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

D5.5
DataVaults Platform – Version 0.5

Editor(s)	Konstantinos Oikonomou, Giannis Ledakis
Lead Beneficiary	UBITECH
Status	Final
Version	1.0
Due Date	30/06/2022
Delivery Date	30/06/2022
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	D5.5 - DataVaults Platform – Version 0.5
Editor(s)	UBITECH - Konstantinos Oikonomou
Contributor(s)	Ubitech – Giannis Ledakis Ubitech – Konstantinos Oikonomou Tecnalia – María José López 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 – Yuri Glickman FRAUNHOFER – Simon Dutkowski FRAUNHOFER – Torben Jastrow PRATO – Elena Palmisano PRATO – Paolo Boscolo
Reviewer(s)	TECNALIA MAGGIOLI

Abstract	This document is a deliverable of WP5 and describes the intermediate release (version 0.5) 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 Version 0.5 of the DataVaults platform. Version 0.5 is the intermediate, stable release of the platform, which 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 Beta 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 Version 0.5 developed in WP3, WP4 and WP5, as well as screenshots from the platform usage and details on the technical assurance of the DataVaults platform. As a result of the improvements implemented for Version 0.5, the majority of features planned for the final release of the platform are now considered implemented and integrated and testing has been included. Finally, based on the development and integration that happened so far, the document includes a detailed overview of the platform installation and also the plan for the subsequent releases.

Table of Contents

1 Introduction 8
1.1 Document structure 8
2 DataVaults Platform Status 9
2.1 Overview 9
2.1.1 DataVaults Cloud Platform 9
2.1.2 Personal DataVaults App 13
2.2 Integrated Platform Status 17
2.2.1 DataVaults Methodology Phases as Part of Version 0.5 18
2.2.2 DataVaults Platform Interfaces 20
3 DataVaults Platform Usage & Installation 21
3.1 Platform Usage 21
3.1.1 Personal App 23
3.1.2 Cloud Platform 32
3.1.3 Secure Analytics Playground 37
3.2 Platform Installation 39
4 Technical Assurance 41
4.1 Continuous Integration 41
4.1.1 Components Availability 41
4.1.2 Continuous Integration Flow 43
4.1.3 Issues Management 44
4.2 Testing 45
5 Release Planning 53
5.1 Release Planning 53
6 Conclusions and Next Steps 54
7 References 55

List of Figures

Figure 1: DataVaults Lifecycle as part of the 0.5 version 19
Figure 2: Highlights from Landing Page of DataVaults 22
Figure 3: Registration on DataVaults 22
Figure 4: Login on DataVaults 23
Figure 5: Personal App Dashboard 23
Figure 6: User Profile Page 24
Figure 7: Adding a Data Source 24

Figure 8: View of User's Vault 25
Figure 9: Sharing an asset – Part of the new Template/Wizard 25
Figure 10: Sharing an asset (step1) 26
Figure 11: Anonymization while sharing an asset 27
Figure 12: Creation of Access Policy for the sharing of an asset 27
Figure 13: Sharing configuration 28
Figure 14: Final Step for Sharing an Asset 28
Figure 15: Request for Dataset 29
Figure 16: Asset View 30
Figure 17: Request for a Questionnaire 30
Figure 18: Participation to a Questionnaire 31
Figure 19: List of Transactions 31
Figure 20: Transactions Analytics Page 32
Figure 21: Cloud Platform Landing Page 32
Figure 22: Data Seeker Profile 33
Figure 23: Data Seeker Organization Profile 33
Figure 24: Searching for datasets in the Query Builder 33
Figure 25: Advanced search 34
Figure 26: List of Questionnaires 34
Figure 27: Request for a questionnaire 35
Figure 28: Creation of a questionnaire 35
Figure 29: Data Seeker's Vault 36
Figure 30: Transactions 36
Figure 31: Dataset requests 37
Figure 32: Configure Deployment of SEAS 38
Figure 33: Configure Datasets for SEAS (manually uploaded for version 0.5) 38
Figure 34: Selection of AI Model for SEAS 38
Figure 35: Local Execution Of SEAS 39
Figure 36: DataVaults Gitlab Repository 42
Figure 37 DataVaults Cloud Platform Repository 43
Figure 38 gitlab-ci.yml file 44
Figure 39: Issue management in DataVaults gitlab 45
Figure 40 - DataVaults Architecture 46
Figure 41 - Test Case Approach 47
Figure 42 - Edge Modules used in the Test Cases 50
Figure 43 - Cloud Modules used in the Test Cases 51
Figure 44 - Final Test Case 51
Figure 45: DataVaults Release Planning 53

List of Tables

Table 1 - Test Case 1: User Management 47
Table 2 - Test case 2: Data Collecting/Ingestion 48
Table 3 - Test case 3: Data Collection & Sharing Configuration 48
Table 4 - Test case 4: Data Transformation & Enrichment 48
Table 5 - Test case 5: Data Asset Sharing Execution 48
Table 6 - Test case 6: Data Assets Purchase & Persona Generation 49
Table 7 - Test case 7: Data Assets Purchase for Incomplete Sharing Configuration 49
Table 8 - Test case 8: Cloud Data Analytics 50

Terms and Abbreviations

ABE	Attribute Based Encryption
API	Application Programming Interface
CO	Confidential
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
SEAS	Secure Analytics System
SSE	Symmetric Searchable Encryption
SSO	Single Sign-On
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 intermediate release of the DataVaults platform, while it also provides the updated status of the components, the usage of the platform, the continuous integration, testing and plan for the upcoming final release. 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 previous platform releases, documented in D5.3 [3] and D5.4 [4] . 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 final release will be published in deliverable 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 Version 0.5.

Section 2 provides the status of Version 0.5 of the DataVaults platform and more specifically the status of each component. The status is analyzed regarding the integration of the component with the platform, as well as its code availability, installation instructions and the VM, IP and port the component is deployed on. This Section also includes the description of the DataVaults methodology phases as part of Version 0.5 and an analysis on the project's lifecycle.

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

Section 4 is dedicated to the technical assurance of the platform and presents details regarding the continuous integration in Version 0.5 and the testing done on this version of the platform.

Section 5 provides the plan and the features to be implemented in the final upcoming release of the platform.

Finally, section 6 concludes the document while section 7 collects the references.

2 DATAVAULTS PLATFORM STATUS

2.1 OVERVIEW

As mentioned already in D5.3 [3] and D5.4 [4], 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 Version 0.5. In order to have a clearer look at the individual components that compose those 2 groups, each component's status, integration, code availability, installation instructions, VM, IP and deployment port 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. Data seekers also have the option to request dataset without a pre-defined price and get notified about the outcome of those requests. 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. 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. Finally, the Cloud Platform Backbone has been localized to 5 languages (English, French, Italian, Spanish, Greek) in order to further help with its adaptation for the pilots' use cases.
- **Code Availability and Installation instructions:** The code for 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.
- **VM, IP and port of the component:** The Cloud Platform is available at https://platform.datavaults.eu and runs on the Cloud Platform VM at port 8080.

2.1.1.2 Persona Generator

- **Status and integration:** Version 1.0 is ready and the full workflow for the persona generator is supported. It has not been integrated with the rest of the DataVaults Cloud Platform.
- **Code Availability and Installation instructions:** The code is available on Gitlab (https://gitlab.com/datavaults/persona_generator/-/tree/main). The installation instructions are as follows:
    1. first run 'docker build --tag persgen .' to download the docker image and set up the docker container
    2. run 'docker run -d -p XXXX:5000 persgen' to run the container and app on port XXXX of the localhost
- **VM, IP and port of the component:** The local port is set to 5000. When setting it up in a container the requirement would be to map the port to the localhost

2.1.1.3 Access Policy Engine

- **Status and integration:** The Access Policy Engine (APEn) functionality has not changed from the Beta version of the Platform. Considering the attributes related to a data seeker and comparing them to the allowed values set by the data owners in the access policies, the Engine grants the access to those data by that data seeker.

  The functionality is exposed as an API, and it is called by 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 of the APEn is available in the 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.

  Currently, it is deployed as a docker where the Cloud Platform is running.

- **VM, IP and port of the component:** The application.properties file in the gitlab contains the Keycloak connection parameters and the port to access the component. Any change in this information should be updated correspondingly.

2.1.1.4 Risk Management Monitor

- **Status and integration:** The Risk Management Monitor provides an estimate of 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.
- **Code Availability and Installation instructions:** The code of the Risk Management Monitor and its installation instructions can be found in the respective DataVaults repository (https://gitlab.com/datavaults/risk-management-monitor)
- **VM, IP and port of the component:** The Risk Management Monitor will be part of the cloud platform deployed at https://platform.datavaults.eu.

2.1.1.5 Secure Analytics Playground

- **Status and integration:** As mentioned in D5.4 [4], "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 subcomponent will define the different features of the Playground configuration and setup, the second one 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. Furthermore, the connection between the different components of the SEAS is already running.
  For this release of the platform, we have implemented two new functionalities.

  1) The first one enables the communication between our component and the Data Explorer component. This connection allows the user to see their available data sets and to use them in the analytics playground and in the visualization.
  2) The second one is to enable the Single Sign On (SSO) between our component and the Keycloak component deployed on DataVaults platform. This SSO is mainly focused to avoid a multiple platform registration between the DataVaults platform and the SEAS component

  Another step for this version, is to deploy the SEAS component inside the DataVaults Cloud, which essentially means to deploy the "Service Analytic Host" and also an Ansible[1] instance in the DataVaults Cloud.

  The next steps of the Platform will be focused on the interaction of the SEAS with the Demonstrators. This collaboration will be focused on two main points:

  1) Based on the functionalities already deployed by the SEAS, generate communication with the Data Providers, which most of the time will be the Demonstrators, to improve the functionalities already deployed.
  2) Based on the demonstrator's feedback, understand the needs of the demonstrators to enable new functionalities or algorithms that help to run, visualize and therebefore understand in a better way their data.

- **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).
- **VM, IP and port of the component:** The "Service Analytic Host" will be run on Port 9000 while the Ansible should use Port 8080 in a non-secure mode and the port 8443

[1] https://docs.ansible.com/ansible/2.5/reference_appendices/tower.html#:~:text=Ansible%20Tower%20(formerly%20'AWX',all%20of%20your%20automation%20tasks.

in a secure deployment. Probably this port may need to be changed in the docker files dependent on the integration and use of ports by other components.

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). Due to the status of developments, there is no deployable version for this
- component. **VM, IP and port of the component:** Currently there is no deployable version for thiscomponent.

2.1.1.7 Data Stream & Contract Composer

- **Status and integration:** The Data Stream 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 and integrated to the relevant pages of the cloud platform.
- **Code Availability and Installation instructions:** The code of the Data Stream and Contract Composer has been integrated to the Cloud Platform backbone.
- **VM, IP and port of the component:** The Data Stream and Contract Composer is part of the cloud platform deployed at https://platform.datavaults.eu.

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.
- **VM, IP and port of the component:** The DLT Engine and the Ledgers utilize 8 VMs in order to provide high fidelity and availability of storing and querying services to the rest of the DataVaults platform

2.1.1.9 Query Builder (FOKUS)

- **Status and integration:** The query builder serves as repository for the metadata. All metadata is managed and stored as linked data in an appropriate format described in

the DataVaults data model. It consists of two components, a triple store, and the query builder itself providing a convenient API which is used by the rest of the cloud platform. The API allows the management of catalogues and datasets and offers query capabilities which hiding the complexity of linked data by reducing the query expression to simple JSON. For more complex queries, the triple store exposes a fully SPARQL 1.1 endpoint.

- **Code Availability and Installation instructions:** The code is hosted in the repositories contained in this gitlab group: https://gitlab.com/datavaults/query-builder. All components have been dockerized for an easier installation process. Docker compose files to deploy the services in an orchestrated fashion can be found in the deployment repository as well. A README contains further build, run and configuration instructions.
- **VM, IP and port of the component:** The Query Builder is part of the cloud platform VM and is run via a docker compose file containing the virtuoso triple store and the query builder. The API endpoint of the query builder is exposed only internally to the VM for other components of the cloud platform on port 8082. The SPARQL endpoint of the triple store is exposed on port 8890.

2.1.1.10 Cloud Platform Data Store

- **Status and integration:** The Cloud Platform Data Store is fully functional in version 0.5 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.
- **VM, IP and port of the component:** The Cloud Platform Data Store is part of the cloud platform deployed at https://platform.datavaults.eu.

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 "My Vault" 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.
- **VM, IP and port of the component:** The Cloud Platform Data Store is part of the cloud platform deployed at https://platform.datavaults.eu.

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 has been updated in v0.50, improving the interconnection with the new components released under this version, while different connectors for accessing data from demonstrators, Twitter and the DataVaults Mobile App have been implemented. Moreover, localisation of the UI has taken place, and now the Personal App supports English, as well as the 4 different languages of the demonstrators.
- **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
- **VM, IP and port of the component:** The Personal App is available at https://app.datavaults.eu/ and runs on the Personal App VM at port 8080.

2.1.2.2 Data Fetcher & Transformation (FOKUS)

- **Status and integration:** The Data Fetcher & Transformation component is integrated into the Personal App Backbone and is used on the "Connect Source" page of the platform to manage and connect the sources that are fetched. Current available sources are Twitter, Andaman7, MIWenergia and Fileupload.
- **Code Availability and Installation instructions:** The code is hosted in the repositories contained in this gitlab group: https://gitlab.com/datavaults/personalapp/data-fetcher-transformer. All components have been dockerized for an easier installation process. Docker compose files to deploy the services in an orchestrated fashion can be found in the deployment repository: https://gitlab.com/datavaults/personalapp/data-fetcher-transformer/deployment
- **VM, IP and port of the component:** The Data Fetcher & Transformation service is only available internally on the Personal App VM. It runs in a docker compose cluster and exposes the API on port 8080. It also exposes a RabbitMQ on port 5672.

2.1.2.3 Private Wallet

- **Status and integration:** In version 0.5, compensation management is included, within the backend wallet, and the merchant with the basic e-shop functionalities is functional, for the next version (v1.0) more functionalities will be added, such as audibility and anti-fraud. Integration with the INFINEON Starter Kit is under development and will be included in version 1.0.
- **Code Availability and Installation instructions:** The back-end code is hosted in the repository https://gitlab.com/datavaults/personalapp/wallet-app-backend and

https://gitlab.com/datavaults/merchant-backend-service-for-wallet-ap. The front-end code is inside the personal app backbone (see section 2.1.2.1). All back-end services are dockerized for easy deployment.

- **VM, IP and port of the component:** The backend wallet is deployed at https://wallet.datavaults.ari-bip.eu/ and the merchant backend at https://merchant.datavaults.ari-bip.eu/ . The port assigned to the wallet is 8081 and to the merchant 8082

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. In v050 the integration with the DAA Engine has been performed, while the updated modules for the Anonymiser and the Access Policy Engine have been integrated. Moreover, a new, easier to use sharing configuration dashboard has been developed, allowing users to choose configuration with max, medium, or low security and privacy guarantees, without going through the whole process of selecting the different options.
- **Code Availability and Installation instructions:** The code for the Sharing configurator has been integrated to the Personal App backbone.
- **VM, IP and port of the component:** The Sharing Configurator is part of the Personal App deployed at https://app.datavaults.eu/

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 and the instructions for the installation are described in the corresponding README files of each gitlab project.
- **VM, IP and port of the component:** The Privacy Metrics Dashboard is part of the Personal App deployed at https://app.datavaults.eu/

2.1.2.6 Anonymiser

- **Status and integration:** Version 1.0 of the Anonymiser has been fully integrated into the Personal DataVaults App. Version 2.0 is complete and incorporates the additional functionality to support Location Privacy (specifically for Latitude and Longitude Geo-Codes). Version 2.0 has not been integrated with the Personal DataVaults App
- **Code Availability and Installation instructions:** Version 1.0 and Version 2.0 are available in Gitlab (https://gitlab.com/datavaults/personalapp/anonymiser). To install Version 2.0 there are two Docker Containers containing an API each and it can be set up using docker compose. To run the anonymizer API application, run "docker-compose up" in the anonymizer-api-v001 directory. If the ports need to be changed

for the anonymizer and location handler APIs you can change them in their respective repository. If the location-handler port is changed, the code that references the location-handler endpoint will need to be changed. To do this go to "AnonHandling.java" and change the 5000 in the line --> URL url = new URL("http://locationapi:5000/dataset/handle_location"); under the anonymizeLocation function to reflect the updated location-handler port.

- **VM, IP and port of the component:** Currently the two services run on Port 8080 and Port 5000 respectively. These may need to be changed in the docker files dependent on the integration and use of ports by other components. If Port 5000 is changed there will need to a minor change made to the anonymizer where it references the location service – it will have to be updated to reference the new Port address so that the correct API calls continue.

2.1.2.7 Access Policy Editor

- **Status and integration:** The Access Policy Editor (APEd) is a step within the sharing configurator of the Personal DataVaults App. The Editor allows the Data Owners to select the valid attributes under which sharing their data. The list of values set as "allowed" for every attribute is stored as part of the Data Sharing Configuration in the blockchain. Two additional functionalities have been implemented in this version, the templates system for saving and manage local policies to be reused, and the "Export" button to present the policy in an IDS or ODRL similar format.
- **Code Availability and Installation instructions:** The code is available as part of the Personal App in its gitlab (mostly https://gitlab.com/datavaults/personal-app-frontend.git) and the instructions for the installation are described in the corresponding README files of each gitlab project.
- **VM, IP and port of the component:** Its implementation is part of the Sharing Configuration component, and it cannot be used in an isolated manner.

2.1.2.8 Transactions Analytics

- **Status and integration:** The Transactions Analytics component (formerly titled as Edge Analytics Engine) is a component that is used to deliver pre-defined statistics relevant to the data retrieval, sharing and transactions operations to data owners. These features have been considered as of highest importance by the demonstrators compared to the initially envisaged analytics to run on the connected data sources, as the heterogeneity of data sources and the effort required by ordinary users to run different analytics has been considered too high. This component has been delivered in the v0.50 version of the Personal App
- **Code Availability and Installation instructions:** The code for the Transactions Analytics component has been integrated to the Personal App backbone.
- **VM, IP and port of the component:** The Transactions Analytics is part of the Personal App deployed at https://app.datavaults.eu/

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 has been delivered in the beta release

- **Code Availability and Installation instructions:** The code for the Data Request Service Resolver has been integrated to the Personal App backbone
- **VM, IP and port of the component:** The Data Request Service Resolver is part of the Personal App deployed at https://app.datavaults.eu/

2.1.2.10 DataVaults Mobile App

- **Status and integration:** The DataVaults Mobile App is to be used as an auxiliary data source for capturing and uploading to the DataVaults Personal App activity data as well as location data of individuals. Moreover, the Mobile App is capable of pushing notifications to the users. The mobile app has been developed and released as a prototype in v0.50 of the platform, and more updates are planned for the next period.
- **Code Availability and Installation instructions:** The code for the DataVaults Mobile App is proprietary. Installation is performed directly from the Google Play Store by downloading the application

2.2 INTEGRATED PLATFORM STATUS

In Version 0.5 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 platform 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 [5] 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 intermediate release, the main focus was to fine-tune the platforms support regarding the integration of actions to properly implement all basic and most of the advanced DataVaults Data Management Lifecycle (DDML) actions and phases.
Regarding the DataVaults Data Analytics Lifecycle (DDAL) and the DataVaults Compensation Lifecycle (DCL), the setup of the backbone elements has been completed and in Version 0.5 the 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 extended to support more data sources to *retrieve data*, while the data schema was improved to facilitate the needs of the Personal Builder. 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/her, 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 too. The Secure Analytics Playground shares the same user database with the cloud platform, so Dataseekers can use the same credentials to access both components.

2.2.1 DataVaults Methodology Phases as Part of Version 0.5

The high-level integrations that may occur between the DataVaults Actors and the DataVaults Platform were also defined in D1.3 [5]. These integrations express the eight Phases of the Methodology, and in this section, we provide a high-level presentation regarding how these Phases are covered in Version 0.5 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.

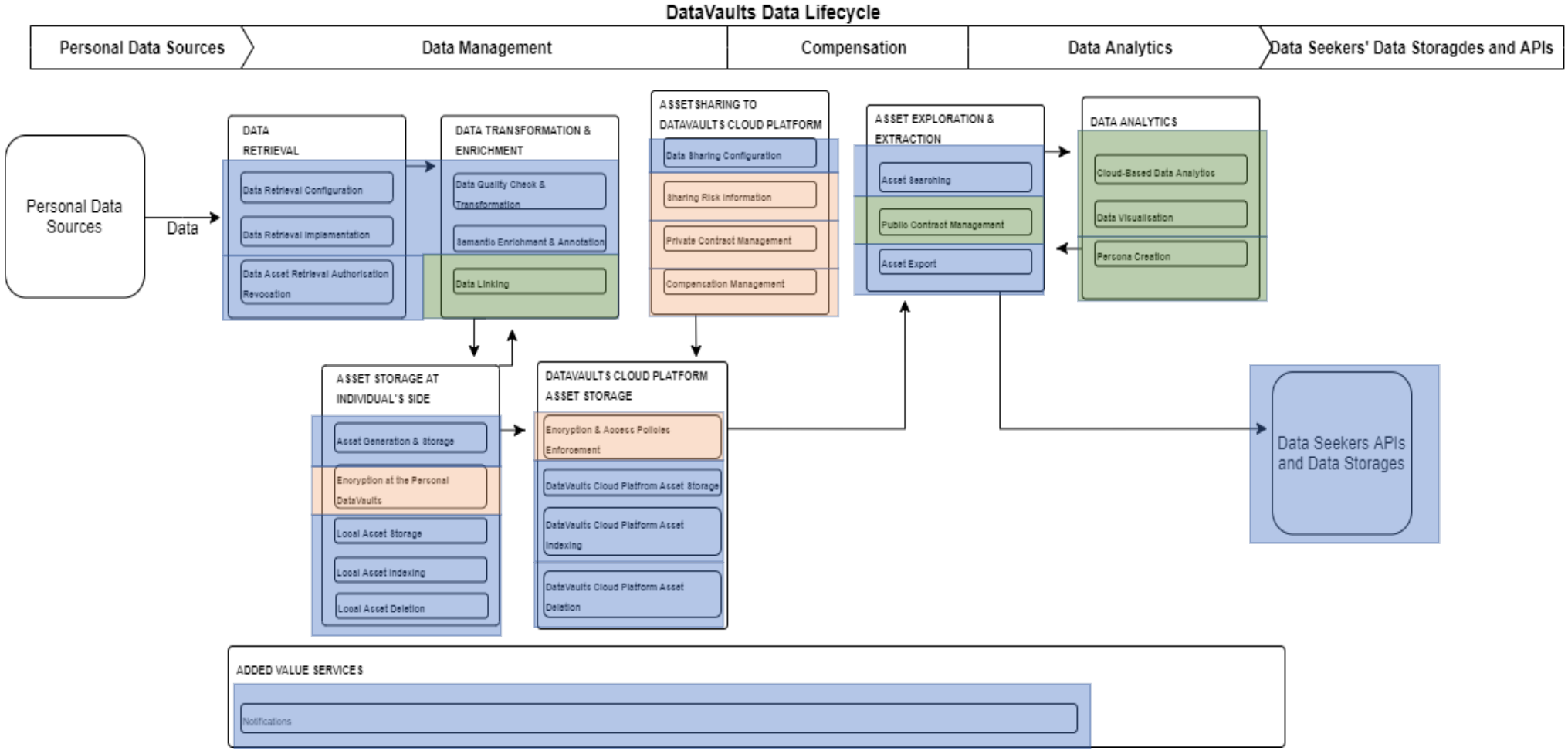


Figure 1: DataVaults Lifecycle as part of the 0.5 version

For easier readability of the document, we provide the description of the phases, as defined in D1.3 [5].

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.

2.2.2 DataVaults Platform Interfaces

After collecting the interaction points between the platform components in D5.2 [2] a set of initial interfaces and APIs were developed and documented in D5.3 [3]. The documentation of those APIs and all the changes that were required as the architecture and integration evolved can be found in the project's Gitlab repository [2]. All API specifications are provided in the OpenAPI V3.0 (OAS)[3] format to ensure homogeneity across all components and enable effortless viewing and testing of the endpoints via the many OAS visualization tools available (Swagger Editor, ReDoc, etc).

[2] https://gitlab.com/datavaults
[3] https://swagger.io/specification

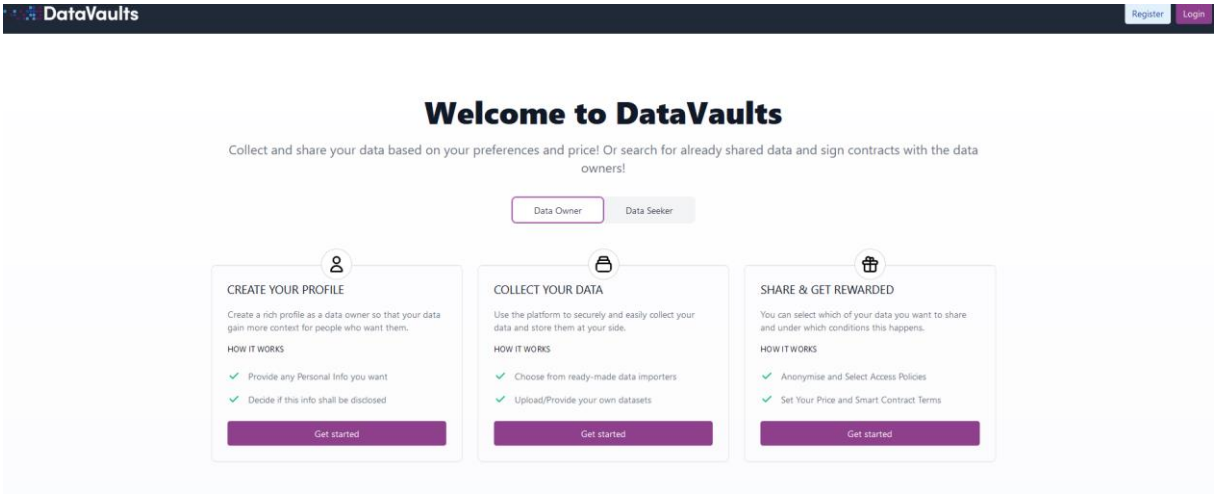
3 DATAVAULTS PLATFORM USAGE & INSTALLATION

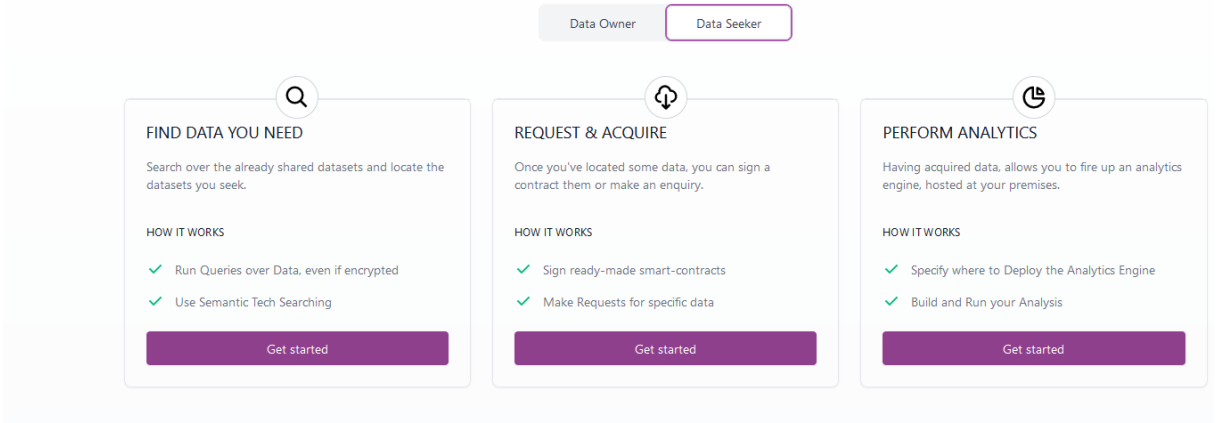
Both the DataVaults Cloud Platform and the DataVaults Personal App have advanced in the 0.5 version of DataVaults, that supports most phases of the DataVaults Lifecycle.

3.1 PLATFORM USAGE

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.





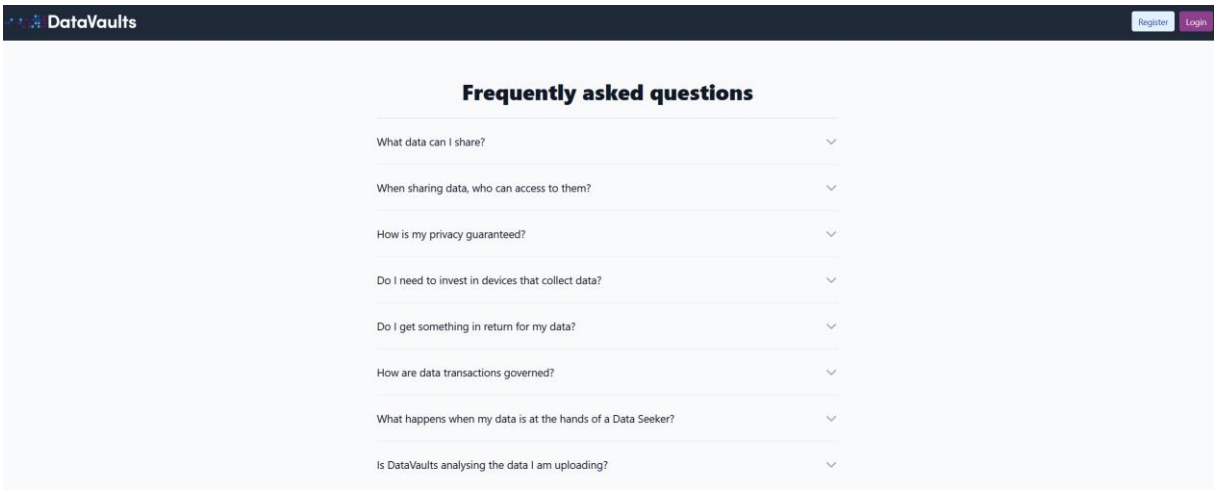


Figure 2: Highlights from Landing Page of DataVaults

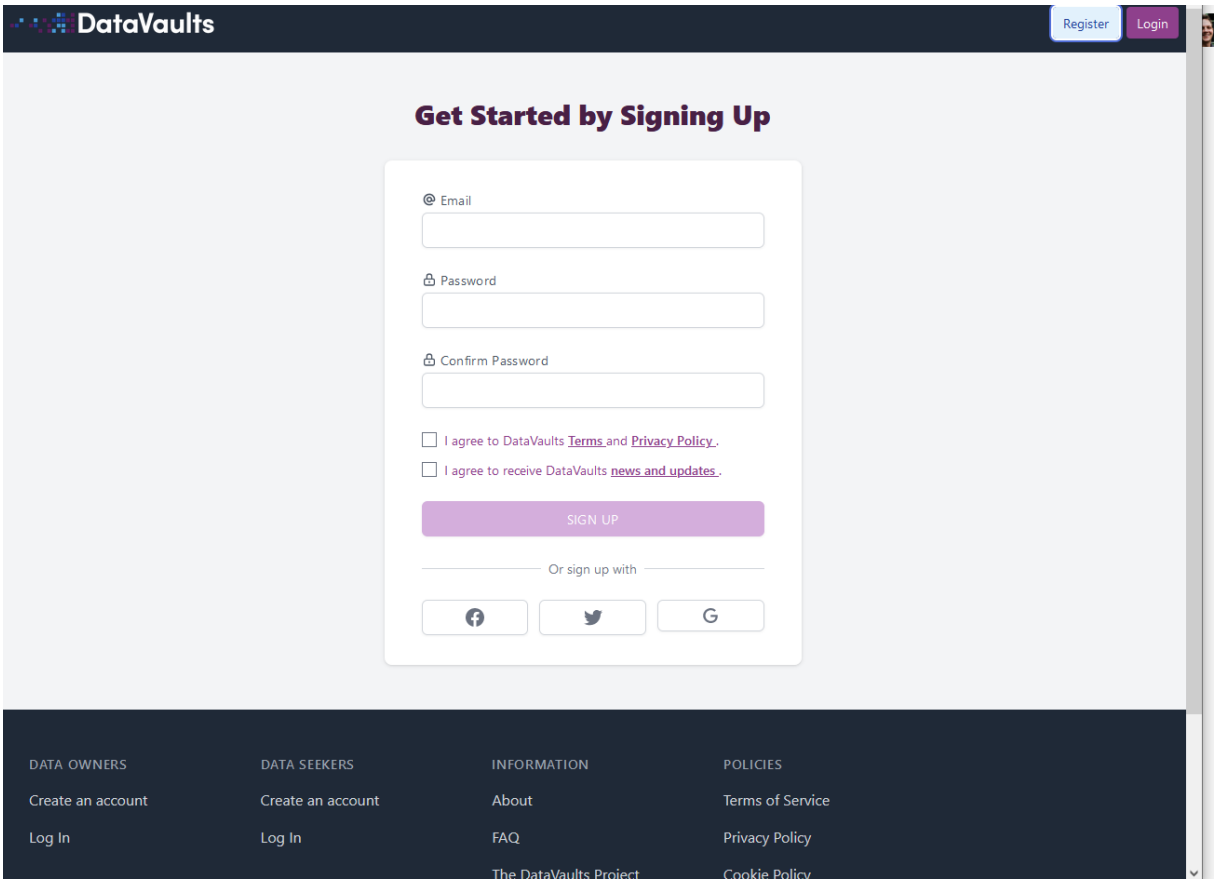


Figure 3: Registration on DataVaults

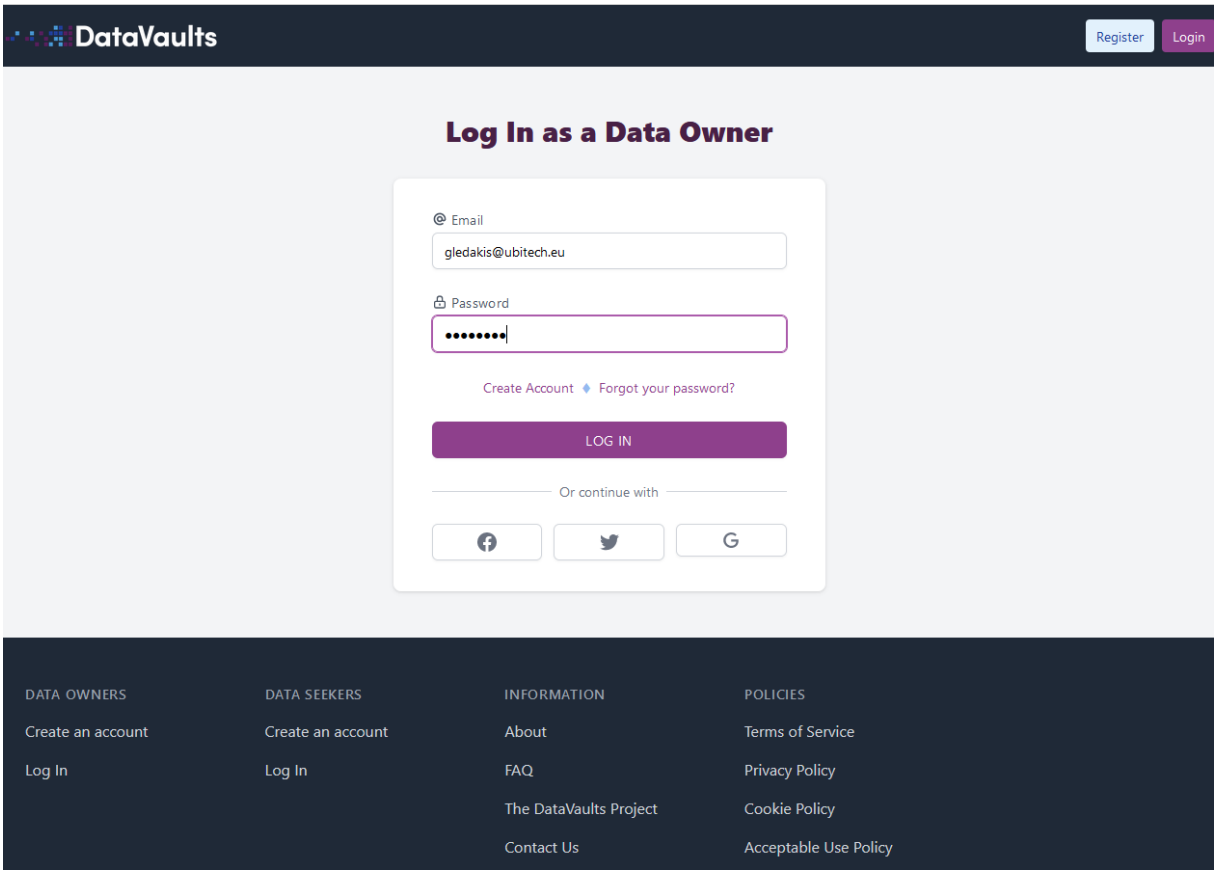


Figure 4: Login on DataVaults

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.

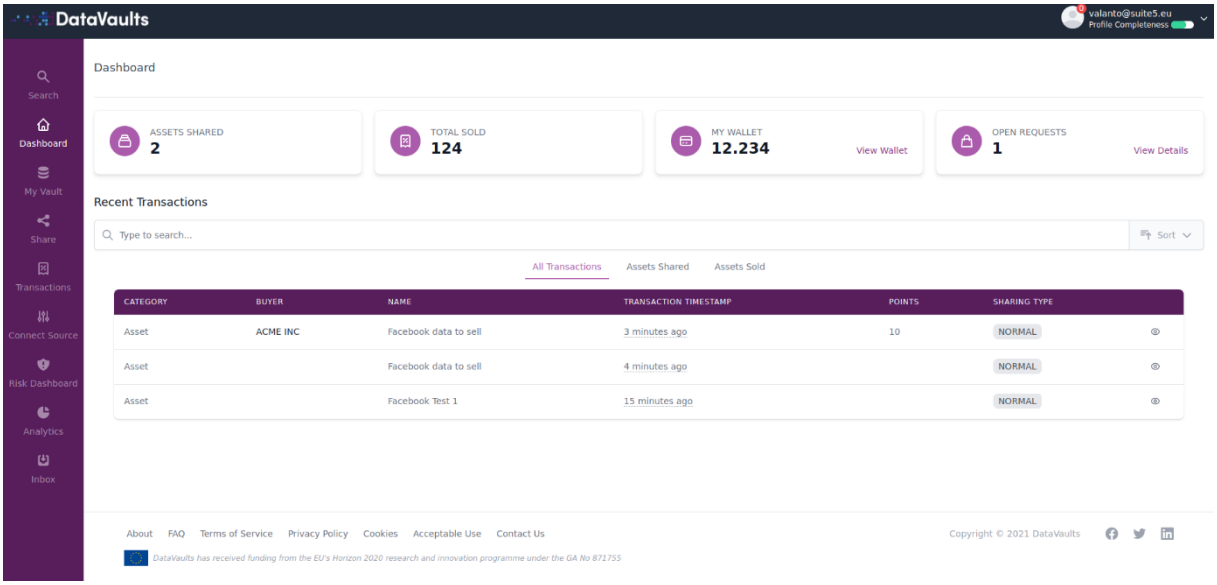


Figure 5: Personal App Dashboard

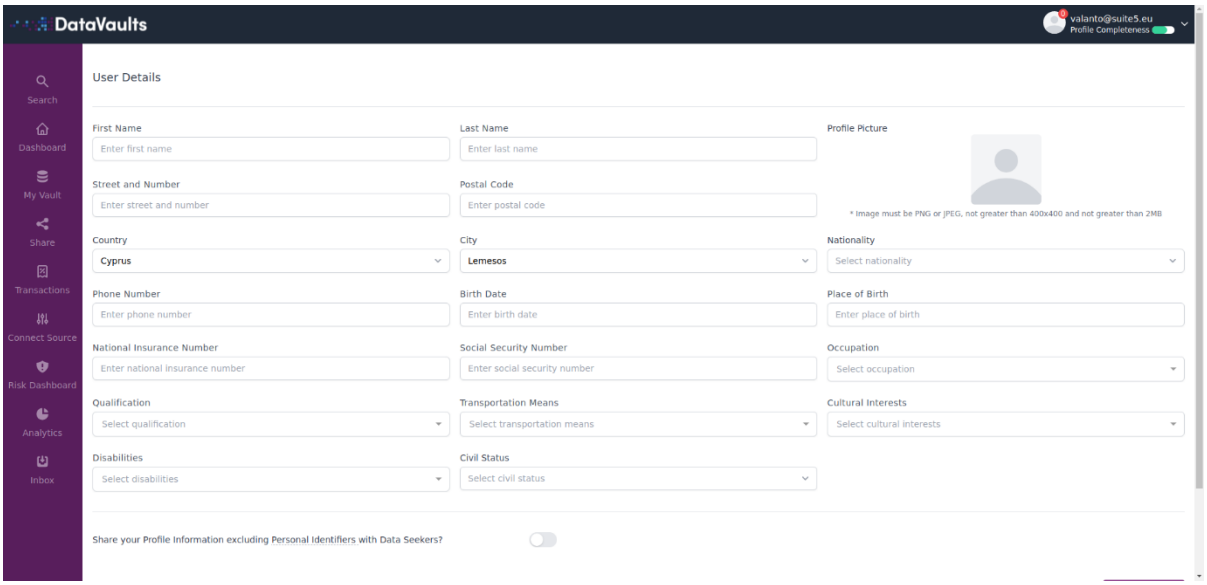


Figure 6: User Profile Page

From Figure 7 to Figure 14, the platform is used by the user in order to connect to a data source and then share to the cloud platform.

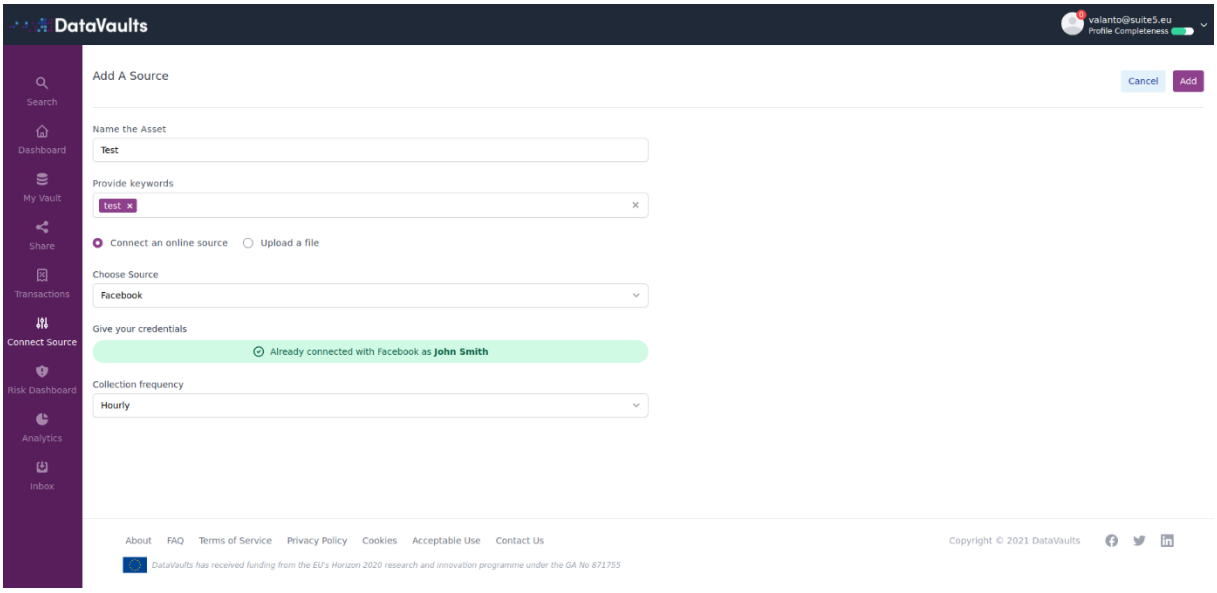


Figure 7: Adding a Data Source

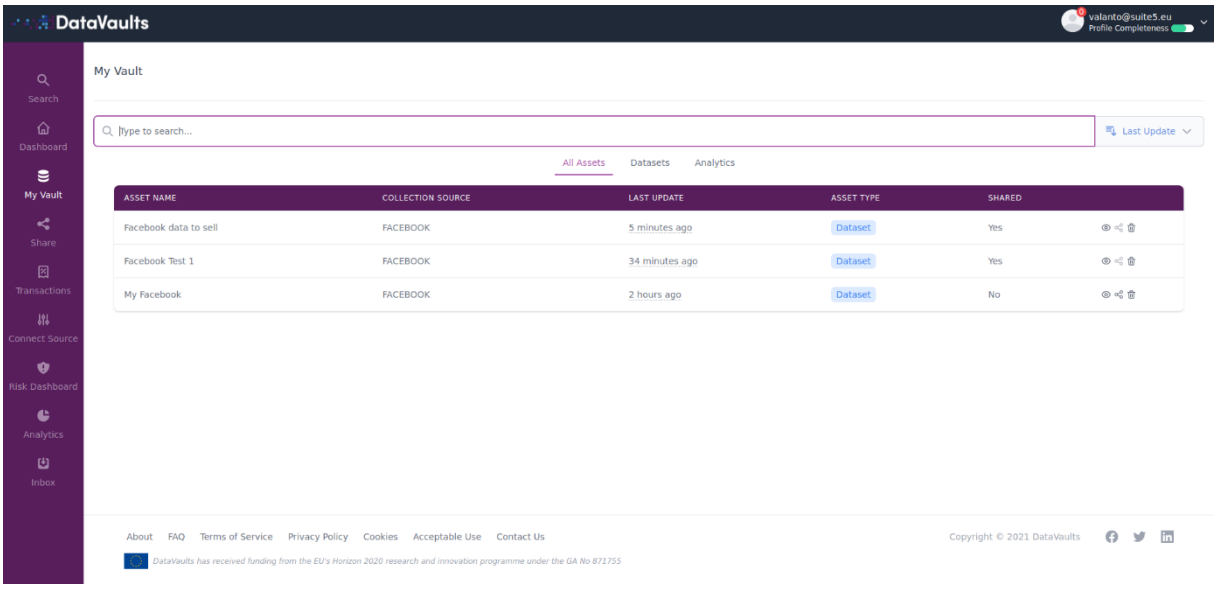


Figure 8: View of User's Vault

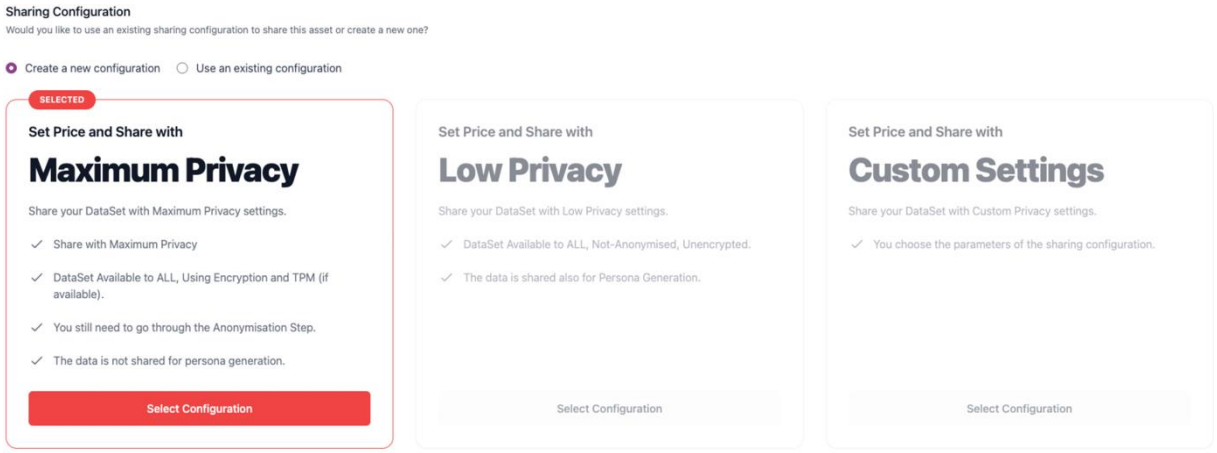


Figure 9: Sharing an asset – Part of the new Template/Wizard

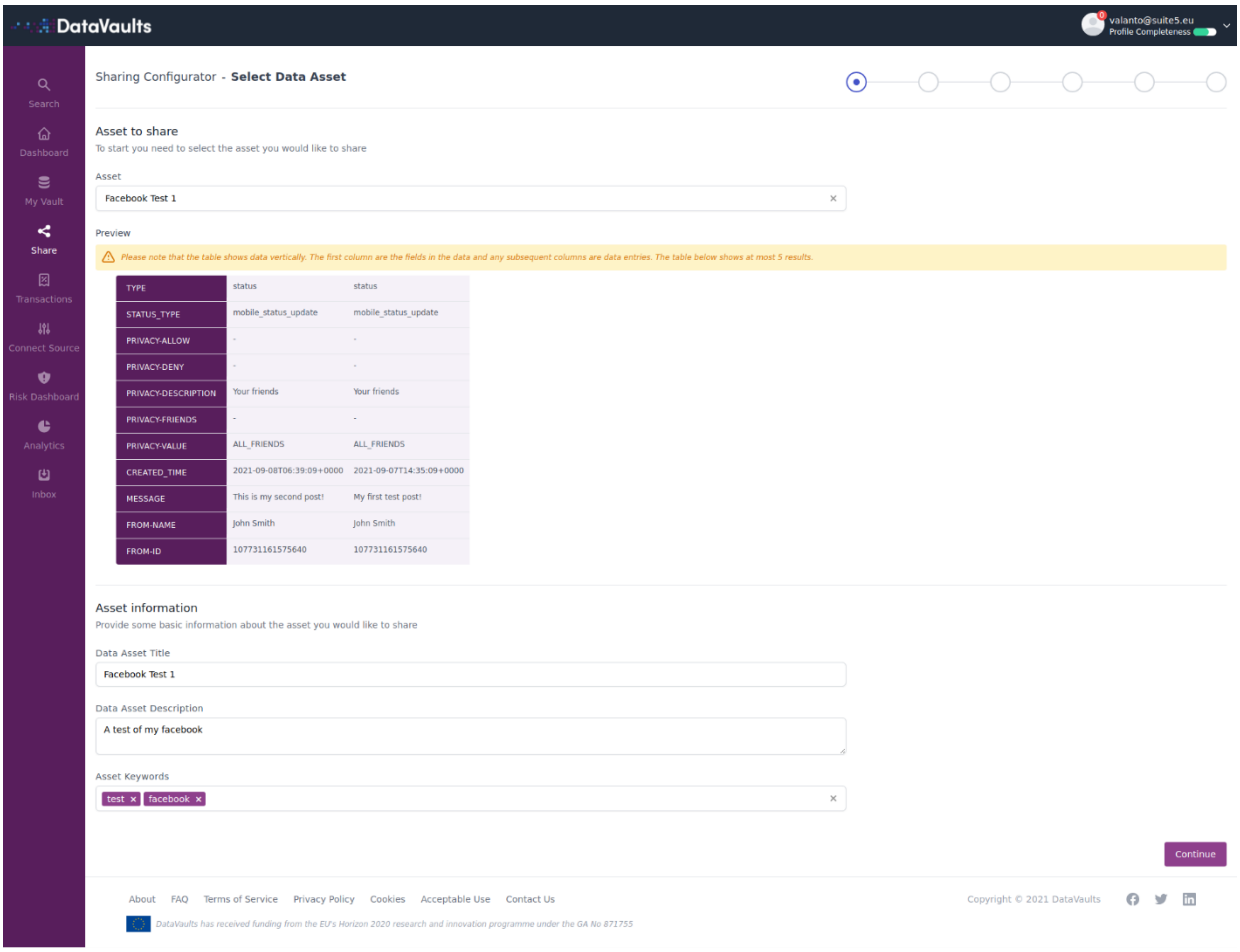


Figure 10: Sharing an asset (step1)

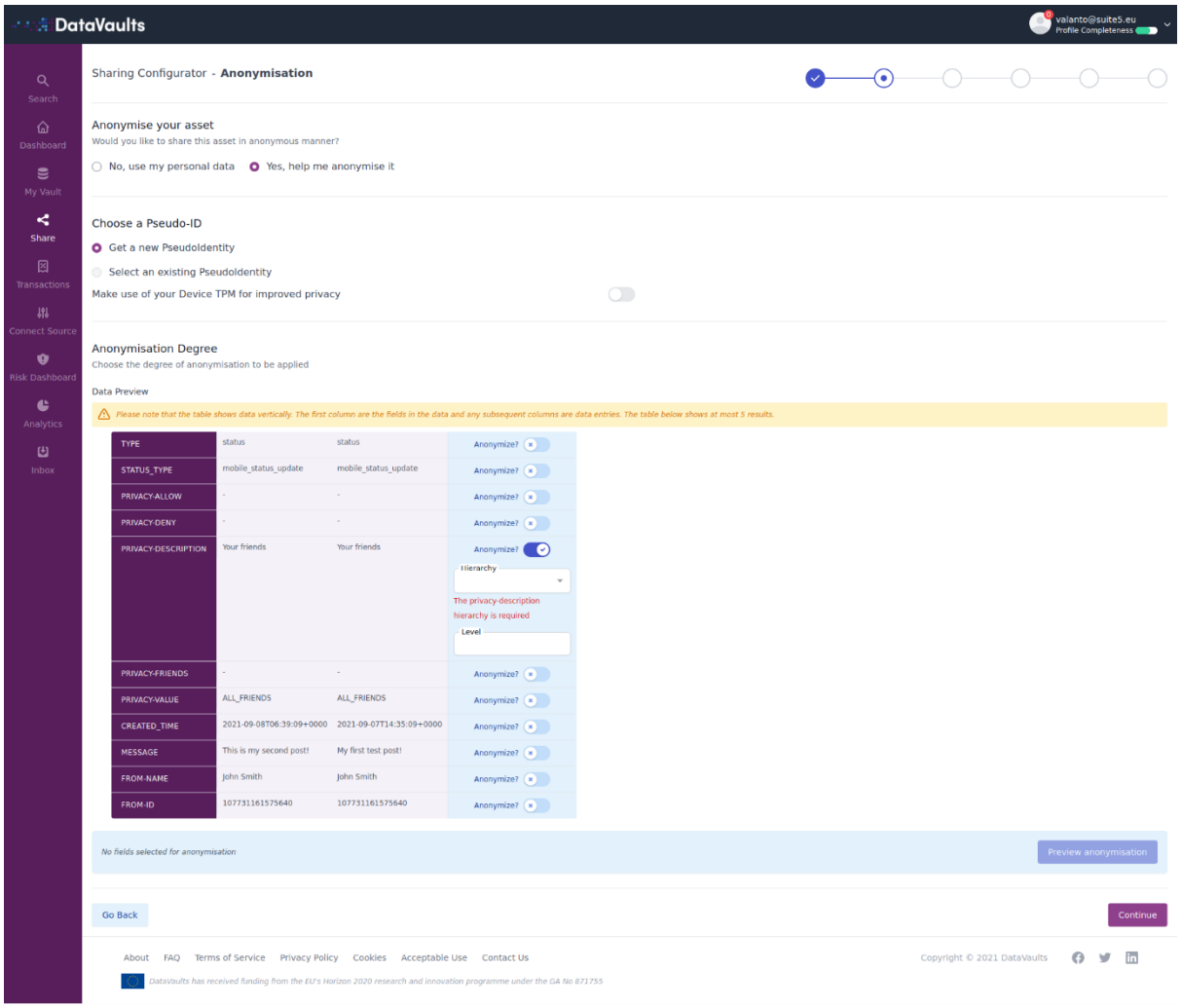


Figure 11: Anonymization while sharing an asset

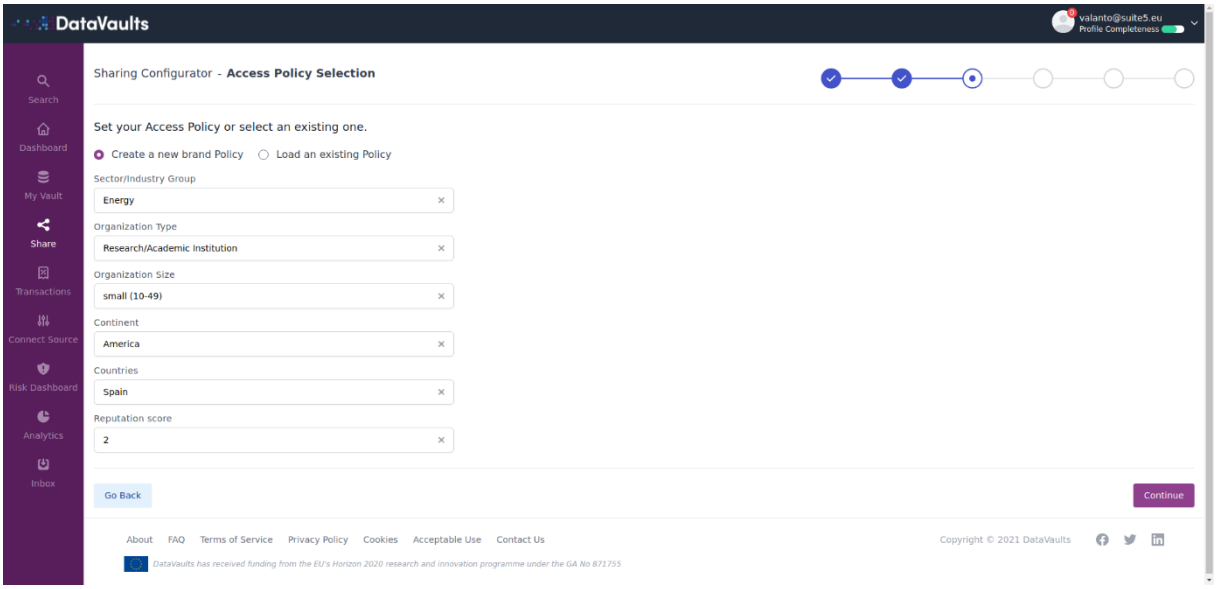


Figure 12: Creation of Access Policy for the sharing of an asset

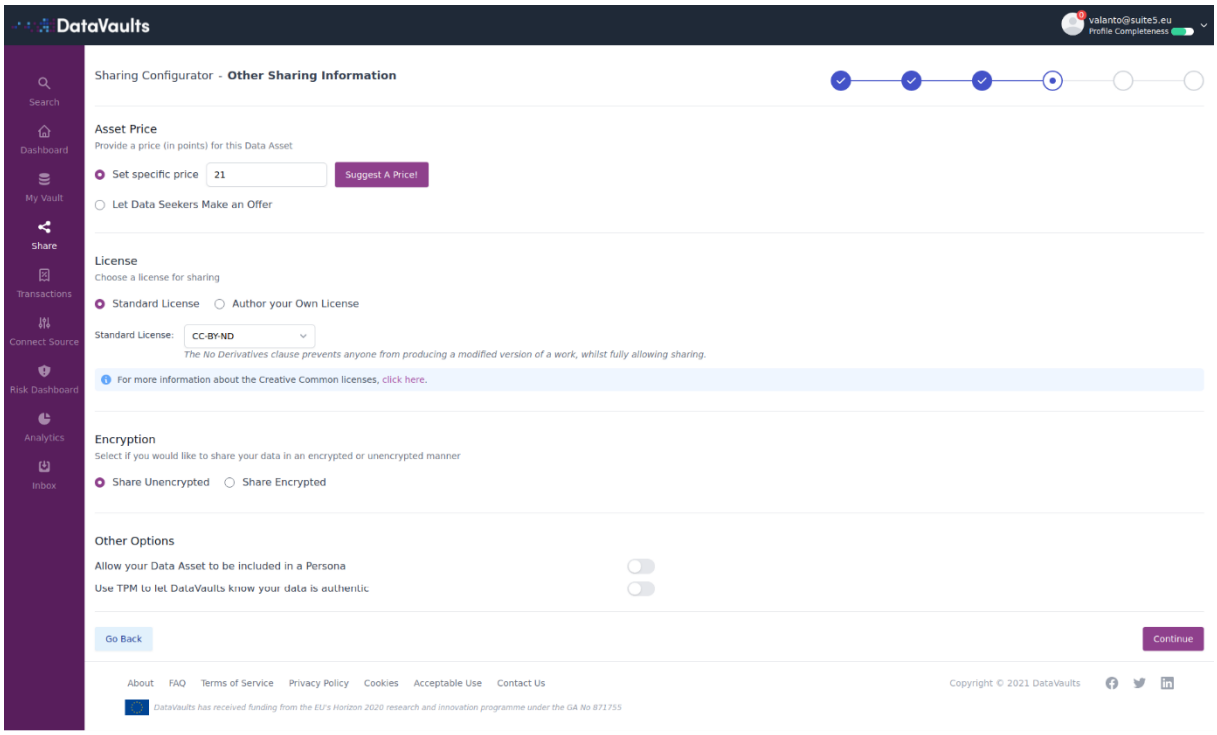


Figure 13: Sharing configuration

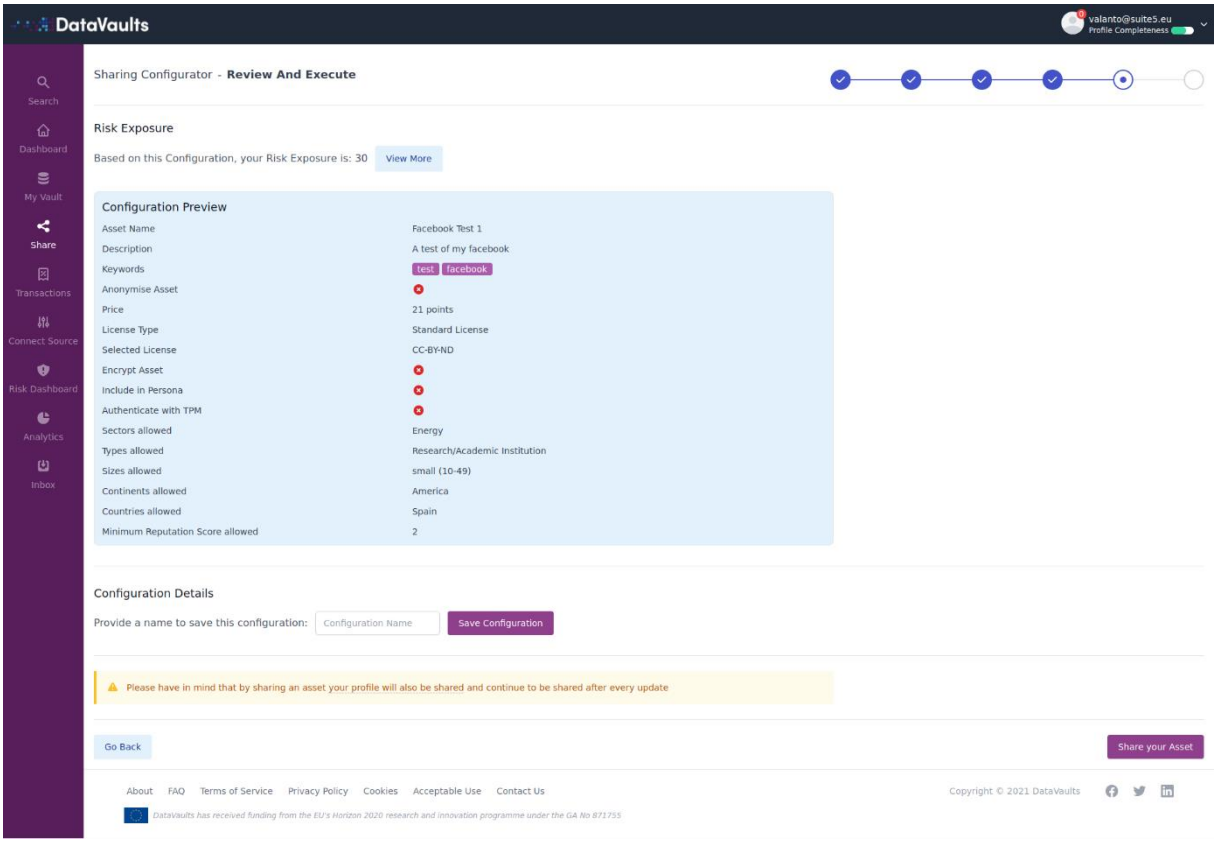


Figure 14: Final Step for Sharing an Asset

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 15 to Figure 18.

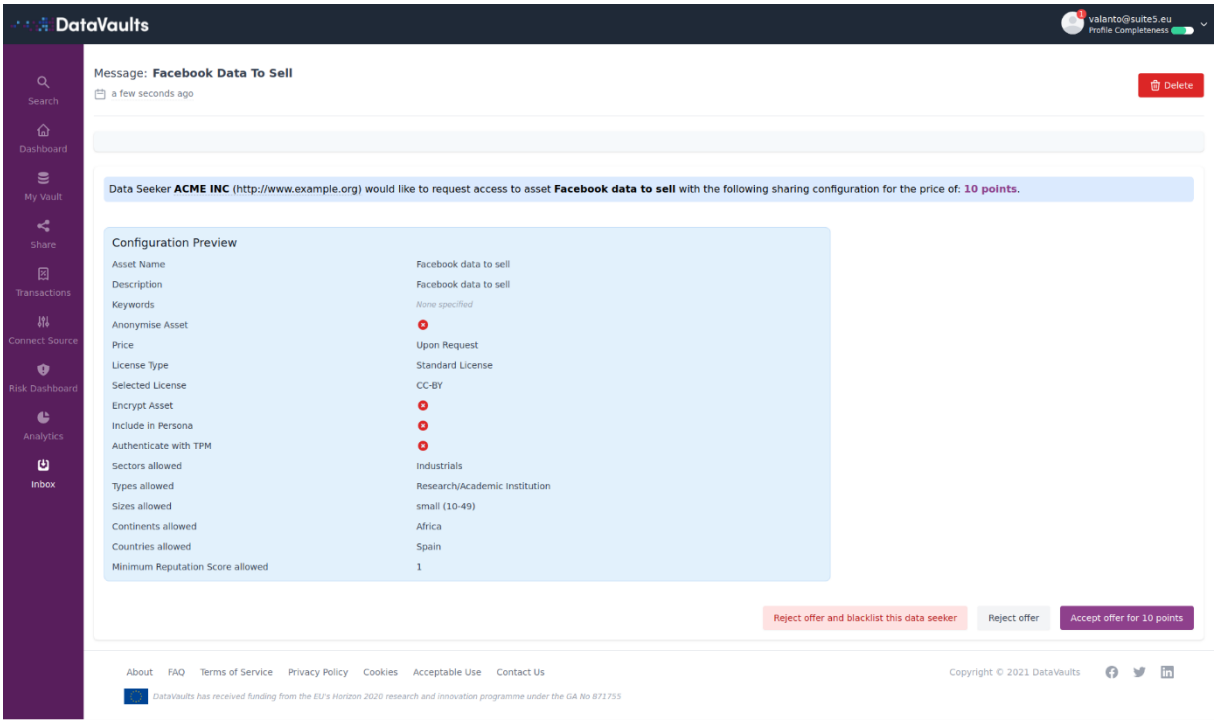


Figure 15: Request for Dataset

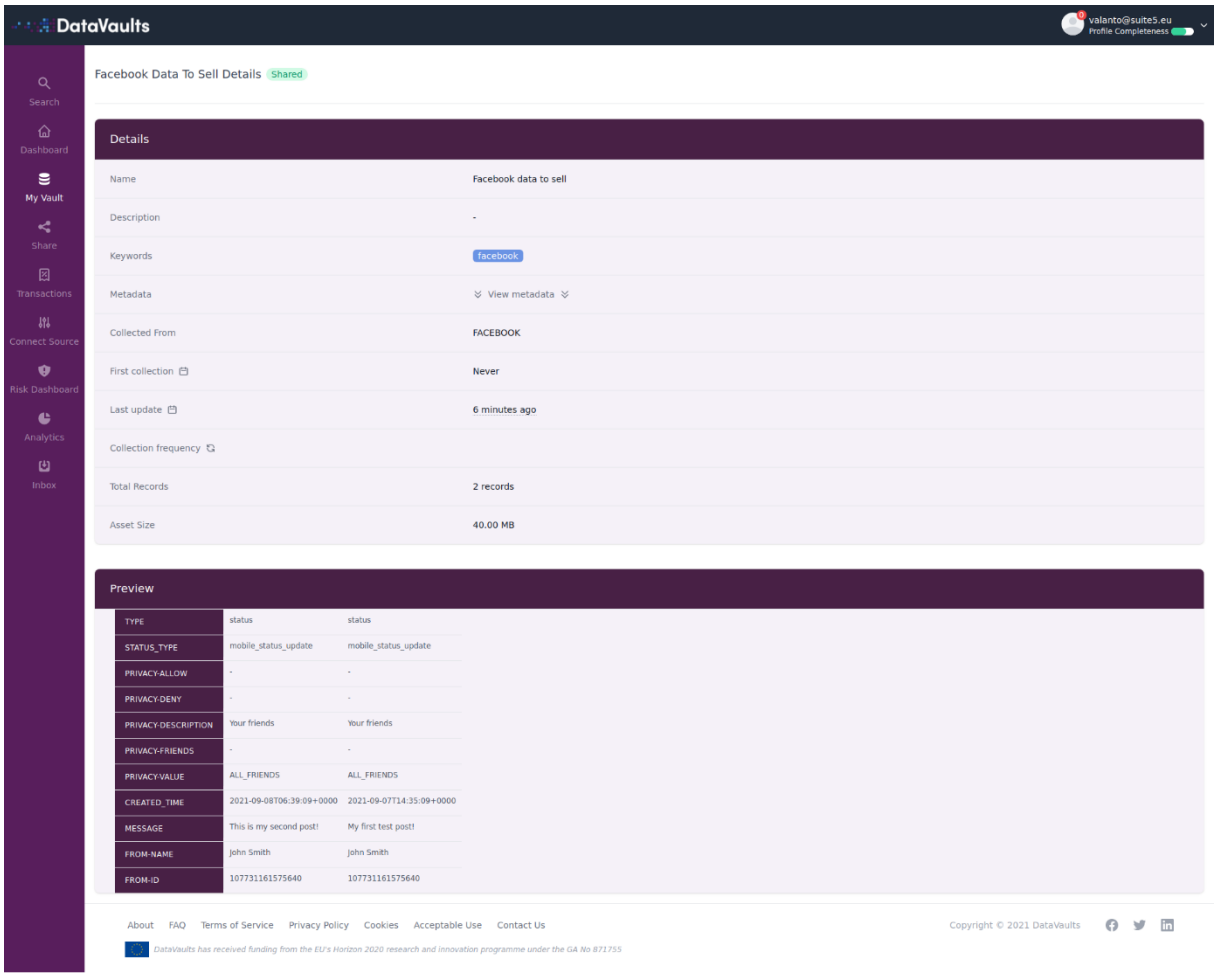


Figure 16: Asset View

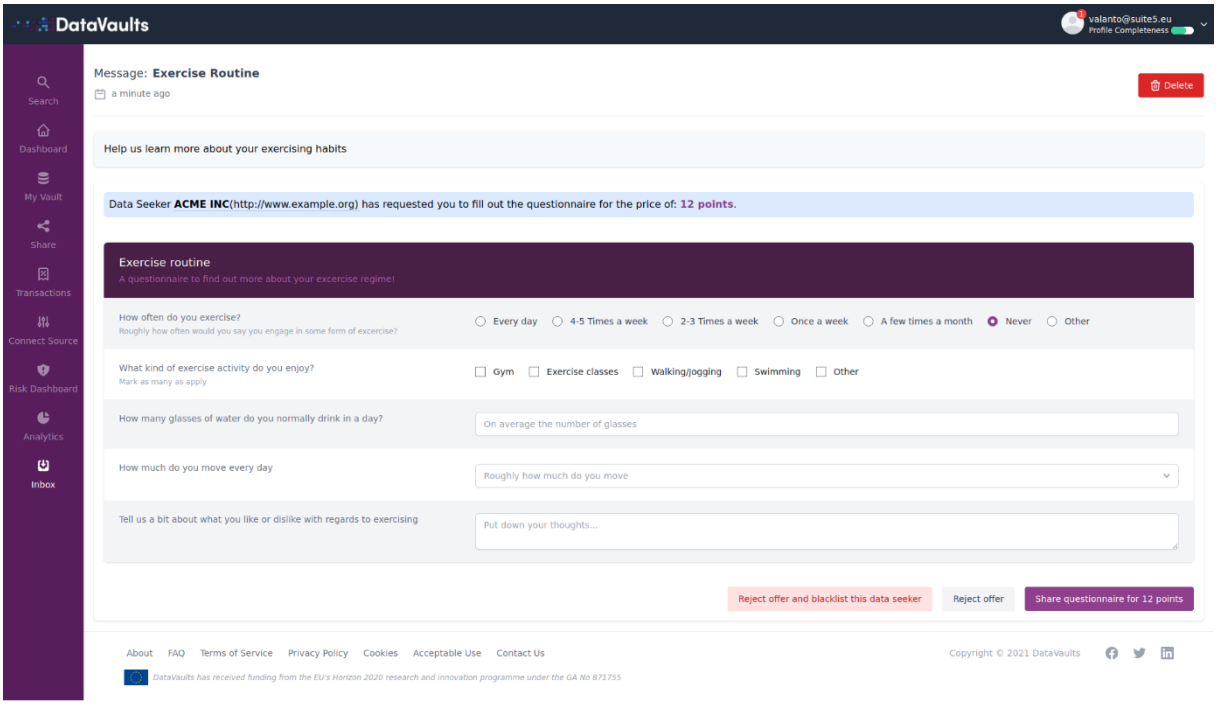
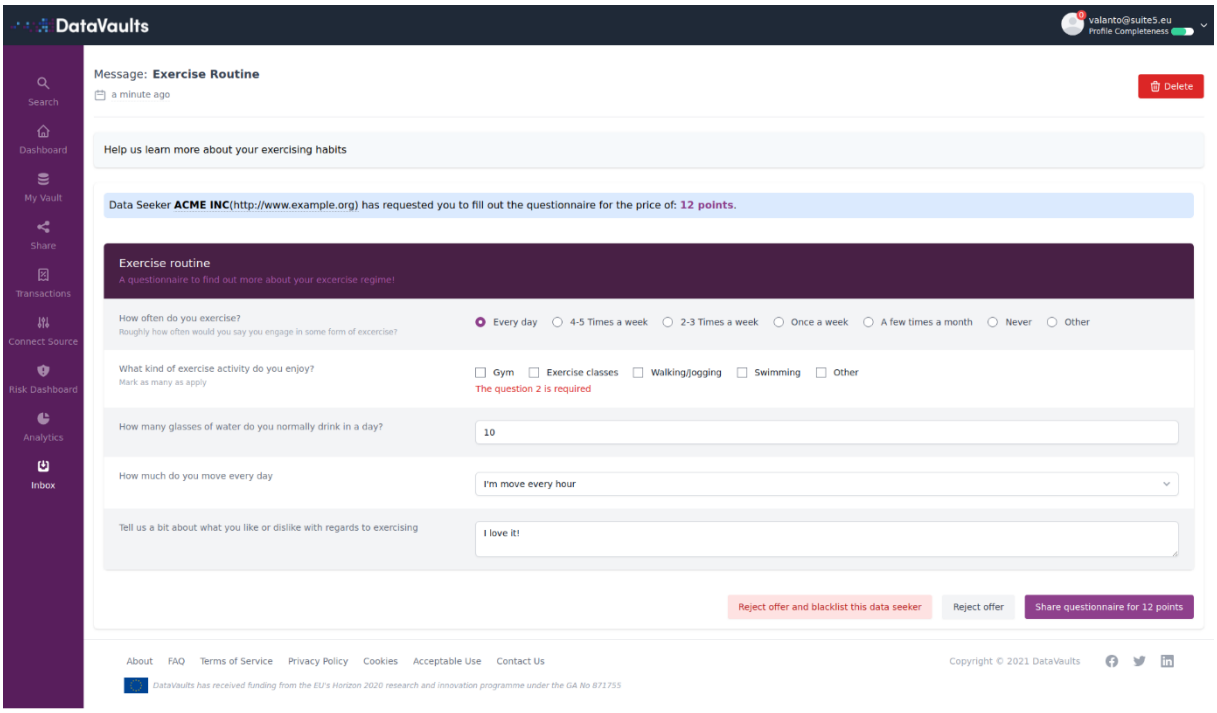


Figure 17: Request for a Questionnaire



The user can always check the transactions related with the data that owns in DataVaults, as shown in Figure 19 and the Transactions Analytics page, as shown in Figure 20.

Figure 18: Participation to a Questionnaire

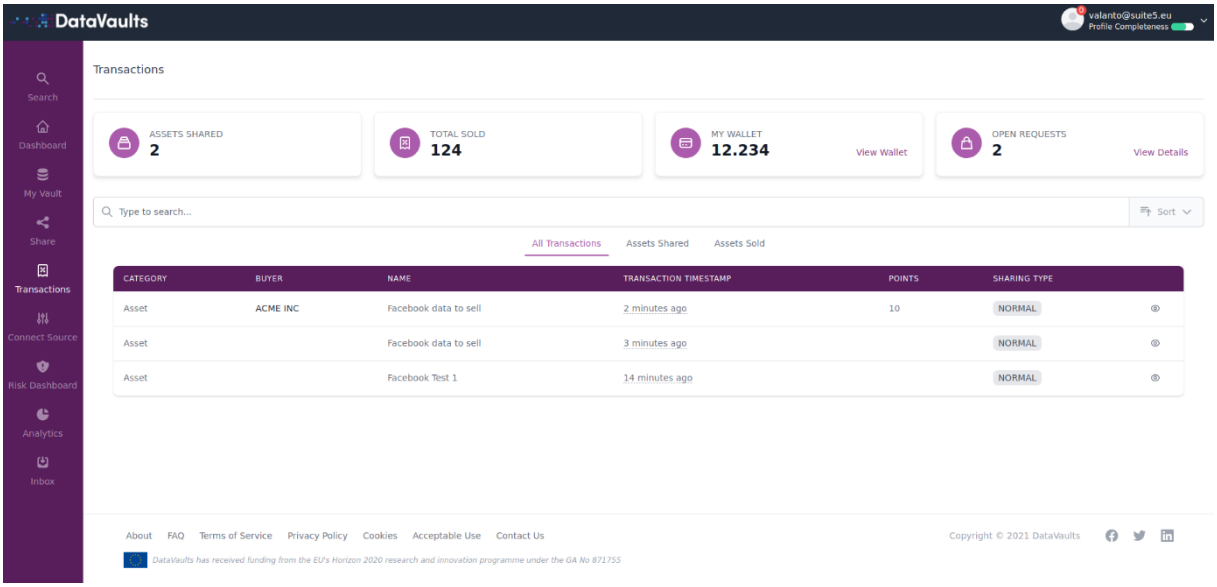


Figure 19: List of Transactions

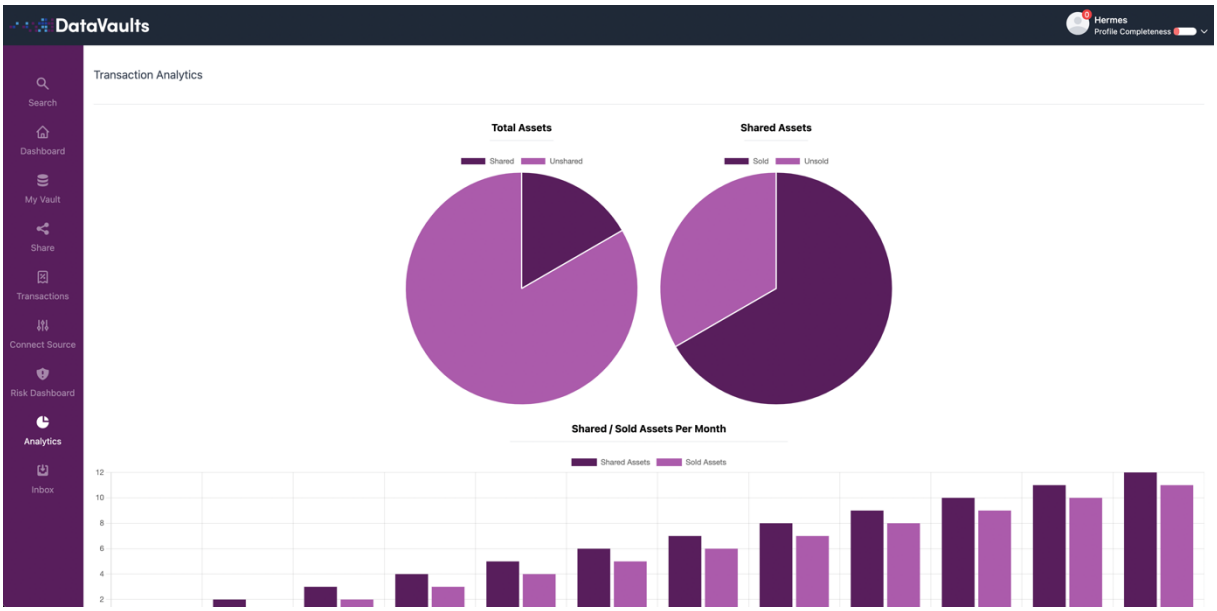


Figure 20: Transactions Analytics Page

3.1.2 Cloud Platform

In addition to the provided APIs and Backbone services, the Cloud Platform provides version 0.5 of the frontend that is used by the Data Seekers, as shown in Figure 21 to Figure 31.

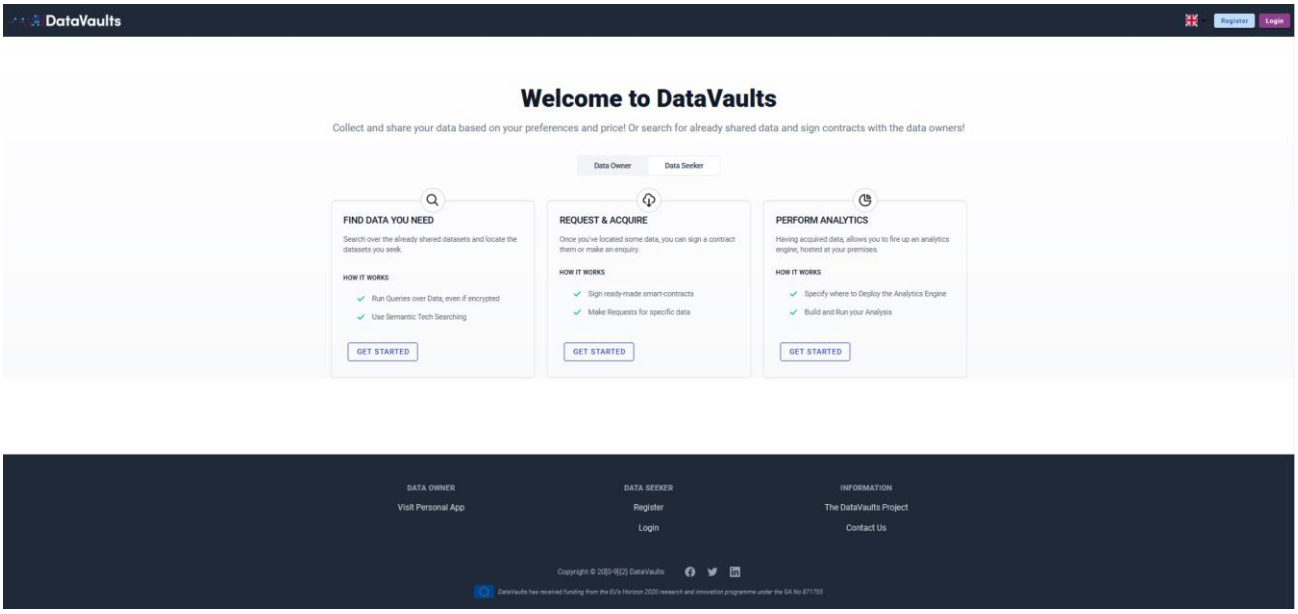


Figure 21: Cloud Platform Landing Page

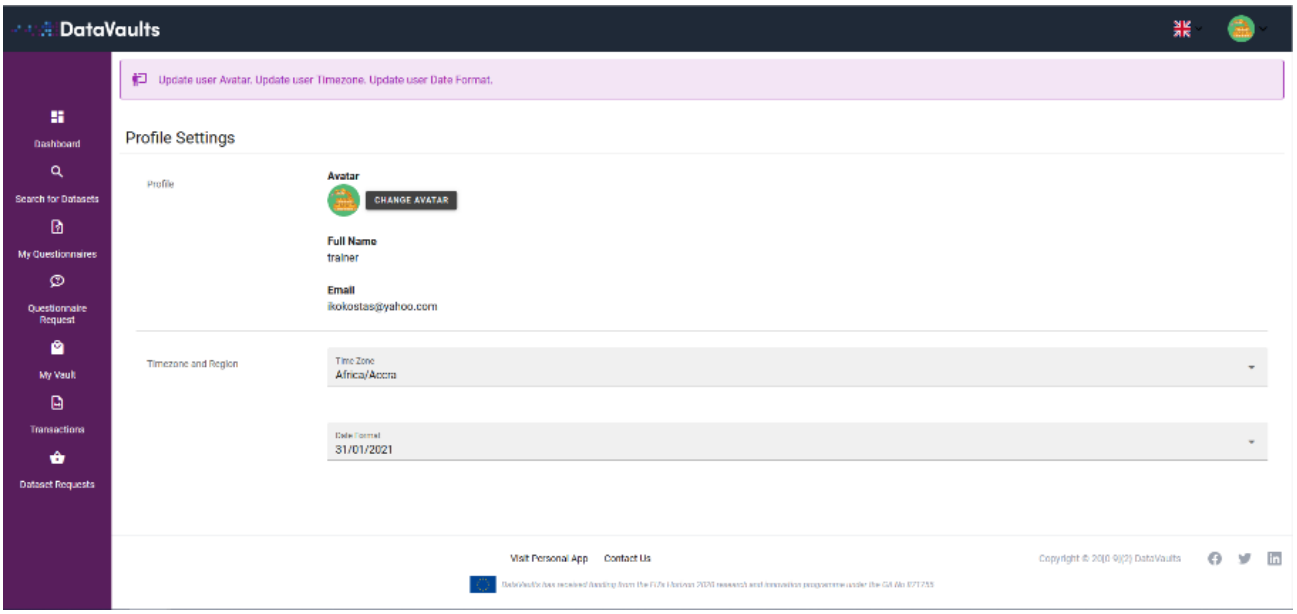


Figure 22: Data Seeker Profile

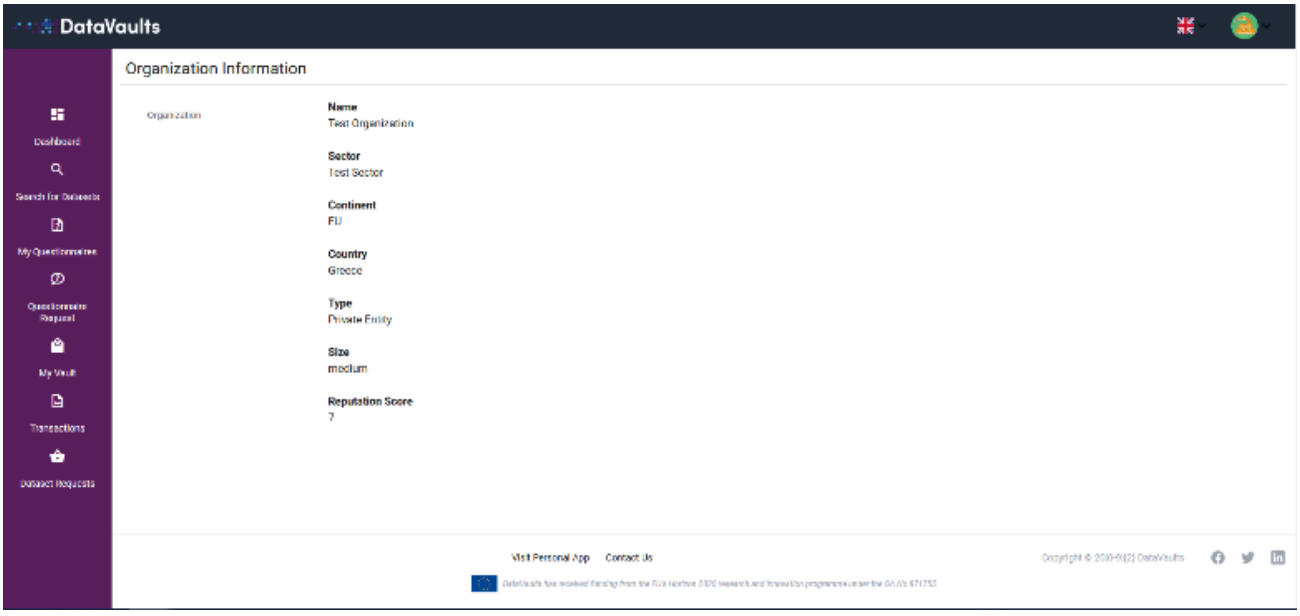


Figure 23: Data Seeker Organization Profile

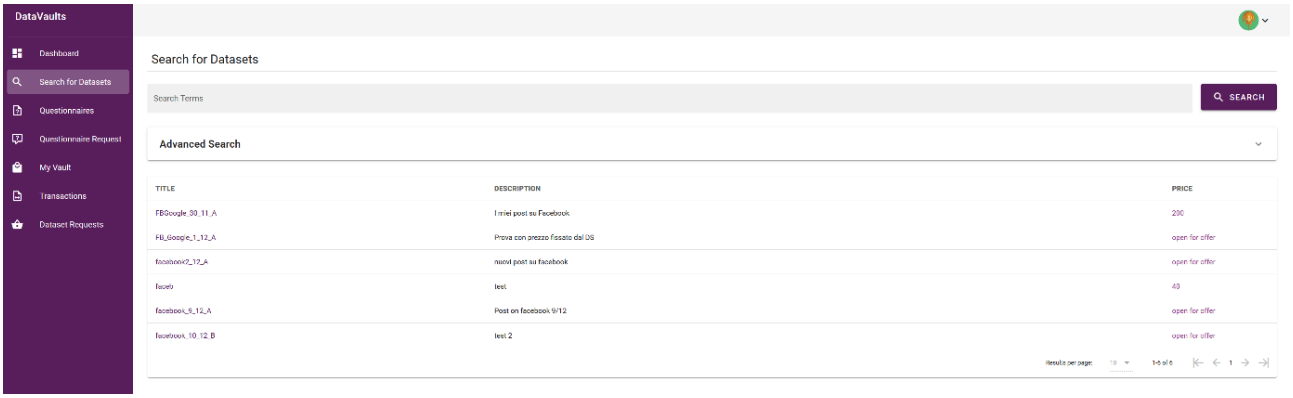


Figure 24: Searching for datasets in the Query Builder

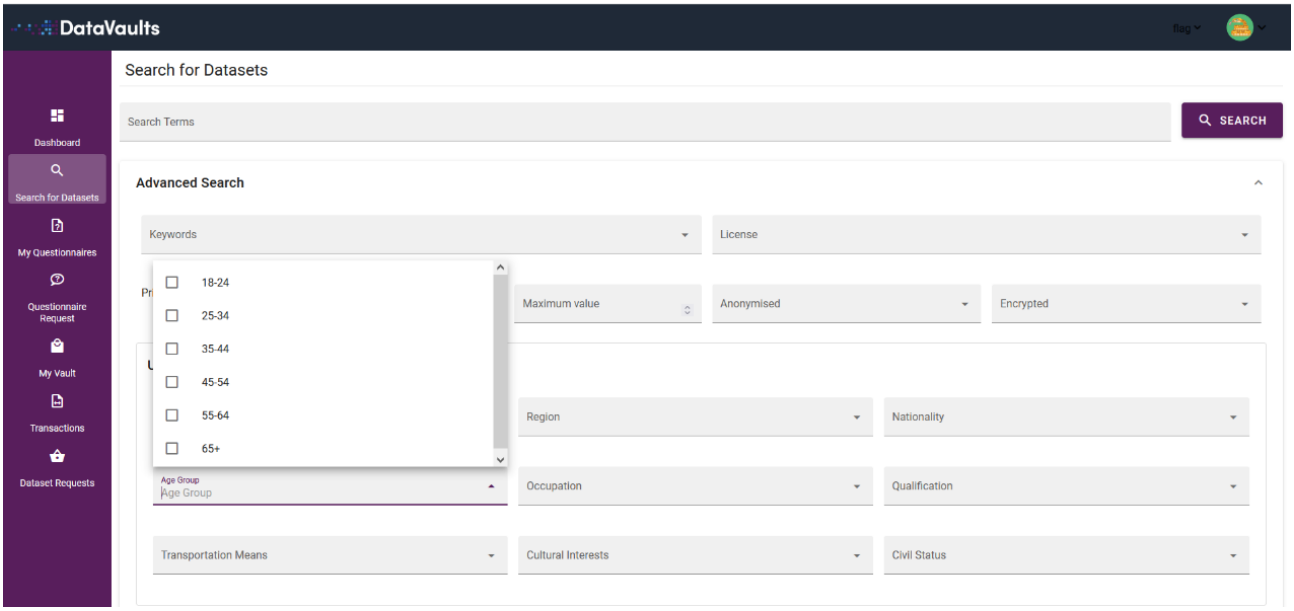


Figure 25: Advanced search

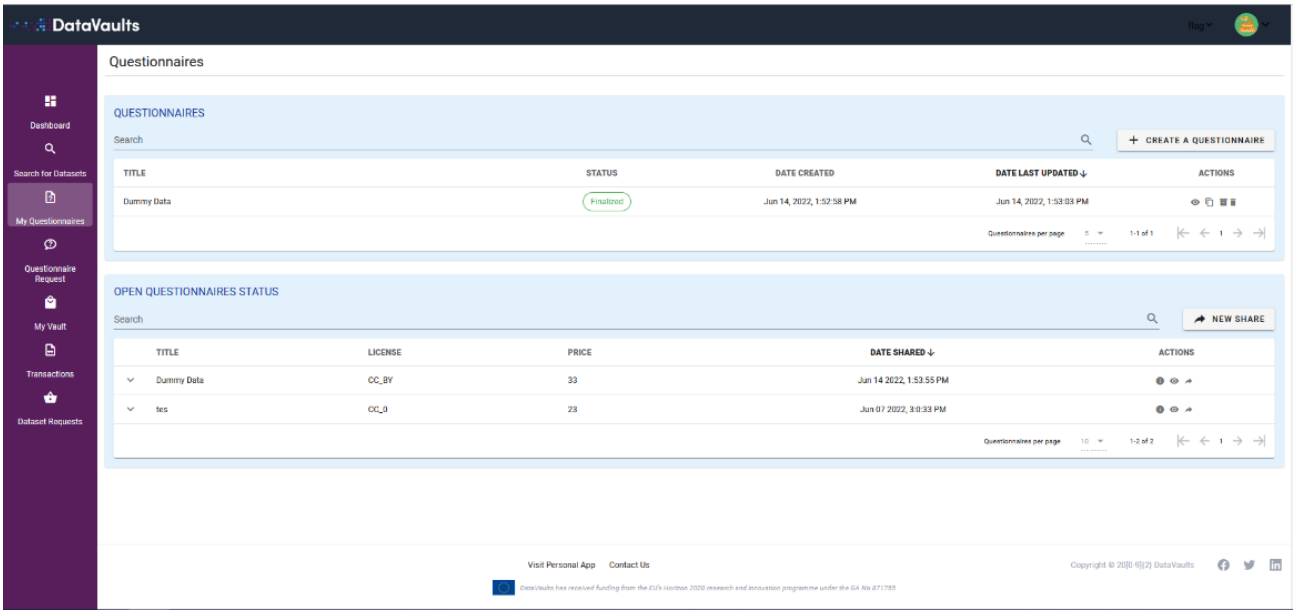


Figure 26: List of Questionnaires

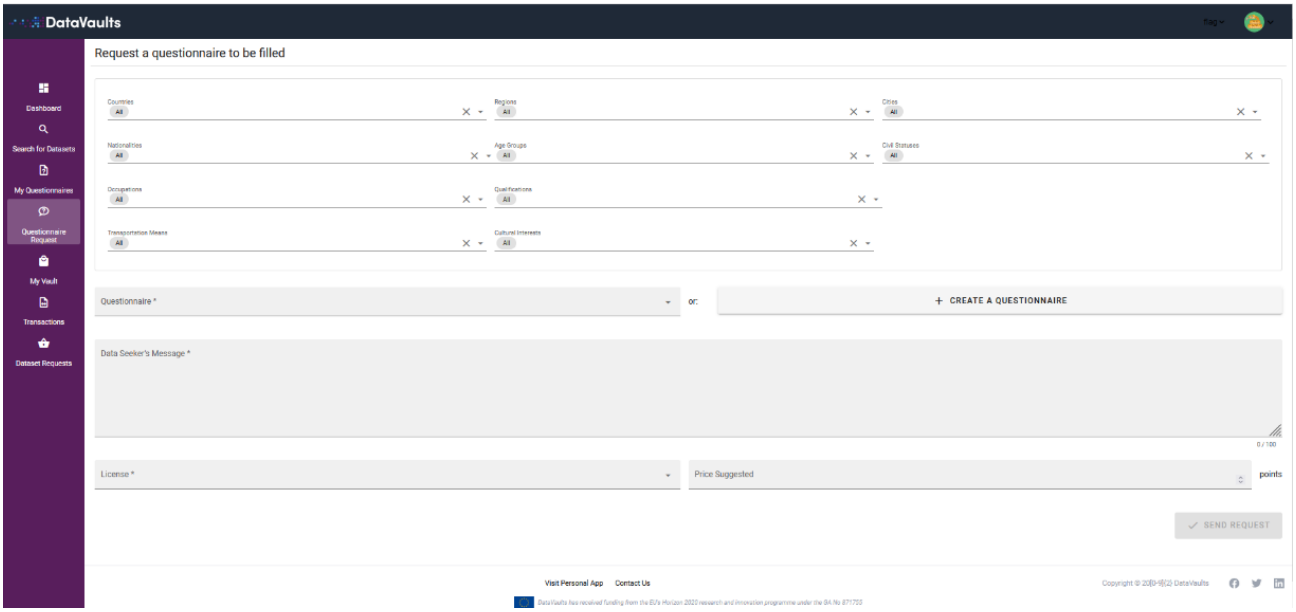


Figure 27: Request for a questionnaire

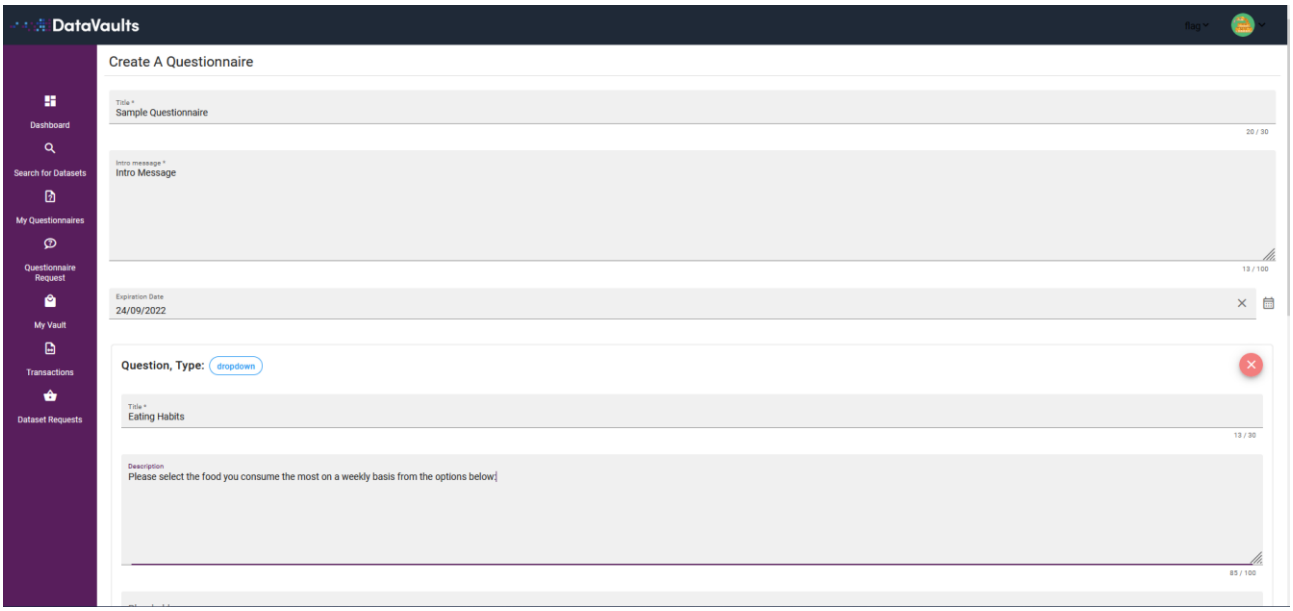


Figure 28: Creation of a questionnaire

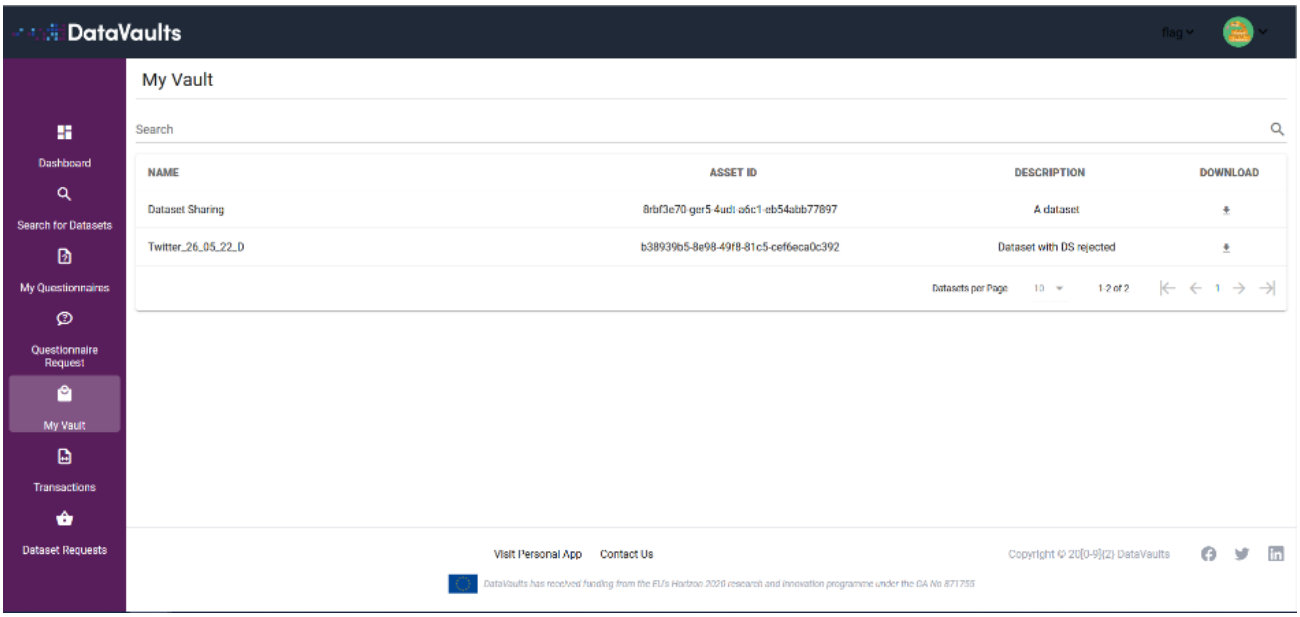


Figure 29: Data Seeker's Vault

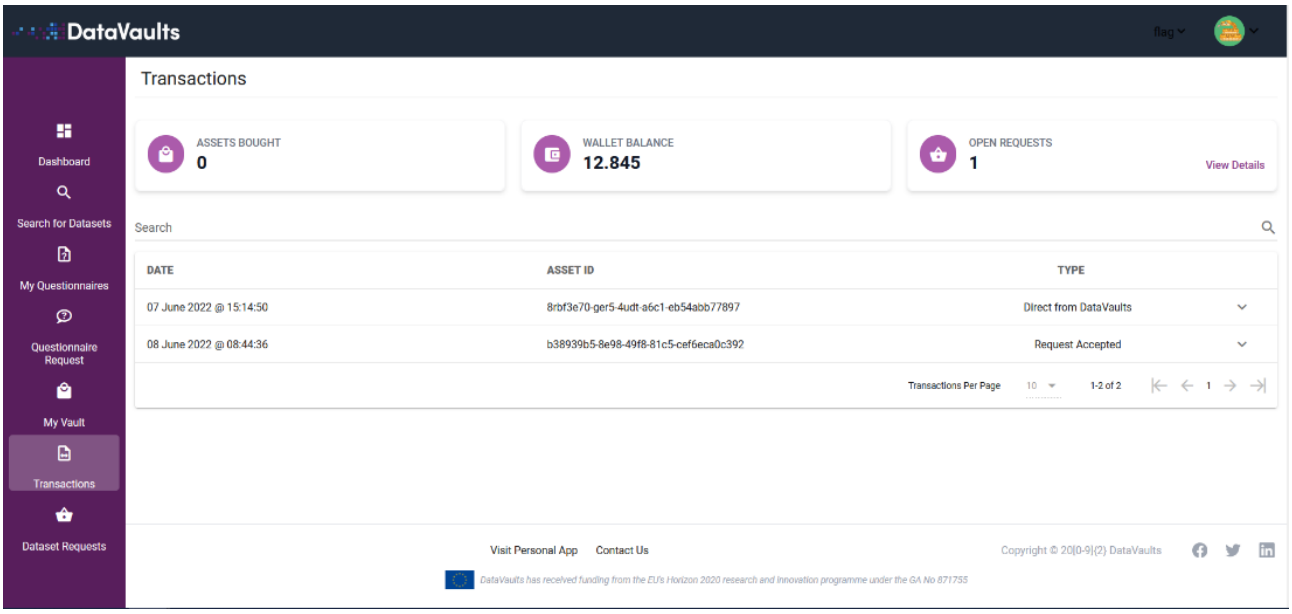


Figure 30: Transactions

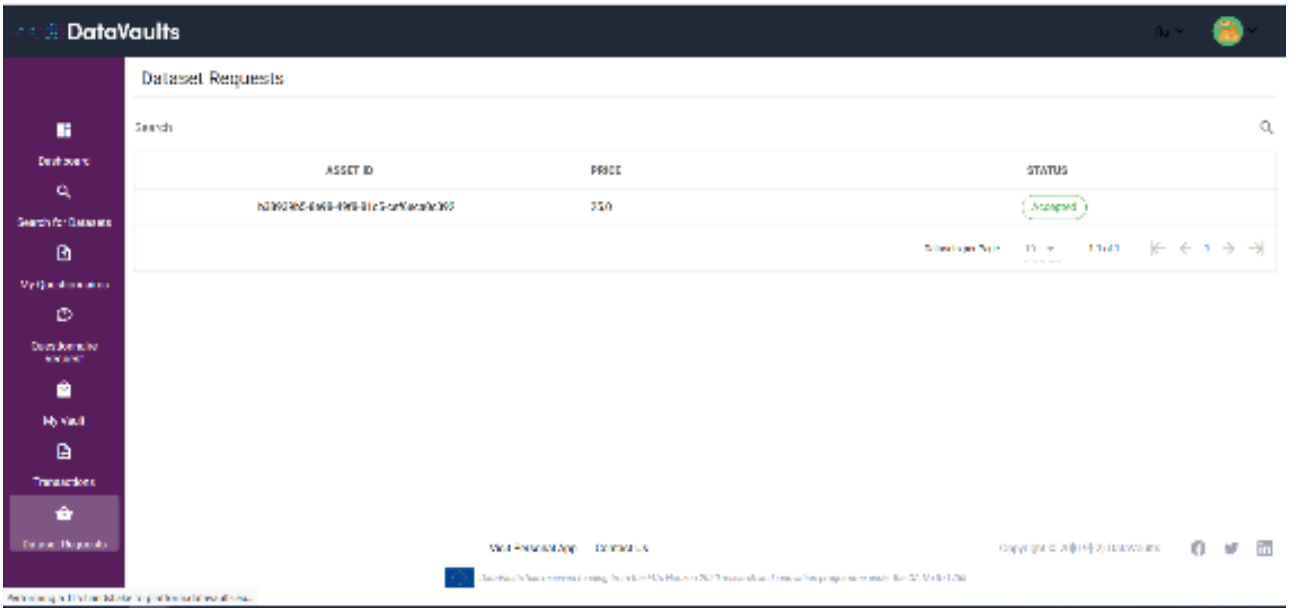


Figure 31: Dataset requests

3.1.3 Secure Analytics Playground

This section of the deliverable describes the possible updates of the SEAS component in terms of frontend functionality. Atos has no updates for its component in this period for the frontend, the main job has been focused on the backend, so the below text has no difference from what is written in Deliverable 5.4 [4].

Secure Analytics Playground provides a first version of the frontend 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.

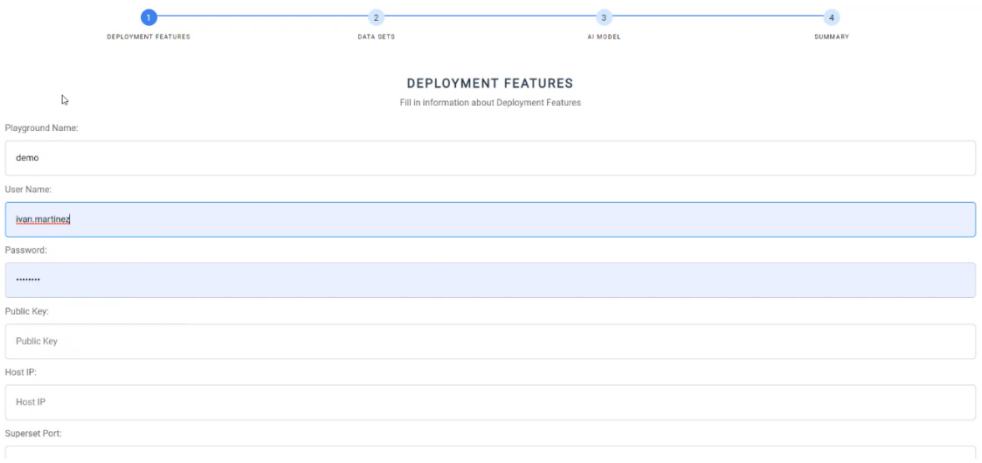


Figure 32: Configure Deployment of SEAS

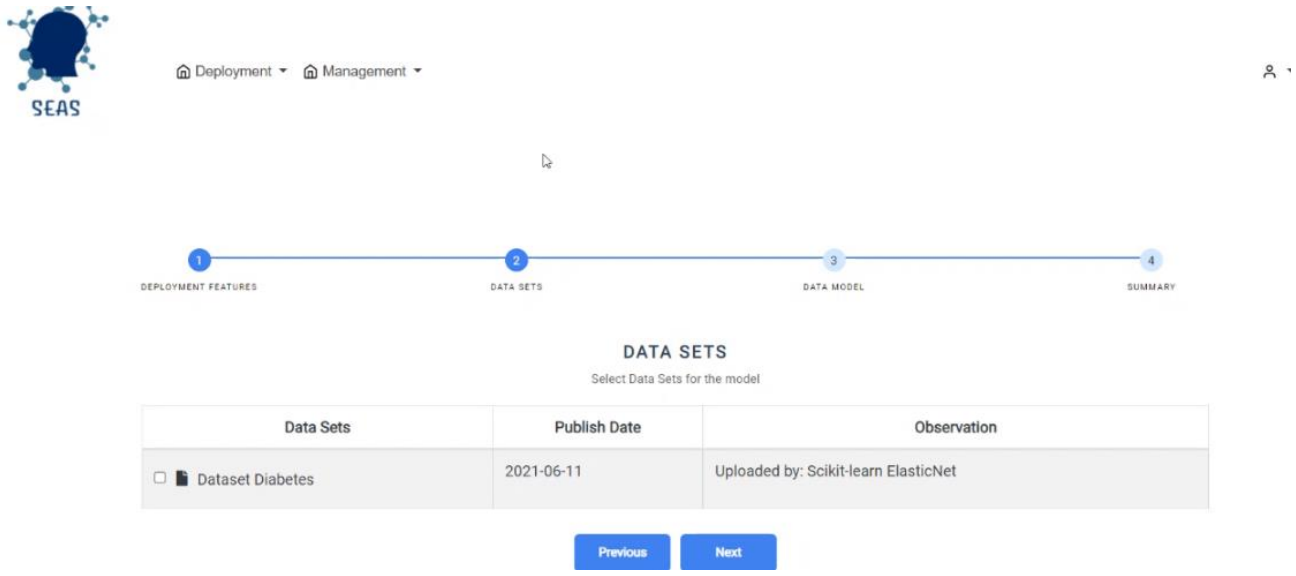


Figure 33: Configure Datasets for SEAS (manually uploaded for version 0.5)

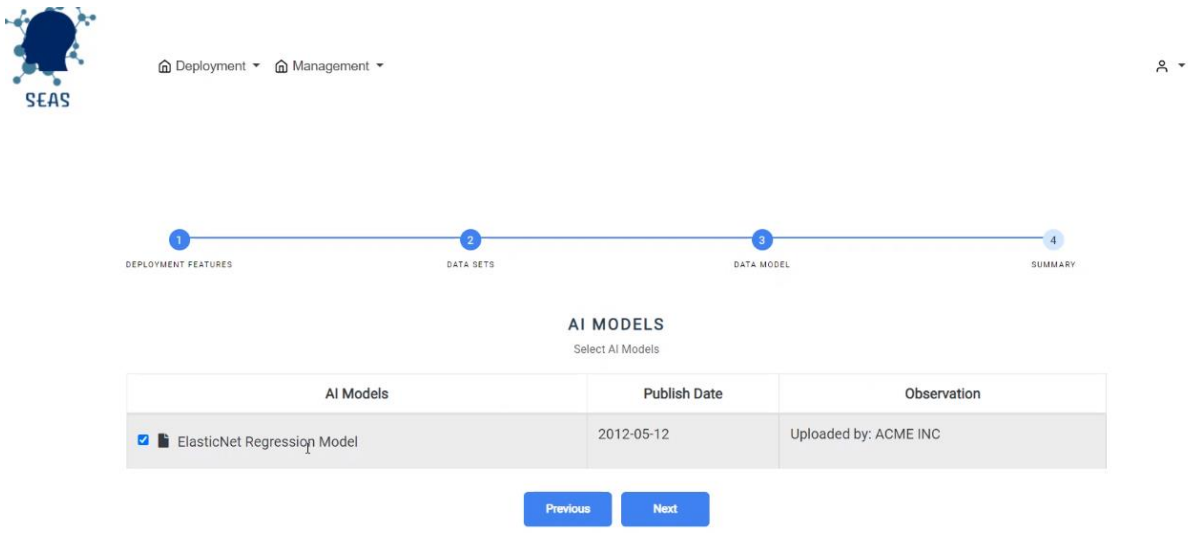


Figure 34: Selection of AI Model for SEAS

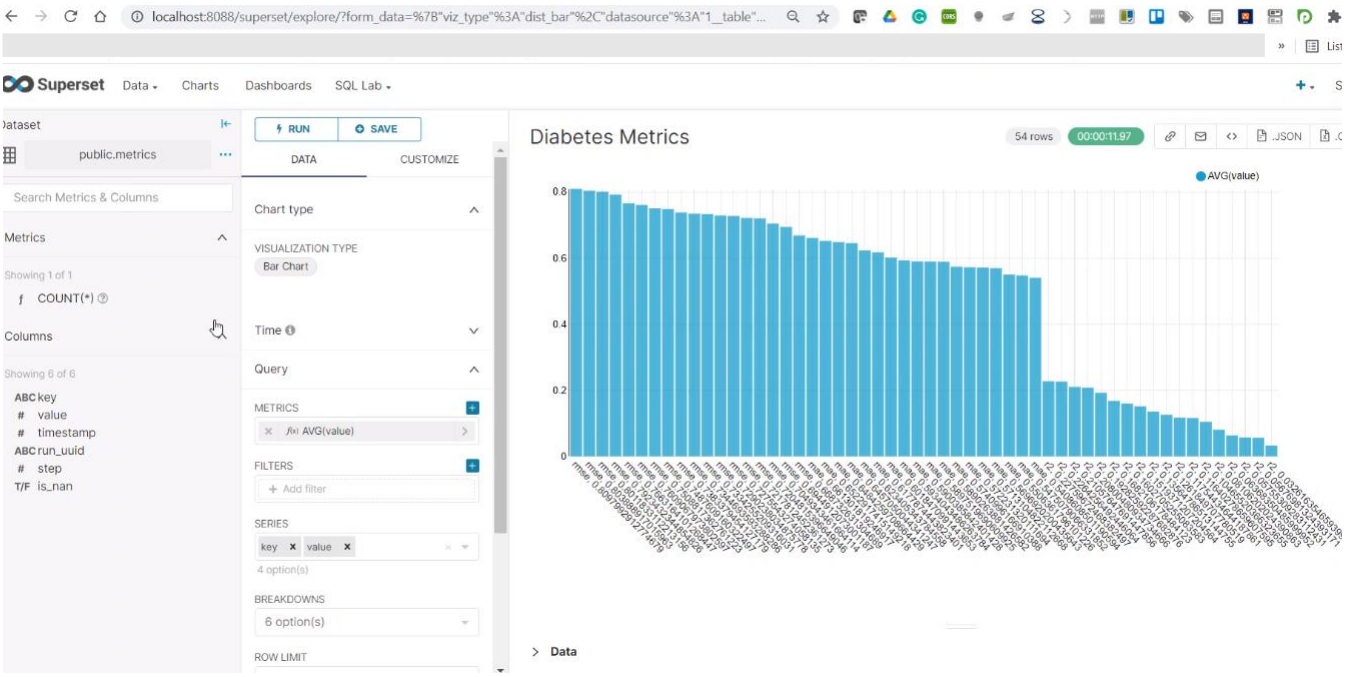


Figure 35: Local Execution Of SEAS

3.2 PLATFORM INSTALLATION

For each component, individual installation instructions have been provided in Section 2.1. Since all components have been dockerized, the main prerequisite to install the DataVaults platform is to have Docker and Docker-Compose installed in all the VMs. The complete list of requirements is:

- 8 VMs with at least 8GB Ram and 4 VCPUs each for the DLT Engine
- 1 VM with at least 4GB Ram and 2 VCPUs for the Identity Manager
- 1 VM with at least 8GB Ram and 2 VCPUs for the Cloud Platform
- 1 VM with at least 8GB Ram and 2 VCPUs for the Personal App
- 1 Android phone with access to the Google Play Store for the Mobile App

Inside the Cloud Platform VM, the list of services that need to be initialized via their respective docker-compose files, as described in Section 2.1.1 is:

- Cloud Platform Backbone
- Persona Generator
- Access Policy + ABE Engine
- Query Builder
- Secure Analytics Playground

Similarly, inside the Personal App VM, the list of services that need to be initialized via their respective docker-compose files, as described in Section 2.1.2 is:

- Personal App Backbone

- Private Wallet
- Data Fetcher & Transformator
- Anonymiser

4 TECHNICAL ASSURANCE

4.1 CONTINUOUS INTEGRATION

To assist the development and integration of the platform components, as well as to ensure the quality of the developed platform, various tools and methodologies have been used. The main tool used from the consortium was GitLab; GitLab has been used not only for the code repository capabilities but also for the Continuous Integration and Continuous Delivery in the frame of the project.

We tried to follow a prevention-based approach on the code quality part during the development and integration of the project's components. Towards this, we suggested using SonarQube and SonarLint, while we increased the importance of the security testing and used security training and guidelines. Finally, in addition to the usage of GitLab for the management of issues, the usage the project's Slack channel for day communication (with over 7500 messages exchanged), also helped us to achieve this goal.

More details about the usage of these tools are provided in the subsections below.

4.1.1 Components Availability

All components of the DataVaults Platform have been developed and uploaded in the Gitlab repository of the project[4], an overview of which is presented in Figure 36.

[4] https://gitlab.ubitech.eu/cs3/datavaults_cloud_platform

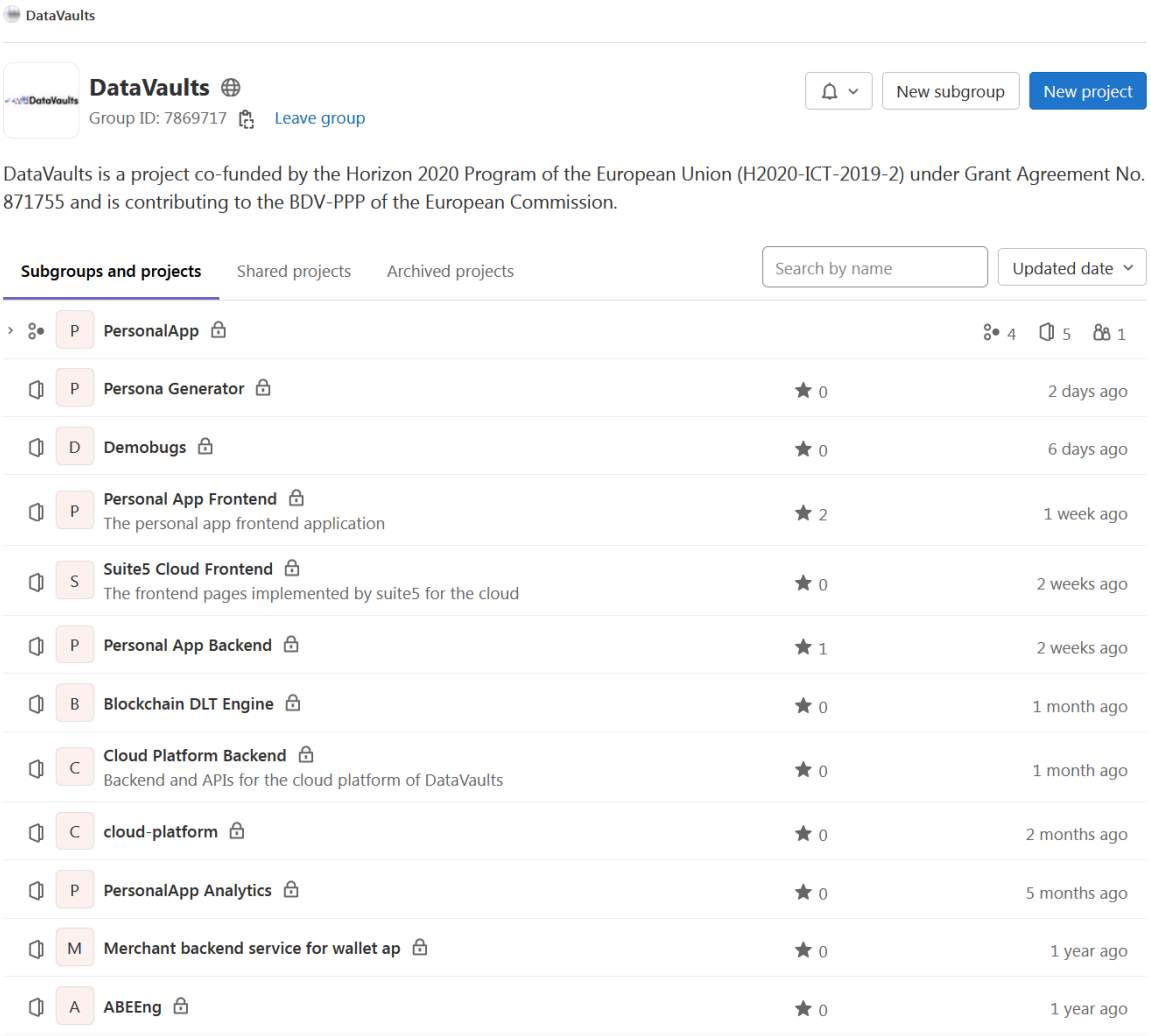


Figure 36: DataVaults Gitlab Repository

This general repository contains subfolders and subrepositories for all the components of the DataVaults platform, each with their own issues and CI management, which will be explained in the following sections. The repository of the Cloud Platform Backbone is presented in Figure 37 as an example of a typical project repository.

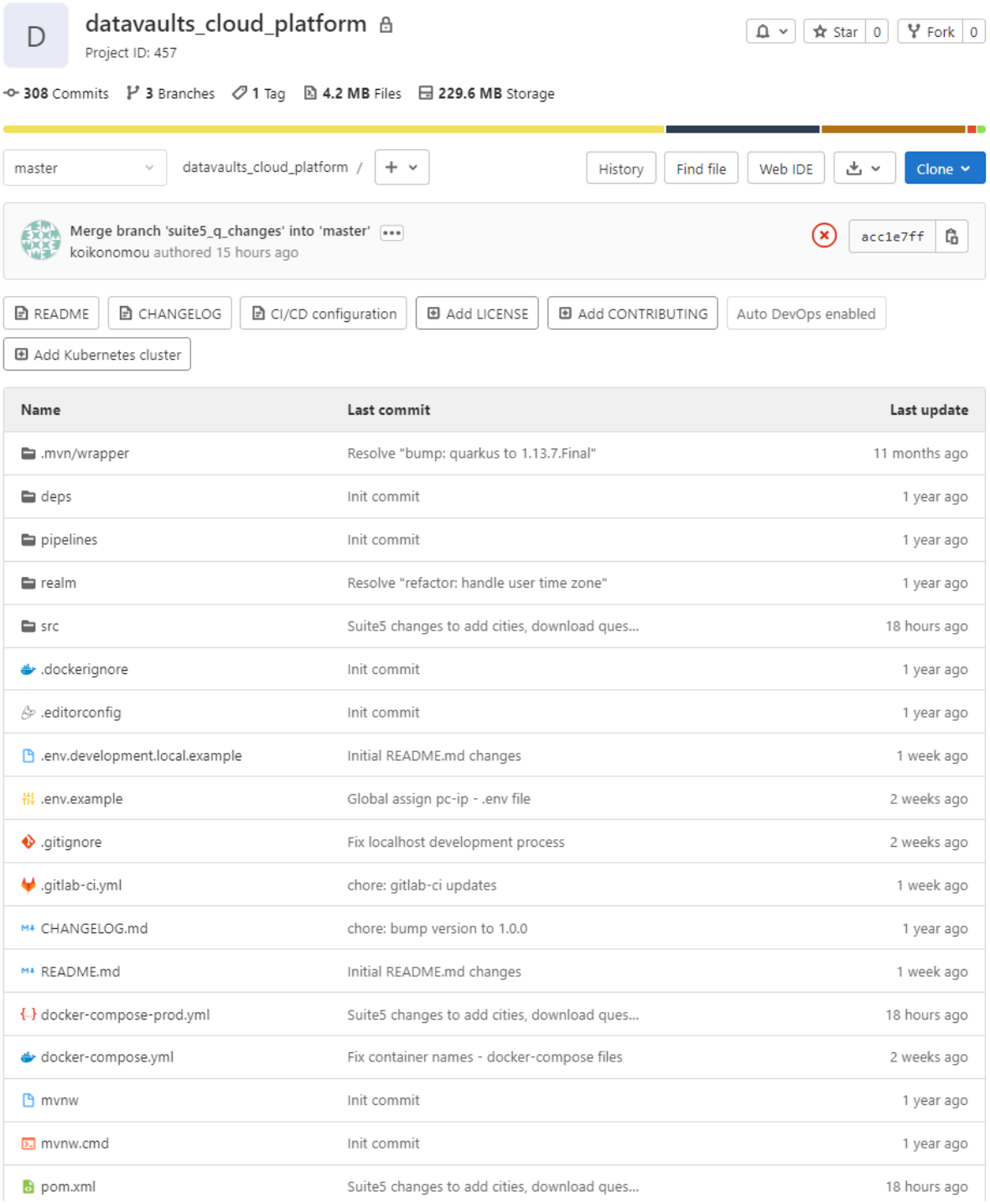


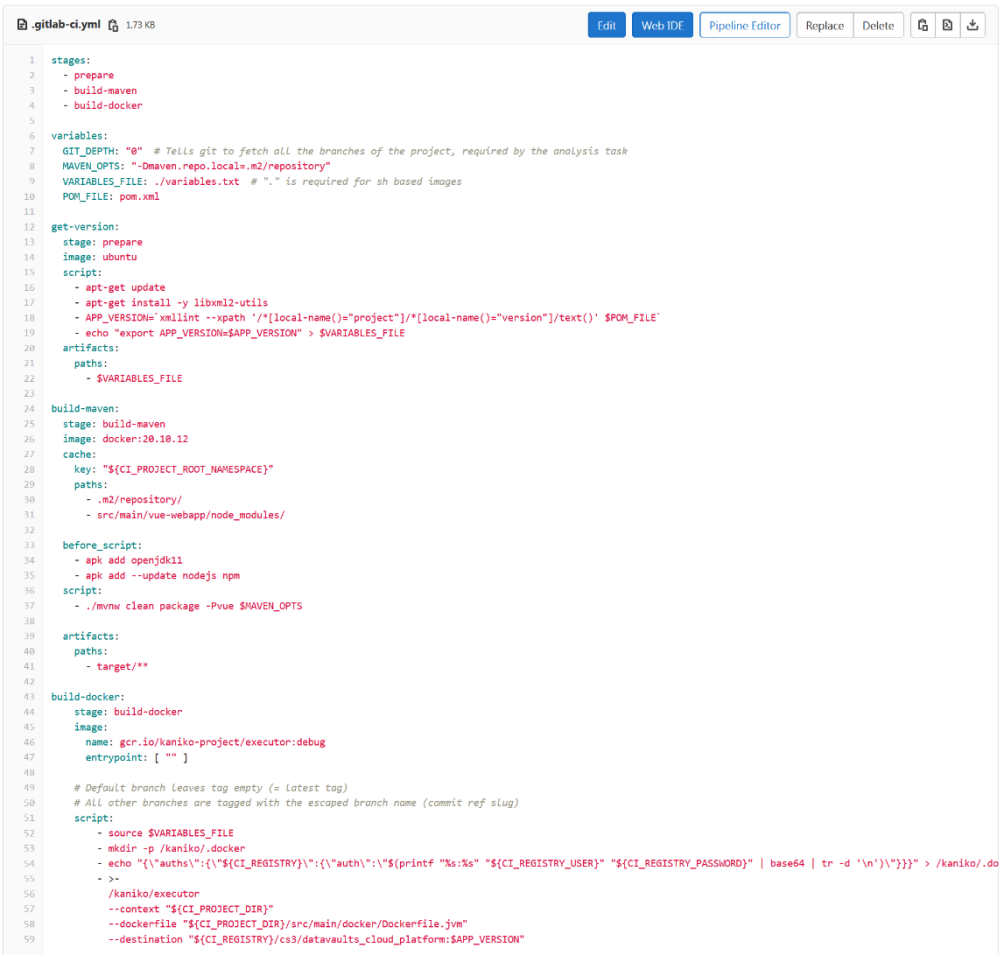
Figure 37 DataVaults Cloud Platform Repository

4.1.2 Continuous Integration Flow

GitLab allows the establishment of automated pipelines that manage the Continuous Integration and Continuous Delivery of the whole project. For this reason, DataVaults uses Gitlab for the CI/CD flow, as well as Kubernetes for CI/CD jobs.

GitLab pipelines can be split into multiple stages, and each of them can be further split into jobs, such as Build, Test, Review, Performance, etc. Those stages are defined in a file called

gitlab-ci.yml that handles the configuration of the pipelines and the jobs contained in them. An example of such a file is depicted in Figure 38.



```
stages:
  - prepare
  - build-maven
  - build-docker

variables:
  GIT_DEPTH: "0"  # Tells git to fetch all the branches of the project, required by the analysis task
  MAVEN_OPTS: "-Dmaven.repo.local=.m2/repository"
  VARIABLES_FILE: ./variables.txt  # "." is required for sh based images
  POM_FILE: pom.xml

get-version:
  stage: prepare
  image: ubuntu
  script:
    - apt-get update
    - apt-get install -y libxml2-utils
    - APP_VERSION=`xmllint --xpath '/*[local-name()="project"]/*[local-name()="version"]/text()' $POM_FILE`
    - echo "export APP_VERSION=$APP_VERSION" > $VARIABLES_FILE
  artifacts:
    paths:
      - $VARIABLES_FILE

build-maven:
  stage: build-maven
  image: docker:20.10.12
  cache:
    key: "${CI_PROJECT_ROOT_NAMESPACE}"
    paths:
      - .m2/repository/
      - src/main/vue-webapp/node_modules/

  before_script:
    - apk add openjdk11
    - apk add --update nodejs npm
  script:
    - ./mvnw clean package -Pvue $MAVEN_OPTS

  artifacts:
    paths:
      - target/**

build-docker:
    stage: build-docker
    image:
      name: gcr.io/kaniko-project/executor:debug
      entrypoint: [ "" ]

    # Default branch leaves tag empty (= latest tag)
    # All other branches are tagged with the escaped branch name (commit ref slug)
    script:
        - source $VARIABLES_FILE
        - mkdir -p /kaniko/.docker
        - echo "{\"auths\":{\"${CI_REGISTRY}\":{\"auth\":\"$(printf "%s:%s" "${CI_REGISTRY_USER}" "${CI_REGISTRY_PASSWORD}" | base64 | tr -d '\n')\"}}}" > /kaniko/.do
        - >-
          /kaniko/executor
          --context "${CI_PROJECT_DIR}"
          --dockerfile "${CI_PROJECT_DIR}/src/main/docker/Dockerfile.jvm"
          --destination "${CI_REGISTRY}/cs3/datavaults_cloud_platform:$APP_VERSION"
```

Figure 38 gitlab-ci.yml file

This file handles the CI of the Cloud platform and the stages included in this specific service are three: prepare and build-maven and build-docker. In each stage, the necessary images are used; the use of scripts is necessary, in addition to predefined GitLab variables which are available in every CI/CD pipeline. The first stage handles the preparation of the environment by downloading the necessary libraries and also setting the correct variables for the pipeline. During the second stage the image of the cloud platform is built, utilizing openjdk, nodejs and maven. Finally, the third stage ensures that the docker image is correctly built and uploaded to the container registry of the particular Gitlab repository.

4.1.3 Issues Management

Regular reporting of any issues identified, as well as any additional features which need to be developed, is a major aspect of the integration process in a collaborative environment. As such, it is important to the DataVaults Continuous Integration process. To achieve this, we used Gitlab's native issue management functionality. Figure 39 depicts the issue management process in DataVaults.

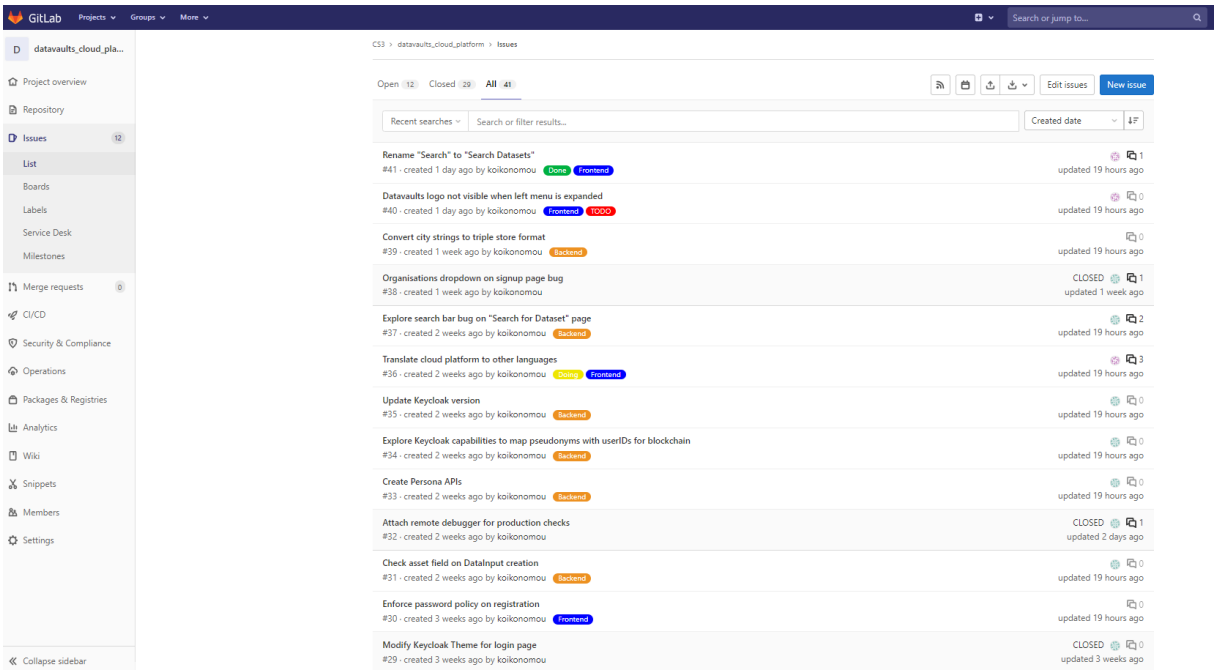


Figure 39: Issue management in DataVaults gitlab

4.2 TESTING

The Integration testing could be understood as a way to test the module one by one and test the behavior as a combined unit. The unit testing is something that is done before the integration testing. Once the modules are unit tested, the next step is to check the combinational behavior.

One of the common problems in testing is the frequent requirements change and combined with the fact that many developers do not perform the required unit testing before deploying changes makes integration testing very important.

In the scope of DataVaults project, integration testing is done to test the different components and modules “combined” that are involved in the entire deployment of the Platform. As is well known, modules can work fine induvial, but this doesn’t mean that issues are not going to appear when we integrated them all together.

The integration testing shouldn’t be understood as a way to test the application at the end of the deployment. This integration test must be done simultaneously with the development and test of the different releases of the DataVaults platform. This test will be especially focused on Version 0.5 and 1.0 of the DataVaults platform.

One of the most important steps to create an integration test is to understand the platform Architecture, the below graph is taken from deliverable 5.2 [2] show how this architecture works in DataVaults case. The DataVaults platform is mainly composed of two layers, the

Cloud, and the Edge. Each layer has its own modules that are also interconnected between them as we can see from Figure 40.

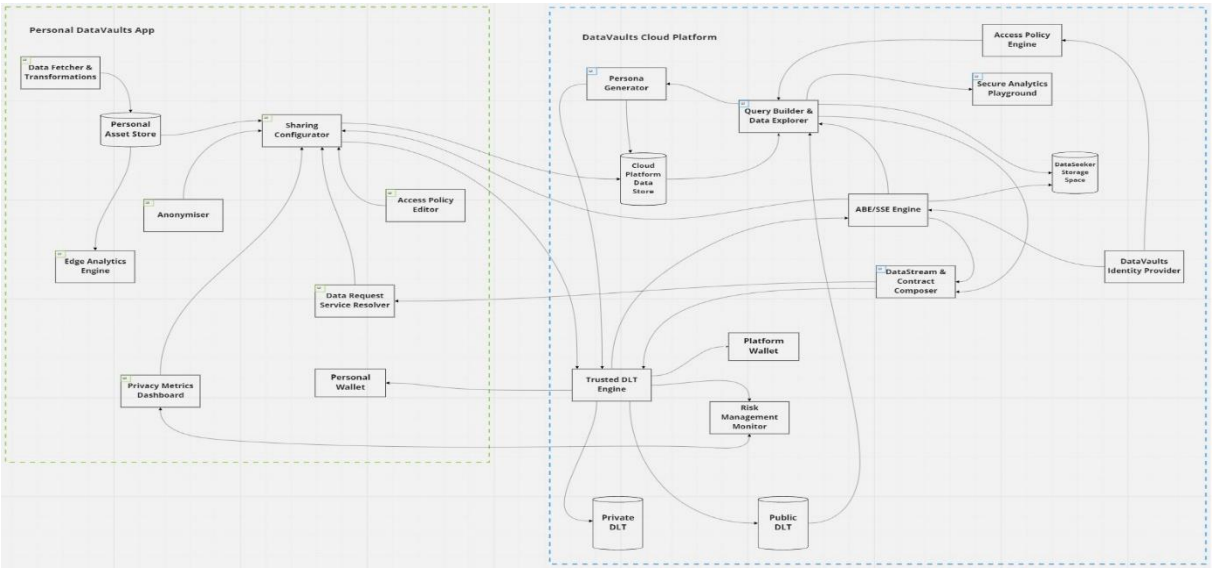


Figure 40 - DataVaults Architecture

A total of 9 different partners with 20 modules are involved in the DataVaults platform. 9 of these modules are part of the edge layer while 11 involved the cloud layer. Some of these modules apart from interacting between them also interact with some third-party tools or APIs which also need to be tested that the data accepted by that API is correct and the response generated is also as expected.

Our approach to creating the integration test is based on created "**Test Cases**". Each "**Test Case**" covers certain "**Functionalities**", which in turn use "**modules**" defined in the DataVaults architecture. Figure 41 shows this approach.

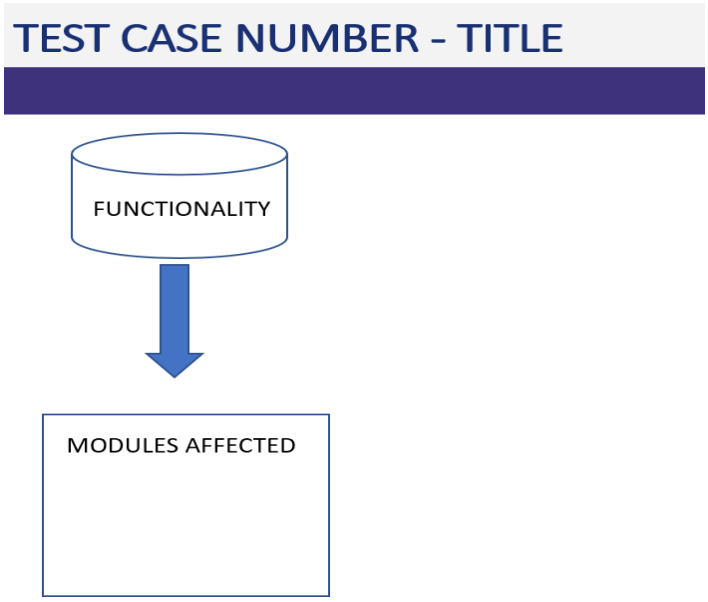


Figure 41 - Test Case Approach

One of the first steps that we have arranged to create the integration test is to understand what each of the modules of the different layers does and how they interact between them. With a clear understanding of the Architecture and the modules we have defined the Test Cases that we want to test.

A total of 8 Test Cases has been implemented with their respective functionalities and associated modules:

Test case 1: USER MANAGEMENT (Table 1)
Test case 2: DATA COLLECTING/INGESTION (Table 2)
Test case 3: DATA COLLECTION & SHARING CONFIGURATION MANAGEMENT (Table 3)
Test case 4: DATA TRANSFORMATION AND ENRICHMENT (Table 4)
Test case 5: DATA ASSET SHARING EXECUTION (Table 5)
Test case 6: DATA ASSETS PURCHASE AND PERSONA GENERATION (Table 6)
Test case 7: DATA ASSETS PURCHASE FOR INCOMPLETE SHARING CONFIG (Table 7)
Test case 8: CLOUD DATA ANALYTICS (Table 8)

Test case 1 USER MANAGEMENT				
Functionality	USER CREATION	USER LOGIN	USER ENRICHMENT	USER LOG OUT
Modules Affected	• Personal App Backbone • Personal Wallet • Trusted DLT Engine	• Personal App Backbone	• Personal App Backbone • Personal Wallet • Trusted DLT Engine	• Personal App Backbone

Table 1 - Test Case 1: User Management

Test case 2 DATA COLLECTING/INGESTION		
Functionality	VIEW AVAILABLE DATA SOURCES	EXECUTE DATA COLLECTION CONF
Modules Affected	• Data Fetcher	• Personal App Backbone • Data Fetcher

Table 2 - Test case 2: Data Collecting/Ingestion

Test case 3 DATA COLLECTION & SHARING CONFIGURATION MANAGEMENT			
Functionality	STORE DATA	SEARCH DATA	CREATE SHARING CONFIGURATION
Modules Affected	• Data Fetcher • Personal App Backbone	• Personal App Backbone	• Personal App Backbone • Sharing Configuration • Anonymizer • Policy Editor • Privacy M Dashboard • Risk management Mon

Table 3 - Test case 3: Data Collection & Sharing Configuration

Test case 4 DATA TRANSFORMATION AND ENRICHMENT		
Functionality	SEARCH AND SELECT AMONG AVAILABLE DATA ASSETS	GENERATE ANALYTICS: ENRICHMENT & TRANSFORMATION
Modules Affected	• Personal App Backbone	• Personal App Backbone • Edge Analytics Engine

Table 4 - Test case 4: Data Transformation & Enrichment

Test case 5 DATA ASSET SHARING EXECUTION			
Functionality	SEARCH/SELECT AMONG AVAILABLE DATA ASSETS	DELETED UNWANTED DATA ASSETS	EXECUTE DATA COLLECTION CONF
Modules Affected	• Data Fetcher	• Personal App Backbone	• Personal App Backbone • Sharing Configuration • Anonymizer • Policy Editor • ABE/SSE Engine • Trusted DLT Engine • Cloud platform Backbone

Table 5 - Test case 5: Data Asset Sharing Execution

Test case 6 DATA ASSETS PURCHASE AND PERSONA GENERATION				
Functionality	SEARCH AMONG AVAILABLE SHARED ASSETS	SELECT DATA ASSETS	BUY DATA ASSET	GENERATE PERSONA
Modules Affected	• Query Builder • Data seeker storage • Cloud Platform Backbone	• Query Builder • Access Policy Engine • ABE/SSE Engine • Trusted DLT Engine	• Platform Wallet • Data Stream Contract Composer • Trusted DLT Engine-Pub/Priv DLT • Personal Wallet • Data request service resolver	• Data seeker storage • Persona Generator • Trusted DLT Engine-Pub/Priv

Table 6 - Test case 6: Data Assets Purchase & Persona Generation

Test case 7 DATA ASSETS PURCHASE FOR INCOMPLETE SHARING CONFIG				
Functionality	SEARCH AMONG AVAILABLE SHARED ASSETS	SELECT DATA ASSETS	BUY DATA ASSET	COMPLETE SHARING CONFIGURATION
Modules Affected	• Query Builder • Data seeker storage • Cloud Platform Backbone	• Query Builder • Access Policy Engine • ABE/SSE Engine • Trusted DLT Engine	• Platform Wallet • Data Stream Contract Composer • Trusted DLT Engine-Pub/Priv DLT • Personal Wallet • Data request service resolver	• Anonymizer • Sharing Configuration • Policy Editor • Privacy M Dashboard • Risk management Mon

Table 7 - Test case 7: Data Assets Purchase for Incomplete Sharing Configuration

Test case 8 CLOUD DATA ANALYTICS			
Functionality	SEARCH /SELECT AMONG AVAILABLE BOUGHT ASSETS	GENERATE ANALYTICS PLAYGROUND	SHARE ANALYTICS

Modules Affected	• Data Seeker Storage	• Secure Analytics Playground Service (SEAS)	• Sharing configurator cloud asset store

Table 8 - Test case 8: Cloud Data Analytics

One of the challenges that we have found during the creation of the test cases is realizing that not all the modules involved in the Platform have an API to expose. These modules that don't expose an API are called by another module internally. The module that exposes an API could be called the "Parent Module" while the one which doesn't expose could be called the "Child module". The way to test these "Child modules" is to call them using these "Parent modules" under certain conditions.

All the Test cases defined were tested with all the technical partners to verify:

1) If their module fits under the functionality described in the test cases.
2) If its module is called by another module, they have checked that both modules are included in the “functionality” where are involved.

As we have mentioned the overall idea of creating these Test Cases is to cover as much as we can all the functionalities of the DataVaults Platform and to test all the modules involved in the platform, Edge (Figure 42) and Cloud (Figure 43) layers. To verify these points, we have matched all the modules with their respective Test Cases just to ensure that all the modules are tested.

EDGE	TEST CASES
DATA FETCHER & TRANSFORMATION **(FOKUS)**	2,3,5
PERSONAL APP BACKBONE **(SUITE5)**	1,2,3,4,5
EDGE ANALYTICS ENGINE **(SUITE5)**	4
ANONYMISER **(ASSENTIAN)**	3,5,7
PRIVACY METRICS DASHBOARD **(UNISYSTEMS)**	3,7
SHARING CONFIGURATION **(SUITE5)**	3,5,7,8
PERSONAL WALLET **(ATOS)**	1
DATA REQUEST SERVICE RESOLVER **(SUITE5/MAGGIOLI)**	6,7

Figure 42 - Edge Modules used in the Test Cases

CLOUD	TEST CASES
PERSONA GENERATOR (ASSENTIAN)	6
CLOUD PLATFORM BACKBONE (UBITECH)	5,6, 7
TRUSTED DLT ENGINE & THE PUBLIC AND PRIVATE LEDGERS (UBITECH)	5,6, 7
QUERY BUILDER AND DATA EXPLORER (FOKUS)	6, 7
PLATFORM WALLET	6, 7
RISK MANAGEMENT MONITOR (UNISYSTEMS)	3,7
ABE/SSE ENGINE (ATOS)	5,6, 7
DATA STREAM DATA COMPOSER (UBITECH)	6, 7
ACCESS POLICY ENGINE (TECNALIA)	6, 7
SECURE ANALYTICS PLAYGROUND (ATOS)	8
DATA SEEKER STORAGE SPACE (UBITECH)	6,7, 8

Figure 43 - Cloud Modules used in the Test Cases

We have selected Postman[5] Collections as the tool for creating the integrations test. Postman collections allow us to automate request runs and generate API documentation and API test and keep our workspace organized to collaborate with other technical partners of the project.

To summarize, Figure 44 shows how will be the end work around this defined methodology.

1) The “Test Case” is defined with their respective “Functionalities”
2) Ensure with the technical partners if its module is executed in the functionality, so we determine the “affected modules” of each functionality.
3) Determine which modules expose an API and which do not.
4) Generate the Postman collections
5) Run our tests.

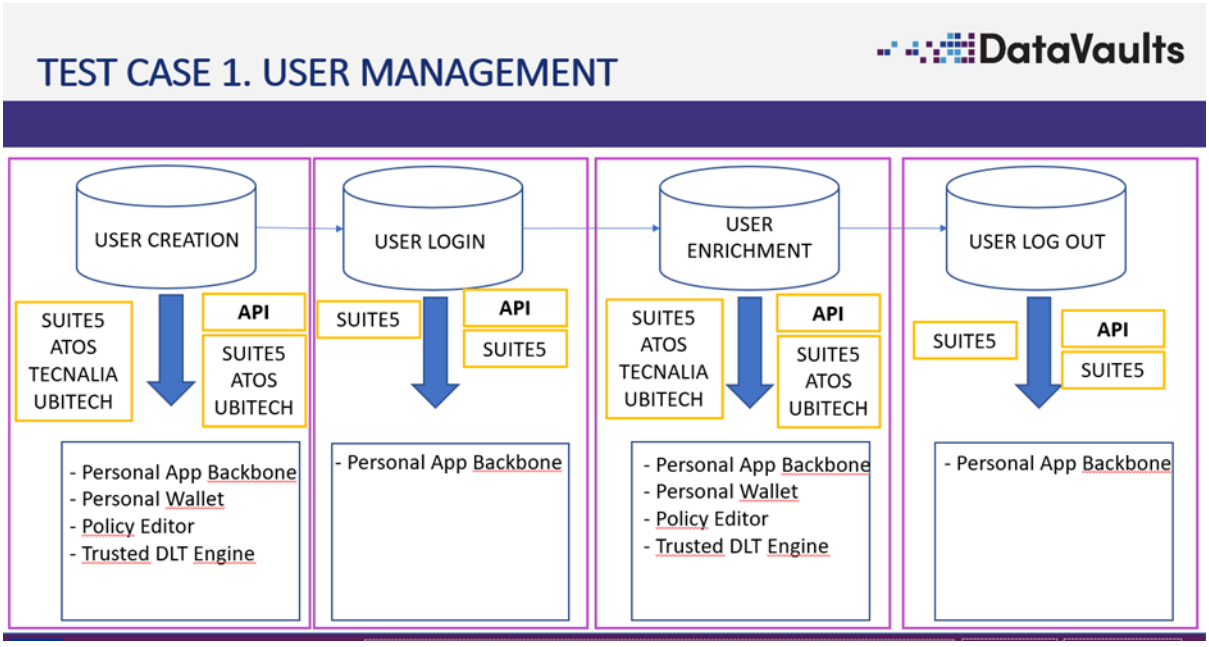


Figure 44 - Final Test Case

[5] https://www.postman.com/

The integration tests described under these lines, shouldn't be immutable in the sense that we can increase the "Test Cases" if we believe is necessary or doesn't cover some functionality. This also will affect the modules implied in these new Test Cases or Functionalities. As we already have mentioned, testing periodically and covering as many functionalities as possible of the platform are the idea for this task.

5 RELEASE PLANNING

5.1 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 final step of the plan which includes version v1.0 of the platform is depicted in Figure 45 below.

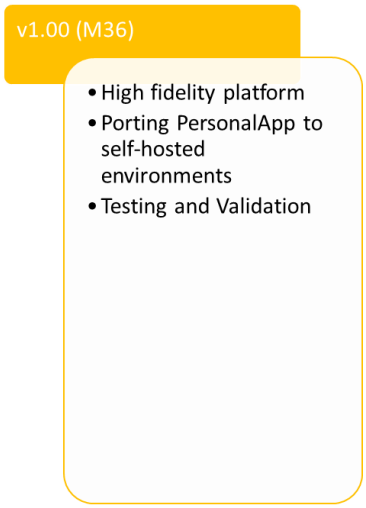


Figure 45: DataVaults Release Planning

In the upcoming V1.0 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 and testing and validation will continue to ensure that all issues observed will be solved in the best way possible.

6 CONCLUSIONS AND NEXT STEPS

In this document, we presented Version 0.5 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, installation instructions, VM, IP and port of their deployment.

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 intermediate release. Improvements since the beta version have been made across various components to better cater for the User Stories and the needs of the pilots that were created from the use of the beta platform.

Moreover, 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. Finally, details regarding the technical assurance of the platform were presented, focusing on the continuous integration and the testing, with the next goal being to release version 1.0 of the platform in M36 to provide a validated and fine-tuned platform that will be tested and ensure high fidelity till the end of the project.

7 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 D5.4, DataVaults Platform - Beta Version (full functional beta), 2021.

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

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

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