Persistent Personal Data Vaults Empowering a Secure and Privacy Preserving Data Storage, Analysis, Sharing and Monetisation Platform

D5.4
DataVaults Platform - Beta Version (full functional beta)

Editor(s)	Konstantinos Oikonomou, Giannis Ledakis
Lead Beneficiary	UBITECH
Status	Final
Version	1.0
Due Date	31/12/2021
Delivery Date	30/12/2021
Dissemination Level	PU

DataVaults is a project co-funded by the European Commission under the Horizon 2020 Programme (H2020-ICT-2019-2) under Grant Agreement No. 871755 and is contributing to the BDV-PPP of the European Commission.

Project	DataVaults – 871755
Work Package	WP5 - DataVaults Platform Continuous Integration
Deliverable	D5.4 - DataVaults Platform -Beta Version (full functional beta)
Editor(s)	UBITECH - Konstantinos Oikonomou
Contributor(s)	Ubitech – Giannis Ledakis Ubitech – Konstantinos Oikonomou Tecnalia – María Jose Lopez Osa ATOS – Miguel Angel Mateo Montero ATOS – Javier Villazan ATOS – Raquel Cortes ATOS – Ivan Martinez Suite5 – Chryssovalanto Kousseti Suite5 – Katerina Zerva ASSENTIAN – Ilesh Dattani Fraunhofer – Yury Glikman Fraunhofer – Simon Dutkowski Fraunhofer – Torben Jastrow PRATO – Elena Palmisano PRATO – Paolo Boscolo
Reviewer(s)	ATOS - Miguel Angel Mateo Montero Fraunhofer – Yury Glikman

Abstract	This document is a deliverable of WP5 and describes the beta release of the DataVaults platform and tools, with supporting documentation. Each component of the platform is assessed regarding its status, its integration to the platform and its coed availability and installation details.

Executive Summary

This document presents the Beta Version of the DataVaults platform. The Beta Version of DataVaults is a major step towards a fully functional version that includes enhanced versions of the platform's backbone services and the UI, provided for further assessment. In this version, integration between the various components developed in the Alpha version continued and the capabilities offered by the platform were extended as a result. Additionally, this document provides the status of the platform components in this Beta Version developed in WP3, WP4 and WP5, as well as an analysis on the methodology phases of the DataVaults platform. As a result of the improvements implemented for the Beta Version, more of those phases and actions are now considered implemented and integrated. Finally, based on the development and integration that happened so far, the document includes a detailed overview of the usage of the platform and also the plan for the subsequent releases.

Table of Contents

List of Figures

Terms and Abbreviations

ABE	Attribute Based Encryption
API	Application Programming Interface
CO	Confidential
D/E	Data Explorer
DCL	DataVaults Compensation Lifecycle
DDAL	DataVaults Data Analytics Lifecycle
DDML	DataVaults Data Management Lifecycle
DLT	Distributed Ledger Technology
GUI	Graphical User Interface
MVP	Minimum Valuable Platform
NFC	Near Field Communication
Q/B	Query Builder
SEAS	Secure Analytics System
SQL	Structured Query Language
SSE	Symmetric Searchable Encryption
TPM	Trusted Platform Module
VM	Virtual Machine
UI	User Interface

1 INTRODUCTION

DataVaults aims to deliver a novel framework and architecture that leverages personal data, coming from diverse sources to help Individuals construct their unified personal data hub, collect at a single point all of their personal data in a secure and trusted manner, and retain ownership and control on what to share and with whom, also receiving compensation for the artefacts they place at the disposal of other third parties.

This document describes the beta release of the DataVaults platform, while it also provides the updated architecture, the APIs, the user stories and the plan for the next releases. It relies on the previously collected technical requirements and user stories presented in the deliverable D5.1 [1] and the architecture defined in D5.2 [2] and builds on the platform release described in D5.3 [3]. It also relies on the work performed in work packages WP3 and WP4, where the technical details of the components have been provided.

The work towards the platform development and integration is performed in four iterations in total, each of them supported by documentation and a short progress report covering needed updates. The upcoming releases will be published in the deliverables D5.5 and D5.6.

1.1 DOCUMENT STRUCTURE

The document is structured as follows:

Section 1 is the introduction, this document structure description and a presentation of the methodology followed towards the delivery of the Beta Version.

Section 2 provides the status of the Beta Version of the DataVaults platform and more specifically the status of each component. The status is analysed regarding the integration of the component with the platform, as well as its code availability and installation instructions. This Section also includes the description of the DataVaults methodology phases as part of the Beta Version and an analysis on the project's lifecycle.

Section 3 provides a brief description and screenshots of most of the UI pages implemented for the Cloud Platform, the Personal App and the Secure Analytics Playground components.

Section 4 provides the plan and the features to be implemented in the two upcoming releases of the platform.

Finally, section 5 concludes the document while section 6 collects the references.

2 DATAVAULTS BETA VERSION STATUS

2.1 OVERVIEW

As mentioned already in D5.3 [3] the DataVaults Platform is split into two main groups of components, the DataVaults Cloud Platform and the DataVaults Personal App, both of which have implemented major changes moving into the beta version. In order to have a clearer look for the individual components that compose those 2 groups, each component's status, integration, code availability and installation instructions will be separately reported in the following sections. Regarding code availability, all repositories mentioned are private repositories that project members have access to and access can also be granted to individuals outside of the consortium upon request.

2.1.1 DataVaults Cloud Platform

The DataVaults Cloud Platform is a cloud service offering a single-entry point for Data Seekers. All the components that are a part of it will be reported in the subsections below, in order to collectively form the status of the Cloud Platform as a whole.

2.1.1.1 Cloud Platform Backbone

- **Status and integration:** The Cloud Platform Backbone currently offers Data Seekers the functionality to search over available datasets, filter them using various criteria and buy the ones they are interested in. Once bought, those datasets are available in their personal "My Vault" page and can be downloaded from there. Furthermore, Data Seekers can create and share questionnaires, as well as browse a record of all the transactions they have performed. The Cloud Platform Backbone is connected with the Personal App through various APIs and a message queue in order to communicate the necessary information regarding asset management, sharing and purchase. It is also connected to the Trusted DLT Engine and the Public and Private Ledgers, Cloud Platform Data Store and Data Stream & Contract Composer components in order to securely store the shared assets and necessary user information. Finally, integration with the Access Policy Engine and the Query Builder ensures that Data Seekers can search and have access only to assets that match their profile.
- **Code Availability and Installation instructions:** The code both the backend and frontend of the Cloud Platform Backbone is available at https://gitlab.ubitech.eu/cs3/datavaults_cloud_platform), along with a comprehensive README file. Installation instructions are included therein and the components have already been dockerized in order to further facilitate an easier installation process.

2.1.1.2 Persona Generator

- **Status and integration:** The core methods to support Persona Creation have been implemented as detailed further in D3.1 [4] and D3.2 [5]. The Data Interpreter supporting data transformation from the JSON to the Pandas DataFrame and the Clustering Method and the Data Insight Method both of which form the core analytical

processes for the persona creation have been integrated. The above-mentioned components are not as yet integrated into the complete Persona Generator Workflow as provided in D3.2 [5] (Section 3.5.1 Figure 25).

- **Code Availability and Installation instructions:** There is currently no integrated version of a persona generator in the beta version of the DataVaults platform

2.1.1.3 Access Policy Engine

- **Status and integration:** The Access Policy Engine in its current version establish if a specific data seeker or company is allowed to access the data in which they are interested. The attributes related to the data seeker are taken from the profiles repository and the access policies from the ledge where they are stored as part of the data owner sharing configuration files. This module is integrated as part of the cloud platform and it is called via API for the Query Builder. Also, the Engine uses APIs to obtain the attributes from the Keycloak system and the access policies from the blockchain.
- **Code Availability and Installation instructions:** The code corresponding to this version is available in Datavaults gitlab (https://gitlab.com/datavaults/personalapp/policies-editor/engine-api.git). The installation is described in this README file (https://gitlab.com/datavaults/personalapp/policies-editor/engine-api/-/blob/main/README.md), allowing the integration as a docker element.

2.1.1.4 Risk Management Monitor

- **Status and integration:** The Risk Management Monitor provides an estimate of the estimate the impact that derives from the interconnections of the assets and datasets and provide a single estimation for the overall impact of a specific asset/vulnerability combination. It is currently not integrated with the Cloud Platform Backbone but will be in the coming versions.
- **Code Availability and Installation instructions:** The code of the Risk Management Monitor and its installation instructions will be integrated to the Cloud Platform backbone.

2.1.1.5 Secure Analytics Playground

- **Status and integration:** The Secure Analytics Playground, named as SEAS, offers a way to share, extract and enrich data using different AI algorithms inside a secure platform. Data Seekers and Data Providers are the target people for the use of this component. SEAS is divided into two subcomponents, "Service Analytics Host" and "Playground & Visualization Host". While the first will define the different features of the Playground configuration and setup, the second subcomponent will allow the user to deploy and execute Machine Learning algorithms that can later be viewed or stored. The graphical user interface and the logic behind the "Service Analytics Host" and the "Playground & Visualization Host", are already deployed. As well, the connection between the different components of the SEAS is already running. The connection between the SEAS and the Data Explorer is the next step that will be defined in the future months.
- **Code Availability and Installation instructions:** The URL where the code of the repository is hosted is: https://scm.atosresearch.eu/ari/bdapc/datavaults/toreador-

frontend.git (this is a closed repository; the source code will be made publicly available with next release V0.5). Detailed installation instructions can be found in the Appendix.

2.1.1.6 ABE/SSE Engine

- **Status and integration:** ABE engine is composed of three modules, namely PolicyMng_ABEEngine, EncryptorDecryptorSrv and KeyManager_ABEEngine. This component is currently under unit testing, to this end all the modules are integrated into a single service in order to perform the testing isolating the behavior of crypto primitives from communication issues. Next steps of this module are communication testing between the three modules and integration with SSE engine.
- **Code Availability and Installation instructions:** The URL where the code of the repository is hosted is https://scm.atosresearch.eu/ari/datavaults_ip_dev (this is a closed repository; the source code will be made publicly available with next release V0.5). Due to the status of developments there is no deployable version for this component.

2.1.1.7 Data Stream & Contract Composer

- **Status and integration:** The DataStream and Contract Composer is responsible for enabling the data sharing actions from the side of the Data Seekers, whether these have to do with the purchasing of assets already made available over the platform, or with assets that are requested directly from data owners (could be either datasets, or questionnaires). As such, this component allows data seeker to either view and get data assets after buying them, or to construct data requests and define questionnaires. The complete component has been implemented already.
- **Code Availability and Installation instructions:** The code of the Data Stream and Contract Composer has been integrated to the Cloud Platform backbone.

2.1.1.8 Trusted DLT Engine and the Public and Private Ledgers

- **Status and integration:** All features regarding the secure logging of all data trading transactions have been successfully implemented, along with support for executing queries for retrieving data from the deployed contracts. Finally, the execution of more complex ledger queries regarding data trading transactions have been finalized.
- **Code Availability and Installation instructions:** The URL where the code of the repository is hosted is https://gitlab.com/datavaults/blockchain-dlt-engine, where also detailed installation instructions can be found in the README file.

2.1.1.9 Query Builder

- **Status and integration:** The query builder is fully functional. The component consists of the triple store and a convenient API, allowing clients easy access to the repository and offers comprehensive search and query capabilities based on an RDF metadata information format. Whereas the metadata is described in RDF, the API partially allows query and search requests expressed as JSON. Beside the query and search capabilities, the API provides a complete assets and profile metadata part.
- **Code Availability and Installation instructions:** The code for the query builder is available at https://gitlab.com/datavaults/personalapp/data-fetcher-transformer. The repository contains README instructions for building and packaging the

component. It contains further a Dockerfile to build a docker image and a docker-compose deployment description in order to run it in a container environment together with the triple store, which is available as official docker image.

2.1.1.10 Cloud Platform Data Store

- **Status and integration:** The Cloud Platform Data Store is fully functional in the beta version of the platform and stores all the necessary user and asset information for the seamless operation of the platform. It is connected only to the Cloud Platform Backbone through a database connection.
- **Code Availability and Installation instructions:** The Cloud Platform Data Store code and installation is done in conjunction with the Cloud Platform Backbone, as can be seen in the relevant Section 2.1.1.1 .

2.1.1.11 Data Explorer

- **Status and integration:** The Data Explorer has been integrated with the Cloud Platform backbone and has been incorporated in the "MyVault" page of the platform, in order to allow the dataseeker to easily browse through and download their owned datasets and download them.
- **Code Availability and Installation instructions:** The Data Explorer code and installation is done in conjunction with the Cloud Platform Backbone, as can be seen in the relevant Section 2.1.1.1 .

2.1.2 Personal DataVaults App

The Personal DataVaults App is at the moment offered as a cloud-based application used by Individuals to operate and exchange data seamlessly. All the components that are a part of it will be reported in the subsections below, in order to collectively form the status of the Personal App as a whole.

2.1.2.1 Personal App Backbone

- **Status and integration:** The Personal App backbone is the main application that is used by data owners to collect and share their data, and is housing all the necessary components that are offered to these users. It is communicating with the DataVaults Cloud platform through various APIs while messaging communication is done via RabbitMQ. As far as it regards the other components that are part of the Personal App, communication is there also facilitated via APIs, connecting the different dockerized components. A fully functional version of the Personal App has been delivered in the beta release, and updated in the future in the backbone will focus on debugging and on the connection of new components that are planned for the next releases
- **Code Availability and Installation instructions:** The code for the Personal App Backbone backend is available at: https://gitlab.com/datavaults/personal-app-backend while the frontend is at: https://gitlab.com/datavaults/personal-app-frontend, along with a comprehensive README file. Installation instructions are included therein and the components has already been dockerized in order to further facilitate an easier installation process

2.1.2.2 Data Fetcher & Transformation

- **Status and integration:** The Data Fetcher & Transformer supports the complete workflow for configuring a source and fetching the data for one source. The source can be configured with an optional scheduled interval in which the source should be harvested. As the access to the data needs authentication information, this can also be provided when configuring the source. This schedule can be paused or completely deleted. The Data Fetcher & Transformer is connected with the Personal App Backbone through various APIs and a message queue in order to communicate the configuration of a source and the result of the data fetching and transformation back to the Personal App Backbone.
- **Code Availability and Installation instructions:** The Code for the Data Fetcher & Transformer is available at https://gitlab.com/datavaults/personalapp/data-fetcher-transformer. All services are dockerized to enable an easy installation process. Installation instructions can be found in the Gitlab Repository.

2.1.2.3 Private Wallet

- **Status and integration:** This component is under development so integration is still pending, with the main steps being integration with the private ledger for the management of compensations and integration with the INFINEON Blockchain starter kit for the management of the crypto tokens.
- **Code Availability and Installation instructions:** This component is formed by the wallet service (back-end and front-end) and several external modules to provide the required functionality and support wallet operations like exchange services for the exchange of compensations to points, audibility and anti-fraud functionality, as well as merchant services which provide basic e-shop functionalities, where users can make use of their points to get products and services. The code is hosted in the repository at https://scm.atosresearch.eu/ari/datavaults_ip_dev (this is a closed repository; the source code will be made publicly available with next release V0.5).

2.1.2.4 Sharing Configurator

- **Status and integration:** The Sharing Configurator component acts as the main component to be used by data owners to share their data, incorporating the functionalities offered by the Anonymizer, the Access Policy Editor and the Privacy Metrics Dashboard. In the beta version, there is full integration with the Anonymiser and the Access Policy Editor, and work in progress is underway for integrating the Privacy Metrics Dashboard and the DAA Engine in the future releases of the platform.
- **Code Availability and Installation instructions:** The code for the Sharing configurator has been integrated to the Personal App backbone.

2.1.2.5 Privacy Metrics Dashboard

- **Status and integration:** The Privacy Metrics Dashboard provides the user the calculated risk regarding the datasets that the user has added to the platform, and also a comprehensive view of current and previous privacy exposure degrees. It has been integrated with the Personal App in order to appear in the preview of a dataset sharing procedure.

- **Code Availability and Installation instructions:** The code is available as part of the Personal app Frontend & Backend and the instructions for the installation are described in the corresponding README files of each gitlab project.

2.1.2.6 Anonymiser

- **Status and integration:** The anonymiser has been implemented and integrated into the current version of the DataVaults platform, with details being provided in D3.1 [4] and D3.2 [5]. The current version supports anonymisation of datasets, obfuscating personally identifiable information, generation of PseudoIDs and support for the capability for Data Providers to configure the level and type of anonymisation applied to their data. The anonymiser is integrated with the sharing configurator from which the anonymiser is called within DataVaults and to which the anonymised data is sent.
- **Code Availability and Installation instructions:** The code is available on GitLab in the link https://gitlab.com/datavaults/personalapp/anonymiser. It can be installed through the use of the docker file in the anonymiser-api-v001 folder in the DataVaults Gitlab repository. The docker file will set up a container running the anonymiser and associated API.

2.1.2.7 Access Policy Editor

- **Status and integration:** The Access Policy Editor is developed as an element of the Personal DataVaults App. The Editor allows the Data Owners to select the valid attributes under which sharing their data. A simple representation of those attributes is presented to them for selecting the desired values and save them as part of the sharing configurator file. The current version does not include lists of values nor reusable policies.
- **Code Availability and Installation instructions:** The code is available as part of the Personal app Frontend & Backend and the instructions for the installation are described in the corresponding README files of each gitlab project.

2.1.2.8 Edge Analytics Engine

- **Status and integration:** The Edge Analytics Engine is a component that is used to deliver simple, pre-defined analytics to data owners, based on some data sources that retrieve data. The integration of the component will be done via APIs to the Personal App backbone, as the analytics engine will be dockerized including the necessary tech stack such as Pandas and scikit-learn. The delivery of the engine is expected to be done in the v0.5 release of the platform
- **Code Availability and Installation instructions:** Code and installation instructions are not currently available as the component will be delivered for the v0.5 release of the platform.

2.1.2.9 Data Request Service Resolver

- **Status and integration:** The Data Request Service Resolver is tasked to capture the requests coming from data seekers and let data owners decide how they handle those. Communication with the relevant component of the cloud platform is done via RabbitMQ, and the full version of this component is delivered with the beta release

- **Code Availability and Installation instructions:** The code for the Sharing configurator has been integrated to the Personal App backbone

2.2 INTEGRATED PLATFORM STATUS

In the Beta Version of the platform, we tried to improve the functionality and performance of the platform services and at the same time to continue the development and enrichment of the second version of the UI to be used. Regarding the deployment, 11 VMs have been used for this version of the platform; 1 for the identity manager, 1 for the components of the Cloud Platform, 1 for the components of the Personal App, and 8 VMs for the Blockchain Network.

In deliverable D1.3 [6] we presented the workflows that the core offerings of the DataVaults and also the corresponding parts of the DataVaults Lifecycle.

- **DataVaults Data Management Lifecycle**: This part of the Lifecycle concerns all data management-related tasks, spanning from data collection, data cleansing and semantic enrichment, storage and sharing, up to data deletion and access revocation.
- **DataVaults Data Analytics Lifecycle**: It contains the steps required for the exploration of data assets and the application of data analytics and visualisation techniques in order to extract meaningful insights.
- **DataVaults Compensation Lifecycle**: This part of the Lifecycle evolves around the creation and management of contracts for the sharing of data and the appropriate compensation of Individuals.

For this beta release the main focus was to fine-tune the platforms support regarding the integration of actions to properly implement all basic and some more advanced DataVaults Data Management Lifecycle (DDML) actions and phases.
Regarding the DataVaults Data Analytics Lifecycle (DDAL) and the DataVaults Compensation Lifecycle (DCL), setup of the backbone elements has been completed and in this beta version focus has been placed on properly integrating them with the DDML.

In more details, in the *Personal App* part, the Data Fetcher component has been developed and used in order to *retrieve data* from various sources, like Facebook or via file uploads. The data that is returned to the Personal App (Personal App VM, https://app.datavaults.eu) and allows the user to start the *sharing process*. This is done by firstly connecting with the Sharing Configurator component, in order to enable the user to configure various sharing aspects for their datasets, such as the Access policy that will be applied to the dataset. The sharing configurator returns to the Personal App the generated sharing configurator and the user has the option to *anonymize the dataset*. If this option is chosen, the dataset along with the sharing configuration are sent to the Anonymizer component (Personal App VM, https://anonymizer.datavaults.eu). Based on the size of the dataset, the correct algorithm is chosen and the data are anonymized so that they are still usable but maintain the privacy of the user.

Finally, the anonymized dataset, along with the sharing configuration and relevant metadata are sent to the *Cloud Platform* (Cloud VM, https://platform.datavaults.eu) using the

corresponding API, along with a valid Keycloak (https://datavaults-auth.euprojects.net) token of the logged in user.

The Cloud Platform receives the dataset and stores it in the database, along with information about the dataset id and owner. The Platform contacts the DataStream & Contract Composer component which in turn calls the Trusted DLT Engine (https://dlt.datavaults.eu) in order to create a contract representing the sharing of the dataset. Simultaneously, the Query Builder (Cloud VM, https://triple-store.datavaults.eu) component is called to also store the new dataset inside the profile of the data owner.

When a Data Seeker wishes to use the Cloud Platform to *search for datasets* that are available for him, the Cloud Platform contacts the Query Builder component to get a list of the available datasets. In turn, the Query builder calls the Access Policy Engine component in order to determine which datasets should be *accessible to the current user*, based on his/her attributes and the implemented Policy. The Access Policy Engine (pe.datavaults.eu) returns a list of the available datasets, if any exist, and those are presented to the Data Seeker. If the Data Seeker wishes to purchase one of the available datasets, the relevant contract is created via the DataStream & Contract Composer component and the dataset is available for download on the Data Seeker's Vault page. Finally, the Personal App is notified about the success of the transaction via the Messaging queue (212.101.173.179) connected to it and the Cloud Platform.

The Data Seekers can also contact the Secure Analytics Playground Component, via the Cloud Platform, in order to be able to run analytics or algorithms over the datasets they have received access to. This last step has to be performed manually as the integration of the Cloud Platform to the Secure Analytics Playground Component is under implementation.

2.2.1 DataVaults Methodology Phases as Part of the Beta Version

The high-level integrations that may occur between the DataVaults Actors and the DataVaults Platform were also defined in D1.3 [6]. These express the eight Phases of the Methodology, and in this section, we provide a high-level presentation regarding how these Phases are covered in the Beta Version of DataVaults Platform.

The implementation and integration status for the actions that support the DataVaults Data Lifecycle are presented in the figure below using the following color coding: Blue means actions already integrated as part of a single workflow, Green means actions implemented but not integrated in the main platform workflow, orange mean actions are not yet ready.

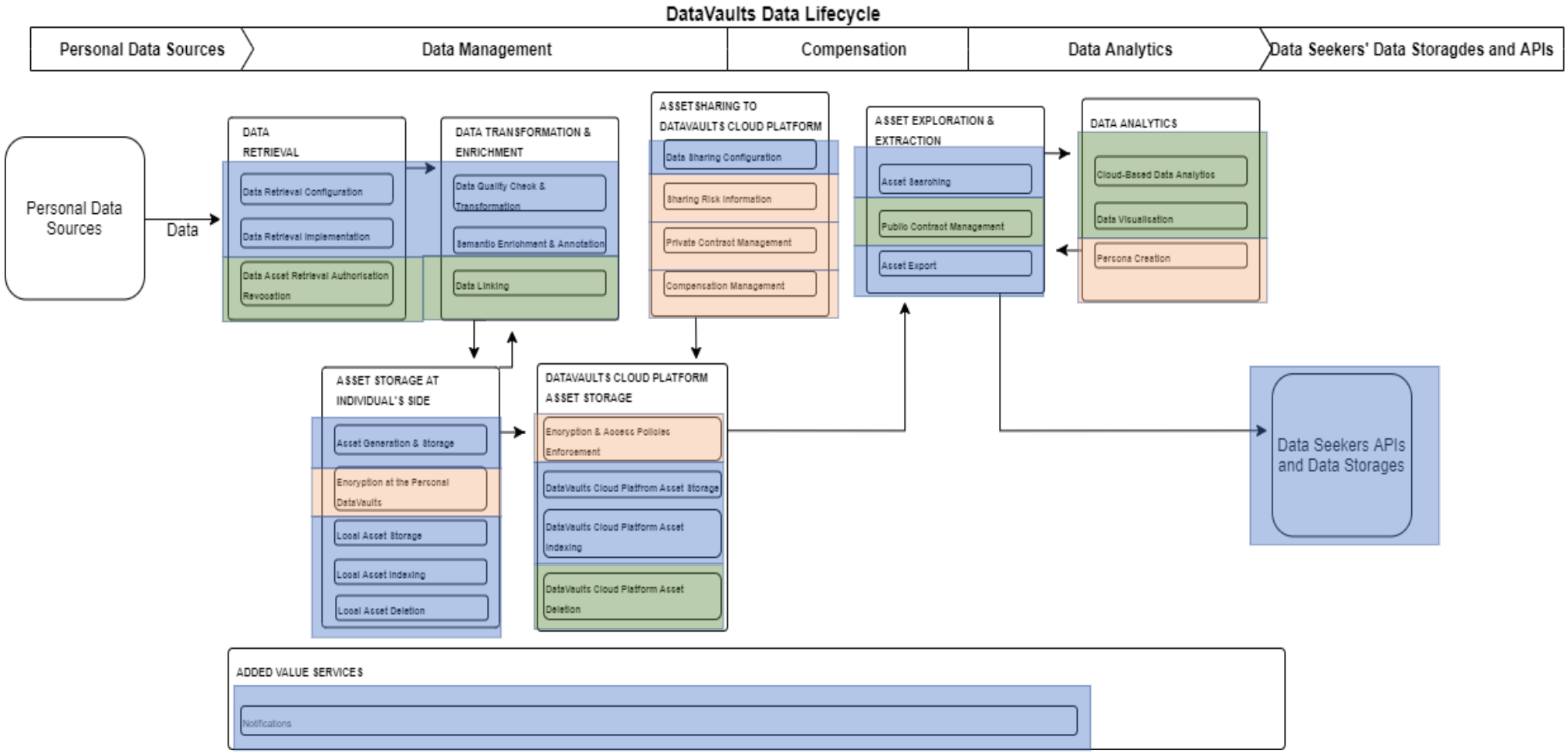


Figure 1: DataVaults Lifecycle as part of the Beta version

For easier readability of the document, we provide the description of the phases, as defined in D1.3 [6].

i. **Data Retrieval** – refers to the configuration, implementation and management of the DataVaults connection to the various data sources, as defined by the Individuals, in order to collect their personal data.
ii. **Data Transformation & Enrichment** - ensures the high quality of the collected data, through automated quality checks and transformation operations. Furthermore, data schema mapping, semantic enrichment and linking processes are foreseen for the maximisation of discoverability and usability of these data.
iii. **Asset Storage at Individual's Side** – is responsible for the persistence of the personal data assets at the DataVaults Personal App of the Individuals in a secure way. These assets include the processed and semantically enriched data collected from the connected data sources, their metadata, and any data assets generated from the application of data analytics by the individual.
iv. **Asset Sharing to DataVaults Cloud Platform** – entails all aspects around the sharing of an Individual's data asset to the DataVaults Cloud Platform. These processes span from the configuration of the various sharing aspects and the creation and management of the corresponding contracts between the Individuals and the Platform, to the continuous update of the Individual's sharing risk information and the activation of the compensation mechanism whenever a data asset is acquired by a Data Seeker.
v. **DataVaults Cloud Platform Asset Storage** – pertains to the actual upload and secure storage of a data asset from the Individual's side to the Cloud Platform, under the terms set during the sharing configuration. Furthermore, this Phase handles a requested deletion of the data asset from the Cloud.
vi. **Asset Exploration & Extraction** – enables the Data Seekers search and acquire data assets that match their needs.
vii. **Data Analytics** – provides cloud-based data analytics and visualisation tools that will shed light on underlying connections and facilitate Data Seekers into getting a better understanding.
viii. **Added Value Services** –includes horizontal DataVaults services that facilitate other core processes.

3 DATAVAULTS BETA PLATFORM USAGE

Both the DataVaults Cloud Platform and the DataVaults Personal App have advanced in the beta version of DataVaults, that supports most phases of the DataVaults Lifecycle. Here we present most interactions that are currently supported through the UI of the Personal App, the Cloud Platform and the Secure Analytics Playground.

In Figure 2, Figure 3 and Figure 4 the view for the unregistered user is provided.

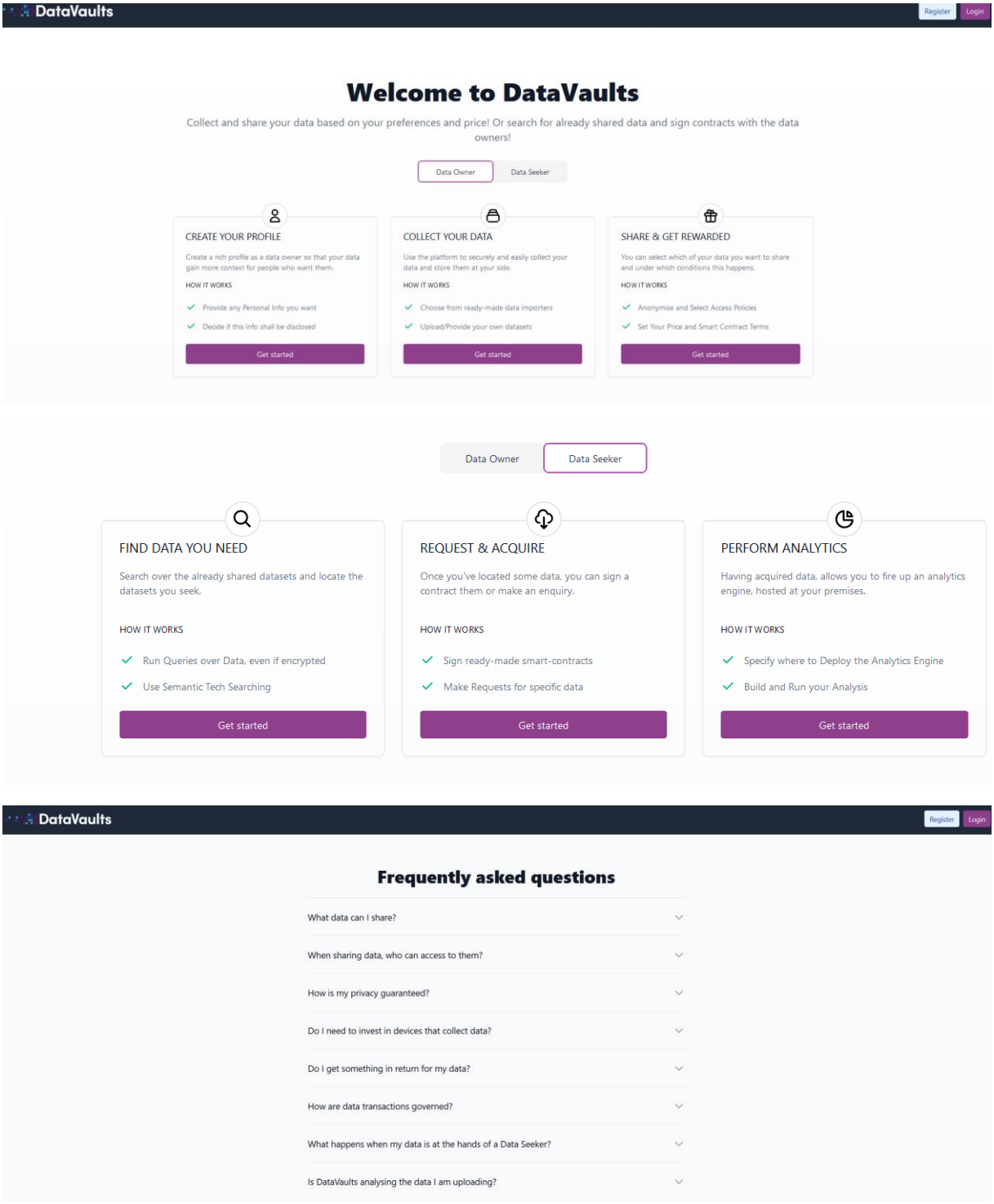


Figure 2: Highlights from Landing Page of DataVaults

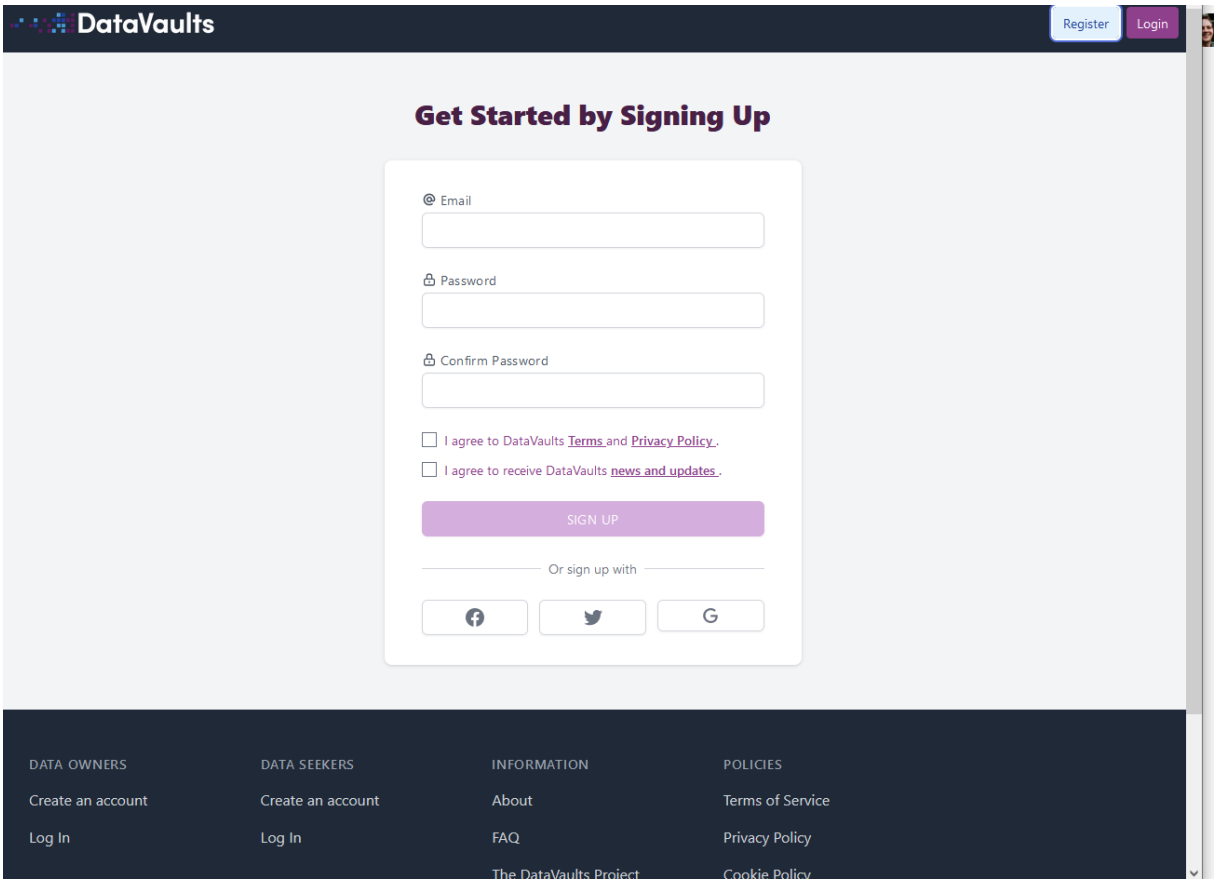


Figure 3: Registration on DataVaults

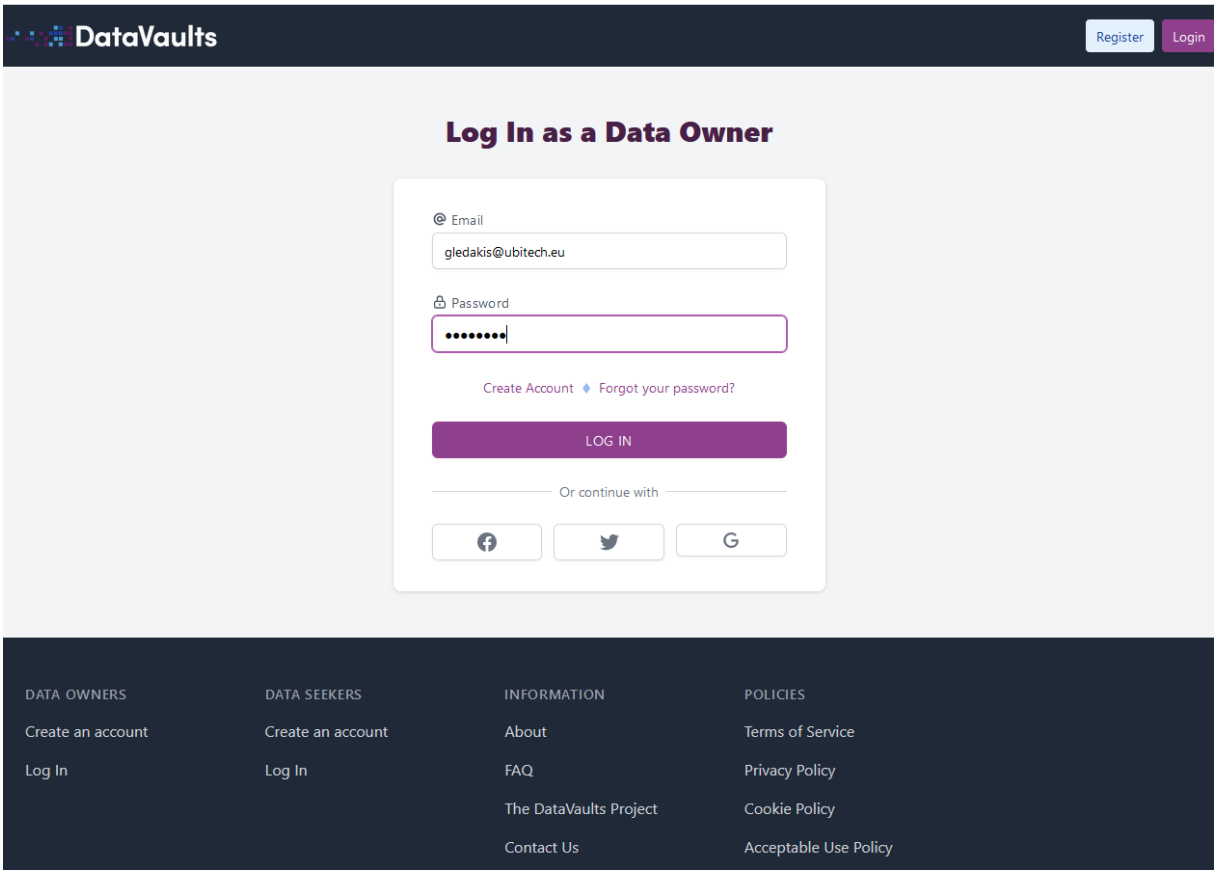


Figure 4: Login on DataVaults

3.1.1 Personal App

In this section, we present the supported functionality for the Personal App. Figure 5 depicts the dashboard of the user, while Figure 6 shows the profile page.

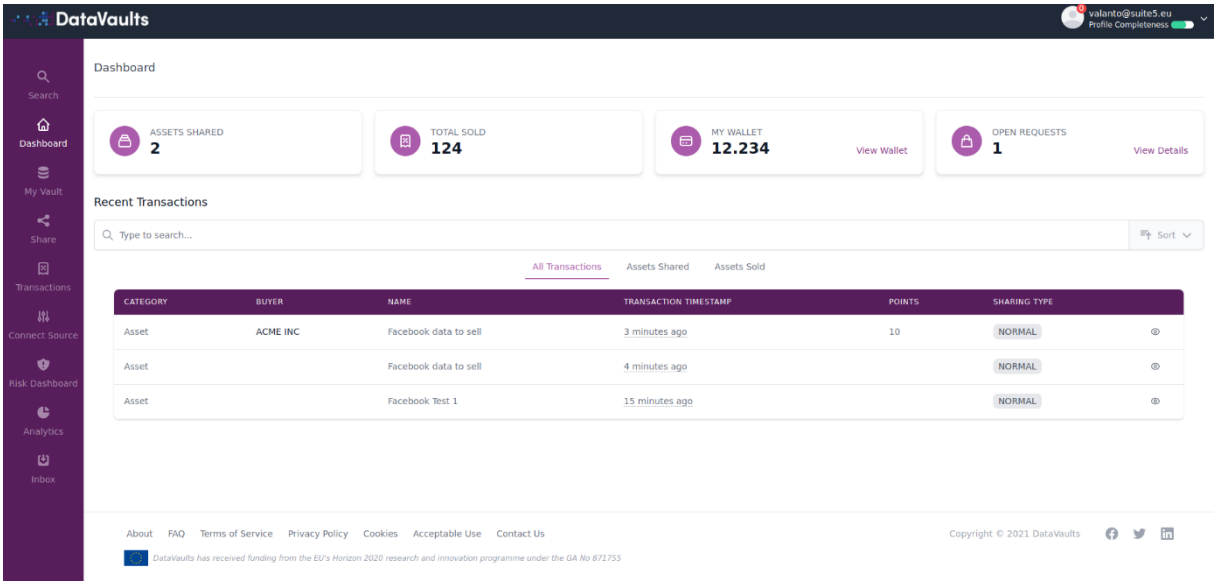


Figure 5: Personal App Dashboard

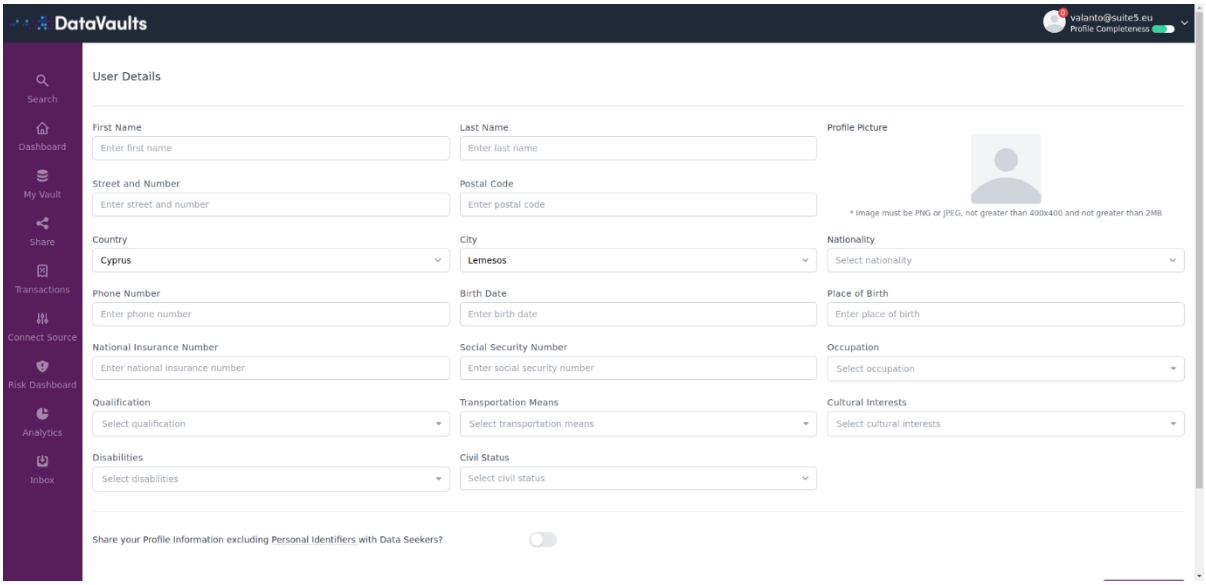


Figure 6: User Profile Page

From Figure 7 to Figure 13 the platform is used by the user in order to connect to a data source and then share to the cloud platform.

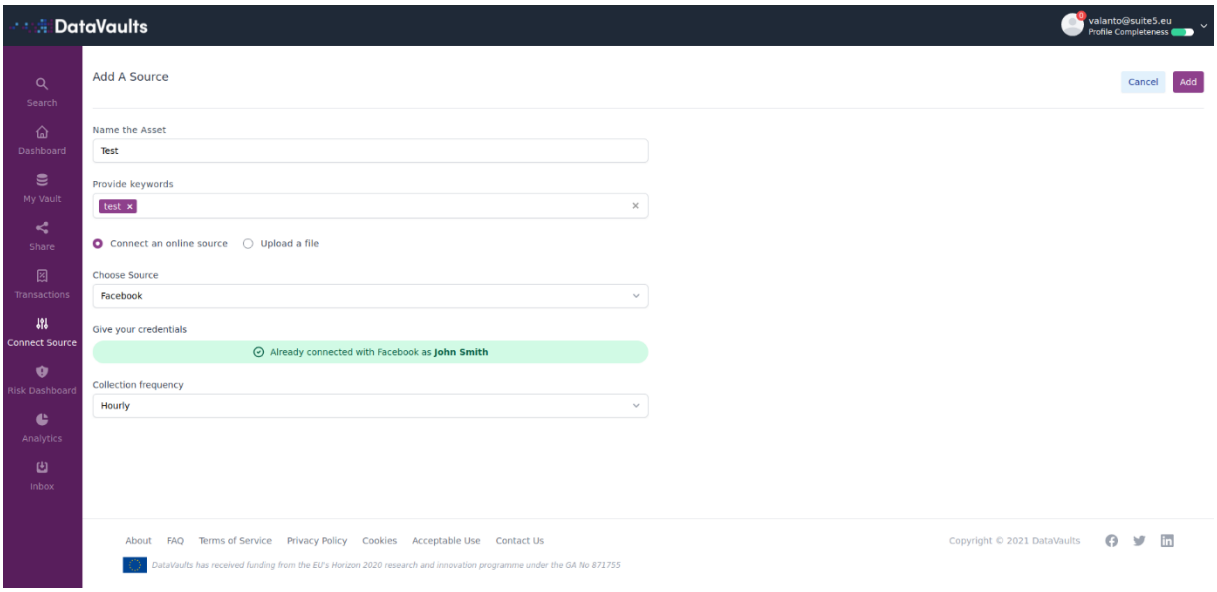


Figure 7: Adding a Data Source

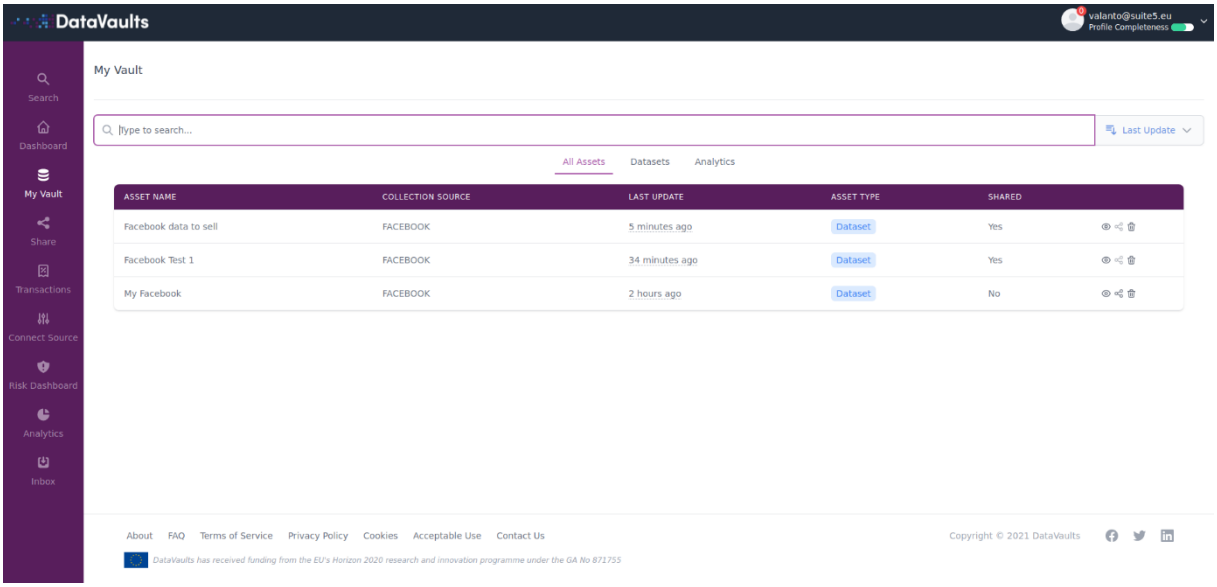


Figure 8: View of User's Vault

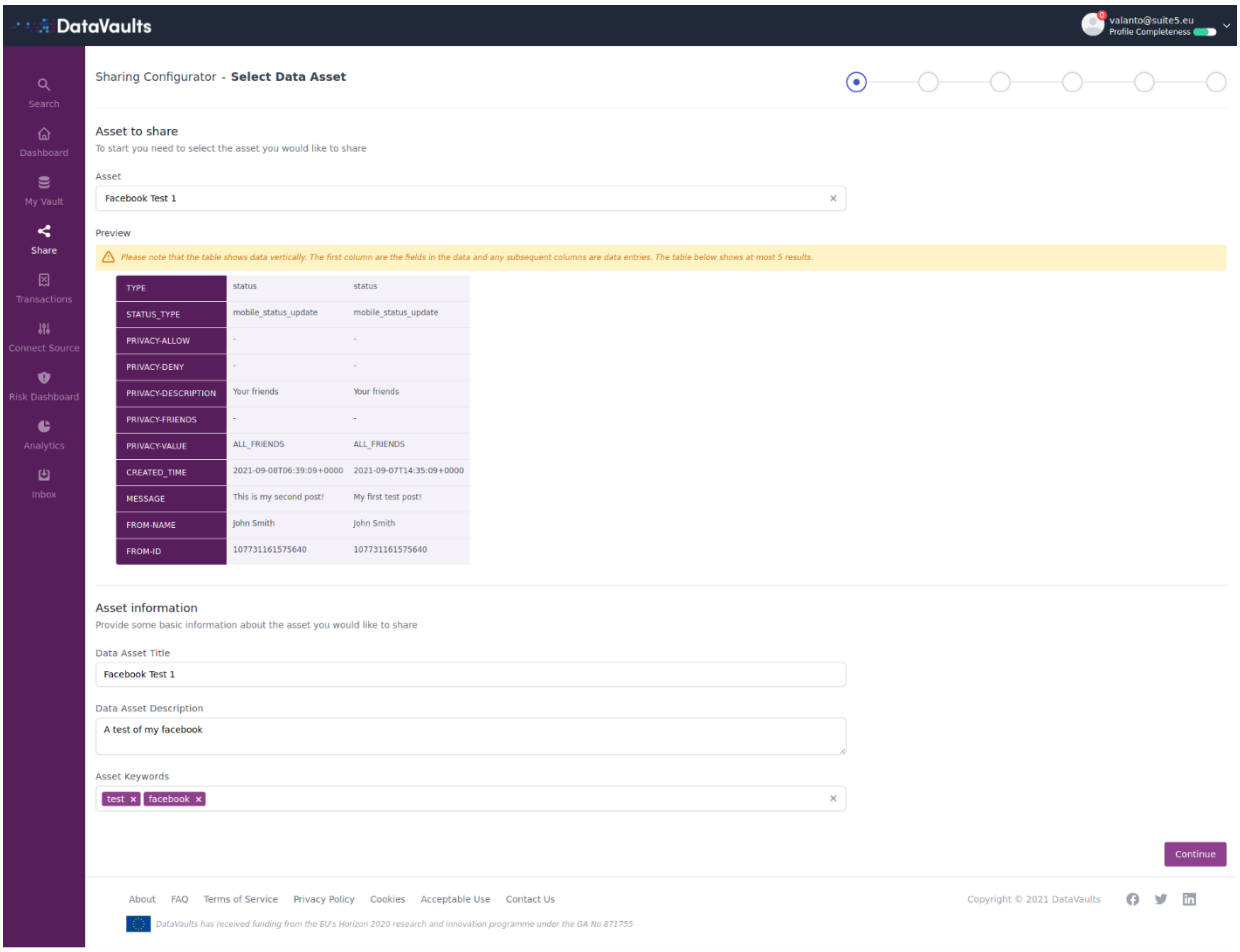


Figure 9: Sharing an asset (step1)

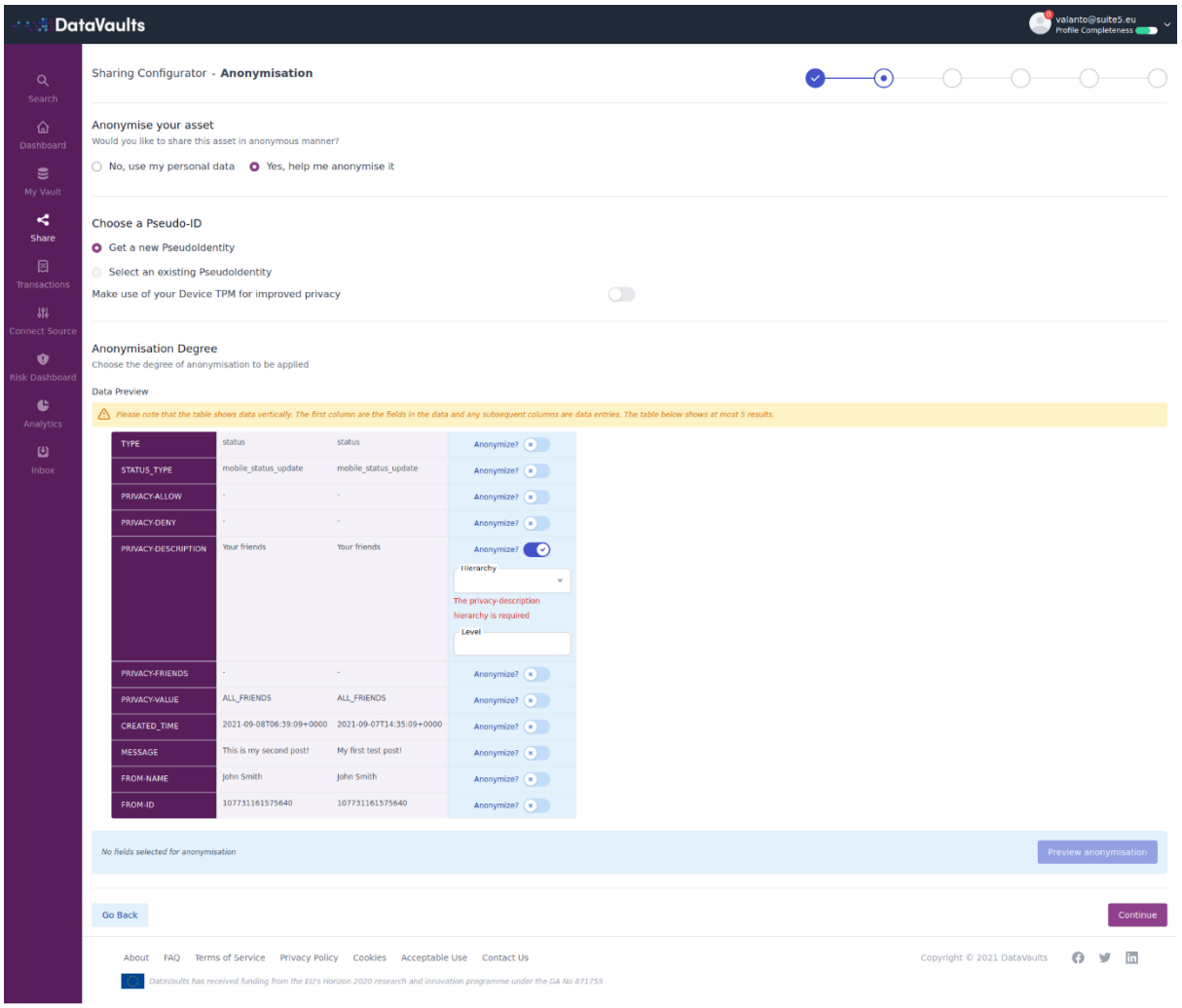


Figure 10: Anonymization while sharing an asset

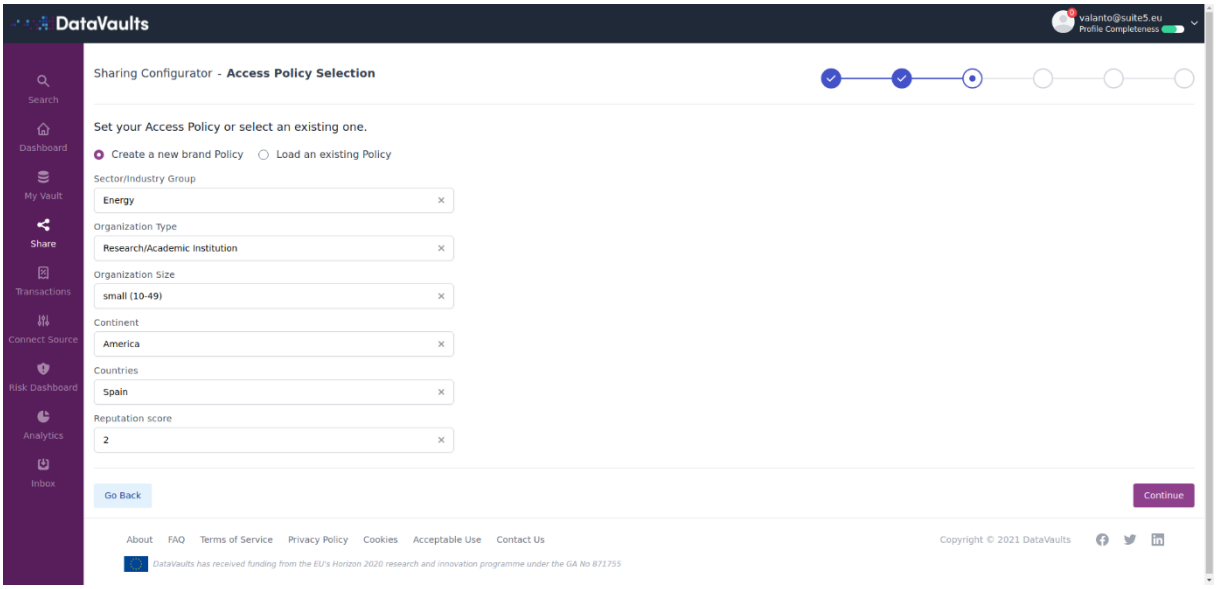


Figure 11: Creation of Access Policy for the sharing of an asset

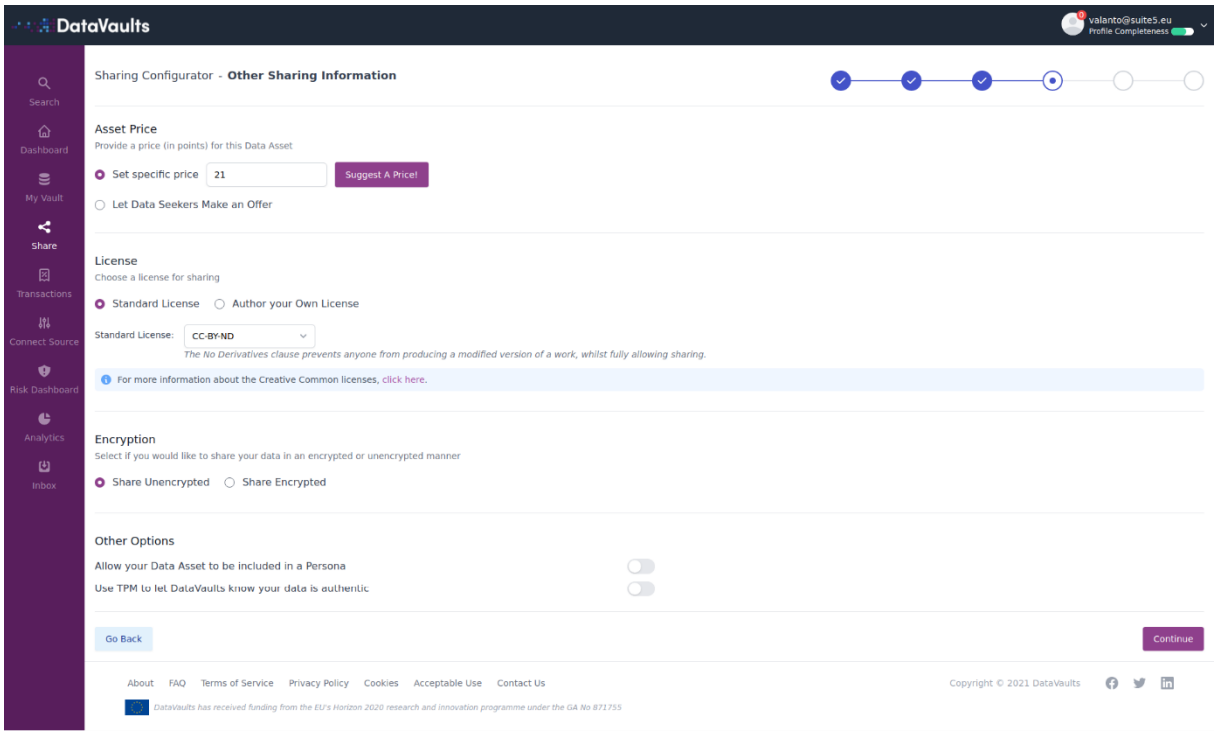


Figure 12: Sharing configuration

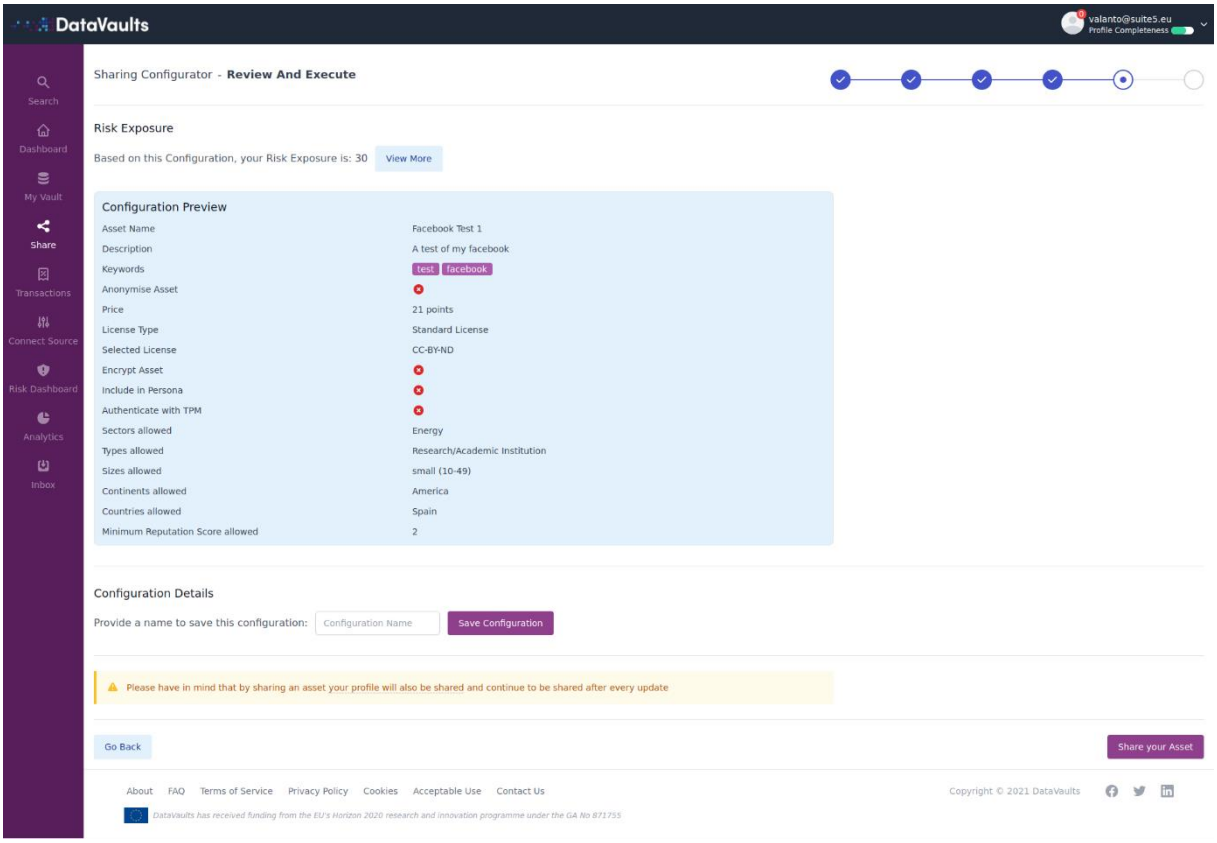


Figure 13: Final Step for Sharing an Asset

In addition to the proactive sharing of a data asset by the User, DataVaults allows a Data Seeker to request for data and also to use questionnaires for the data collection, as shown from Figure 14 to Figure 17.

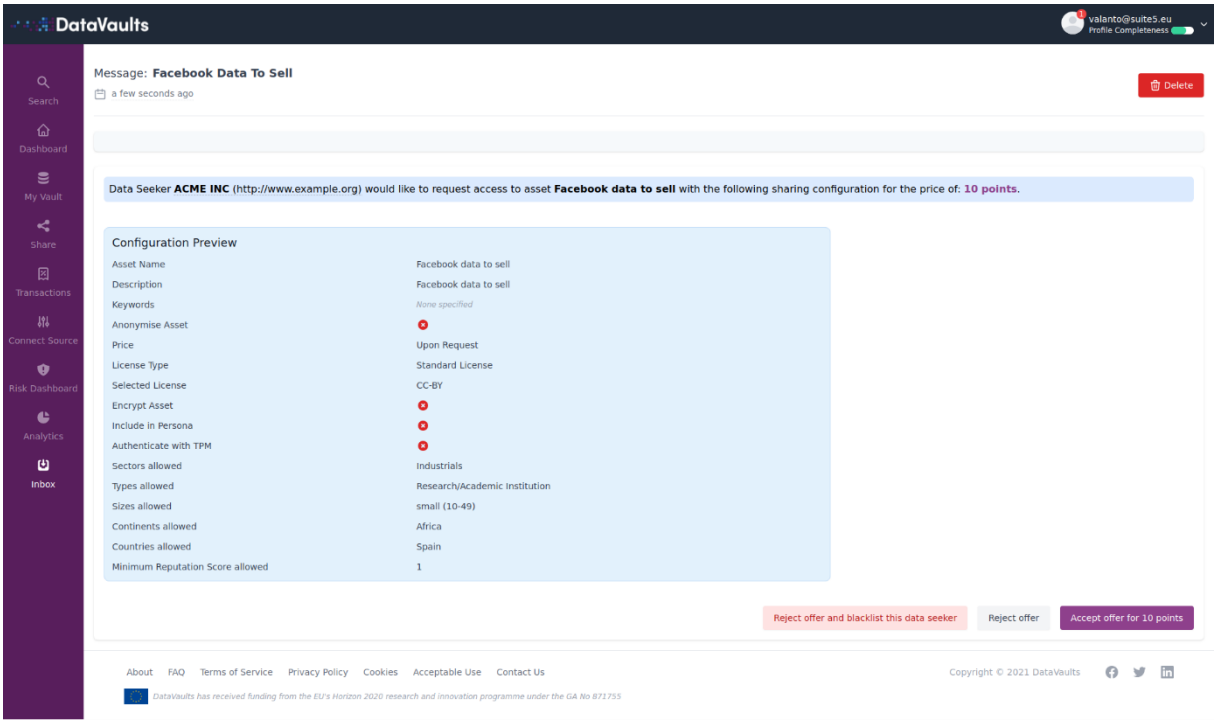


Figure 14: Request for Dataset

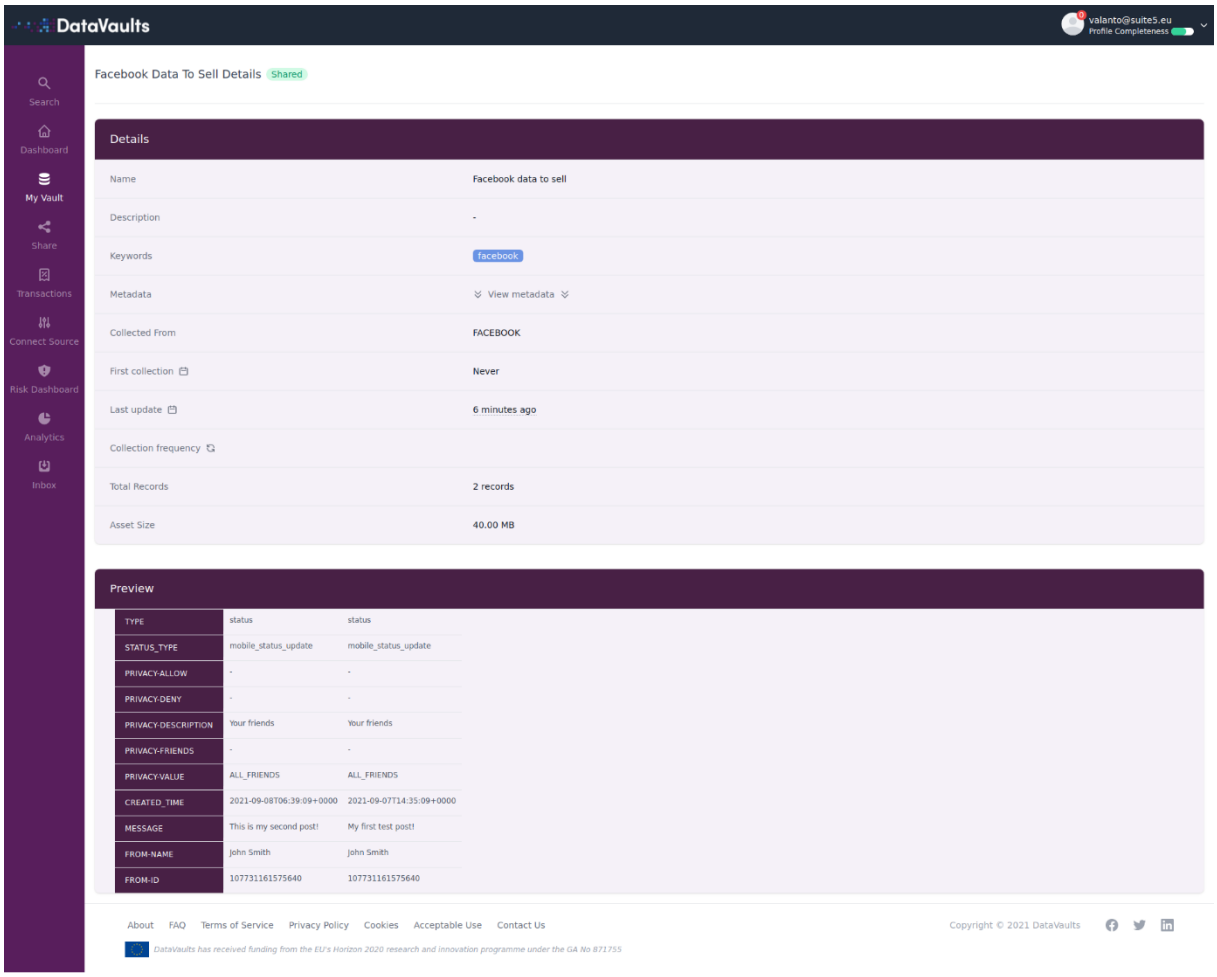


Figure 15: Asset View

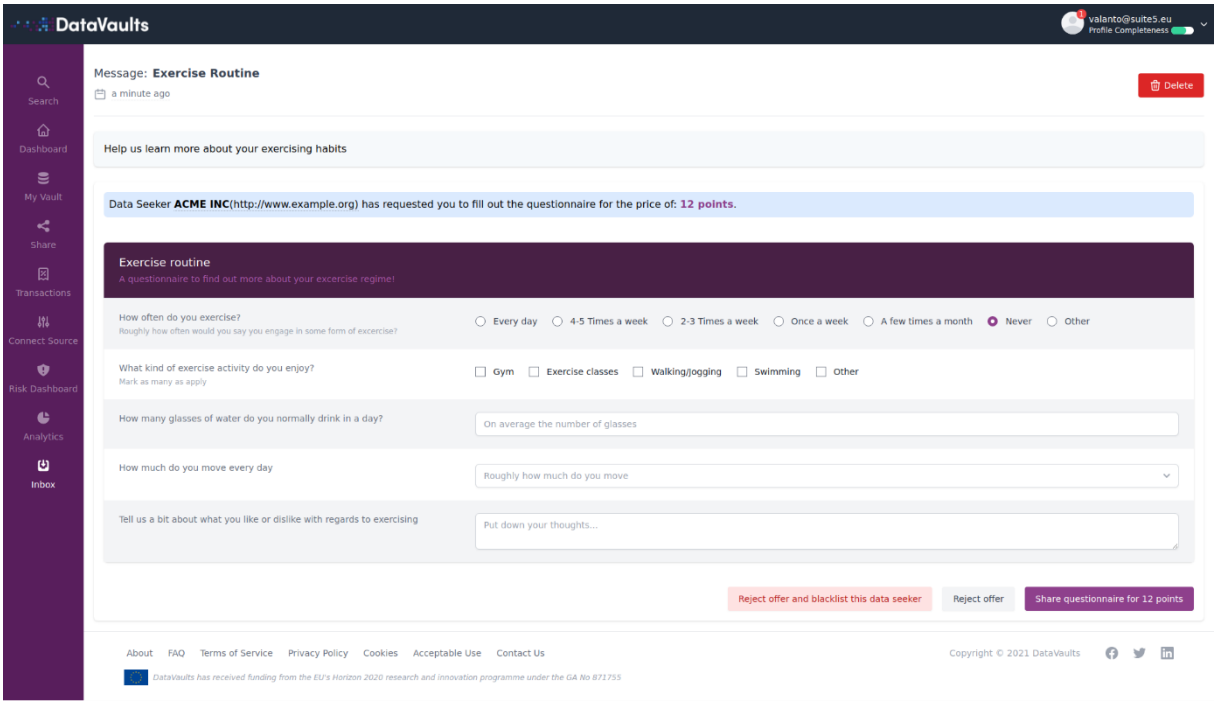
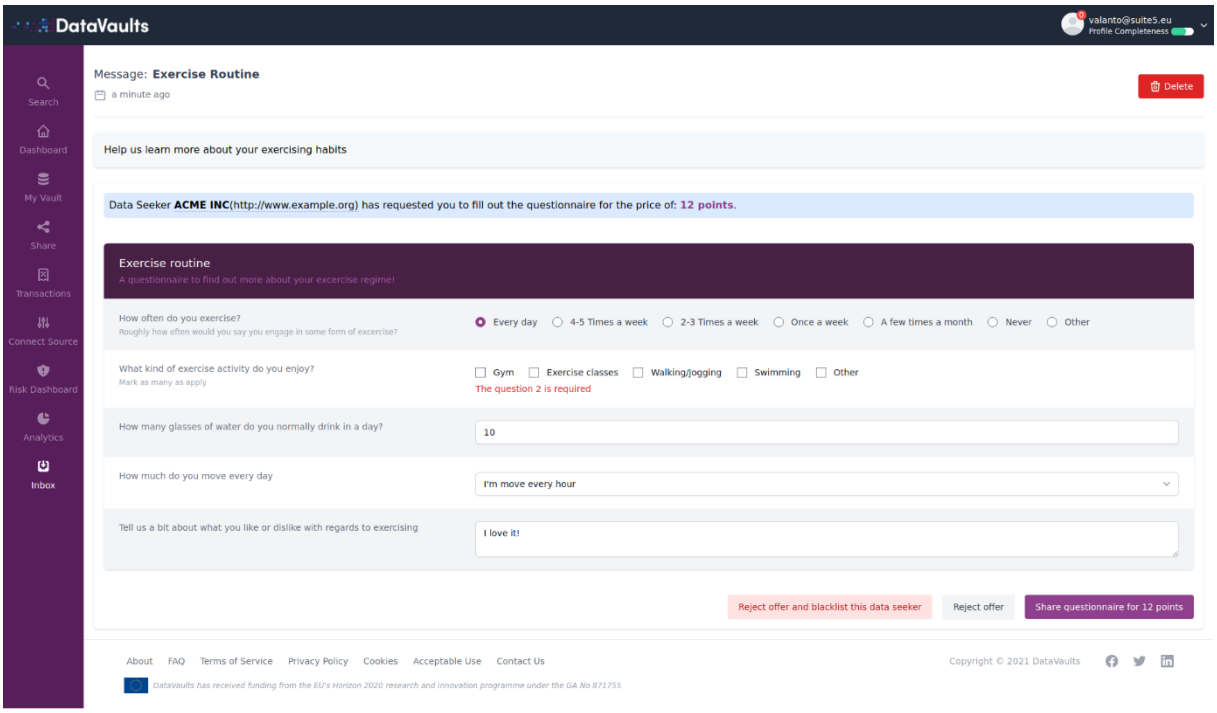


Figure 16: Request for a Questionnaire



The user can always check the transactions related with the data that owns in DataVaults, as shown in Figure 18.

Figure 17: Participation to a Questionnaire

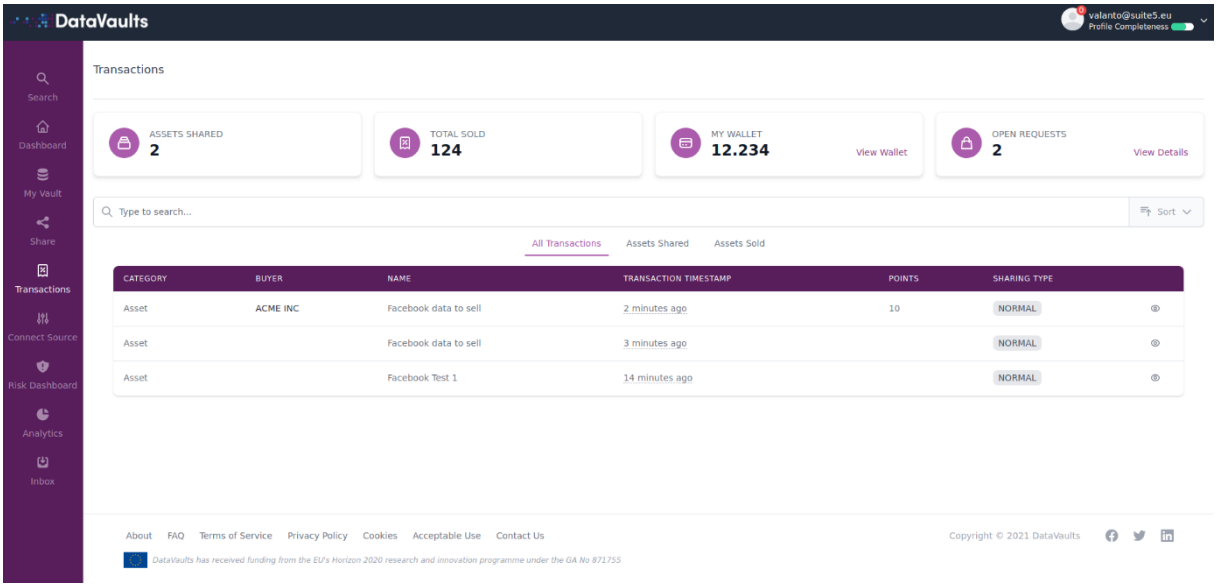


Figure 18: View of Transactions

3.1.2 Cloud Platform

In addition to the provided APIs and Backbone services, the Cloud Platform provides a beta version of the frontend that is used by the Data Seekers, as shown in Figure 19 to Figure 28.

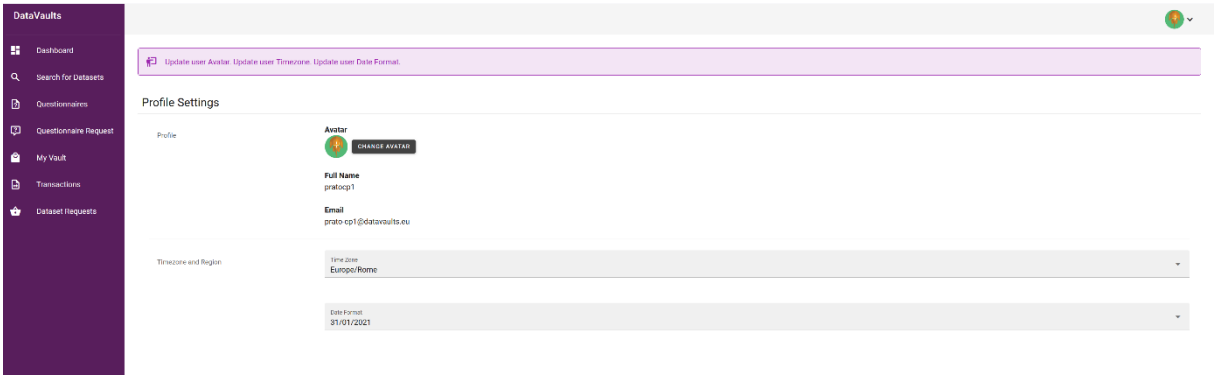


Figure 19: Data Seeker Profile

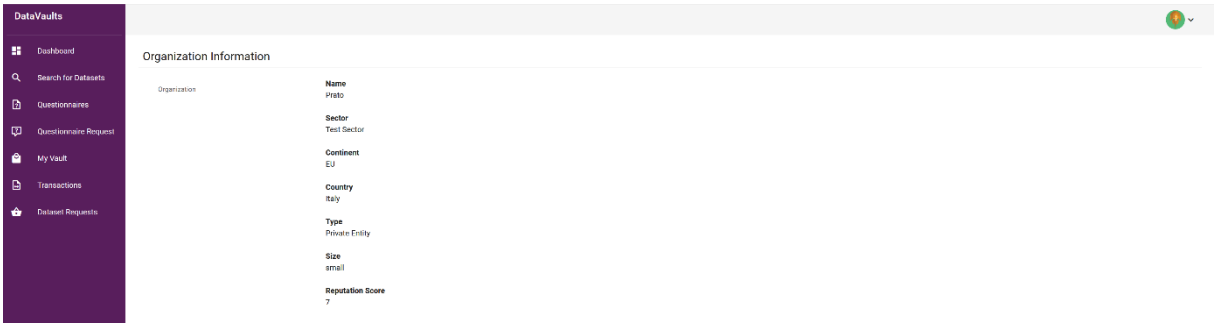


Figure 20: Data Seeker Organization Profile

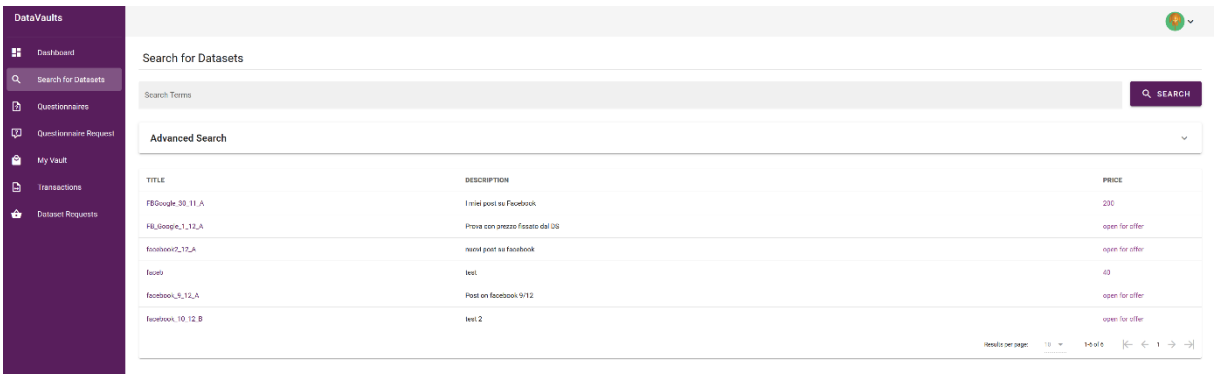


Figure 21: Searching for datasets in the Query Builder

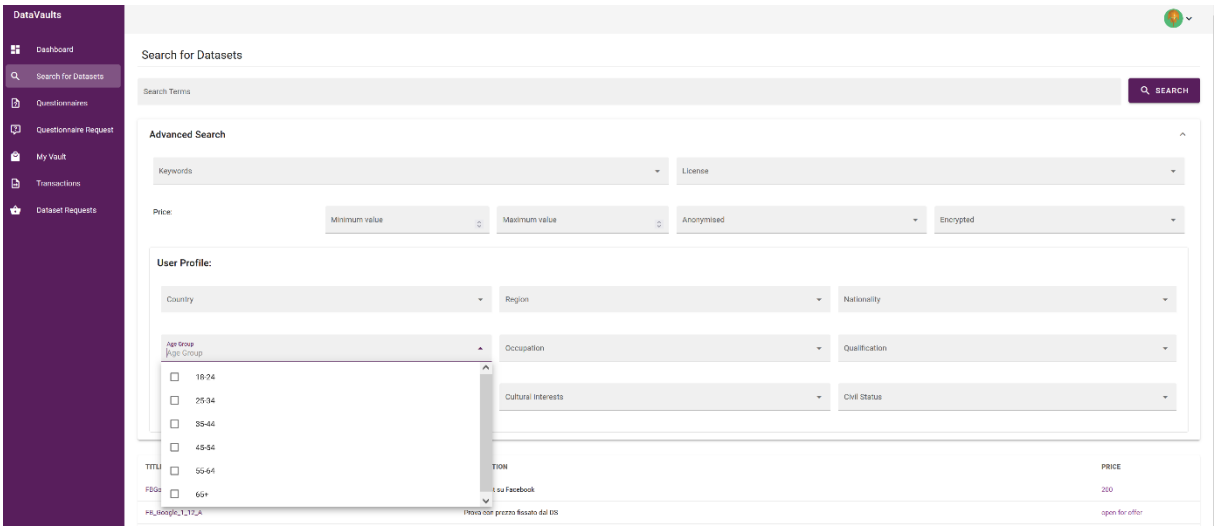


Figure 22: Advanced search

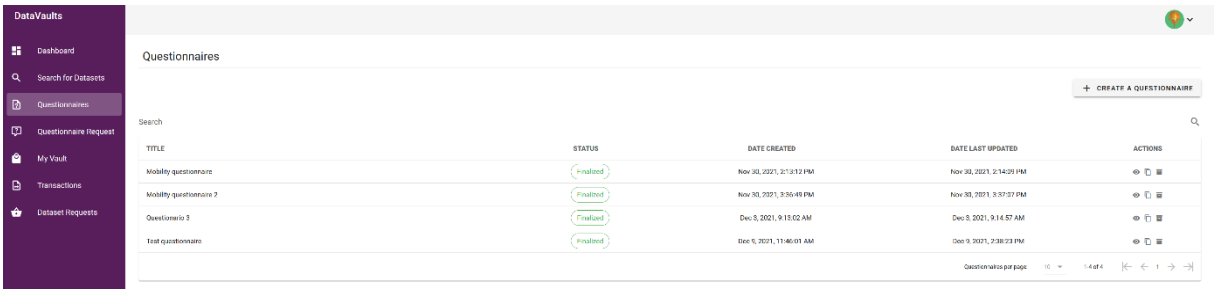


Figure 23: List of Questionnaires

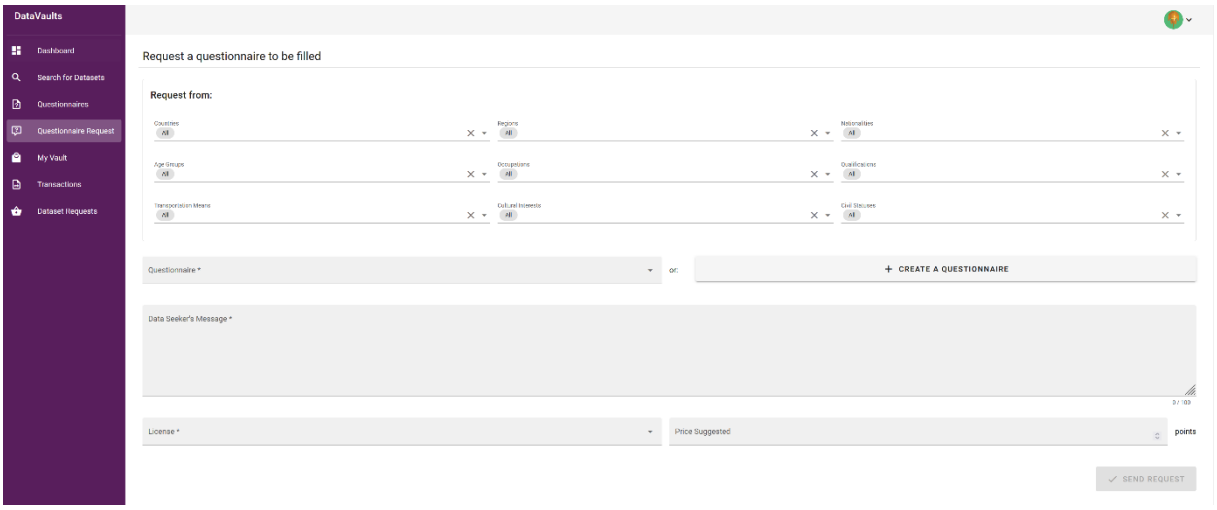


Figure 24: Request for a questionnaire

Figure 25: Creation of a questionnaire

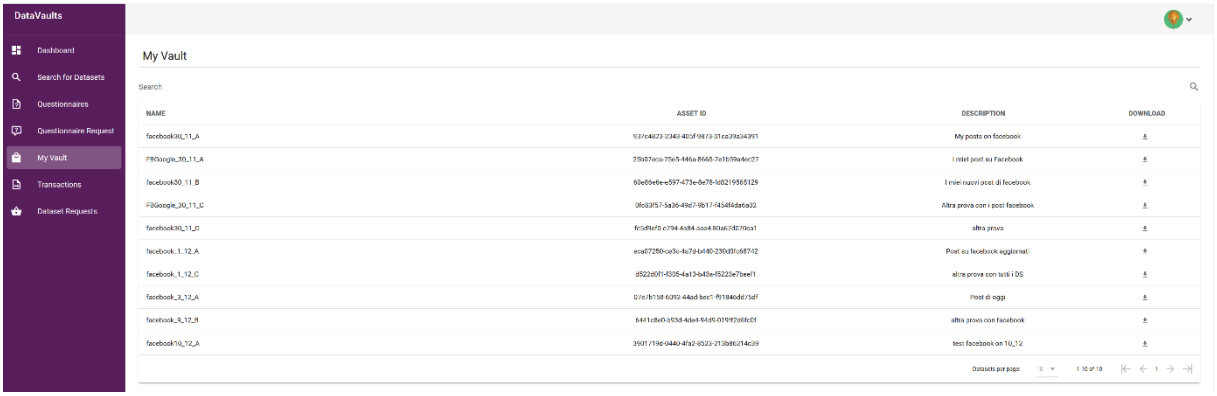


Figure 26: Data Seeker's Vault

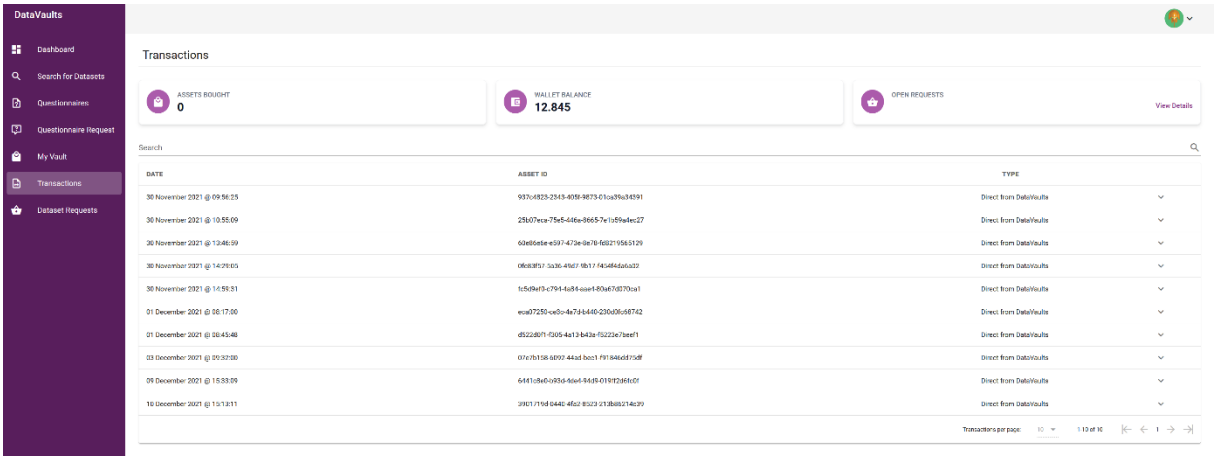


Figure 27: Transactions

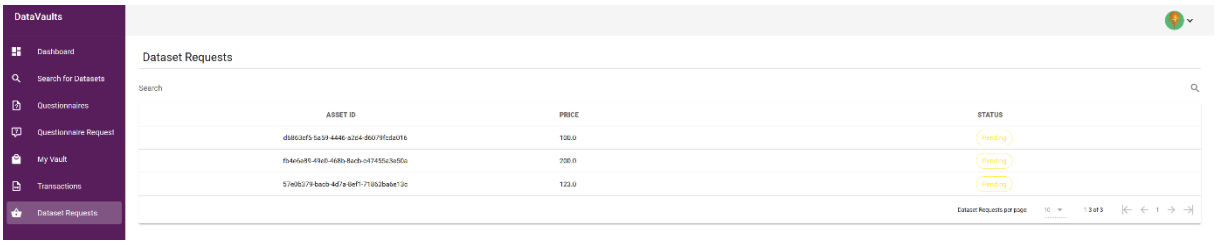


Figure 28: Dataset requests

3.1.3 Secure Analytics Playground

Secure Analytics Playground provides a first version of the fronted that is used by the Data Analysts, as shown in the figures below. Figure 29, Figure 30 and Figure 31 show the way of configuring the new deployment of the SEAS in the DataVaults cloud, by selecting the appropriate deployment features (e.g. IP of the server and credentials where the deployment will take place), plus the selection of available datasets and AI models to deploy. Figure 32 shows an example of visualization of a given analysis in the deployed SEAS.

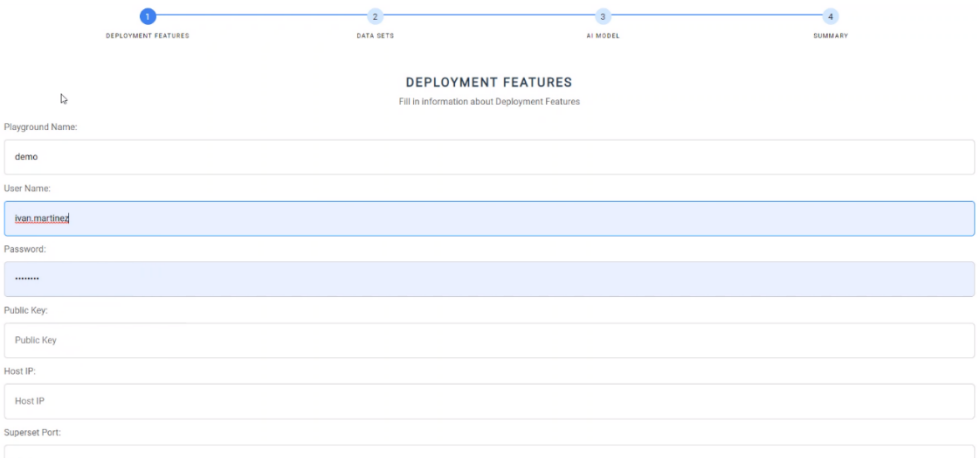


Figure 29: Configure Deployment of SEAS

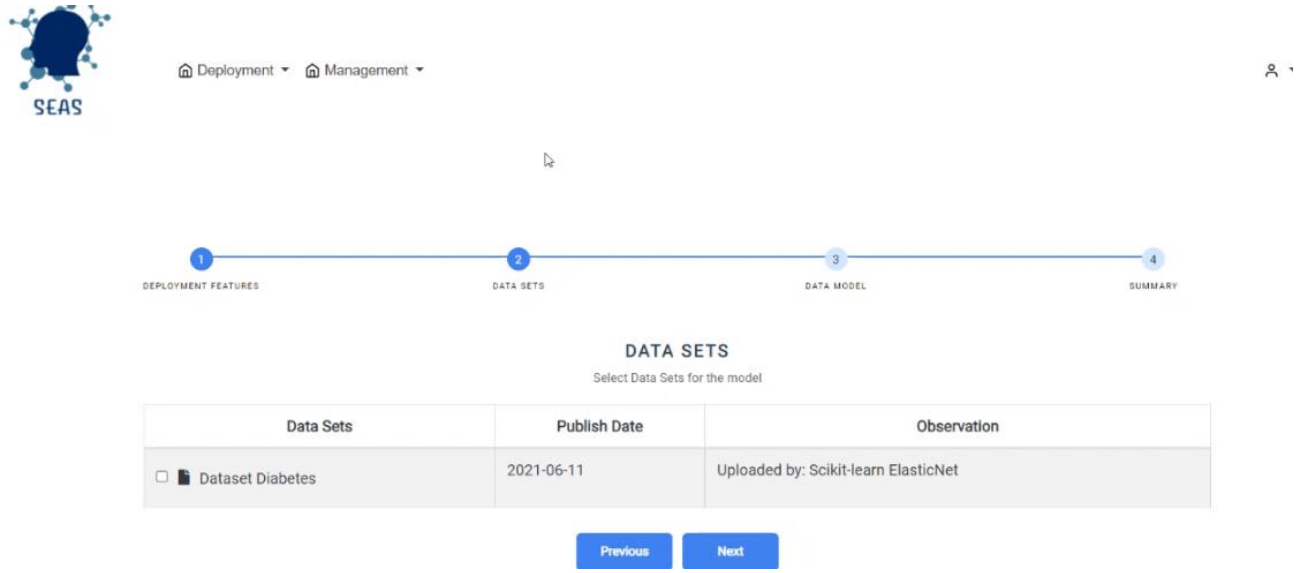


Figure 30: Configure Datasets for SEAS (manually uploaded for Beta version)

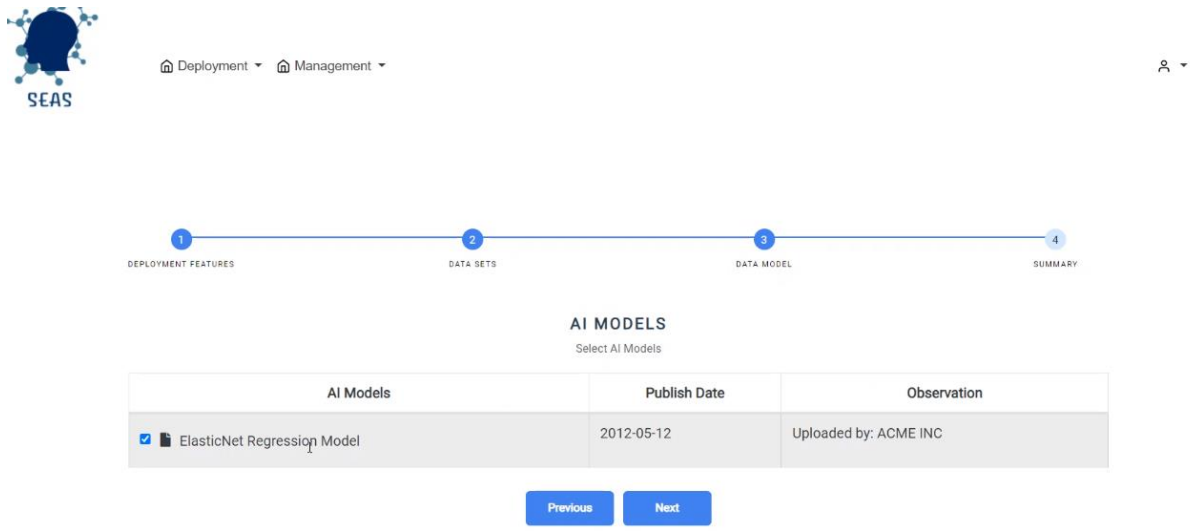


Figure 31: Selection of AI Model for SEAS

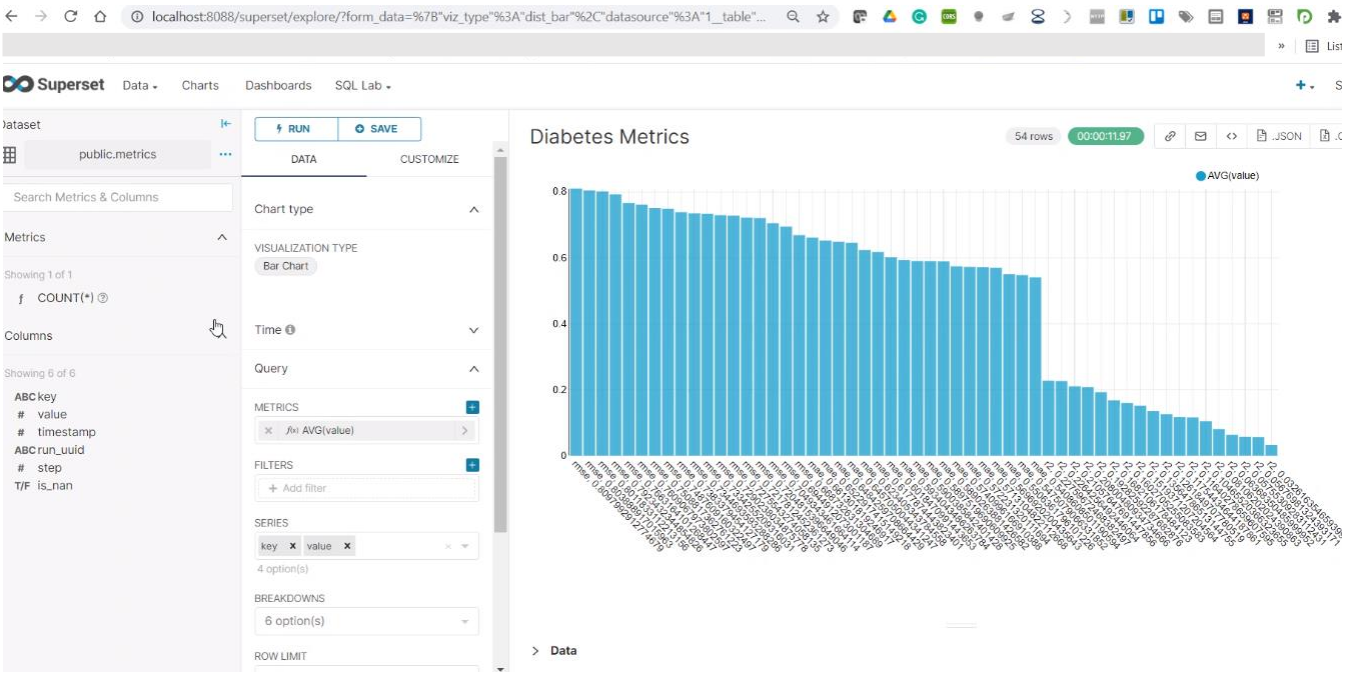


Figure 32: Local Execution Of SEAS

4 RELEASE PLANNING

4.1 RELEASE PLANNING

Based on the followed methodology, we tried not only to check the user stories validity but also to validate the backlog planning, again by using the collaborative spreadsheet of the user stories. The plan is depicted in Figure 33 below.

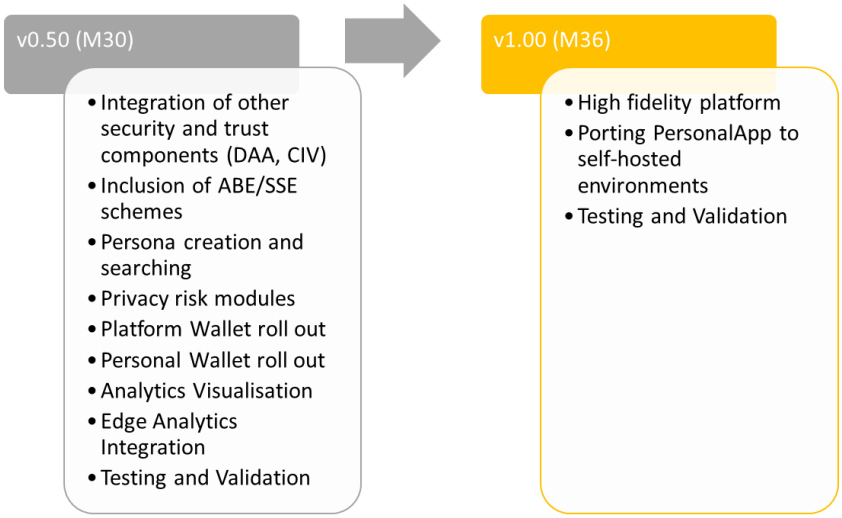


Figure 33: DataVaults Release Planning

In the upcoming V0.5 version of the platform we actually plan to deliver an integrated platform that includes all components and most of the functionalities of the platform. The planned user stories for this release can be found in an online spreadsheet[1]. Finally, with the delivery of V1.00 of the platform, all planned user stories will be delivered to provide a high-fidelity platform. Also, the porting of the Personal App to self-hosted environments will be examined.

[1] ttps://docs.google.com/spreadsheets/d/18kYwSAz4QZJV0UQAkBCWPhOxRrkGaASWgzEoLP0i4eI/edit?usp=sharing

5 CONCLUSIONS AND NEXT STEPS

In this document, we presented the Beta Version of the DataVaults platform. We started first by presenting the status of the DataVaults platform and more specifically the status of each component, regarding their integration, code availability and installation instructions.

Also, for the support of the design and development process, we used an online collaborative tool to create mockup screens that are one of the main contents of this document.

All technical partners worked for the improvement and enhancement of the platform services (such as databases, message queues, identity manager, blockchain) and the development and integration of the components for the delivery of a well-rounded platform for this Beta release.

Finally, the updated screenshots from the usage of the different components of the platform user stories and the plan for the next releases have been provided, with the next goal being to provide version 0.5 of the platform in M30 to provide a fully working platform that will be validated and fine-tuned until the end of the project.

6 REFERENCES

[1] DataVaults Deliverable D5.1, User Stories and Non-Functional Requirements, 2021.

[2] DataVaults Deliverable D5.2, System Architecture, 2021.

[3] DataVaults Deliverable D5.3, DataVaults Platform - Alpha Version, 2021.

[4] DataVaults Deliverable D3.1, Security, Privacy and Trust Bundles - Version 1, 2021.

[5] DataVaults Deliverable D3.2, Security, Privacy and Trust Bundles - Version 2, 2021.

[6] DataVaults Deliverable D1.3, DataVaults MVP and Usage Scenarios, 2021.

[7] DataVaults Deliverable D1.4, Updated DataVaults Concept,2021.

[8] DataVaults Deliverable D4.1, Data Sharing, Value Generation and Intelligence Bundles Version 1, 2021.

APPENDIX A: INSTALLATION INSTRUCTIONS

Secure Analytics Playground

Configuration

For the integration with ansible tower and kafka to work we need to set up the variables in the configuration file:

conf/application.conf:

Modify the file with the correct variables:

```
datavaults{
  ansibleTower.endpoint = "[endpoint_value]"
  kafka.broker = "[broker_value]"
}
```

endpoint_value: Url to the ansible tower API

broker_value: Hostname:port of the kafka broker to write the results into.

conf/application.conf:

Modify the file with the correct variables:

```
db{
default.driver=com.mysql.jdbc.Driver
default.url="<JDBC_CONNECTION_STRING>"
default.username="<USERNAME>"
default.password="<PASSWORD>"
}
```

JDBC_CONNECTION_STRING: Connection string via JDBC to the database. e.g: “jdbc:mysql://127.0.0.1/datavaults?useSSL=false”

USERNAME: Username to connect to the database

PASSWORD: Password to connect to the database

A local file containing seas credentials it is need. There are a concrete propoertie to define it:

```
seas.credentialsFile.path = “C:/temp/seas.cred”
```

Finally, the local credential file have to include the following properties:

seas.ftp.user=xxx

seas.ftp.password=xxx

seas.ftp.ip=xxx

awx.user=xxx

awx.password=xxxx

Debug

For debug we can use this stackoverflow answer: Debug

Basically, configure IntelliJ to debug on remote on port 9999 and launch sbt with debug to that port.

With sbt:

sbt -jvm-debug 9999 run

Running

Run this using sbt.

This has a precompiled standalone version of sbt to run the project locally.

Simply run :

sbt run

And then go to http://localhost:9000 to see the running web application.

Controllers:

Results_Controller: Controller in charge of managing everything related to the results.For now it handles the render of the results list as well as querying the database for the results of the running user.

UserController: Controls user related information. It controlls the creation of a new user, displaying information of the current user, showing the inventories and credentials of the current user.

Ansible_Tower_Controller: Controls everything related with ansible tower. Any operation over the benchmarks, creation of new credentials, creation of new inventories.

Modules:

The DataVaults Web Platform is comprised by the core platform plus a set of extra modules in charge of processing data or centering shared tools.

commons: Set of Ebean classes in charge of handling the DB storage. This is shared among any modules that needs to read/write from/to the backend DB.

resultsSender: Module used for injecting results via web to the platform. This module basically mimics the functionality of the Python library deployed with the ansible playbooks. Its main functionality is to get a file as input and send it to kafka with a predefined structure.

kafkaconsumer: This is a module designed to run independently. It monitors the kafka topics searching for new results. Once it finds any of them, it writes it directly to the results DB without changing it.

metricsSpawner: Another module to be run independently. This module polls the DB for new results that are still not processed into metrics. Once an unprocessed result is found it parses it to detect the metrics and store them in the DB.

Deploy:

Extracted and adapted from the play framework documentation

The dist task builds a binary version of your application that you can deploy to a server without any dependency on SBT, the only thing the server needs is a Java installation.

To use it simply write in the console

```
$ sbt dist
```

This produces a ZIP file containing all JAR files needed to run your application in the target/universal folder of your application.

To run the application, unzip the file on the target server, and then run the script in the bin directory.

The name of the script is your application name, and it comes in two versions, a bash shell script, and a windows .bat script.

$ unzip datavaults-platform-1.0-SNAPSHOT.zip

```
$ chmod +x datavaults-platform-1.0-SNAPSHOT/bin/datavaults-platform

$ datavaults-platform-1.0-SNAPSHOT/bin/datavaults-platform -
Dplay.http.secret.key=[your_secret_key] -
Dconfig.file=/full/path/to/conf/production.conf &
```