Superset/Caravel

Originally developed by Airbnb and documented by

Gowtham Sai (Gowtham95india@gmail.com)

Introduction:

Superset is data exploration platform designed to be visual intuitive and interactive.

Features:

1. A rich set of data visualisation, integrated from some of the best visualization libraries.
2. Create and share simple dashboards.
3. An extensible, high-granularity security/permission model allowing intricate rules on who can access individual features and the dataset.
4. Enterprise-ready authentication with integration with major authentication providers (database, OpenID, LDAP, OAuth & REMOTE_USER through Flask AppBuilder).
5. A simple semantic layer, allowing users to control how data sources are displayed in the UI by defining which fields should show up in which drop-down and which aggregation and function metrics are made available to the user.
6. Integration with most RDBMS through SqlAlchemy.
7. Deep integration with Druid.io.

Installation:

Dependencies:

Cryptography is the major dependency, a Python library to encrypt connection passwords.

Installation Instructions:

Go through Superset Installation document and follow OS Dependencies, PIP Upgrade.

Please read explanation of each step of Superset Installation and Initialisation here (If you don't want to use default configuration of Superset) else continue with that part.

Configuration:

There are 2 ways to load the configuration in superset. One way is to modify the config.py file from which Superset loads all the default configuration. Second way is to use custom configuration file and put all the custom configurations in that file. You can find the sample test superset_config.py file here.

Superset will read superset_config.py file if and only if the file is in PYTHONPATH which means site-packages directory. You can find the directory using the below line of code shooting it directly in terminal.

```
1. # Python one liner to find site packages. Use from terminal directly.
2. python -c "import site; print site.getsitepackages()"
```

In Ubuntu machines you might be getting some different output which gives the path of dist-packages. If it is the case, you can navigate to parent directory from dist-packages and you can find site-pacakages.

If you check superset_config.py file, we are getting all the values from environment variables. To avoid loss of global variable, we will make it system wide and persistent between swtich user by using superset_variables.py file to **/etc/prfile.d/** directory. When you switch the user, all the environment variables will be lost. To avoid this, we are using sudo with -E tag. Please feel free to modify the superset_variables.py file. Just in case, if you made any changes to the file, you need to exit the session and login again or you can export the changed values or if want to remove you can use unset to remove the values.

Important things in this configuration file is setting SQLALCHEMY_DATABASE_URI. It accepts only SQALI URL's. Please go through SQLI URL's to use the appropriate URL for the database you want to use. Another important thing is to use CACHE_CONFIG to fasten up the results in Asynchronous Queries and Dashboard Rendering. CACHE_CONFIG should be a JSON like the below one.

```
CACHE_CONFIG="{\"CACHE_TYPE\": \"redis\", \"CACHE_DEFAULT_TIMEOUT\": 300, \"CACHE_KEY_PREFIX\": \"caravel_\", \"CACHE_REDIS_HOST\": \"127.0.0.1\", \"CACHE_REDIS_PORT\": 6379, \"CACHE_REDIS_DB\": 1, \"CACHE_REDIS_URL\": \"redis://localhost:6379/\"}"
```

To run asynchronous tasks, you need to configure CeleryConfig in config.py file with BROKER_URL, CELERY_RESULT_BACKEND, CELERY_ANNOTATIONS.

Industry best practice is to use RabbitMQ as the BROKER_URL and Redis as the CELERY_RESULT_BACKEND. We assume that you are following this guide for production environment and hence we used all CeleryConfig with the RabbitMQ as BROKER_URL and Redis as CELERY_RESULT_BACKEND. If you don't want to use them, please config.py and comment out the CeleryConfig class and set the value to None.

To install RabbitMQ Server and Redis (Ubuntu):

```
# By default AWS Ubuntu stack having this required repo. If not You can install from RabbitMQ
apt-get install rabbitmq-server
```

To Install RabbitMQ Serevr (CentOS):

```
# By default CentOS won't be having package. You can follow RabbitMQ guide.
yum -y update
yum install wget
wget http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
wget http://rpms.famillecollet.com/enterprise/remi-release-6.rpm
sudo rpm -Uvh remi-release-6*.rpm epel-release-6*.rpm
yum install -y erlang
rpm --import http://www.rabbitmq.com/rabbitmq-signing-key-public.asc
yum install rabbitmq-server

# To start the server once installed.
rabbitmq-server
```

If you get any issues, in installing them, please find support online. Best way is to read the logs and understand what is causing the issue. If in

case you couldn't get help even after checking the stackoverflow and Online forums, mail the log to author or me and raise issue in Github if is not raised already.

Don't forget to setup the RESULTS_BACKEND to redis.Redis() after importing redis in config.py.

How to Start:

Before firing up the server, start all the dependencies.

To Start redis:

```
# To start Redis (For Caching)
redis-server

# To RabbitMQ Server (As a Broker)
rabbitmq-server

# To Start Celery (For Tasks)
celery -A superset.sql_lab:celery_app worker --loglevel=debug -autoscale=5,15
```

It is good practice to run them using screen with -S tag to name them as these are continuously required to run in the backend. You can now start the main app server by shooting runserver command.

```
# Start the local server
superset runserver
```

Local server will be running on port **8088** by default. You can always change this using --port. If you are behind firewall make sure, port 8088 is open for inbound traffic.

If interested continue reading configuring Nginx as main server. It is always good practice to use Nginx/Apache as main server above your local server.

Nginx Installation & Configuration:

You can install Nginx easily by using the following commands.

To install Nginx in Ubuntu Machine:

```
# Nginx Installation.
apt-get install nginx
```

To install Nginx in CentOS:

```
# Nginx Installation.
yum install nginx
```

Create file with extension conf in the directory **/etc/nginx/conf.d** and insert the following lines into that file. Change the server_name with the domain that points to the machine and proxy_pass with url where the server

is running. Make sure you can access from Nginx Server to App Server if both are running on different machines.

```
server {
  listen       80;
  server_name  mrv.gowtham-sai.com;
  location / {
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header Host $http_host;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_http_version 1.1;
    proxy_cache_bypass $http_upgrade;
    proxy_pass http://127.0.0.1:8088;
  }
}
```

Please notice that all the configurations in config.py file will be over written by superset_config.py file.

Explanation of Superset Installation Steps:

```
# Install superset
pip install superset
```

This step will install superset from python library pack. Possible error message will be **bash: pip command not found.** Which means pip is not installed in the system. To fix this issue please go through Installation and follow the steps mentioned there.

Before following this step, if you want to use tweaked code, you can use the following line to install it from Github. (Voonik required source).

```
# Install superset from Github (Voonik Required Source)
pip install git+https://github.com/Gowtham95india/superset.git
```

```
# Create Admin User
fabmanager create-admin --app superset
```

This step will create admin user. Admin user is like super user of linux. He will be having all the permission by default. All the permissions will be initialized in the next step.

```
# Initialize the database
superset db upgrade
```

This step will create database skeleton structure. While it runs sit back and relax.

Before following this step, if you want to configure any other database instead of default sql db used by superset, please view configuration section here.

```
# Create default roles and permissions
superset init
```

This step will create default roles and permissions. All these roles can be configured through config file. For detailed explanation you can check configuration section of the document.

```
# Start the local server
superset runserver
```

This will run the superset server in production mode on port **8088** which can be configured through config file or via the command using **-p** tag. If you want this to run in development mode by using **-d** tag.

* Superset is tested using Python 2.7 and Python 3.4+. Python 3 is the recommended version; Python 2.6 won't be supported. The project was original named as **Panoramix,** was renamed to **Caravel** in March 2016, and is currently named as **Superset** as of November 2016.