Tailored IoT & BigData Sandboxes and Testbeds for Smart, Autonomous and Personalized Services in the European Finance and Insurance Services Ecosystem

# D4.17 – Visualization Front-End for Aggregated Information - II

| | |
|---|---|
| **Revision Number** | 3.0 |
| **Task Reference** | T4.6 |
| **Lead Beneficiary** | ENG |
| **Responsible** | Susanna Bonura – Domenico Messina |
| **Partners** | Participating partners in Task according to DOA |
| **Deliverable Type** | Report (R) |
| **Dissemination Level** | Public (PU) |
| **Due Date** | 2022-03-31 |
| **Delivered Date** | 2022-03-28 |
| **Internal Reviewers** | RRD, ATOS |
| **Quality Assurance** | INNOV |
| **Acceptance** | WP Leader Accepted and/or Coordinator Accepted |
| **EC Project Officer** | Beatrice Plazzotta |
| **Programme** | HORIZON 2020 - ICT-11-2018 |
| | This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement no 856632 |

# Contributing Partners

| Partner Acronym | Role[1] | Author(s)[2] |
|---|---|---|
| **ENG** | Lead Beneficiary | Domenico Messina |
| **RRD** | Internal Reviewer | Stephanie Jansen - Kosterink |
| **ATOS** | Internal Reviewer | Ignacio Elicegui |
| **INNOV** | Quality Assurance | John Soldatos |

# Revision History

| Version | Date | Partner(s) | Description |
|---|---|---|---|
| 0.1 | 2022-02-03 | ENG | ToC Version |
| 0.2 | 2022-02-09 | ENG | Added section 1 |
| 0.3 | 2022-02-17 | ENG | Added section 2 |
| 0.4 | 2022-02-25 | ENG | Added section 3 |
| 0.5 | 2022-02-28 | ENG | Added section 4 |
| 0.6 | 2022-03-02 | ENG | Added section 5 |
| 0.7 | 2022-03-02 | ENG | Added section 6 |
| 0.8 | 2022-03-18 | ENG | Updated sections |
| 0.9 | 2022-03-23 | ENG | Version ready for Internal Review and Quality Assurance |
| 1.0 | 2022-03-24 | RRD, ATOS, INNOV | Version with internal reviews and quality check |
| 2.0 | 2022-03-25 | ENG | Feedback coming from internal review and quality check implemented |
| 3.0 | 2022-03-29 | GFT | Ready to be submitted |

[1] Lead Beneficiary, Contributor, Internal Reviewer, Quality Assurance

[2] Can be left void

# Executive Summary

This software deliverable reports the work conducted in T4.6 of the INFINITECH project. The main goal of this task is to deliver a visualization front-end software component to be integrated wherever financial data needs to be visualized and analysed, with special emphasis on data streaming. In accordance with the INFINITECH specifications the implementation of such a component is based on microservices' approach.

Towards this end, the scope of this report is to present the final version of this component. Compared to the previous version, that included just a set of the high-fidelity mock-ups, this report describes the features and functionalities of the developed finalized software.

Within Pilot#10, this user interface will be used to visualize suspicious fraudulent transactions coming from the ML-pipelines according to the dashboard layout requested by the user.

Veesualive is delivered as a Docker Image and deployed as a blueprint according to "the INFINITECH way" development process, and will also be available as an asset in the INFINITECH Marketplace.

# Table of Contents

# List of Figures

# Abbreviations/Acronyms

| Abbreviation | Definition |
|---|---|
| API | Application Programming Interface |
| DB | Database |
| GUI | Graphical User Interface |
| HTML | HyperText Markup Language |
| LDAP | Lightweight Directory Access Protocol |
| JDBC | Java Database Connectivity |
| ML | Machine Learning |
| OAuth | Open Authorization |
| Repo | Repository |
| SQL | Structured Query Language |
| UI | User Interface |
| UX | User Experience |

# 1 Introduction

This document is the second version of INFINITECH's Visualization Front-end for Aggregated Information and the two of them summarize the work done in T4.6 of the INFINITECH project. This concrete version reports the final result of the web-based framework, named Veesualive, for the visualization of aggregated results developed in the scope of the project, and more generally of all relevant information to the financial user. The tool has been widely used and integrated within Pilot#10 sandbox.

As declared in the first version of this deliverable, where the motivation for this solution is widely explained, the main goal of the work conducted in T4.6 is to deliver a visualization front-end software component to be integrated wherever financial data needs to be visualized and analysed with special emphasis on data streaming. In accordance with the INFINITECH specifications the implementation of such a component is based on microservices approach.

## 1.1 Objective of the Deliverable

T4.6 revolves around the development of a tool capable of visualizing and monitoring trends and mining financial big data. The proposed solution to achieve this objective is Veesualive, a front-end framework that lets the user visualize and explore real-time data and carry out the desired goals in several different ways. Available 24/7, this business intelligence platform can generate plenty of charts based on data coming from various sources, from databases, to files and APIs, making it the perfect solution for Fintech/Insurance Tech applications.

This deliverable aims to present the final version of Veesualive, the visualization component developed during T4.6 activities. This report includes a deep analysis of the tool, with a breakdown of its architecture and an overview of the main features and developed functionalities, all of this accompanied by a set of screenshots to better explain how the user interaction happens.

## 1.2 Insights from other Tasks and Deliverables

Insights from other tasks and deliverables are the same as the ones declared in the deliverable D4.16 (repo).

## 1.3 Structure

Deliverable D4.17 is organized in five main sections as indicated in the table of contents.

- Section 2 presents broadly the solution’s architecture design.
- Section 3 describes the container-based approach and its local installation using Docker Compose.
- Section 4 provides a guide for the Installation on Kubernetes.
- Section 5 provides a user guide for Veesualive, also delivered as online documentation within the Veesualive UI.
- Section 6 gives some key information on the application project.

# 2 Real time visualization architecture

Veesualive represents the INFINITECH Visualization Tool in charge of providing advanced data exploration and visualization functionalities, in order to exploit data coming from the acquisition from different data sources as well as the output of querying and filtering results that will come from other tools such as the Query Builder (tasks T4.2).

In order to design and implement Veesualive, the technical requirements were described in the deliverable D4.16. They were elicited within the Pilot#10 design and implementation, as well as on the INFINITECH specifications, this section explains the requirements currently addressed by the visualization front-end component and the implementation details currently envisioned for this component.

Starting from the achievements and requirements presented in D4.16, the architecture here presented was designed (Figure 1). It is worth noticing that references to the requirement identifiers (e.g. *[FE_xx]*) defined in D4.16 were added throughout this section.

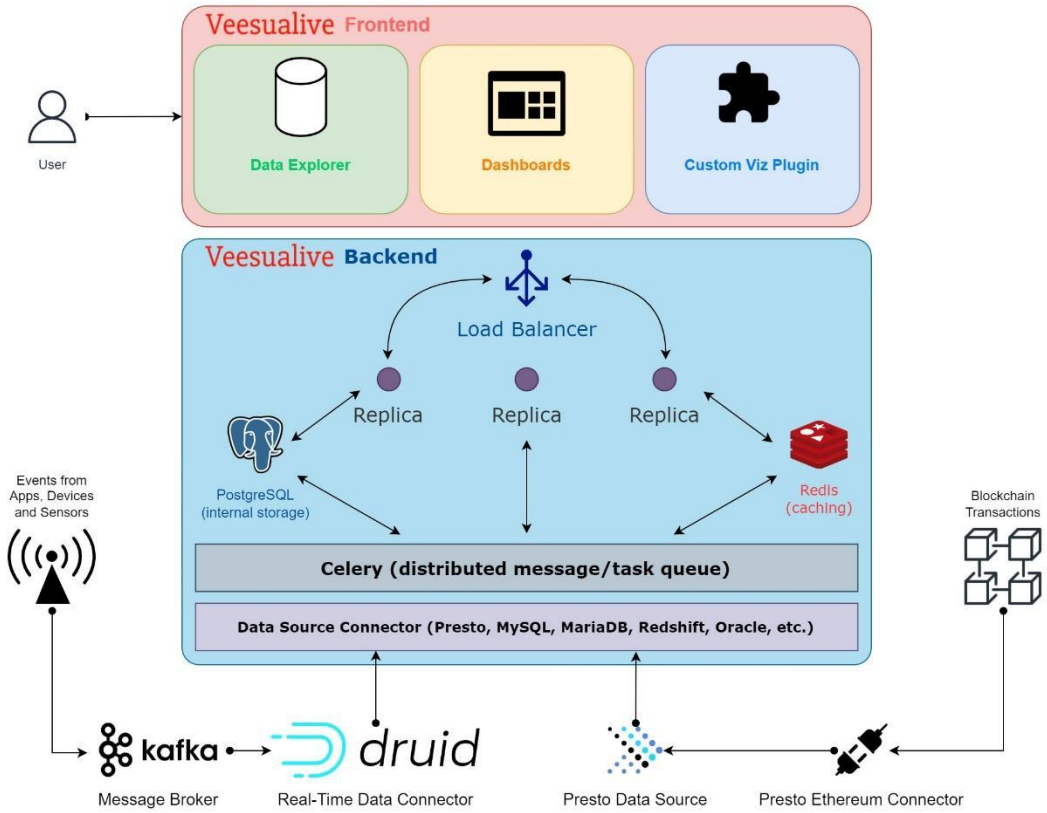


Figure 1 - Veesualive Architecture Diagram

Veesualive is based on Apache Superset[3], a modern, open-source data exploration & visualization platform designed to support all data personas in an organization.

Veesualive is distributed as a web application. Its main parts are:

[3] https://superset.apache.org/

- **Frontend** – interactive Graphical User Interface (GUI) [FE_13] that clients can connect to and explore from their web browser.
- **Backend** – the core of the software, responsible for managing the requests coming from the client and performing all the heavy processing.

As you can notice from the architecture breakdown depicted in Figure 1, the strong point of the platform is its modern, cloud-native, microservice-based, and scalable architecture [FE_05, FE_06, FE_12] that lets it run either in *sequential* or *distributed mode*:

- In **sequential mode**, there is only one main process on the server side that has to take care of the computing, so queries cannot last more than 60 seconds to be solved.
- In **distributed mode**, Veesualive can rely on a load balancer to handle the workload among multiple server *replicas*, copies of the same server-side program running on different nodes of a cluster, thus significantly improving performances. This mode allows the tool to scale the processing based on the current open tasks, if one of the workers is too busy with computing, the load balancer can jump in and distribute the next tasks to another worker.

This modular structure makes Veesualive easily adjustable to fit different scenarios according to the needs and processing efforts. The backend includes indeed the following components:

- A PostgreSQL[4] [FE_04] database, that represents the **internal persistence layer**.
- An **in-memory key-value storage** Redis[5], used for caching queries and aggregated results [FE_02].
- A **shared bus** based on Celery[6], a distributed task queue focused on real-time processing and task scheduling that supports the exchange and processing of vast amounts of messages [FE_03]. Veesualive relies on it to exchanged and distribute messages and tasks among the replicas.
- The **Data Source Connector**, a component that collects a lot of drivers and technologies to link the tool to different sources.

In particular, what makes Veesualive compatible with plenty of data sources is the SQLAlchemy[7] Dialect, that can connect to any SQL-speaking database. Moreover, it is possible to significantly extend the list of supported databases thanks to its DB-API-based technology, that allows to install additional drivers for external sources or data proxies, like Presto[8], Amazon Redshift[9] or Apache Hive[10].

Thanks to this extensibility it was possible to create a real-time data flow, from streaming sources to live dashboards: the Data Source Connector has been integrated with Apache Druid[11], a real-time database designed for data pipelines and workflows; druid supervisors connect to the message broker Apache Kafka[12] and spawn tasks that subscribe to the topics. In this way Veesualive, thanks to the druid driver, is able to continuously ingest data streams [FE_01]. Through the Data Source Connector, it was also possible to make

[4] https://www.postgresql.org/

[5] https://redis.io/

[6] https://github.com/celery/celery

[7] https://www.sqlalchemy.org/

[8] https://prestodb.io/

[9] https://pypi.org/project/sqlalchemy-redshift/

[10] https://hive.apache.org/

[11] https://druid.apache.org/

[12] https://kafka.apache.org/

Veesualive retrieve and visualize the transactions inside the blocks of a blockchain, exploiting a third-party tool called Presto Ethereum Connector[13] that enables running SQL queries from Presto to the APIs exposed by Ethereum.

On the client side, Veesualive offers a rich, modern and interactive UI that lets the user get the most out of data and obtain useful insights upon visualization. In order to access restricted areas of the platform each user must provide their credentials to authenticate; in fact, Veesualive can be integrated with major authentication providers, like OpenID, LDAP or OAuth [FE_16, FE_09]. Upon performing the login, the user is then able to configure connections to data sources, manage owned database and datasets, play with tons of options to personalize queries [FE_14], define custom dimensions and metrics [FE_11] and fine-tune custom chart configurations [FE_15]. Veesualive boasts as many as 60 different chart types [FE_08] between Correlation, Distribution, Hierarchy, Dispersion and other categories, giving the user hundreds of possible combinations and as much freedom as possible in creating unique custom dashboards [FE_07]. A deeper dive in Veesualive GUI is addressed in section 5.

[13] https://ethereum.org/

# 3 Local installation using Docker Compose

Veesualive can be installed locally in a few minutes thanks to Docker Compose[14] on a Linux or Mac OSX environment. In a Windows computer, it is suggested to set up a Virtual Machine on the device and install the tool on the virtual Linux environment.

## 3.1 Prerequisites

- Registered account to access INFINITECH Repositories
- Docker and Docker Compose installed

## 3.2 Installation

1. Clone Veesualive git repository

```
$ git clone https://gitlab.infinitech-h2020.eu/presentation/veesualive
```

2. Launch Veesualive stack through Docker Compose

```
$ cd veesualive
$ docker-compose -f docker-compose-non-dev.yml pull
$ docker-compose -f docker-compose-non-dev.yml up
```

With these commands, Docker Compose will take care of downloading the required Docker images and set up all the services the tool needs to run properly. After a few minutes everything will be up and running for the users to start exploring data on a local environment. To connect to Veesualive, it is enough to open the web browser and go to the following address:

```
http://localhost:8088
```

[14] https://docs.docker.com/compose/

# 4 Installation on Kubernetes

Veesualive can run in distributed mode on a Kubernetes cluster thanks to the Helm chart[15] provided by the official Superset helm repository.

## 4.1 Prerequisites

- Registered account to access INFINITECH Repositories
- Kubernetes cluster
- Helm installed

## 4.2 Running

1. Add the Superset helm repository

```
$ helm repo add superset https://apache.github.io/superset
"superset" has been added to your repositories
```

2. View Charts in repo

```
$ helm search repo superset
NAME                    CHART VERSION   APP VERSION     DESCRIPTION
superset/superset       0.1.1           1.0             Apache Superset is a
modern, enterprise-ready b...
```

3. Configure the desired Chart settings crafting a configuration file similar to the *values.yaml* file included in the repository, from where the user should override the default values with his own settings.

4. Install and run

```
$ helm upgrade --install --values my-values.yaml superset superset/superset
```

After that, some pods should start popping up, like so:

```
$ kubectl get pods
NAME                                    READY   STATUS      RESTARTS   AGE
superset-celerybeat-7cdcc9575f-k6xmc    1/1     Running     0          119s
superset-f5c9c667-dw9lp                 1/1     Running     0          4m7s
superset-f5c9c667-fk8bk                 1/1     Running     0          4m11s
superset-init-db-zlm9z                  0/1     Completed   0          111s
superset-postgresql-0                   1/1     Running     0          6d20h
superset-redis-master-0                 1/1     Running     0          6d20h
superset-worker-75b48bbcc-jmmjr         1/1     Running     0          4m8s
superset-worker-75b48bbcc-qrq49         1/1     Running     0          4m12s
```

Of course, the actual pods shown may vary depending on the user values override, but generally we should expect:

- A number of `superset-xxxx-yyyy` and `superset-worker-xxxx-yyyy` equal to the `replicaCount` value.
- One `superset-postgresql-0` (if enabled by configuration)
- One `superset-redis-master-0` (if enabled by configuration)

[15] https://helm.sh/docs/topics/charts/

- One `superset-celerybeat-xxxx-yyyy` pod if the flag `supersetCeleryBeat.enabled` is set to *true* in the values overrides.

For more advanced settings and deep configurations please refer to the official Superset Installation Guide[16].

[16] https://superset.apache.org/docs/installation/running-on-kubernetes#important-settings

# 5 Veesualive online documentation

As already mentioned, Veesualive is provided with an intuitive graphical interface to facilitate the user in taking advantage of the several functionalities offered by the tool. This section illustrates the main pages of the platform and summarizes their purpose and features.

The link to the Gitlab repository containing Veesualive Documentation project is the following:

https://gitlab.infinitech-h2020.eu/presentation/veesualive-docs

Such guide is integrated within the Veesualive UI in order to facilitate the user experience.

This document has been written based on an instance of Veesualive included in the INFINITECH environment.

## 5.1 Login

Before accessing the platform, the user lands to the login screen **(Figure 2).**

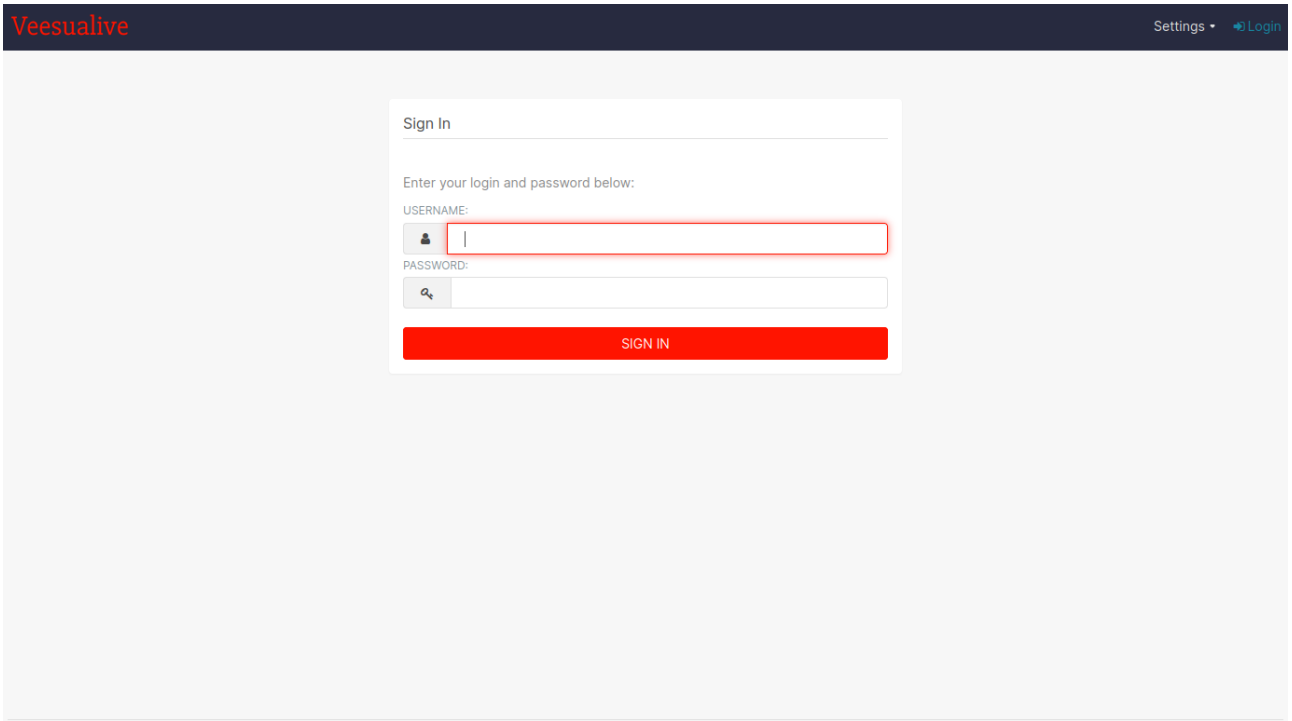


Figure 2 - Veesualive User Guide – Login

Here, the user is asked to provide username and password as credentials to authenticate. Bear in mind that for this purpose it is possible to configure Veesualive in order to be integrated with existing third-party authentication mechanisms. In case something is wrong with the information provided, the user is warned that login is invalid.

## 5.2 Home

Upon performing login with credentials, the user lands on a homepage containing an overview of the current situation in the platform, the recent actions, dashboards, charts and datasets the user has already added in Veesualive or were delivered as a sample out of the box during the tool initialization (**Figure 3**).

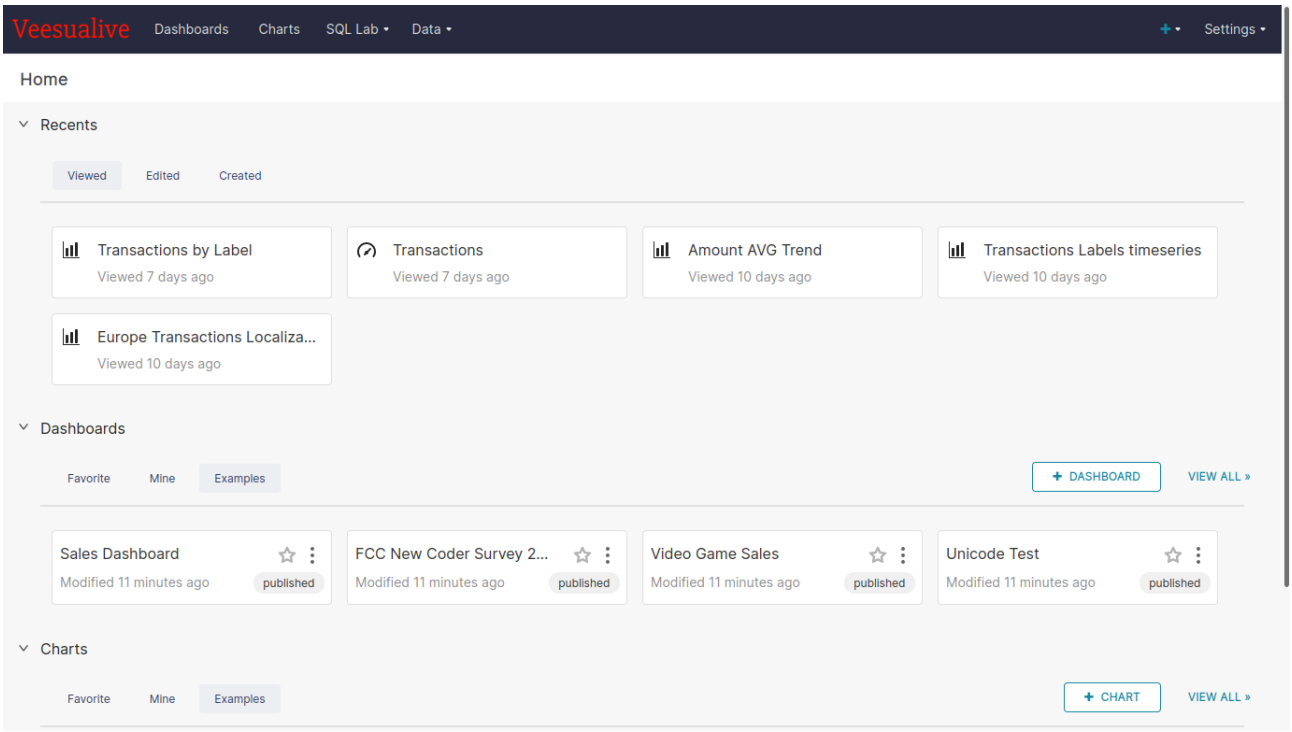


Figure 3 - Veesualive User Guide – Home

## 5.3 Databases

To perform queries and plot resulting data, the user has to add some datasets to his collection; to do so, the tool needs to connect to some actual data source first. This can be achieved in the Databases section, depicted in **Figure 4**.

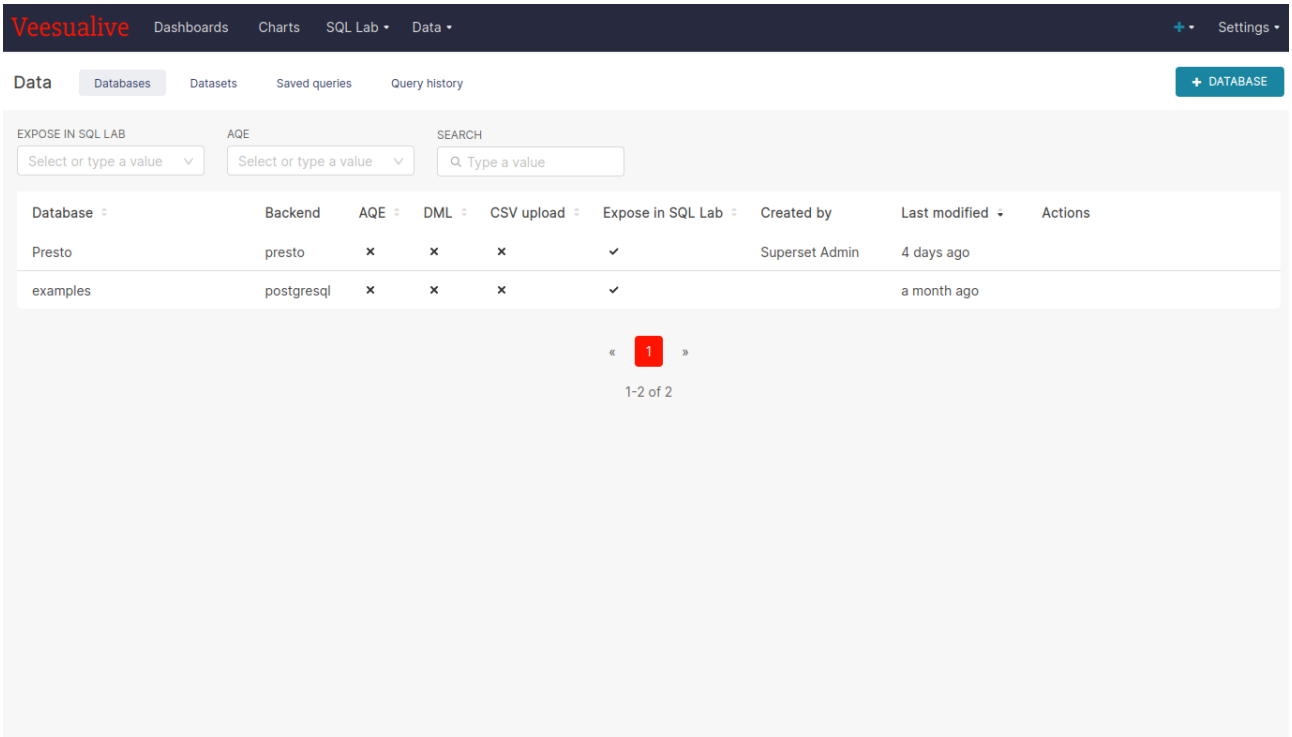


Figure 4 - Veesualive User Guide – Databases

Databases list is provided with a search input and combo boxes to filter the entries in the list of databases available. On the top right there is also a button to register new databases in the platform.

### 5.3.1 Database Registration

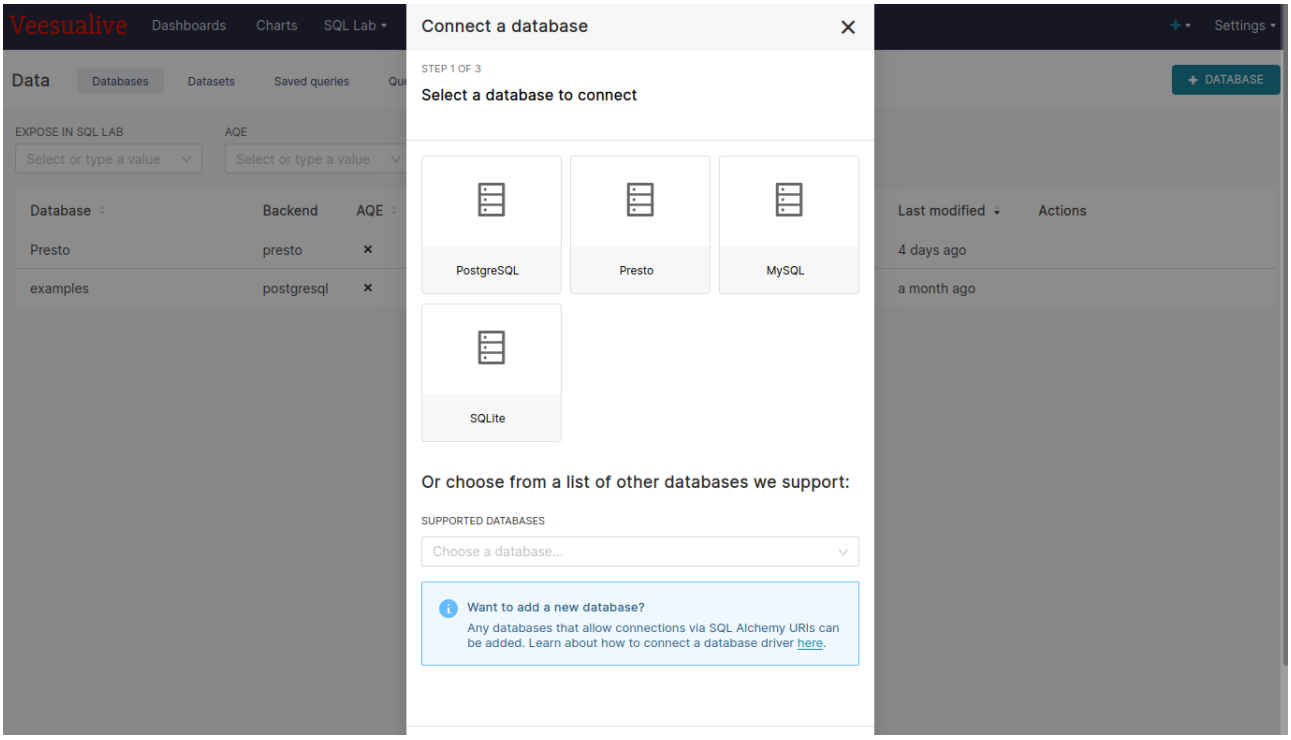


Figure 5 - Veesualive User Guide - New Database

Upon clicking on the “New Database” button, a modal is shown with a multi-step configuration where the user is asked to select the database to connect among the predefined ones or other supported data source types (**Figure 5**); he then has to fill in some information such as *host*, *port*, *credentials* or *SQLAlchemy URL* depending on the data source type selected (**Figure 6**).

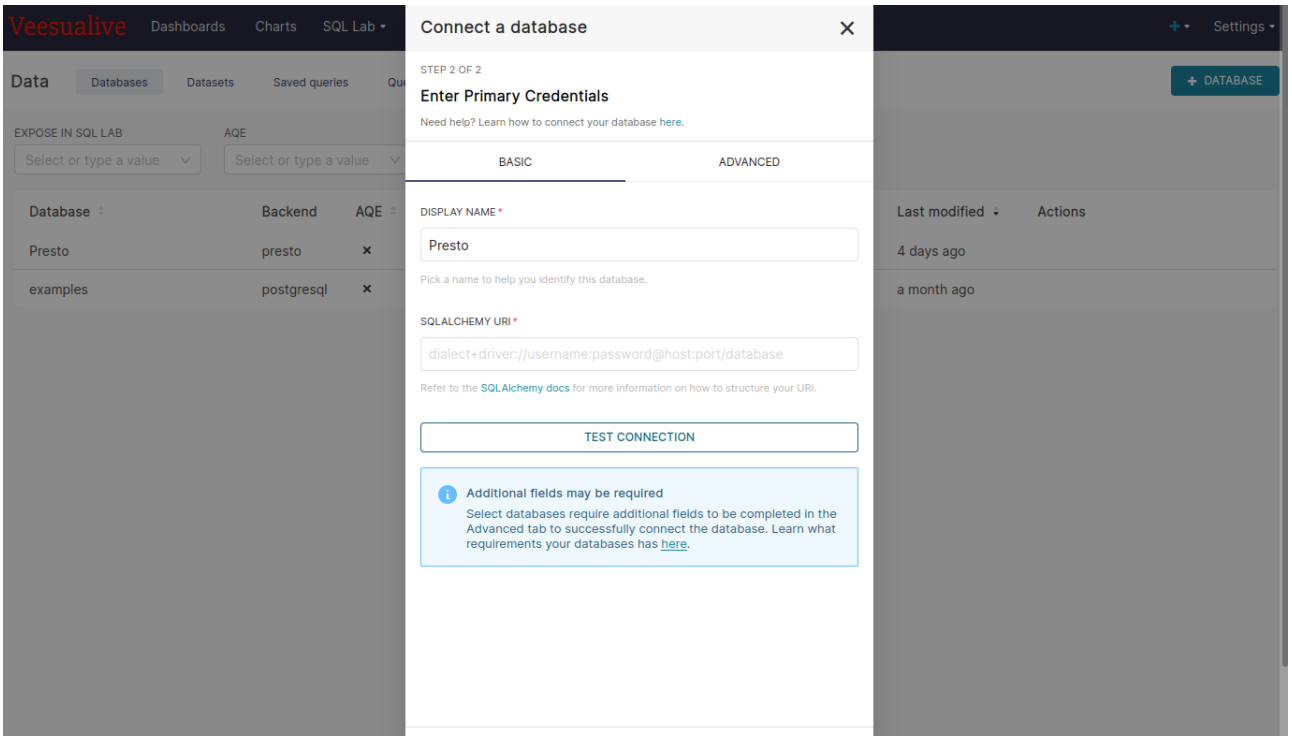


Figure 6 - Veesualive User Guide - New Database Settings

## 5.4 Datasets

In the Datasets section the user is allowed to manage its tables, register new ones or browse among the ones that were previously added.

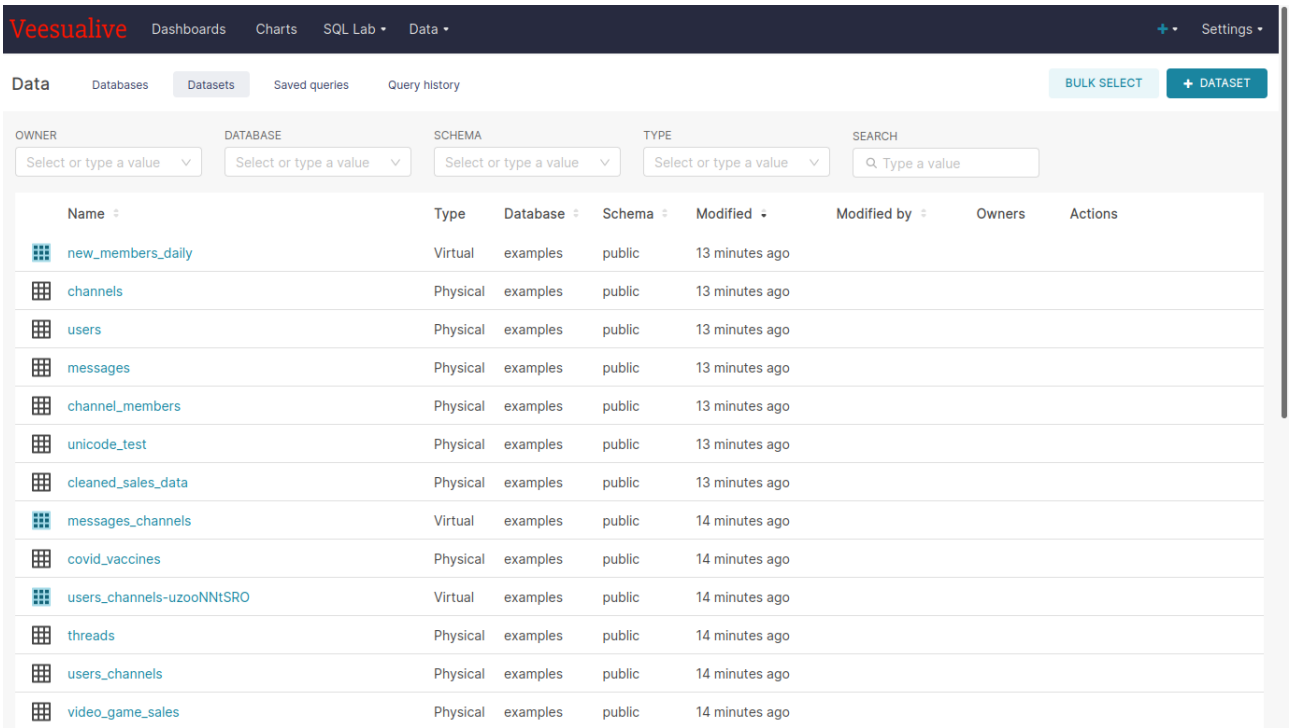


Figure 7 - Veesualive User Guide – Datasets

As **Figure 7** shows, the datasets list contains the list of all the tables available. On the upper part the list is provided with a search bar to input free text and some filters to select, for example, the datasets *owner*, the *database* they are stored in or their *type*. Each dataset has action buttons to edit, export or delete it.

On the top right the user is able to select one or more datasets to export or delete them all at once, or a "New Dataset" button to register a new dataset.

### 5.4.1 Dataset Registration

The "New Dataset" button triggers the opening of a modal asking the user to select one *database* among the ones available in the platform, the *schema* and *table* inside the database selected (**Figure 8**).

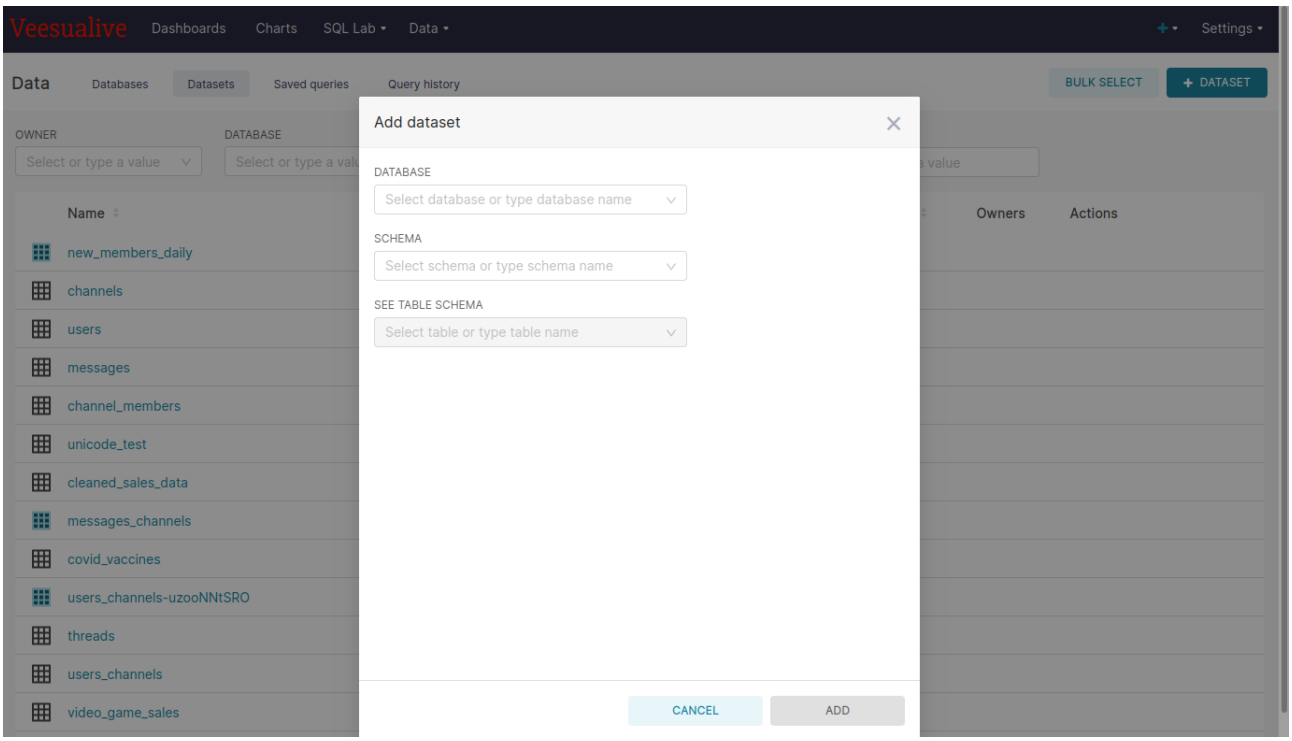


Figure 8 - Veesualive User Guide - New Dataset

## 5.5 Charts

Charts section is about managing charts configurations that the user wanted to save because of their impressive and effective looks and creating new stunning visualizations.

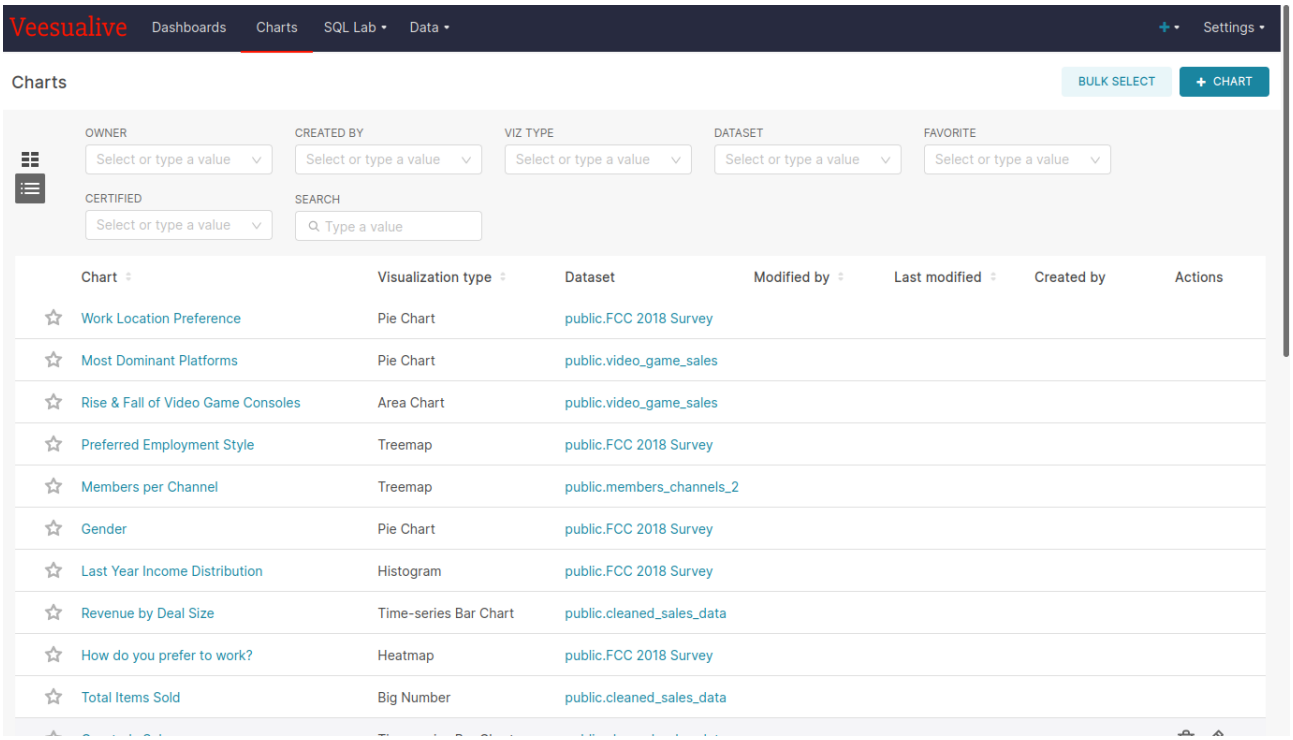


Figure 9 - Veesualive User Guide – Charts

**Figure 9** shows the list of charts stored in the platform, with actions for each of them to edit or delete it. In the upper part the user is able to perform some browsing searching for some keywords in the chart *name* or filtering by *owner*, *visualization type*, *dataset* and others. On the left it is possible to switch between *list mode* or *cards mode*, while on the top right you can enable *bulk selection* or configure a new visualization.

## 5.5.1 Chart Configuration

As soon as the user clicks on the “New Chart” button he is redirected to the Chart Configuration Wizard, depicted in **Figure 10**.

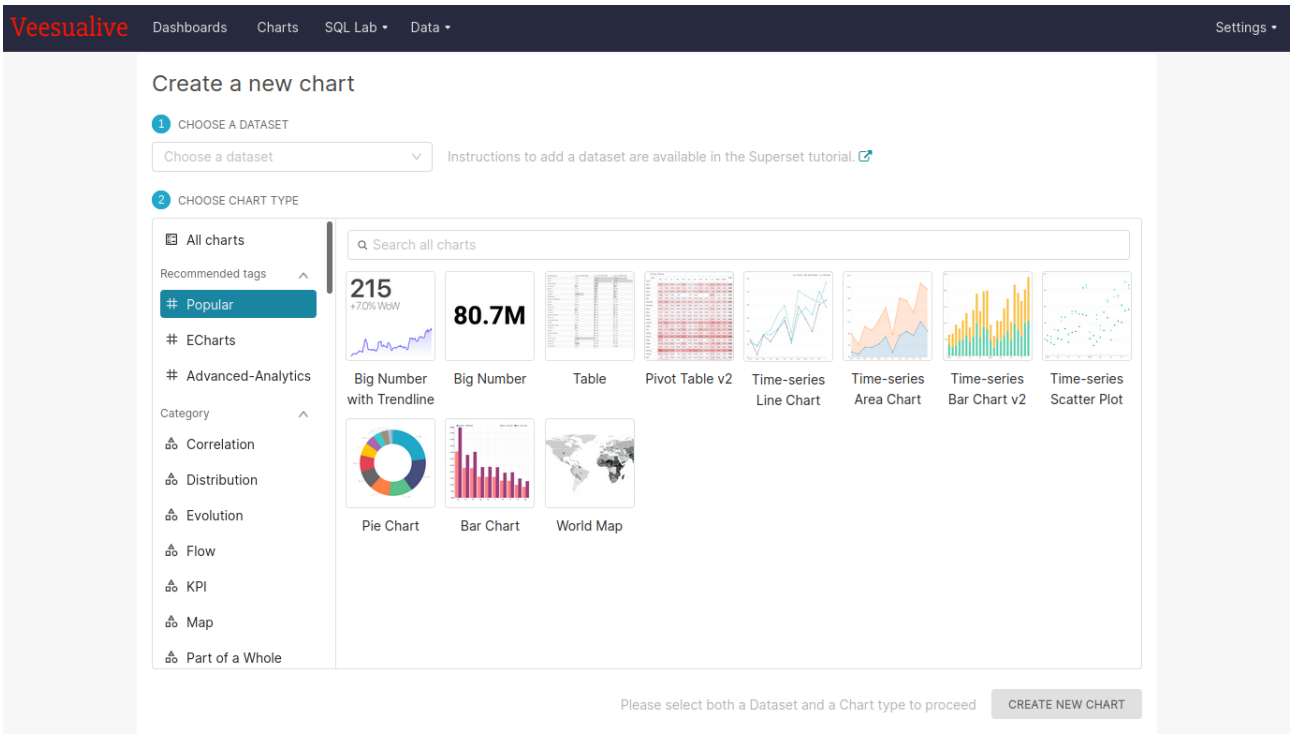


Figure 10 - Veesualive User Guide - Chart Configuration visualization type selection

Here, the user is asked to choose a dataset among the ones that were previously registered and then he is put in front of a collection of more than fifty chart types to choose among for the desired visualization.

Once he has chosen the most suitable visualization type for his purpose, the user is redirected to the selected chart customization, where he is able to personalize the *query* to perform to the chosen dataset and adjust the data that will be fed to the visualization, selecting the desired *columns* and setting additional *metrics* or *filters* or *limiting* the number of rows. Upon each change to the query, the user is required to trigger the request again through the "Run" button on top, or he is free to save the query for later use or for other charts as well (**Figure 11**). The right half of the page contains a preview of the chart that reflects the changes performed in the query and in the chart options, that include basic general settings like chart *title*, *colors*, *legend* and *labels*, but also some very advanced, chart-specific configurations.

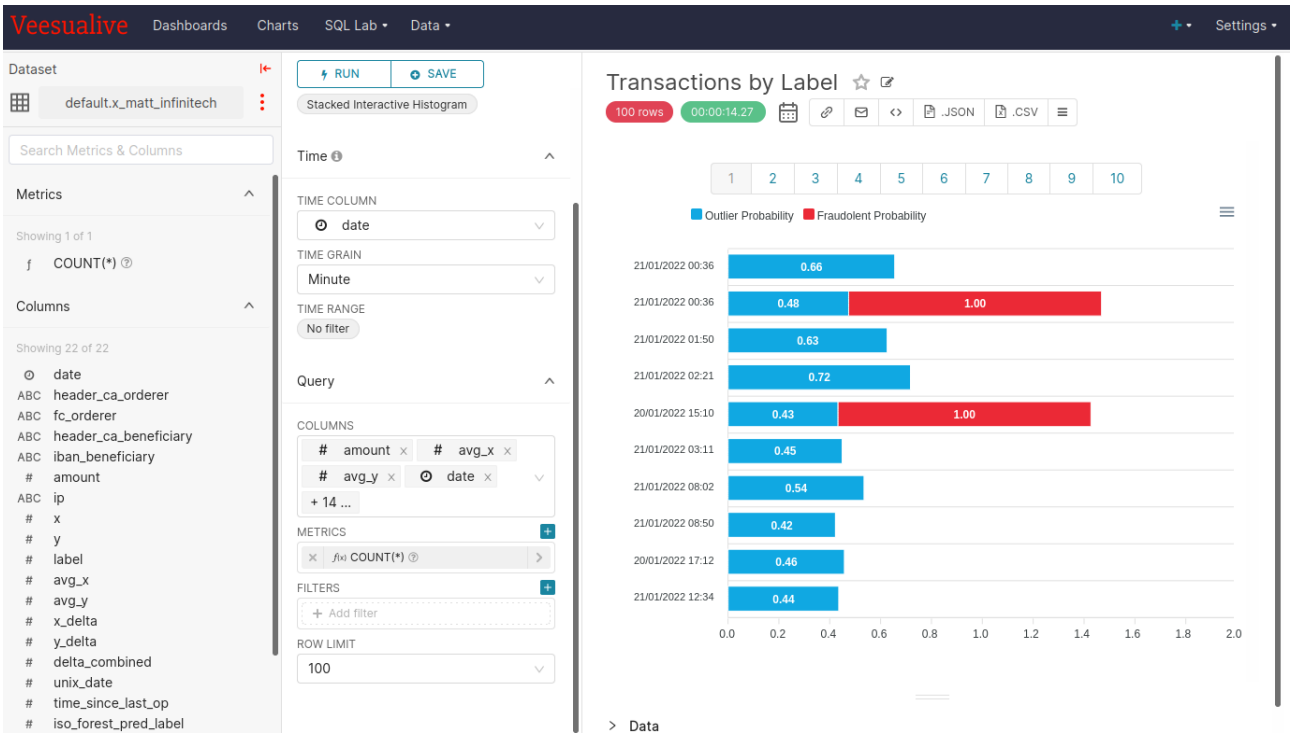


Figure 11 - Veesualive User Guide - Chart Configuration

## 5.6 Dashboards

Dashboards Section revolves around the management of the custom views the user composes as he wishes to have a clear idea of the overall situation about a particular topic or collection of data at a glance. **Figure 12** shows the list of dashboards available in the platform, with a few options to perform a basic browsing among the list and two buttons on top to enable bulk select mode or add a new dashboard.

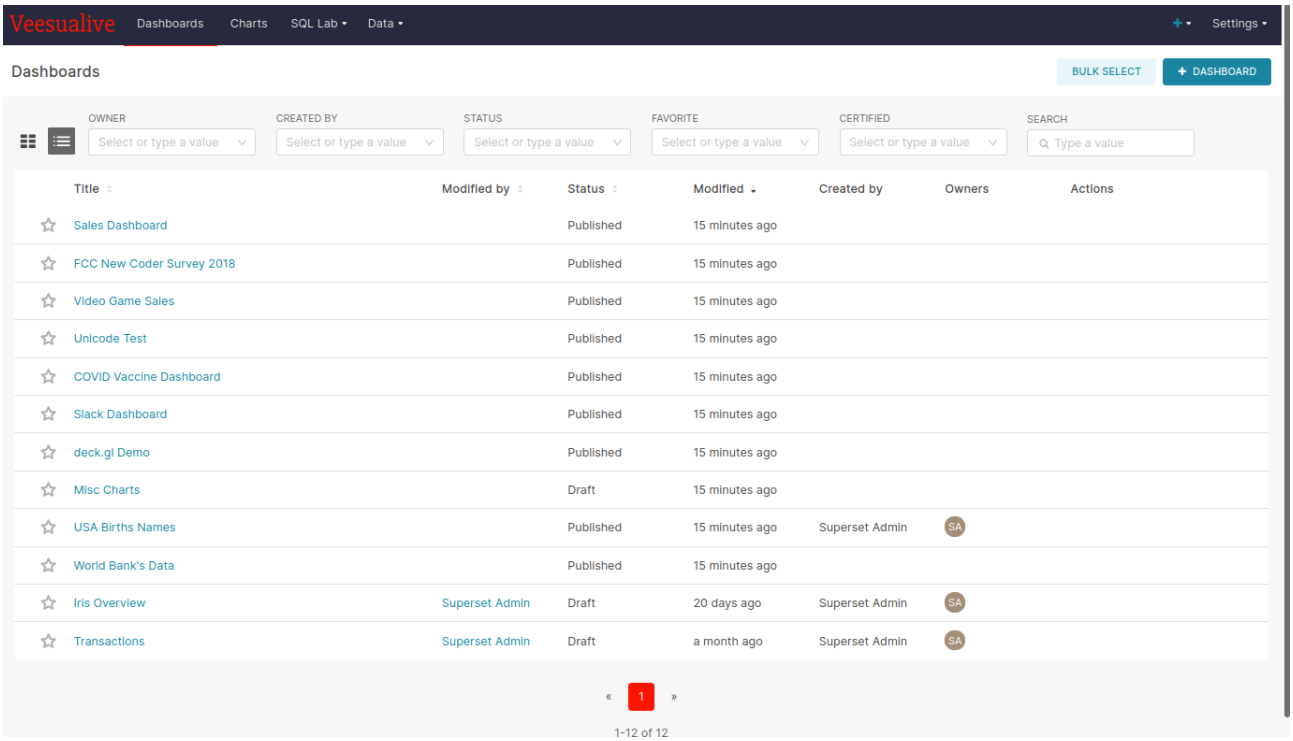


Figure 12 - Veesualive User Guide – Dashboards

## 5.6.1 Dashboard Configuration

Upon clicking on “New Dashboard”, the user is put in front of a canvas in which he is able to build up a dashboard by dragging the single elements from the palette on the right and dropping them wherever he prefers in order to achieve the desired layout and look.

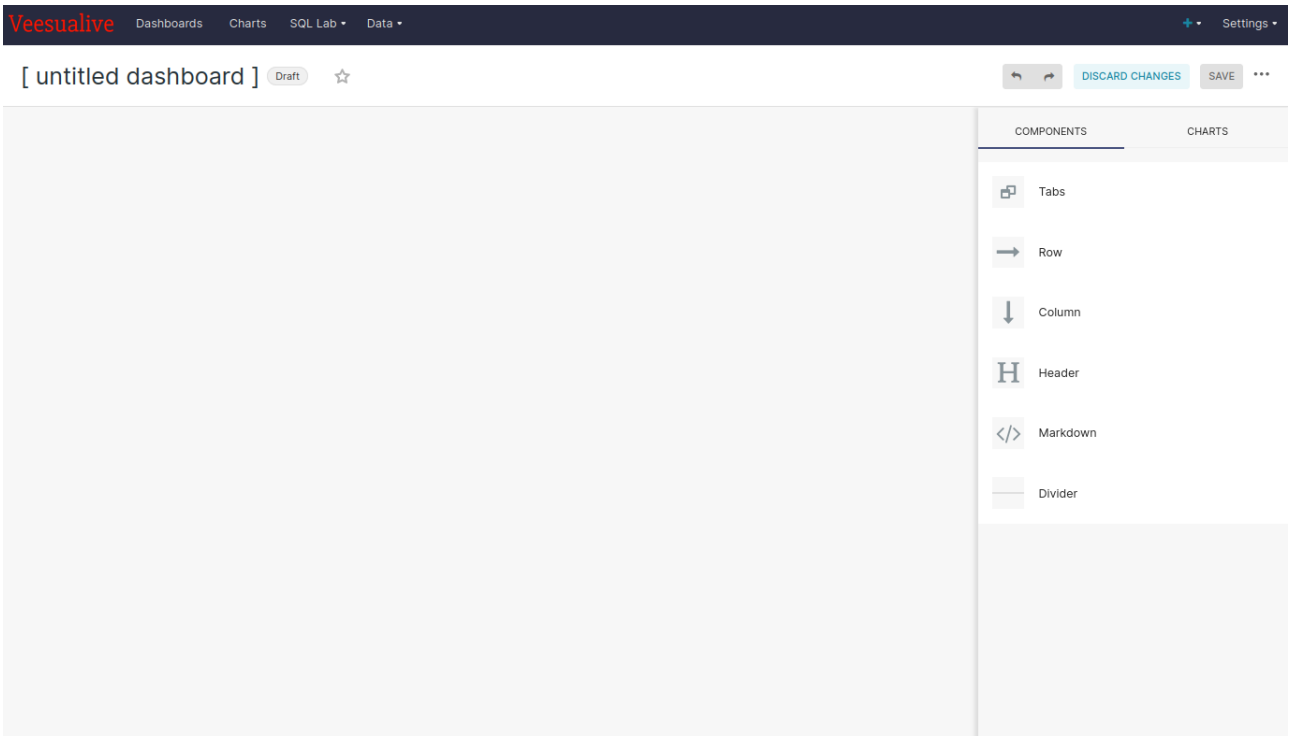


Figure 13 - Veesualive User Guide - Dashboard Configuration

As can be seen from **Figure 13**, the palette includes static components that focus on the layout, such as *rows*, *columns* and *dividers*, or basic HTML Elements like *tables*, *tabs* and *headers*, as well as the list of all the charts available in the platform, leaving the user almost unlimited freedom and hundreds of possible combinations for his custom views’ composition.

## 5.6.2 Dashboard Visualization

If you click on one of the dashboards in the list you land to the dashboard detail, where you can fully enjoy their visualization and interact with the different components.

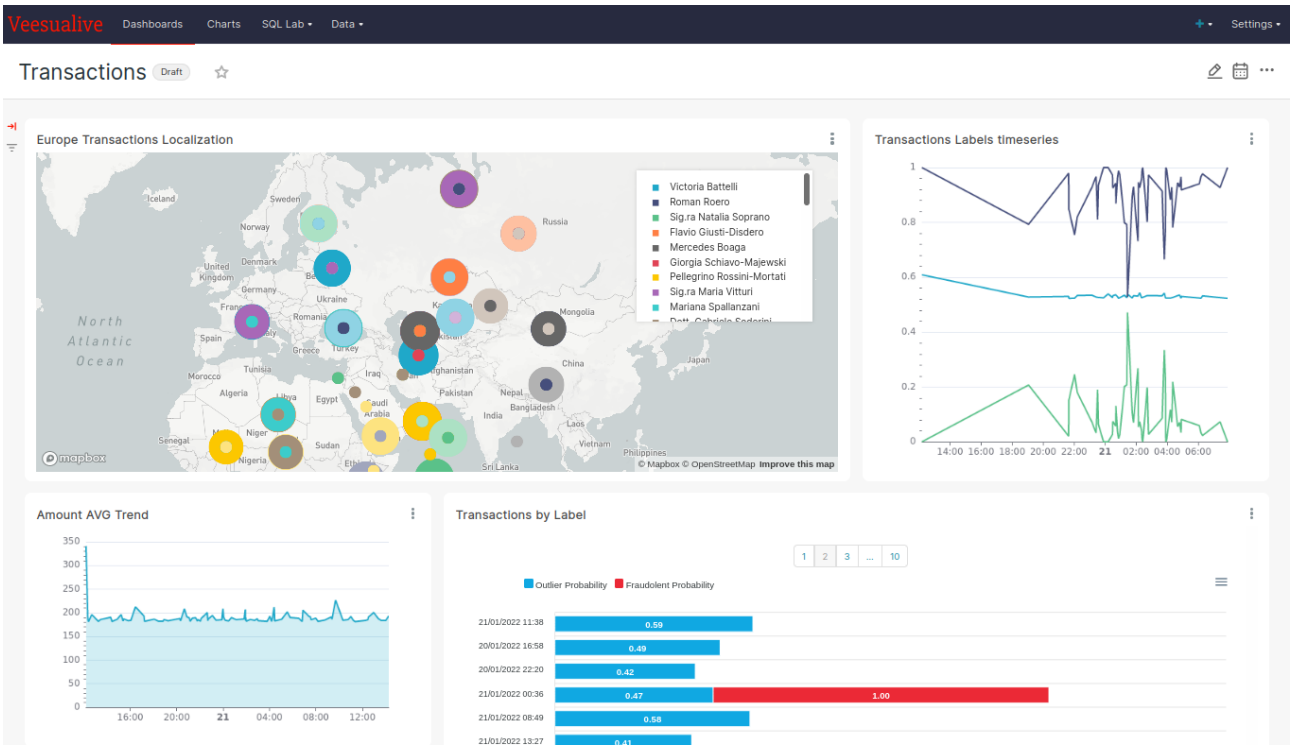


Figure 14 - Veesualive User Guide - Dashboard Visualization

As you can see from **Figure 14**, in visualization mode the user can admire the different charts and elements that compose the dashboard and play with filters and settings, if any, to see the visualizations change accordingly.

# 6 Conclusions

The purpose of this deliverable has been to deliver the final release of Veesualive, the output coming from the task T4.6 - Situation Awareness Front-End over Aggregated Information.

The deliverable is built on top of the main outcomes and the knowledge extracted from the WP2-3-4 and Pilot#10 feedback (within WP7) in order to provide the first version of Veesualive, the visualization front-end component that will be tested and validated within the Pilot#10 sandbox, where streaming warnings and alerts need to be shown to the fraud analyst after the ML-based-system predictions.

Veesualive is licensed under the **Apache License 2.0**.

Official Gitlab repository for source code: https://gitlab.infinitech-h2020.eu/presentation/veesualive.

Project README file: https://gitlab.infinitech-h2020.eu/presentation/veesualive/blob/master/README.md.

Veesualive will be available in the INFINITECH Marketplace.