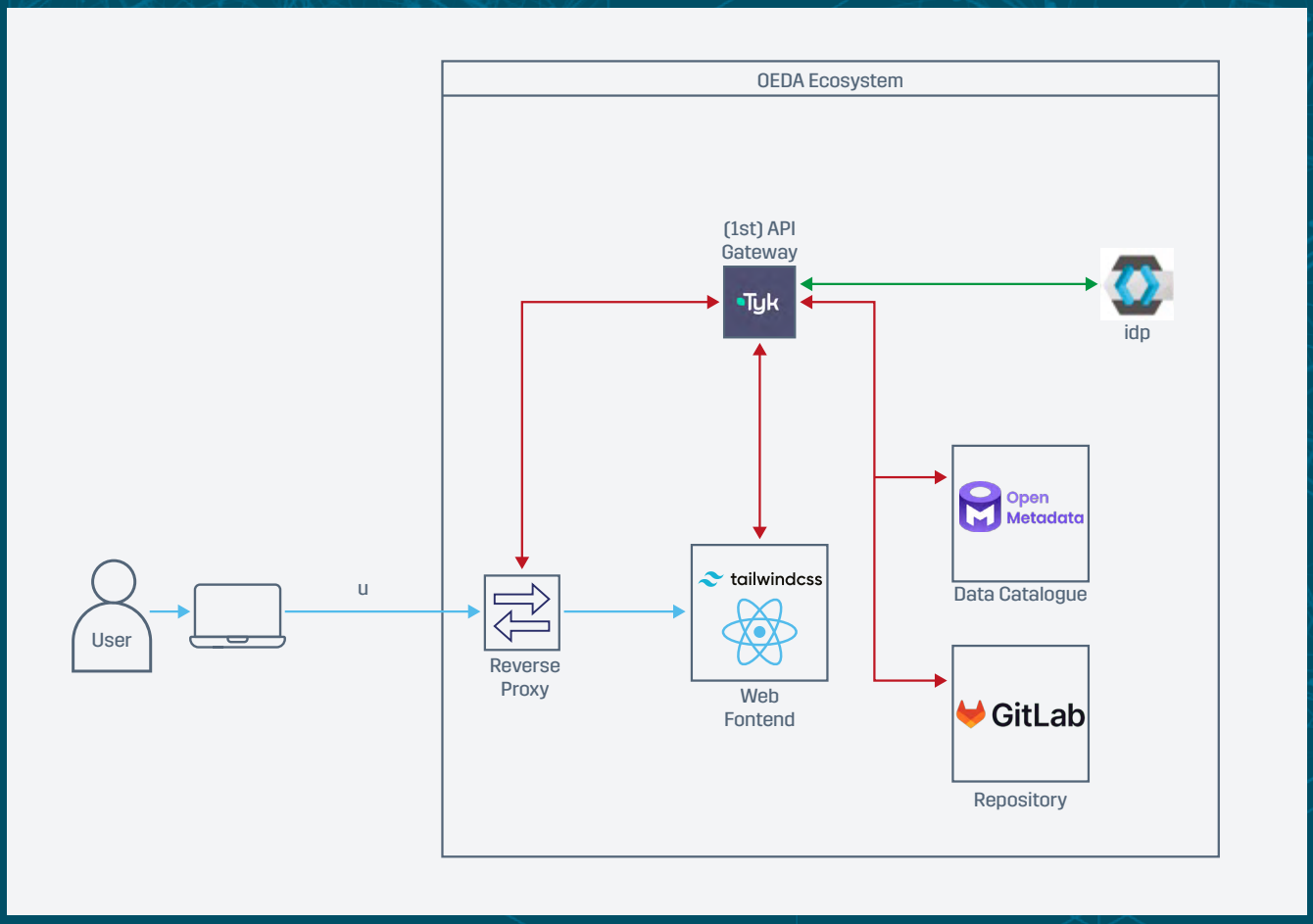



Figure 1: User Interface Workflow

Each application is shown as a single box but in practice to support the scale of users and meet reliability expectations, they would in fact be deployed with multiple instances. Co-ordination, or load balancing, between them is done by the Reverse Proxy (e.g. Cloudflare or Nginx). It also provides a basic level of protection against common threats and directs the user to the chosen instance of the web frontend or application. It is proposed a unifying interface is provided rather than direct access to the underlying applications to enable a better comparison with existing integrated data platforms and provide a better experience.

To access the catalogue (e.g. open metadata), the user must authenticate (AuthN) through the web frontend (via the blue stream), which is done using an internal API call via the 1st API Gateway (e.g. Tyk via red stream) to the IdP (e.g. Keycloak via green stream). Note that there are no direct connections between the applications as all internal communication is made via the API Gateway to provide segmentation and isolation of services consistent with Zero Trust. Depending on which aspect of the catalogue is accessed or repository, further authorisation (AuthZ) checks may be made. This demonstrates how the Data Sharing Fabric and catalogue work together.

The figure below shows the next step in accessing a source from the upstream data provider:

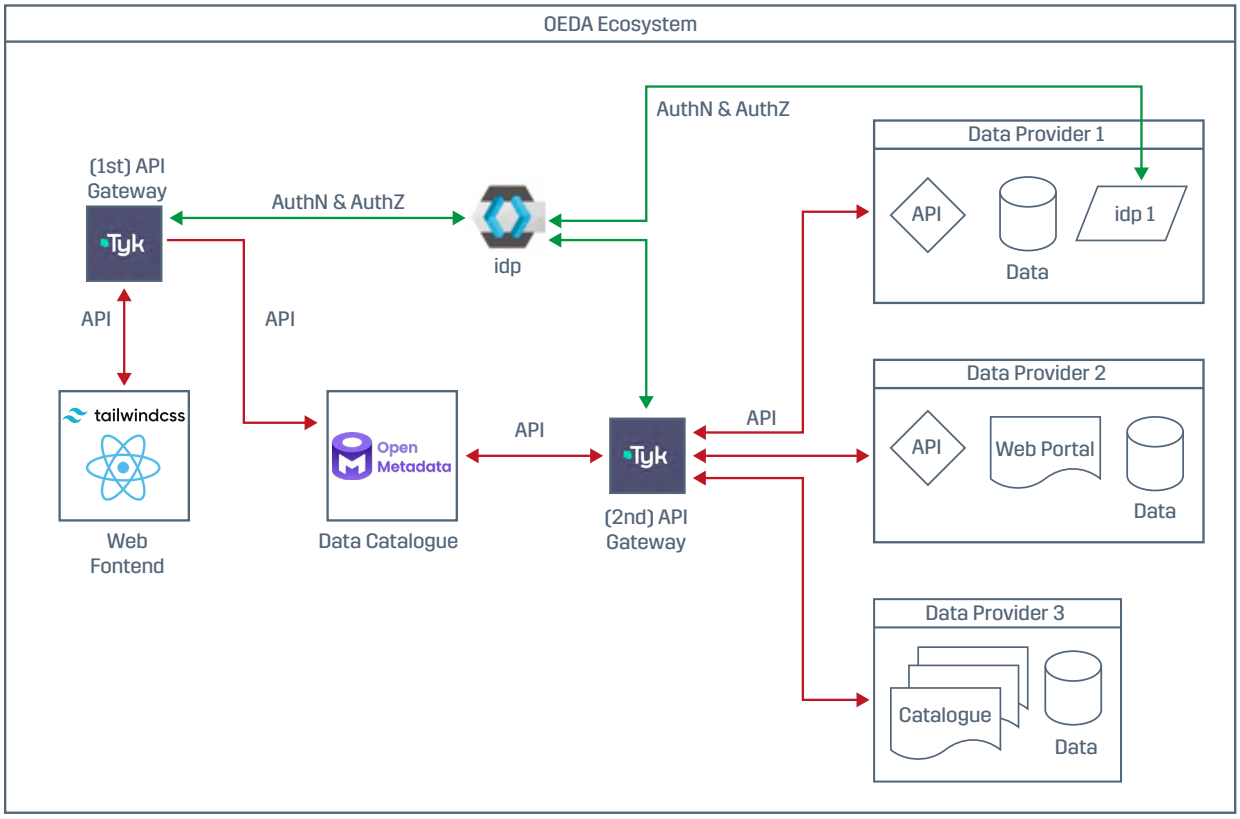



Figure 2: User Access to Data Provider Workflow

Once a user has decided to access a source hosted by an external data provider, clicking on a link in the Web Frontend triggers an internal API call to the Catalogue to retrieve the relevant details of the provider and initiate contact with them. It should be noted that a 2nd API Gateway is used to ensure the request is authorised from an authenticated user and protect the upstream providers. The first type of Provider may have their own IdP, which through the use of common IdP protocol such as SAML can be configured to accept users authenticated and authorised by the OEDA IdP or Data Sharing Fabric. Subsequent access to the data can be through HTTP redirection or revealing a time limited token and API endpoint to the User to initiate a download of the data.

Data Provider 2 has no independent IdP and therefore the use of long-term security tokens can be used to issue bespoke and time limited access to OEDA authenticated users to the provider's API (similar to the Icebreak One implementation). Data Provider 3 is running their own version of the OEDA platform and therefore the metadata is compatible but may choose not to support direct access to the data or direct authentication, and therefore the user will be redirected to the separate instance of OEDA.

The final workflow is demonstrating API access, in particular supporting machine-to-machine communication:

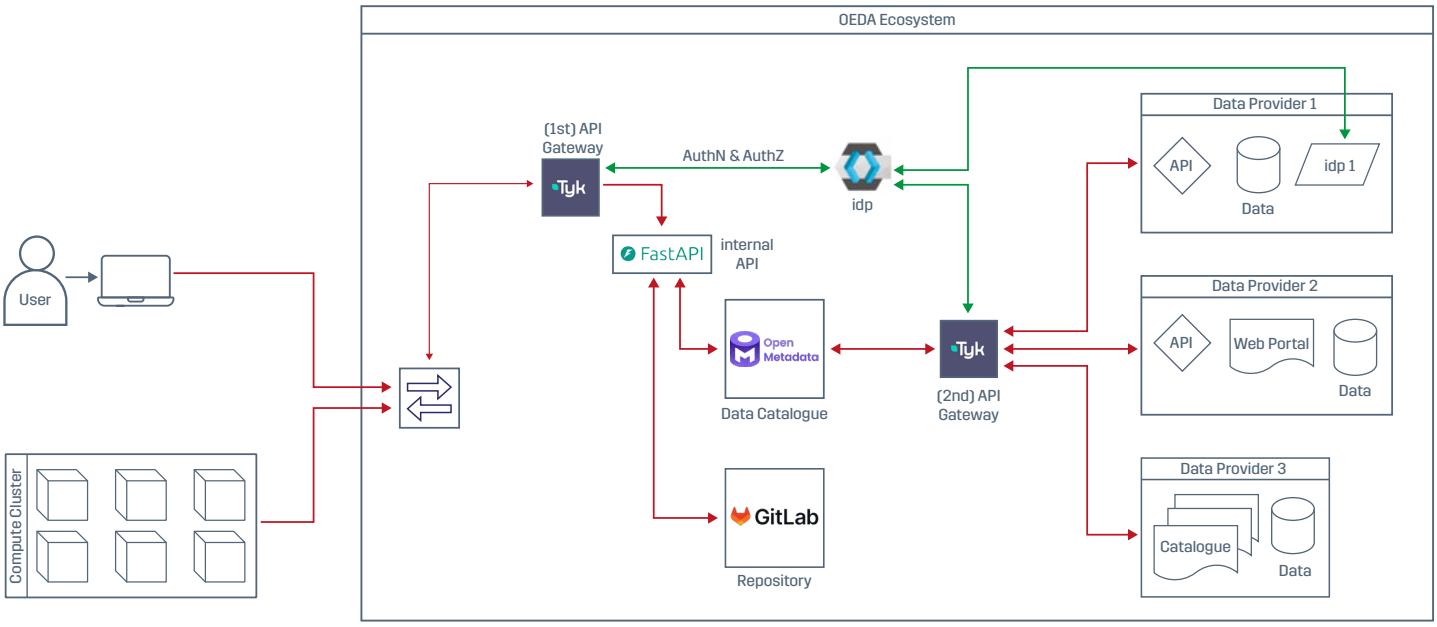



Figure 3: API Workflow

The figure depicts a user seeking to authenticate and authorise their compute cluster to automatically engage and process data from OEDA, for example assessing high fidelity wind forecasts on a daily basis. The user initiates an API call to secure long-term access using features of the OAuth 2.0 / OIDC standard. This allows the IdP to provide two tokens, the first is for access (or access token) and a second refresh token that can be used to securely request another access token when the first expires.

The request enters via the Reverse Proxy, which directs all external API calls to the 1st API Gateway, which undertakes the usual authentication and authorisation checks using the IdP before giving access to the internal API. This processes the user request, in this instance by triggering the OIDC flow to authenticate the compute cluster with additional secrets to enable the machine to automatically refresh its access and therefore to maintain long-term automated access.

With the freshly minted access token, the compute cluster can automatically search and identify the latest datasets from the catalogue and access the upstream data providers using the same mechanism as used for human users. Combining all three figures shows a representative architecture:

The core functionality of the data catalogue is provided by open metadata which demonstrates its key features with animations in its documentation[31] including a metadata aggregator that could meet the OEDA Requirements dependent on data provider compatibility. At a high level, it consists of a number of connectors to integrate with data sources directly, profile them, provide a sample and enable users to provide rich documentation and context. This includes the provision of both baseline and custom metadata attributes, all of which are searchable.

It also provides an environment to connect with data providers, provide feedback, discuss datasets, make requests, construct dataset lineage graphs and receive notifications to a number of popular chat programs. The ingestion framework is itself based on Apache Airflow and can be customised to integrate a custom schedule or be triggered by external events such as receiving a webhook or observed change in data source state.

In terms of baseline features relative to the OEDS definition of a data catalogue, it has a data search capability through data discovery and access to the catalogue through a rich API. It has a built-in reporting function through data insights, this can be further customised through a dedicated application utilising the API and surfacing through the open-source API definition and visualised using the web framework. This is underpinned by a metadata store based on a MySQL database indexed using the optional Elasticsearch integration.

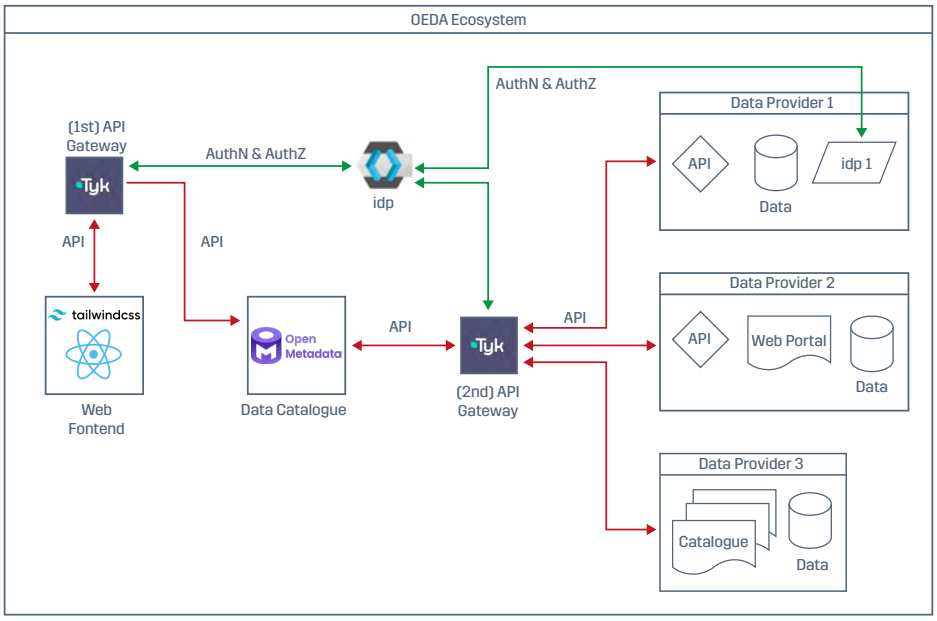



Figure 4: Example Architecture Overview

[31] OpenMetadata (2023) - OpenMetadata Features

The metadata aggregator is addressed using the Metadata Ingestion framework, which can be customised or extended with additional connectors to support a wider range of data providers and sources beyond tabular formats. Finally, federation between open metadata instances is not a built-in feature but the API enables such an integration in combination with the MySQL connector.

Relative to the OEDA Requirements, open metadata is open source software with a permissible licence (Req ID. E3). It can support potentially an unlimited number of attributes given compute limitations through custom properties (Req ID. E4). The rich metadata documentation (Req ID. D4) environment can support links to the upstream data providers transiting through the API Gateway to support authentication and authorization to the provider. The combination of tiering to highlight existing important datasets and conversation threads support the prioritisation of datasets for release, update or additional context (Req ID. E4 & E7). This also satisfies the requirement to provide direct feedback to the data asset owners (Req ID. E8).

Metrics for each data source is available via the API (Req ID. E6) and has native capabilities to provide visibility of relationships between the datasets. This is achieved with three features from open metadata: the use of tags or metadata attributes, automatic lineage from data association for tabular data or manually edited data lineage. This satisfies all of the energy sector based OEDA Requirements.

Open metadata has no direct capability to support the integration of git base repositories to ingest metadata or API configurations of the data providers. However, as the API is accessible, a custom workflow can be constructed (once agreed with data providers) to support such a feature. The repository can be hosted externally on an external git repository platform or hosted within OEDA with the choice of Gitlab (Req ID. D1). The orchestration of the ingestion process can be facilitated with the workflow orchestrator selected (e.g. Apache Airflow or Dagster).

The choice of identity provider dictates to what extent the highest standard of secure user and machine access is permitted on the platform. Whilst Keycloak offers good compatibility with most options, enhanced capabilities such as the use of Biometric Security keys requires the use of plugins and additional configurations. If good cybersecurity practices are observed and an external IdP is selected, then the full range of secure access methods is supported by the majority of providers (e.g. Okta). The same IdP also satisfy the requirement to permit secure long-term machine-to-machine communication using for example client workflow for OIDC (Req ID. D2).

Open metadata supports data profiling of tabular and some object storage based data sources if connected directly (Req ID. D3). For other machine readable formats, there are a number of options available depending on whether direct access to the data is permitted or not. Smaller datasets could be ingested into a computational framework within OEDA, a profiler executed and the results fed back in open metadata using the API. For larger datasets, external computational clusters could be orchestrated with the workflow orchestrators identified (e.g. Apache Airflow / Dagster). The same approach can also be adopted to host previews of the data that are not directly supported within open metadata.

Open metadata also supports rich formatting of content as discussed previously or alternatively linking to a git repository provides additional context and documentation opportunities (Req ID. D4). These repositories can also be used to define analytical environments that can be directly loaded into an external Binder instance to support data exploration or to an internally hosted Jupyter Hub instance (Req ID. D5). The relationship between the datasets, either directly or through manual linking, meets the data lineage requirements (Req. ID E9).

The final requirement is to support a variety of metadata formats beyond tabular data and the current attribute-oriented metadata. There are a number of options; a custom application could be used to capture the metadata and use an API to populate the hosted git repository. The custom metadata structure could be stored within a document database and integrated into open metadata using a standard connector, or a separate user interface could be provided for more complex datasets (Req ID. D6).

In summary, a design for OEDA has been proposed that meets both the OEDA and data practitioner requirements and therefore demonstrates that OEDA is technically feasible. All of the technologies cited are mature, have a wide range of suppliers and are common for a Digital economy: the use of databases, APIs, workflow orchestrators, secure access & authorisation and web-based technologies. This suggests that the confidence level is high with respect to technical feasibility.

# 8.0

# Technical Deployment

The previous section outlined a potential architecture and supporting technical description that satisfies the OEDA Requirements. The purpose of this section is to illustrate how such a service could be deployed, some of the key considerations and an illustration of the level of complexity in managing and maintaining such a service. As discussed previously there is a large matrix of combinations for deployment environments (On-Prem, Cloud & Hybrid) and approaches (Bare Metal, Virtualized, Containerised & Serverless) and implementation strategies (Make, Build and Buy).

The following table summarises the combination of deployment environments for each of the key components identified within the example architecture and is independent of the implementation strategy. The classification of support is defined in terms of three categories: native where the product directly supports this approach (as per its public documentation) and then either Yes or No, which reflects the opinion of the authors:

| Component / Deployment Option | On-Prem | Cloud | Hybrid | Bare Metal | Virtualised | Containerised | Serverless |
|---|---|---|---|---|---|---|---|
| Identity Provider (e.g. Keycloak) | Native | Native | Yes | Native | Yes | Native | No |
| API Gateway (e.g. Tyk) | Native | Native | Yes | Native | Yes | Native | No |
| Data Catalogue (e.g. Open Metadata) | Native | Native | Yes | Native | Yes | Native | No |
| Version Control System (e.g. GitlLab) | Native | Native | Yes | Native | Yes | Native | No |
| Computational Framework (e.g. JupyterHub) | Native | Native | Yes | Native | Yes | Native | No |
| Workflow Orchestrator (e.g. Dagster) | Native | Native | Yes | Native | Yes | Native | No |
| Custom APIs (e.g. FastAPI) | Native | Native | Yes | Native | Yes | Native | Yes |
| Web Applications (e.g. React with TailwindCSS) | Native | Native | Yes | Native | Yes | Native | No |

Table 1: Summary of Deployment Options

All of the example technologies utilised support deployment on locally owned hardware or in a Data Centre (On-Prem) as well as being suitable for deployment in cloud-based environments (e.g. AWS) either directly or using a SaaS equivalent. Hybrid deployment can mean either supporting the migration of services from On-Prem to Cloud or supporting both simultaneously. None of the example applications utilised directly support this use case but given the underlying technologies it should be feasible through careful selection of the product. For example, for the Workflow Orchestrator, Dagster is easier to migrate than say Apache Airflow but that is also a function of the approach i.e. Containerised applications are easier to migrate compared to Bare Metal installations.

All of the examples support Bare Metal installations and do not require access to specialised hardware and therefore are also suitable in virtualised environments running on Bare Metal (e.g. VMware ESXi or the open-source Xen or with an underlying operating system (e.g. the open source KVM or Oracle VirtualBox). Equally all of the examples support operation in Containers with native support for the OCI or Docker container format. These can be deployed using all common container orchestrators such as Docker Compose, Nomad and Kubernetes and their SaaS equivalents (e.g. Amazon Elastic Container service). The final deployment option of serverless is not suitable for this type of application and is unlikely to be supported without significant changes with the exception of supporting some API-based processing. Given the wealth of support for the other approaches it supports the conclusion that OEDA is technically feasible with high confidence.

**Given the wealth of support for the other approaches it supports the conclusion that OEDA is technically feasible with high confidence.**

# 9.0

# Implementation Strategies

Three broad technical deployment strategies called Make, Build and Buy have previously been defined. Make is taking existing applications, combining, extending and configuring them to meet the design goals. Then deploying and maintaining the subsequent services with minimal external (commercial support). This reflects the free to use element of most open-source software.

The concept of Build is to take ready-made deployed services and combine them much like in the Make approach to deliver the design intent. This is the Software-As-A-Service paradigm, common with open-source projects, where a commercial entity is set up to deploy and support the open source application. The final concept is called Buy and could be buying the platform from a third party that itself Makes or Builds it or an integrated data platform provider such as Datalku or Palantir's Foundry.

The focus of this report has been on determining the technical feasibility and associated confidence in delivering OEDA. The example architecture is one permutation on how the requirements could be met. Once a platform has been made, built or bought, there is a range of activities in how to deploy it outside of the technical arena such as onboarding users, training administrators and providing ongoing support. These topics are outside the scope of this report but are required considerations irrespective of the data sharing platform.

Whilst technically all three deployment strategies of Make, Build and Buy are likely to meet the OEDA Requirements, there are additional considerations to account for in choosing the specific strategy related to People, Technical Risk, the Output itself and ultimately all linked back to Cost. It is recommended that these features are assessed as part of the tendering process for the OEDA Data Sharing Platform.

The purpose of the following assessment is to provide at this early stage an indication of the relative impact on these considerations of choosing a particular deployment strategy. For many of the categories the same discussion could be had for the purchase of household furniture. For example, you could purchase the raw materials to construct, assemble and maintain a garden bench (Make) or purchase a flat pack version that requires assembly or integration only (Build), or finally Buy a fully assembled bench. The Make approach is likely to cost the least (depending on the tools required) but also require the most time to setup (or Develop), whereas the opposite is true for the Buy approach.

The table below is intended to convey or more critically prompt these types of considerations by starting with the example architecture and considering the relative impact of using the other two options of Build and Buy. For example, the Make approach requires the host organisation to stand-up a Development team to create the OEDA platform, but the Buy approach requires none. Both options require someone to administer the platform however, that is true independent on how the platform is sourced. Therefore, the assessment below is limited to the platform itself in terms of development, customization or adaptation to meet the OEDA Requirements. Support, maintenance and feature development are not assessed.

A key assumption is therefore that the output of the Make and Build approach is of a similar standard to the Buy approach in terms of maintenance and ease of use. For example, a common mistake in developing a platform is to make it easy for developers to maintain rather than the end users, something commercial providers focus on. This manifests itself as requiring a greater number of people to ensure the user interface is of the same quality as commercial providers.

The Make approach was used to establish a baseline and a qualitative assessment was conducted using a rating system of high, medium and low against the following key considerations. As stated previously, as the intent of this report is to establish technical feasibility a full combination of options was not assessed but recommended for the next phase of OEDA.

**The focus of this report has been on determining the technical feasibility and associated confidence in delivering OEDA.**

| Category | Consideration | Make | Build | Buy | Comments |
|---|---|---|---|---|---|
| People/Cost | Team Size | High | Medium | Low | The number of people due to a range of factors. |
| | Skills Mix | High | High | Low | Experience level and variety of skills required. |
| Technical Risk/Cost | Trend in Developmental Effort | High to Medium | Medium to Low | Low | The level of effort from the team in developing the platform over the time period. |
| | Responsiveness | Low | Medium | High | High refers to the relatively quick delivery of the platform or minimum time period. |
| | Maintain Burden | High | Medium | Low | Level of effort in skill type and resource to maintain. |
| | Scaling Challenges* | High | Medium | Low | Complexity in scaling to meet usage through number of users, data providers and size of datasets. |
| Cost | Tech Stack Cost | Low | Medium | High | Relative cost of the tech stack (e.g. combination of capabilities and applications). |
| Output | Requirements Completion Confidence | High | High | Medium | Likelihood of meeting all of the requirements. |
| | Organisational Distraction | High | High | Low | Level of deviation from host organisation's core focus. |

*Ease of platform scalability and not of underlying compute and storage

Table 2: Relative Assessment for the Different Deployment Strategies

The Team Size is a function of the scale and complexity of the task, which the example architecture illustrates requires component integration through the use of new code and sufficient redundancy in the core skills for a sector wide, Enterprise grade platform. The number of people required increases non-linearly due to additional supervision and support required. For example, a team of three can perform a degree of self-coordination, whereas a team of 10 may require a team lead and additional supervisors for effective team working like all organisations. The Make implementation requires the greater number of people, the Build requires less as the burden of running a service is reduced, and the Buy requires the least where minimal development or customization is required.

The Skills Mix reflects the variety of roles, skills and experience required to develop the platform. There is no difference potentially between the Make and Build approach as both require similar levels of expertise. For example, the number of database administrators may reduce between Make and Build but not necessarily the need for one. Ensuring redundancy in the skill sets is reflected in the previous consideration.

The Trend in Development Effort combines the number of people and the time taken to develop the platform. The Make consideration requires the most development effort at the beginning but also during the lifetime of the project to maintain and grow the platform. Some companies adopt an agile approach to level out their development load and permit smaller teams to start with, however in relative terms, the technical burden is greater over a longer period with the Make approach. Build alleviates some of these issues leveraging SaaS products and greatly reduces the long-term support due to the stronger foundations. Buy has minimal development and by definition is likely to have matured outside of this particular implementation.

The Responsiveness refers to how quickly the platform is available and meets all or the majority of the OEDA Requirements. A Make approach is likely to require the longest to establish even with a large team at the beginning due to the nature of software and platform development. A Buy approach is about adopting an existing platform and relatively can be the most responsive i.e. the time between a Purchase Order and platform access can be months and the customisation process is likely to be significantly less than constructing a platform.

The Maintenance Burden reflects the level of effort in skill type and resource to maintain, it is also a function of technical risk. The advantage of leveraging existing open-source technologies is that they are matured by the wider community, whereas bespoke code written for OEDA will only have a single application and therefore more risk. The relative reduction in maintenance burden reflects the relative increase in platform maturity.

Scaling challenges should not be confused with the provision of compute, storage and networking but about how scalable the architecture is. For example, open metadata uses a MySQL database as the Metadata Store. In its default configuration, it is not suitable for Enterprise use as it does not meet the High Availability requirements e.g. if the single node fails, the whole service fails. This can be mitigated with Clustering Technologies such as Percona's XtraDB Cluster which manages a multi-node setup but scaling to increase capacity is a difficult task and may require manual intervention. For the Make approach, there is therefore a significant burden to ensure underlying services are scaled in a controlled manner, whereas in a Build setup, it is possible to get MySQL SaaS from Percona itself. In the Buy approach, there are unlikely to be any scaling issues to consider as the feature is typically built into the platform.

The Tech Stack Cost reflects the cost of the products and services; in the case of the majority of the open-source software, they are free for Commercial Use, whereas for the Buy approach, where all services are provided fully integrated, it is the most expensive. It should be noted that the total cost is a function of all these considerations and will be implementation or project specific. Despite the steep upfront costs some companies utilise the Buy approach as that enables access to a platform quickly, whereas for other organisations the steep initial costs is precisely why they adopt a Make or Build approach as they do not envisage using all the capabilities from an integrated platform provider.

The baseline assessment conducted for technical feasibility was based on the Make approach using an example architecture that met the OEDA Requirements. As stated previously, there is no (obvious) commercial provider for a data sharing platform that meets the OEDS definitions for a catalogue and Data Sharing Fabric. The Completion Confidence reflects to what extent all of the OEDA Requirements will be met. As with most technical projects, the balance between all the considerations is project and implementation specific and cannot be assessed without additional inputs. In this instance, the Buy strategy is utilising analytics platforms that contain data catalogues and therefore will not have been designed for this use case, whereas a Make approach should be able to achieve all of the requirements.

The final consideration is about whether the development of a data sharing platform should be a core activity of the organisation and has an impact across all of the considerations. An organisation dedicated to the extraction of hydrocarbons may not be suited to provide the human resources support for a data team, working practices and culture. This will manifest itself as difficulty in recruitment and retention. Stakeholders in the organisation may struggle to understand the technical risk given that it is a different field and therefore provide the right support.

For some organisations, adopting or growing a digital capability can be seen as a distraction and therefore to retain organisational focus may choose to adopt the Buy approach despite cost concerns. Given the time taken to Make or Build a platform, it requires considerable organisational backing and therefore could be seen as a distraction. For an organisation with a rich data ecosystem, the impact will be minimal and they may prefer the Make or Build approach to better integrate with their existing systems and reduce overall costs.

As the discussion illustrates, there is no "right" answer and much like the purchase of furniture, it is dependent on other factors outside of pure technical considerations. Nonetheless, some key considerations have been highlighted and should be incorporated into the formal platform selection process.

# 10.0

# Conclusion

An example technical architecture that meets the proposed OEDA Requirements was created and demonstrates that OEDA is not only technically feasible but the technology base is mature, has a range of potential suppliers and investment in this area would facilitate key skills in the wider Digital economy. A number of permutations were recognised in Deployment environments (On-Prem, Cloud & Hybrid) approaches (Bare Metal, Virtualized, Containerised & Serverless) and Implementation Strategies (Make, Build & Buy) and considered in concluding that the associated confidence level was high. In the opinion of InDHu, this is similar to the UK Government's definition of Technology Readiness Level (TRL) of 6.

# 11.0

# Appendix A: OEDA Requirements

The two tables are an extract from OEDA Report 1 - Data Sharing Landscape:

| Req. ID | Requirement | Source(s) |
|---|---|---|
| E1 | OEDA shall support the OEDS defined Data Catalogue. | From Action 2.1: **Offshore Energy Data Catalogue** (OEDC). |
| E2 | OEDA shall support the OEDS defined **Data Sharing Fabric**. | From Action 2.2: **Data Sharing Fabric** (DSF). |
| E3 | OEDA shall be based on open-source software and open standards. It should facilitate the **Presumed Open** principle. | The principle of being as "Open as possible" as expanded in the EDiT[32] report as: "Wherever possible, it is proposed that these should be based on open-source software, open data licences and open standards" |
| E4 | OEDA shall support a customisable set of attributes to act as metadata and have the means to define differing levels of priorities and controls. | Several **metadata** attributes have been defined, in effect the superset from **Ice Breaker One** on **Open Net Zero**[33], **EDVP**[34] and **Dublin Core**[35] but recognising the need to set and control differing priorities. |
| E5 | OEDA shall support external URL redirects, HTTP based APIs, the means to redirect to static files and other protocols to support streaming applications. | The ONS **Energy Data Visibility** project stated the protocols initially should be HTTP based, but recognised with maturity it should support streaming applications. |
| E6 | OEDA shall support metrics regarding the data. | The EDVP identified the need to surface and measure data quality - the subjective component in assessing data quality will be influenced by existing Industry standards-based initiatives. The implication is users manually submitting feedback. |
| E7 | OEDA shall support means for prioritising data sets, either for release, update or additional context. | Multiple reports including EDVP and EDTF cited a two-phase approach to data sharing, where users can see a list of potential sources and request them. These are then prioritised for release based on requests received. |
| E8 | OEDA shall support a mechanism to enable users to provide direct feedback to data providers. | Multiple reports have cited providing feedback between users and Data Providers, the former to help improve the data sources and the latter to support internal business cases. |
| E9 | OEDA shall display lineage or provide the means to define a lineage between datasets. OEDA shall support datasets to be related using attributes. | EDVP also identified the need to establish both Data Provider led and user driven relationship mapping between datasets. |
| E10 | OEDA shall support and maintain support for the highest security standards in the field of **Authentication**, **Authorisation** and **Zero Trust** (including defence in depth). | OEDS report states in Action 3.2 **Cyber Security**: "The offshore energy sector should continue to prioritise cyber security, adhering to cyber security best practice and disseminate progress to the wider sector to help developing industries." |

Table 3: Technical Requirements derived from the Energy Sector

[32] Energy Systems Catapult (2022) - Delivering a Digitalised Energy System
[33] Icebreaker One & Open Net Zero (2023) - Open Net Zero by Icebreaker One
[34] Hippo Digital (2020) - Energy Data Visibility [Discovery report]
[35] Dublin Core Metadata Initiative (2023) - DublinCore

| Req. ID | Requirement | Source(s) |
|---|---|---|
| D1 | OEDA shall support the use of internal and external repositories for dataset documentation, context, data samples, API definitions and other assets. | Data Industry expectations around open source software development and documentation culture. |
| D2 | OEDA shall support the use of long held security tokens including but not limited to client and server-side certificates - mutual Transport Layer Security (mTLS) with Hardware Security Modules (HSM) and/or rotated authentication tokens (i.e., OAuth 2.0 / OIDC). | Recommendations from wider energy sector reports are tilted towards human interaction. The OEDS report explicitly states the use of machine-to-machine interactions. The data industry expects the use of standard protocols and approaches. |
| D3 | OEDA shall support data profiling for machine readable formats and support the hosting of sample data for user preview. | Data industry expectations for data format, structure and size are required prior to previewing the data - particularly important for larger datasets. |
| D4 | OEDA shall support rich formatting of content. | The open-source development culture also provides rich documentation around a project that users can collaborate on, which can also be hosted externally. |
| D5 | OEDA shall support the exploration of data with either internal or external platforms. | Kaggle has demonstrated that users prefer to make their own assessments of the data rather than rely on data provider attributes. This includes the principle of the data being Open to Explore, either externally much like the Python Data ecosystem with Binder or internally through hosted Jupyter computational notebooks. |
| D6 | OEDA shall have the means to support a variety of metadata formats (beyond the current attribute-oriented needs). | Data Industry expectations for data format, structure and size are required prior to previewing the data - particularly important for larger datasets. |

Table 4: Proposed Requirements from the Data Industry

Net Zero Technology Centre

Technology Driving Transition

**Contact number:**
+44 (0)1224 063200

**Media enquiries:**
pressoffice@netzerotc.com

**Net Zero Technology Centre**
20 Queens Road, Aberdeen AB15 4ZT

www.netzerotc.com