Maestro Server - Cloud Inventory Documentation

Release 0.6

Felipe Signorini

Jul 03, 2020

Contents:

1 Overview **3**
- 1.1 What is Maestro Server . . . 3
- 1.2 What problems does it solve? . . . 3
- 1.3 How do I use it? . . . 4

2 Quick Start **7**
- 2.1 Running locally . . . 7
- 2.2 Persistent data . . . 7
- 2.3 Using external Database . . . 8
- 2.4 Using external RabbitMQ . . . 8
- 2.5 Using S3 to store files . . . 8
- 2.6 Using external SMTP . . . 9
- 2.7 Complete docker compose . . . 9

3 Installing Maestro **11**
- 3.1 Using Docker Compose . . . 11
  - 3.1.1 Overview . . . 11
  - 3.1.2 Running locally . . . 11
  - 3.1.3 Spin up the API server in a different server . . . 15
  - 3.1.4 Productionize . . . 15
- 3.2 Advanced setups . . . 16
  - 3.2.1 SMTP Config . . . 16
  - 3.2.2 Using external store engine as S3 . . . 16
  - 3.2.3 Using external Database . . . 18
  - 3.2.4 Using external RabbitMQ . . . 19
  - 3.2.5 JWT Tokens . . . 20
  - 3.2.6 Service Discovery Configuration . . . 23
  - 3.2.7 Themes . . . 24
- 3.3 Services configurations . . . 27
  - 3.3.1 High Architecture . . . 27
  - 3.3.2 Client App . . . 28
  - 3.3.3 Server APP . . . 29
  - 3.3.4 Discovery App . . . 30
  - 3.3.5 Reports App . . . 31
  - 3.3.6 Analytics App . . . 32
  - 3.3.7 Analytics Front . . . 33
  - 3.3.8 Data App . . . 34

3.3.9 Scheduler App . . . 35
3.3.10 Audit App . . . 36
3.3.11 WebSocket App . . . 37
3.4 High availability . . . 37
3.4.1 12 Factory and Horizontal Scaling . . . 37
3.4.2 Scheduler Beat App . . . 40
3.4.3 HealthChecks . . . 40
3.4.4 Running on Kubernetes . . . 40

4 User Guide **43**
4.1 Cloud Inventory . . . 43
4.1.1 Inventory . . . 44
4.1.2 Options and configurations . . . 59
4.1.3 History Track . . . 61
4.2 Auto Discovery . . . 62
4.2.1 Connecting on AWS . . . 63
4.2.2 Connecting on Azure . . . 65
4.2.3 Connecting on Digital Ocean . . . 67
4.2.4 Digital Ocean Spaces . . . 69
4.2.5 Connecting on OpenStack . . . 71
4.2.6 Using Ansible Facts . . . 73
4.2.7 Using Terrafom State File . . . 76
4.2.8 Import using JSON files . . . 77
4.3 Graphs - Architecture maps . . . 79
4.3.1 Business Graphs . . . 79
4.3.2 Using the dependency tree wizard . . . 84
4.4 Reports - Generate advanced reports . . . 86
4.4.1 Reports . . . 86
4.4.2 Report Charts . . . 91
4.4.3 Scheduler . . . 92
4.5 ACLs - Users and Teams . . . 94
4.5.1 Access rules . . . 94
4.5.2 Teams . . . 96

5 Developer Guide **99**
5.1 Architecture . . . 99
5.1.1 FrontEnd - Client App . . . 100
5.1.2 Server App . . . 102
5.1.3 Discovery App . . . 105
5.1.4 Reports App . . . 108
5.1.5 Scheduler App . . . 110
5.1.6 Analytics Maestro . . . 112
5.1.7 Analytics Front . . . 114
5.1.8 Data APP . . . 116
5.1.9 Audit App . . . 118
5.1.10 WebSocket APP . . . 120
5.2 APIs . . . 121
5.2.1 Server API . . . 122
5.2.2 Discovery API . . . 122
5.2.3 Report API . . . 122
5.2.4 Analytics API . . . 122
5.2.5 Data API . . . 122
5.2.6 Analytics Front API . . . 122
5.2.7 Audit API . . . 122

5.3 Graphs Analytics Algorithm . . . 122
5.3.1 Making graph lookup on the mongodb . . . 123
5.3.2 Creating a networkX graph . . . 124
5.3.3 Rules . . . 125
5.3.4 Enrichment data phase . . . 128
5.3.5 Draw phase . . . 129
5.4 Lints . . . 131
5.4.1 JavaScript (Client App) . . . 131
5.4.2 NodeJs (Server App) . . . 131
5.4.3 Python 3 (Discovery, Scheduler and Reports) . . . 132
5.5 Tests . . . 132
5.5.1 Server APP . . . 132
5.5.2 Discovery APP . . . 132
5.5.3 Reports APP . . . 132
5.5.4 Data Layer APP . . . 133
5.5.5 Analytics Apps . . . 133
5.5.6 Analytics Front . . . 133
5.5.7 Audit App . . . 133
5.6 Quality Assurance . . . 134
5.6.1 Client Maestro . . . 134
5.6.2 Server App . . . 134
5.6.3 Discovery Maestro . . . 134
5.6.4 Report Maestro . . . 134
5.6.5 Scheduler Maestro . . . 135
5.6.6 Data Layer API . . . 135
5.6.7 Analytics App . . . 135
5.6.8 Analytics Front . . . 135
5.6.9 Audit App . . . 135
5.7 Third Party . . . 136
5.8 CI and CD . . . 136
5.9 Versions . . . 136
5.9.1 v0.6x - Candidate release . . . 136
5.9.2 v0.5x - Beta . . . 137
5.9.3 v0.4x - Beta . . . 137
5.9.4 v0.3x - Beta . . . 137
5.9.5 v0.2x - Alpha . . . 138

6 Troubleshooting **139**

7 Contrib **141**
7.1 Reporting issues . . . 141
7.2 Submitting patches . . . 141

8 Donate **143**

9 Contact **145**
9.1 Feature request . . . 145

10 License **147**

The docs are separated into 3 parts, the first one is about installation and setup Maestro, the second is User Guide and how you create and manage Maestro in the business point of view, and the last we have a developer guide to help to contribute for the project.

CHAPTER 1

Overview

1.1 What is Maestro Server

Maestro Server is an open source software platform for management and discovery servers, apps and system for Hybrid IT. Can manage small and large environments, to be able to visualize the latest multi-cloud environment state.

You will be able to:

- Centralize and visualize the latest state multi-cloud environment
- Continuously discover new servers and services of all environments
- Powerful reports, you can create a relation with servers, services, apps and clients
- Automatically populate inventory with ansible, logging jobs, audit and cordenate multiple teams.
- Tracking all changes of your infrastructure

1.2 What problems does it solve?

Maestro had built to solve some problems founded in operating multi-cloud environments, multi shared devops culture and multi clients, where turns hard to keep track the latest environment state, bottlenecks to apply a compliance in all teams, visualization gaps to understand the infrastructure state, access security flaws for internals employees and out of date documentation.

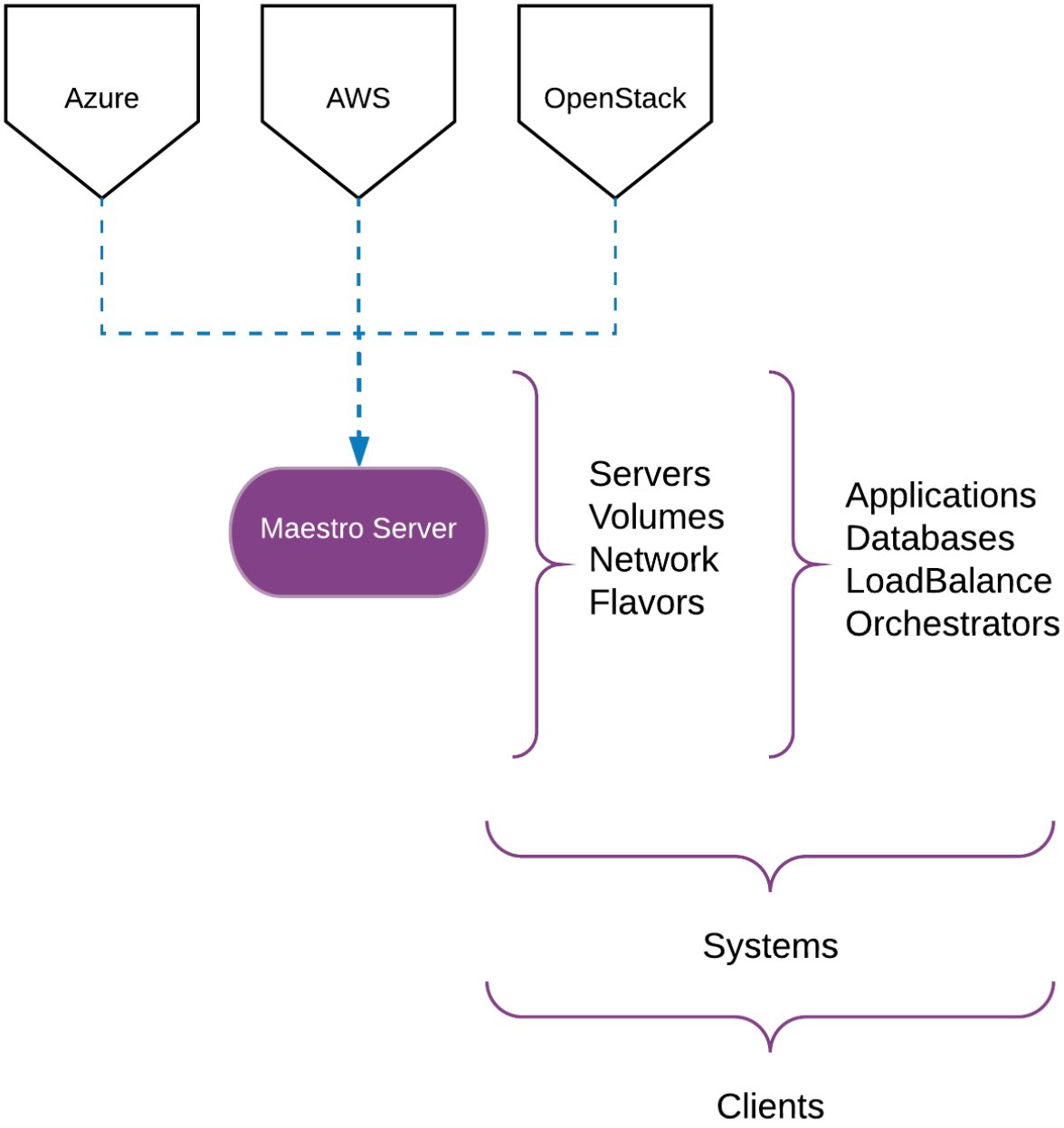
- How can we audit your env?
- How control and keep track your environment?
- How garantee if my documentation is updated?
- Witch servers belong this client?

Maestro comes to help IT operation teams to organize and audit multicloud infrastructure, it come to substitute CMDB systems, auto-discovery servers, services and apps, be organized in a smart way, it's possible to classify each service, like database, message queues, vpns, api gateway, service mesh and etc, to create a relation between servers and

services, docs clusters and points target, relate services, system and clients. Maestro come for you, to be a complete and simple cloud inventory.

1.3 How do I use it?

It able to analysis your full state environment of all providers you have, centralize all information about datacenters, servers, loadbalance, orchestrations tools, volumes, vpns and etc, keep track their relations, can create complex and powerful reports, analysis costs, growing up velocity, standards services names, network configurations and available deploys for each server.



See demo cloud inventory here.

CHAPTER 2

Quick Start

It had three ways to install maestro. The quick one is to use a standalone docker [easy way], if you like more control over the installation, you can run multiple docker images per service [Recommended], and the last you can install from the source [Dev].

2.1 Running locally

You can use a standalone docker to spin up a single maestro instance.

```
docker run -p 80:80 -p 8888:8888 -p 8000:8000 -p 9999:9999 maestroserver/standalone-
↪maestro
```

- You need to expose ports **80**, **8888**, **8000** and **9999**
- You can access by browser over 80 port.

2.2 Persistent data

Docker have a empheral disk, with means if you remove the container all data will be lost. You can handle it making volumes, the list of folder to expose are:

- **/data/db:** It is all data recorded on mongo db.
- **/data/server-app/public/:** Profile images uploaded
- **/data/analytics-front/public:** Architecture artifacts exposed externally.

```
mkdir ./db ./server/public ./analytics/public

docker run
-v ./db:/data/db
-v ./server/public:/data/server-app/public/
```

(continues on next page)

(continued from previous page)

```
-v ./analytics/public:/data/analytics-front/public
maestroserver/standalone-maestro
```

2.3 Using external Database

It do recommend to spin up a mongodb externally, you can set the `MAESTRO_MONGO_URI` env variable.

Env Variables	Default	Description
MAESTRO_MONGO_URI	mongodb://localhost:27017	Can be mongodb or mongo+srv://

As an example

```
docker run
    -p 80:80
    -p 8888:8888
    -p 8000:8000
    -p 9999:9999
    -e MAESTRO_MONGO_URI=mongodb://external.mongo.com:27017
    maestroserver/standalone-maestro
```

Optionally, you can replace the db name, setting the `MAESTRO_MONGO_DATABASE` env var.

Env Variables	Default	Description
MAESTRO_MONGO_DATABASE	maestro-client	Database name

2.4 Using external RabbitMQ

You can spin up a rabbitmq externally, it's uses *CELERY_BROKER_URL* env variable.

Env Variables	Default	Description
CELERY_BROKER_URL	amqp://localhost:5672	Amqp endpoint

```
docker run
    -p 80:80
    -p 8888:8888
    -p 8000:8000
    -p 9999:9999
    -e CELERY_BROKER_URL=amqp://external.rabbitmq.com:5672
    maestroserver/standalone-maestro
```

2.5 Using S3 to store files

You can use S3 Amazon storage object service to store artifacts and profiles images over a reliable storage system.

Env variables

UPLOAD_TYPE	S3
AWS_ACCESS_KEY_ID	XXXXXXXXXX
AWS_SECRET_ACCESS_KEY	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
AWS_DEFAULT_REGION	us-east-1
AWS_S3_BUCKET_NAME	maestroserver

```
docker run
    -e AWS_ACCESS_KEY_ID='XXXXXXXXXX'
    -e AWS_SECRET_ACCESS_KEY='XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
    -e AWS_DEFAULT_REGION='us-east-1'
    maestroserver/standalone-maestro
```

2.6 Using external SMTP

You can use a external smtp service as SendGrid, AWS SeS or any smtp server. Go to server application and set:

SMTP_PORT	
SMTP_HOST	
SMTP_SENDER	
SMTP_USERNAME	
SMTP_PASSWORD	
SMTP_USETSL	Enable TLS connect
SMTP_IGNORE	Ignore the validation of security connection

```
docker run
    -e SMTP_PORT=465
    -e SMTP_HOST=smtp.gmail.com
    -e SMTP_SENDER='mysender@gmail.com'
    -e SMTP_USERNAME=myusername
    -e SMTP_PASSWORD=mysecret
    -e SMTP_USETSL=true
    maestroserver/standalone-maestro
```

2.7 Complete docker compose

Minimal setup

```
services:
    maestro:
        image: maestroserver/standalone-maestro
        ports:
        - 80:80
        - 8888:8888
        - 8000:8000
        - 9999:9999
        volumes:
        - mongodata:/data/db
        - artifacts_server:/data/server-app/public/
        - artifacts_analytics:/data/artifacts
```

(continues on next page)

(continued from previous page)

```
volumes:
    mongodata: {}
    artifacts_server: {}
    artifacts_analytics: {}
```

Recommended reliable setup, using a mongodb, rabbitmq, smtp and store file set externally.

```
services:
    maestro:
        image: maestroserver/standalone-maestro
        ports:
        - 80:80
        - 8888:8888
        - 8000:8000
        - 9999:9999
        environment:
        - AWS_ACCESS_KEY_ID='XXXXXXXXXX'
        - AWS_SECRET_ACCESS_KEY='XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
        - AWS_DEFAULT_REGION='us-east-1'
        - MAESTRO_MONGO_URI=mongodb://external.mongo.com:27017
        - CELERY_BROKER_URL=amqp://external.rabbitmq.com:5672
        - SMTP_PORT=465
        - SMTP_HOST=smtp.gmail.com
        - SMTP_SENDER='mysender@gmail.com'
        - SMTP_USERNAME=myusername
        - SMTP_PASSWORD=mysecret
        - SMTP_USETSL=true
```

Note: Standalone docker use the same env vars found it in all services.

Note: Standalone uses supervisord to manage all services inside of one docker, if you like to spin up one docker per service, go to installation.

Warning: Don't spin up a multiple standalone docker, it will duplicate the schedule tasks, if you need to make a production high availability setup, go to installation per service.

CHAPTER 3

Installing Maestro

3.1 Using Docker Compose

To get Maestro up in just a few minutes go to Standalone installation.; However if you like to get more control over the installation you can spin up a one docker per service.

3.1.1 Overview

There are a list of all services:

Client App	FrontEnd client	Vue2 + Bootstrap 3
Server App	Primary API, authentication, crud and manager	NodeJs 8.11 Kraken
Discovery App	Auto discovery and crawlers	Python 3.6, flask
Scheduler App	Jobs manager with celery beat	Python 3.6, celery
Reports App	Reports generator	Python 3.6, flask
Analytics App	Analytics Maestro - Graphs Generator	Python 3.6, flask
Analytics Front	Analytics Front	NodeJs 8.11 Kraken
Data DB App	Data layer	Python 3.6, flask
Audit App	History tracker service	NodeJs 8.11 Kraken
WebSocket APP	WebSocket - Events	Go, Centrifugo

3.1.2 Running locally

You can use docker to spin up a maestro bundle, you can copy and execute the docker-compose file describe below.

Note: PS: Docker compose will be able to create and manager all networks and communication between services.

PS: Containers is prepared to run in production.

Note: Download docker-compose file.

```
version: '3'

services:
    client:
        image: maestroserver/client-maestro
        ports:
        - "80:80"
        environment:
        - "API_URL=http://localhost:8888"
        - "STATIC_URL=http://localhost:8888/static/" # <- It need to have the slash
        - "ANALYTICS_URL=http://localhost:9999"
        - "WEBSOCKET_URL=ws://localhost:8000"
        depends_on:
        - server

    server:
        image: maestroserver/server-maestro
        ports:
        - "8888:8888"
        environment:
        - "MAESTRO_MONGO_URI=mongodb://mongodb"
        - "MAESTRO_MONGO_DATABASE=maestro-client"
        - "MAESTRO_DISCOVERY_URI=http://discovery:5000"
        - "MAESTRO_ANALYTICS_URI=http://analytics:5020"
        - "MAESTRO_ANALYTICS_FRONT_URI=http://analytics_front:9999"
        - "MAESTRO_REPORT_URI=http://reports:5005"
        - "SMTP_PORT=25"
        - "SMTP_HOST=maildev"
        - "SMTP_SENDER=myemail@gmail.com"
        - "SMTP_IGNORE=true"
        volumes:
        - artifacts_server:/data/public/
        depends_on:
        - mongodb
        - discovery
        - reports

    discovery:
        image: maestroserver/discovery-maestro
        ports:
        - "5000:5000"
        environment:
        - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
        - "MAESTRO_DATA_URI=http://data:5010"
        depends_on:
        - rabbitmq
        - data

    discovery_worker:
        image: maestroserver/discovery-maestro-celery
        environment:
        - "MAESTRO_DATA_URI=http://data:5010"
        - "MAESTRO_WEBSOCKET_URI=http://ws:8000"
        - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
```

(continues on next page)

(continued from previous page)

```
    depends_on:
    - rabbitmq
    - data

reports:
    image: maestroserver/reports-maestro
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "MAESTRO_MONGO_URI=mongodb://mongodb"
    - "MAESTRO_MONGO_DATABASE=maestro-reports"
    depends_on:
    - rabbitmq
    - mongodb

reports_worker:
    image: maestroserver/reports-maestro-celery
    environment:
    - "MAESTRO_REPORT_URI=http://reports:5005"
    - "MAESTRO_DATA_URI=http://data:5010"
    - "MAESTRO_WEBSOCKET_URI=http://ws:8000"
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    depends_on:
    - rabbitmq
    - data

scheduler:
    image: maestroserver/scheduler-maestro
    environment:
    - "MAESTRO_DATA_URI=http://data:5010"
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "MAESTRO_MONGO_URI=mongodb://mongodb"
    - "MAESTRO_MONGO_DATABASE=maestro-client"
    depends_on:
    - mongodb
    - rabbitmq

scheduler_worker:
    image: maestroserver/scheduler-maestro-celery
    environment:
    - "MAESTRO_DATA_URI=http://data:5010"
    - "MAESTRO_DISCOVERY_URI=http://discovery:5000"
    - "MAESTRO_ANALYTICS_URI=http://analytics:5020"
    - "MAESTRO_REPORT_URI=http://reports:5005"
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    depends_on:
    - rabbitmq
    - data

analytics:
    image: maestroserver/analytics-maestro
    ports:
    - "5020:5020"
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "MAESTRO_DATA_URI=http://data:5010"
    depends_on:
    - rabbitmq
```

(continues on next page)

(continued from previous page)

```
      - data

  analytics_worker:
      image: maestroserver/analytics-maestro-celery
      environment:
      - "MAESTRO_DATA_URI=http://data:5010"
      - "MAESTRO_ANALYTICS_FRONT_URI=http://analytics_front:9999"
      - "MAESTRO_WEBSOCKET_URI=http://ws:8000"
      - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
      - "CELERYD_MAX_TASKS_PER_CHILD=2"
      depends_on:
      - rabbitmq
      - data

  analytics_front:
      image: maestroserver/analytics-front-maestro
      ports:
      - "9999:9999"
      volumes:
      - artifacts_analytics:/data/artifacts/
      environment:
      - "MAESTRO_MONGO_URI=mongodb://mongodb"
      - "MAESTRO_MONGO_DATABASE=maestro-client"

  data:
      image: maestroserver/data-maestro
      environment:
      - "MAESTRO_MONGO_URI=mongodb://mongodb"
      - "MAESTRO_MONGO_DATABASE=maestro-client"
      depends_on:
      - mongodb

  audit:
      image: maestroserver/audit-app-maestro
      environment:
      - "MAESTRO_MONGO_URI=mongodb://mongodb"
      - "MAESTRO_MONGO_DATABASE=maestro-audit"
      - "MAESTRO_DATA_URI=http://data:5010"

  ws:
      image: maestroserver/websocket-maestro
      ports:
      - "8000:8000"

  rabbitmq:
      hostname: "discovery-rabbit"
      image: rabbitmq:3-management
      ports:
      - "15672:15672"
      - "5672:5672"

  mongodb:
      image: mongo
      volumes:
      - mongodata:/data/db
      ports:
      - "27017:27017"
```

(continues on next page)

(continued from previous page)

```
    maildev:
        image: djfarrelly/maildev
        mem_limit: 80m
        ports:
        - "1025:25"
        - "1080:80"


volumes:
    mongodata: {}
    artifacts_server: {}
    artifacts_analytics: {}
```

3.1.3 Spin up the API server in a different server

By default the client server uses the same domain name to connect into server api, websocket and analytics front api; However if you like to switch this configuration you can use env vars to set all urls.

By default if you run the client service over `//example.maestro`, the client will try to access the server api by `//example.maestro:8888`, the analytic front by `//example.maestro:9999` and the websocket by `ws(s)//example.maestro:8000`

```
services:
    client:
        image: maestroserver/client-maestro
        environment:
        - "API_URL=http://server.api.endpoint:8888"
        - "STATIC_URL=http://server.api.endpoint:8888/static/" # <- It need to have␣
↪the slash
        - "ANALYTICS_URL=http://analytics.front.endpoint:9999"
        - "WEBSOCKET_URL=ws://websocket.endpoint:8000"
```

3.1.4 Productionize

Should you follow the steps below to run the Maestro on production.

- Using external Database and RabbitMq - More details about external DB.
- Using a reliable store engine as AWS S3 - More details about upload.
- Configuration a third-party SMTP system - More details about SMTP.
- Spin up two or more instance of client, server, discovery, reports, analytics and data. [Expect websocket and scheduler]
- Set a unique value for each `SECRETJWT` key - More details about tokens.
- Use a external loadbalance to handle ssl connections.

3.2 Advanced setups

3.2.1 SMTP Config

Services

- server

You can use an external smtp service as SendGrid, AWS SeS or any smtp server. Go to server application and set:

SMTP_PORT	465	
SMTP_HOST	smtp.gmail.com	
SMTP_SENDER	'maestrosmtp@gmail.com'	
SMTP_USERNAME	'maestrosmtp'	
SMTP_PASSWORD	'XXXX'	
SMTP_USETSL	truelfalse	Enable TLS connect
SMTP_IGNORE	truelfalse	During the connection, validate security connection?

Example

```
services:
    server:
        image: maestroserver/server-maestro
        ports:
        - "8888:8888"
        environment:
        - SMTP_PORT=465
        - SMTP_HOST=smtp.gmail.com
        - SMTP_SENDER='mysender@gmail.com'
        - SMTP_USERNAME=myusername
        - SMTP_PASSWORD=mysecret
        - SMTP_USETSL=true
```

3.2.2 Using external store engine as S3

Services

- server
- analytics_front

You can choose two upload mode, a local file or using S3 storage.

The upload system was used on two points:

server-app	Using on avatar users, teams and projects images.
analytics app	To store artifacts such as graphs, svgs and pngs

Local Storage

For a single node, the file will be stored on a local disk.

Env variables

UPLOAD_TYPE	Local
LOCAL_DIR	/public/static/

```
server:
    image: maestroserver/server-maestro
    environment:
    - UPLOAD_TYPE=Local
    - LOCAL_DIR=/public/static/

client:
    image: maestroserver/client-maestro
    environment:
    - STATIC_URL='http://server-app:8888/static/'
```

Note: These are the default configurations, you don't need to declare these values.

AWS S3 Storage

You can use a S3 Amazon storage object service to store an upload files.

Env variables

UPLOAD_TYPE	S3
AWS_ACCESS_KEY_ID	XXXXXXXXXX
AWS_SECRET_ACCESS_KEY	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
AWS_DEFAULT_REGION	us-east-1
AWS_S3_BUCKET_NAME	maestroserver
AWS_ENDPOINT	S3 endpoint

```
server:
    image: maestroserver/server-maestro
    environment:
    - AWS_ACCESS_KEY_ID='XXXXXXXXXX'
    - AWS_SECRET_ACCESS_KEY='XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
    - AWS_DEFAULT_REGION='us-east-1'
    - AWS_S3_BUCKET_NAME='maestroserver'


client:
    image: maestroserver/client-maestro
    environment:
    - STATIC_URL='https://{my_aws_endpoint}.s3.aws.com.br/{mybucketname}/'
```

Note:

- Remember to set the right path on `STATIC_URL` endpoint into client-app.
- The bucket need to be public.

Digital Ocean Spaces

You can use Digital ocean space, they uses the same S3 protocol, but rather than AWS you need to set `AWS_ENDPOINT`.

Env variables

UPLOAD_TYPE	S3
AWS_ACCESS_KEY_ID	XXXXXXXXXX
AWS_SECRET_ACCESS_KEY	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
AWS_DEFAULT_REGION	ny3
AWS_S3_BUCKET_NAME	maestroserver
AWS_ENDPOINT	S3 endpoint

- Endpoint can be ny3.spacesdigitalocean
- Access and secret can be get on spaces dashboard.
- AWS_DEFAULT_REGION can be ny3

3.2.3 Using external Database

Services

- server
- reports
- scheduler
- analytics_front
- data
- audit

You should spin up a mongodb externally, you can do using the `MAESTRO_MONGO_URI` env variable.

Env Variables	Default	Description
MAESTRO_MONGO_URI	mongodb://localhost:27017	Can be mongodb or mongo+srv://

```
services:
server:
    image: maestroserver/server-maestro
    environment:
    - "MAESTRO_MONGO_URI=mongodb://{external.mongo.url}"
    - "MAESTRO_MONGO_DATABASE=maestro-client"
```

(continues on next page)

(continued from previous page)

```
reports:
    image: maestroserver/reports-maestro
    environment:
    - "MAESTRO_MONGO_URI=mongodb://{external.mongo.url}"
    - "MAESTRO_MONGO_DATABASE=maestro-reports"

scheduler:
    image: maestroserver/scheduler-maestro
    environment:
    - "MAESTRO_MONGO_URI=mongodb://{external.mongo.url}"
    - "MAESTRO_MONGO_DATABASE=maestro-scheduler"

analytics_front:
    image: maestroserver/analytics-front-maestro
    environment:
    - "MAESTRO_MONGO_URI=mongodb://{external.mongo.url}"
    - "MAESTRO_MONGO_DATABASE=maestro-client" # <------ It need to be the same db of␣
↪server-api

data:
    image: maestroserver/data-maestro
    environment:
    - "MAESTRO_MONGO_URI=mongodb://{external.mongo.url}"
    - "MAESTRO_MONGO_DATABASE=maestro-client" # <------ It need to be the same db of␣
↪server-api

audit:
    image: maestroserver/audit-app-maestro
    environment:
    - "MAESTRO_MONGO_URI=mongodb://{external.mongo.url}"
    - "MAESTRO_MONGO_DATABASE=maestro-audit"
```

You can replace the db name using the `MAESTRO_MONGO_DATABASE` env var.

Env Variables	Default	Description
MAESTRO_MONGO_DATABASE	maestro-client	Database name

3.2.4 Using external RabbitMQ

Services

- discovery
- discovery_worker
- reports
- reports_worker
- analytics
- analytics_worker
- scheduler
- scheduler_worker

You can spin up a rabbitmq externally, you can do using the *CELERY_BROKER_URL* env variable.

Env Variables	Default	Description
CELERY_BROKER_URL	amqp://localhost:5672	Amqp endpoint

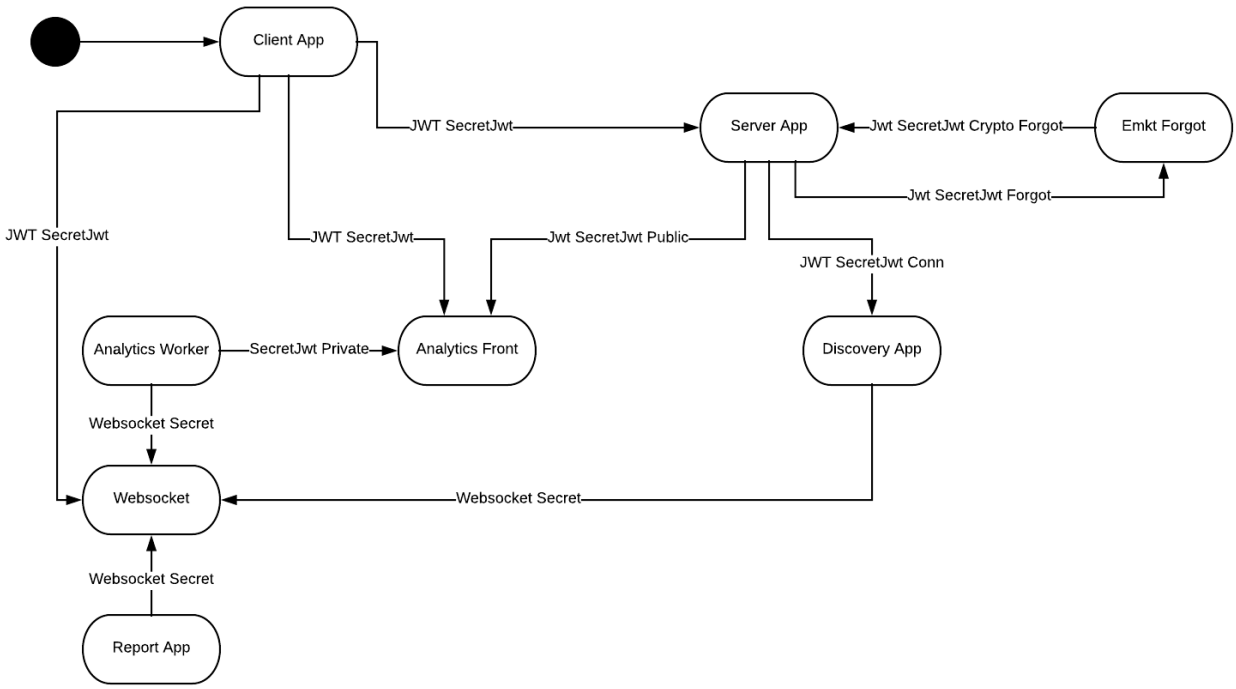
```
services:
discovery:
    image: maestroserver/discovery-maestro
    ports:
    - "5000:5000"
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "MAESTRO_DATA_URI=http://data:5010"
    depends_on:
    - rabbitmq
    - data

discovery_worker:
    image: maestroserver/discovery-maestro-celery
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"

reports:
    image: maestroserver/reports-maestro
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"

reports_worker:
    image: maestroserver/reports-maestro-celery
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"

scheduler:
    image: maestroserver/scheduler-maestro
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"

scheduler_worker:
    image: maestroserver/scheduler-maestro-celery
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"

analytics:
    image: maestroserver/analytics-maestro
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"

analytics_worker:
    image: maestroserver/analytics-maestro-celery
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
```

3.2.5 JWT Tokens

Maestro uses JWT token to handle the authentication/authorization task, those tasks are:

- Authenticate users
- Authenticate private requests between the services
- Authenticate public requests as websockets

High level architecture:

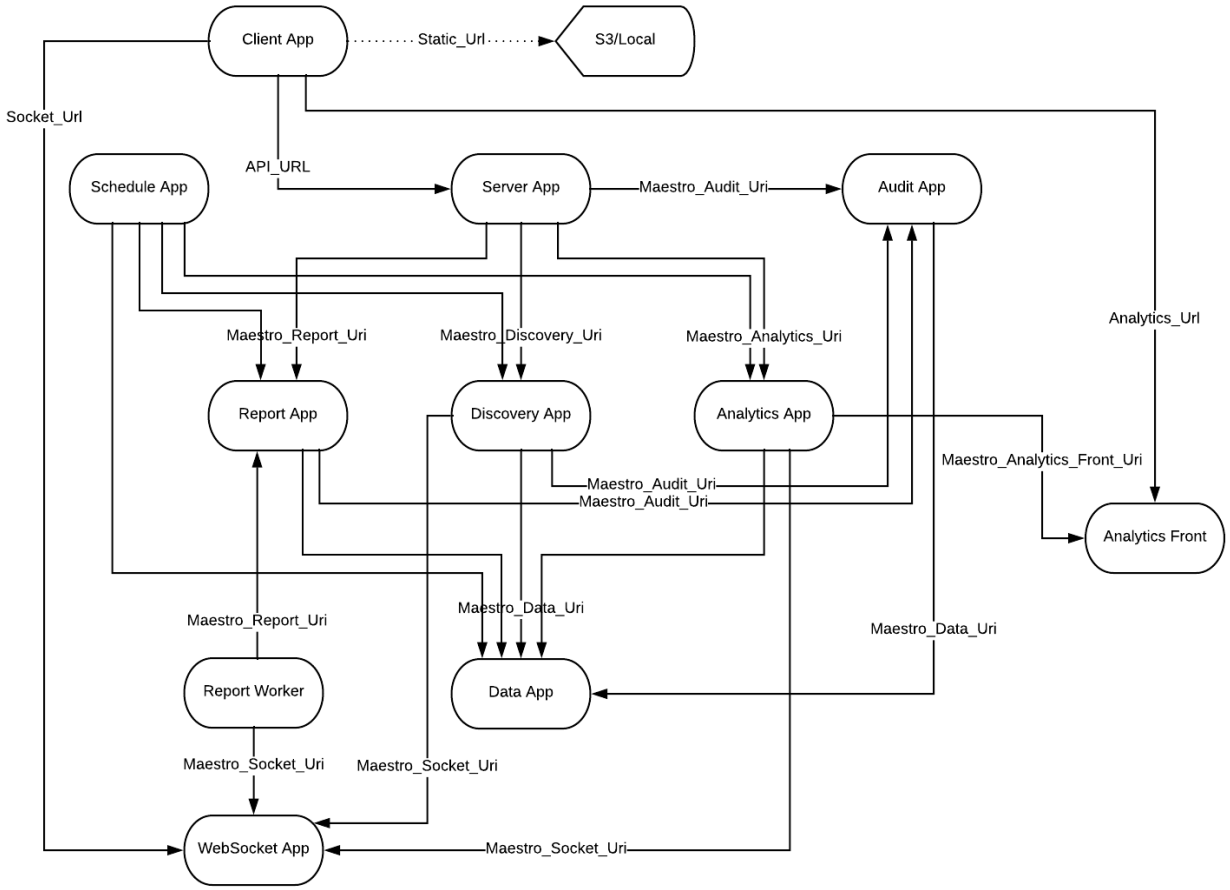


JWT Name	Context	Owned by	Used by	
SecreteJwt	Authenticate/Authorization users	Server App	Client App	Jwt user auth
			Discovery App	To crawler 3 party provider
			Analytics Front	Jwt user auth
			WebSocket	To authorize to connect on websocket
SecretJwt Public	Auth shared links (public access)	Server App	Analytics Front	Used to authorize to access a public graphs
SecretJwt Crpto Forgot	First secret key, request forgot password	Server App	Client App	
SecretJwt Forgot	Second secret key, confirm forgot password	Server App	Server App	
SecretJwt Socket	Authorization users to connect to websocket	Websocket App	Analytics App	To authorize to send a messsage to websocket message bus
			Discovery App	
SecretJwt Private	Private Authenticate	Server	Analytics App	Security key between services
			Discovery App	
			Report App	
		Discovery App	Data App	
			Audit App	
		Reports App	Data App	
			Audit App	
			Report App	Report Worker -> Report Api
		Analytics App	Data App	
		Analytics App (Worker)	Analytics Front	To be able to send artifacts to analytics front

- **Owned** - Token accountant service
- **Context** - High-level description
- **Used** - It was used by

3.2.6 Service Discovery Configuration

This section describes the service discovery configuration. The Maestro server uses env vars to set the configuration between applications, as an example the server-app uses the `MAESTRO_DISCOVERY_URI` to figure out where the discovery app is.



Service	To discovery		Context	Protocol
Client App	Server App	API_URL	SPA application	Rest
	WebSocket App	WEBSOCKET_URL	Received status message (service bus)	WebSocket
	Analytics Front	ANALYTICS_URL	Show graphs on business analytics	Iframe HTTP
Server App	Report App	MAESTRO_REPORT_URI	Create any reports	Rest
	Discovery App	MAESTRO_DISCOVERY_URI	Execute crawler actions	Rest
	Analytics App	MAESTRO_ANALITYCS_URI	Create business graphs	Rest
	Audit App	MAESTRO_AUDIT_URI	Send any update to audit	Rest
Report App	Data App	MAESTRO_DATA_URI	Update report status	Rest
	Audit App	MAESTRO_AUDIT_URI	Send any update to audit	Rest
	WebSocket App	MAESTRO_WEBSOCKET_URI	Send to client any status	WebSocket
Discovery App	Data App	MAESTRO_DATA_URI		Rest
	Audit App	MAESTRO_AUDIT_URI	Send any update to audit	Rest
	WebSocket App	MAESTRO_WEBSOCKET_URI		WebSocket
Analytics App	Data App	MAESTRO_DATA_URI	Populate meta data in analytics entity	Rest
	Analytics Front	MAESTRO_ANALYTICS_FRONT_URI	Post svgs	Rest
	WebSocket App	MAESTRO_WEBSOCKET_URI	Send to client any status	Socket
Scheduler App	Report App	MAESTRO_REPORT_URI	Automated and manage reports	Rest
	Discovery App	MAESTRO_DISCOVERY_URI	Automated and manage discovery	Rest
	Analytics App	MAESTRO_ANALITYCS_URI	Automated and manage analçytics	Rest
	Data App	MAESTRO_DATA_URI	Dump connections parameters.	Rest
Audit App	Data App	MAESTRO_DATA_URI	Update any sync rule	Rest

3.2.7 Themes

Services

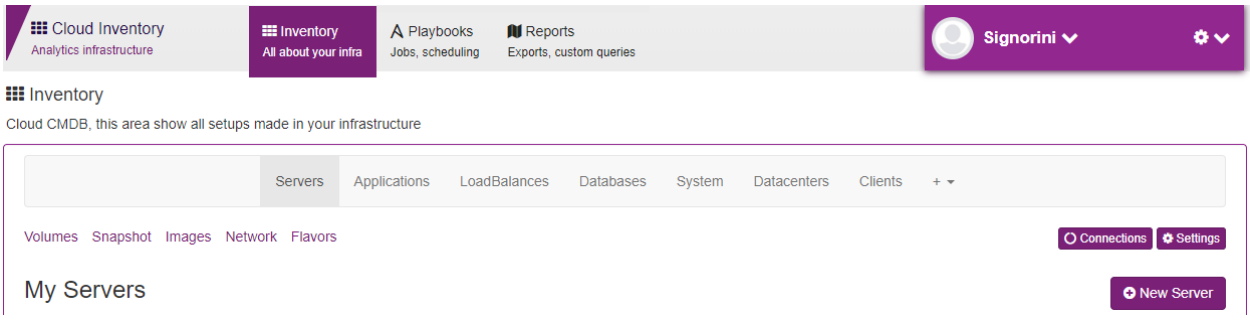
- client

You can change the client theme.

```
client:
    image: maestroserver/client-maestro
    ports:
    - "80:80"
    environment:
    - "API_URL=http://localhost:8888"
    - "THEME=gold"
```

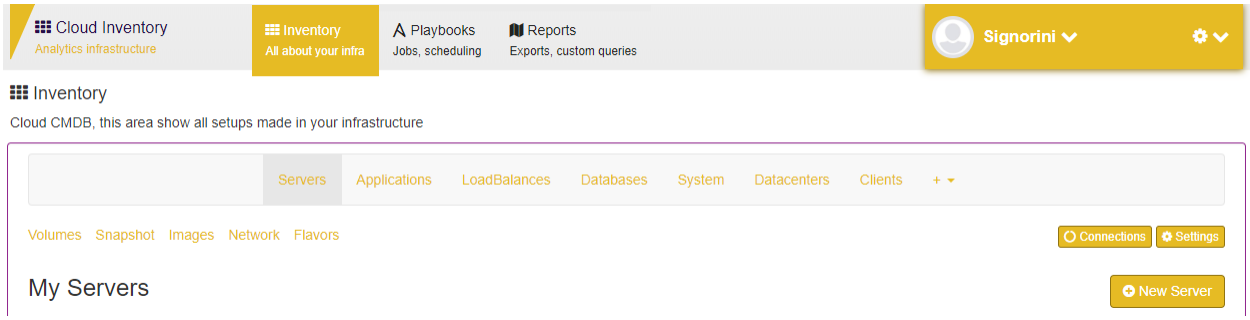
There are some options to choose.

Default



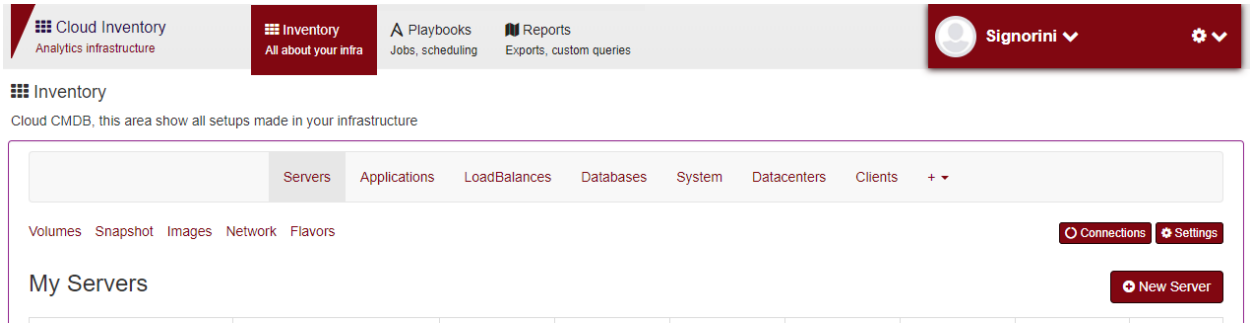
THEME=lotus

Gold



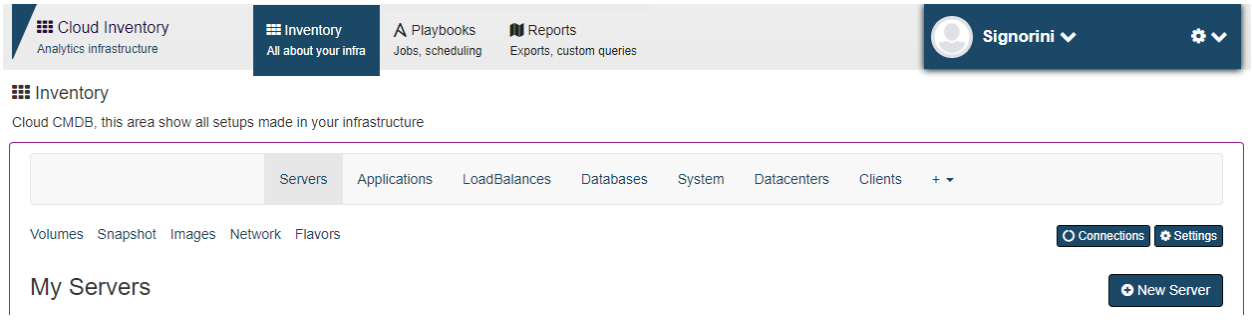
THEME=gold

Wine



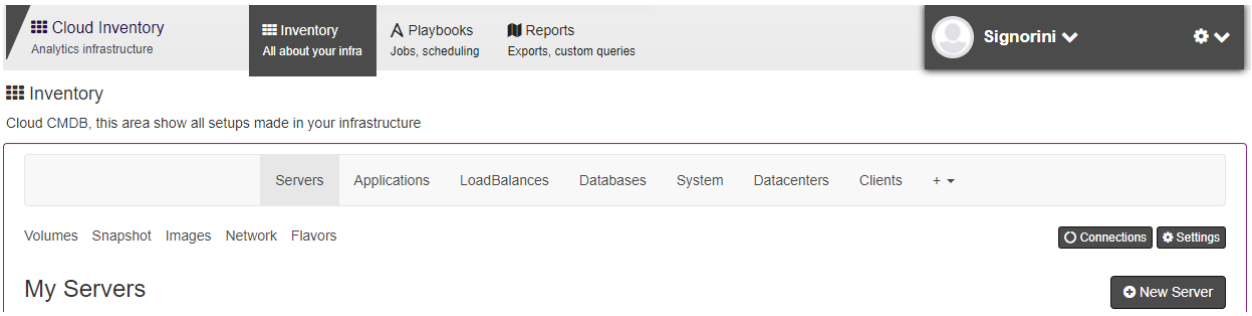
THEME=wine

Blue



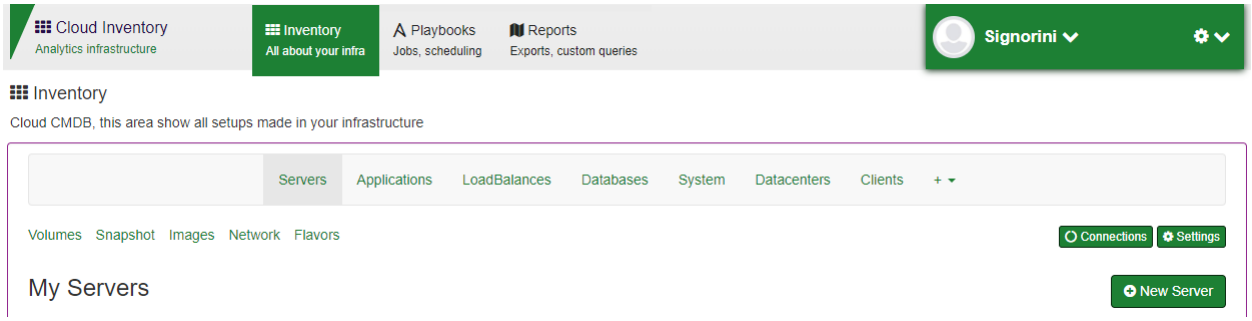
THEME=blue

Dark



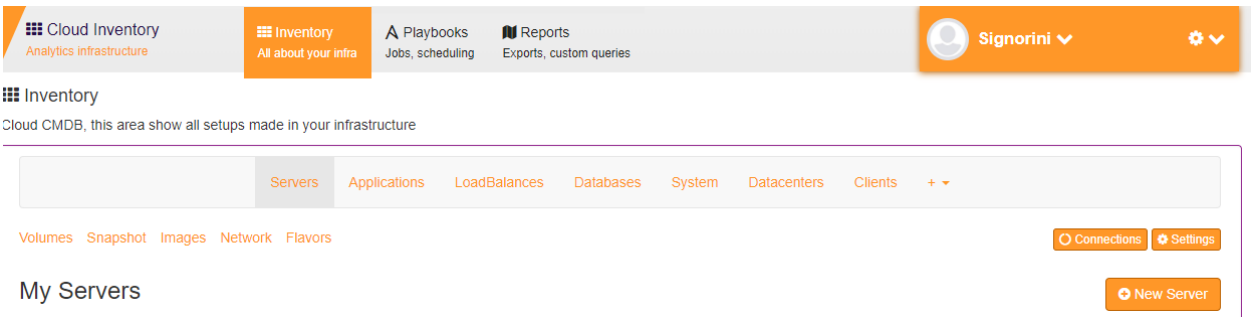
THEME=dark

Green



THEME=green

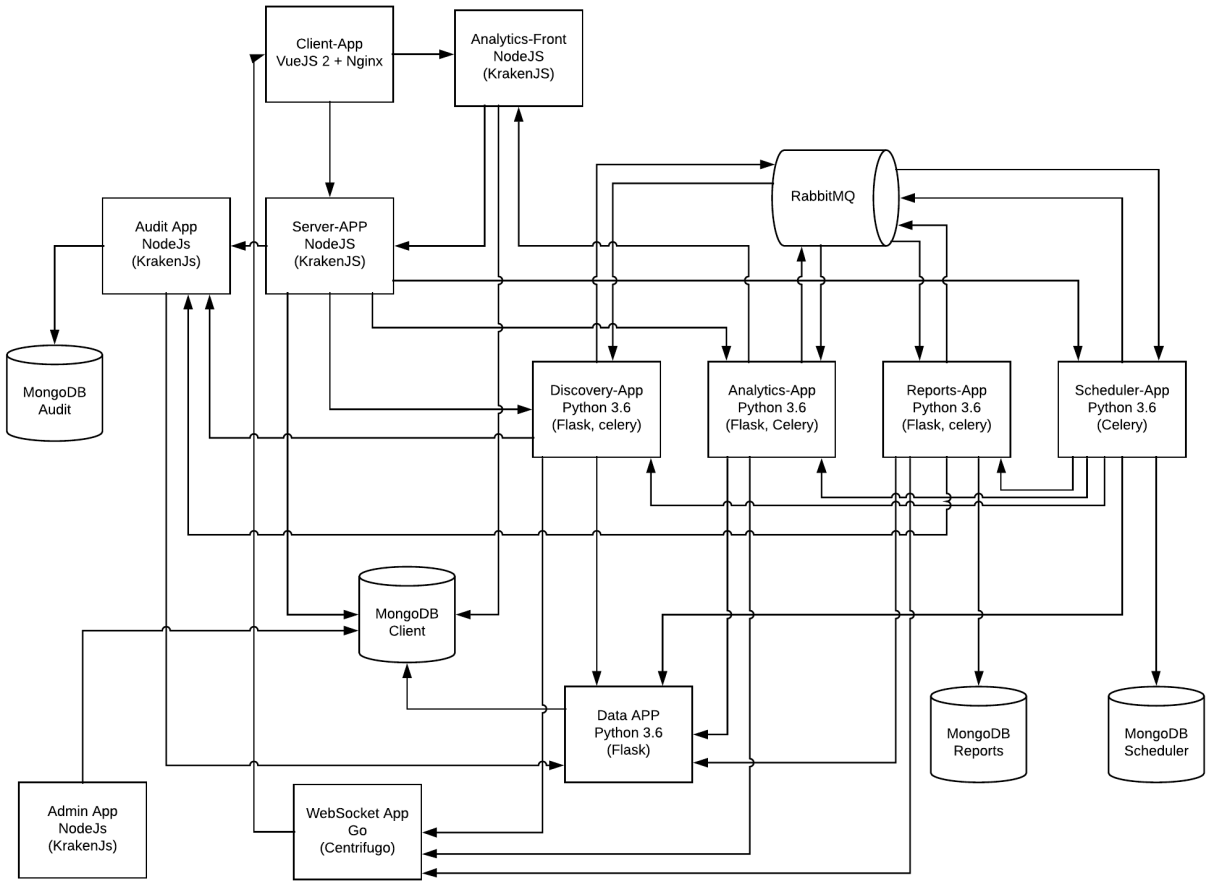
Orange



THEME=orange

3.3 Services configurations

3.3.1 High Architecture



This section will deep dive over each configuration found it on each Maestro service.

A minimum installation require:

- Client App
- Server App
- MongoDB

To uses a synchronous discovery features with AWS and/or other providers, do you need:

- Discovery App
- Data App
- RabbitMq

To have an auto update over discovery/reports/analytics api you need to install the scheduler app.

- Scheduler App

To create and export reports you need to have the reports app installed:

- Reports App
- Data App
- RabbitMq

To create a business analytics graphs, public and shared these maps, you need to install these apps:

- Analytics App
- Analytics Front App
- Data App
- RabbitMq

And if you like to tracking history, you should install:

- Audit App

3.3.2 Client App

Installation by docker-compose

```
client:
    image: maestroserver/client-maestro
    ports:
    - "80:80"
    environment:
    - "API_URL=http://server-app:8888"
    - "STATIC_URL=http://server-app:8888/static/" # ensure to add slash in the end
    - "ANALYTICS_URL=http://localhost:9999"
```

```
docker run -p 80:80
-e 'API_URL=http://localhost:8888'
-e 'STATIC_URL=http://localhost:8888/static/'
-e "ANALYTICS_URL=http://localhost:9999"
maestroserver/client-maestro
```

Warning:

- **API_URL:** Set the endpoint provide by `server-app`.
- **ANALYTICS_URL:** Set the endpoint provide by `analytics-front`.
- **STATIC_URL:** Set the the static url provide by `server-app`. - More details on upload setup.

Env variables

Env Variables	Example	Description
API_URL	http://localhost:8888	Server App Url
STATIC_URL	/static	Full path static files
ANALYTICS_URL	http://localhost:9999	Analytics App Url
WEBSOCKET_URL	ws://localhost:8000	Websocket Url
LOGO	/static/imgs/logo300.png	Logo URL used on login page
THEME	theme-lotus	Theme (gold|wine|blue|green|dark)

3.3.3 Server APP

Installation by docker

```
server:
    image: maestroserver/server-maestro
    ports:
    - "8888:8888"
    environment:
    - "MAESTRO_MONGO_URI=mongodb://mongodb"
    - "MAESTRO_MONGO_DATABASE=maestro-client"
    - "MAESTRO_DISCOVERY_URI=http://discovery:5000"
    - "MAESTRO_ANALYTICS_URI=http://analytics:5020"
    - "MAESTRO_REPORT_URI=http://reports:5005"
    - "MAESTRO_AUDIT_URI=http://audit:10900"
```

```
docker run -p 8888:8888
    -e "MAESTRO_MONGO_URI=mongodb://mongodb"
    -e "MAESTRO_MONGO_DATABASE=maestro-client"
    -e "MAESTRO_DISCOVERY_URI=http://localhost:5000"
    -e "MAESTRO_REPORT_URI=http://localhost:5005"
    -e "MAESTRO_ANALYTICS_URI=http://localhost:5020"
    -e "MAESTRO_AUDIT_URI=http://audit:10900"
    maestroserver/server-maestro
```

Warning:

- **MAESTRO_MONGO_URI:** - It must be the full url -`mongodb://{MAESTRO_MONGO_URI}/{MAESTRO_MONGO_DATABASE}`
- **MAESTRO_MONGO_DATABASE:** - The mongodb database name (ex: maestro-client)
- **SMTP_X:** - It used to send transactional emails - More details about SMTP.
- **MAESTRO_UPLOAD_TYPE:** - Can be a local or S3 - More details about upload.
- **MAESTRO_SECRETJWT_PUBLIC:** - Hash used only do public shared resources, must be different of `MAESTRO_SECRETJWT` - More details about tokens.

Env variables

Env Variables	Example	Description
MAESTRO_PORT	8888	
NODE_ENV	development|production	
MAESTRO_MONGO_URI	mongodb://localhost	DB string connection
MAESTRO_MONGO_DATABASE	maestro-client	Database name
MAESTRO_SECRETJWT	XXXX	Secret key - session
MAESTRO_SECRETJWT_FORGOT	XXXX	Secret key - forgot request
MAESTRO_SECRET_CRYPTO_FORGOT	XXXX	Secret key - forgot content
MAESTRO_SECRETJWT_PUBLIC	XXX	Secret key - public shared
MAESTRO_SECRETJWT_PRIVATE	XXX	Secret Key - JWT private connections
MAESTRO_NOAUTH	XXX	Secret Pass to validate private connections
MAESTRO_DISCOVERY_URL	http://localhost:5000	Url discovery-app (flask)
MAESTRO_REPORT_URL	http://localhost:5005	Url reports-app (flask)
MAESTRO_ANALYTICS_URI	http://localhost:5020	Url Analytics-app (flask)
MAESTRO_AUDIT_URI	http://localhost:10900	Url Audit-app (krakenjs)
MAESTRO_TIMEOUT	1000	Timeout micro service request
SMTP_PORT	1025	
SMTP_HOST	localhost	
SMTP_SENDER	myemail@XXXX	
SMTP_IGNORE	true|false	
SMTP_USETSL	true|false	
SMTP_USERNAME		
SMTP_PASSWORD		
AWS_ACCESS_KEY_ID	XXXX	
AWS_SECRET_ACCESS_KEY	XXXX	
AWS_DEFAULT_REGION	us-east-1	
AWS_S3_BUCKET_NAME	maestroserver	Bucket name
AWS_S3_PRIVATE_BUCKET_NAME	privatebucket	Used to upload internal files, as an example ansible facts ar
MAESTRO_UPLOAD_TYPE	S3 or Local	Upload mode
LOCAL_DIR	/public/static/	Where files will be uploaded
MAESTRO_TMP	$rootDirectory	Tmp folder used on upload files process
MAESTRO_AUDIT_DISABLED	false	Disable the audit services
MAESTRO_REPORT_DISABLED	false	Disable the report services
MAESTRO_DISCOVERY_DISABLED	false	Disable the discovery service

3.3.4 Discovery App

Installation by docker

```
discovery:
    image: maestroserver/discovery-maestro
    ports:
    - "5000:5000"
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "MAESTRO_DATA_URI=http://data:5010"

discovery_worker:
    image: maestroserver/discovery-maestro-celery
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "MAESTRO_DATA_URI=http://data:5010"
    - "MAESTRO_AUDIT_URI=http://audit:10900"
```

```
docker run -p 5000:5000  -e "MAESTRO_DATA_URI=http://localhost:5010" -e "CELERY_
↪BROKER_URL=amqp://rabbitmq:5672" maestroserver/discovery-maestro

docker run \
    -e "MAESTRO_DATA_URI=http://localhost:5010" \
    -e "CELERY_BROKER_URL=amqp://rabbitmq:5672" \
    -e "MAESTRO_AUDIT_URI=http://localhost:10900" \
    -e "MAESTRO_SERVER_URI=http://localhost:8888" \
    maestroserver/discovery-maestro-celery
```

Warning:

- **MAESTRO_DATA_URI:** - Data App enpoint API - default port is 5000
- **MAESTRO_AUDIT_URI:** - Audit App endpoint API - default port is 10900
- **MAESTRO_WEBSOCKET_URI:** - Websocket endpoint, this one is HTTP
- **MAESTRO_SERVER_URI** - Server endpoint

Env variables

Env Variables	Example	Description
MAESTRO_PORT	5000	Port used
MAESTRO_DATA_URI	http://localhost:5010	Data Layer API URL
MAESTRO_AUDIT_URI	http://localhost:10900	Audit App - API URL
MAESTRO_WEBSOCKET_URI	http://localhost:8000	Webosocket App - API URL
MAESTRO_SERVER_URI	http://localhost:8888	Server App - API URL
MAESTRO_SECRETJWT	XXX	Same that Server App
MAESTRO_SECRETJWT_PRIVATE	XXX	Secret Key - JWT private connections
MAESTRO_NOAUTH	XXX	Secret Pass to validate private connections
MAESTRO_WEBSOCKET_SECRET	XXX	Secret Key - JWT Websocket connections
MAESTRO_TRANSLATE_QTD	200	Prefetch translation process
MAESTRO_GWORKERS	2	Gunicorn multi process
CELERY_BROKER_URL	amqp://rabbitmq:5672	RabbitMQ connection
CELERYD_TASK_TIME_LIMIT	10	Timeout workers

3.3.5 Reports App

Installation by docker

```
reports:
    image: maestroserver/reports-maestro
    ports:
    - "5005:5005"
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "MAESTRO_MONGO_URI=mongodb://mongodb"
    - "MAESTRO_MONGO_DATABASE=maestro-reports"

reports_worker:
    image: maestroserver/reports-maestro-celery
    environment:
```

(continues on next page)

(continued from previous page)

```
    - "MAESTRO_REPORT_URI=http://reports:5005"
    - "MAESTRO_DATA_URI=http://data:5010"
    - "MAESTRO_AUDIT_URI=http://audit:10900"
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
```

Warning:

- **MAESTRO_REPORT_URI:** - Reports enpoint API - default port is 5005, It used by reports workers
- **MAESTRO_DATA_URI:** - Data enpoint API - default port is 5000
- **MAESTRO_AUDIT_URI:** - Audit Endpoint API - default port is 10900
- **MAESTRO_WEBSOCKET_URI:** - Websocket endpoint, this one is HTTP

```
docker run -p 5005 -e "MAESTRO_DATA_URI=http://localhost:5010" -e "CELERY_BROKER_
↪URL=amqp://rabbitmq:5672" -e 'MAESTRO_MONGO_URI=localhost' maestroserver/reports-
↪maestro

docker run \
    -e "MAESTRO_DATA_URI=http://localhost:5010" \
    -e "MAESTRO_REPORT_URI=http://localhost:5005" \
    -e "CELERY_BROKER_URL=amqp://rabbitmq:5672" \
    -e "MAESTRO_AUDIT_URI=http://audit:10900" \
    maestroserver/reports-maestro-celery
```

Env variables

Env Variables	Example	Description
MAESTRO_PORT	5005	Port used
MAESTRO_MONGO_URI	localhost	Mongo Url conn
MAESTRO_MONGO_DATABASE	maestro-reports	Db name, its differente of servers-app
MAESTRO_DATA_URI	http://localhost:5010	Data layer api
MAESTRO_REPORT_URI	http://localhost:5005	Report api
MAESTRO_AUDIT_URI	http://localhost:10900	Audit App - API URL
MAESTRO_WEBSOCKET_URI	http://localhost:8000	Webosocket App - API URL
MAESTRO_SECRETJWT_PRIVATE	XXX	Secret Key - JWT private connections
MAESTRO_NOAUTH	XXX	Secret Pass to validate private connections
MAESTRO_WEBSOCKET_SECRET	XXX	Secret Key - JWT Websocket connections
MAESTRO_REPORT_RESULT_QTD	1500	Limit default
MAESTRO_INSERT_QTD	20	Prefetch data insert
MAESTRO_GWORKERS	2	Gworkers thread pool
CELERY_BROKER_URL	amqp://rabbitmq:5672	RabbitMQ connection

3.3.6 Analytics App

Installation by docker

```
analytics:
    image: maestroserver/analytics-maestro
    ports:
    - "5020:5020"
```

(continues on next page)

(continued from previous page)

```
    environment:
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "MAESTRO_DATA_URI=http://data:5010"

analytics_worker:
    image: maestroserver/analytics-maestro-celery
    environment:
    - "MAESTRO_DATA_URI=http://data:5010"
    - "MAESTRO_ANALYTICS_FRONT_URI=http://analytics_front:9999"
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "CELERYD_MAX_TASKS_PER_CHILD=2"
```

Warning:

- **MAESTRO_ANALYTICS_FRONT_URI:** - Analytics Front enpoint API - default port is 9999
- **MAESTRO_DATA_URI:** - Data enpoint API - default port is 5000
- **MAESTRO_WEBSOCKET_URI:** - Websocket endpoint, this one is HTTP

```
docker run -p 5020
    -e "MAESTRO_DATA_URI=http://localhost:5010"
    -e "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    -e 'MAESTRO_MONGO_URI=localhost'
    maestroserver/analytics-maestro

docker run
    -e "MAESTRO_DATA_URI=http://localhost:5010"
    -e "MAESTRO_ANALYTICS_FRONT_URI=http://localhost:9999"
    -e "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    maestroserver/analytics-maestro-celery
```

Env variables

Env Variables	Example	Description
MAESTRO_PORT	5020	Port
MAESTRO_DATA_URI	http://localhost:5010	Data Layer API URL
MAESTRO_ANALYTICS_FRONT_URI	http://localhost:9999	Analytics Front URL
MAESTRO_WEBSOCKET_URI	http://localhost:8000	Webosocket App - API URL
MAESTRO_SECRETJWT_PRIVATE	XXX	Secret Key - JWT private connections
MAESTRO_NOAUTH	XXX	Secret Pass to validate private connections
MAESTRO_WEBSOCKET_SECRET	XXX	Secret Key - JWT Websocket connections
MAESTRO_GWORKERS	2	Gunicorn multi process
CELERY_BROKER_URL	amqp://rabbitmq:5672	RabbitMQ connection
CELERYD_TASK_TIME_LIMIT	10	Timeout workers

3.3.7 Analytics Front

Installation by docker

```
reports:
    image: maestroserver/analytics-front-maestro
```

(continues on next page)

(continued from previous page)

```
    ports:
    - "9999:9999"
    environment:
    - "MAESTRO_MONGO_URI=mongodb://mongodb"
    - "MAESTRO_MONGO_DATABASE=maestro-client"
```

Warning:

- **MAESTRO_REPORT_URI:** - Reports enpoint API - default port is 5005
- **MAESTRO_DATA_URI:** - Data enpoint API - default port is 5000
- **MAESTRO_WEBSOCKET_URI:** - Websocket endpoint, this one is HTTP

```
docker run -p 5005
    -e "MAESTRO_MONGO_URI=mongodb://mongodb"
    -e "MAESTRO_MONGO_DATABASE=maestro-client"
    maestroserver/analytics-front-maestro
```

Env variables

Env Variables	Example	Description
MAESTRO_PORT	9999	
API_URL	http://localhost:8888	Server app Url
NODE_ENV	development|production	
MAESTRO_MONGO_URI	localhost	DB string connection
MAESTRO_MONGO_DATABASE	maestro-client	Database name
MAESTRO_SECRETJWT	XXXX	Secret key - server app
MAESTRO_SECRETJWT_PRIVATE	XXX	Secret Key - JWT private connections
MAESTRO_NOAUTH	XXX	Secret Pass to validate private connections
MAESTRO_SECRETJWT_PUBLIC	XXXX	Secret key - same as on server app
AWS_ACCESS_KEY_ID	XXXX	
AWS_SECRET_ACCESS_KEY	XXXX	
AWS_DEFAULT_REGION	us-east-1	
AWS_S3_BUCKET_NAME	maestroserver	
MAESTRO_UPLOAD_TYPE	S3/Local	Upload mode
LOCAL_DIR	/public/static/	Where files will be uploaded
PWD	$rootDirectory	PWD process

3.3.8 Data App

Installation by docker

```
data:
    image: maestroserver/data-maestro
    ports:
    - "5010:5010"
    environment:
        - "MAESTRO_MONGO_URI=mongodb://mongodb"
        - "MAESTRO_MONGO_DATABASE=maestro-client"
```

```
docker run -p 5010 -e "MAESTRO_MONGO_URI=mongodb://mongodb" -e "MAESTRO_MONGO_
↪DATABASE=maestro-client" maestroserver/data-maestro
```

Env variables

Env Variables	Example	Description
MAESTRO_PORT	5010	Port used
MAESTRO_MONGO_URI	localhost	Mongo Url conn
MAESTRO_MONGO_DATABASE	maestro-client	Db name, its differente of servers-app
MAESTRO_GWORKERS	2	Gunicorn multi process
MAESTRO_INSERT_QTD	200	Throughput insert used on reports collection
MAESTRO_SECRETJWT_PRIVATE	XXX	Secret Key - JWT private connections
MAESTRO_NOAUTH	XXX	Secret Pass to validate private connections

3.3.9 Scheduler App

Installation by docker

```
scheduler:
    image: maestroserver/scheduler-maestro
    environment:
    - "MAESTRO_DATA_URI=http://data:5010"
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "MAESTRO_MONGO_URI=mongodb://mongodb"
    - "MAESTRO_MONGO_DATABASE=maestro-client"

scheduler_worker:
    image: maestroserver/scheduler-maestro-celery
    environment:
    - "MAESTRO_DATA_URI=http://data:5010"
    - "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    - "MAESTRO_DISCOVERY_URI=http://discovery:5000"
    - "MAESTRO_ANALYTICS_URI=http://analytics:5020"
    - "MAESTRO_REPORT_URI=http://reports:5005"
```

```
docker run
    -e "MAESTRO_DATA_URI=http://localhost:5010"
    -e "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    maestroserver/scheduler-maestro

docker run
    -e "MAESTRO_DATA_URI=http://localhost:5010"
    -e "MAESTRO_DISCOVERY_URI=http://localhost:5000"
    -e "MAESTRO_ANALYTICS_URI=http://localhost:5020"
    -e "MAESTRO_REPORT_URI=http://localhost:5005"
    -e "CELERY_BROKER_URL=amqp://rabbitmq:5672"
    maestroserver/scheduler-maestro-celery
```

Warning:

- **MAESTRO_DATA_URI:** - Data API - default port is 5000

Danger:

- You can only spin up an one schedule instance, if you do it will have a duplicate job execution.

Env variables

Env Variables	Example	Description
MAESTRO_DATA_URI	http://localhost:5010	Data Layer API URL
MAESTRO_DISCOVERY_URI	http://localhost:5000	Discovery App URL
MAESTRO_ANALYTICS_URI	http://localhost:5020	Analytics App URL
MAESTRO_REPORT_URI	http://localhost:5005	Reports App URL
MAESTRO_MONGO_URI	localhost	MongoDB URI
MAESTRO_MONGO_DATABASE	maestro-client	Mongo Database name
CELERY_BROKER_URL	amqp://rabbitmq:5672	RabbitMQ connection
MAESTRO_SECRETJWT_PRIVATE	XXX	Secret Key - JWT private connections
MAESTRO_NOAUTH	XXX	Secret Pass to validate private connections

3.3.10 Audit App

Installation by docker

```
audit:
    image: maestroserver/audit-app-maestro
    ports:
    - "10900:10900"
    environment:
    - "MAESTRO_MONGO_URI=mongodb://mongodb"
    - "MAESTRO_MONGO_DATABASE=maestro-audit"
    - "MAESTRO_DATA_URI=http://data:5010"
```

Warning:

- **MAESTRO_DATA_URI:** - Data API - default port is 5000

```
docker run -p 10900
    -e "MAESTRO_MONGO_URI=mongodb://mongodb"
    -e "MAESTRO_MONGO_DATABASE=maestro-audit"
    maestroserver/audit-app-maestro
```

Env variables

Env Variables	Example	Description
MAESTRO_PORT	10900	
NODE_ENV	development|production	
MAESTRO_MONGO_URI	localhost	DB string connection
MAESTRO_MONGO_DATABASE	maestro-audit	Database name
MAESTRO_TIMEOUT	1000	Timeout any http private request
MAESTRO_DATA_URI	http://localhost:5010	Data App - API URL
MAESTRO_SECRETJWT_PRIVATE	XXX	Secret Key - JWT private connections
MAESTRO_NOAUTH	XXX	Secret Pass to validate private connections

3.3.11 WebSocket App

Installation by docker

```
data:
    image: maestroserver/websocket-maestro
    ports:
    - "8000:8000"
```

```
docker run -p 8000:800 maestroserver/websocket-maestro
```

Env variables

Env Variables	Example	Description
MAESTRO_WEBSOCKET_SECRET	backSecretToken	Token to authenticate backends apps
MAESTRO_SECRETJWT	frontSecretToken	Token to autheticate front end users
CENTRIFUGO_ADMIN	adminPassword	Admin password
CENTRIFUGO_ADMIN_SECRET	adminSecretToken	Token to autheticate administrator users
CENTRIFUGO_TLSAUTO	true	Auto SSL using Let Encrypt
CENTRIFUGO_TLSAUTO_HTTP	true	Auto SSL using AcmeV1 Let Encrypt
CENTRIFUGO_TLS_PORT	:80	Can be used to set address for handling http_01 ACME challenge, default value is :80
CENTRIFUGO_TLS	true	Using dev ssl certs to run custom certs
CENTRIFUGO_TLS_KEY	/tmp/certs/server.key	Full path ssl key (Expose by folder bind on docker)
CENTRIFUGO_TLS_CERT	/tmp/certs/server.key	Full path ssl certs

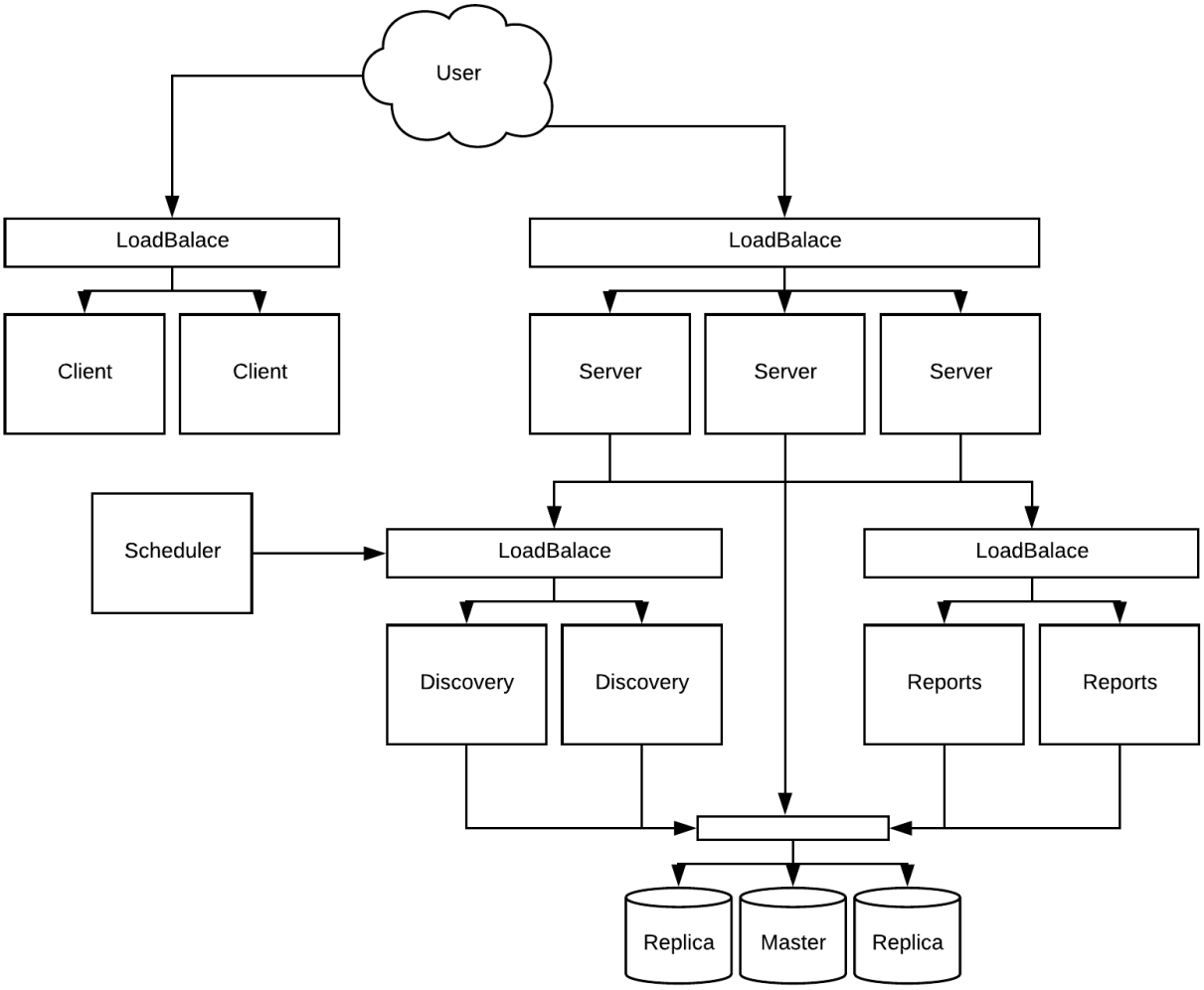
3.4 High availability

3.4.1 12 Factory and Horizontal Scaling

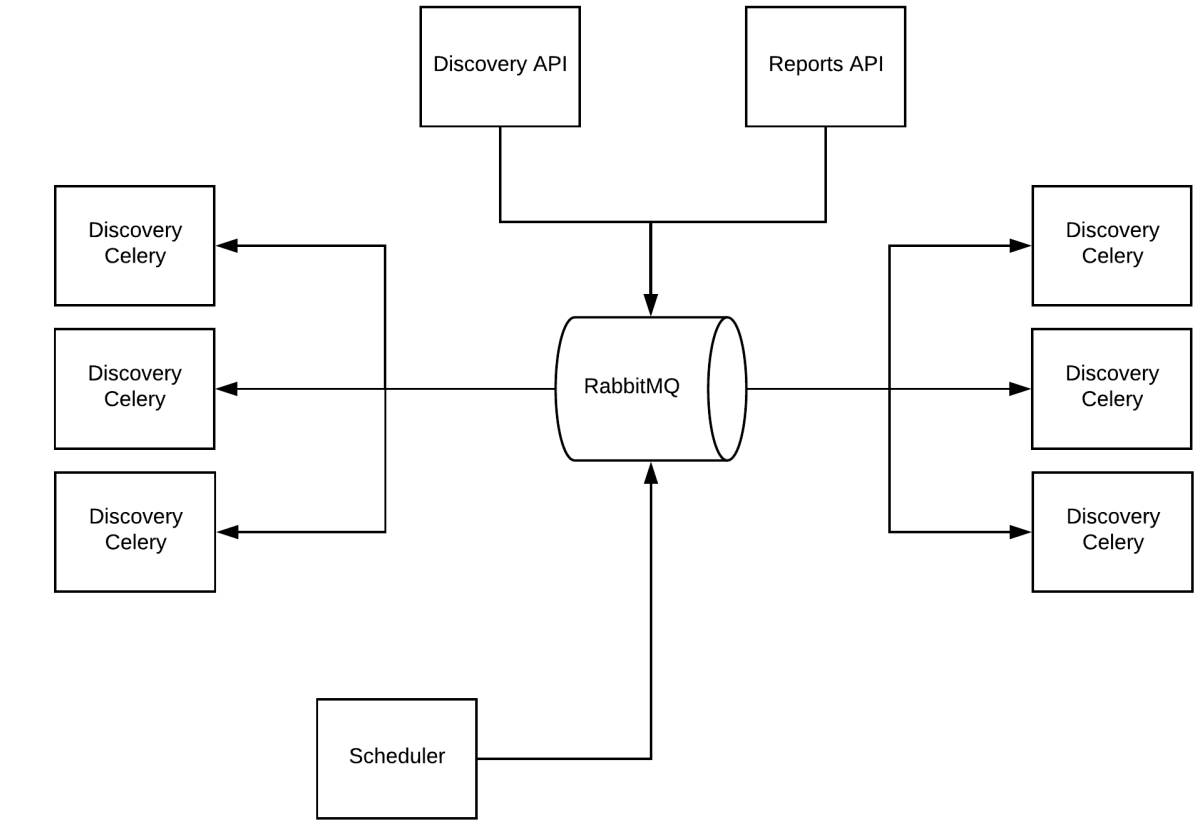
This section describes some tips you can use to be able to productionize the Maestro.

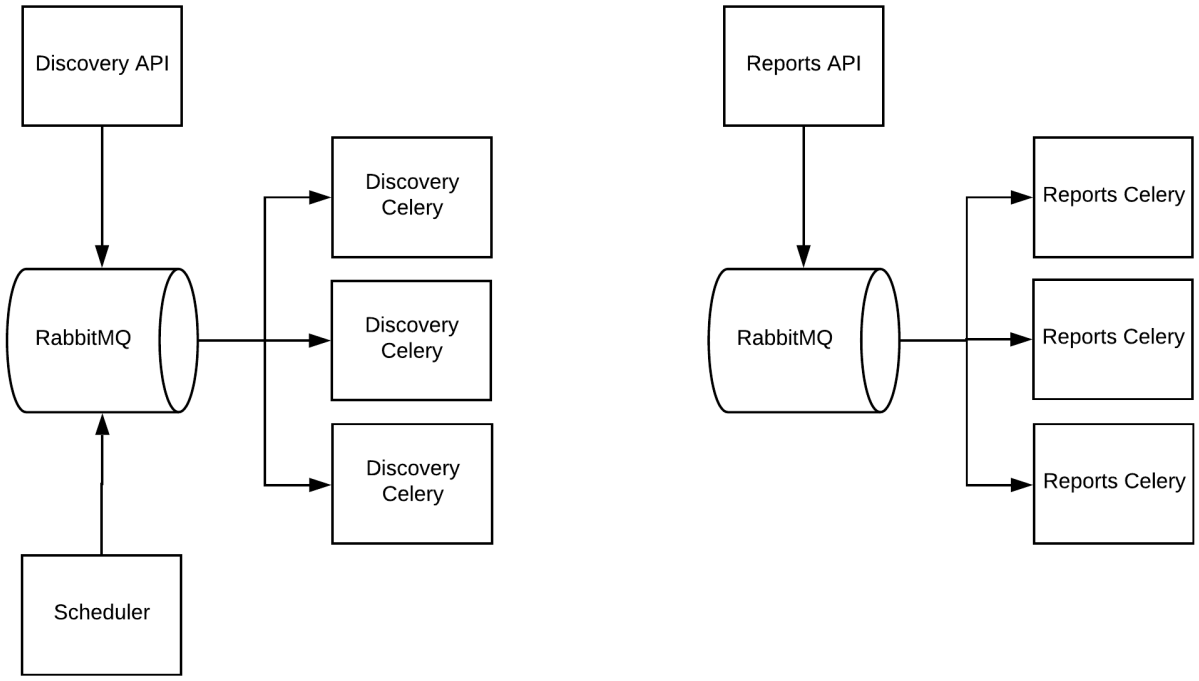
- The first and most important is to avoid to use any local configuration as a local upload file system, local mongodb and rabbitmq.
  - You should use a reliable storage engine as S3 - More details about upload.
  - You can use atlas mongodb to manage your mongo db externally. - More details about external DB.
  - Configuration a third-party SMTP - More details about SMTP.
  - Set a unique value for each `SECRETJWT` key - More details about tokens.
- Spin up an nginx/loadbalance over any public endpoint to handle ssl configuration.
- Discovery, reports and analytics services are compound by two parts, one it's the api, and the other is the workers, you don't need to deploy it on the same server.

Follow a single example,



It's possible to improve the reliability over discovery and reports services.


Discovery API
Reports API
Discovery Celery
Discovery Celery
Discovery Celery
RabbitMQ
Discovery Celery
Discovery Celery
Discovery Celery
Scheduler


Discovery API
Discovery Celery
RabbitMQ
Discovery Celery
Discovery Celery
Scheduler
Reports API
Reports Celery
RabbitMQ
Reports Celery
Reports Celery

3.4.2 Scheduler Beat App

Danger: Scheduler app have two parts, the producer called beat and the workers, the beat isn't able to have multiple instance on the same time, be careful. To minimize the drawback, the beat schedule is an isolated and an stateless service (if fall, you can call up the beat again).

3.4.3 HealthChecks

You can you the / path to do the healthchecks.

- Front end, show in right-footer.
- http://{server-api}:8888/
- http://{discovery-api}:5000/
- http://{reports-api}:5005/
- http://{analytics-maestro}:5020/
- http://{analytics-front}:9999/
- http://{audit}:10900/

3.4.4 Running on Kubernetes

To run Maestro over kubernetes, you can uses those deployment files found it on k8s deployments,

Creating secrets files

The first step it will be to create those secrets.

- mongo_srv.txt
- smtp.txt
- storage.txt

And populate accordlingly. Running these commands.

```
kubectl create secret generic smtp --from-env-file secrets/smtp.txt
kubectl create secret generic mongo_srv --from-env-file secrets/mongo_srv.txt
kubectl create secret generic storage --from-env-file secrets/storage.txt
```

storage.txt

```
AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=
AWS_S3_BUCKET_NAME=
```

mongo_srv.txt

```
MAESTRO_MONGO_URI=mongo+srv://mongodb:27017
```

smtp.txt

```
SMTP_PORT=
SMTP_HOST=
SMTP_SENDER=
SMTP_USERNAME=
SMTP_PASSWORD=
SMTP_USETSL=
```

To check if everything it's ok, you can run:

```
> kubectl get secrets

NAME                  TYPE                                  DATA      AGE
mongosrv              Opaque                                1         24d
smtp                  Opaque                                6         18d
storage               Opaque                                4         17d
```

Deploying services

```
source run.sh
```

And

Create the third-party services.

```
kubectl apply -f mongo/
kubectl apply -f rabbitmq/
kubectl apply -f maildev/
```

Deploying the Maestro bundle services

```
kubectl apply -f maestro-websocket/
kubectl apply -f maestro-data/
kubectl apply -f maestro-discovery/
kubectl apply -f maestro-reports/
kubectl apply -f maestro-analytics/
kubectl apply -f maestro-analytics-front/
kubectl apply -f maestro-audit/
kubectl apply -f maestro-scheduler/
kubectl apply -f maestro-server/
kubectl apply -f maestro-client/
```

Checking deployments

```
> kubectl get deployments

NAME                        DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
maestro-client              1         1         1            1           6d
maestro-analytics           1         1         1            1           6d
maestro-analytics-front     1         1         1            1           6d
maestro-analytics-worker    1         1         1            1           6d
maestro-audit               1         1         1            1           6d
maestro-data                1         1         1            1           24d
maestro-discovery           1         1         1            1           6d
maestro-discovery-worker    1         1         1            1           6d
```

(continues on next page)

(continued from previous page)

```
maestro-reports            1         1         1         1         6d
maestro-reports-worker     1         1         1         1         6d
maestro-scheduler          1         1         1         1         6d
maestro-scheduler-worker   1         1         1         1         6d
maestro-server             2         2         2         2         6d
maestro-websocket          1         1         1         1         6d
rabbitmq                   1         1         1         1         24d
```

Checking exposed services

```
> kubectl get svc

NAME                       TYPE           CLUSTER-IP      EXTERNAL-IP       PORT(S)
↪                    AGE
external-analytics-front   LoadBalancer   10.XX.252.63    XX.XX.XX.XX       9999:30859/
↪TCP                 23d
external-server            LoadBalancer   10.XX.245.248   XX.XX.XX.XX       8888:31254/
↪TCP                 23d
external-client            LoadBalancer   10.XX.245.248   XX.XX.XX.XX       80:31254/
↪TCP                   23d
external-websocket         LoadBalancer   10.XX.253.161   XX.XX.XX.XX       8443:30705/
↪TCP,80:31146/TCP    21d

internal-analytics         ClusterIP      10.XX.240.129   <none>            5020/TCP
↪                    6d
internal-analytics-front   ClusterIP      10.XX.243.157   <none>            9999/TCP
↪                    23d
internal-audit             ClusterIP      10.XX.243.250   <none>            10900/TCP
↪                    6d
internal-data              ClusterIP      10.XX.244.111   <none>            5010/TCP
↪                    24d
internal-discovery         ClusterIP      10.XX.240.202   <none>            5000/TCP
↪                    6d
internal-rabbit            ClusterIP      10.XX.243.117   <none>            5672/TCP,
↪15672/TCP             24d
internal-reports           ClusterIP      10.XX.241.218   <none>            5005/TCP
↪                    6d
internal-websocket         ClusterIP      10.XX.241.159   <none>            8000/TCP
↪                    21d
```

Note: It must have 4 public endpoint, the client service, server app, analytics front and websocket system.

CHAPTER 4

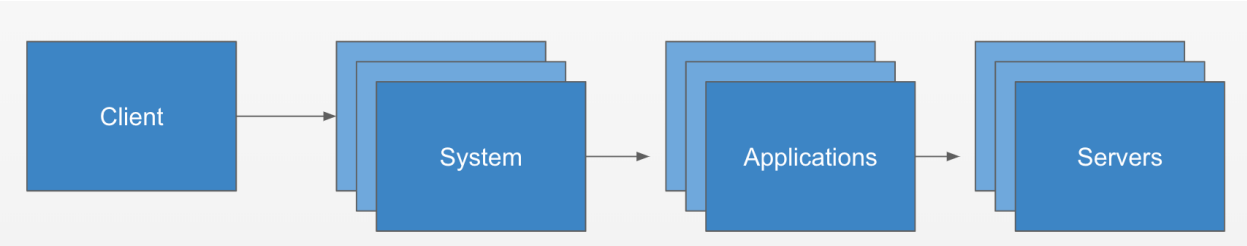
User Guide

In this section we will cover how the maestro server works from the user's point of view, if you want to install and configure the Maestro server you should go to the installation section, if you would like to develop a new functionality or a new service, you should go to the developer section.

Maestro is an inventory system for multi platform environments, multi-cloud for enterprise companies. It aim to organize in a single dashboard with relation between servers, applications, systems and clients.

The dashboard was divided into three parts:

- **Cloud inventory:** The first part you will figure out the whole inventory, such as servers, applications and systems as well as the relationship between them. In this area you can also connect third-party providers to self-discover and self-update.
- **Analytics:** In the second part you can view the relationships between applications, systems architecture, a map of dependencies and can even share these information in third-party applications as Confluence, GitHub and more.
- **Reports:** In this area you can generate advanced reports such as the list of servers for a given client.



4.1 Cloud Inventory

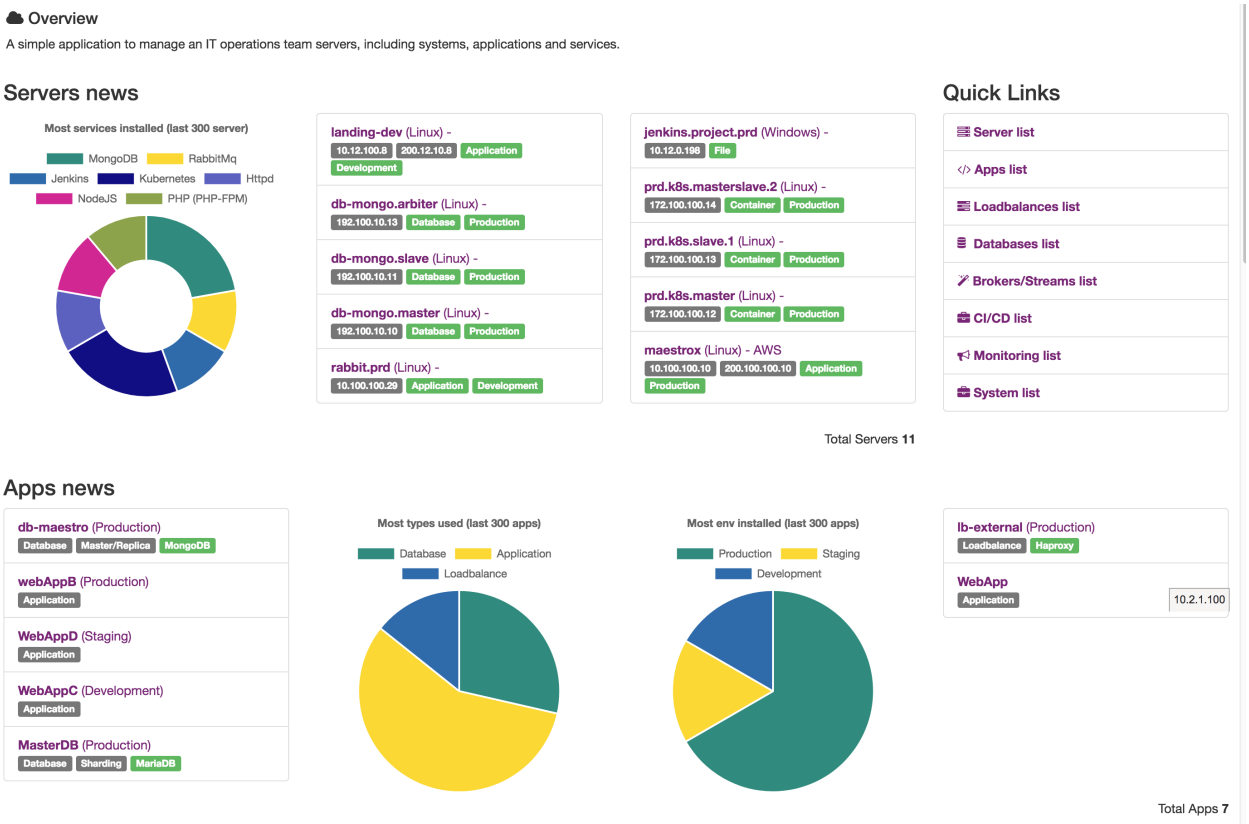
We can use to organize each part of our architecture by:

4.1.1 Inventory

You can organize your servers, applications, cloud resources, systems, and clients on a single and powerful dashboard.

You will be able to:

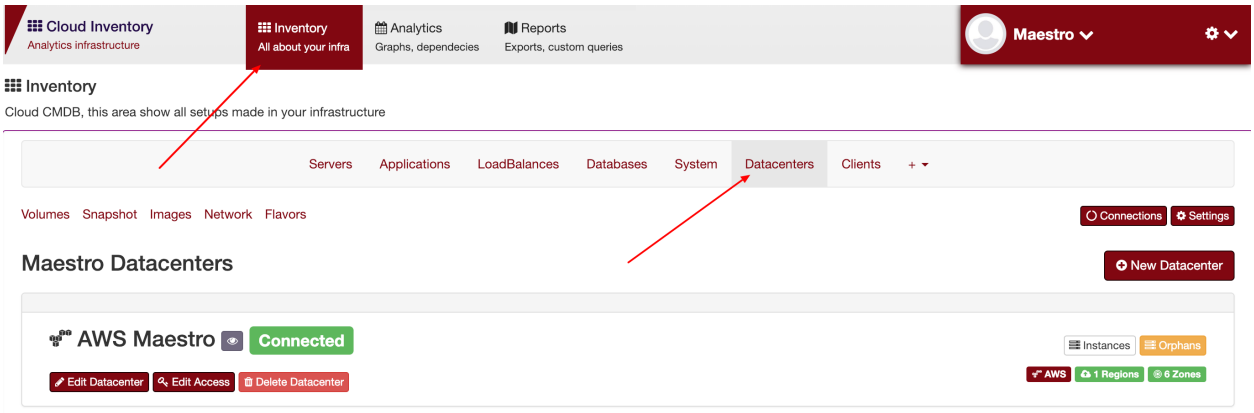
- Control multi-environment, multi-cloud and multi-regions using a single dashboard.
- Track an application ownership
- Easy to visualize a relationship between microservices
- Correlation between teams/systems
- Track costs
- Easy way to do documents of high architecture systems



Datacenters

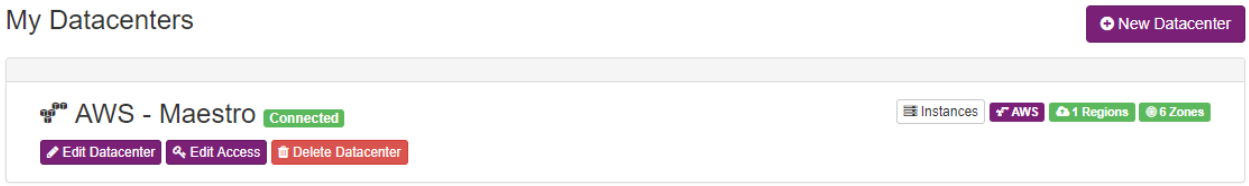
Inventory > Datacenter

A datacenter, can be a building, dedicated space within a building, or a group of buildings used to house computer systems and associated components, can be a cloud account, a space reserved to execute resources provide by third-party company.

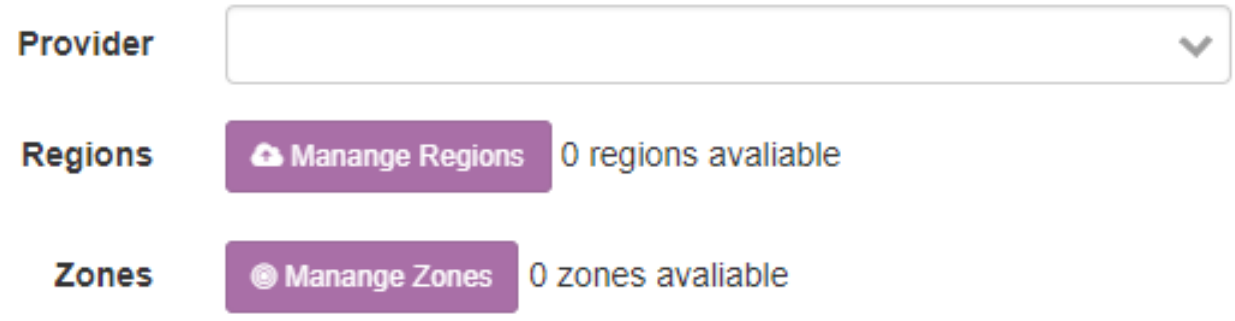


You should insert any type of datacenters can be a cloud third-party datacenter, a specific space or a group of bare metal servers.

Field	Description
Name	Datacenter name
Provider	The third-party provider, or create a new one
Regions	Selecting a region/s
Zones	Selecting a zone/s

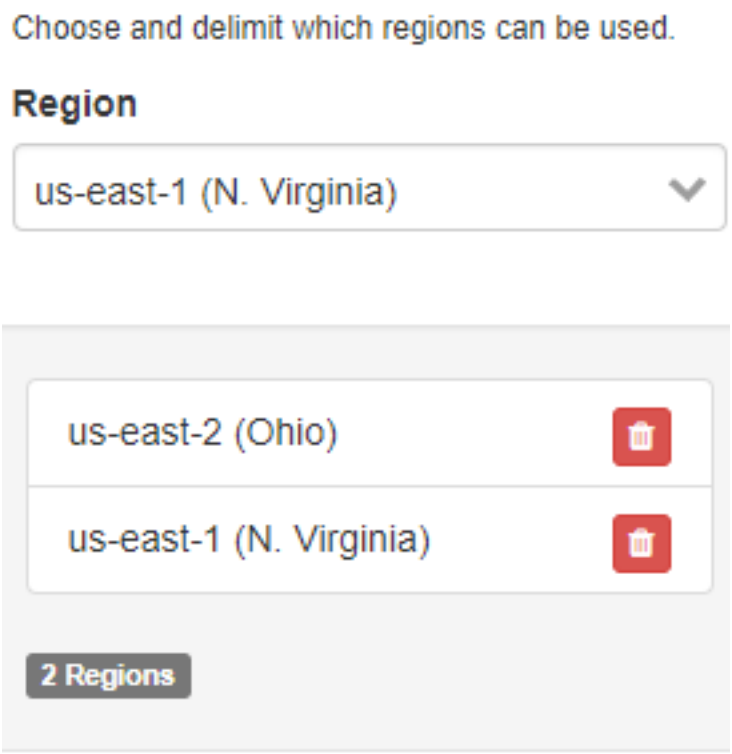


List of your datacenters.



You can select a provider, regions and zones.

Selecting an existed region.



Servers

Inventory > Server

Server is a computer or a single program instance, which manages access to a centralized resource.

Field	Description
Hostname	Hostname
Ipv4 Private	Ipv4 private, It will warning if there are any duplication,
Ipv4 Public	Ipv4 public, only for external servers.
OS	Operation system can be Linux adn Windows. Distro can be ubuntu, centos or any other.
CPU	CPU
Memory	Memory
Environment	Production | Development | Stage | . . .

Selecting the OS

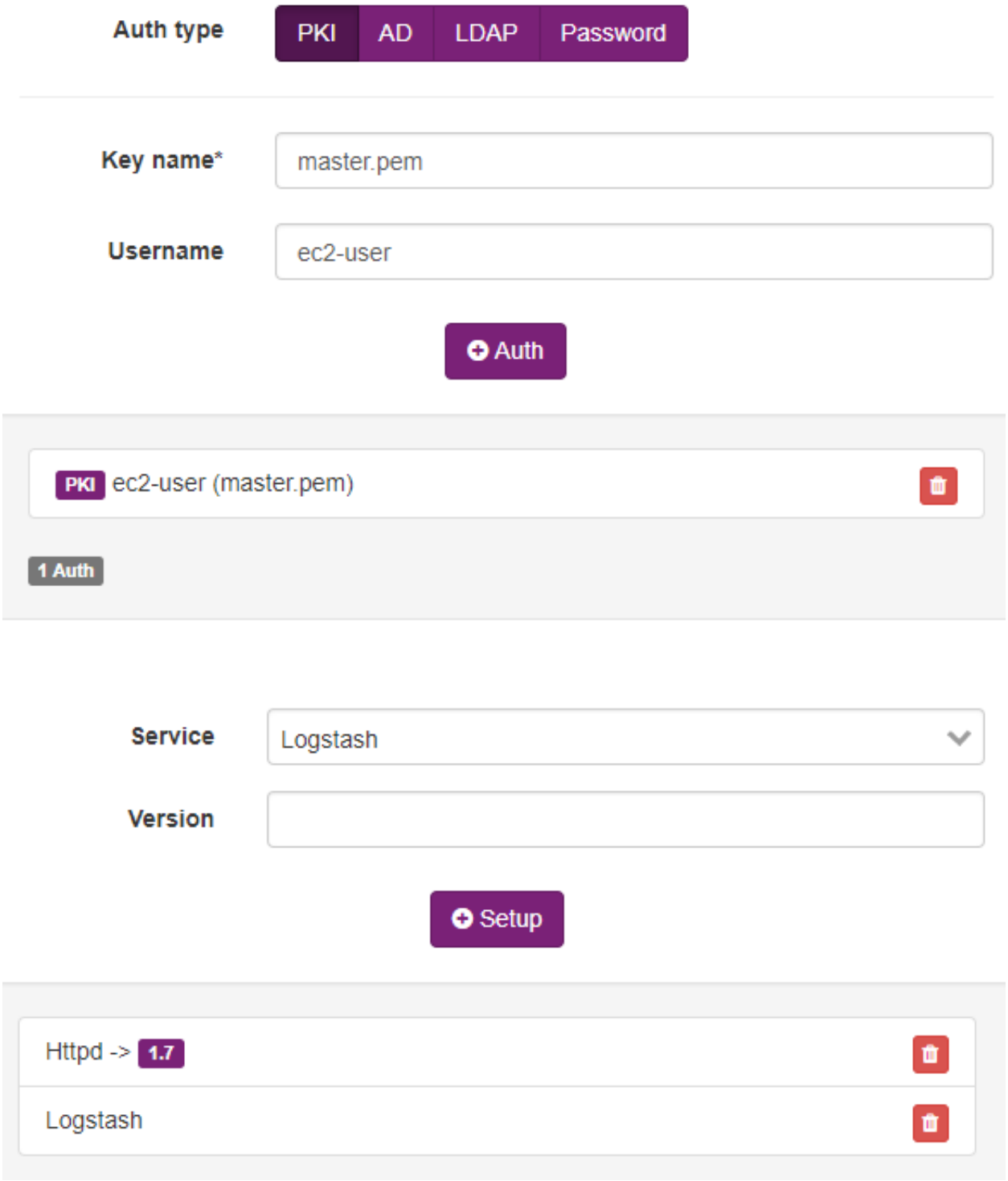
Server details

Field	Description
Storage	Storage configuration as a mount path, size in GB and if is a boot device.
Data-center	Providers, region and zones, used by cloud datacenters, you can put the instance id on id_instance field, avoiding Maestro to duplicate this server.
Auth	Dummy information about how the team can loggin into servers.
Service	Show up all services running, It can be used on `Application Manager` page to track the service configuration.

Assing a dc name, region and zone on that server.

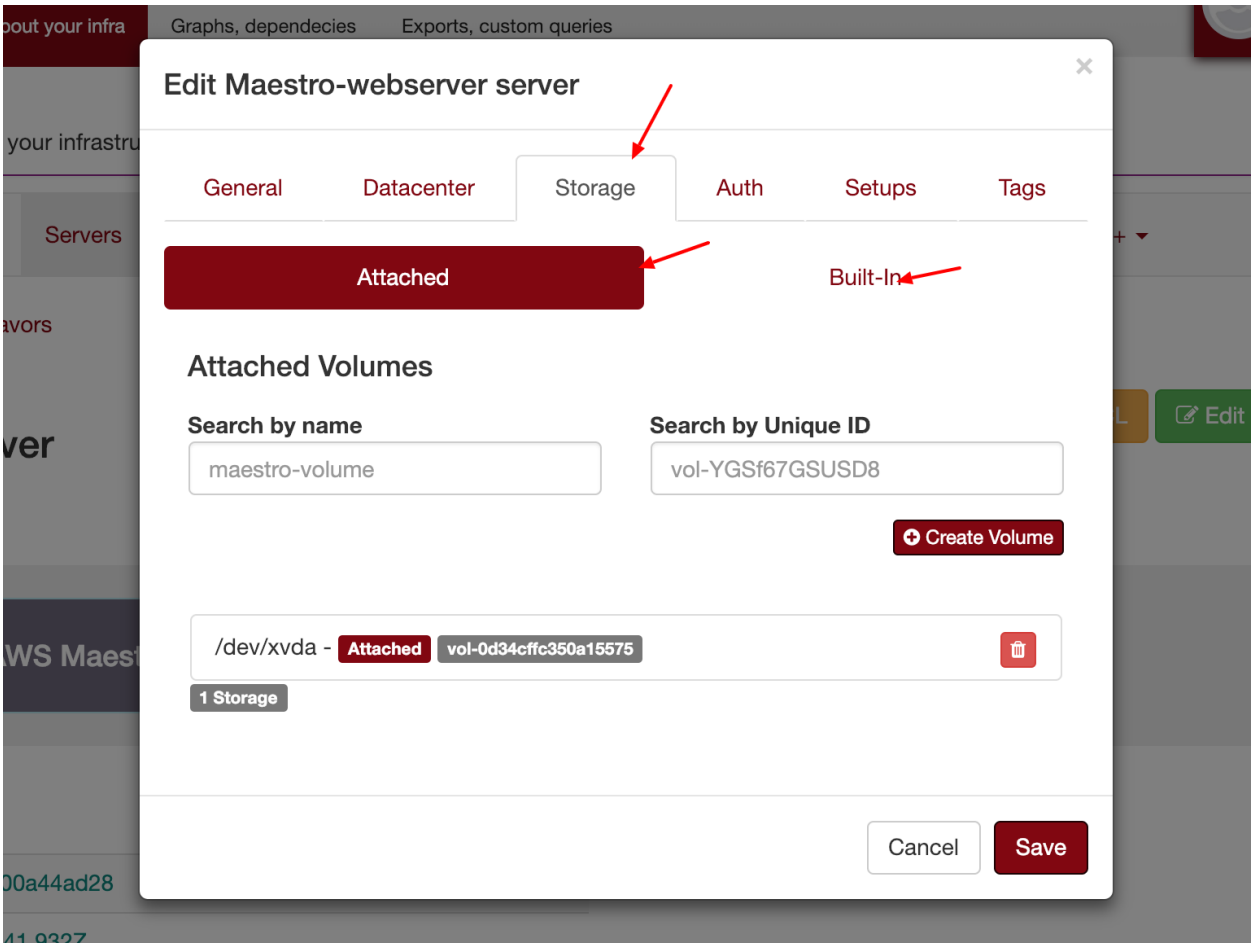
describe how you can to access and authenticate on that server.

Note: Services can be a very usefull field, Maestro are able to correlate services installed on servers and applications, as an example, you can create an Oracle Database on Databases applications, then you can create a new server and assign this server to that database, Maestro automatically do a service/application bound.



Related services.

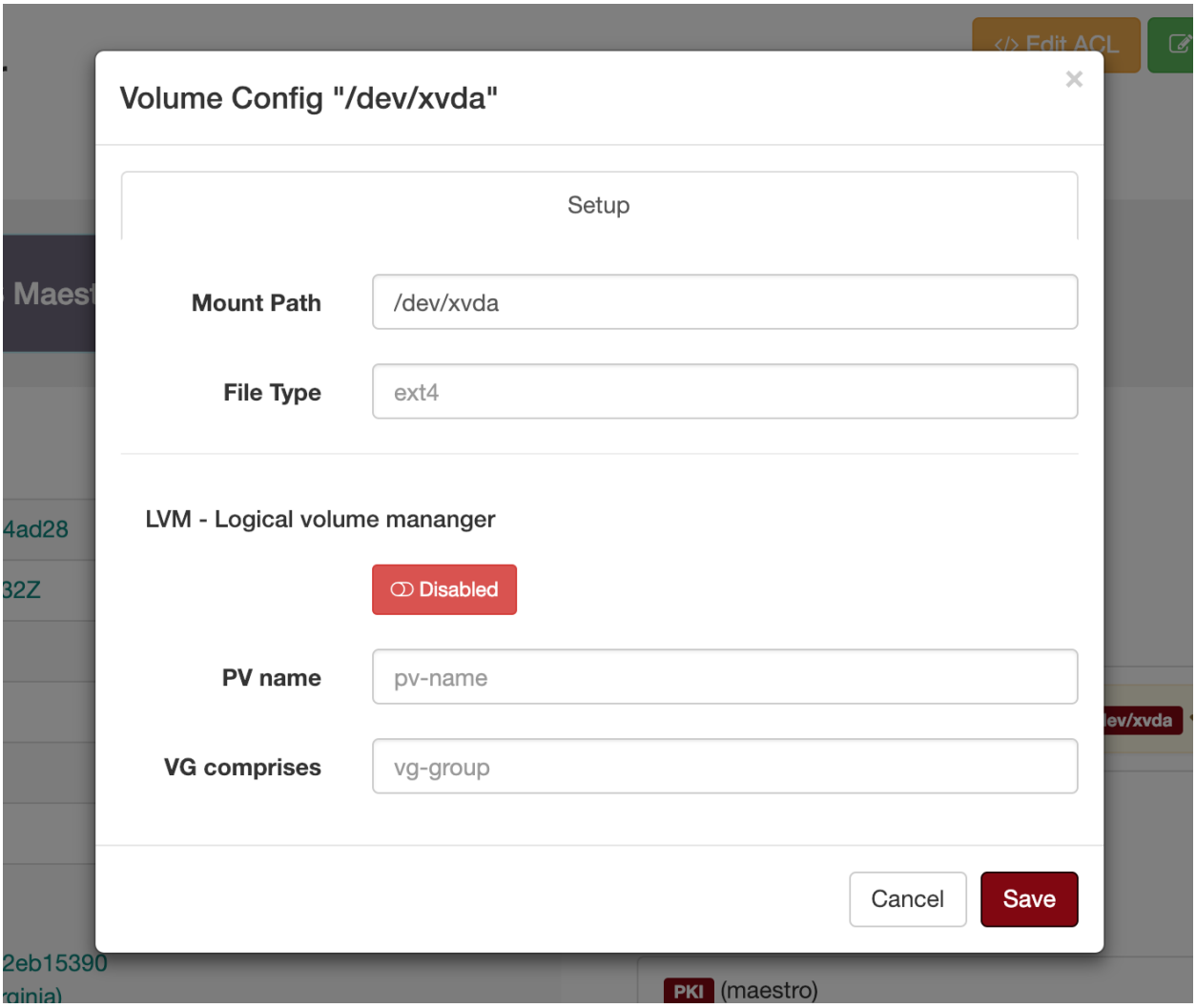
Volumes



Can be attached or built-in:

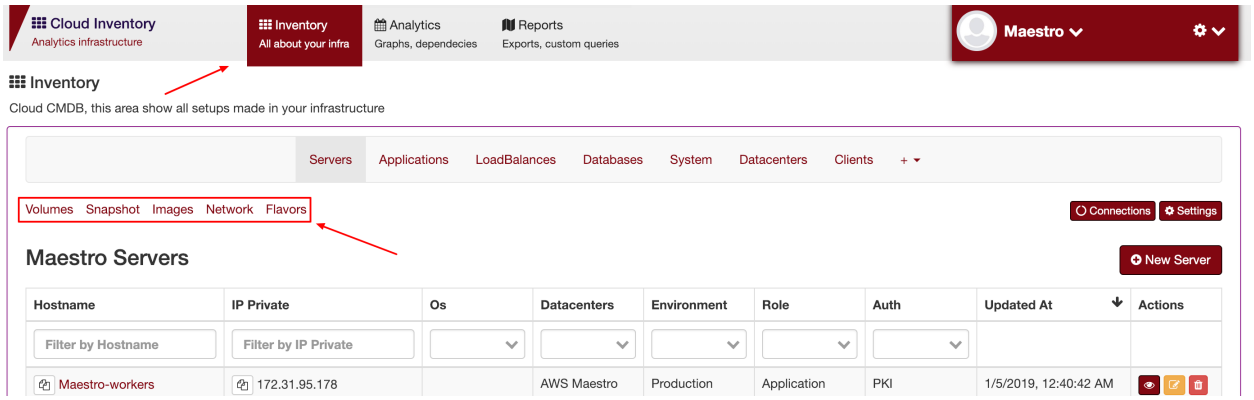
- **Attached** is a network storage or distributed storage service (ex: NFS)
- **built-in** is a hard drive set in that server, very common on bare metal.

You will be able to describe where the mount path are, which file type, and a virtual volume configuration (LVM).



Cloud Server Resources

Volumes, flavors and images are servers resources provide by cloud providers, on top of servers you can create/list those resources.



- **Volumes**: List of volumes (ex: EBS, HardDisk)
- **Flavors**: Instance flavors.

- **Images**: List of images, it used to build new servers. [As a template]
- **Network**: Network provider resources, as an example security groups, acls, vpcs, subnets and etc.

Apps

Inventory > *Application*

Applications are a program or group of programs designed for business responsibility.

Apps fields:

Field	Description
Name	Hostname
Environment	Production | Development | Stage
Language	What language this application was made.
Cluster mode	

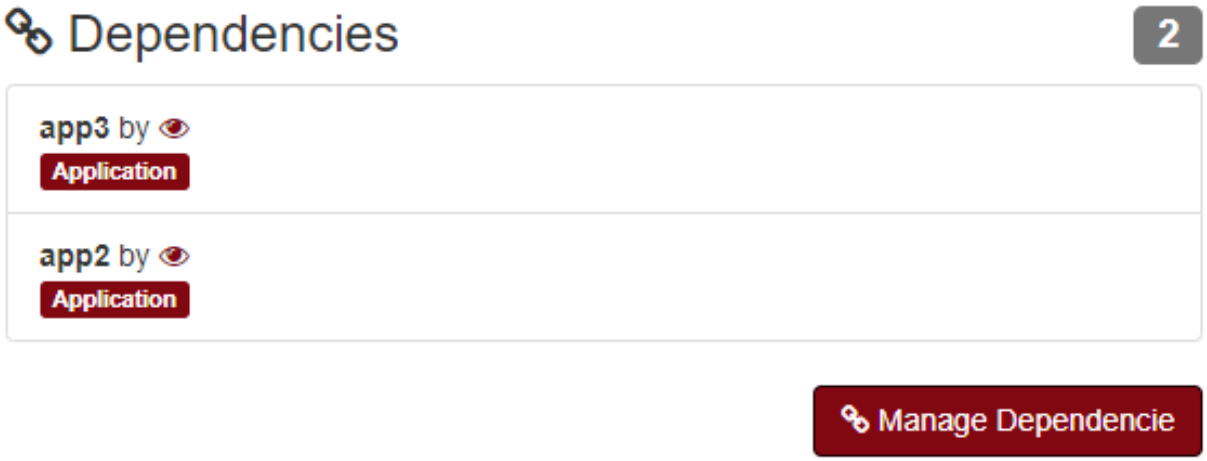
Specification

Field	Description
Role	Endpoint, commands, health check and more.
System	Accountant system/s.
Server	Where the application are running.
Deploy	List of ways to deploy this app.

Selecting a language that applications was made. As an example, node or php.

Add dependency

Note: A given applications with connects to this application, as an example webserver connects to database, so database is a dependency of webserver.

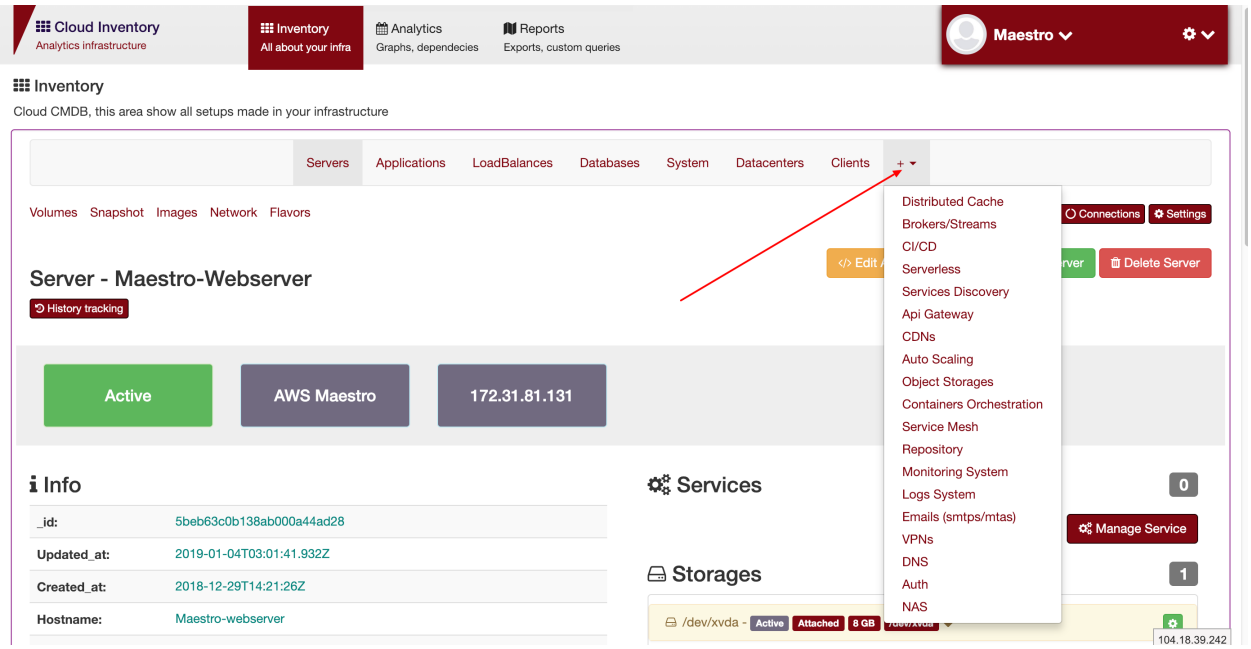


Adding dependencies.

Resources

Inventory > ${Resource}

Resources is a no-business application, can be brokers, databases, loadbalances, service logs, dns and more.

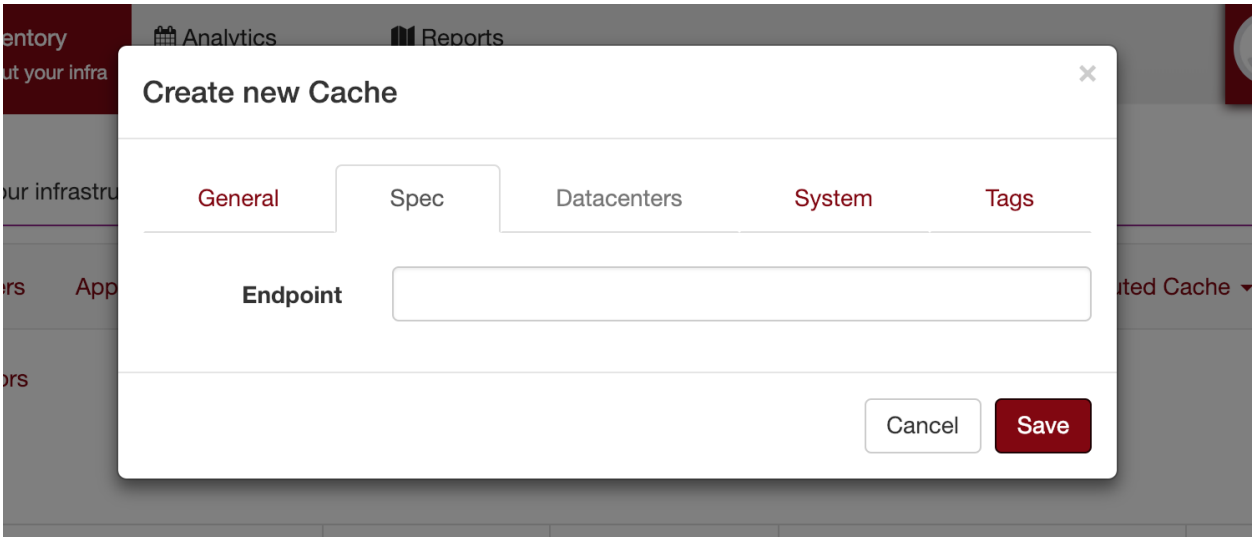


Resources types:

Family	Description
Distributed cache	Cache system, as a Redis, Memcache and etc.
Brokers/Streams	Message or streams system, can be RabbitMQ, SQS, Kafka, Spark Streams and more.
CI/CD	Ci Tools, as an Jenkins, Atlassian Stack, AWS Pipeline and more.
Serverless	Cloud functions, as an AWS lambdas, step functions, google function, Kubeless and more.
Services Discovery	Consul, etcD, hystrix can be consired as a service discovery.
Api Gateway	Api Gateway service, like Kong, AWS api gateway and/or a nginx.
CDNs	CDNs services, cloudflare, akamai, cloud front and etc.
Auto Scaling	Autoscaling setup
Objects Storages	Objects storages, S3, GlusterFS, Ceph, DO Storages and more.
Containers Orchestration	Main pieces of orchestration tools, kubernetes master/slave node, eks nodes, docker swarm nodes, mesos and etc
Service Mesh	Like Linkerd, IstIO, Consul or AWS x-ray
Repository	Nexus3, npm repository, docker repository, S3, private pip, nuget, gems, maven and more
Monitoring System	Prometheus, New Relic, Data dog, zabbix, nagios and etc
Logs System	ELK stack, data dog, graylog and etc
Emails	SMTP servers, postfix, or third service as a sendgrid
VPNs	VPNs Gateways
DNS	Bind9, route 53 and etc.
Auth	Authetication/Authorization systems, as an AD, LDAP, IAMs and etc
NAS	NAS Gateway
Corporate	ERP, internal services, as an Hana SAP, Protheus and more.

Specification

Field	Description
System	Accountant system/s.
Server	Where the resource are running.
Cluster	The service are running on a cluster mode.
Spec	Endpoint, commands, health check and more.



Databases

Inventory > Database

Databases are a programs to manage data store, can be relational and no relational.

The database inventory have a exclusive form for Oracle and MySQL, otherwise the generic form are able to fit on all databases types.

Field	Description
Oracle	You can register ASM DB, CDBs, RAC, grid system and/or golden gate backups
MySQL	It able to register features as Master/Slave, Aurora cluster, backups setups and more.

Oracle

Support version 10g, 11g and 12g

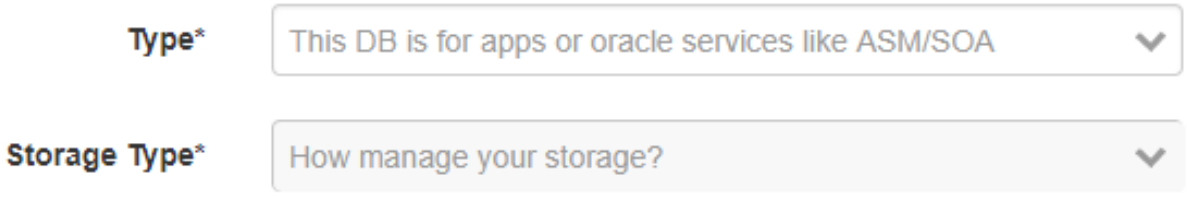
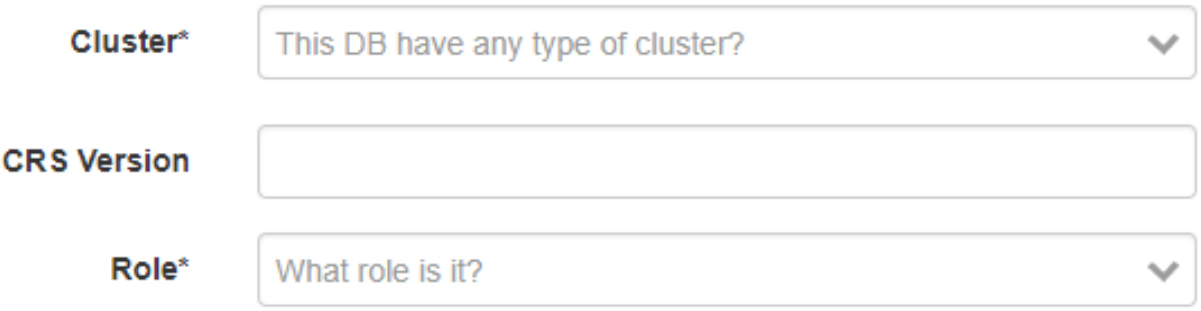
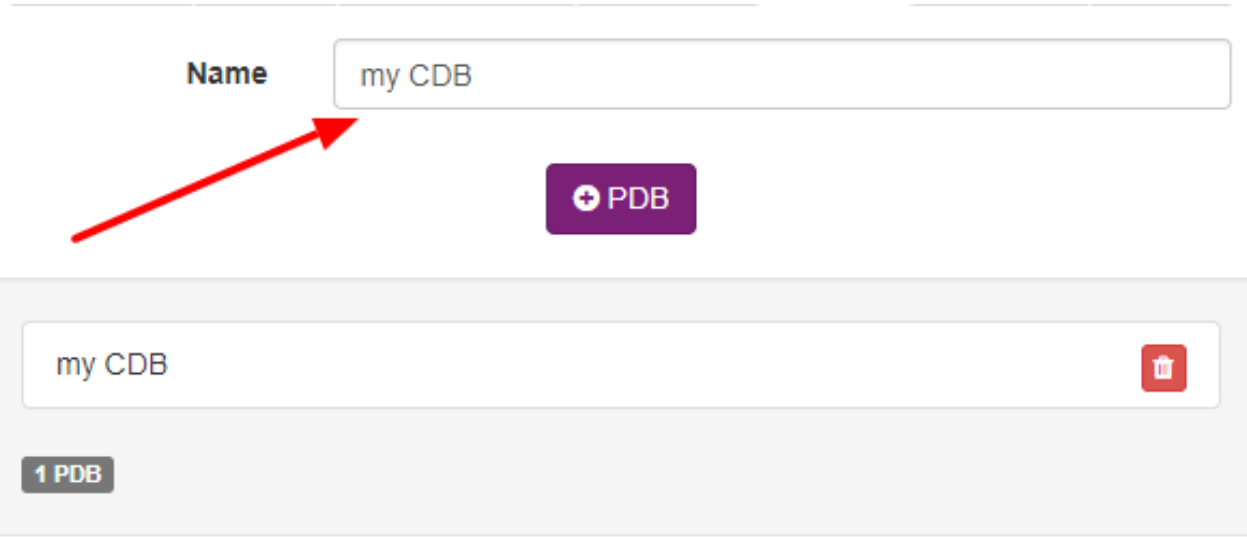


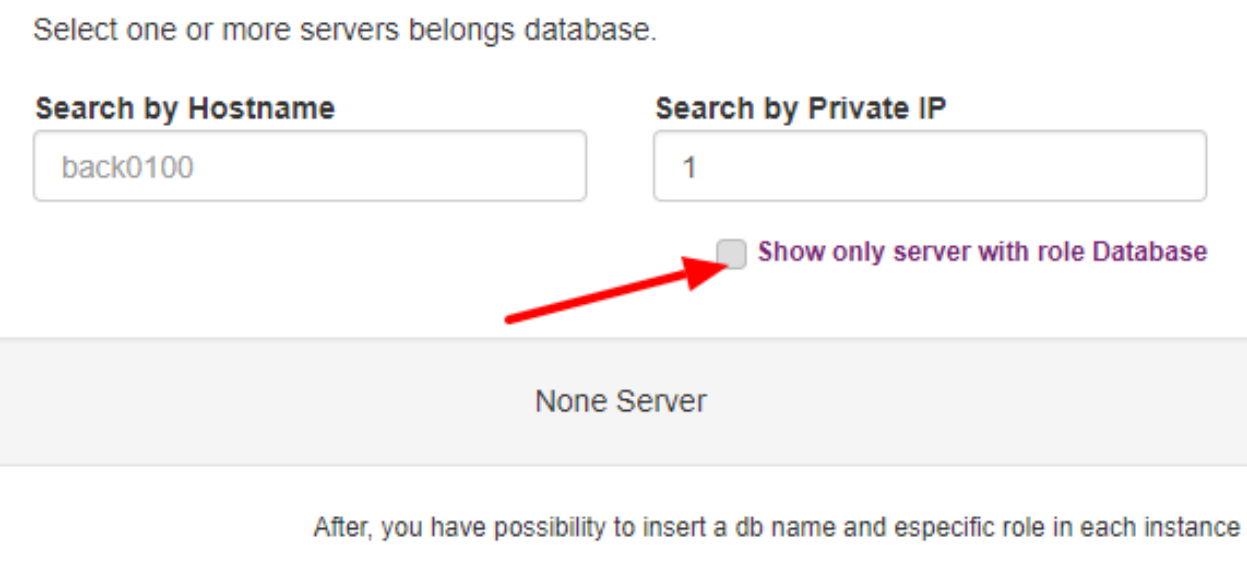
Fig. 1: Choose how Oracle will be storage the data, as a local disk, ASM or distributed storage system.



Choose how Oracle will be run, single node, RAC/Grid mode.



Which CDBS run on oracle database.



Which servers this db ran, if is a single node, a rac or it running on multiple servers.

MySQL

Support MySQL, AWS Aurora, MariaDB, Percona and etc

Which version and mode this db are.

Generic database

Generic support for all databases

Field	Description
Spec	Endpoint, port, commands, health check and more.
Datacenter	A given datacenter.
Server	Which servers this database are running.
CDBS	CDBS used by Oracle DBs.
System	Accountant system/s.

LoadBalances

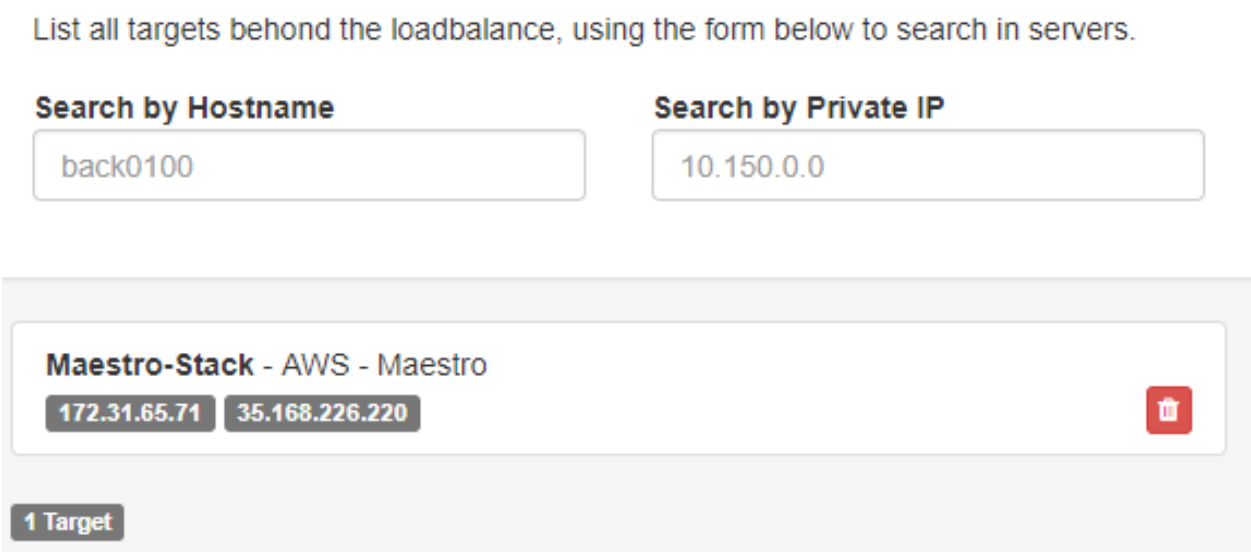
Inventory > *Loadbalance*

In computing, load balancing refers to the process of distributing a set of tasks over a set of resources, with the aim of making their overall processing more efficient. Wikipedia

Field	Description
Service	The loadbalance source.
Targets	To proxied applications
Servers	To proxied servers
Spec	Endpoint, healthcheck and more

Adding the healthcheck rule.

Selecting applications.

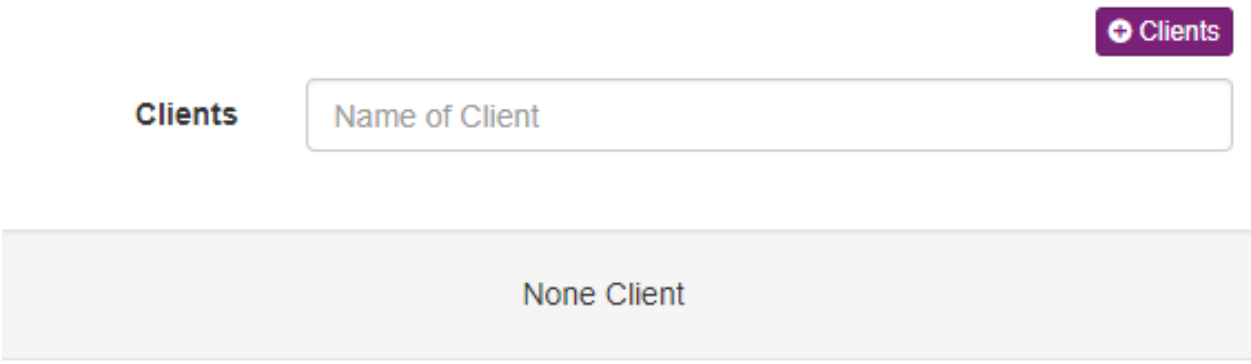


System

Inventory > *System*

A group of application and resources.

Field	Description
Links	Useful links
Clients	Accountant client/s.



Selecting the accountant client.

Clients

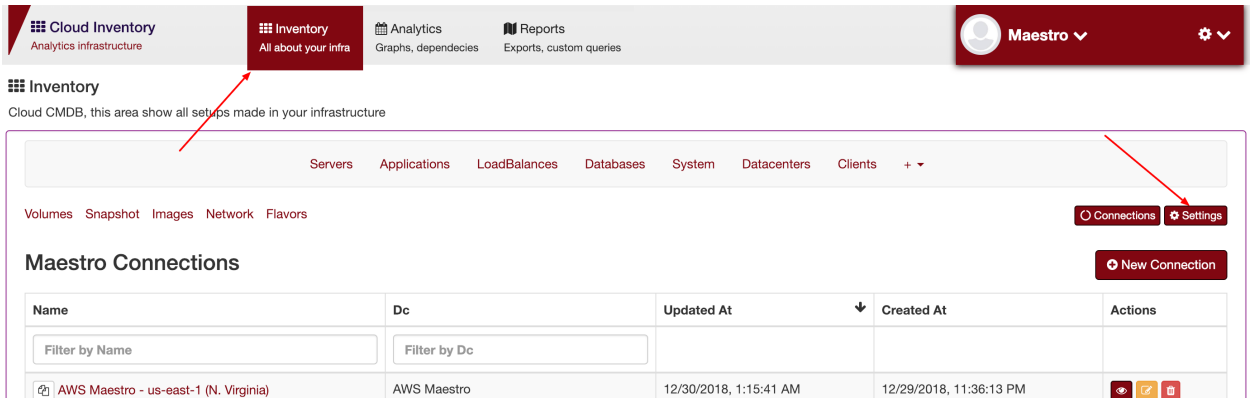
Inventory > *Clients*

Client can be a company and/or a team and/or a person, who owned a group of systems.

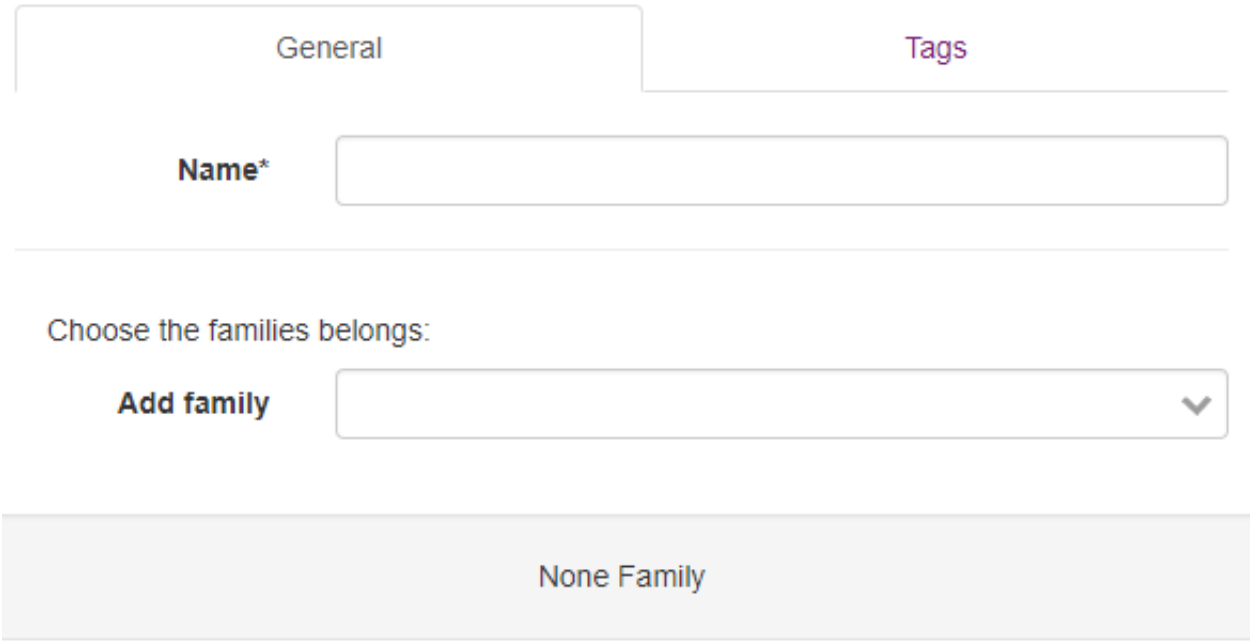
Field	Description
Contacts/Channel	Contact information

Services

Inventory > Settings > Services

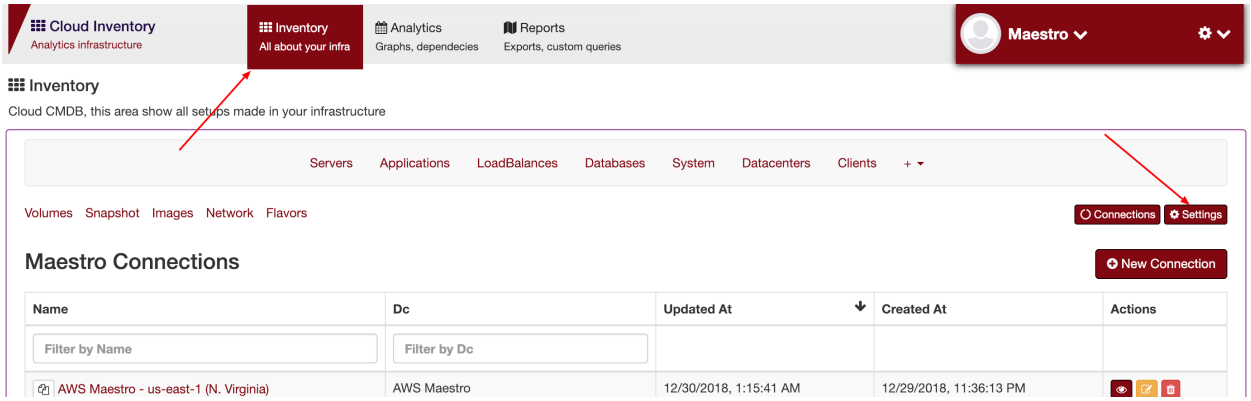


Services running on that server.



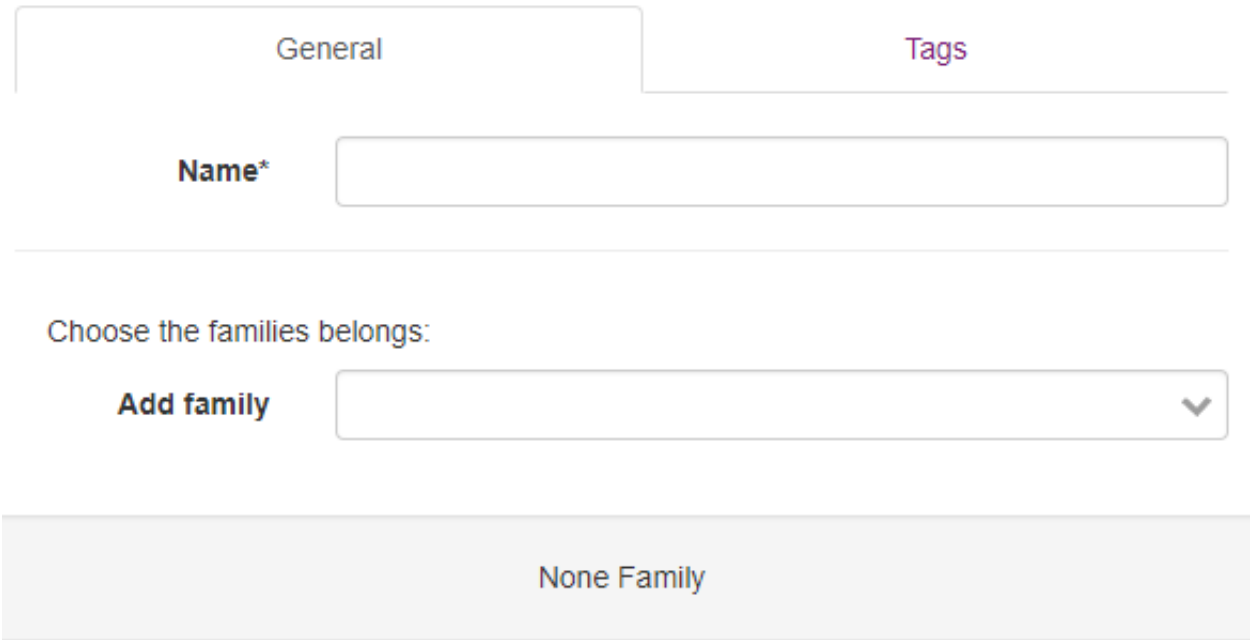
Creating a new service.

4.1.2 Options and configurations



Services

To create a new service, you can go to `settings -> services` and click on add new service:



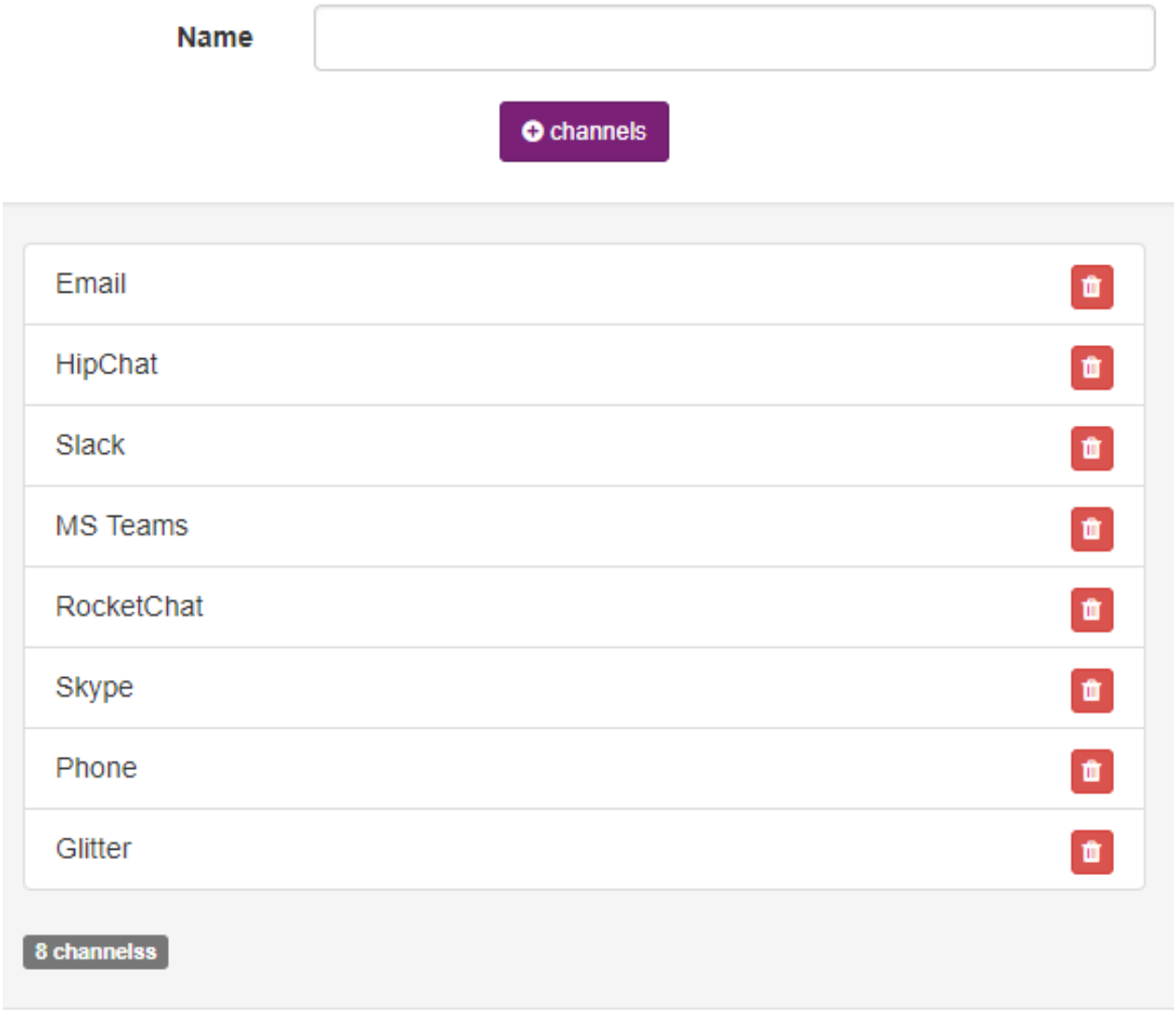
You can add, remove or update any service filled on Maestro database.

Config Options

You can add or change any option value.

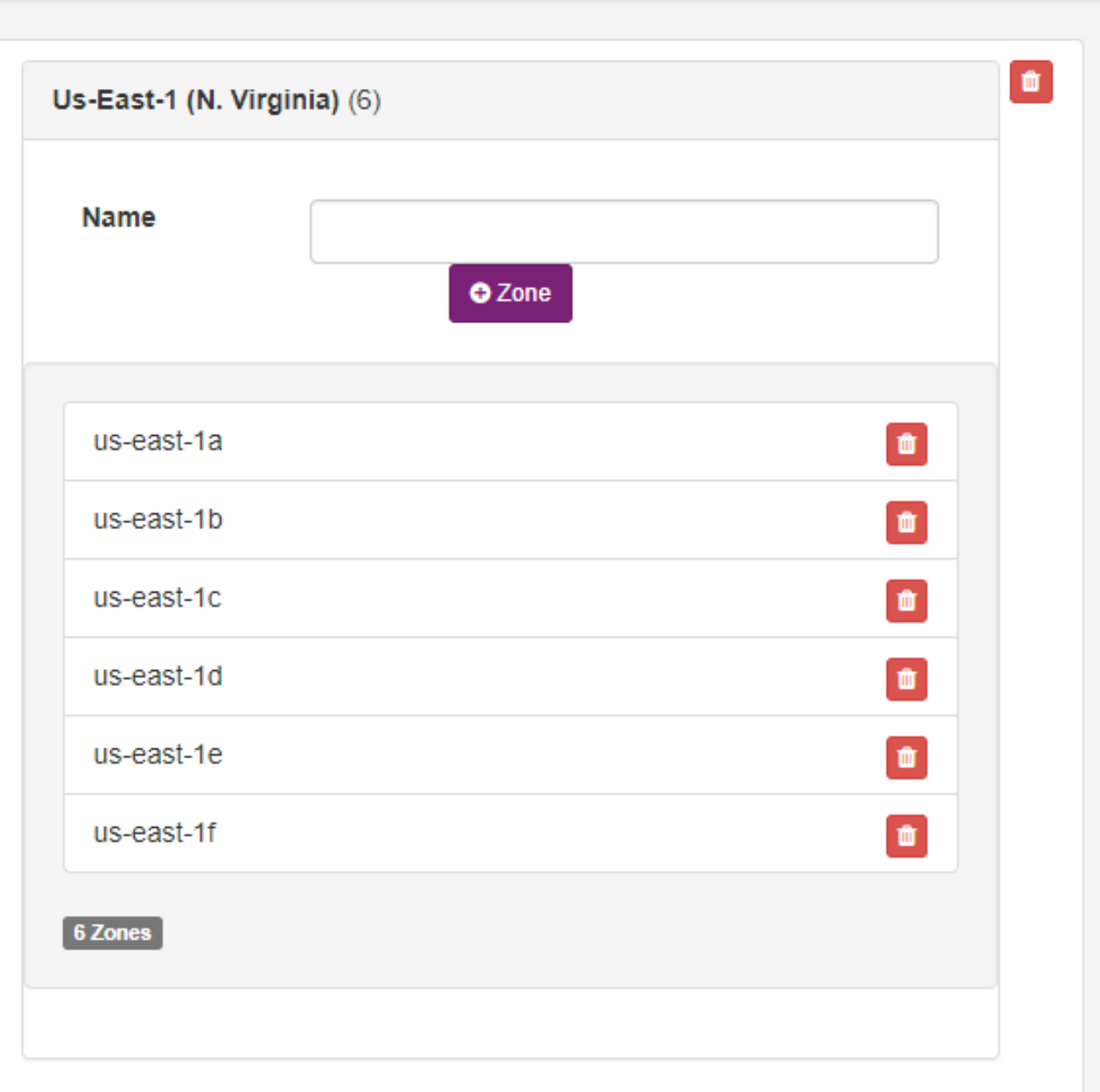
application_options	Applications options
clients_options	
connections	Time scheduler and crawler connections
database_options	
datacenter_options	
env_options	
server_options	
services_options	Services initial setup
system_options	

As an example, those are contacts found out it on `clients_options`.



Regions and zones

You can add a new region and/or a zone, go to `settings -> regions and zones`:

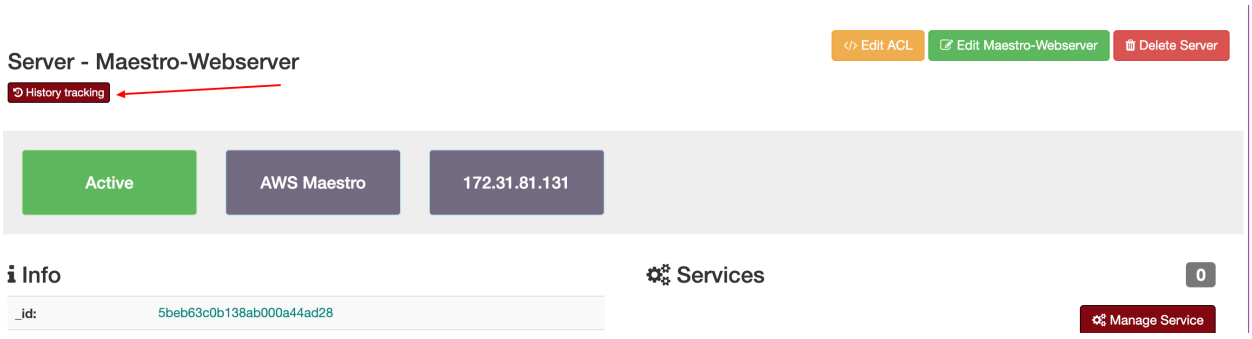


The default regions and zones.

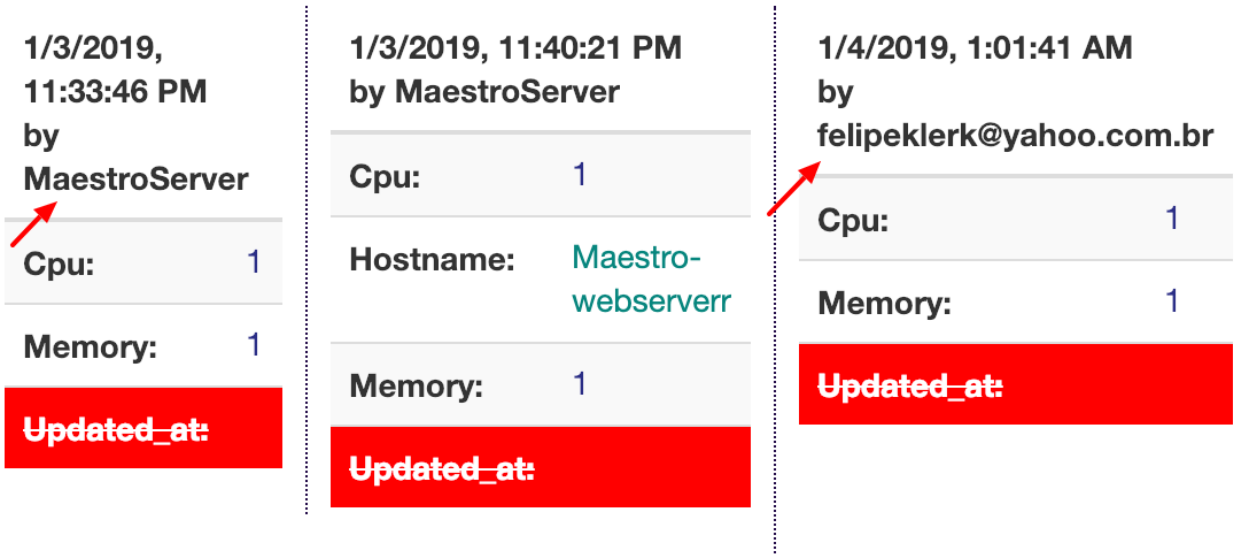
4.1.3 History Track

Inventory > Single Application > History Track

You can visualise all changes were made by users or by crawlers as a discovery or analytics. The audit service can analyse the difference between an old and a new entry and then record it.



Example of tracking changes page.



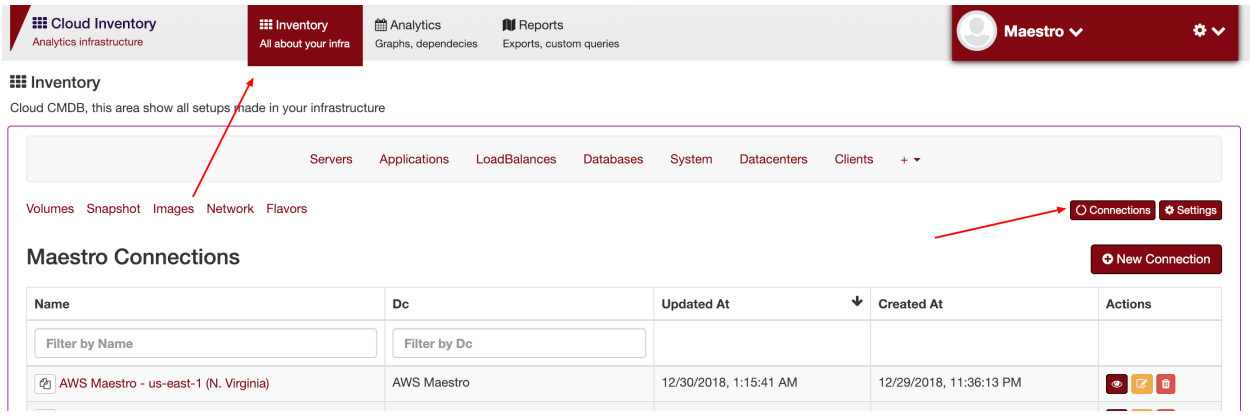
4.2 Auto Discovery

Maestro can connect in multiples cloud providers. You can track in a single dashboard, everything was created on multi-cloud and multi-region architecture.

To set up a new connection, you should follow three steps.

1 - Create datacenter on Maestro (select all regions used on that provider)

2 - Create a new connection on a given datacenter. - Go to inventory > connections.



3 - Allowing Maestro server to reach out a third provider using a readonly cloud credential such as aws access/secret key, azure subscription and more.

Maestro is able to connect on:

4.2.1 Connecting on AWS

To connect an one aws account, Maestro need to have an access_key and secret_key

---

Go to IAM service

Go to iam services on you AWS account dashboard.

Create an user - SecurityAudit

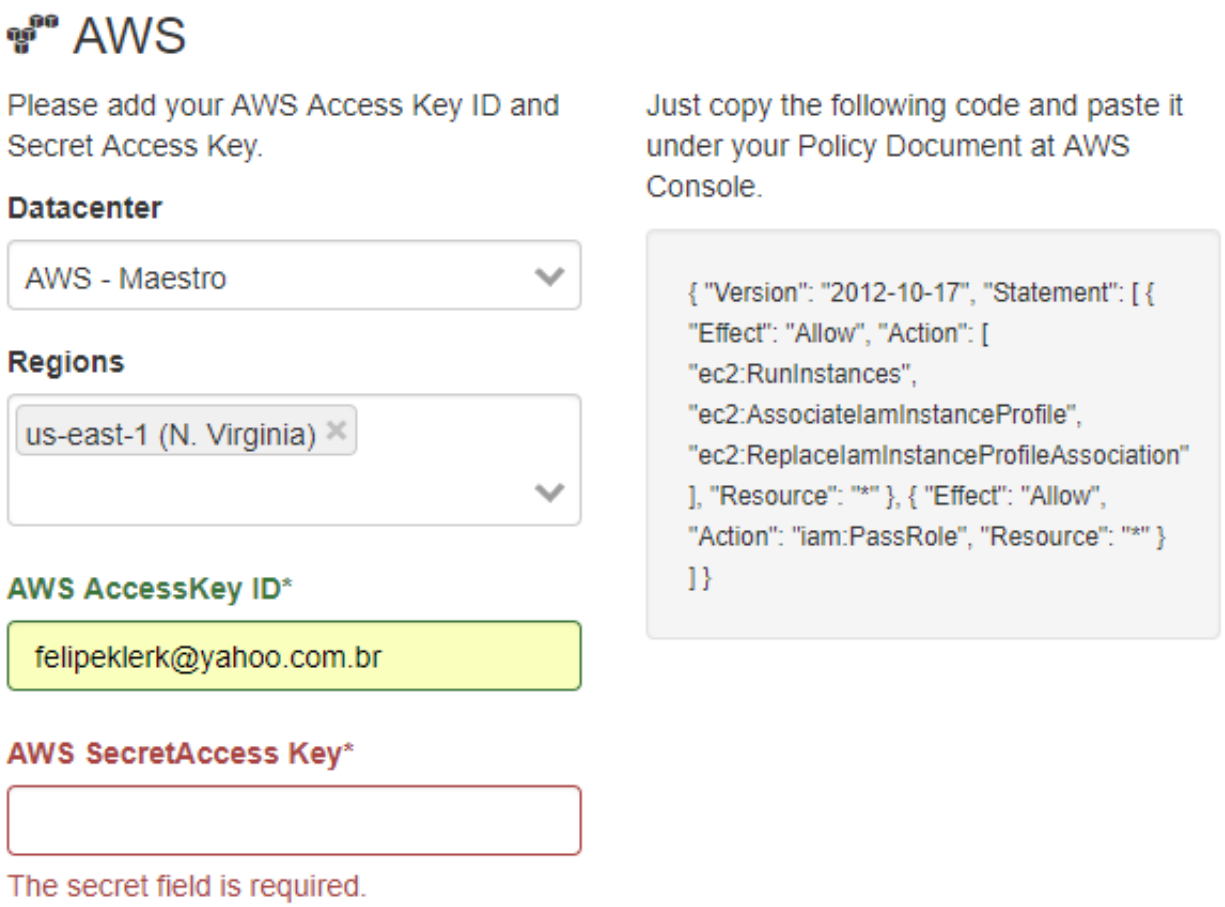
1. Go to user tab
2. Add user, select the access type as a `programmatic access`
3. Choose to attach an existed policy on user
4. Select `SecurityAudit` policy

Getting AWS Key and Secret Key

Copy and paste the aws key and secret key

List of permissions to grant.

server-List	ec2 describe_instances
loadbalance-list	describe_load_balancers and describe_load_balancers
dbs-list	rds describe_db_instances
storage-object-list	s3 list_buckets
volumes-list	ec2 describe_volumes
cdns-list	cloudfront list_distributions
snapshot-list	ec2 describe_snapshots
images-list	ec2 describe_images
autoscaling-List	autoscaling describe_auto_scaling_groups
brokers-List	sqs list_queues
cache-List	elasticache describe_cache_clusters
smtp-List	ses list_identities
serverless-List	lambda list_functions
serverless-support-List	lambda list_layers
dynamodb-List	dynamodb list_tables
gateway-List	apigateway get_rest_apis
security-list	ec2 describe_security_groups
network-list	ec2 describe_vpcs, describe_subnets, describe_vpc_peering_connections, describe_vpn_gateways, describe_vpc_endpoints, describe_route_tables, describe_network_interfaces, describe_nat_gateways and describe_network_acls



Setup connection on AWS

Note: PS: There is scheduler job activated by default, each resource type have your own window time, server-list will be updated for every 5 minutes, networks for every 2 weeks.

4.2.2 Connecting on Azure

To register use client id, tenant id, subscription id and secret token

---

Create and/or get Client ID

Create application in Azure Active Directory and you can then note the application ID.

1. Sign in to your Azure Account through the Azure portal.
2. Select Azure Active Directory.
3. Select App registrations.
4. Get Client ID and Tenant ID.

Generate Authentication Key

Provide Permission, select the application created and

1. Go to Settings, then Required permissions.
2. Click Add -> Select an API -> Windows Azure Service Management API and click Select.
3. Select required Delegated Permissions, click Select and then click Done.
4. Create a secret key
5. Select the application and go to Settings and Keys.
6. Add a description and expiry duration for the key and click Save.
7. The value of the key appears in the Value field.

Get tenant ID

When programmatically signing in, you need to pass the tenant ID with your authentication request.

1. Select Azure Active Directory.
2. Select Properties.
3. Copy the Directory ID to get your tenant ID.

Acquire Subscription ID

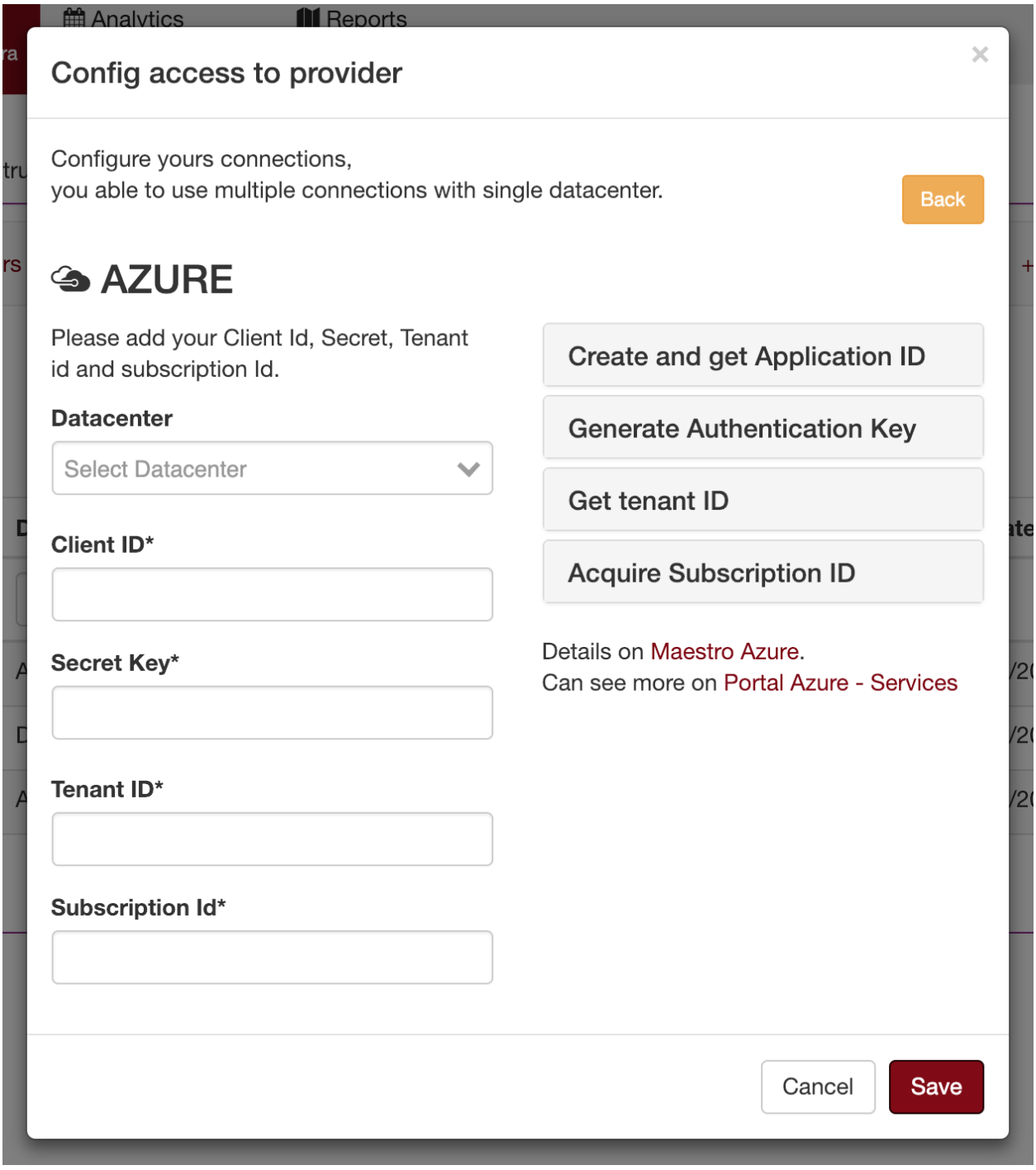
Grant permission for the application to access subscription that you want to configure.

1. Assign a role to the new application.
2. On the Azure portal, navigate to Subscriptions.
3. Select the subscription for which you want to grant permission to the application and note the subscription ID.
4. To grant permission to the application you created, choose Access Control (IAM).
5. Go to Add and Select a role. Pick the role as Reader. A Reader can view everything, but cannot make any changes to the resources of a subscription.
6. Select Azure AD user, group, or application in Assign Access to dropdown.
7. Type the application name in Select drop-down and select the application you created.

List of permissions to grant.

server-List	compute virtual_machines
volumes-list	compute disks
snapshot-list	compute snapshots
images-list	compute images
network-list	network network_interfaces network public_ip_addresses network route_tables network virtual_networks

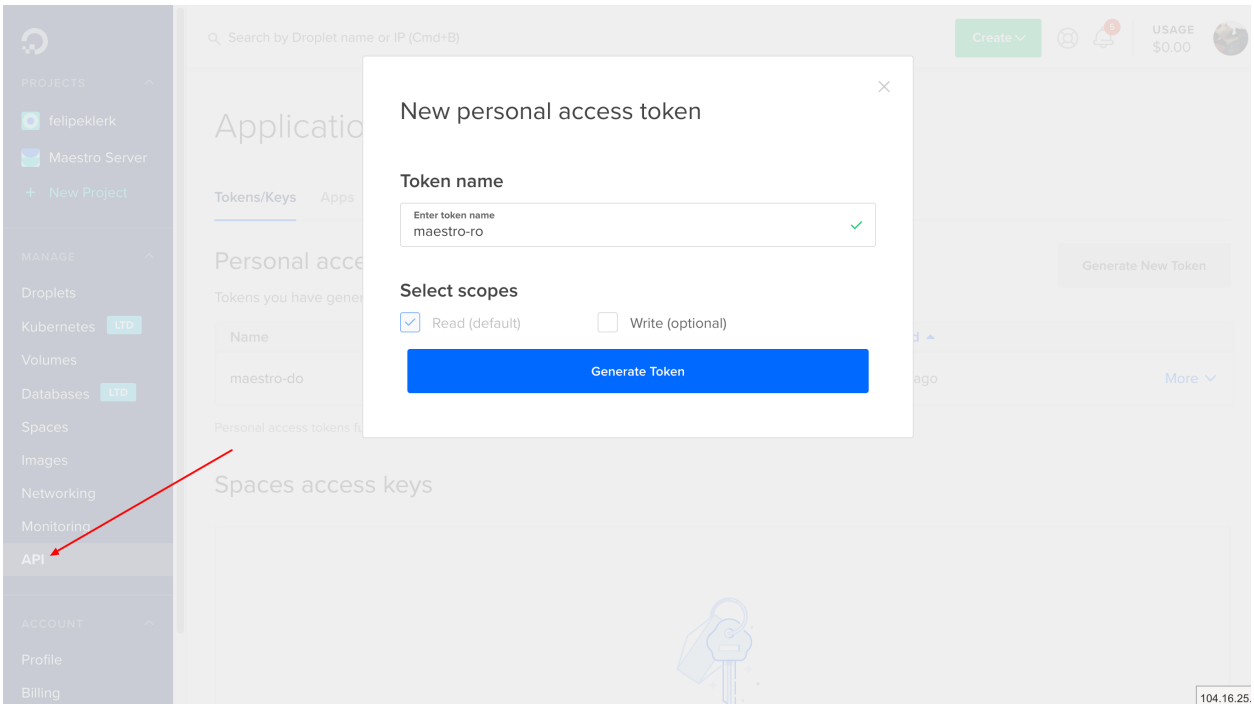
---



Setup connection with Azure

4.2.3 Connecting on Digital Ocean

To get the application token. Go to:



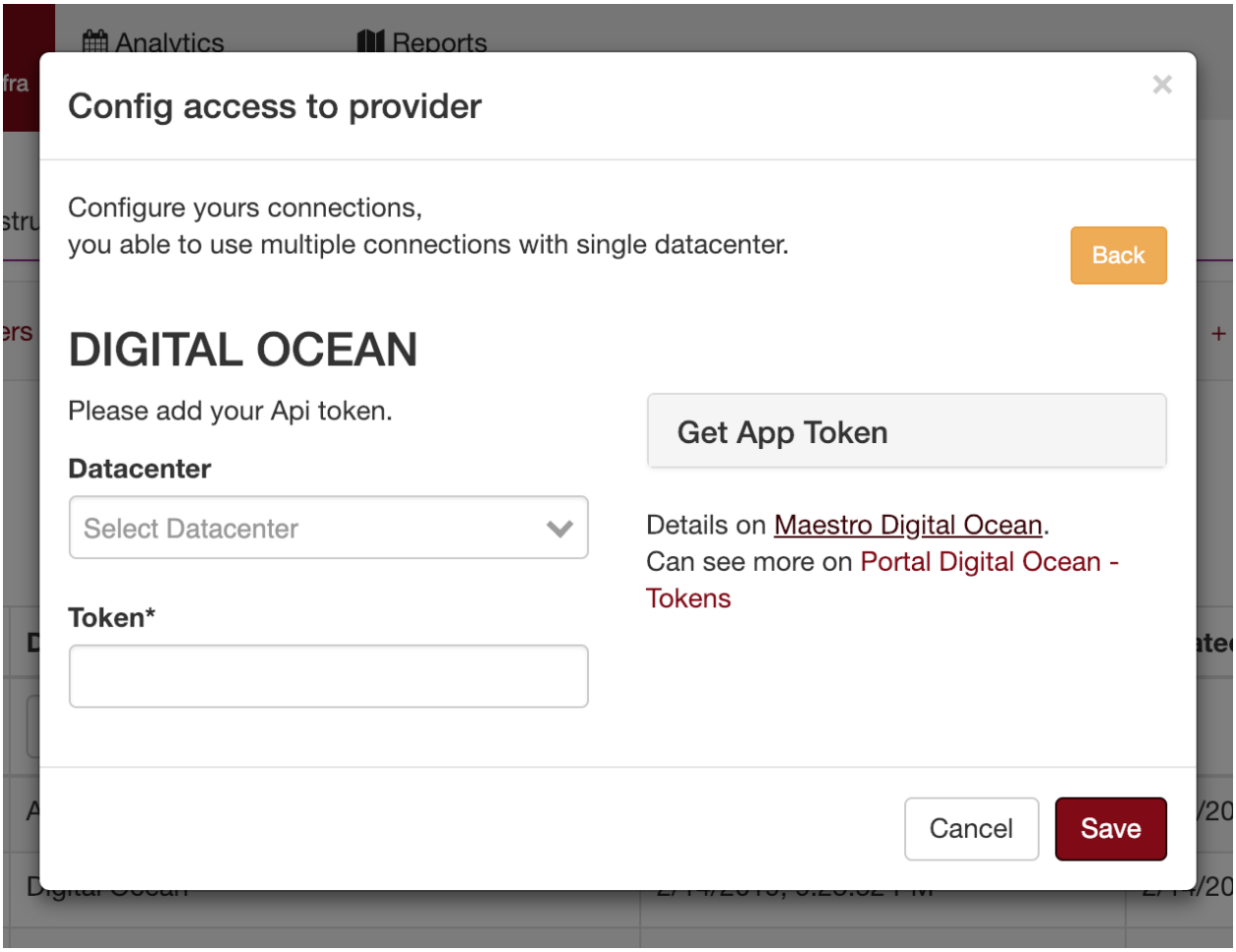
Getting the App Token

To create a new token, go to Digital Ocean dashboard:

1. Click on the API on the main menu
2. Go to the Applications & API
3. On the Tokens/Keys tab. Go to the Personal access tokens section
4. Click on to `Generate New Token`.

List of permissions to grant.

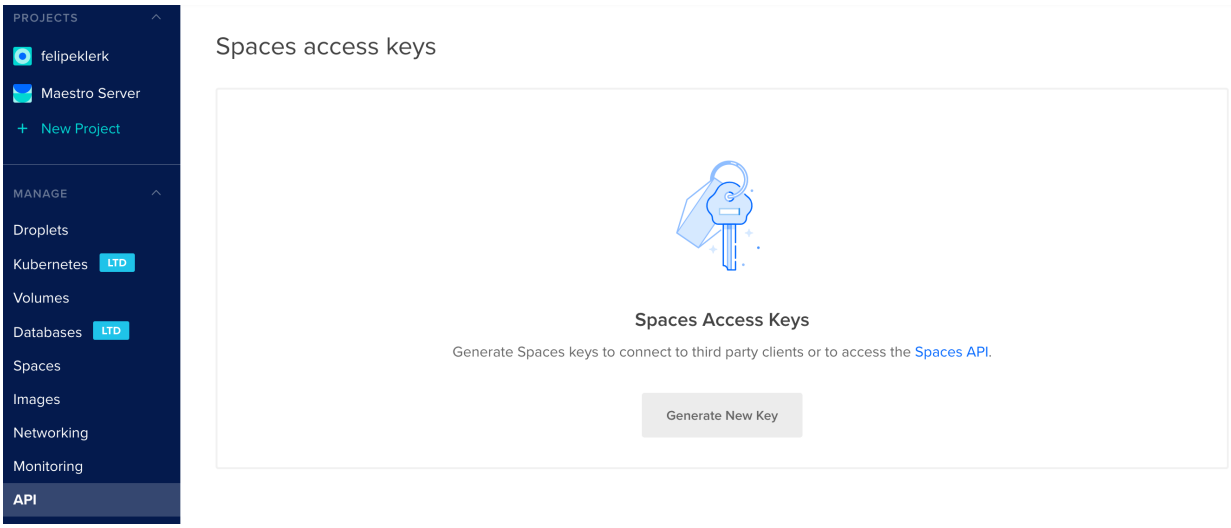
server-List	get_all_droplets
loadbalance-list	get_all_load_balancers
volumes-list	get_all_volumes
snapshot-list	get_all_snapshots
cdns-list	get_all_cdns
container-orchestration-list	get_all_kubernetes
images-list	get_my_images
network-list	get_all_firewalls



Setup connection with Digital Ocean

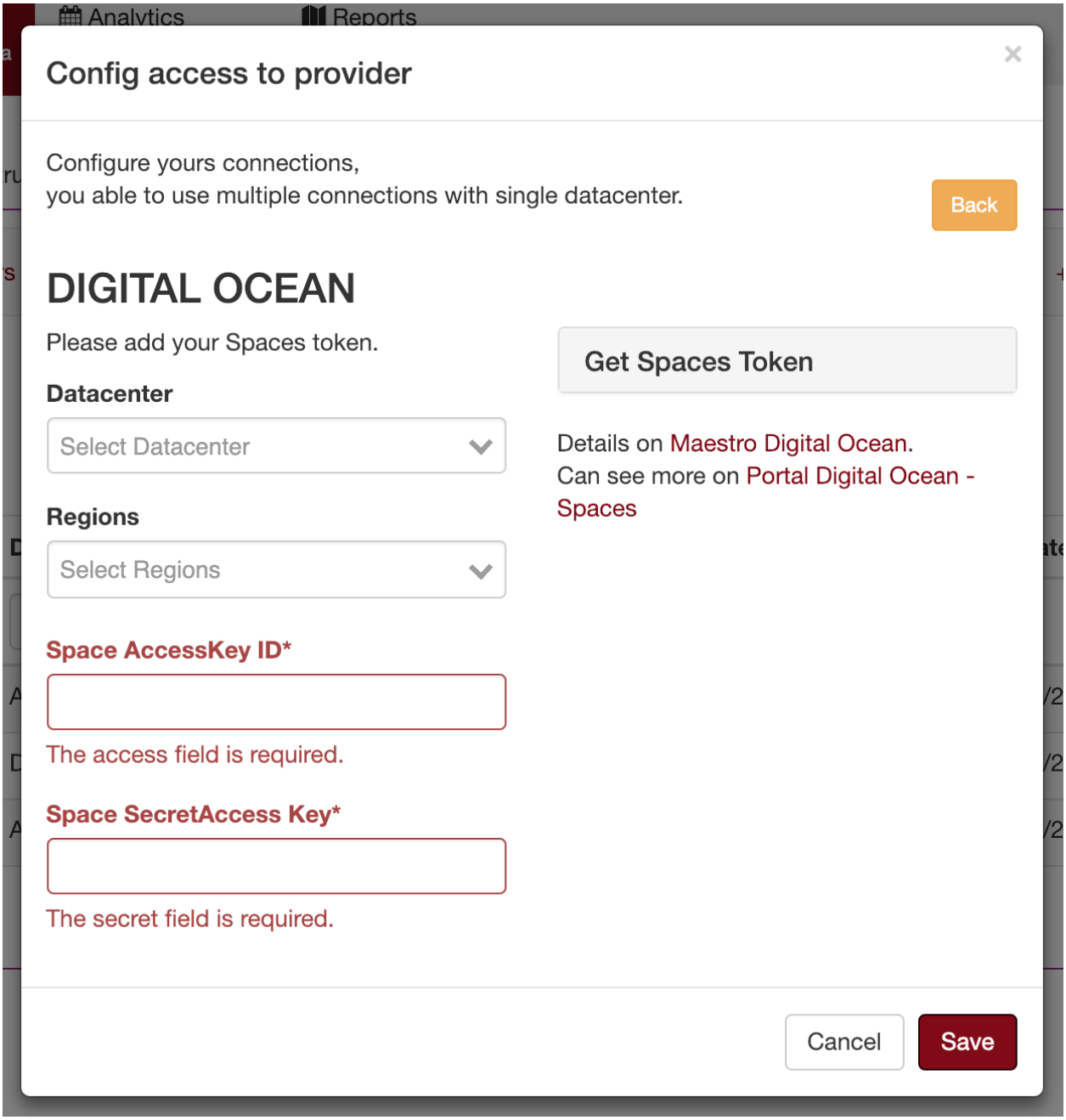
4.2.4 Digital Ocean Spaces

To register spaces key and secret key.



Getting Spaces Token

1. Click on the API on the main menu
2. Go to the Spaces token
3. On the Tokens/Keys tab.
4. Click on the `Generate New Token on Spaces`, and gets the key and secret key.



Setup connection on Digital Ocean Spaces

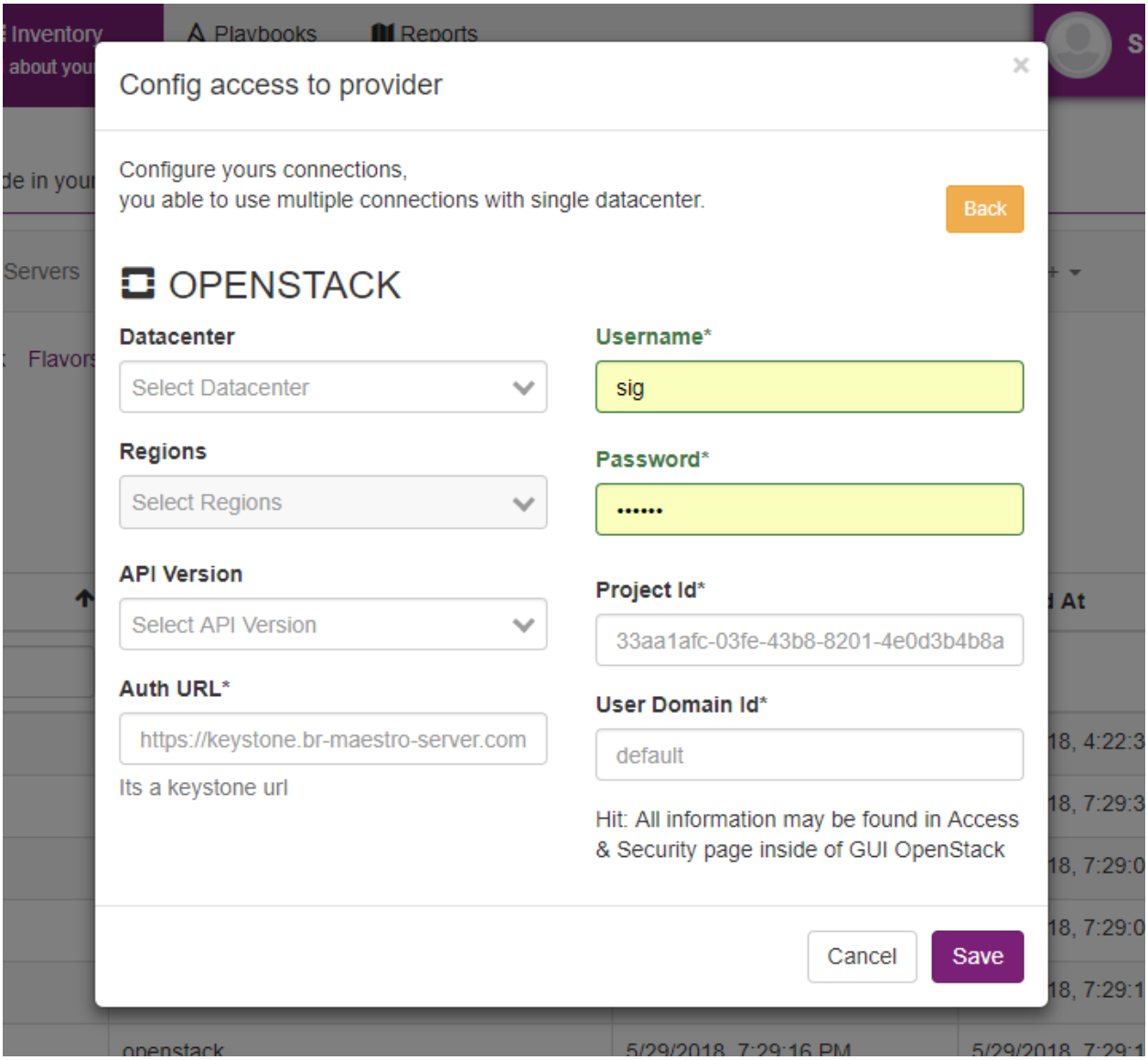
4.2.5 Connecting on OpenStack

To register one openstack account, use project name, url api, user, and password.

List of permissions to grant.

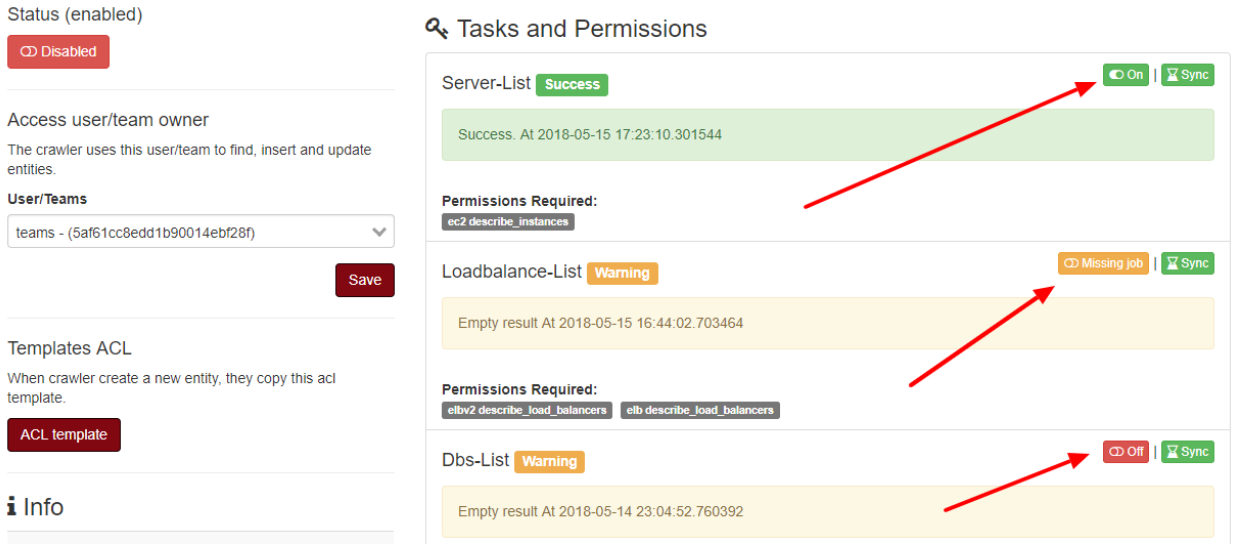
Server-List:	servers compute
Loadbalance-list:	load_balancers load_balancer
volumes-list:	volumes block_store
snapshot-list:	block_store snapshots
images-list:	compute images
security-list:	network security_groups
flavor-list:	compute flavors
network-list:	network networks, subnets, ports and routers

If you like, choose how the resource will be synchronized with an active and inactive button.



Setupconnection with OpenStack

Note: PS: PS: There is scheduler job activated by default, each resource type have specifc window time, server-list will be updated for every 5 minutes, networks for every 2 weeks.



Enable and disable the job

4.2.6 Using Ansible Facts

You can use ansible as a CMDB, first, you can generate Ansible output for your hosts, running

```
mkdir out
ansible -m setup --tree out/ all
```

Ansible will generate one file per host, next is to create a new connection on the resulting folder, Maestro can uses three method to get those files.

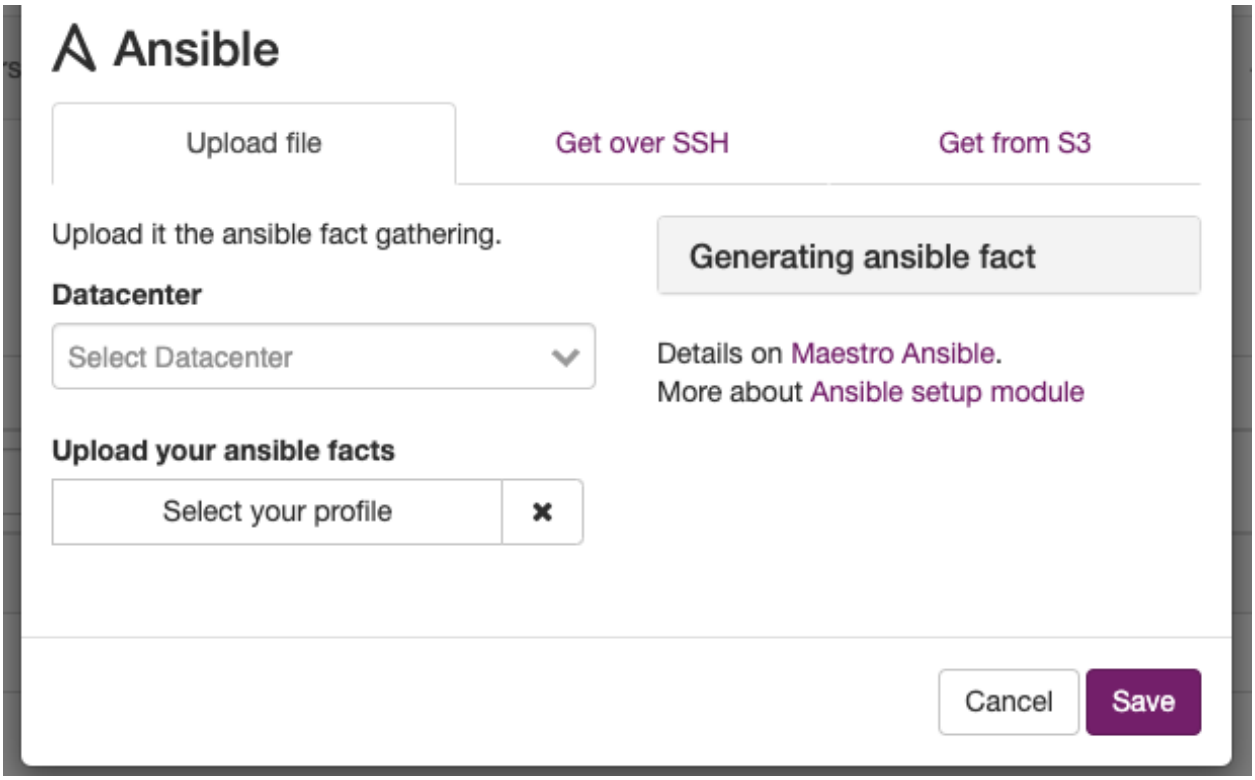
- Upload file
- Over ssh
- On S3 Bucket

Automatize the update process.

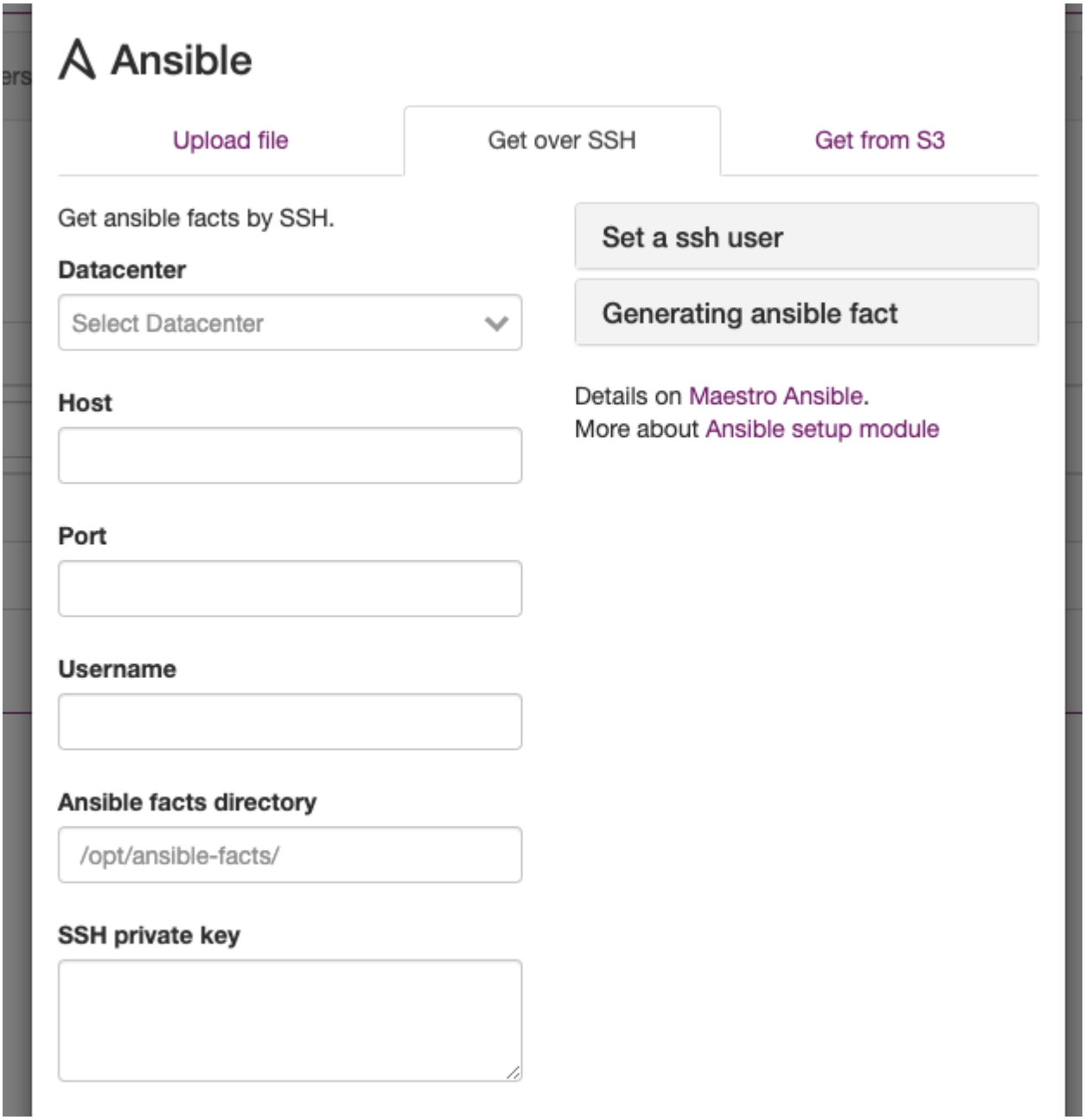
You can create cron jobs over ansible facts onto ansible manager server to automatize the update process.

Resources

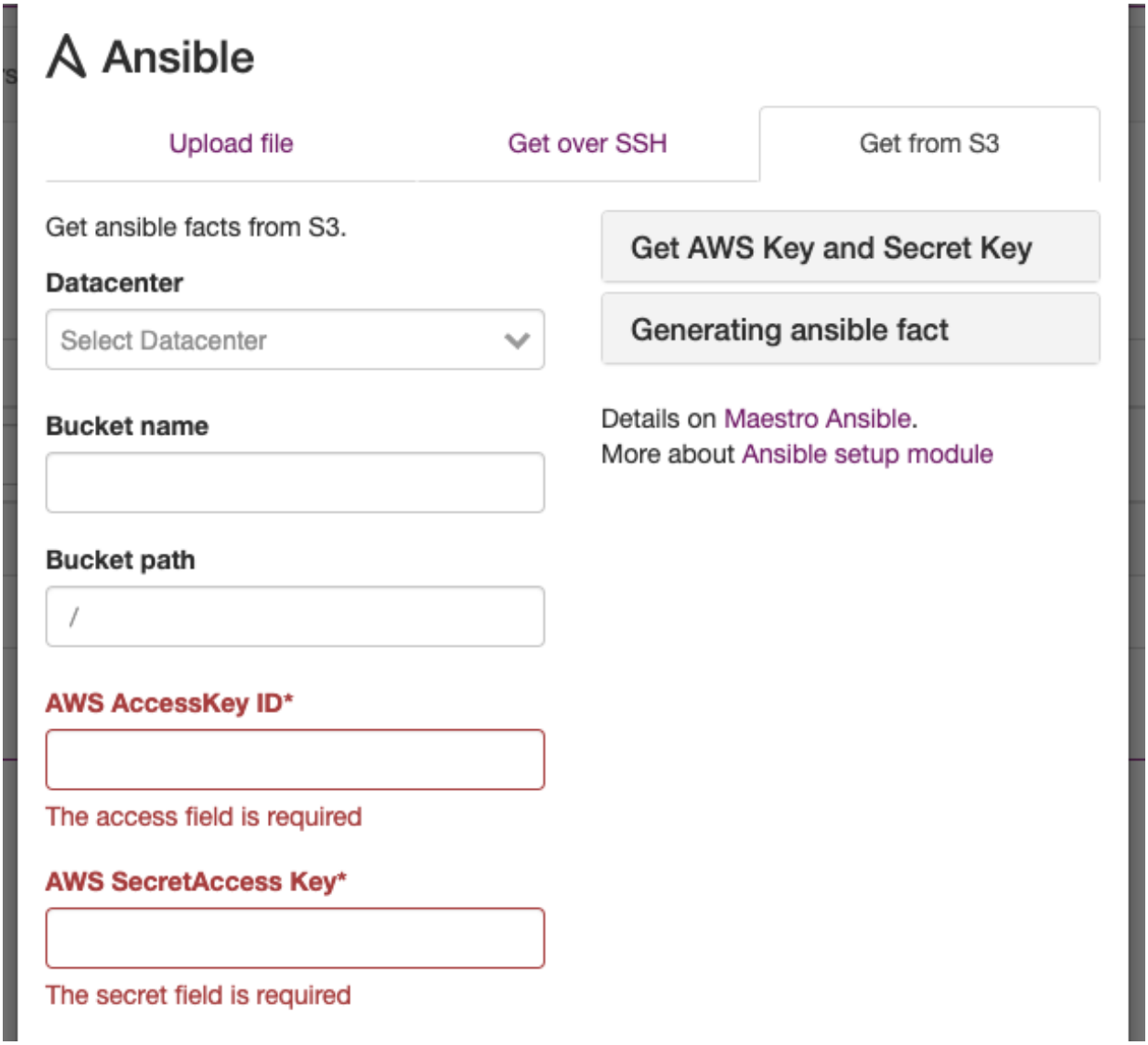
Server-List:
volumes-list:



Upload ansible facts


Ansible
Upload file
Get over SSH
Get from S3
Get ansible facts by SSH.
Datacenter
Select Datacenter
Host
Port
Username
Ansible facts directory
/opt/ansible-facts/
SSH private key
Set a ssh user
Generating ansible fact
Details on Maestro Ansible.
More about Ansible setup module

Set over ssh



Using S3 bucket

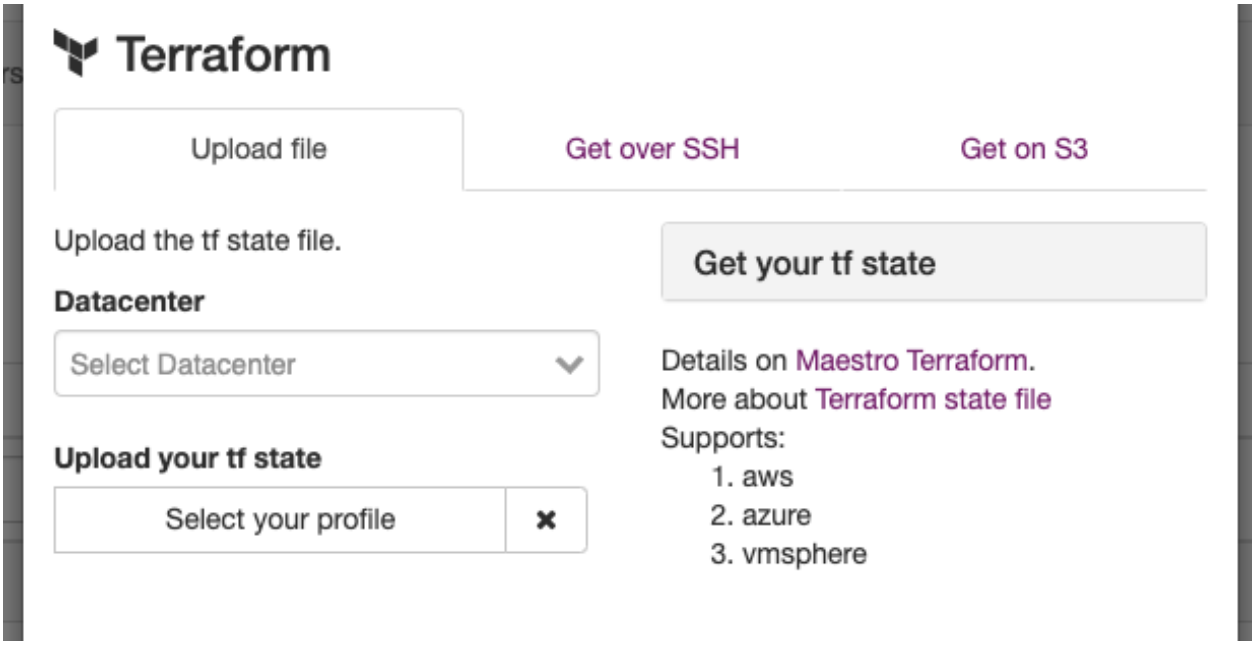
Note: PS: PS: There is scheduler job activated by default, each resource type have specifc window time, server-list will be updated for every 5 minutes, networks for every 2 weeks.

4.2.7 Using Terrafom State File

You can use terraform statefile as a CMDB.

Maestro can uses three method to get those files.

- By upload file
- Over ssh
- On S3 Bucket



You can use the same directory as the remote state folder.

Providers Support

Maestro can crawler and find information based on:

Provider	Servers	Volumes	Network	Images	Flavors	Applications
AWS	yes	yes				
Azure						
OpenStack						
DigitalOcean						
VMSphere						

yes - Maestro can find and get informations about that resource {empty} - That resource will be supported in a future releases. no - Maestro won't support that feature

---

Note: PS: There is scheduler job activated by default, each resource type have specifc window time, server-list will be updated for every 5 minutes, networks for every 2 weeks.

4.2.8 Import using JSON files

You can import servers from json files. Maestro can uses three method to get those files.

- By upload file
- Over ssh
- On S3 Bucket

Resources

server-List:
volumes-list:
snapshot-list:
images-list:
applications-list
flavor-list:

Example of json file

```
{
   "servers": [{
      "name" : "myname",
      "hostname" : "myhostname",
      "ipv4_private" : "127.0.0.2",
      "ipv4_public" : "89.89.89.89",
      "os" : {
            "base" : "Linux",
            "dist" : "Ubuntu",
            "version" : "14"
      },
      "datacenters" : {
            "name" : "random-1",
            "provider" : "randomdc",
            "region" : "region-1",
            "zone" : "zon1"
      },
      "role" : "Application",
      "environment" : "Production",
      "services" : [{}],
      "tags" : [{}],
      "cpu" : 2,
      "memory" : 2,
      "storage" : []
   }],
   "applications": [{
      "name" : "myname",
      "family": "Applications"
   }],
   "volumes": [{
      "name" : "vvolume",
      "size": "500"
   }],
   "flavors": [{
      "name" : "flavors"
   }],
   "snapshots": [{
      "name" : "snashots",
      "size": "500"
   }],
   "images": [{
      "name" : "myimages",
```

(continues on next page)

(continued from previous page)

```
        "size": "500"
    }]
}
```

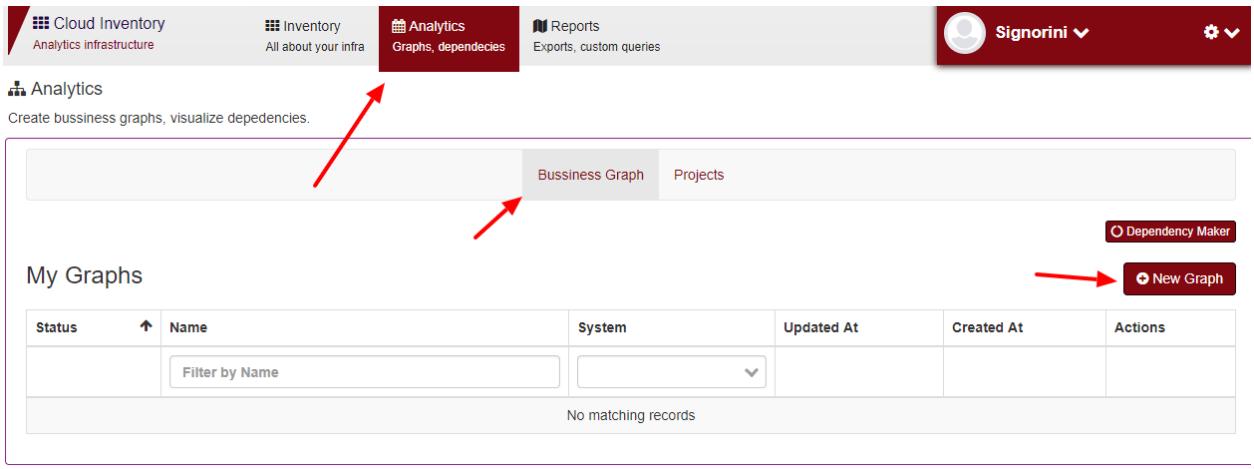
4.3 Graphs - Architecture maps

Visualize your cloud architecture

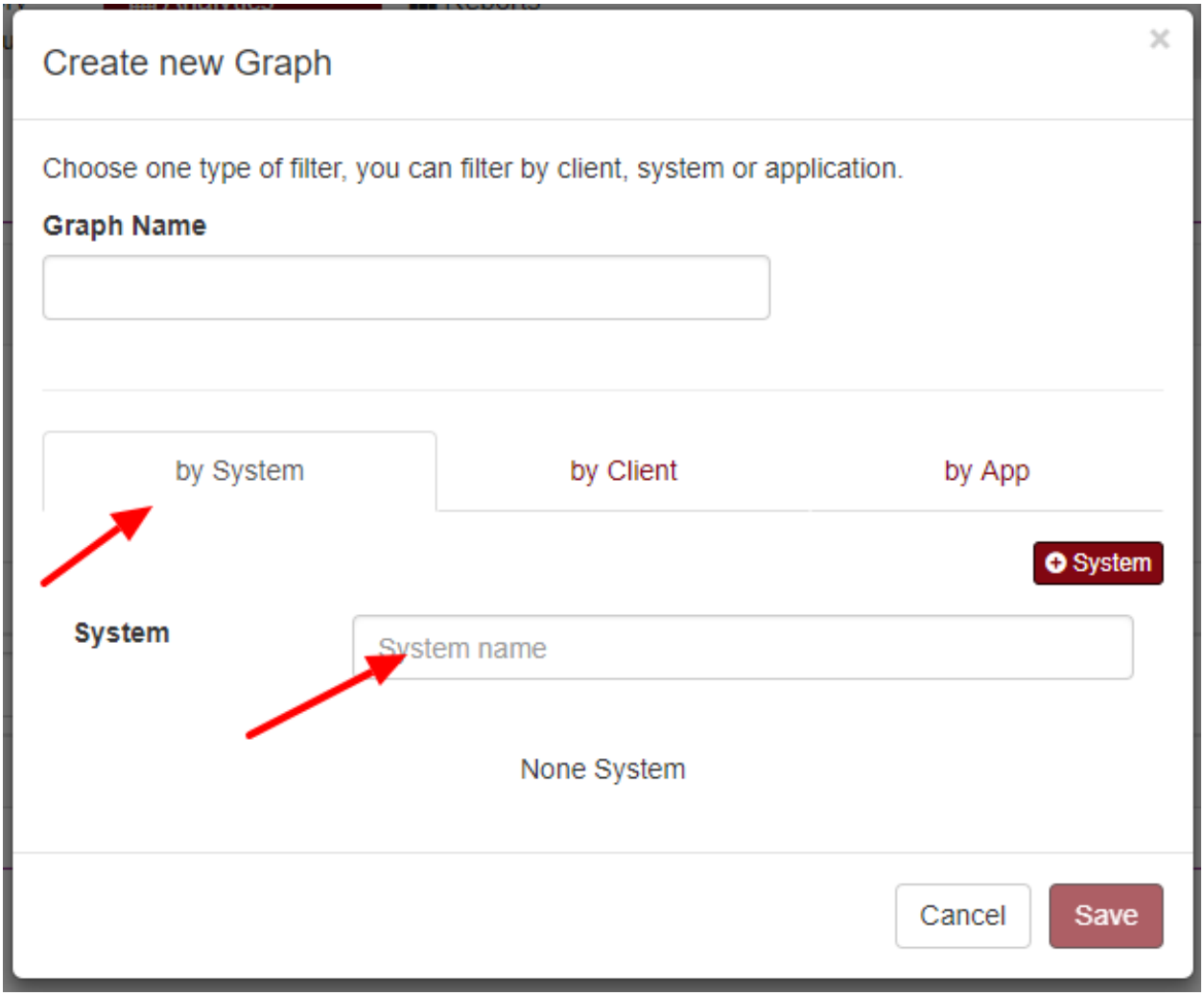
4.3.1 Business Graphs

You can create a diagram of your architecture, can be one or more systems/application. To create a diagram, Maestro uses the dependency field, the fast way to set connections between applications it using the dependency tree feature.

Go to Analytics > business Graph > New Graph



The first modal shows three options, you can start using a client, a system or an application.

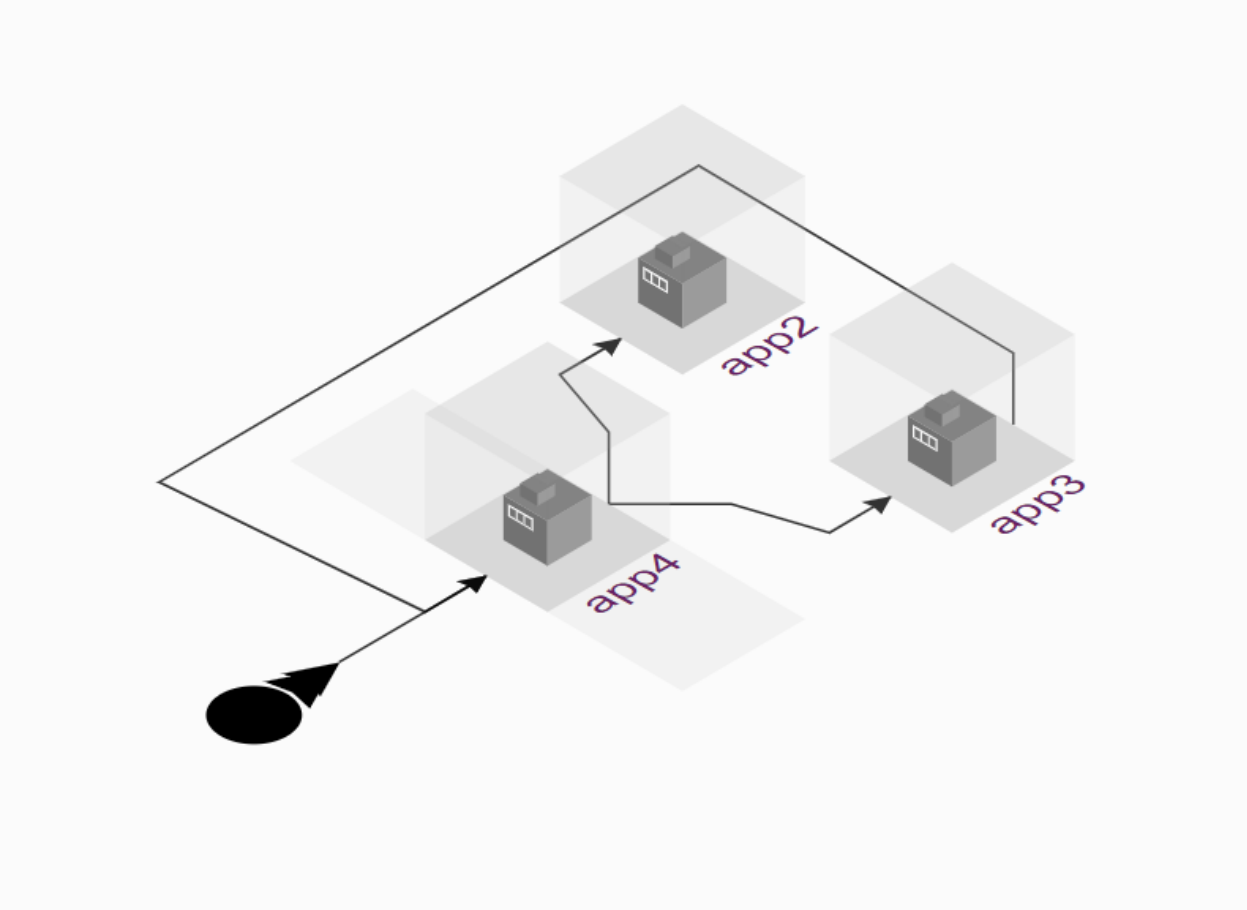


by System	It uses all entry applications set on those systems.
by Client	It uses all systems set on those clients.
by App	A entry given application

Entries applications

Entry applications are the diagram root branch, normally represents the first application hit by users, common categories are cdns, proxies, loadbalances and/or webservices.

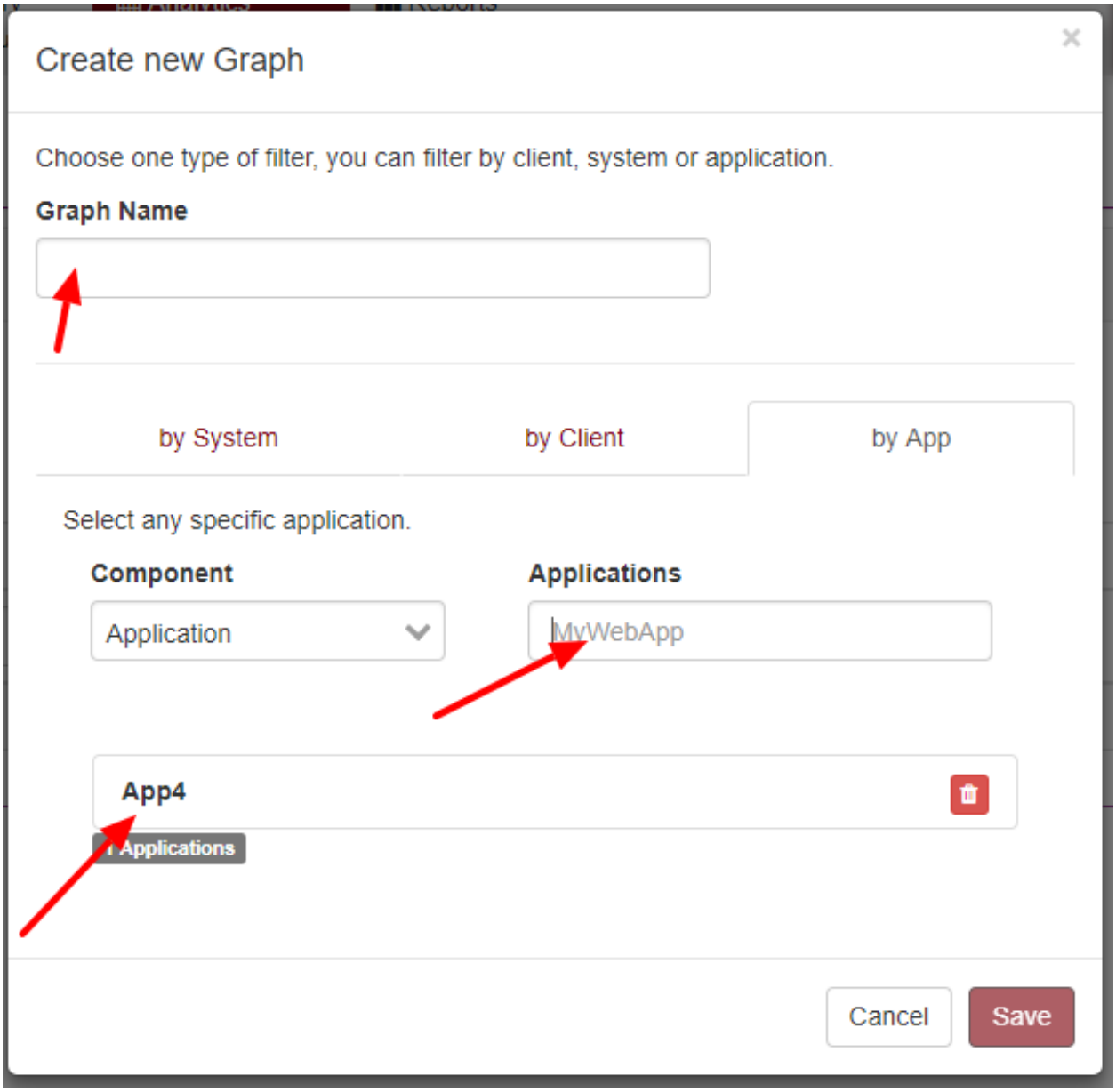
Using the dependency tree wizard.



In this example, app4 is the entry application.

Note: You can choose with applications can be used as an entry point on each system. (On entry app tab).

Creating a new diagram, selecting an entry application.



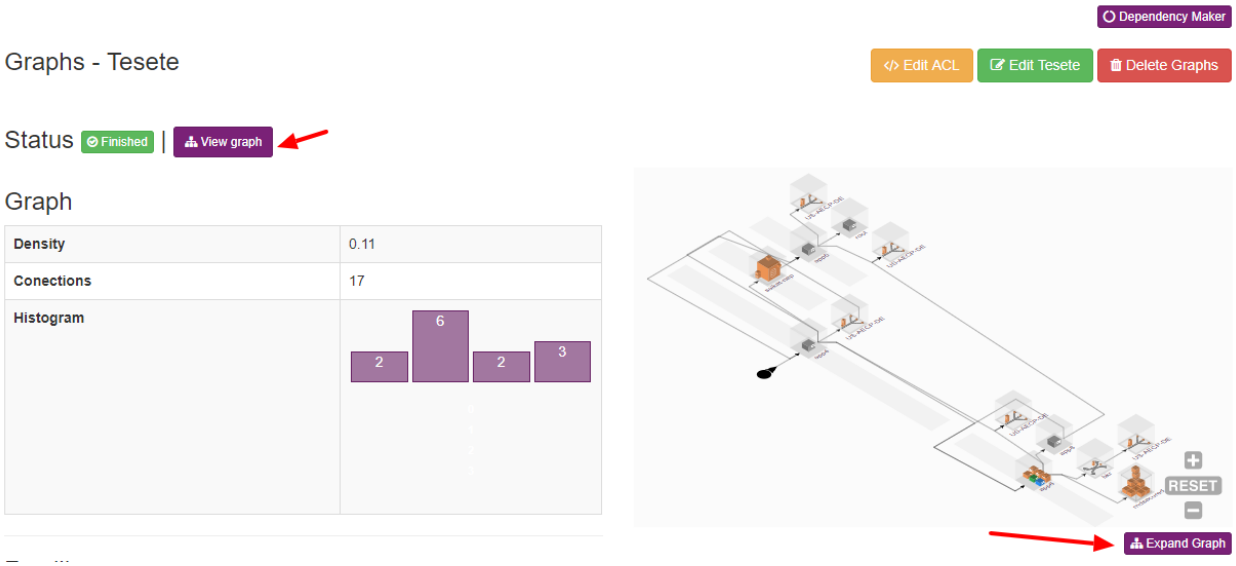
You can analyses density, total connections, histograms, accountant clients, systems and applications linked on that architecture.

- **Density** - The density for undirected graphs is [d = frac{m}{n(n-1)},] where (n) is the number of nodes and (m) is the number of edges in (G).

The density is 0 for a graph without any edges and 1 for a complete balance diagram. The density of multigraphs can be higher than 1.

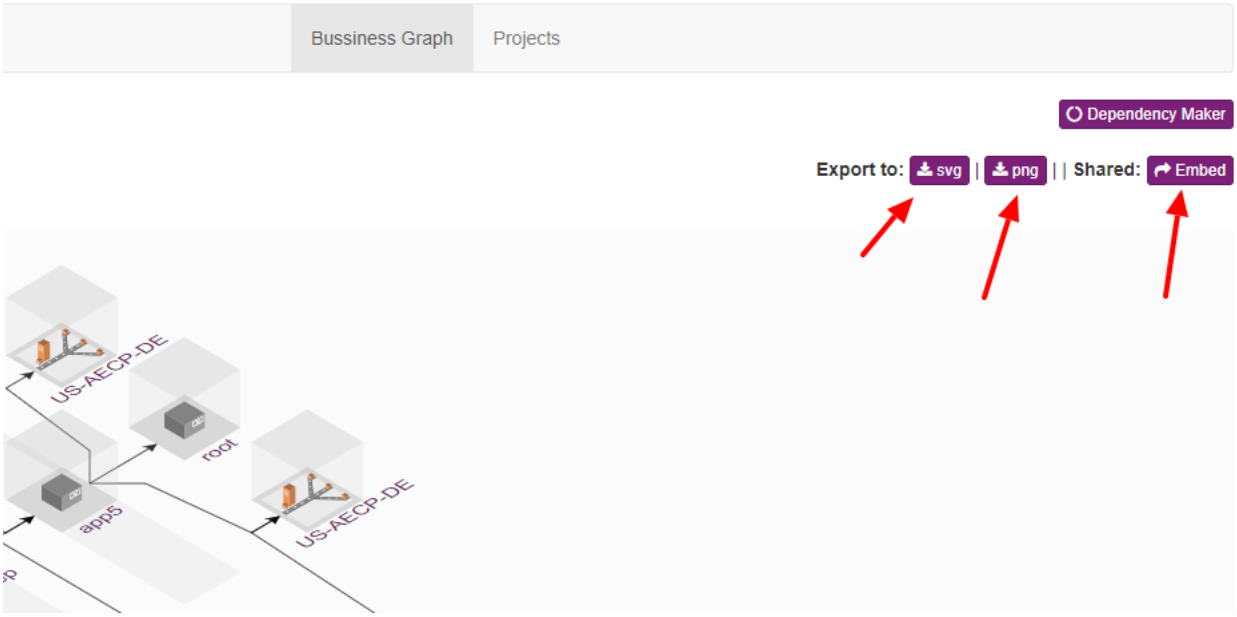
More detail - NetworkX Graph - Density.

- **Histogram** - Total by deep dependency.

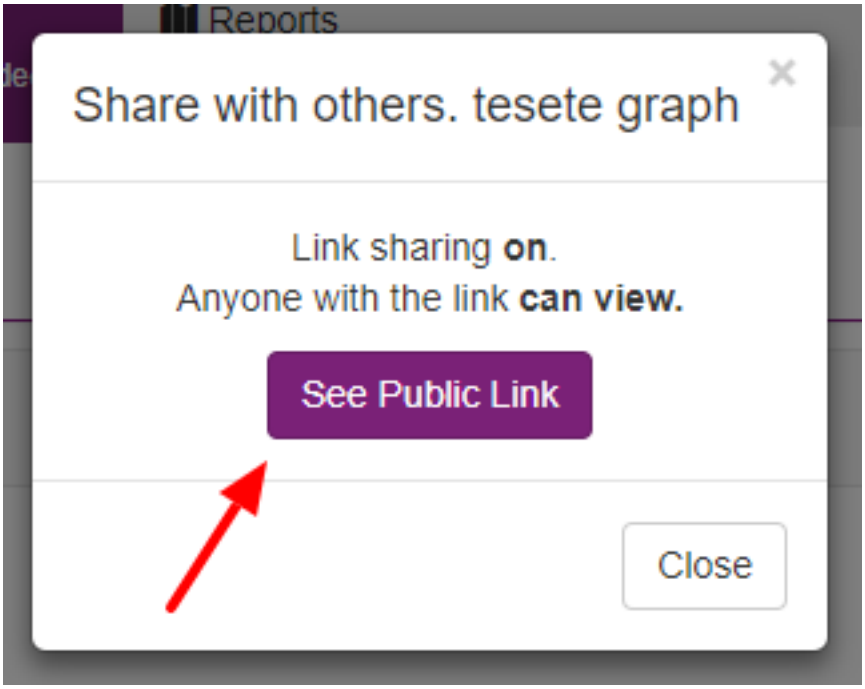


You can expand the diagram.

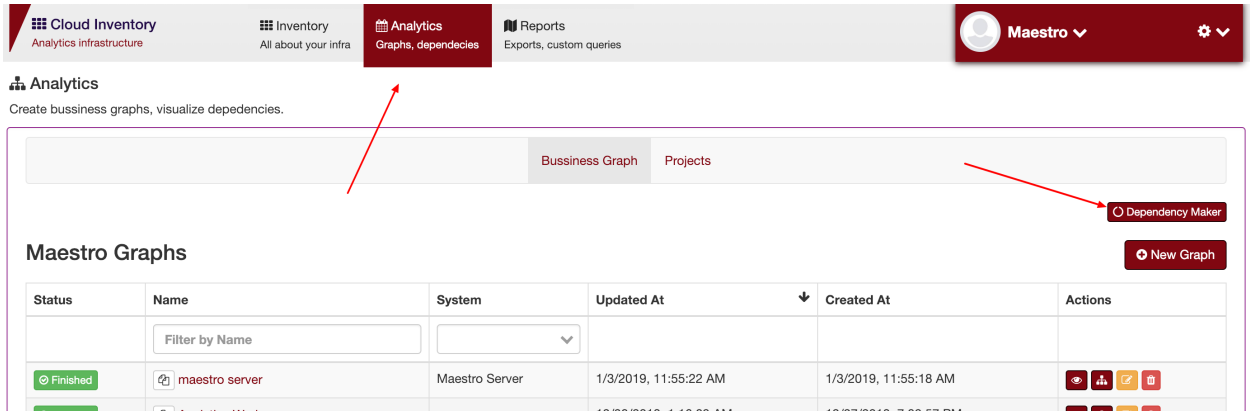
You can export the diagram in SVG, png or share that graph. Also, you can mouse over on lines to see each type of connection between each application.



On a shared page, you can click on “see a public link”, it will generate a shared link to embed on external tools, such as Confluence.

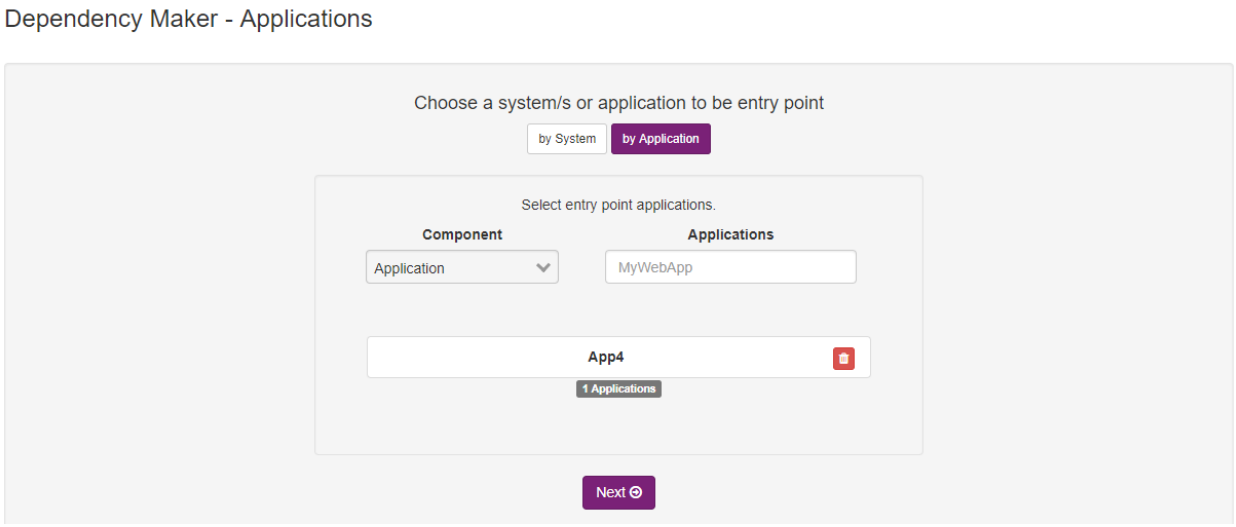


4.3.2 Using the dependency tree wizard

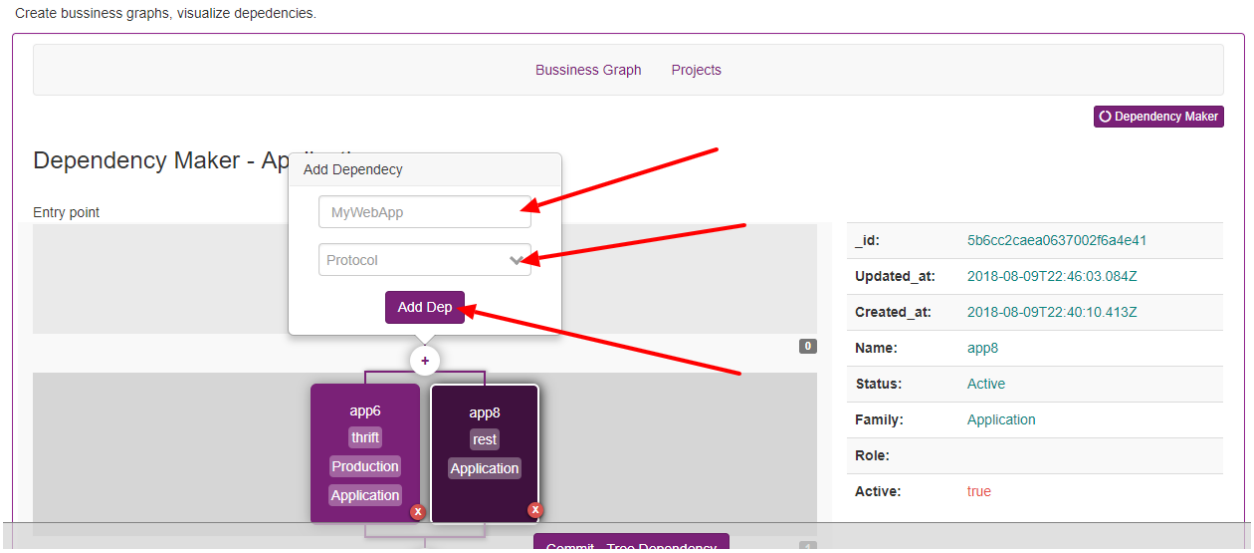


To create diagrams you need to link each applications using the dependency field. However, you can use the Dependency wizard, and this feature allows you to create and connect each application in a single and fast page.

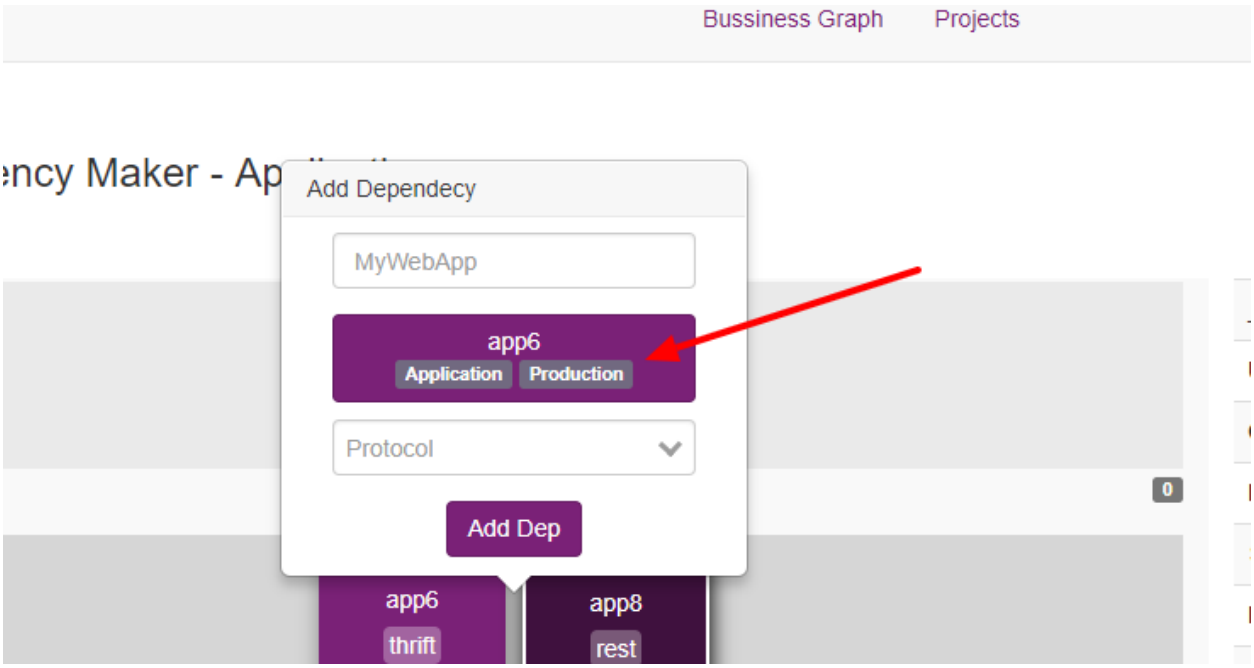
Go to dependency tree, and you can use an existed system, or a client or an application.



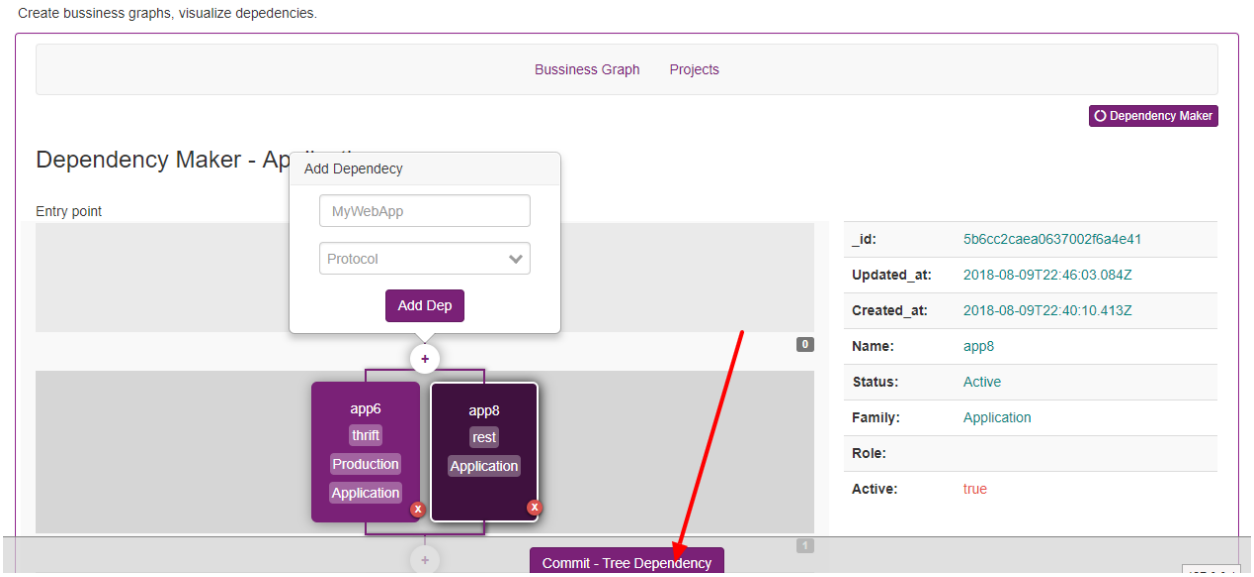
To connect in an application, you can click on plus button and select those applications; you can set the way those applications are connected, can be rest, grpc, tcp and etc.



Clicking in an app



To finish the diagram, click on commit. All done.

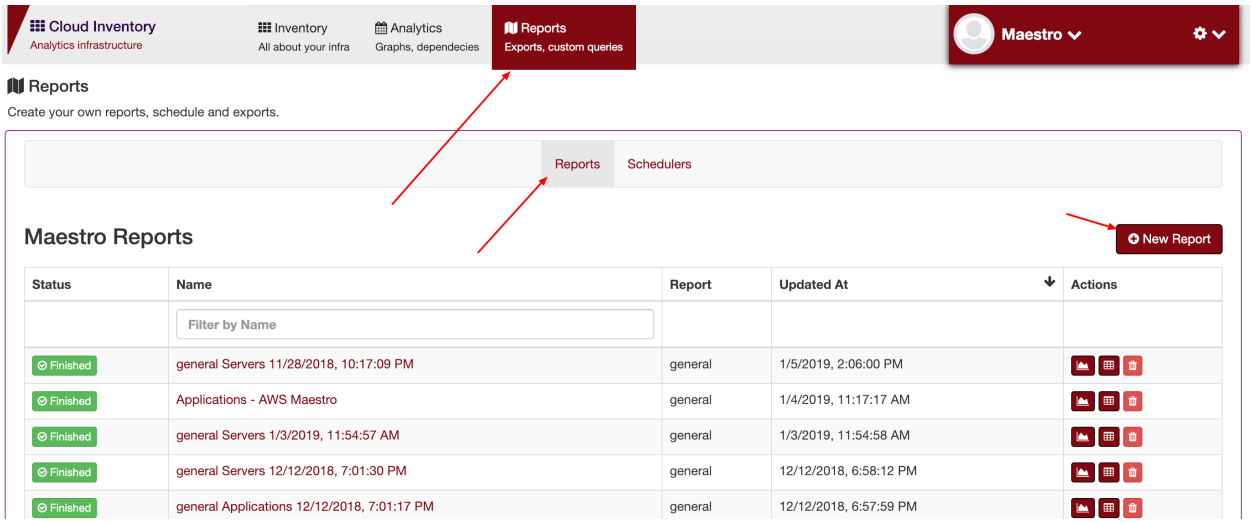


4.4 Reports - Generate advanced reports

4.4.1 Reports

Maestro has two types of reports.

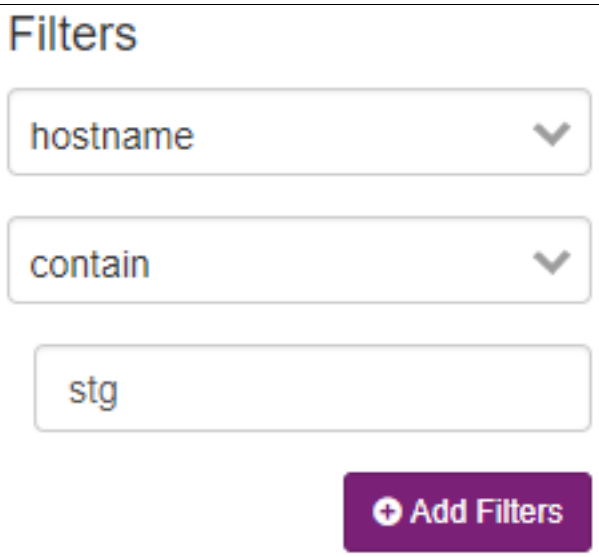
- **Generic:** it is a single resource, it can have any filter
- **Pivot:** It is a multi-resource, you can create a report link clients -> system -> applications -> servers.



Single table report

The general report is a single resource report, you can add any type of filters such as by datacenters, a name, a type, any field can be used as a filter.

Follow some filters examples:

Hostname/name	string	equal/contains	
Get all hostname contains stg.			

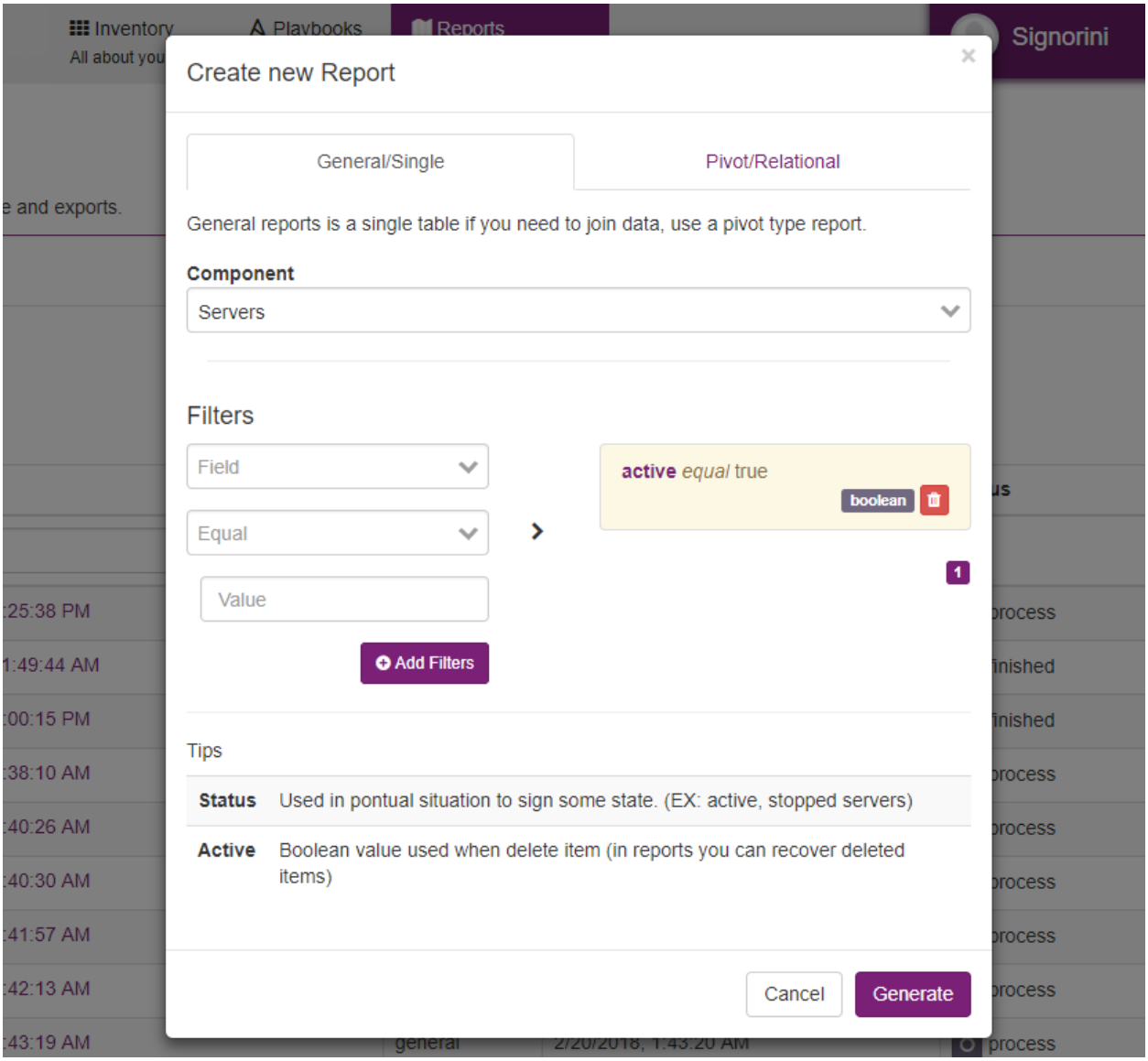
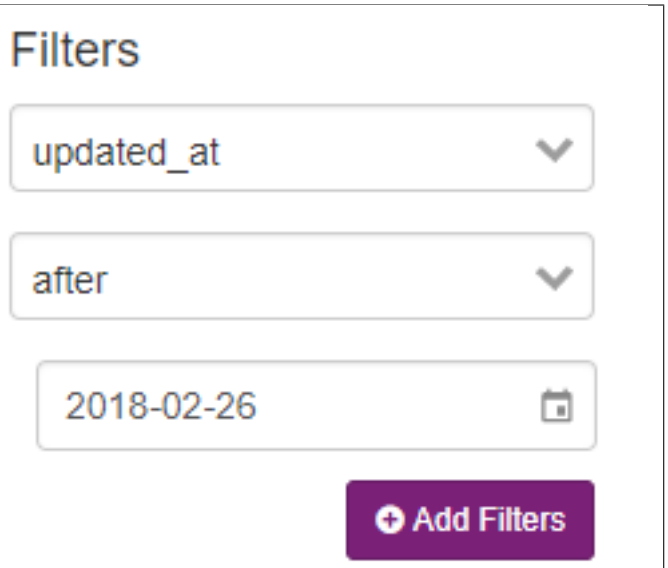
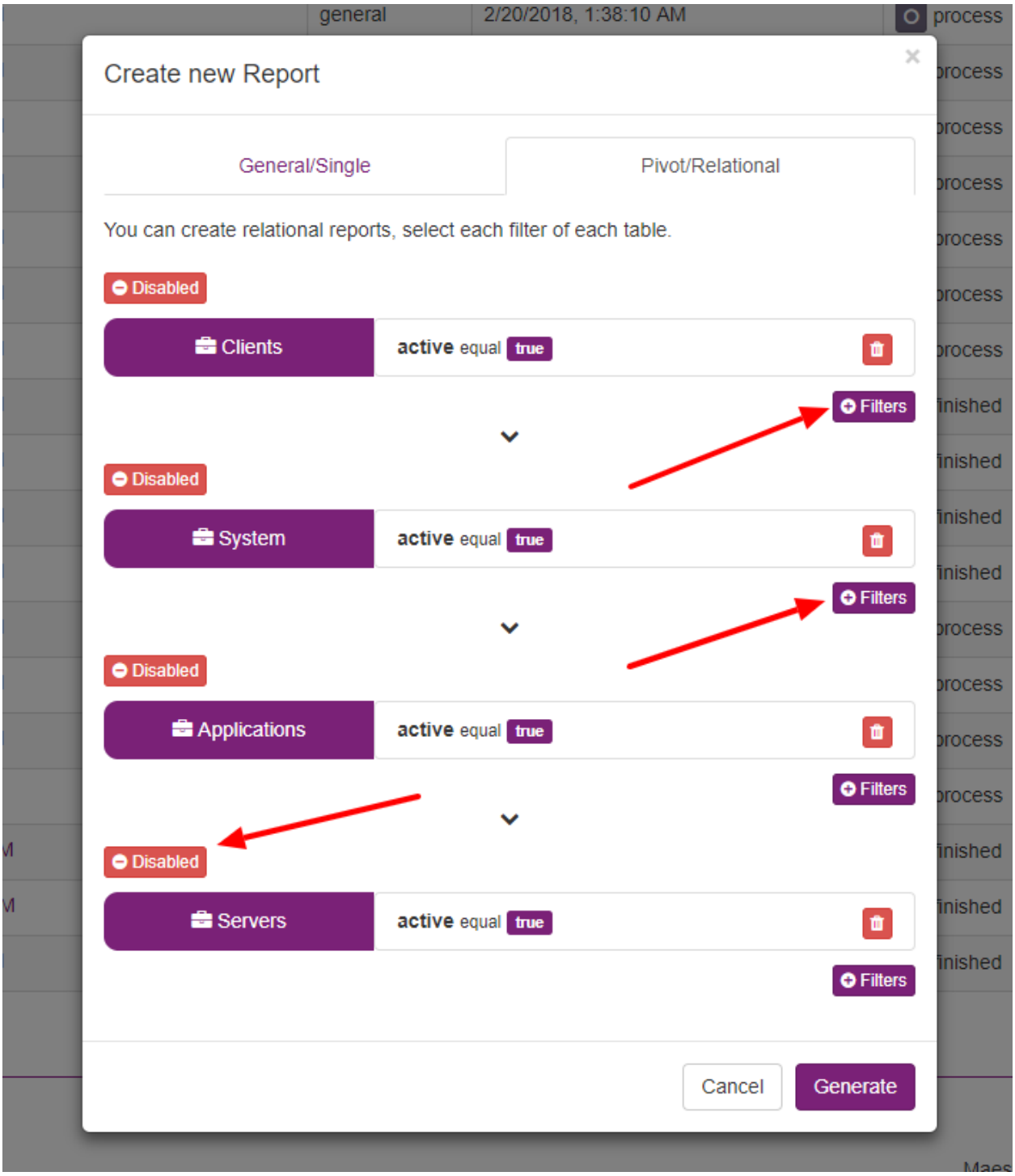


Fig. 2: Generic report

Updated_at	date	after/equal/before	
Select only items updated on this month			

Pivot table reports

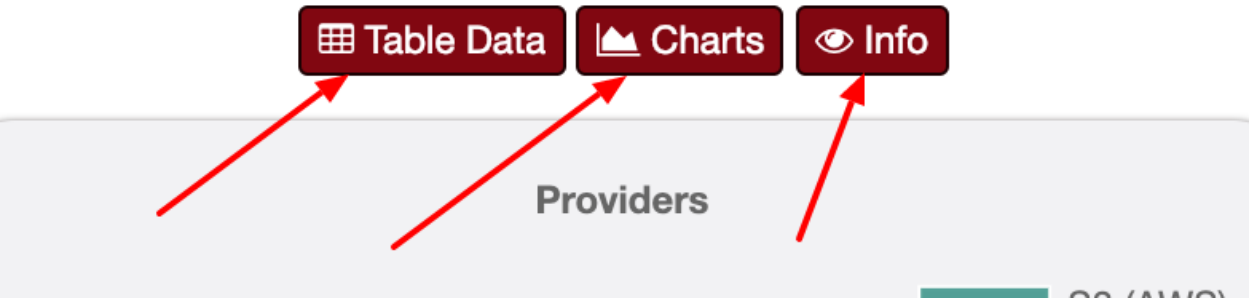
Pivot reports can create reports using multiple resources, and there are well-defined connections between each resource, the order is a client -> system -> app -> servers, you can remove one resource type. However, you need to have a link between them, for example, you can create a report with clients and systems, but can't to create a client -> servers.



Nesting resources.

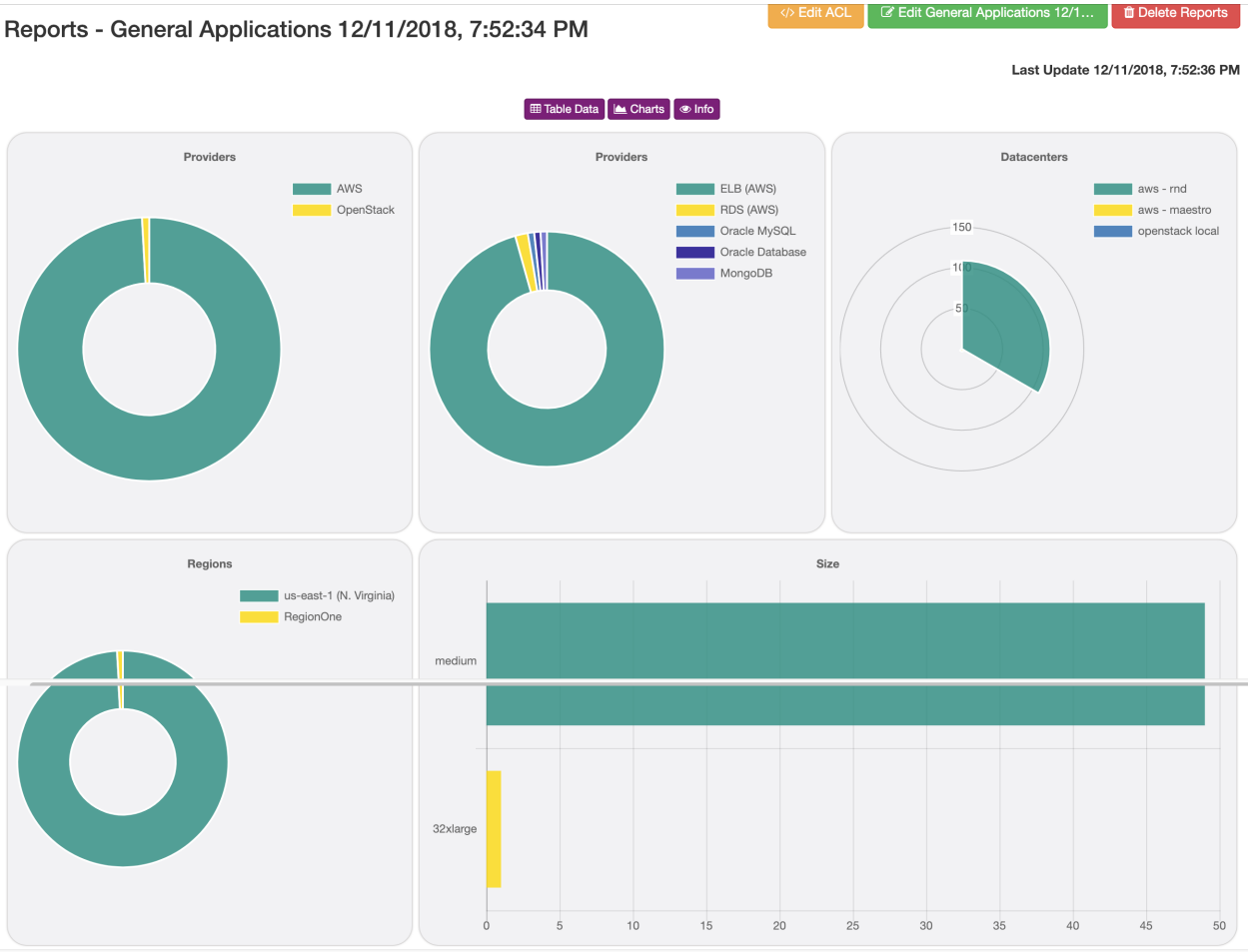
Each report has three pages

- **Charts:** Visualize the result on charts and diagrams.
- **Table:** Raw result table.
- **Info:** Information about the reports, such as status, filters and more.

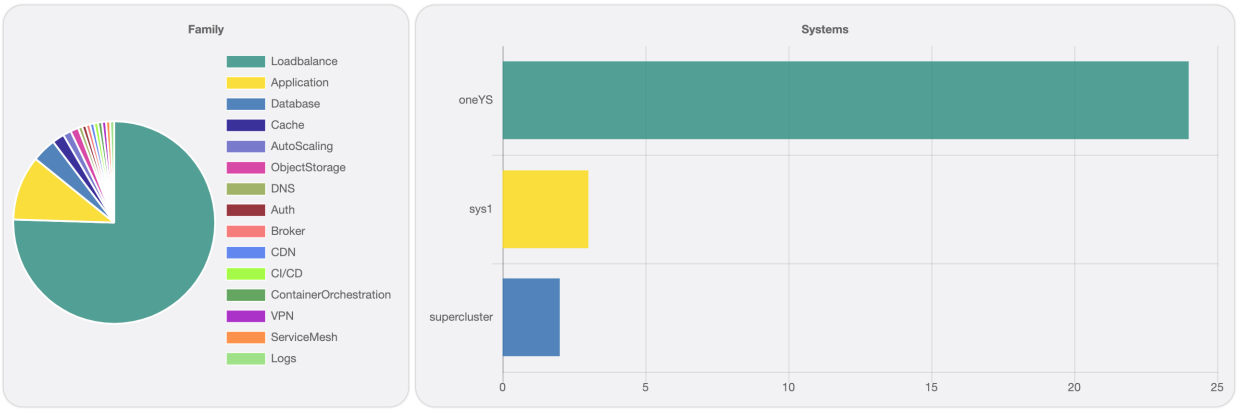


4.4.2 Report Charts

Reports > Single Report > Charts



Applications charts



Aggregate fields:

- Datacenter - Providers
- Datacenter - Resource
- Datacenter - Instance type
- Datacenter - Regions
- Datacenter - Zones
- Tags
- Sizes
- Application - Family
- Application - Dependencies
- Application - Deploys
- System by Application
- Clients by System
- System - Entry Applications

4.4.3 Scheduler

The scheduler is a time-based job scheduler, and it is responsible for managing and executing job cross Maestro, it used to synchronize the cloud providers data, to update reports and can be used by users.

To list all schedules, go to reports -> scheduler.

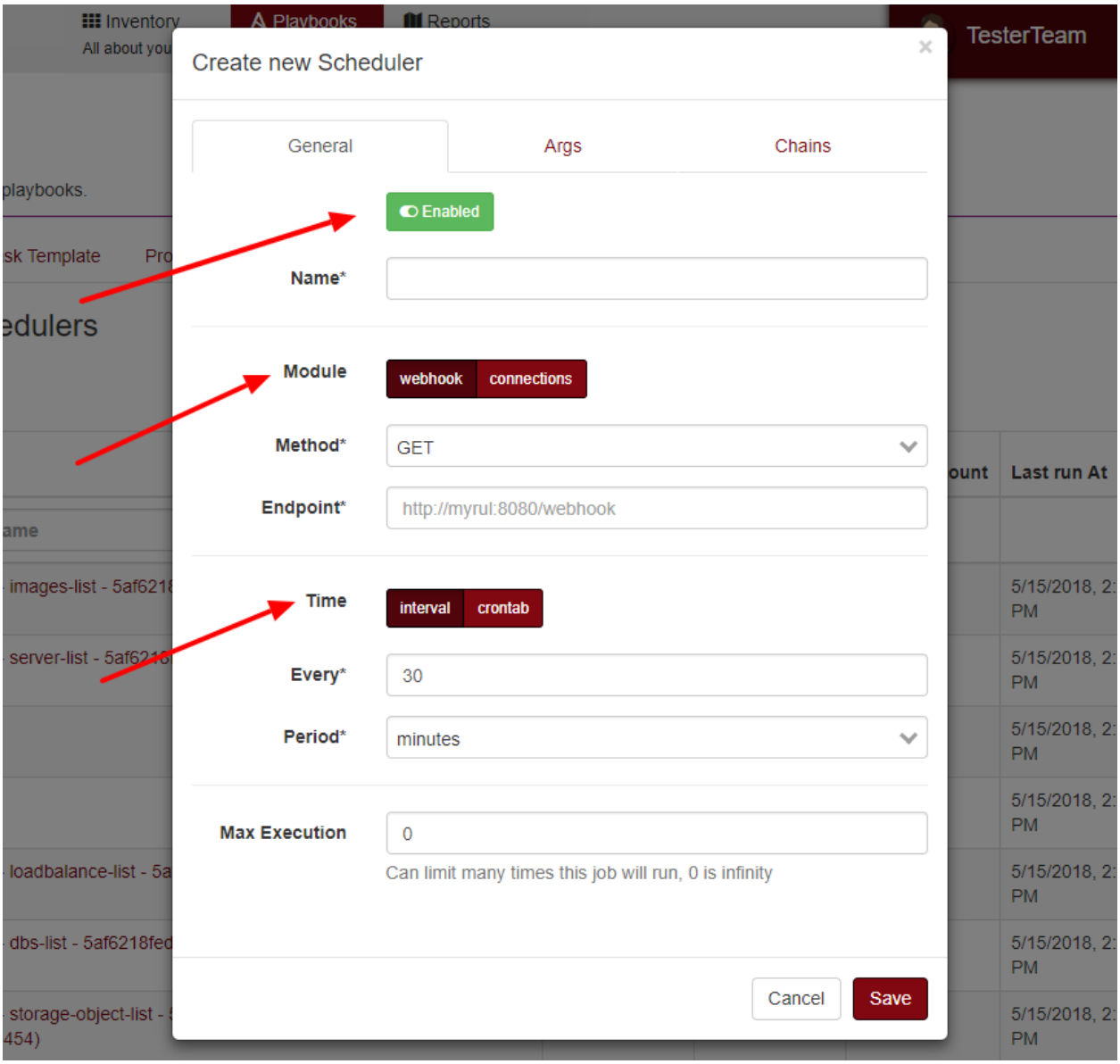
Enabled	Name	Modules	Period type	Total run count	Last run At	Actions
	Filter by Name					
	connections - images-list - 5af6218fedd1b90014ebf291 (1526403642643)	connections	interval	1	5/15/2018, 2:00:46 PM	
	connections - server-list - 5af6218fedd1b90014ebf291 (1526400982609)	connections	interval	27	5/15/2018, 2:23:09 PM	

As an example, we can see schedulers accountable to automatic sync a cloud provider data on Maestro.

	connections - server-list - 5af6218fedd1b90014ebf291 (1526400982609)	connections	interval	27	5/15/2018, 2:23:09 PM	
	localhost	webhook	interval	229	5/15/2018, 2:24:06 PM	
	discovery	webhook	interval	224	5/15/2018, 2:24:06 PM	
	connections - loadbalance-list - 5af6218fedd1b90014ebf291 (1526403636203)	connections	interval	1	5/15/2018, 2:00:40 PM	
	connections - dbs-list - 5af6218fedd1b90014ebf291 (1526403637338)	connections	interval	1	5/15/2018, 2:00:40 PM	

Creating a custom job.

You can create a custom job.



4.5 ACLs - Users and Teams

4.5.1 Access rules

The Maestro ACL is composed of multiple entity type and each entity has a one rule.

Entities can be:

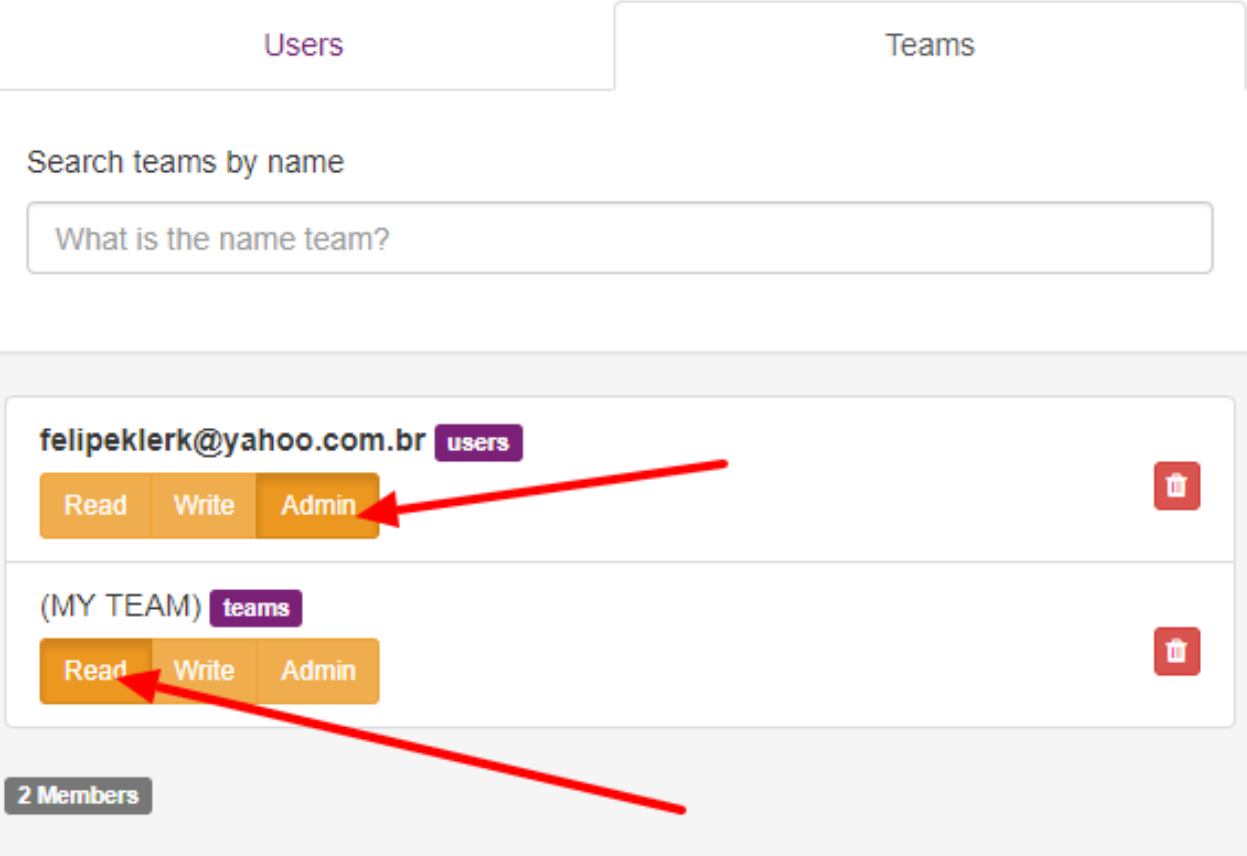
- a user
- a team

Rules can be:

Read:	Read access
Write:	Can read and update
Admin:	Can create and delete

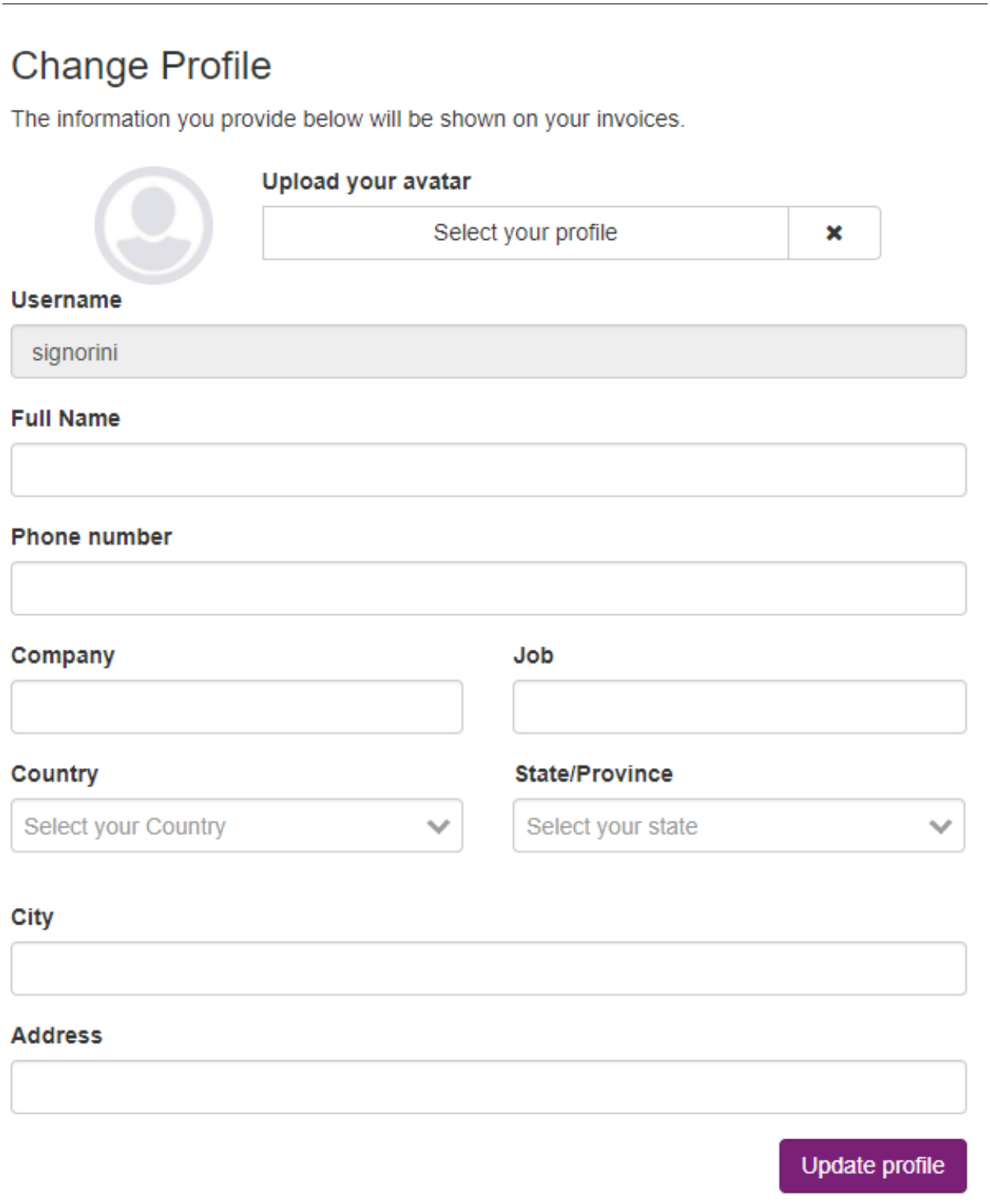
- The authentication control system is set at the resource level, that means each record has your own acl rule.
- You can create teams to share the same access to multiple users, and under the hood the user assume the team identity and then the team can access that record.

The ACL modal can be found on any resources such as servers, applications, graphs, reports and more.



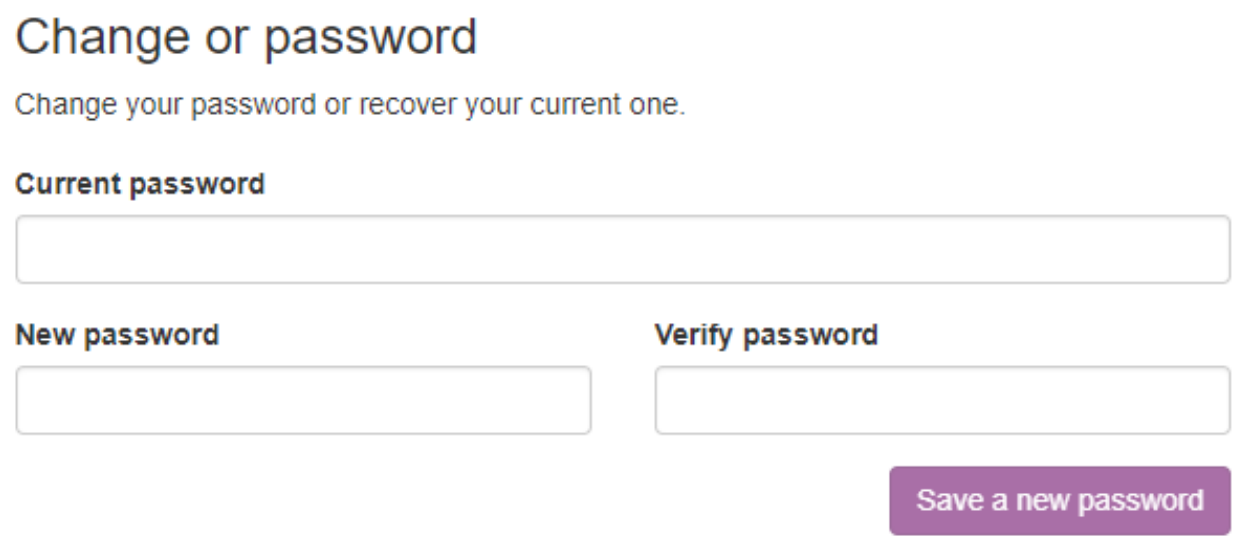
Users

You can update your profile.



Change password

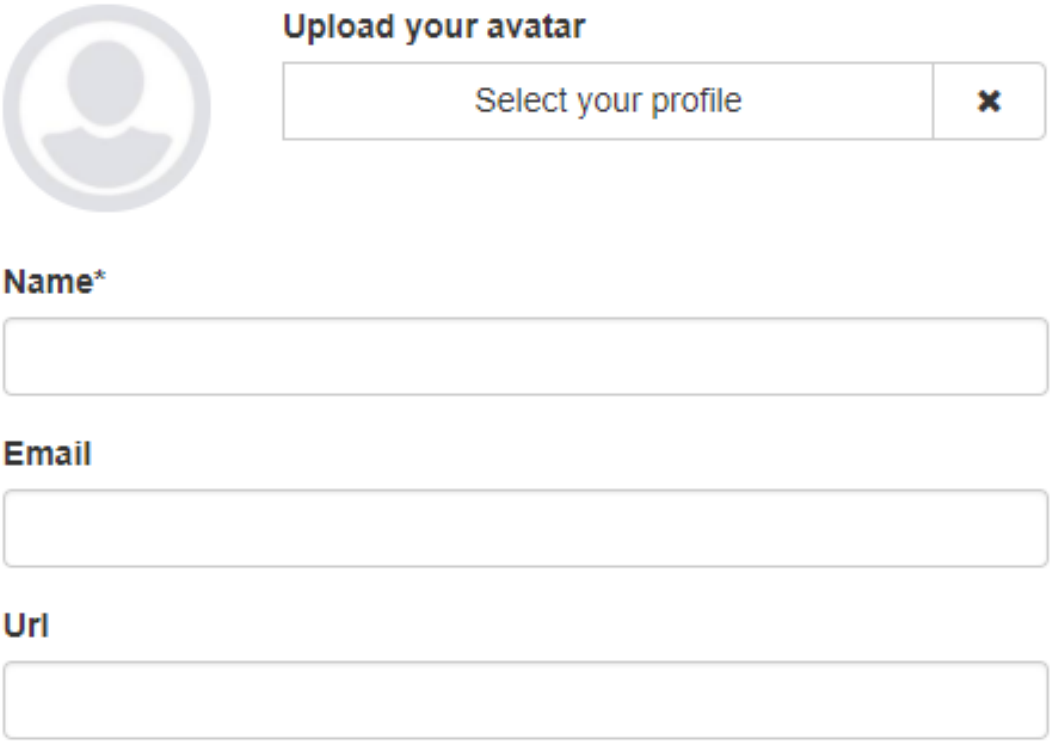
If you like to change the access password, you can go to profile > change password



4.5.2 Teams

To create a team, go to the main menu on the right corner, and click on the Teams page.

Each team has a name, email, avatar and members.


Upload your avatar
Select your profile
Name*
Email
Url

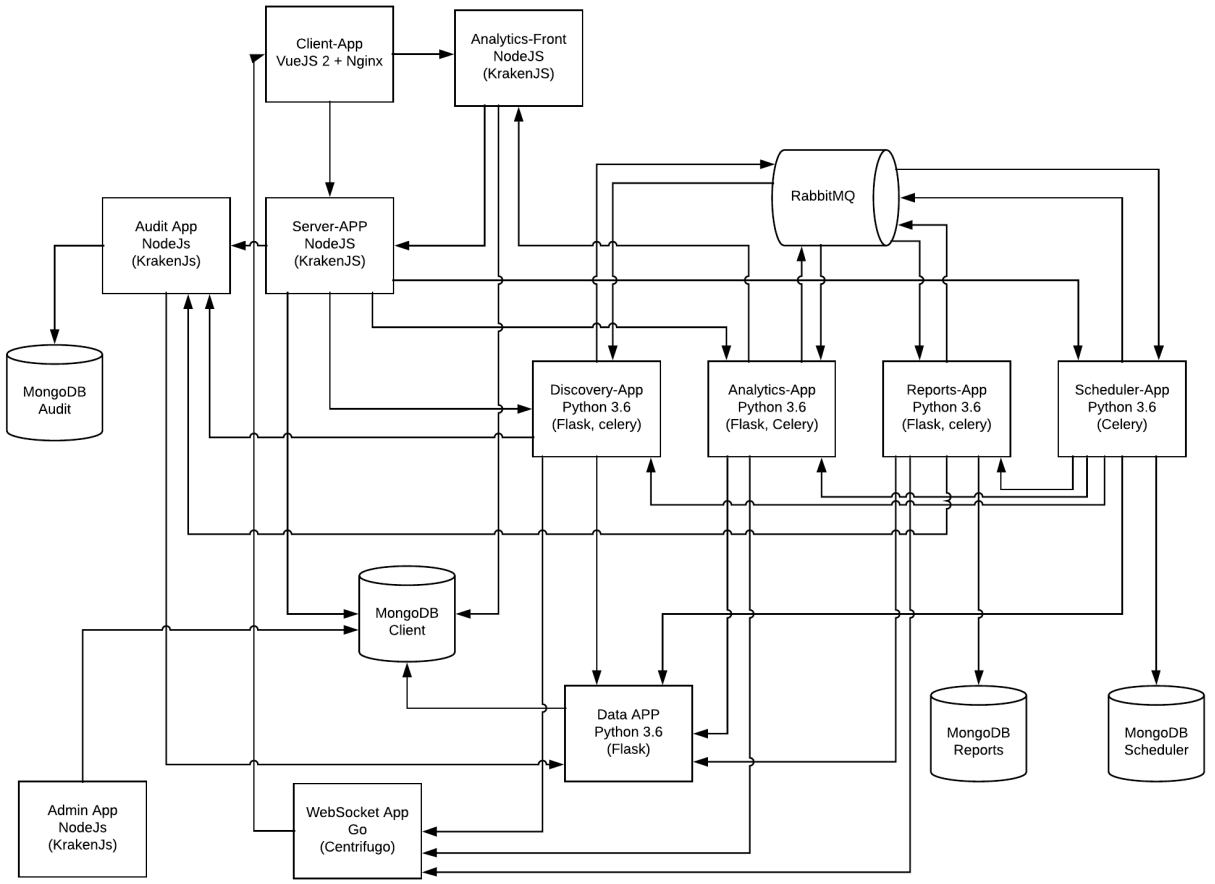
CHAPTER 5

Developer Guide

This chapter will explain a internal concepts about Maestro, if you like to contribute to the code this is the right place to start.

5.1 Architecture

This section describes advanced configurations, architecture and setups for developer. Maestro are organized by services made in nodejs and python, and they use mongodb as a datastore and rabbitmq as a broker, we build and deploy the application using docker.

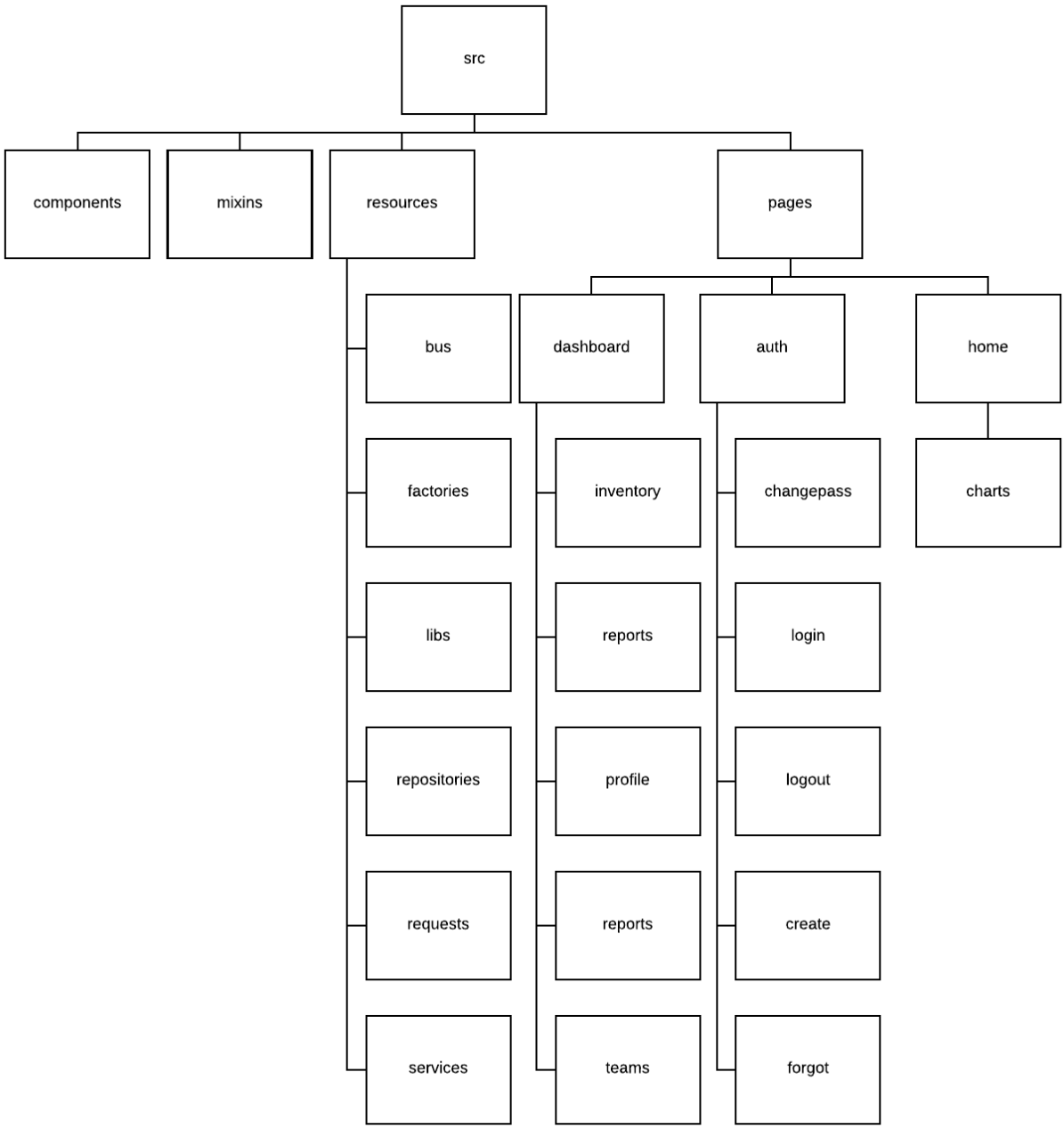


5.1.1 FrontEnd - Client App

The front end application, made using Vue2.

- Html and Js client
- Single page app (SPA)
- Cache layer

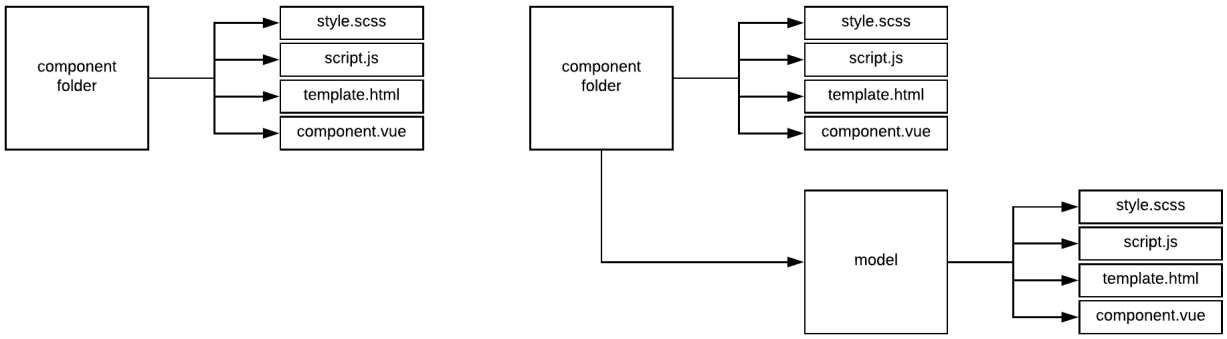
Vue2 Macro Architecture



Important topics

- Front end application are divided on:
  - **src/pages:** templates and business rules (domain layer)
  - **resources:** factories, modals, and cache managers (infrastructure layer)

A single component structure:



Installing node

- Nodejs >= 7.4

Download the repository

```
git clone https://github.com/maestro-server/client-app.git
```

Installing dependencies

```
npm install
```

Build

```
npm run build
```

Dev server

```
npm run serve
```

5.1.2 Server App

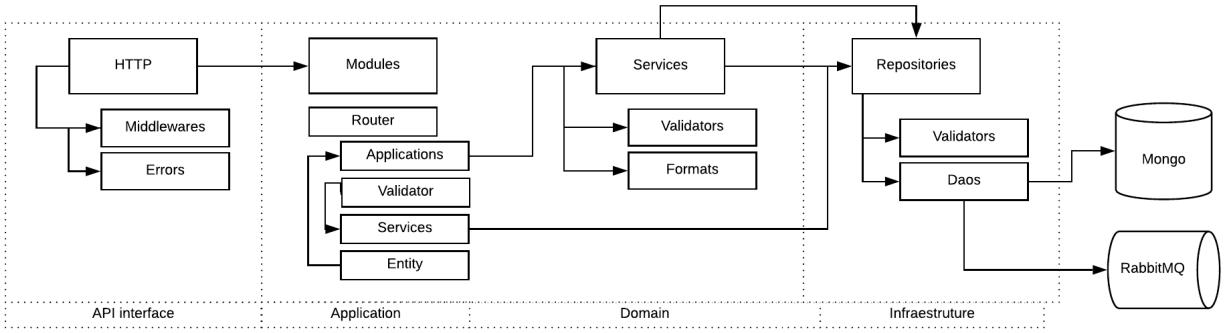
Server app is the main service; also they act as a middleware to authenticate and authorize users, it connect to the database and connect to others services.

- Authentication and authorization
- Validate and create entities (crud ops)
- Proxy to others services

> **Warning:** This service need to be expose externally

- Server is made with KrakenJs.

- We use DDD to organize the code, they have an infra, repositories, entities (values objects), interfaces, application, and domain folders. DDD in Node Apps



Setup dev env

```
cd devtool/

docker-compose up -d
```

It will run a mongodb and a fake stmp server

Installing node

- Nodejs >=8
- MongoDB
- Gcc + python (bcrypt package)

Download the repository

```
git clone https://github.com/maestro-server/server-app.git
```

Installing dependencies

```
cd server-app
npm install
```

Configure env variables

create .env file

```
SMTP_PORT=1025
SMTP_HOST=localhost
SMTP_SENDER='maestro@gmail.com'
SMTP_IGNORE=true

MAESTRO_PORT=8888
MAESTRO_MONGO_URI='localhost'
MAESTRO_MONGO_DATABASE='maestro-client'

MAESTRO_DISCOVERY_URI=http://localhost:5000 // list and get status connection
```

(continues on next page)

(continued from previous page)

```
MAESTRO_REPORT_URI=http://localhost:5005 // create and get reports data
MAESTRO_ANALYTICS_URI=http://analytics:5020 // create analytics report
MAESTRO_ANALYTICS_FRONT_URI=http://analytics_front:9999 // get analytics html
MAESTRO_AUDIT_URI=http://audit:10900 // notify audit update event and get history␣
↪track
```

and run the app

```
npm run server
```

Multiple env

Every config can be pass by env variables, but if you like, can be organize by .env files,

Name	Desc
.env	Default
.env.test	Used on run test
.env.development	node_env is set development
.env.production	node_env is set production

Database migration

Run the migration command.

```
npm run migrate

# to rollback the migration, run
npm run down_migration
```

We use PM2 to handle multiple threads, following the configuration.

PM2:

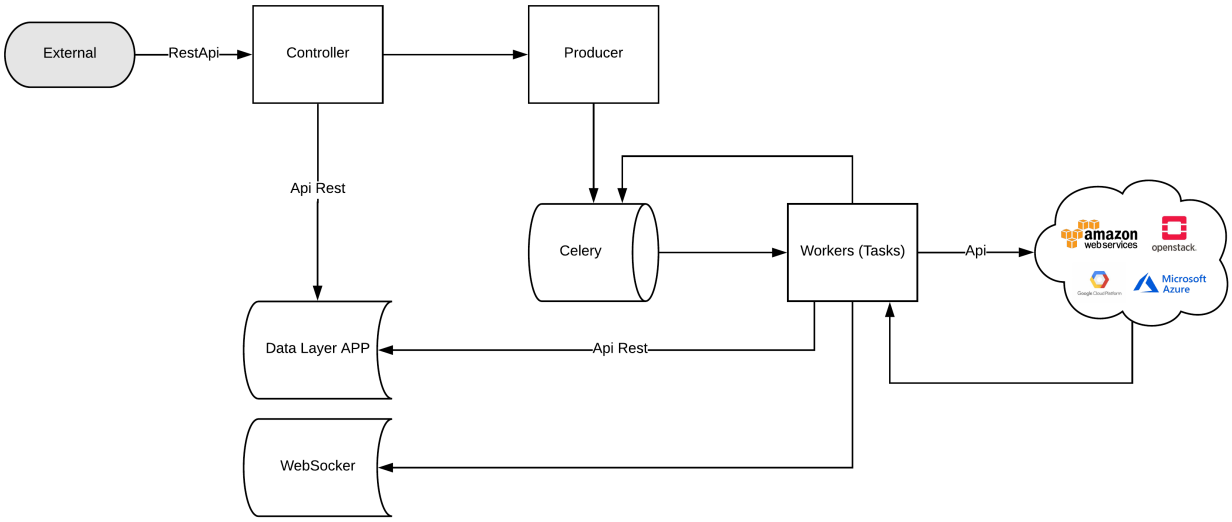
```
npm install -g pm2

# Create a file pm2.json

{
"apps": [{
    "name": "server-maestro",
    "script": "./server.js",
    "env": {
    "production": true,
    "PORT": 8888
    }
}]
}
```

```
pm2 start --json pm2.json
```

5.1.3 Discovery App

Discovery App is a crawler accountable to connect to cloud providers.

- To manager and authenticate on each cloud provider
- Translate cloud data to maestro data.

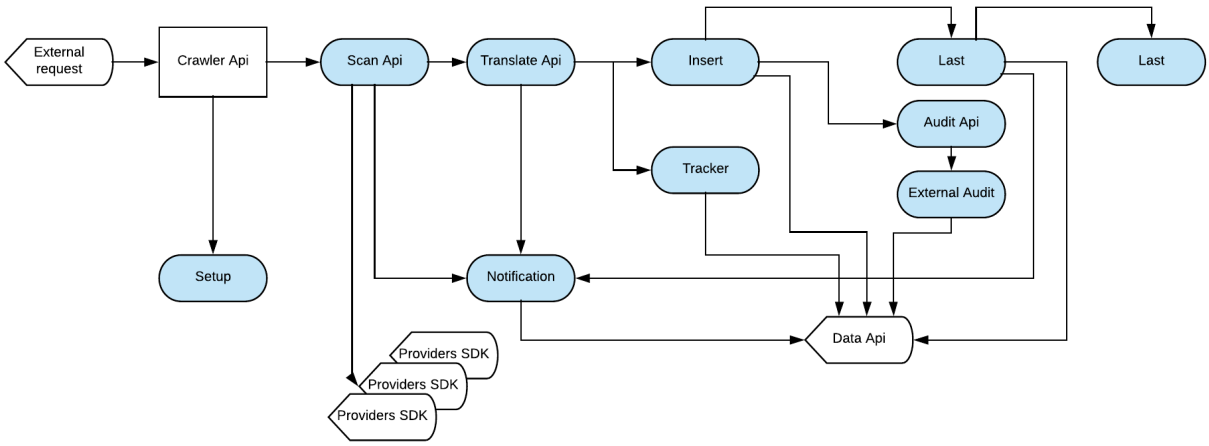


Discovery app use Flask, on python >3.5.

Setup dev env

```
cd devtool/

docker-compose up -d
```

Highlights

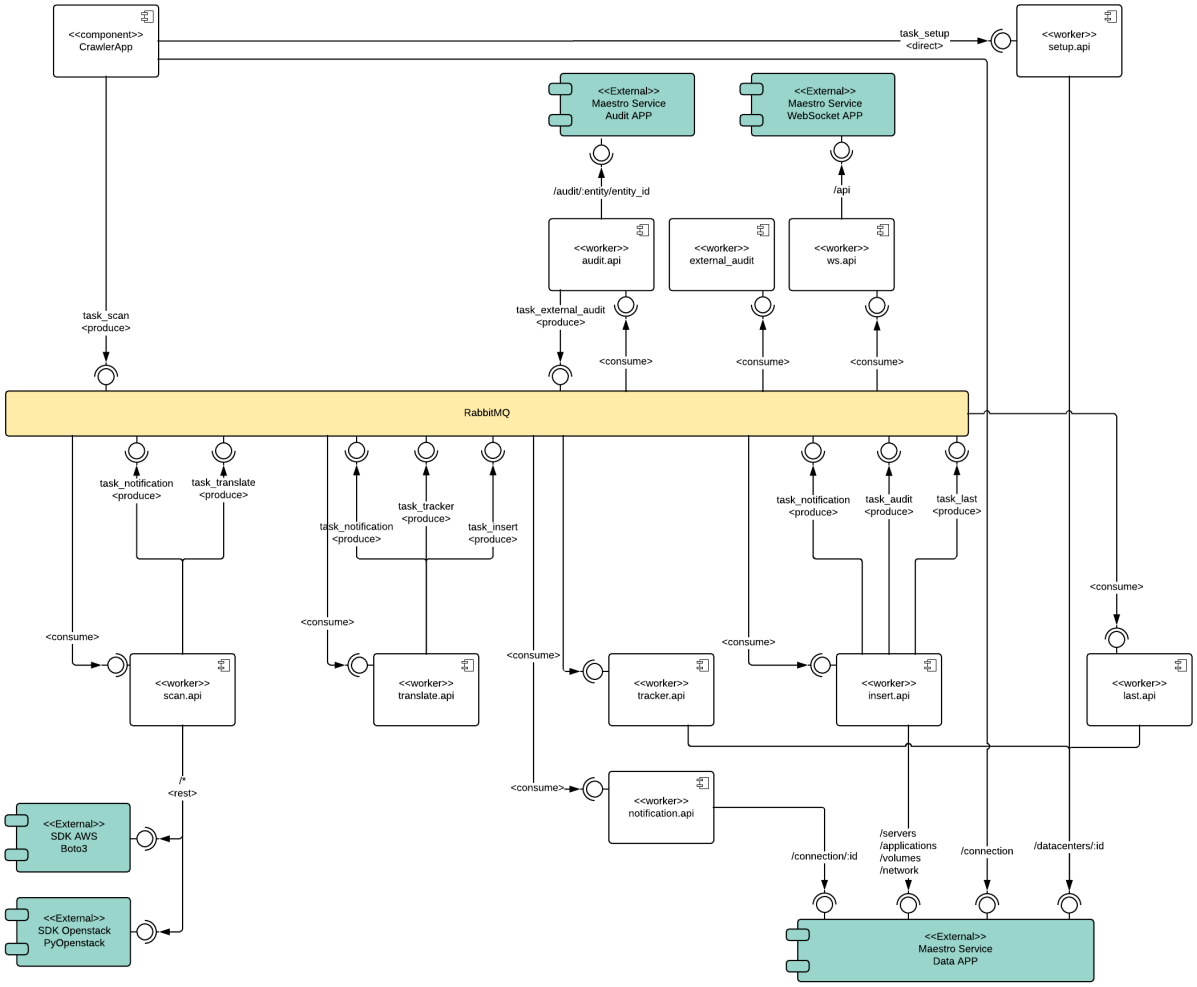


- The discovery are divided in modules:
  - **api:** To authenticate on cloud providers.
  - **translate:** Normalize the data.

- **setup:** Reset the tracker stats (it used on datacenters to get the orphans instances)
- **tracker:** recreate the tracker stats
- **insert:** insert/update data on mongodb
- **audit:** prepare and transform a data to send to the `external audit`
- **external_audit:** Send a http request to `Audit app`
- **ws:** Send a http notification to `websocket api`

Components Diagram

Follow an example of request flow.



Flower - Debug Celery

Real-time monitoring using Celery Events

- Task progress and history
- Ability to show task details (arguments, start time, runtime, and more)

- Graphs and statistics

```
pip install flower

flower -A app.celery

npm run flower
```

Installation with python 3

- Python >3.4
- RabbitMQ

Download the repository

```
git clone https://github.com/maestro-server/discovery-api.git
```

Installing dependencies

```
pip install -r requeriments.txt
```

Running

```
python -m flask run.py

or

FLASK_APP=run.py FLASK_DEBUG=1 flask run

or

npm run server
```

Running workers

```
celery -A app.celery worker -E -Q discovery --hostname=discovery@%h --loglevel=info

or

npm run celery
```

Warning: On production we use gunicorn to handle multiple threads.

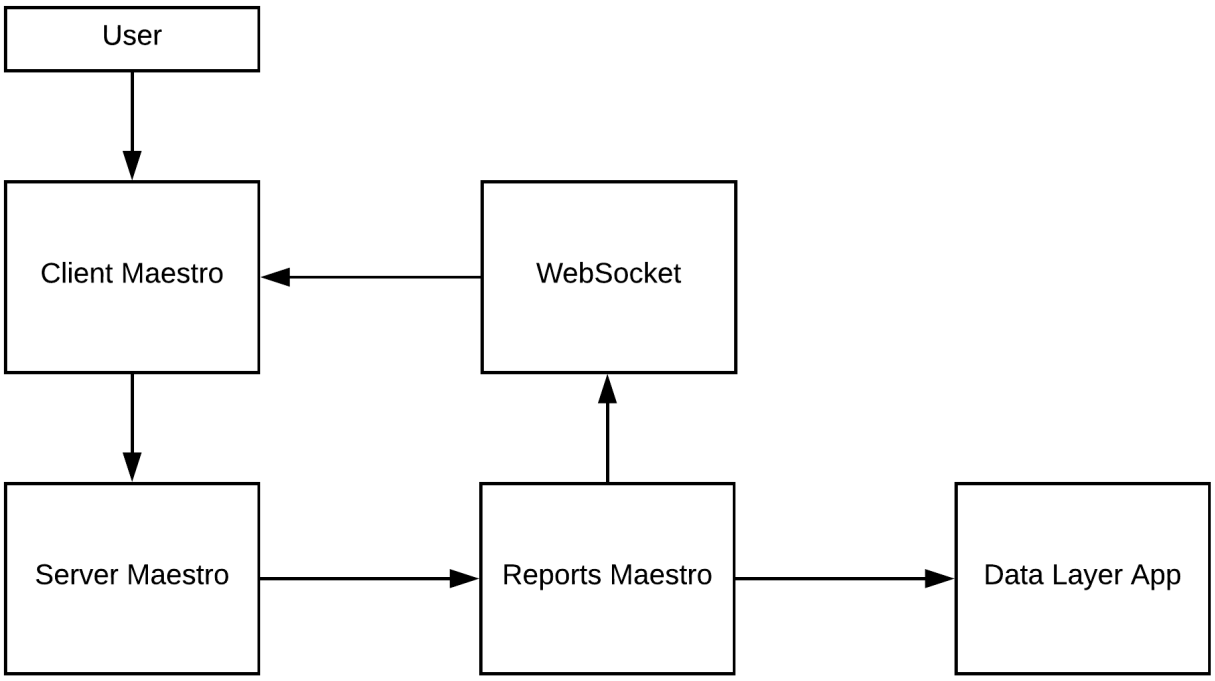
```
# gunicorn_config.py

import os

bind = "0.0.0.0:" + str(os.environ.get("MAESTRO_PORT", 5000))
workers = os.environ.get("MAESTRO_GWORKERS", 2)
```

5.1.4 Reports App

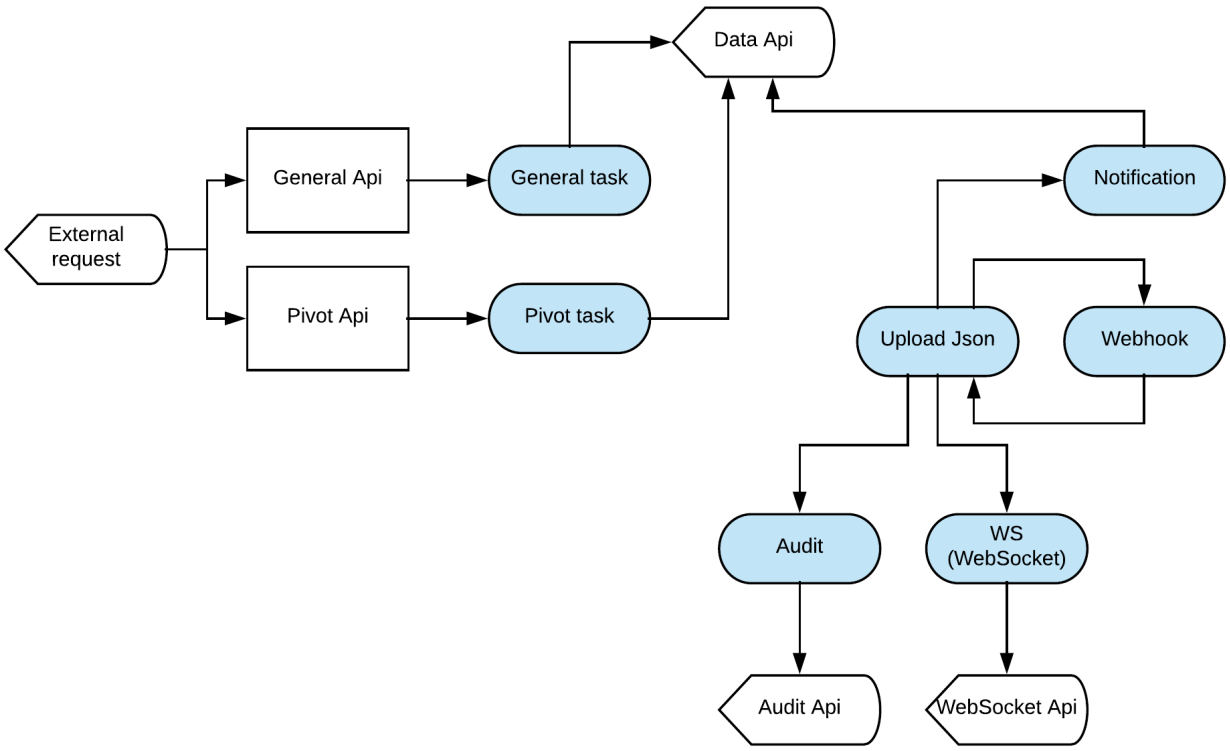
Application to aggregate, filter and generate reports.

- Parse complex queries and generate reports
- Manage storage and control each technical flow
- Transform reports on artifacts such as pdf, csv or json
- Save results on database

- Reports app use Flask, on python >3.5.



Highlights



- The module description:
  - **general/pivot:** get and filter data (communicate with discovery api)
  - **notification:** send a notification to data/audit services
  - **upload:** send results to the webhook
  - **webhook:** insert/update data on mongodb [report database]
  - **aggregation** - Execute aggregation tasks and save on report collections
  - **notify** - Send a notification to data app

Installation with python 3

- Python >3.4
- RabbitMQ
- MongoDB

Download the repository

```
git clone https://github.com/maestro-server/report-app.git
```

Running

```
python -m flask run.py --port 5005

or
```

(continues on next page)

(continued from previous page)

```
FLASK_APP=run.py FLASK_DEBUG=1 flask run --port 5005

or

npm run server
```

Running workers

```
celery -A app.celery worker -E -Q report --hostname=report@%h --loglevel=info

or

npm run celery
```

Warning: On production we use gunicorn to handle multiple threads.

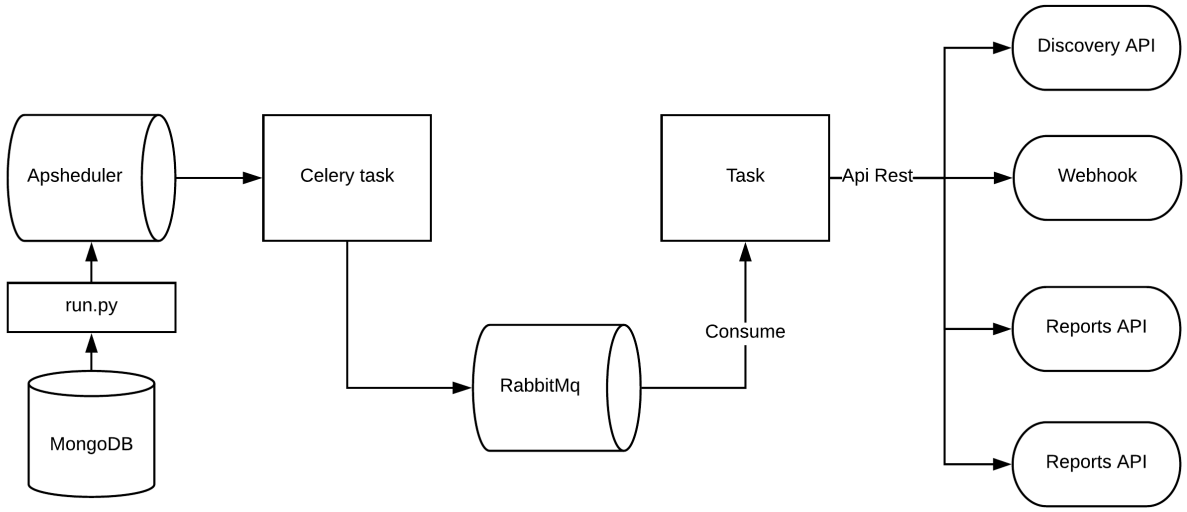
```
# gunicorn_config.py

import os

bind = "0.0.0.0:" + str(os.environ.get("MAESTRO_PORT", 5005))
workers = os.environ.get("MAESTRO_GWORKERS", 2)
```

5.1.5 Scheduler App

Scheduler App is accountable to manage and execute internal jobs.

- Schedule jobs, interval or crontab
- Do chain jobs

Scheduler use apscheduler to control scheduler jobs, Apscheduler documentation



Installation with python 3

- Python >3.4
- RabbitMQ
- MongoDB

Download the repository

```
git clone https://github.com/maestro-server/scheduler-app.git
```

Highlights

- Every 5 seconds the beat gets jobs on `schedulers collection` on mongodb.
- Beat can do:
  - **webhook:** Call HTTP request accordingly arguments.
  - **connection:** Sync a cloud data.
  - **report:** Generate/update a report.
- Support tasks.
  - **chain and chain_exec:** If this job have a chain job this tasks will do it.
  - **depleted_job:** Error handler to get any error and take the job out.
  - **notify_event:** Send a notification.

Installation with python 3

- Python >3.4
- RabbitMQ
- MongoDB

Download the repository

```
git clone https://github.com/maestro-server/scheduler-app.git
```

Running scheduler beat

```
npm run beat
```

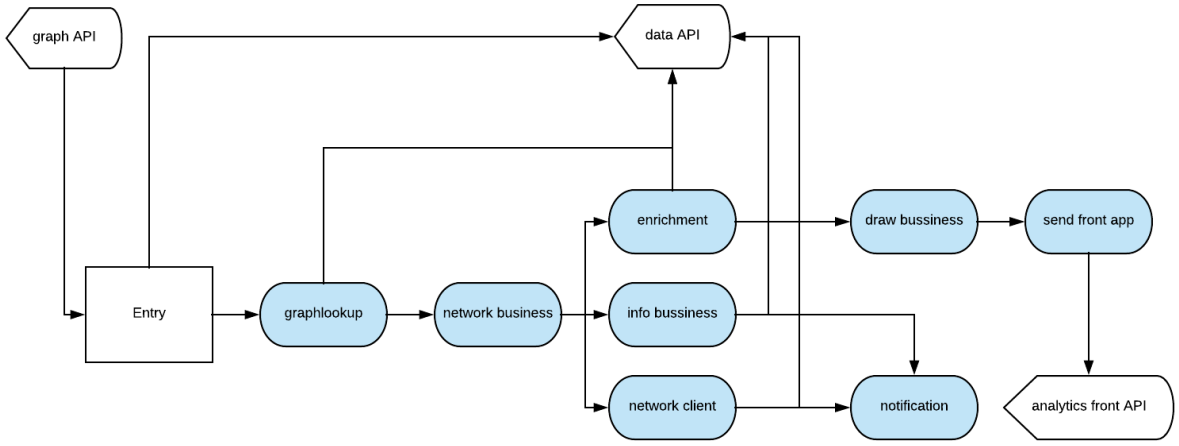
Running workers

```
celery -A app.celery worker -E --hostname=scheduler@%h --loglevel=info

or

npm run celery
```

5.1.6 Analytics Maestro

Accountant to get and create a application dependency tree and build diagrams:

- Create business graphs
- Drawing diagrams



Analytics app use Flask, on python >3.5.

Setup dev env

```
cd devtool/

docker-compose up -d
```

It will be set a rabbitmq and a redis

Highlights

- The diagram lookup and draw process are compound by:
  - **entry:** The first task, they get all entries application and send to graphlookup.
  - **graphlookup:** Requesting the db data over `Data App`, doing an application lookup using a MongoDB $graphLookup feature.
  - **network business:** Do a grid tree, and then send to `enrichment task` and `info task`.
  - **enrichment:** Getting servers.
  - **info business:** Calculate histogram, counts, density and connections.
  - **network client:** Getting clients.
  - **draw business:** Draw svgs.
  - **notification:** Send updates to `Data App`.
  - **send front app:** Send the svg to `Analytics Front app`.

Flower - Debug Celery

Real-time monitoring using Celery Events

- Task progress and history
- Ability to show task details (arguments, start time, runtime, and more)
- Graphs and statistics

```
pip install flower

flower -A app.celery

npm run flower
```

Installation guide

- Python >3.4
- RabbitMQ

Download the repository

```
git clone https://github.com/maestro-server/discovery-api.git
```

Installing dependencies

```
pip install -r requeriments.txt
```

Running

```
python -m flask run.py

or

FLASK_APP=run.py FLASK_DEBUG=1 flask run

or

npm run server
```

Running workers

```
celery -A app.celery worker -E -Q analytics --loglevel=info

or

npm run celery
```

Warning: On production we use gunicorn to handle multiple threads.

```
# gunicorn_config.py

import os

bind = "0.0.0.0:" + str(os.environ.get("MAESTRO_PORT", 5020))
workers = os.environ.get("MAESTRO_GWORKERS", 2)
```

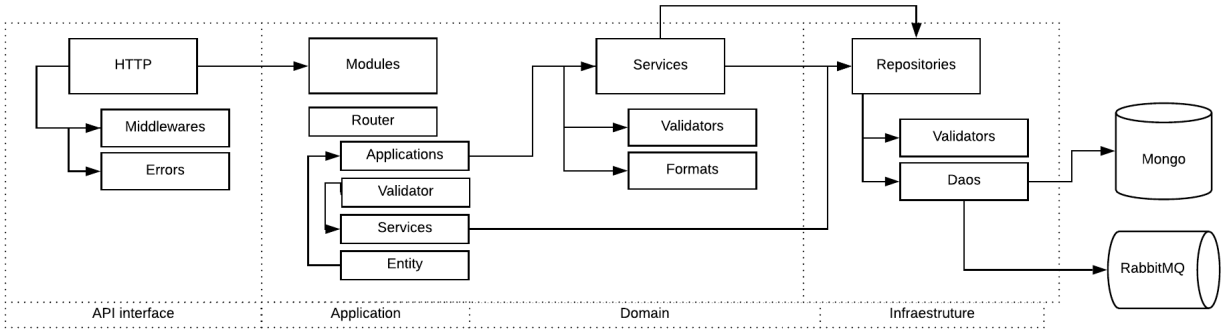
5.1.7 Analytics Front

Analytics Front Application is accountable to expose diagrams to the user:

- Public/private authorization
- Expose svgs diagrams
- Upload private SVGs

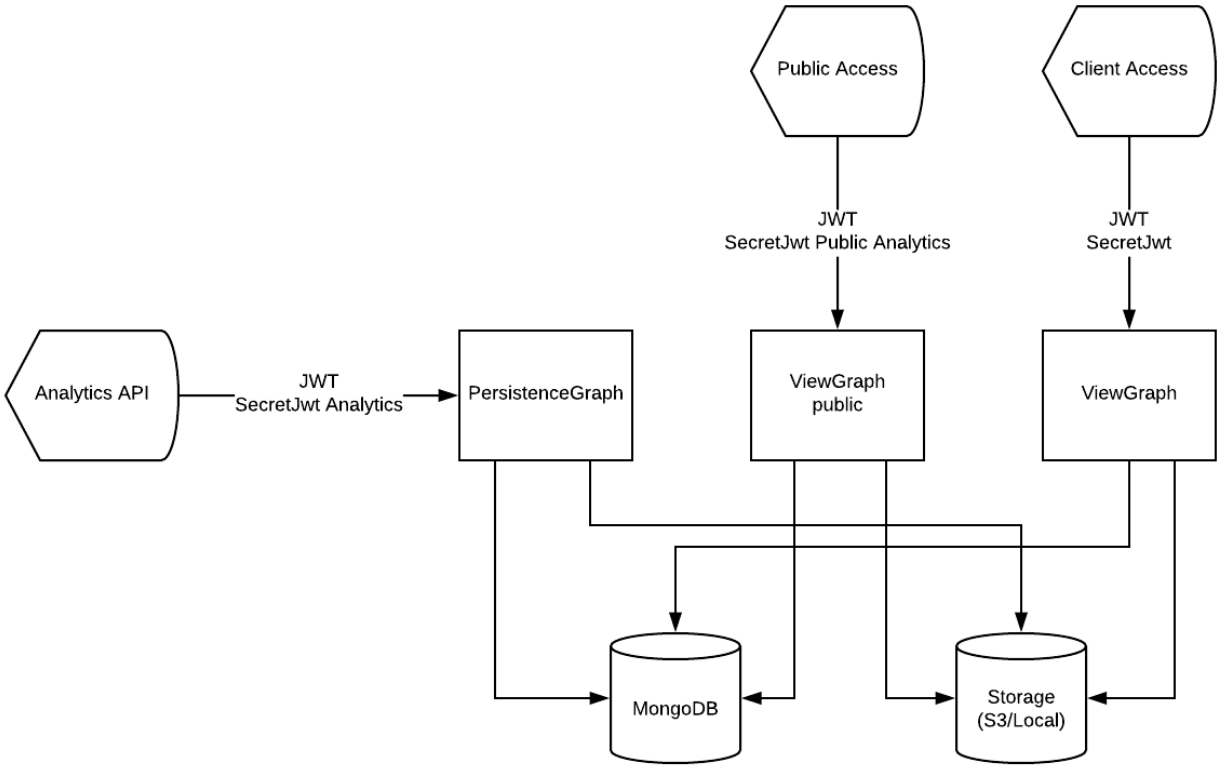
Warning: This service need to expose an external access

We use DDD approach to organize a code, they have an infra, repositories, entities (values objects), interfaces, application, and domain folders. DDD in Node Apps



Analytics is made with KrakenJs.

Follow a module flow diagram:



Installing node

- Nodejs >=8
- MongoDB >=3.4
- RabbitMQ
- AWS S3 (To use as a external storage)

To Download the repository, go to:

```
git clone https://github.com/maestro-server/analytics-front.git
```

Installing dependencies

```
cd analytics-front
npm install
```

Configure env variables

create .env file

```
MAESTRO_PORT=9999
MAESTRO_MONGO_URI='localhost'
MAESTRO_MONGO_DATABASE='maestro-client'
```

and

```
npm run server
```

Multiple env

Every config can be pass by env variables, but if you like, can be organize by .env files,

Name	Desc
.env	Default
.env.test	Used on run test
.env.development	node_env is set development
.env.production	node_env is set production

Migrate setup data

create .env file

```
npm run migrate
```

We use PM2 to handle multiple threads, following the configuration.

PM2:

```
npm install -g pm2

# Create a file pm2.json

{
"apps": [{
    "name": "analytics-front",
    "script": "./server.js",
    "env": {
        "production": true,
        "NODE_ENV": "production",
        "PORT": 9999
    }
}]
}
```

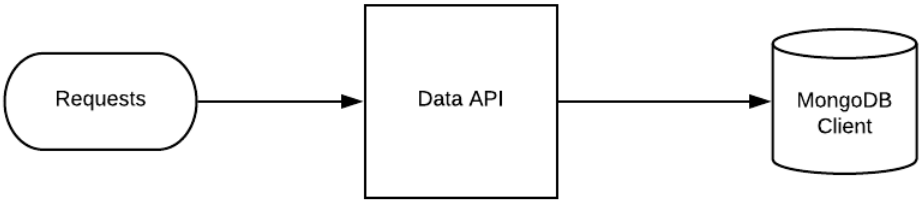
```
pm2 start --json pm2.json
```

5.1.8 Data APP

Data app is a gateway connection to the mongodb.

- CRUD database operations

Data app use Flask, on python >3.5.



Setup dev env

```
pip install

    FLASK_APP=run.py FLASK_DEBUG=1 flask run --port=5010

    or

    npm run server
```

Mongo service

```
cd devtool/

docker-compose up -d
```

Running a mongodb

Installation with python 3

- Python >3.4
- MongoDB

Download the repository

```
git clone https://github.com/maestro-server/data-app.git
```

Install run api

```
python -m flask run.py --port 5010

or

FLASK_APP=run.py FLASK_DEBUG=1 flask run --port 5010

or

npm run server
```

Warning: On production we use gunicorn to handle multiple threads.

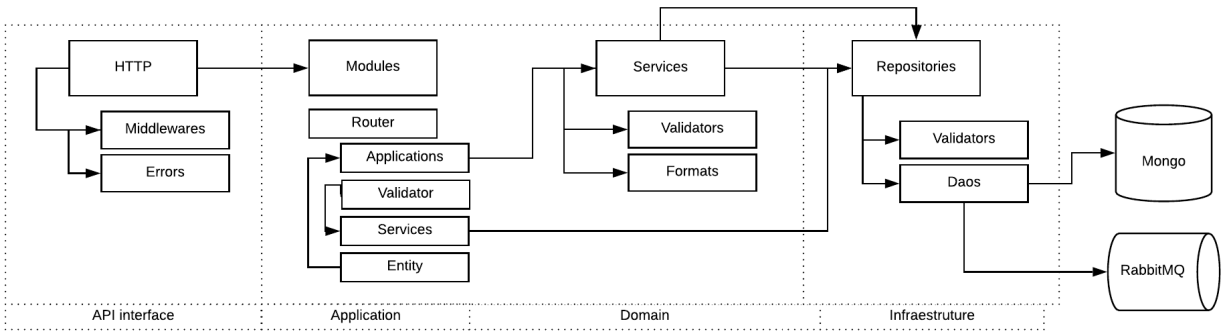
```
# gunicorn_config.py

import os

bind = "0.0.0.0:" + str(os.environ.get("MAESTRO_PORT", 5010))
workers = os.environ.get("MAESTRO_GWORKERS", 2)
```

5.1.9 Audit App

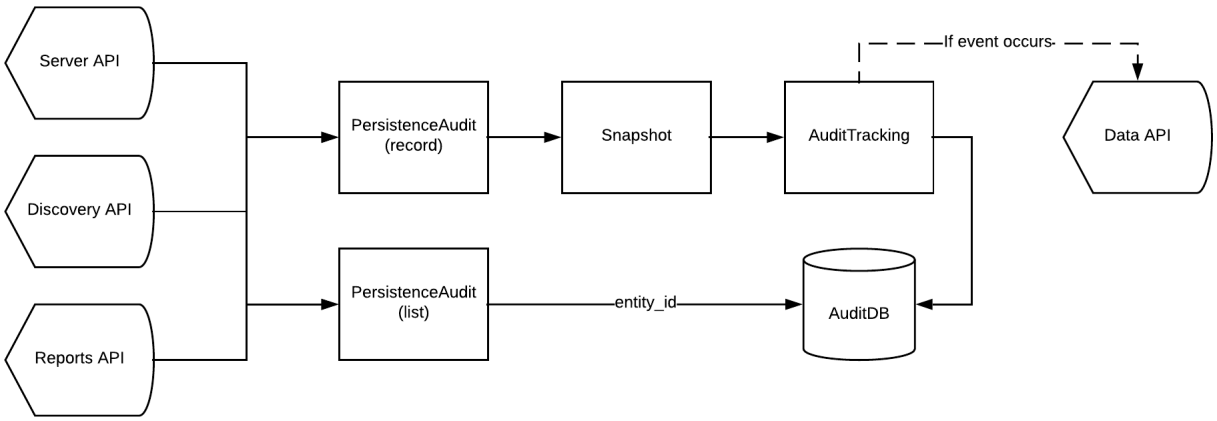
Audit App is a single application to track and record resources change:

- Track resources changes
- Create a change tree
- Store those data

- Audit is made with KrakenJs.
- We use DDD approach to organize a code, they have an infra, repositories, entities (values objects), interfaces, application, and domain folders. DDD in Node Apps



Follow a module flow diagram:



Installing node

- Nodejs 8 or above
- MongoDB 3.x

Download the repository

```
git clone https://github.com/maestro-server/audit-app.git
```

Installing dependencies

```
cd audit-app
npm install
```

Configure env variables

create .env file

```
MAESTRO_PORT=10900
MAESTRO_MONGO_URI='localhost'
MAESTRO_MONGO_DATABASE='maestro-audit'
MAESTRO_DATA_URI="localhost:5005"
```

and

```
npm run server
```

Multiple env

You can use .env files the set configurations

Name	Desc
.env	Default
.env.test	Used on tests
.env.development	node_env was set `development`
.env.production	node_env was set `production`

We use PM2 to handle multiple threads, following the configuration.

PM2:

```
npm install -g pm2

# Create a file pm2.json

{
"apps": [{
    "name": "audit-app",
    "script": "./server.js",
    "env": {
        "production": true,
        "NODE_ENV": "production",
        "PORT": 10900
    }
}]
}
```

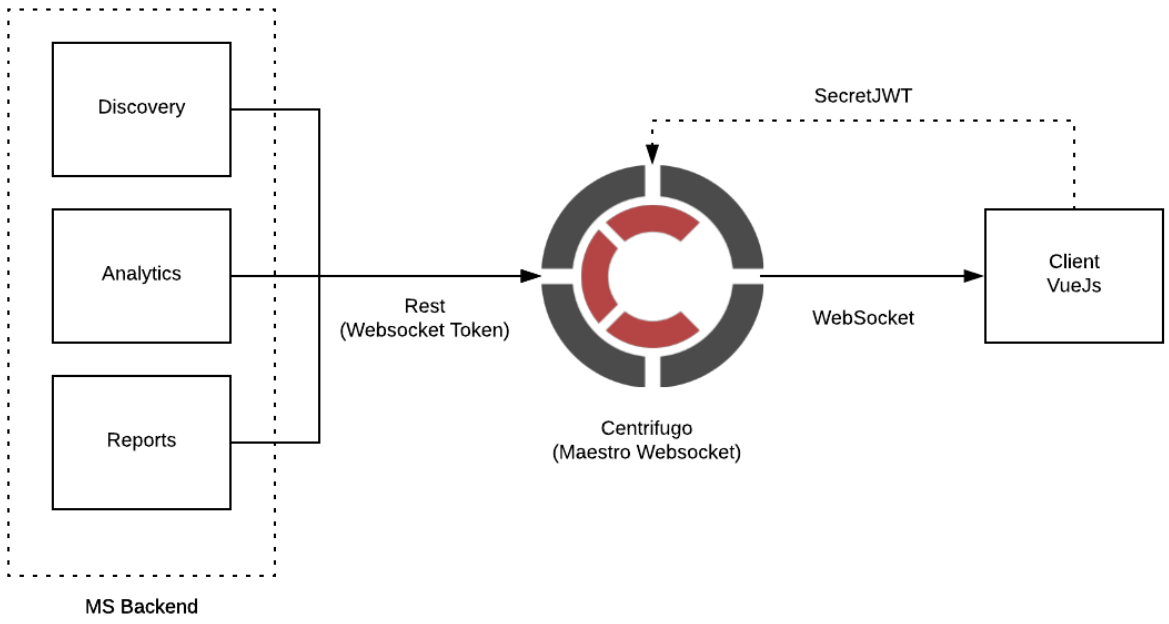
```
pm2 start --json pm2.json
```

5.1.10 WebSocket APP

Centrifugo server. It is a websocket + rest server, the websocket is used by client to get a real time notification, and the rest is used by internal maestro do send a notification to the client.

- Client notification using websockets

Websocket implement a Centrifugo OpenSource project (Centrifugo OpenSource project).



Setup dev env

```
# Generate config
docker run maestro-websocket centrifugo genconfig

# Run websocket
docker run -e MAESTRO_WEBSOCKET_SECRET='secret' -e MAESTRO_SECRETJWT='jwttoken'␣
↪maestroserver/websocket-maestro

# Run centrifugo with admin enabled
docker run -e CENTRIFUGO_ADMIN='pass' -e CENTRIFUGO_ADMIN_SECRET='jwttoken'␣
↪maestroserver/websocket-maestro
```

Download the repository (Centrifugal project)

```
git clone https://github.com/centrifugal/centrifugo
```

Endpoints

Client access

```
var centrifuge = new Centrifuge('ws://{server}/connection/websocket');

centrifuge.subscribe("news", function(message) {
    console.log(message);
});

centrifuge.connect();
```

Backend access

```
import json
import requests

command = {
    "method": "publish",
    "params": {
        "channel": "maestro#${ID-USER}",
        "data": {
            "notify": { // call notify
                "title": "<string>",
                "msg": "<string>",
                "type": "danger|warning|info|success"
            },
            "event": {
                "caller": "<string>" //custom event on client
            }
        }
    }
}
```

5.2 APIs

The communication between each service was made by *rest*, and we use the *api docs* tool to create the api doc.

5.2.1 Server API

You can see the server docs here.

5.2.2 Discovery API

You can see the discovery docs here.

5.2.3 Report API

You can see the report docs here.

5.2.4 Analytics API

You can see the analytics docs here.

5.2.5 Data API

You can see the data server docs here.

5.2.6 Analytics Front API

You can see the analytics front docs here.

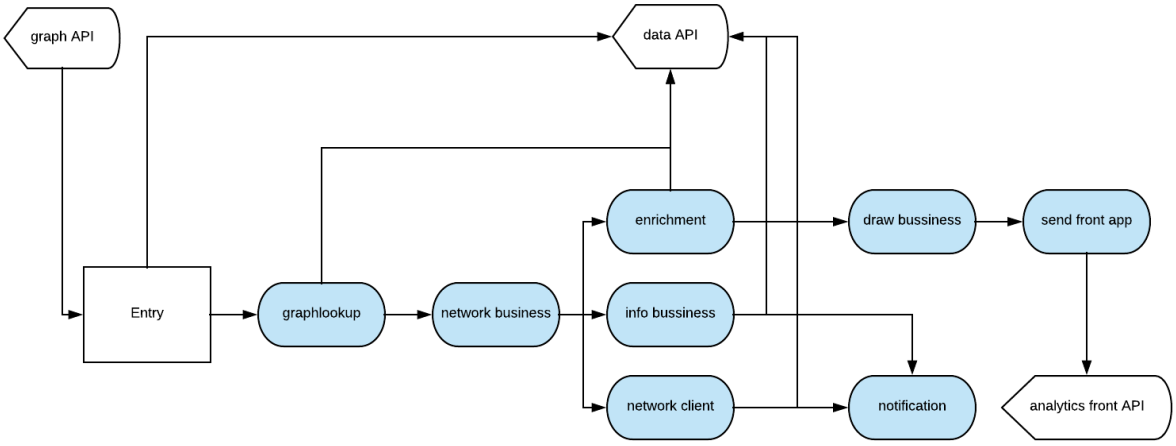
5.2.7 Audit API

You can see the audit docs here.

5.3 Graphs Analytics Algorithm

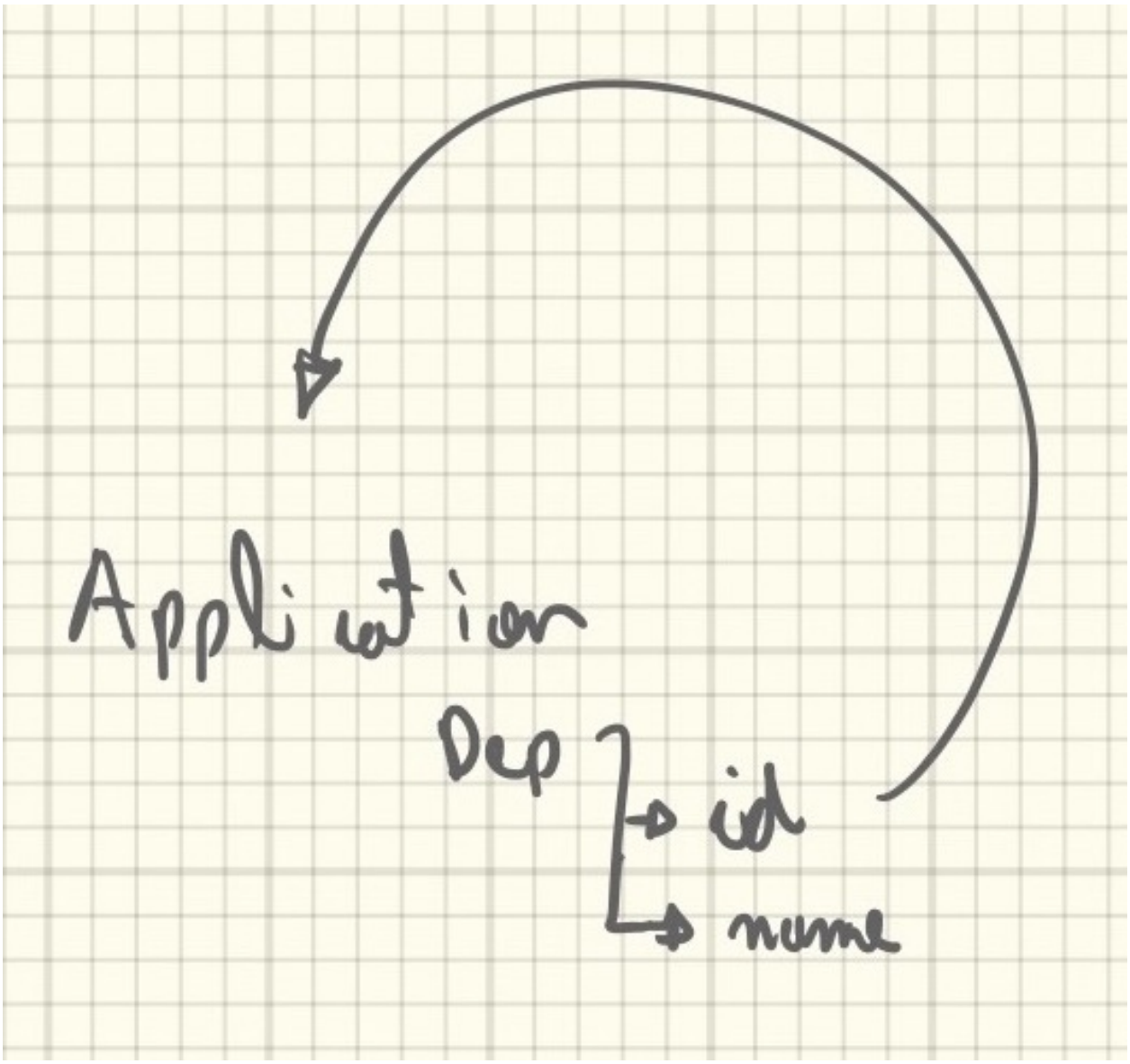
This section will describe about analytics graph algorithm.

- The analytics work flow



5.3.1 Making graph lookup on the mongodb

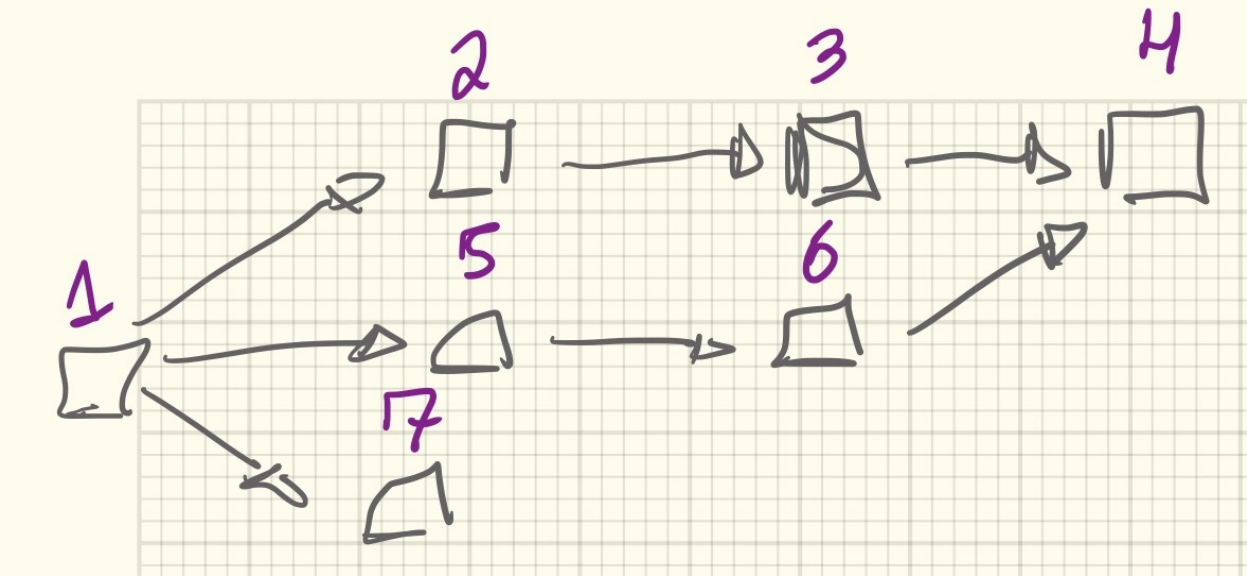
The graph lookup creates a python dict using mongodb graph lookup feature, they use the `application id` on `dependency field`.



5.3.2 Creating a networkX graph

The next step is to create a networkX object based on graph lookup.

We have a recursive function inside each leaf on the tree, the order will be applied using a well defined rules, the results will be a new graph tree and a position matrix for each leaf, this result fixed sorts, duplication and conflicts issues.



An example of code example showing a recursive function

```
def _recursive_draw(self, app, i=0, OHelper=HelperOrderedSuccers):
    if i > 30:
        return

    for item in app:
        if not self._grid.in_index(item):
            node = self._graph.nodes[item]
            helper = self.add_pos_grid(node)

            succ = OHelper(helper).get_succers()
            self._recursive_draw(succ, i + 1)
```

5.3.3 Rules

Follow all rules with can be applied during the create of a new tree. Those rules can be overread each other.

Growing node

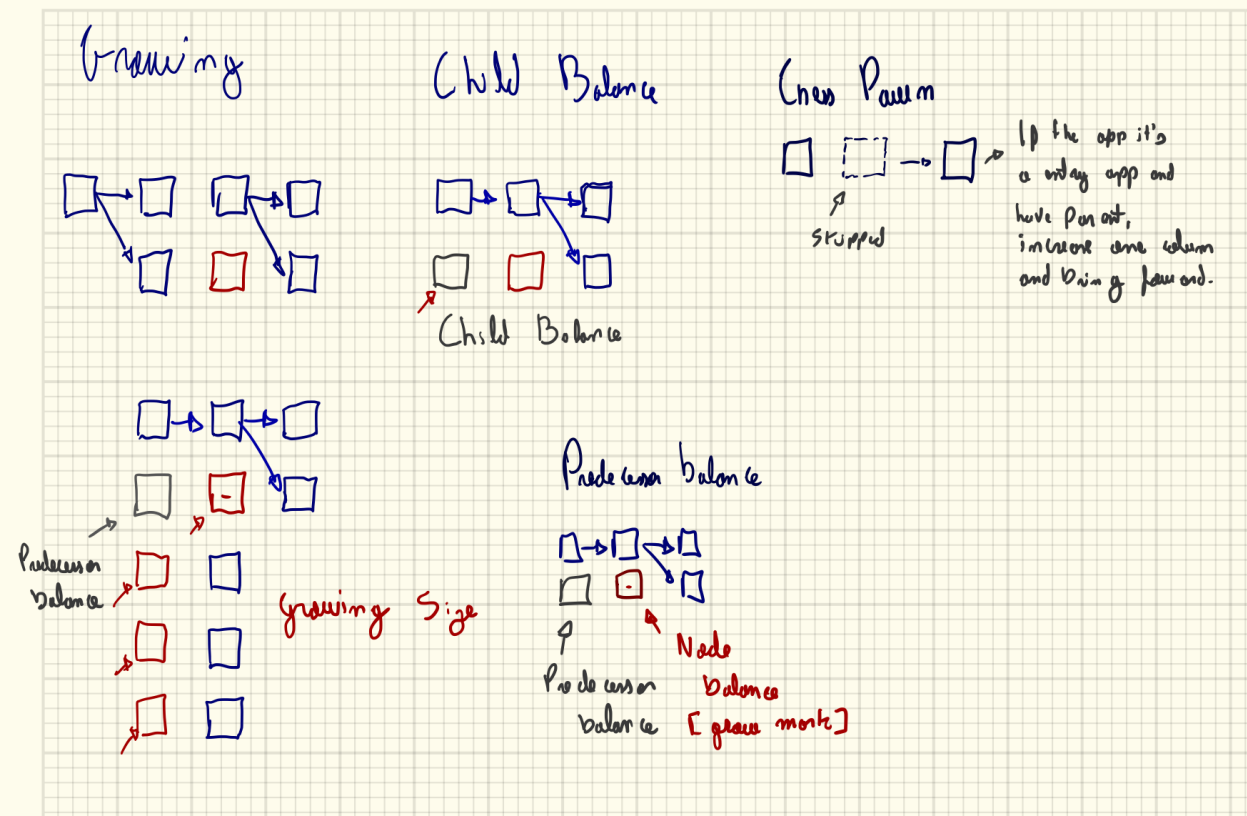
- **When:** If the node have more than one child, growing the node to be equal of the number of child
- **Transform:** Set the node size to be equal to the number of child

Child Balance

- **When:** If the parent node have more than two child.
- **Transform:** Create a dummy item beside to node parent.

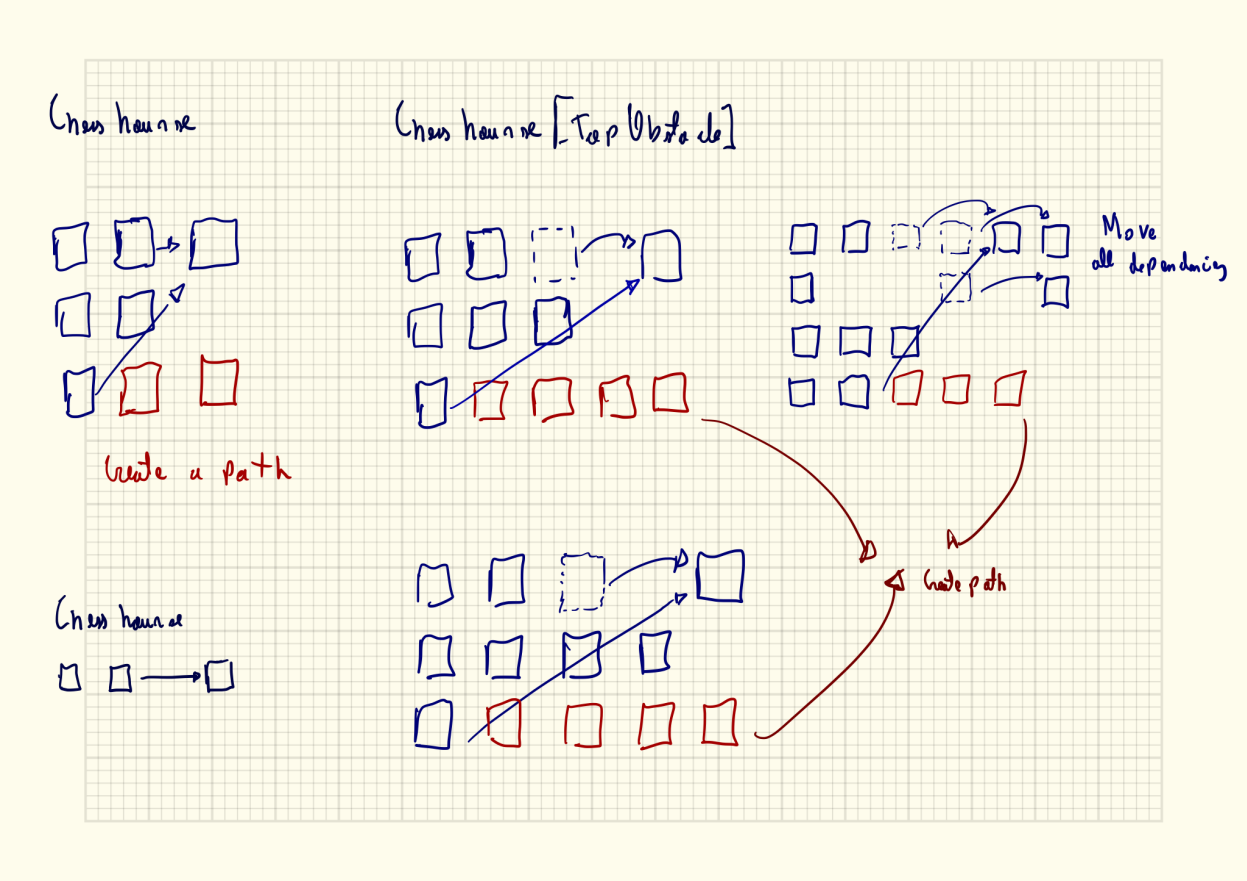
Chess Pawn

- **When:** If the app is an entry point and have parent.
- **Transform:** Skipped one column



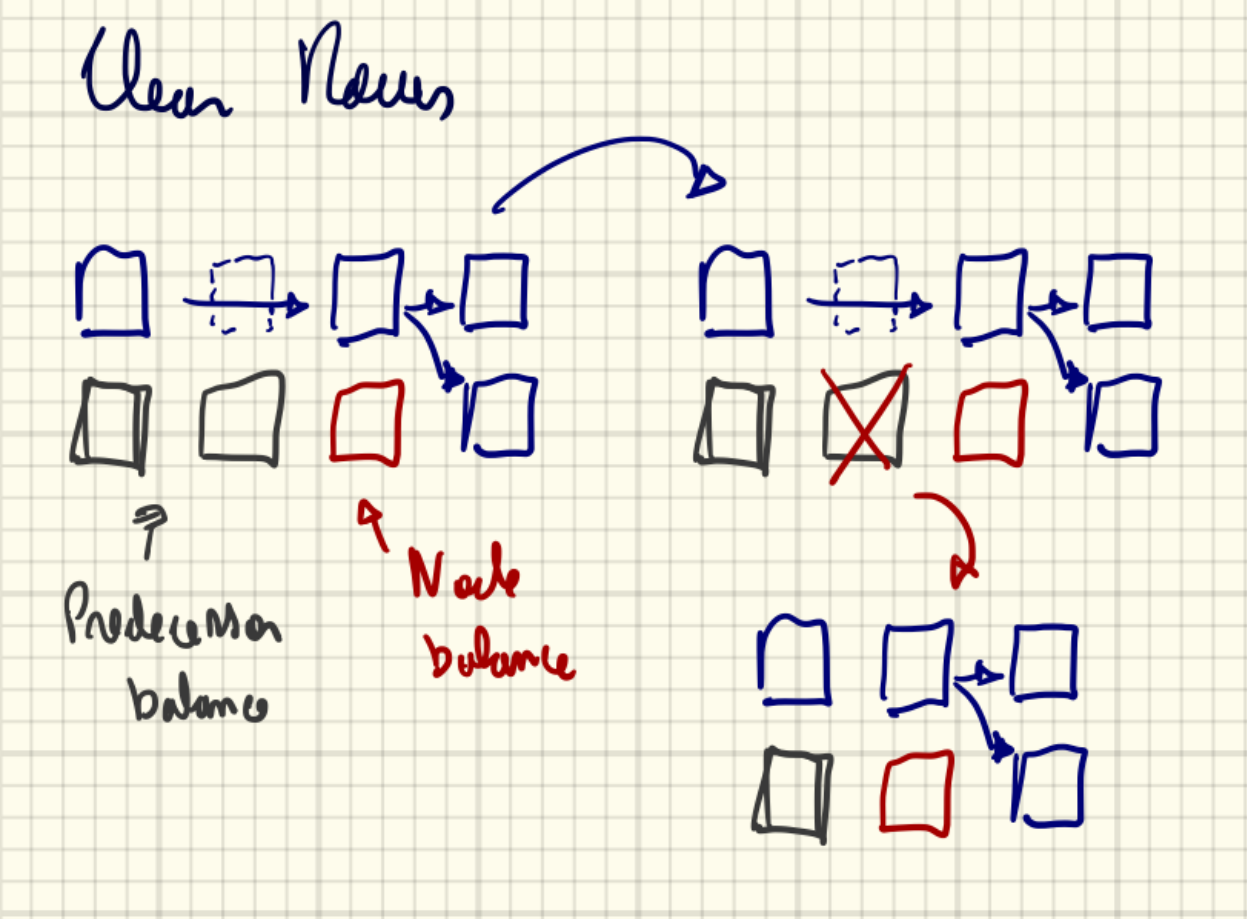
Chess horse

- **When:** If the node have a top obstacle which other nodes point out to a common dependency.
- **Transform:** First push back the dependency to a clear column, and then create a dummy path to the new column.



Clear rows

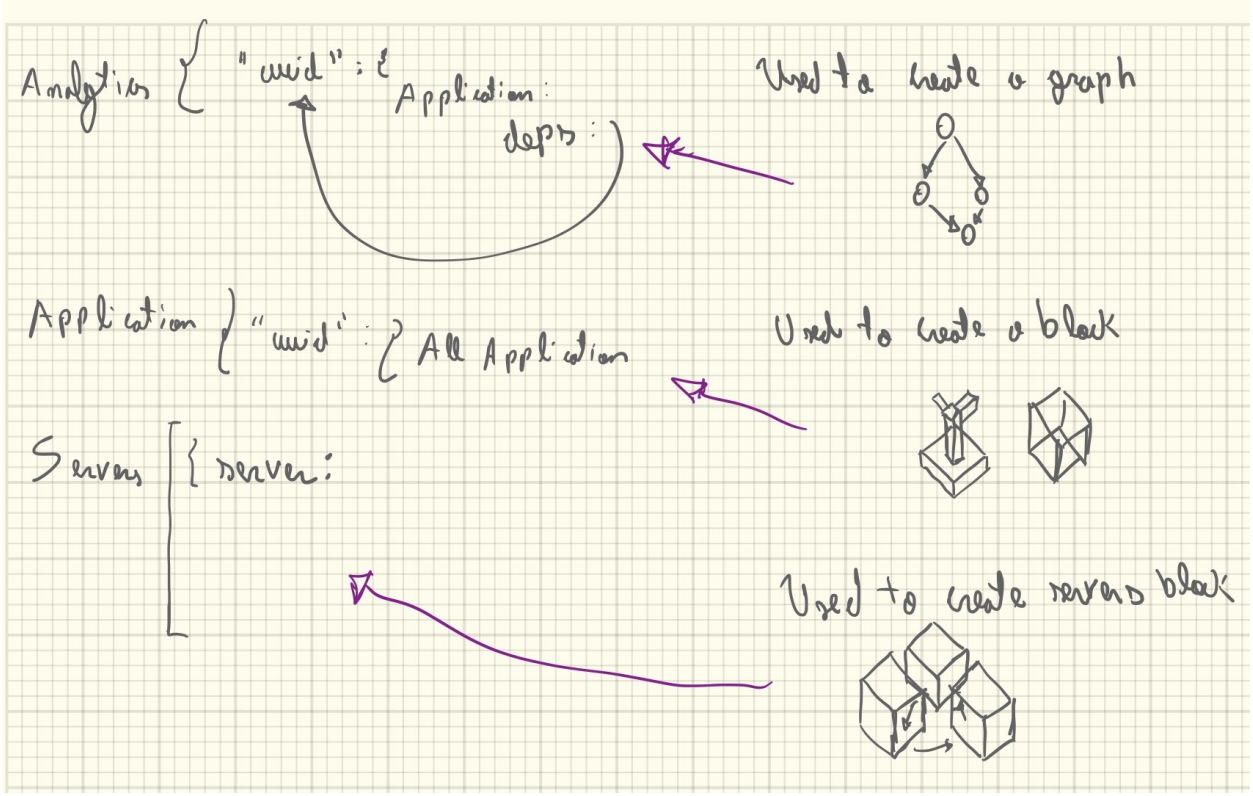
- **When:** If a whole column was empty.
- **Transform:** Delete these column and rebalance the grid.



5.3.4 Enrichment data phase

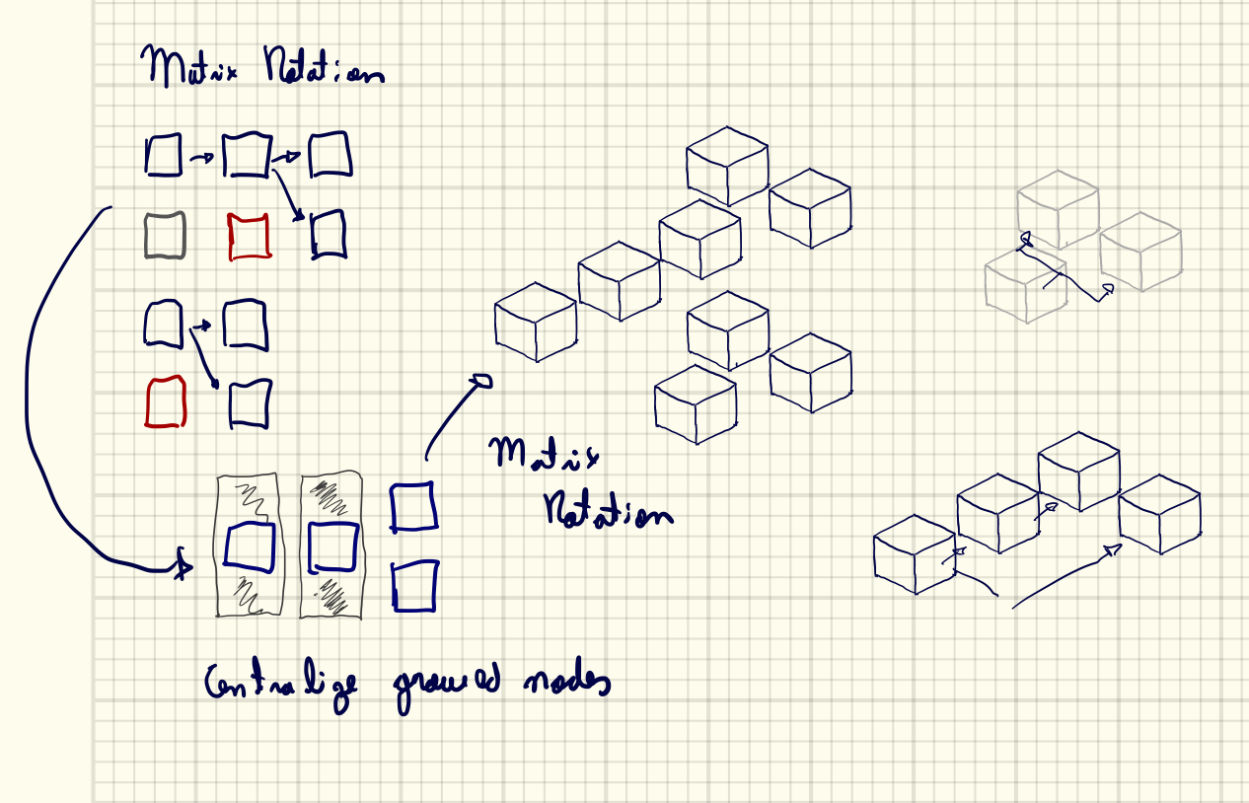
Next step is an enrichment data layer. To filled with a data server information.

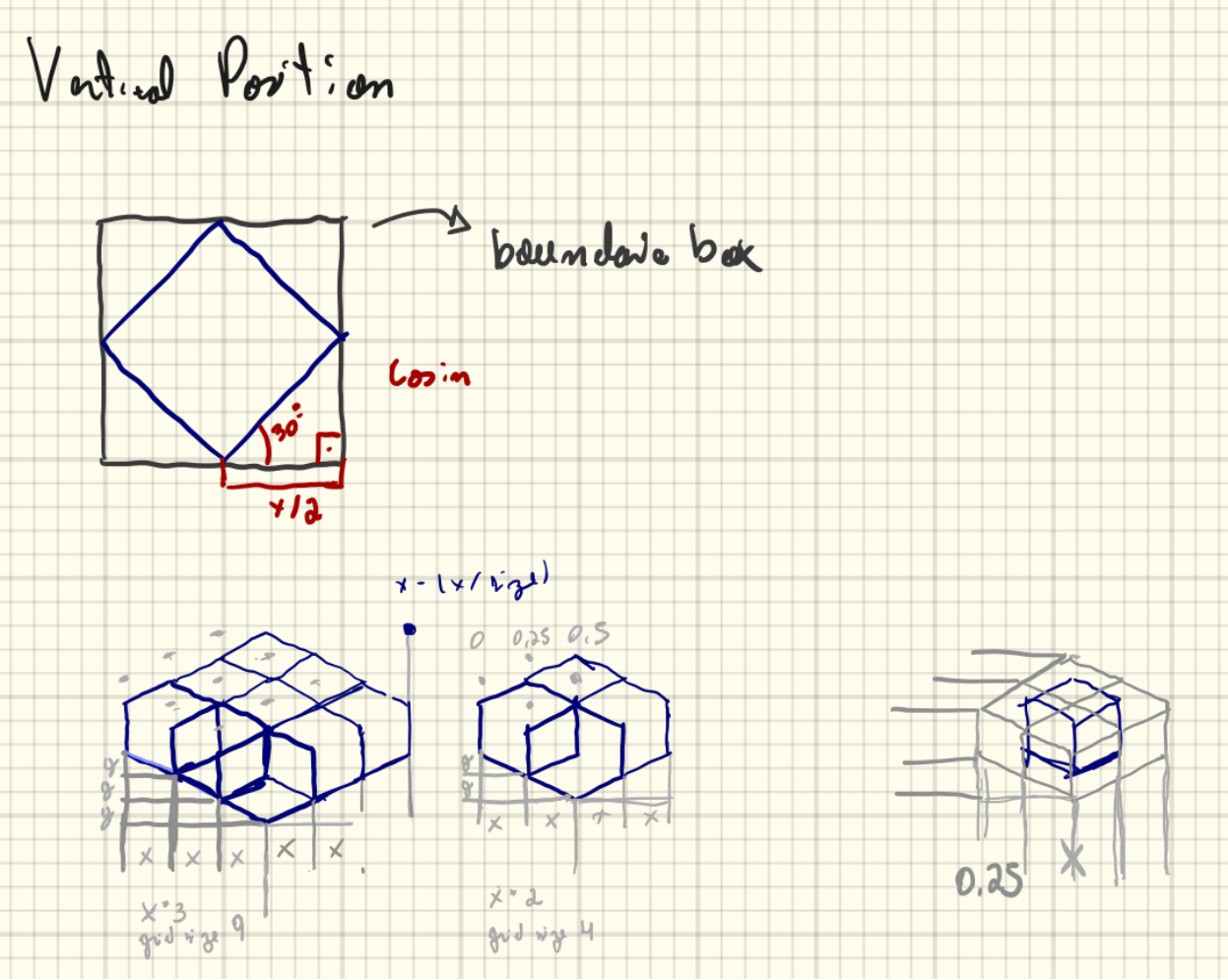
The enrichment step gets two dataset the first one is a json python dict represent as a graph tree, and the second one is a matrix position grid.



5.3.5 Draw phase

The last but not least, it is the dra step, they get the graph tree, matrix position and servers data to make the svgs.


Matrix Notation
Matrix
Notation
Centralized grouped nodes



5.4 Lints

This section describe about lint tools.

5.4.1 JavaScript (Client App)

Uses eslint,

```
npm run lint
```

5.4.2 NodeJs (Server App)

Describe on server-app/.eslintrc

```
npm run lint
```

5.4.3 Python 3 (Discovery, Scheduler and Reports)

pytlint using the default config.

```
npm run lint
```

5.5 Tests

This section describe about test tools.

5.5.1 Server APP

Server uses Mocha + Chai and Sinon to execute tests, and to create a coverage report they use Istambul

```
npm run test

npm run e2e

npm run unit

#you can use a tdd approach to test the code
npm run tdd
```

```
gulp test_e2e
```

Coverage

```
istanbul cover ./node_modules/mocha/bin/_mocha test/**/*js
```

Coveralls

5.5.2 Discovery APP

Testing with pytest

```
npm run test

python -m unittest discover
```

Coveralls

5.5.3 Reports APP

Uses pytest

```
npm run test

python -m unittest discover
```

Coveralls

5.5.4 Data Layer APP

Testing with pytest

```
npm run test

python -m unittest discover
```

Coveralls

5.5.5 Analytics Apps

Testing with pytest

```
npm run test

python -m unittest discover
```

Coveralls

5.5.6 Analytics Front

Testing with pytest

```
npm run e2e
```

Coveralls

5.5.7 Audit App

Testing with pytest

```
npm run e2e
```

Coveralls

5.6 Quality Assurance

5.6.1 Client Maestro

Codacy	
Travis	
CodeClimate	

5.6.2 Server App

CodeClimate	
Travis	
DavidDm	
Codacy	
Coveralls	

5.6.3 Discovery Maestro

Codacy	
Travis	
CodeClimate	

5.6.4 Report Maestro

Codacy	
Travis	
CodeClimate	

5.6.5 Scheduler Maestro

Codacy	
Travis	
CodeClimate	

5.6.6 Data Layer API

Codacy	
Travis	
CodeClimate	

5.6.7 Analytics App

Codacy	
Travis	
CodeClimate	

5.6.8 Analytics Front

Codacy	
Travis	
CodeClimate	

5.6.9 Audit App

Codacy	
Travis	
CodeClimate	

5.7 Third Party

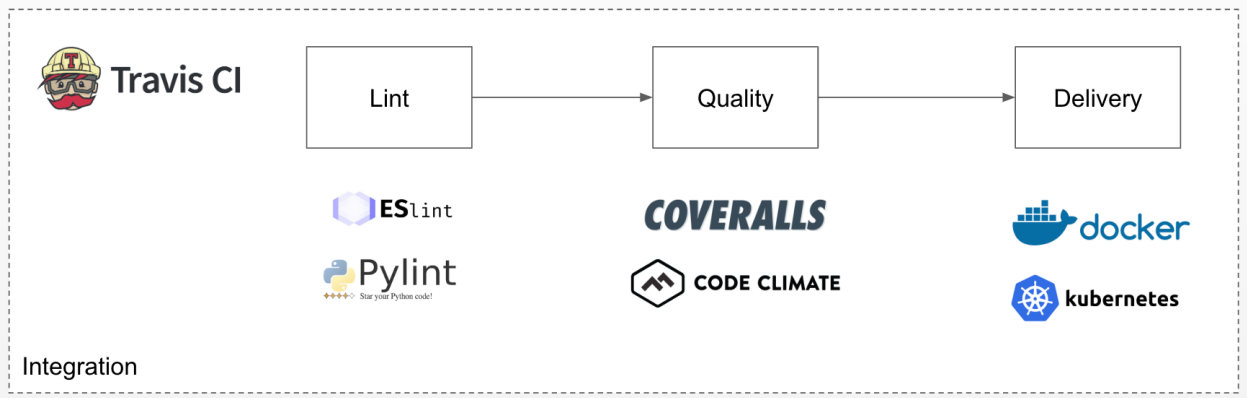
Third Party Support

Provider	Library
AWS	Boto3
OpenStack	OpenStackSDK
Azure	Azure sdk
DigitalOcean	Do SDK

5.8 CI and CD

We use Travis as a CI.

Travis - Maestro dashboard



5.9 Versions

Compatible mapping versions between services

5.9.1 v0.6x - Candidate release

Client	0.15.x
Server	0.6.x
Discovery	0.6.x
Scheduler	0.6.x
Data	0.6.x
Reports	0.6.x
Analytics	0.6.x
Analytics Front	0.6.x
Audit	0.6.x

5.9.2 v0.5x - Beta

Break changes - All services of version 0.5.x isn't compatible with early versions.

Client	0.14.x
Server	0.5.x
Discovery	0.5.x
Scheduler	0.5.x
Data	0.5.x
Reports	0.5.x
Analytics	0.5.x
Analytics Front	0.5.x
Audit	0.5.x

5.9.3 v0.4x - Beta

Break changes - All services of version 0.4.x isn't compatible with early versions.

Client	0.13.x
Server	0.4.x
Discovery	0.4.x
Scheduler	0.4.x
Data	0.4.x
Reports	0.4.x
Analytics	0.4.x
Analytics Front	0.4.x
WebSocket	0.4.x

5.9.4 v0.3x - Beta

Client	0.12.x
Server	0.3.x
Discovery	0.3.x
Scheduler	0.3.x
Data	0.3.x
Reports	0.2.x

5.9.5 v0.2x - Alpha

Client	0.11.x
Server	0.2.x
Discovery	0.2.x
Scheduler	0.2.x
Data	0.1.x
Reports	0.1.x

CHAPTER 6

Troubleshooting

1 - AWS was not able to validate the provided access credentials

I got this error using a valid AWS AK/SK the DescribeInstances operation consistently fails. The other BOTO3 calls work so it's something with this specific call.

```
server-list:
state: danger
msg: An error occurred (AuthFailure) when calling the DescribeInstances operation:␣
↪AWS was not able to validate the provided access credentials At XXXXX
```

- Do the clock is right on your host?

This message error normally happens when it has a wrong clock configuration, docker uses the host timezone. If yes can you try to use ntpdate on the host and then spin up again the discovery-maestro and discovery-maestro-workers https://stackoverflow.com/questions/24551592/how-to-make-sure-dockers-time-syncs-with-that-of-the-host

- Can be caused by a weird circumstance of running a local version at the same time as a cloud hosted one. Some services ran locally others on the cloud due to the way docker-compose was setup.

---

2 - My client got Can't connect to Maestro Server

- The server api are running?
- Your client service have the right configuration?

```
client:
     image: maestroserver/client-maestro
     environment:
     - "API_URL=//maestro.xxx:8888"   <----------------- Server API
     - "STATIC_URL=//maestro.xxx:8888/static" <--------- Static Files
     - "ANALYTICS_URL=//maestro.xxx:9999" <------------- Analytics Front
     - "WEBSOCKET_URL=wss://xxx:8000" <----------------- WebSocket
```

---

3 - Through Unauthorized error during the synchronization - Permission error

If through Unauthorized error, you need to grant ready only permission, as an example on AWS you should create IAM and grant full ready only permissions.

---

4 - The warning status never change

Can be a RabbitMq issue or the Discovery workers weren't running, you can restart the rabbitmq and start the service discovery workers.

You always can check the service logs:

```
docker-compose logs discovery-maestro
# or
docker-compose logs discovery-celery # this one is the discovery workers
```

CHAPTER 7

Contrib

7.1 Reporting issues

- Describe what you expected to happen.
- If possible, include a minimal, complete, and verifiable example to help us identify the issue. This also helps check that the issue is not with your own code.
- Describe what actually happened. Include the full traceback if there was an exception.

7.2 Submitting patches

- All test need to be pass
- All lint need to be green
- Include tests if your patch is supposed to solve a bug, and explain clearly under which circumstances the bug happens. Make sure the test fails without your patch.

Note: All contribution will be accept by Pull Request

CHAPTER 8

Donate

I have made Maestro Server with my heart, think to solve a real operation IT problem. Its not easy, take time and resources.

The donation will be user to:

- Create new features, implement new providers.
- Maintenance libs, securities flaws, and technical points.
- All pages are hosted on AWS
- Demo service is hosted on AWS, and we would like to use kubernetes environment.
- Use telemetry and monitoring services to improve the system.

If you could, you can help me, buy me a coffee, together we can keep the project up and create excited new features.

CHAPTER 9

Contact

Do you have any question, comments, feedback or question about Maestro Server? Please send me a message.

9.1 Feature request

Do you like a new feature? You can open a new request on Github.

Feature request.

CHAPTER 10

License

GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <https://fsf.org/> Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change

the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

'This License' refers to version 3 of the GNU General Public License.

'Copyright' also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

'The Program' refers to any copyrightable work licensed under this License. Each licensee is addressed as 'you'. 'Licensees' and 'recipients' may be individuals or organizations.

To 'modify' a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a 'modified version' of the earlier work or a work 'based on' the earlier work.

A 'covered work' means either the unmodified Program or a work based on the Program.

To 'propagate' a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To 'convey' a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays 'Appropriate Legal Notices' to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The 'source code' for a work means the preferred form of the work for making modifications to it. 'Object code' means any non-source form of a work.

A 'Standard Interface' means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The 'System Libraries' of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A 'Major Component', in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The 'Corresponding Source' for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free

programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to 'keep intact all notices'.

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an 'aggregate' if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A 'User Product' is either (1) a 'consumer product', which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, 'normally used' refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

'Installation Information' for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does

not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

'Additional permissions' are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered 'further restrictions' within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An 'entity transaction' is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A 'contributor' is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's 'contributor version'.

A contributor's 'essential patent claims' are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, 'control' includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a 'patent license' is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To 'grant' such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange,

in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. 'Knowingly relying' means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is 'discriminatory' if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License 'or any later version' applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM 'AS IS' WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the 'copyright' line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.> Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <https://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

<program> Copyright (C) <year> <name of author> This program comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an 'about box'.

You should also get your employer (if you work as a programmer) or school, if any, to sign a 'copyright disclaimer' for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <https://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <https://www.gnu.org/licenses/why-not-lgpl.html>.