

AGILE Development in Cloud Computing Environments

WS 2023-24



Reporting Component (APACHE Superset)

Masters in Engineering

Information Technology

Group Members

1.	Jenny Arulraj Nadar	1427226
2.	Karthik Kothamangala Sreenath	1438341
3.	Madhushree Manjunatha Lakshmidevi	1445185
4.	Padmini Manjunatha	1427336
5.	Sriram Karunanithi	1438891

Introduction

In response to the evolving needs of managing and optimizing service operations, this project aims to develop a robust system that seamlessly integrates user authentication, real-time service request reporting, and sophisticated data analytics. With a particular focus on executive board requirements, the system will leverage Apache Superset as its data visualization software, ensuring interactive and visually engaging dashboards for efficient decision-making. The system's foundation rests on the principles of data security, scalability, and flexibility, providing users with an intuitive interface to navigate and extract valuable insights. Through collaboration with the customer, the system can be customized to meet specific business action sets the stage for a comprehensive and user-centric approach, blending technological capabilities with practical insights for effective decision support.

Agile Methodologies & Principles

Agile Principles:

Agile methodologies are a collection of incremental and iterative software development techniques that place an emphasis on customer satisfaction, adaptability, and teamwork. The shortcomings of conventional, linear development models gave rise to these approaches.

Its core values place a higher priority on client participation than contractual agreements, flexibility in response to change than strict planning, and the delivery of useful software over copious documentation. Cross-functional teams are encouraged to collaborate through agile by working in brief, time-limited iterations called sprints. Teams are empowered to swiftly handle changing requirements and deliver incremental value thanks to these iterative cycles that enable continuous feedback. Agile fosters a dynamic and customer-focused development culture by prioritizing openness, efficient communication, and an early emphasis on producing a minimum viable product. Acknowledged as a fundamental component of modern software development and project management, the Agile approach fosters an attitude of flexibility, adaptability, and a dedication to ongoing enhancement

Agile Methodologies:

- **Scrum:** A popular Agile framework that divides work into time-boxed iterations called sprints, with a focus on specific roles (Scrum Master, Product Owner, and Team).
- **Kanban:** A visual management method that emphasizes continuous delivery and encourages teams to pull work from a prioritized backlog as capacity allows.
- **Extreme Programming (XP):** A set of practices that emphasizes customer satisfaction, frequent releases, and collaboration, with a focus on coding, testing, and listening to customer feedback.
- **Lean Software Development:** Adapted from Lean manufacturing principles, it emphasizes the elimination of waste, delivering as fast as possible, and continuous improvement.

- **Dynamic Systems Development Method (DSDM):** A framework that provides a set of principles, processes, and roles to deliver projects on time and within budget while meeting customer requirements.
- **Feature – Driven Development (FDD):** An iterative and incremental software development methodology that organizes software features into a hierarchical model, emphasizing feature completion.

Scrum and Kanban

Scrum:

Scrum is a fundamental component of the Agile approach. It is a popular and adaptable framework for managing complicated projects that exhibits flexibility, teamwork, and transparency. As such, it is a vital tool for incremental and iterative project delivery in a variety of sectors.

- **Sprint Planning:**
Utilize the Scrum framework's sprint planning meetings to define the scope of work for each iteration or sprint, ensuring that the development team has a clear understanding of the tasks at hand.
- **Backlog Refinement:**
Regularly engage in backlog refinement sessions to review and prioritize the user stories and features. This collaborative effort involves the product owner and the development team to ensure alignment with evolving project goals.
- **Daily stand-ups:**
Hold stand-up meetings every day to encourage team members to communicate and work together. This facilitates prompt settlement of issues, provision of progress reports, and detection of any obstacles to the team's workflow.
- **Sprint review and retrospective:**
Organize a sprint review at the conclusion of each sprint to present the finished product to stakeholders. In addition, hold a retrospective to evaluate the sprint, pinpoint areas that need work, and apply adjustments for increased productivity in upcoming sprint

Scrumroles:

To guarantee defined roles and productive teamwork, assign and follow Scrum roles, such as Development Team, Product Owner, and Scrum Master.



Fig1: Overview of Sprint planning and daily scrum

Kanban:

A Kanban board is a visual project management tool that helps teams manage and visualize their work in progress. It originated from the Toyota Production System and has been adapted for knowledge work and software development. The board typically consists of a physical or digital space divided into columns representing different stages of a workflow.

- **Visualize the Workflow:**

Implement a Kanban board to visually represent the workflow, including stages such as "To Do," "In Progress," and "Done." This provides transparency into the status of each task and facilitates efficient task management.

- **Work-in-progress (WIP) limits:**

For each step of the workflow, establish and enforce work-in-progress (WIP) boundaries to avoid bottlenecks and maximize workflow. This keeps things moving along at a regular pace and prevents the team from taking on too much work at once.

- **Continuous Delivery:**

Adopt the continuous delivery Kanban approach, which permits features to be released as soon as they are finished and put through testing. This is consistent with the Agile philosophy of regularly releasing functional software.

- **Kanban Metrics:**

Utilize Kanban metrics to gauge and enhance the development process's efficiency, such as lead and cycle times. These metrics include information about the average delivery speed as well as how long it takes a job to go through the process.

- **Adaptability:**

Take advantage of Kanban's adaptability by modifying the workflow as necessary, accommodating urgent activities, and adjusting to changes in priority. This adaptability fits very nicely with the Agile values of accepting change and producing value on a constant basis.

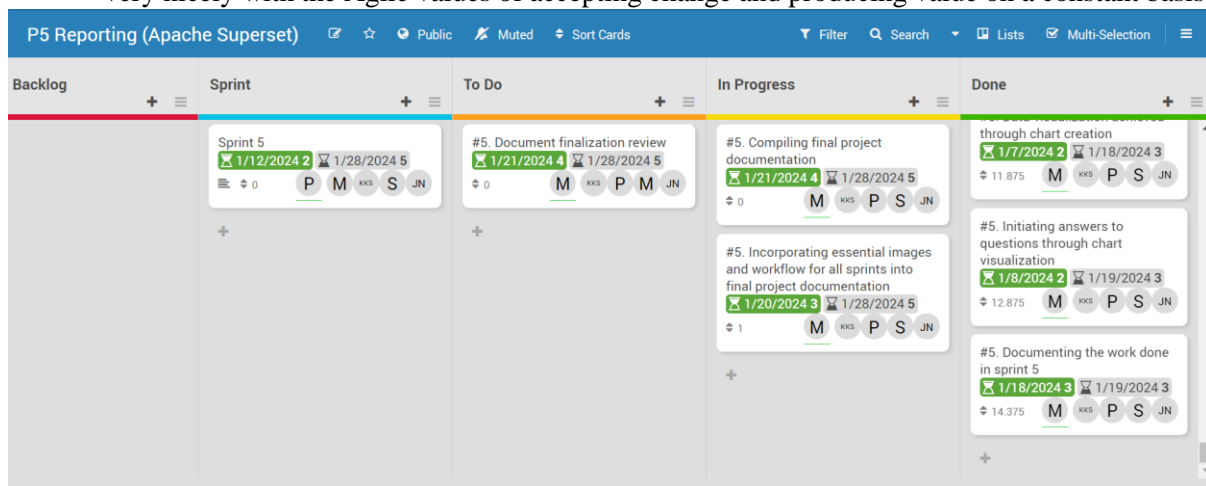


Fig2: Overview of Kanban board for Project Management



Apache Superset

A cutting-edge, enterprise-ready business intelligence web application is Apache Superset. Users of all skill levels can easily examine and visualize their data with this fast, lightweight, intuitive tool that offers a plethora of possibilities for everything from basic pie charts to intricate decks, GL geographical maps. Superset is renowned for having an intuitive user interface that makes it possible to create dynamic dashboards with a wide range of visualizations, including graphs, charts, and geographic maps. Ad hoc querying, which enables real-time data exploration without the constraints of pre-made reports, is its strongest point. Superset supports a broad range of data sources, including popular databases and cloud services, to provide access to a variety of data environments.

Key Features:

- **Connectivity:**

Superset is compatible with a large number of data sources, such as Apache Druid, well-known databases (including MySQL, PostgreSQL, and Oracle), and data warehouses (like Amazon Redshift and Google BigQuery). It can establish connections to many data environments with ease.

- **Interactive Dashboard:**

Interactive dashboards featuring tables, charts, and other visualizations are created by users. Because of the dashboards' great degree of customization, users can modify their design and layout to suit their own requirements.

- **Data Exploration:**

An easy-to-use interface is offered by Superset for data exploration and drilling down. Data can be filtered, grouped, and aggregated by users to reveal trends and provide new insights.

- **SQL Labs:**

Users can create and run SQL queries from within Superset with the help of the SQL Lab tool. This makes it possible to analyze and explore data in more sophisticated ways.

- **Chart Library:**

A vast collection of graphs and visualizations, such as pie charts, bar charts, and line charts, are available on the site. From these visualizations, users can select the one that best represents their data.

- **Security and Access Control:**
Strong security features in Superset give administrators control over permissions and access management. Secure data access is ensured by its support for integration with several authentication providers.
- **Collaboration and Sharing:**
By sharing dashboards and visualizations with other team members, users may work together. Superset makes it simple to share content via URLs or integrate dashboards into other programs.
- **Dynamic Datasets:**
Superset facilitates the creation of customizable and reusable charts and dashboards by supporting dynamic datasets. This encourages reporting that is efficient and consistent.
- **Extensibility:**
Because of Superset's expandable architecture, users can incorporate unique visualizations and expand its functionality. This adaptability meets specific business objectives as well as changing data requirements.
- **Scheduled Reports:**
To get automated updates on particular parameters, users can schedule reports. With the help of this function, stakeholders are kept informed without having to actively use the site.
- **Community and Support**
Superset, being an open-source project, enjoys the advantages of a thriving contributor community. In addition to sharing best practices and receiving support from the community, users may also contribute to the platform's continuous development.

Sprint Planning

A Scrum framework activity called sprint planning allows the team to select the items from the product backlog to work on during the sprint and to outline their preliminary plan for completing those items.

Within the Scrum framework, sprint planning is an essential activity that is carefully planned to get the Scrum Team ready for a forthcoming sprint. Every sprint begins with this cooperative meeting, which is attended by the Development Team, Scrum Master, and Product Owner. The team decides which backlog items to address in the upcoming sprint and outlines the specific tasks needed to produce a potentially shippable product increment during the Sprint Planning session. The meeting has a time restriction and is focused on three main goals: creating a comprehensive plan for task execution, ranking backlog items according to team capacity and significance, and gaining a clear grasp of the sprint goal. In addition to providing a roadmap for the entire sprint, sprint planning encourages team members to be cooperative, transparent, and dedicated.

Sprint – 1 (Length - 2 weeks):

We came together as a team to synchronize our objectives, establish standards, and create schedules. Collaboratively, we crafted a strategy outlining the main goals and tasks for multiple sprints. Recognizing the Agile principle of flexibility, we outlined initial project requirements while acknowledging the possibility of adjustments and backlog prioritization for subsequent sprints. Personally, we delved into the intricacies of setting up Apache Superset's development environment and dependencies to ensure seamless progress. To ensure alignment, we assigned a Product Owner to bridge development efforts with overarching goals. Additionally, a dedicated Kanban Master was tasked with optimizing workflow efficiency. Leveraging the Kanban framework, we prioritized efficient reporting and pursued continuous improvement. Our collective focus remained on exceeding expectations and fostering adaptability as we worked together closely.

Sprint – 2 (Length - 2 weeks):

We meticulously curated a dummy dataset in CSV format, carefully mimicking the anticipated real data, to ready our platform for dynamic data integration. Seamlessly integrating this dummy data into Apache Superset, we conducted thorough testing and analysis before proceeding with real data integration. Analyzing the dummy dataset with precision, we added dimensions, metrics, and filters to enhance data understanding and visualization. To ensure inclusivity and ownership, we held collaborative sessions where team members reviewed and contributed to the practice dashboard, valuing diverse viewpoints. We also took proactive measures to deepen our understanding of data characteristics, facilitating strategic integration and analysis. By immersing ourselves in Apache Superset, generating dummy data for testing, and researching adapters, we collaboratively ensured seamless integration with other team members' APIs and diverse data formats.

Sprint – 3 (Length - 2 weeks):

As a team, we collaborated on developing a Python program in Visual Studio Code to seamlessly convert a specified database format into CSV files, a critical step in our data preparation pipeline. Our focus centered on leveraging Apache Superset's capabilities to display and analyze the CSV data, aiming for insightful information and meaningful visuals. Currently, we're deeply involved in

developing an Application Programming Interface (API) to automate the linkage between Apache Superset's database and the produced CSV data, with the aim of improving efficiency and accessibility. This API initiative is geared towards streamlining the data visualization and analysis process, ensuring a smooth workflow, and maximizing Apache Superset's capabilities. Through a comprehensive approach, we prioritize adherence to best practices and readiness for further enhancements in automation and efficiency within the data visualization and analysis process. Additionally, we're exploring Apache Superset with SQL queries, manually connecting it to MySQL, SQLite, SQLAlchemy, and Postgres Servers, and establishing connections between Apache Superset's database and VS Code via MySQL, SQLite, SQLAlchemy, and Postgres extensions.

Sprint – 4 (Length - 2 weeks):

Despite facing persistent difficulties in establishing a seamless connection between our API and the MySQL server, our team has taken proactive measures to tackle and overcome these challenges. We're diving into the intricate details of our API's integration with MySQL, dedicating ourselves to troubleshooting and overcoming any obstacles hindering the connection's success. Simultaneously, we're facing difficulties in establishing a connection with the Superset database, requiring continued efforts to identify and resolve underlying issues. To ensure we maintain a strong and reliable connection aligned with our project goals, we're continuously diagnosing and correcting problems.

In parallel, we're exploring alternative options to sustain the momentum of data visualization. This includes manually converting JSON data into CSV format using web tools and uploading it to Apache Superset, allowing us to continue visualizing data in accordance with project specifications. This multifaceted approach underscores our team's agility and determination in overcoming technical hurdles. As we actively work to resolve connectivity issues with Superset and MySQL databases, we're also exploring alternative methods to visualize and analyze data, ensuring the project progresses smoothly. Regardless of the changing technical landscape, our dedication to delivering insightful work remains steadfast, reflecting our strategic and adaptable approach.

Sprint - 5 (Length - 2 weeks):

As we embark on our project journey, we're laying down the foundation by meticulously outlining the initial project concept through a carefully crafted flowchart. With a keen eye for detail, we're diving deep into the data gathered from various teams, aiming to unlock valuable insights and perspectives. Leveraging the power of visualization, we're translating this data into meaningful charts, enabling us to uncover patterns, trends, and correlations. These charts not only facilitate in-depth analysis but also spark discussions and address essential project queries. Every step of the way, our team is fully immersed, ensuring that our visualizations are not only informative but also actionable, driving the project forward with purpose and clarity.

Reporting Component(Apache Superset)

Project Description:

Within the intricate framework of this endeavor, we embark on the meticulous construction of a dynamic reporting component, with the robust Apache Superset as the architectural cornerstone. This platform, revered for its technical finesse in rendering complex datasets into actionable insights, is poised to be the bedrock of our reporting infrastructure. Our strategy unfolds as a finely tuned symphony of data insights, with contributions from diverse project groups harmoniously orchestrated. The pivotal node in this intricate data orchestration is a centralized PostgreSQL server, meticulously engineered to provide a secure, unified storage space, facilitating the seamless integration of datasets sourced from diverse origins. Moving beyond the realm of traditional data representation, our overarching objective is to engineer a visually compelling reporting system. This system transcends the singular pursuit of addressing the professor's specific inquiries; it morphs into a collaborative haven, nurturing active participation and cultivating a culture of data-driven decision-making among project participants. In essence, our project strives to be more than a reporting tool; it aims to be a catalyst for transformative insights and collaborative exploration in the realm of data analytics, leveraging the prowess of advanced technologies.

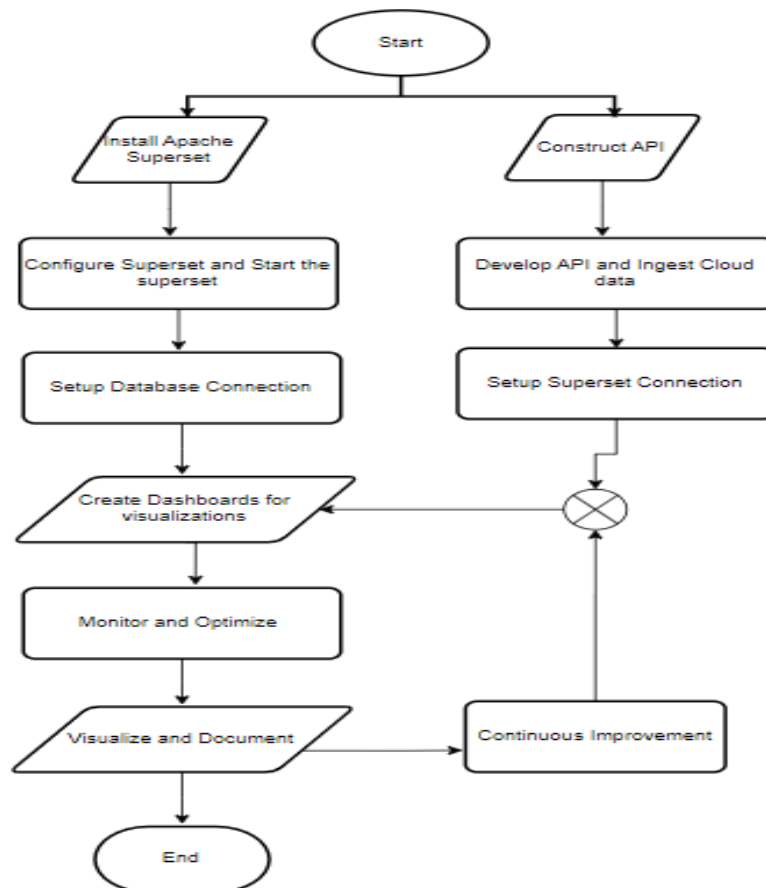


Fig3: Flowchart

Installation Method

In the ever-evolving landscape of data analytics, the ability to transform raw data into meaningful insights is paramount. Recognizing the crucial role of visualization in deciphering complex datasets, our project embarks on a journey with Apache Superset, a cutting-edge open-source data exploration and visualization platform. This section delves into the detailed installation process of Apache Superset, a cornerstone in our agile development approach. From prerequisites to troubleshooting, we guide you through the steps to unleash the power of Superset, offering a robust foundation for visually compelling and actionable data presentations. As we traverse the installation pathway, envision the seamless integration of Superset into your data analytics arsenal, paving the way for a transformative and agile approach to deriving insights from your data landscape. Let's embark on this installation journey, propelling your data visualization capabilities to new heights.

1. Installation of Apache Superset on Windows:

The fastest way to try Apache Superset locally is by using Docker and Docker Compose on a Linux or Mac OS computer. Superset does not have official support for Windows, so we have provided a VM workaround below.

The installation of Apache Superset on Windows is streamlined through the use of Docker. Follow these steps to effortlessly set up Apache Superset for your data analytics endeavors.

1) Install Docker Desktop for Windows:

Begin by downloading and installing Docker Desktop for Windows. This can be done by navigating to [Docker's official website](https://docs.docker.com/docker-for-windows/install/) and following the installation instructions.

2) Apache Superset Installation using Docker Compose:

Utilizing Docker Compose simplifies the deployment process. Execute the following steps to initiate the Apache Superset installation:

- **Clone the Superset Repository:**

Clone the Apache superset incubator repository. To do this use the following command :

```
$git clone https://github.com/apache/incubator-superset.git
```

- **Navigate to the Superset Directory:**

Move into the incubator-superset directory

```
$cd incubator-superset
```

- **Launch Docker Compose:**

Initiate the Docker Compose setup:

```
$docker-compose up
```

This process takes around 10 to 20 minutes to complete the whole setup. After the installation is completed then all the images will be available on the docker desktop.

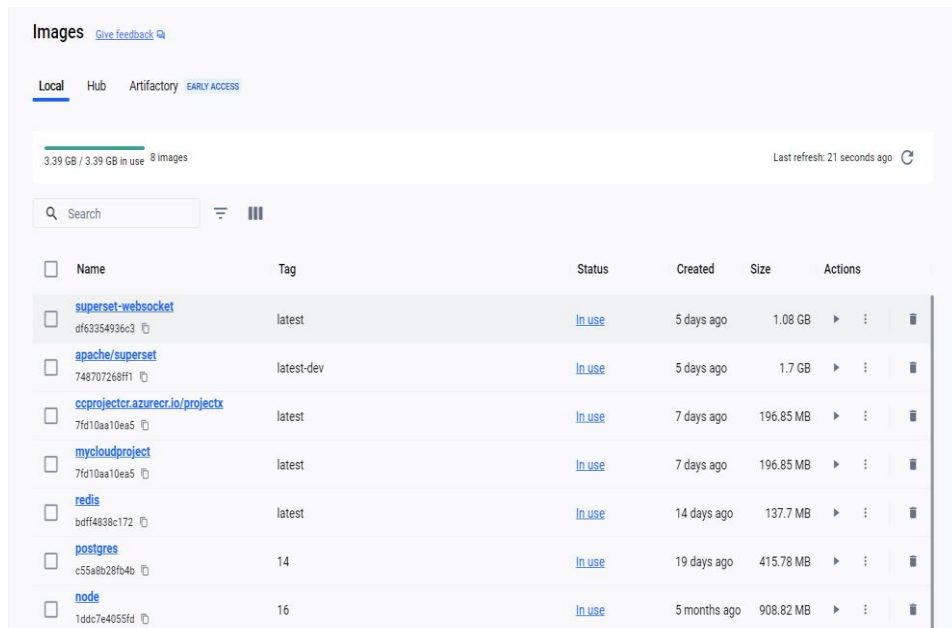


Fig 4: Docker images installed after installing Apache Superset.

3) Access Apache Superset

To access Apache Superset, open your preferred web browser and go to <http://localhost:8088>

Use the credentials as follows to login into the Apache Superset.

Username: admin

Password: admin

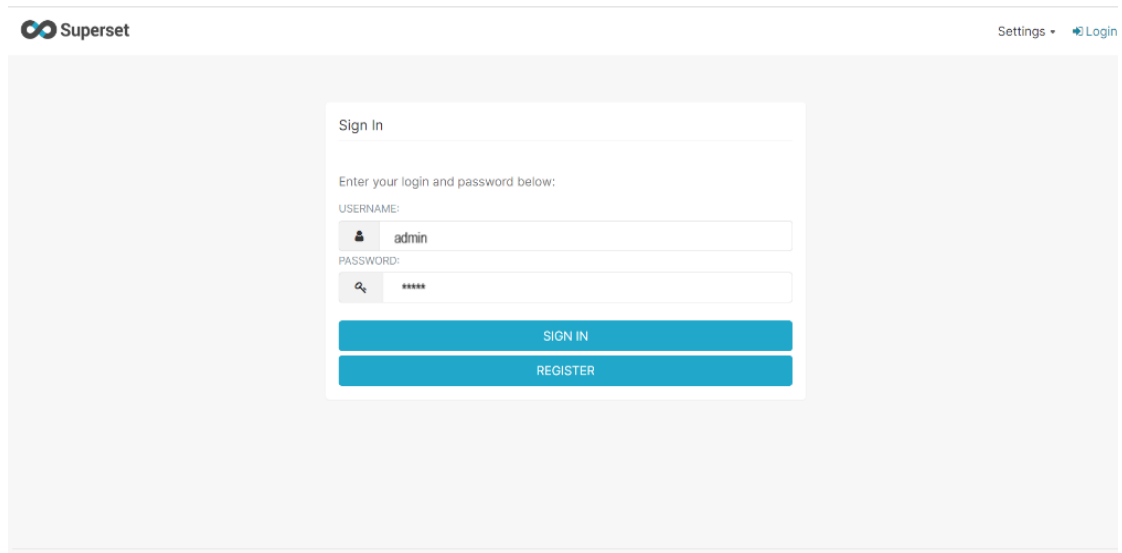


Fig 5: Landing page of Apache Superset

2. Installation of Apache Superset on Ubuntu:

Apache Superset's installation on Ubuntu is a straightforward process that involves configuring the environment, setting up dependencies, and launching the Superset server. Each command and statement

serves a specific purpose, contributing to the seamless integration of Apache Superset. Let's delve into the technical intricacies of the installation process:

Prerequisite:

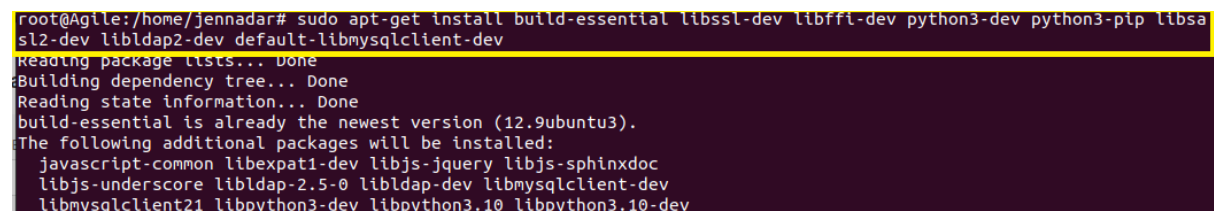
Before venturing into the installation, ensure that Python 3.7 or a later version installed on your system. Confirm the Python version using:

```
python --version
```

Step1: Installing Essential Packages

To fortify the foundation for Apache Superset, install the necessary packages with the following commands:

```
sudo apt-get install build-essential libssl-dev libffi-dev python3-dev python3-pip libsass2-dev libldap2-dev default-libmysqlclient-dev
```



```
root@Agile:/home/jennadar# sudo apt-get install build-essential libssl-dev libffi-dev python3-dev python3-pip libsass2-dev libldap2-dev default-libmysqlclient-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
build-essential is already the newest version (12.9ubuntu3).
The following additional packages will be installed:
  javascript-common libexpat1-dev libjs-jquery libjs-sphinxdoc
  libjs-underscore libldap2.5-0 libldap-dev libmysqlclient-dev
  libmysqlclient21 libpython3-dev libpython3.10 libpython3.10-dev
```

Fig 6: Installation of dependencies on CMD.

```
sudo apt-get install python3-setuptools
```

```
pip3 install --upgrade pip
```

```
pip install --upgrade setuptools pip
```

These commands ensure that your system is equipped with essential tools and dependencies.

Step2: Create Python Virtual Environment

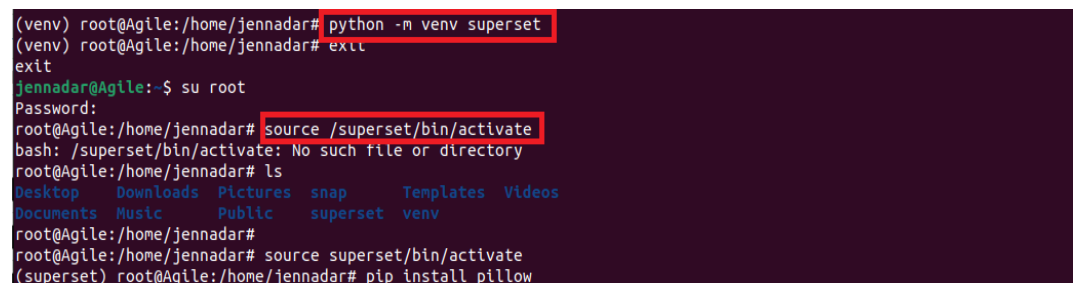
Isolate the Apache Superset installation by creating a Python virtual environment

To create a python environment:

```
python -m venv superset
```

Activate the virtual environment:

```
. superset/bin/activate
```



```
(venv) root@Agile:/home/jennadar# python -m venv superset
(venv) root@Agile:/home/jennadar# exit
exit
jennadar@Agile:~$ su root
Password:
root@Agile:/home/jennadar# source /superset/bin/activate
bash: /superset/bin/activate: No such file or directory
root@Agile:/home/jennadar# ls
Desktop  Downloads  Pictures  snap      Templates  Videos
Documents  Music      Public    superset  venv
root@Agile:/home/jennadar#
root@Agile:/home/jennadar# source superset/bin/activate
(superset) root@Agile:/home/jennadar# pip install pillow
```

Fig 7: Creating a virtual environment for Apache Superset.

Step3: Install Apache Superset

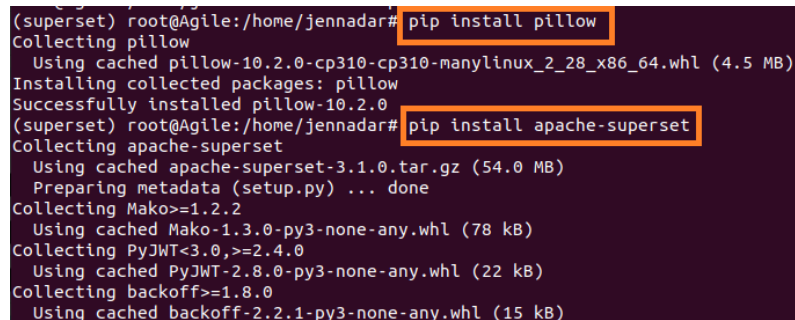
Navigate through the installation Process by executing the following command:

Installation of pillow:

```
pip install pillow
```

Installation of Apache Superset:

```
pip install apache-superset
```



```
(superset) root@Agile:/home/jennadar# pip install pillow
Collecting pillow
  Using cached pillow-10.2.0-cp310-cp310-manylinux_2_28_x86_64.whl (4.5 MB)
Installing collected packages: pillow
Successfully installed pillow-10.2.0
(superset) root@Agile:/home/jennadar# pip install apache-superset
Collecting apache-superset
  Using cached apache-superset-3.1.0.tar.gz (54.0 MB)
  Preparing metadata (setup.py) ... done
Collecting Mako>=1.2.2
  Using cached Mako-1.3.0-py3-none-any.whl (78 kB)
Collecting PyJWT<3.0,>=2.4.0
  Using cached PyJWT-2.8.0-py3-none-any.whl (22 kB)
Collecting backoff>=1.8.0
  Using cached backoff-2.2.1-py3-none-any.whl (15 kB)
```

Fig 8: Installing Apache Superset on Ubuntu.

Now as we all know Apache superset is a FLASK Application. So we need to link it also. So for that use the following command:

```
export FLASK_APP=superset
```

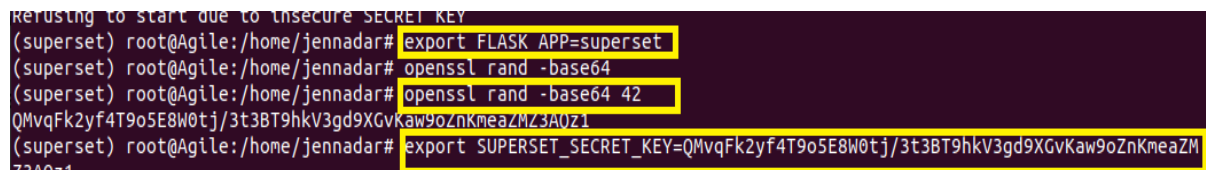
Step4: Create a Secret Key

Generate a secret key to enhance the security of Apache Superset. To generate the key you will need openssl. Command to generate the key is as follows

```
openssl rand -base64 42
```

Now u need to store the generated key in Apache Superset using the command.

```
export SUPERSET_SECRET_KEY=<Generated-Secret-Key>
```



```
Refusing to start due to insecure SECRET KEY
(superset) root@Agile:/home/jennadar# export FLASK_APP=superset
(superset) root@Agile:/home/jennadar# openssl rand -base64
(superset) root@Agile:/home/jennadar# openssl rand -base64 42
QMvqFk2yf4T9o5E8W0tj/3t3BT9hkV3gd9XGvKaw9oZnKmeaZM73A0Z1
(superset) root@Agile:/home/jennadar# export SUPERSET_SECRET_KEY=QMvqFk2yf4T9o5E8W0tj/3t3BT9hkV3gd9XGvKaw9oZnKmeaZM73A0Z1
```

Fig 9: Creating Secret Key for Apache Superset.

This key is crucial for securing sensitive information within Apache Superset.

Step5: Initialize Database

Initiate the database setup process:

```
superset db upgrade
```

This command ensures that the database schema is updated to support Apache Superset.

Step6: Create Apache Superset User:

Establish an administrative user for Apache Superset along with loading example data and initializing the application:

```
superset fab create-admin
```

This command will start to create a user with username admin and it will prompt you to enter a

password.

Now you need to load all the examples on the Apache Superset under the username admin. For that you need to enter

superset load-examples

```
(superset) root@Agile:/home/jennadar# superset fab create-admin
logging was configured successfully
2024-01-19 15:20:54,984:INFO:superset.utils.logging_configurator:logging was configured successfully
2024-01-19 15:20:54,989:INFO:root:Configured event logger of type <class 'superset.utils.log.DBEventLogger'>
/home/jennadar/superset/lib/python3.10/site-packages/flask_limiter/extension.py:336: UserWarning: Using the in-memory storage for tracking rate limits as no storage was explicitly specified. This is not recommended for production use. See: https://flask-limiter.readthedocs.io#configuring-a-storage-backend for documentation about configuring the storage backend.
  warnings.warn(
Username [admin]: admin
User first name [admin]: Jenny
User last name [user]: Nadar
Email [admin@fab.org]: jennynadar9@gmail.com
Password:
Repeat for confirmation:
Recognized Database Authentications.
Admin User admin created.
(superset) root@Agile:/home/jennadar# superset load_examples
logging was configured successfully
2024-01-19 15:21:45,380:INFO:superset.utils.logging_configurator:logging was configured successfully
2024-01-19 15:21:45,386:INFO:root:Configured event logger of type <class 'superset.utils.log.DBEventLogger'>
/home/jennadar/superset/lib/python3.10/site-packages/flask_limiter/extension.py:336: UserWarning: Using the in-memory storage for tracking rate limits as no storage was explicitly specified. This is not recommended for production use. See: https://flask-limiter.readthedocs.io#configuring-a-storage-backend for documentation about configuring the storage backend.
```

Fig10: Create an user of Apache Superset and load the examples.

Step7: Open Apache Superset on a Web Browser

Now the final step is to run Apache Superset. For that you need to type the following commands.

superset init

After that you need to run Apache superset on the localhost(127.0.0.1) and port 8088.

superset run -p 8088 --with-threads --reload --debugger

Once this is done then you need to open a browser and enter <http://127.0.0.1:8088>

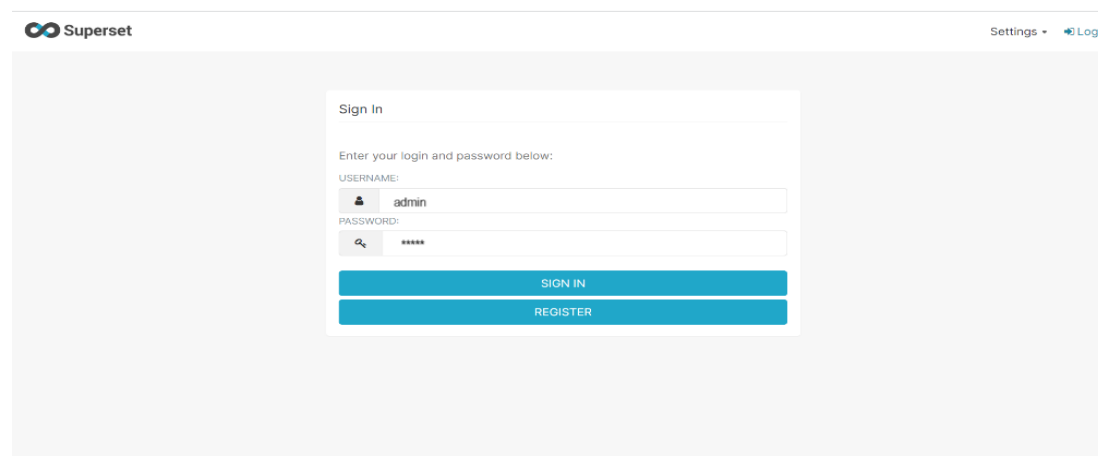


Fig 11: Landing page of Apache Superset

Methodology

Our implementation approach involves a meticulous blend of technical prowess and collaboration. We established a robust PostgreSQL server to serve as the central repository, employing Python scripts and data integration tools for seamless retrieval and integration of datasets from diverse sources. Apache Superset is configured to connect with the PostgreSQL server, offering an intuitive interface for dynamic data visualization. The design of the reporting component is customized to address the professor's specific questions, ensuring that each visualization panel provides actionable insights. User training and documentation are prioritized to facilitate easy navigation. By prioritizing data integrity, collaborative exchange, and user-friendly visualization, our approach seeks to elevate the project's data exploration experience to new heights.

The whole process to dynamically integrate Apache Superset with Postgresql Server is explained below:

Installation of Postgres:

PostgreSQL is an advanced open-source relational database management system known for its standards compliance, extensibility and support for complex data types. With a focus on ACID properties, scalability, and a vibrant open source community. PostgreSQL is widely used for storing and managing structured data in various applications. Begin the process by installing PostgreSQL, a relational database management system, crucial for our data storage needs. To install postgresQL on ubuntu follow the instruction below.

Step1: Installing all the dependencies:

Make sure your package list is up to date.

```
sudo apt update
```

Install the PostgreSQL package:

```
sudo apt install postgresql postgresql-contrib
```

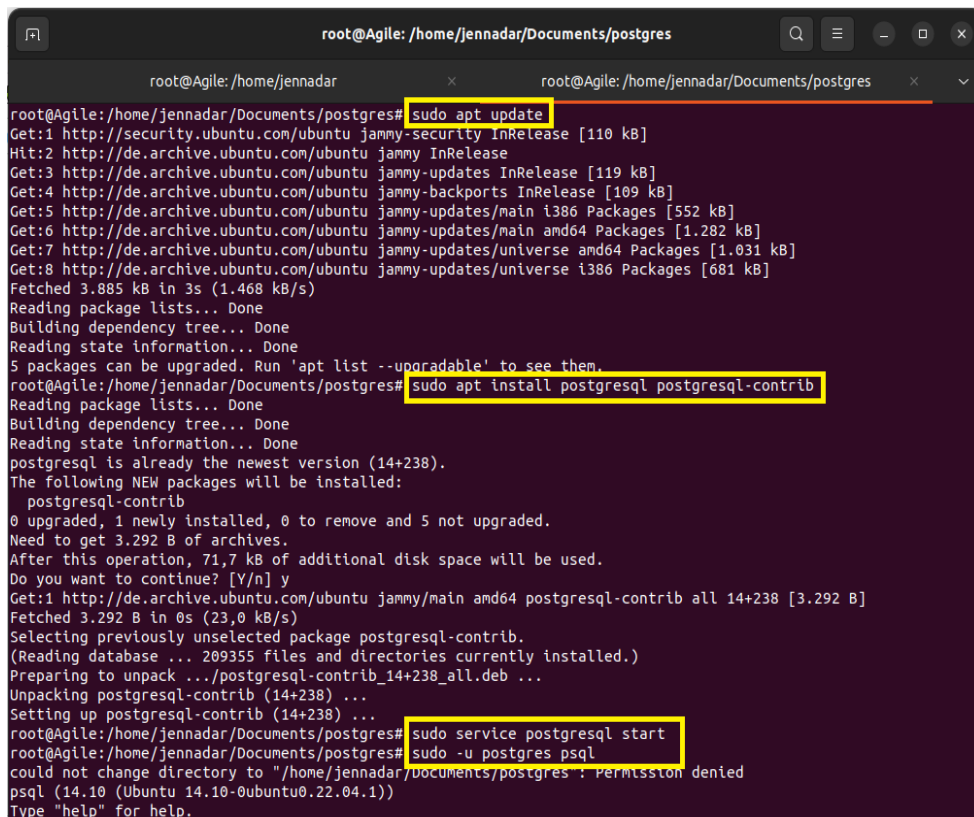
```
sudo apt-get install libpq-dev
```

Step2: Installing the postgresql driver

This driver will help to connect the postgresql server to another application.

```
pip install psycopg2-binary
```

```
pip install psycopg2
```



```
root@Agile: /home/jennadar/Documents/postgres# sudo apt update
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:2 http://de.archive.ubuntu.com/ubuntu jammy InRelease
Get:3 http://de.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:4 http://de.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:5 http://de.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [552 kB]
Get:6 http://de.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1.282 kB]
Get:7 http://de.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1.031 kB]
Get:8 http://de.archive.ubuntu.com/ubuntu jammy-updates/universe i386 Packages [681 kB]
Fetched 3.885 kB in 3s (1.468 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
5 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@Agile: /home/jennadar/Documents/postgres# sudo apt install postgresql postgresql-contrib
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
postgresql is already the newest version (14+238).
The following NEW packages will be installed:
  postgresql-contrib
0 upgraded, 1 newly installed, 0 to remove and 5 not upgraded.
Need to get 3.292 B of archives.
After this operation, 71,7 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://de.archive.ubuntu.com/ubuntu jammy/main amd64 postgresql-contrib all 14+238 [3.292 B]
Fetched 3.292 B in 0s (23,0 kB/s)
Selecting previously unselected package postgresql-contrib.
(Reading database ... 209355 files and directories currently installed.)
Preparing to unpack .../postgresql-contrib_14+238_all.deb ...
Unpacking postgresql-contrib (14+238) ...
Setting up postgresql-contrib (14+238) ...
root@Agile: /home/jennadar/Documents/postgres# sudo service postgresql start
root@Agile: /home/jennadar/Documents/postgres# sudo -u postgres psql
could not change directory to "/home/jennadar/Documents/postgres": Permission denied
psql (14.10 (Ubuntu 14.10-0ubuntu0.22.04.1))
Type "help" for help.
```

Fig 12: Installing PostgreSQL driver.

Step3: Creating an SQLAlchemy URI:

We need a SQLAlchemy Uri to be generated so that we can use this URI to connect postgresql and Apache Superset. To create this uri, we will install the sqlalchemy packages

```
pip install sqlalchemy
```

SQLAlchemy uri

```
postgresql+driver_name://username:password@localhost:5432/database_name
```

Configuring and working with Postgresql:

We need to start a postgresql server and create a database on this server.

To start a postgresql server you need to use the command

```
sudo service postgresql start
```

The postgresql server can be access via PostgreSQL shell

```
sudo -u postgres psql
```

While in the PostgreSQL shell you can create a new database and user if needed. Use following sql script.

```
CREATE DATABASE database_db; #Create a database
```

```
CREATE USER admin WITH ENCRYPTED PASSWORD 'changeme'; #Create a User
```



```
ALTER ROLE admin SET client_encoding TO 'utf8';
```

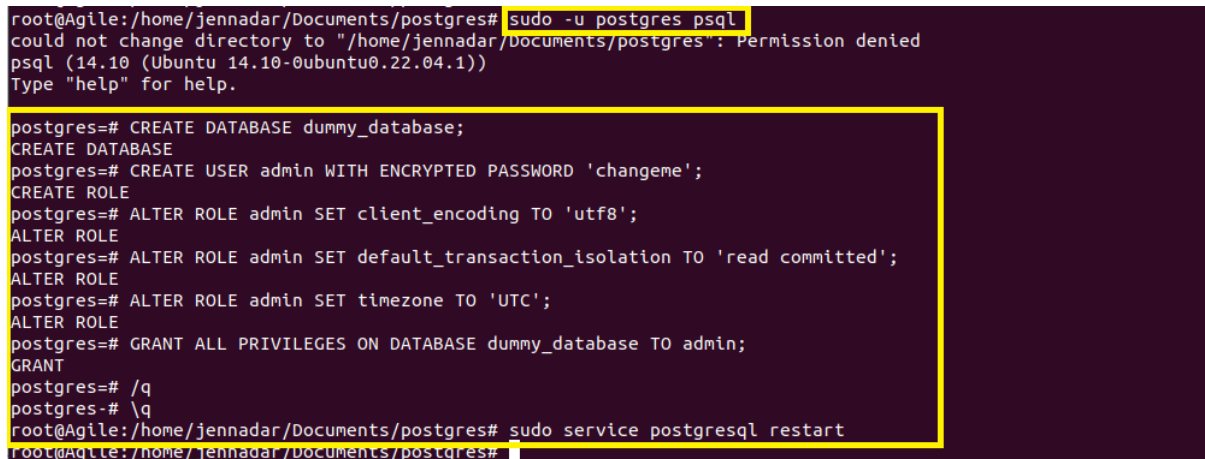
```
ALTER ROLE admin SET default_transaction_isolation TO 'read committed';
```

```
ALTER ROLE admin SET timezone TO 'UTC';
```

```
GRANT ALL PRIVILEGES ON DATABASE database_db TO admin; #Give permission to the database and user
```

Now restart the postgres server by using the command:

```
sudo service postgresql restart
```



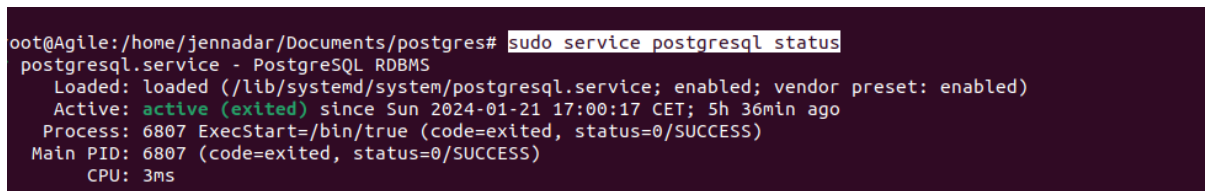
```
root@Agile:/home/jennadar/Documents/postgres# sudo -u postgres psql
could not change directory to "/home/jennadar/Documents/postgres": Permission denied
psql (14.10 (Ubuntu 14.10-0ubuntu0.22.04.1))
Type "help" for help.

postgres=# CREATE DATABASE dummy_database;
CREATE DATABASE
postgres=# CREATE USER admin WITH ENCRYPTED PASSWORD 'changeme';
CREATE ROLE
postgres=# ALTER ROLE admin SET client_encoding TO 'utf8';
ALTER ROLE
postgres=# ALTER ROLE admin SET default_transaction_isolation TO 'read committed';
ALTER ROLE
postgres=# ALTER ROLE admin SET timezone TO 'UTC';
ALTER ROLE
postgres=# GRANT ALL PRIVILEGES ON DATABASE dummy_database TO admin;
GRANT
postgres=# /q
postgres-# \q
root@Agile:/home/jennadar/Documents/postgres# sudo service postgresql restart
root@Agile:/home/jennadar/Documents/postgres#
```

Fig 13: Starting postgres service and creating Database and User.

To check the status of the server you can use

```
sudo service postgresql status
```



```
oot@Agile:/home/jennadar/Documents/postgres# sudo service postgresql status
postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; vendor preset: enabled)
   Active: active (exited) since Sun 2024-01-21 17:00:17 CET; 5h 36min ago
     Process: 6807 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
    Main PID: 6807 (code=exited, status=0/SUCCESS)
       CPU: 3ms
```

Fig14: Checking Postgres Server Status.

Configuring Apache Superset:

We need to configure Apache Superset, so that you don't need to provide the SECRET_KEY again and again when you start Apache superset. For this you need to configure your application, you need to create a file name “superset_config.py”. Add this file to the Superset config path using the command. export SUPERSET_CONFIG_PATH=/path/location/to/superset_config.py

Now the SECRET_KEY, ALCHEMY_URI, FLASK_APP variables will be set in this config file. So when you restart the Apache Superset then all these variables will be set and the script will connect the server that you created locally to Apache Superset.

Connecting Apache Superset and Postgresql Server:

An alternative way to connect the PostgreSQL Server is to copy the value of ALCHEMY_URI variable from the Superset_config.py file and paste it in the webpage of the Apache Superset under the database section.

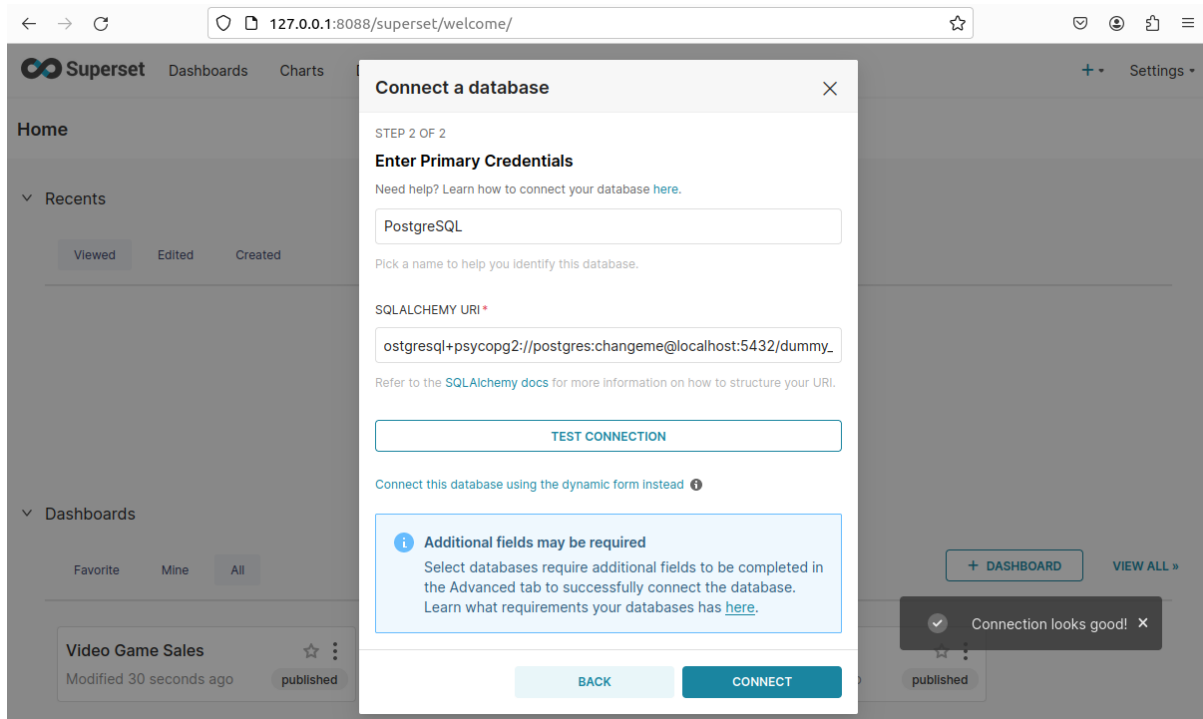


Fig15: Connecting Apache and PostgreSQL with ALCHEMY URI

When you click on the ‘Test Connection’ button it will ping you and tell you whether the URI is correct or not. Once you receive the ‘Connection looks good’ string then clicking on the ‘CONNECT’ button will form a secure connection between Apache Superset and the PostgreSQL server.

Creating tables in Postgresql server:

For creating a table, you can either write a SQL script on Apache Superset Application as shown.

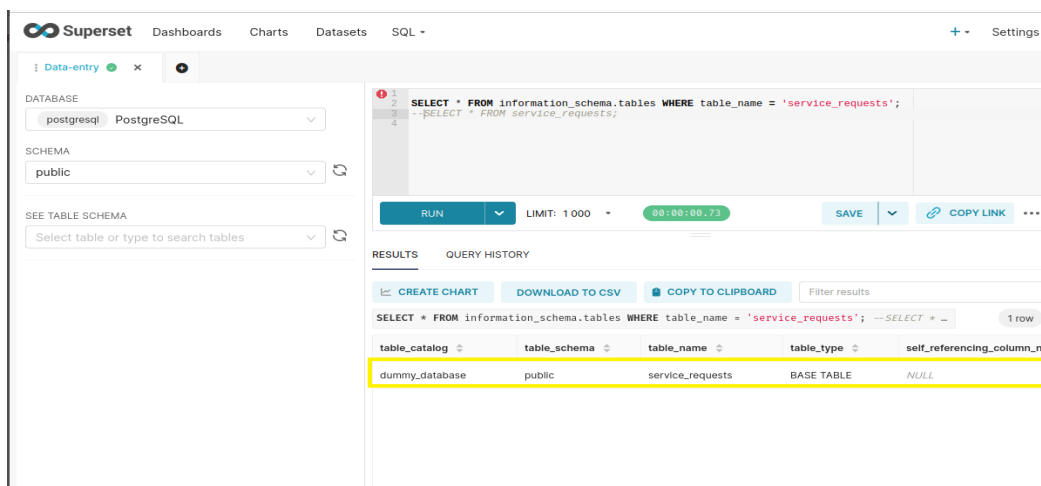


Fig16: Creating SQL Queries on Apache Superset SQL command prompt.

Another alternative is to write a python Script which will connect the Sql server to Apache Superset via a driver. This script will also read the data downloaded by another group API and store it in a csv format

and then create a SQL script depending on the data and upload the csv files as tables into the PostgreSQL server and commit the changes on Apache Superset.

Data Visualization:

Various chart types available in Apache Superset

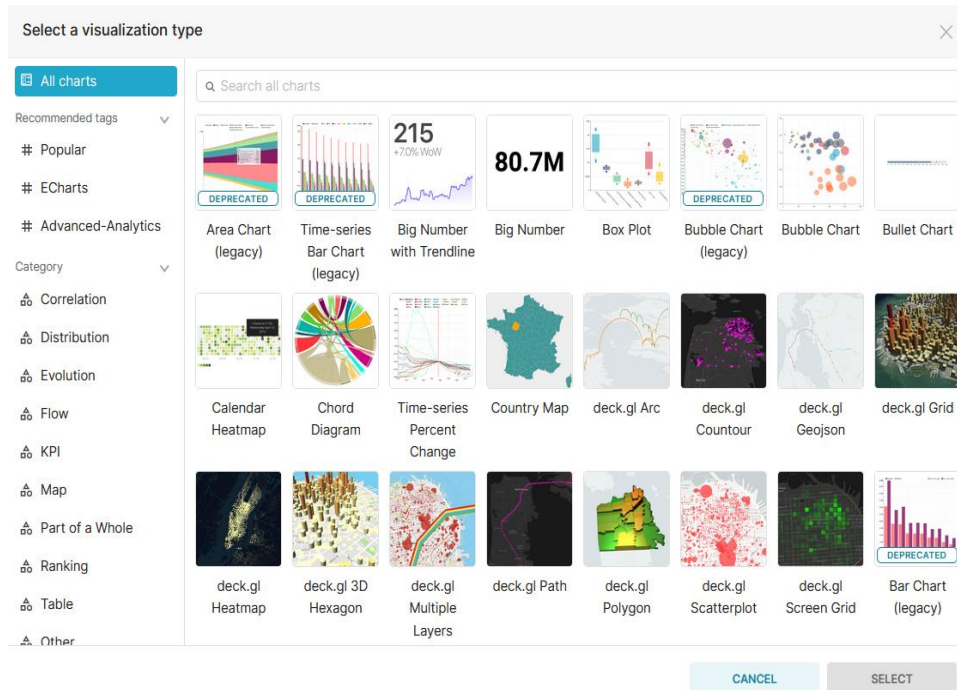


Fig17: Different kinds of chart present in Apache Superset

- **Line Chart:** The Line Chart in Apache Superset is used to represent data points as linked lines. This sort of chart is very good for displaying trends and patterns across time, making it easy for viewers to notice data fluctuations and changes.
- **Bar Chart:** This visualization uses rectangular bars to represent data. It is a useful tool for presenting relative amounts and trends within a dataset due to its ability to compare values across categories.
- **Area Chart:** It fills the space beneath the line with a visual representation of cumulative data. This chart is useful for showing patterns over time and emphasizing the overall size of numbers.
- **Time Series:** These are specialized visualizations designed for time-based data. These charts are necessary for studying and displaying temporal trends, making them an important component of time-sensitive data analysis.
- **Pie Chart:** It is a circular graph that depicts data in slices for different categories. It is a useful visualization for exhibiting bits of a larger picture and emphasizing the proportions of each category in the dataset.
- **Scatter Plot:** These represent individual data points on a two-dimensional graph. This sort of graphic is useful for spotting correlations and outliers in data, allowing for a more in-depth knowledge of data distributions.

- **Bubble Chart:** It is a Scatter Plot variant that incorporates the size of each data point to give it an extra dimension. Users can see three dimensions at once with this richer layout, giving the data more context.
- **Heatmap:** The Heatmap displays data as a matrix with colors denoting varying degrees of intensity. By emphasizing changes in values with color gradients, this sort of chart is very useful for exposing patterns and trends in vast datasets.
- **Table:** This provides a basic display of raw data in a tabular style. It is a crucial visualization for consumers who want a more thorough and organized view of the underlying dataset.
- **Treemap:** This represents hierarchical data with nested rectangles. This image is excellent for depicting hierarchical relationships within a dataset, which helps to understand the structure of complicated data hierarchies.
- **Box Plot:** These are useful for displaying statistical information regarding data distribution. They give a visual overview of the distribution, including the median, quartiles, and likely outliers, to help in the detection of data trends.
- **Sunburst chart:** This uses concentric circles to depict hierarchical data, allowing viewers to see how different levels of a hierarchy relate. This chart is very useful for displaying complicated hierarchical organizations.
- **Gantt Charts:** These charts are used to describe project schedules and timetables. They offer a visual depiction of project tasks, durations, and dependencies, making them a vital project management tool.
- **Bullet Chart:** This type of chart compares performance to a set of goals or benchmarks. It is an effective tool for tracking goal progress, which makes it simpler for users to evaluate performance against pre-established standards.
- **World Map:** Global geographic data is shown on a global scale with World Maps in Apache Superset. They let users see geographic trends and patterns in their datasets, which is extremely helpful for spatial analysis.
- **Time Cloud:** This uses a cloud-like visualization to show temporal data. This creative method offers an alternative viewpoint on the temporal patterns and trends found in the data.

Parameters and Customization:

The image demonstrates the platform's adaptability and user-centric design by highlighting important configuration options and parameters in Apache Superset. These capabilities highlight Superset's adaptability in meeting different data analysis and reporting demands by enabling users to customize their analytical experiences. The user-specific configurations serve as an example of how Apache Superset may be tailored and used to create a dynamic tool for a variety of users and analytical activities.

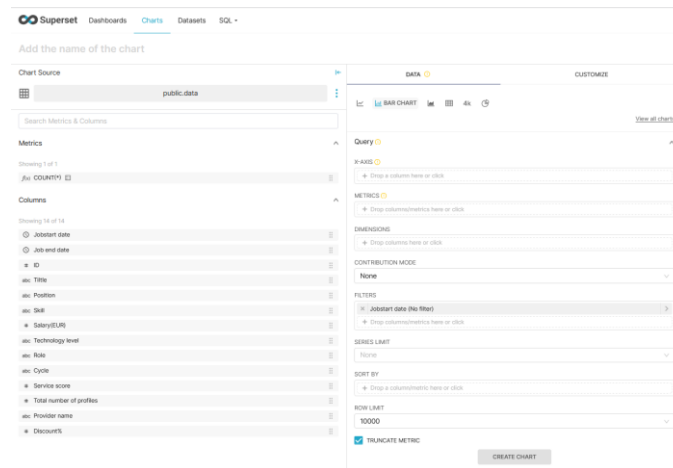


Fig18: Parameters to configure the charts

- **Group By:** Users can define how the data is grouped for aggregate using the Group By option. With categorical data, this is especially helpful since it allows users to aggregate and visualize data according to specified categories.
- **Metrics:** Metrics determine the quantitative values that will be presented in the visualization. Users may choose which numerical measurements to study, such as counts, sums, averages, or other appropriate metrics based on their dataset.
- **Filters:** Filters allow users to interactively limit down datasets based on certain criteria. This dynamic feature enables users to narrow in on certain sections of data and get more tailored insights.
- **Time Range:** The Time Range option is critical for creating time-based charts. It allows users to specify a time period for analysis. This feature is especially useful for evaluating temporal patterns, since it allows for a deep analysis of data within a certain era.
- **Color Scheme:** The Color Scheme field allows users to choose the color palette for the chart. This adjustment improves the visual attractiveness of the visualization and helps to convey extra information, such as category differences or data intensity.
- **Aggregation Functions:** Superset includes a variety of Aggregation Functions, including sum, average, and count. These capabilities allow users to summarize and analyze data in a variety of ways, depending on the analytical aims.
- **Sort Order:** The Sort Order option controls the order of data points in the display. Users can tailor the sorting criteria to emphasize particular trends or patterns in the data.
- **Legends:** Legends regulate how legends are shown in order to identify data categories. They are necessary for clarifying the meaning of the colors or symbols used in the chart, hence boosting general comprehension.
- **Axis Configuration:** It enables users to change axis labels, names, and scales. This option is useful for adjusting the chart's axes to better reflect the features of the data being shown.
- **Annotations:** Annotations allow you to add more information or custom text directly into the chart. This function is very useful for emphasizing certain data points, trends, or occurrences in the display.
- **Custom SQL Queries:** Advanced users will appreciate the option to use Custom SQL Queries, which give unequaled versatility. This feature enables users to get and change data directly using SQL queries, providing a strong tool for specialized analysis.

Integration Hurdles

Our first attempt to connect Apache Superset to a MySQL server was unsuccessful due to various challenges. Potential reasons for the failure include incorrect authentication credentials, misconfigured network settings, inadequate MySQL user privileges, server configuration issues, compatibility discrepancies, and firewall/security software hindrances. Troubleshooting efforts were impeded by limited error logging and debugging. To rectify this, meticulous review of authentication, network, and MySQL configurations is essential. Ensuring compatibility between Superset and MySQL versions, adjusting firewall rules, and consulting community resources for guidance are recommended steps. Addressing these factors comprehensively will enhance the likelihood of establishing a successful connection.

Later, efforts to connect Apache Superset to a SQLAlchemy server proved futile due to various reasons. Possible causes include incorrect authentication credentials, misconfigured networking settings, inadequate privileges for the SQLAlchemy user, server configuration discrepancies, version incompatibility issues, and interference from firewall or security measures. The failure to establish a connection was further compounded by limited error logging and debugging capabilities. To address this, a comprehensive review of SQLAlchemy authentication, networking, and server configurations is essential. Ensuring compatibility between Superset and SQLAlchemy versions, adjusting firewall settings, and seeking assistance from the SQLAlchemy community are recommended steps to overcome these challenges and successfully establish the connection.

Following the unsuccessful connection attempt between Apache Superset and SQLite, prompted by heightened security measures in Superset's upgraded version, we are pivoting towards a new strategy. Our focus now shifts to exploring alternative database options that align with Superset's security protocols while ensuring seamless integration. PostgreSQL emerges as a promising candidate due to its robust features and compatibility with Superset's upgraded framework. By leveraging PostgreSQL, renowned for its scalability and reliability, we anticipate overcoming the challenges encountered with SQLite. This strategic shift underscores our commitment to adopting solutions that not only meet our project requirements but also adhere to stringent security standards.

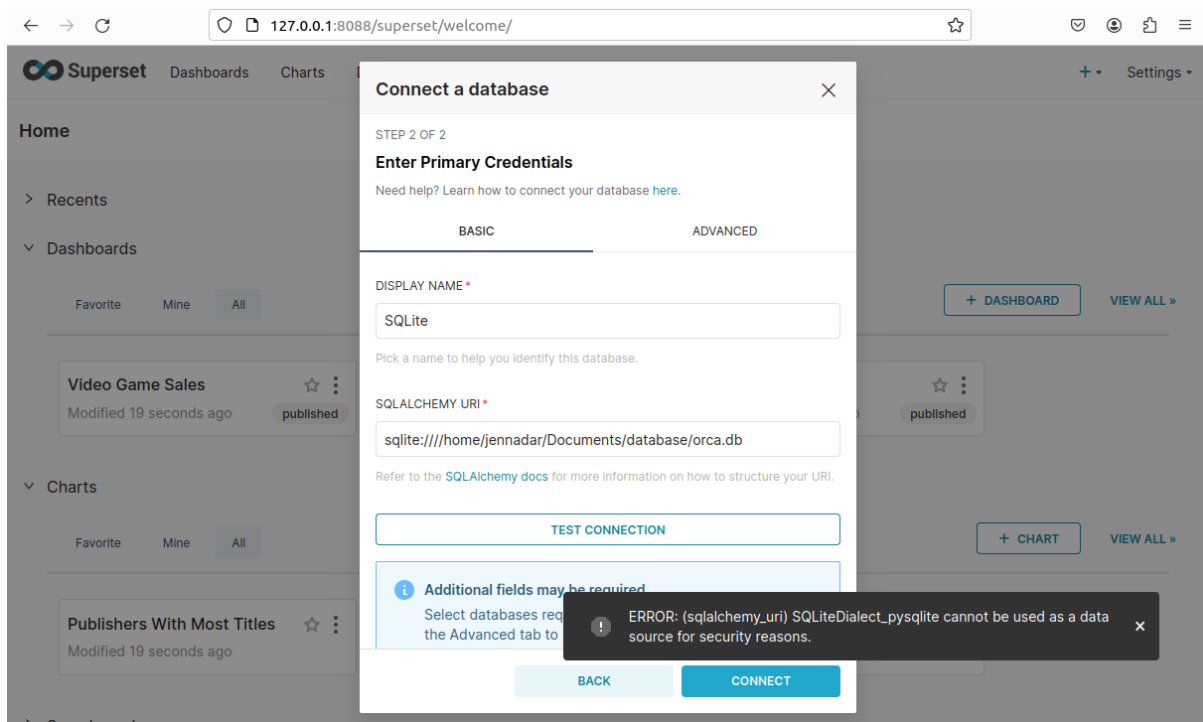


Fig 19: Failed integration of SQLite and Apache Superset.

In addressing this challenge, we strategically opted to transition to PostgreSQL as our SQL server solution, aiming to overcome compatibility issues and align seamlessly with Apache Superset's upgraded framework.

Output

1. Which provider offers most of the profiles? Which provider never offers profiles?

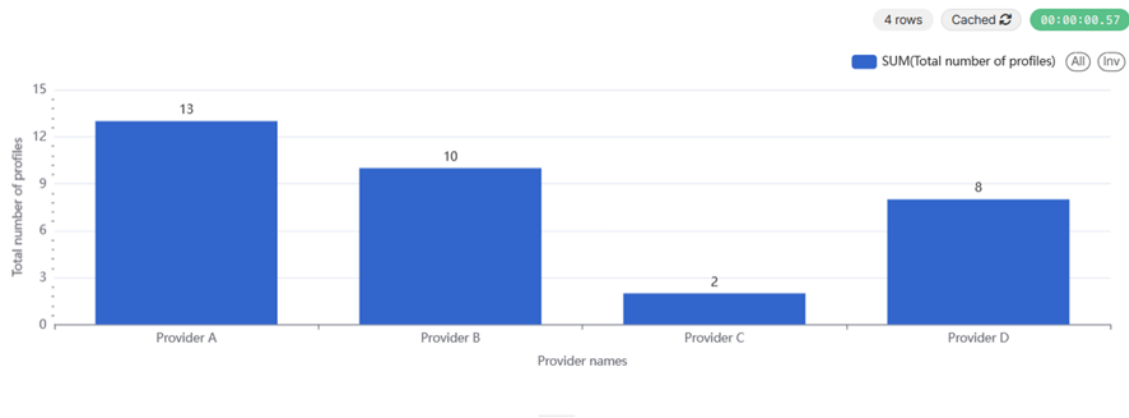


Fig20: Barchart

Provider A offers most of the profiles and there are no providers who do not provide profiles.

A vertical bar chart has been created, with the X-axis named "Provider Name" and a margin of 30. The Y-axis, labeled "Total Number of Profiles," has a margin of 30. The X and Y axes are formatted with "adaptive formatting." This graphic successfully gathers and shows data on the total number of profiles across different providers, ensuring the representation is clear and easy to understand.

2. Which provider has the best service score (evaluation) or rather delivers best quality?



Fig21: Sunburst Chart

Provider D has the best service score.

The Sunburst chart in Apache Superset is designed with provider names at its center, emphasizing hierarchy. The average service score is the fundamental indicator used to provide a brief performance

report. Users may use the graphic to learn how service scores are distributed among providers. The graphic, which now includes a job start date filter, provides for a more concentrated investigation over a certain time period. This setup enables rapid and intelligent comparisons of service ratings among providers, assisting with performance evaluation and decision-making.

3. How often does the company get discounts from providers? What is the average discount in percent

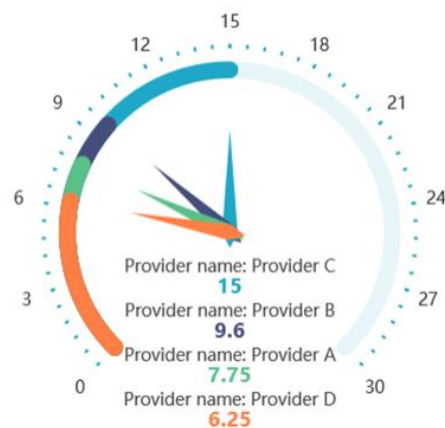


Fig22: Gauge Chart

Provider C gets the highest discount of 15%

The chart is set up with the dimension "Provider name" and the measure "Discount Percentage." This structure enables customers to simply examine and grasp how many discussions have taken place, as well as the regularity with which the organization obtains discounts from various providers.

4. Which IT services / service roles are frequently requested?

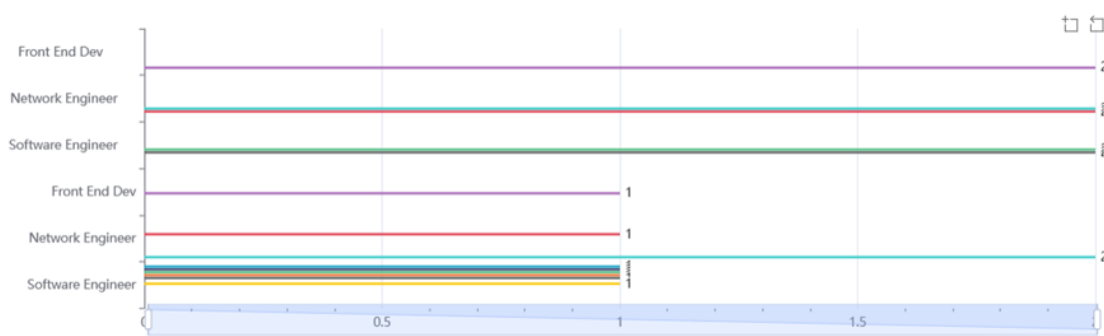


Fig23: Horizontal Bar graph

Front end Dev, Network Engineer, Software engineer roles are frequently requested

The chart is set up with "Role" as the dimension and "Total number of profiles" as the metric, offering a clear picture of the demand for various IT services and jobs in the dataset.

5. What are the periods for the service requests from creation until completion?

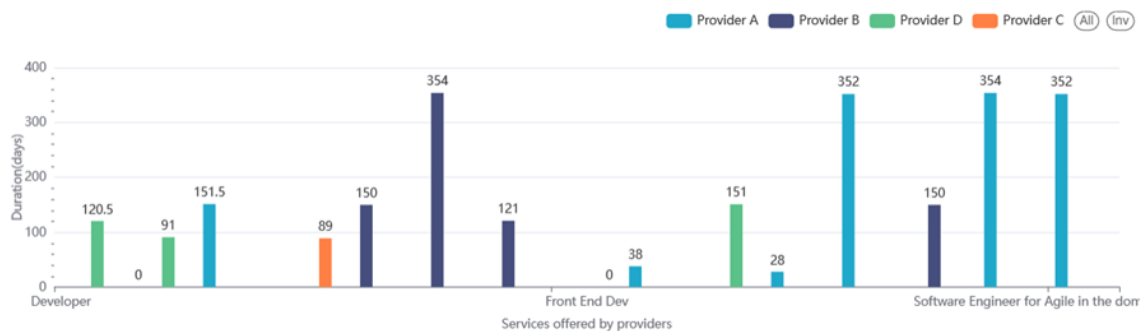


Fig24: Vertical bar graph

The time period for each service request is represented on a Vertical bar graph above.

The dimensions "Provider name" and "Duration" are the metrics that dictate the graph's arrangement. By showing the length on a vertical bar graph, users may easily compare and assess the average time periods, allowing them to see patterns and potential variations in the efficiency of service completion between providers.

6. What is the total amount of expense for all running service requests?

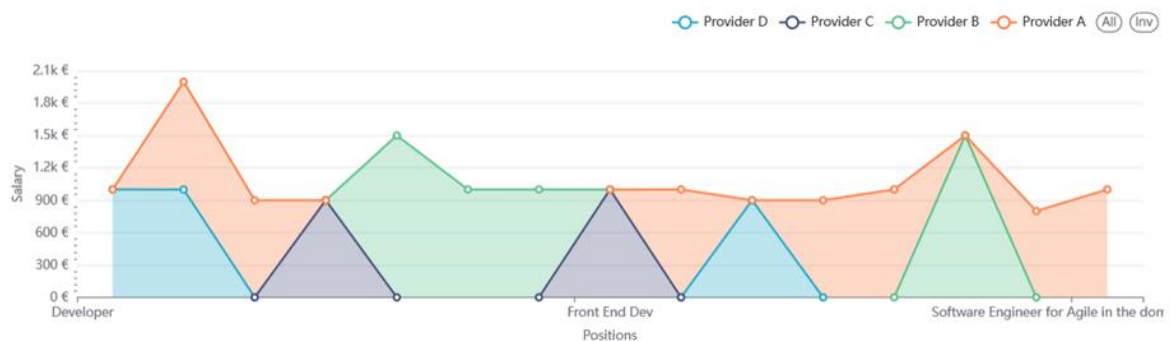
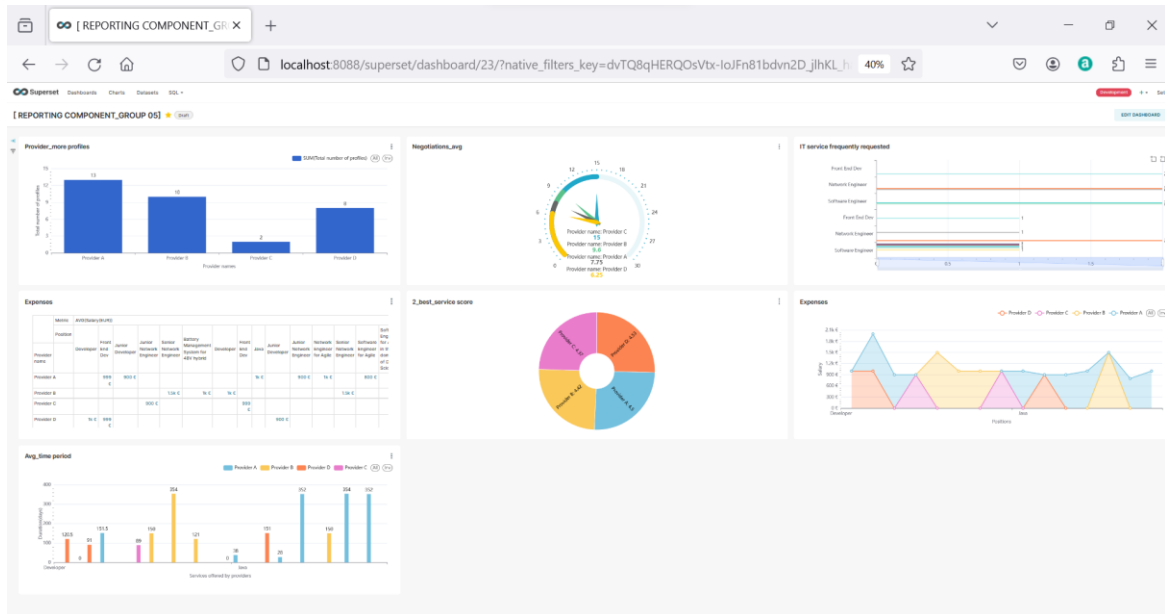


Fig25: Line Chart

The arrangement of the chart provides a targeted depiction of costs associated with service requests, with "Provider name" as the dimension and "Salary" as the metric. With the use of this graphic tool, users may visually see the trend in total costs, which may be shown on a quarterly or annual basis.

Result



The comprehensive analysis of the company's service operations is presented through an interactive and informative dashboard created using Apache Superset. This dashboard serves as a powerful tool for executives to visualize key metrics and make informed decisions. The dashboard, designed with user-friendly navigation and visually appealing charts, empowers the executive board to have a real-time overview of critical aspects of service operations. It enables quick decision-making, fosters transparency, and facilitates strategic planning based on data-driven insights. The use of Apache Superset ensures a seamless and intuitive user experience, enhancing the effectiveness of the dashboard as a strategic management tool.

In summary, the Apache Superset-powered dashboard is a robust and dynamic tool that empowers executives to gain actionable insights from the company's service operations. Its user-friendly design, real-time capabilities, and interactive features make it an invaluable asset for data-driven decision-making and strategic planning.

Conclusion

The adoption of Apache Superset as the data visualization and reporting component has proven to be a transformative leap for our organization, ushering in a new era of insightful decision-making and strategic planning. This robust platform has not only met but exceeded expectations, offering a myriad of features that have streamlined the analysis of our service operations. As we draw the curtains on this journey, it is evident that Apache Superset has become an indispensable ally in our pursuit of data-driven excellence.

Empowering Executives with Intuitive Visualization: One of the standout features of Apache Superset is its user-friendly interface. Executives, regardless of their technical background, can seamlessly navigate through complex datasets and visualize key metrics effortlessly. The interactive dashboards have become a hub for real-time insights, empowering decision-makers to make informed choices with just a glance.

Real-Time Analytics for Timely Decision-Making: Apache Superset's ability to provide real-time analytics has significantly elevated the agility of our decision-making processes. Executives can now access the latest information, ensuring that decisions are based on the most current trends and patterns. This real-time aspect has proven invaluable in adapting to the dynamic landscape of our service operations.

Versatility in Visual Representations: The platform's versatility in visual representations has allowed us to tailor the dashboard to our specific needs. Whether it's showcasing provider profiles, evaluating service quality, or delving into expense trends, Apache Superset offers a rich palette of visualization options. The ability to choose from various chart types and customize them has enhanced the clarity and impact of our reports.

Data Security and User Authentication: Recognizing the sensitivity of the data being handled, Apache Superset's robust security features, including user authentication, have instilled confidence in our executives. The secure login ensures that only authorized personnel have access to critical information, safeguarding our company's proprietary and confidential data.

Facilitating Collaboration and Continuous Improvement: The inclusion of collaborative features within Apache Superset has fostered a culture of continuous improvement. Executives and analysts can seamlessly collaborate, leaving notes, comments, and suggestions directly on the dashboard. This ongoing interaction ensures that the analysis remains relevant and aligns with the evolving needs of our organization.

Scalability and Adaptability: As our organization grows and evolves, Apache Superset has proven to be highly scalable and adaptable. It accommodates the increasing volume and complexity of data, ensuring that our data visualization and reporting capabilities remain robust and effective in meeting the evolving demands of our dynamic business environment.

In conclusion, Apache Superset has emerged as a cornerstone in our data analytics strategy, offering an unparalleled platform for visualizing, interpreting, and acting upon complex datasets. Its impact goes beyond mere reporting; it has become an essential tool in our decision-making arsenal. As we move forward, we are excited about the possibilities that Apache Superset continues to unlock for our organization, propelling us toward a future where data-driven insights steer our success.