

CONSTRUISEZ ET TESTEZ UNE INFRASTRUCTURE DE DONNÉES

Projet Data Engineer – KELLENI Antoine
[lien GitHub](#) 1

ROADMAP DE LA MISSION

- 1. Contexte & objectifs**
2. Démarche technique
3. Schéma de l'architecture & logigramme
4. Schéma de la BDD
5. Connecteur Airbyte
6. Reporting
7. Déploiement AWS

Contexte & objectifs

Migration et intégration d'une infrastructure de données vers le cloud AWS

Besoin de centraliser et fiabiliser les données issues de plusieurs sources météo (GreenCoop).

Objectif :

automatiser la collecte, transformation, stockage et analyse des données pour simplifier l'exploitation des données dans MongoDB.

Infrastructure cloud (AWS), connecteurs Airbyte, containers Docker, base MongoDB, pipeline ETL Python.

ROADMAP DE LA MISSION

1. Contexte & objectifs
2. **Démarche technique**
3. Schéma de l'architecture & logigramme
4. Schéma de la BDD
5. Connecteur Airbyte
6. Reporting
7. Déploiement AWS

Démarche technique et justification des choix

Démarche technique

Collecte : Airbyte connecte les 3 fichiers bruts (JSON, XLSX) à AWS S3.

Transformation : script Python transform_to_mongo_json.py nettoie, normalise les données et va produire 2 datasets exploitable par MongoDB. (stations.json, measurements.json). + d'autres scripts que nous allons voir en détail plus tard.

Stockage & conteneurisation : Docker + docker-compose pour exécuter MongoDB localement.

Déploiement : migration et exécution sur AWS ECS avec persistance S3

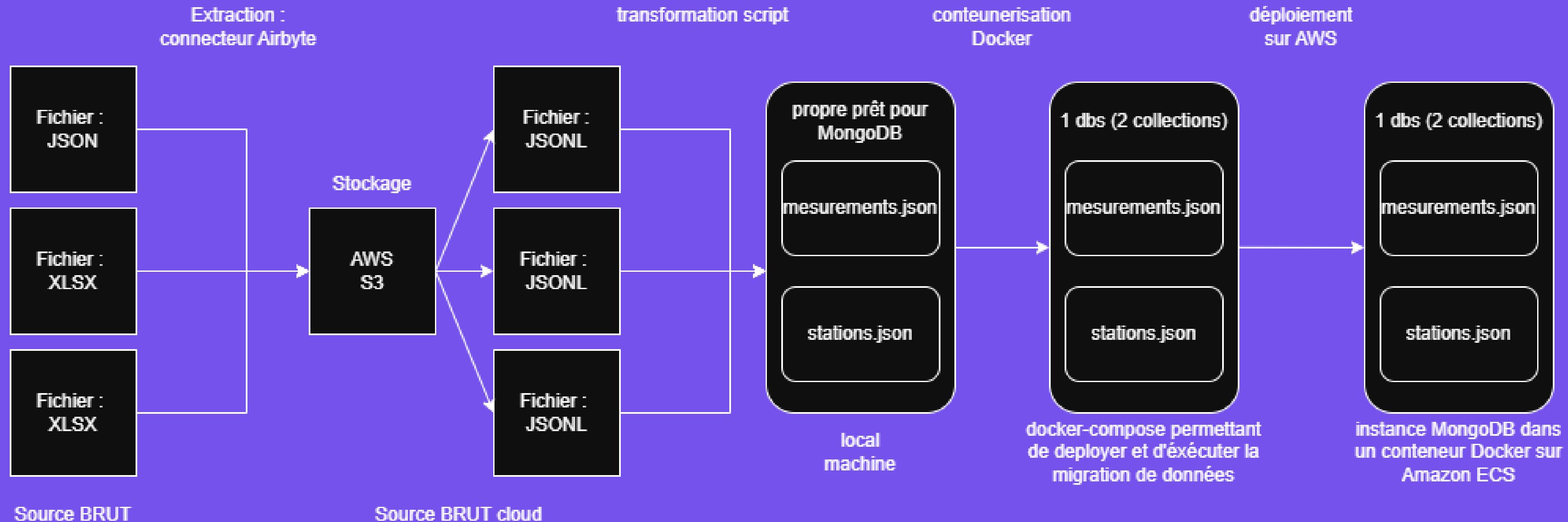
Justification des choix

- Airbyte → automatisation de la collecte.
- Python → flexibilité sur les formats JSON complexes.
- MongoDB → stockage idéal pour données semi-structurées.
- Docker/ECS → portabilité, déploiement simplifié.
- S3 → stockage centralisé et fiable dans le cloud.

ROADMAP DE LA MISSION

1. Contexte & objectifs
2. Démarche technique
- 3. Schéma de l'architecture & logigramme**
4. Schéma de la BDD
5. Connecteur Airbyte
6. Reporting
7. Déploiement AWS

Schéma de l'architecture/pipeline ETL



Source : 1 fichier JSON bruts + 2 fichiers XLSX (S3)

Extraction : connecteur Airbyte

Transformation : script Python

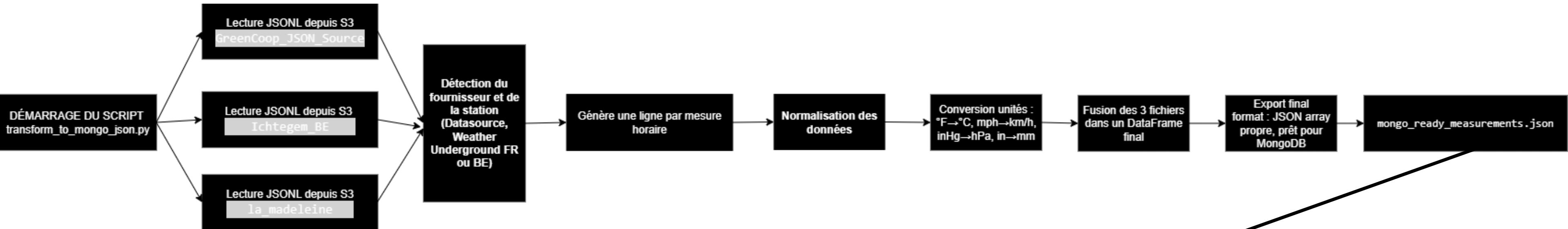
Chargement : MongoDB conteneurisé dans ECS

Orchestration : Docker Compose / ECS Tasks

Logigramme du Script generate_stations_all_from_s3.py

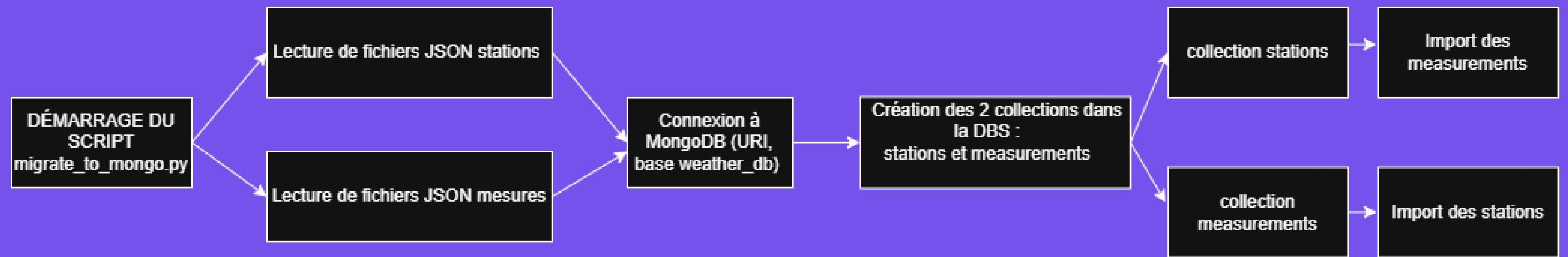


Logigramme du Script transform_to_mongo_json.py



```
{
  _id: ObjectId('68feaf4985c646fea8b462f6'),
  id_station: '07015',
  dh_utc: '2024-10-05 00:00:00',
  Date: '2024-10-05',
  DateTime: '2024-10-05 02:00:00',
  humidite: 89,
  nebulosite: null,
  neige_au_sol: null,
  pluie_1h: 0,
  pluie_3h: 0,
  point_de_rosee: 5.9,
  pression: 1020.7,
  temperature: 7.6,
  temps_omm: null,
  vent_direction: 90,
  vent_moyen: 3.6,
  vent_rafales: 7.2,
  visibilite: 6000
},
```

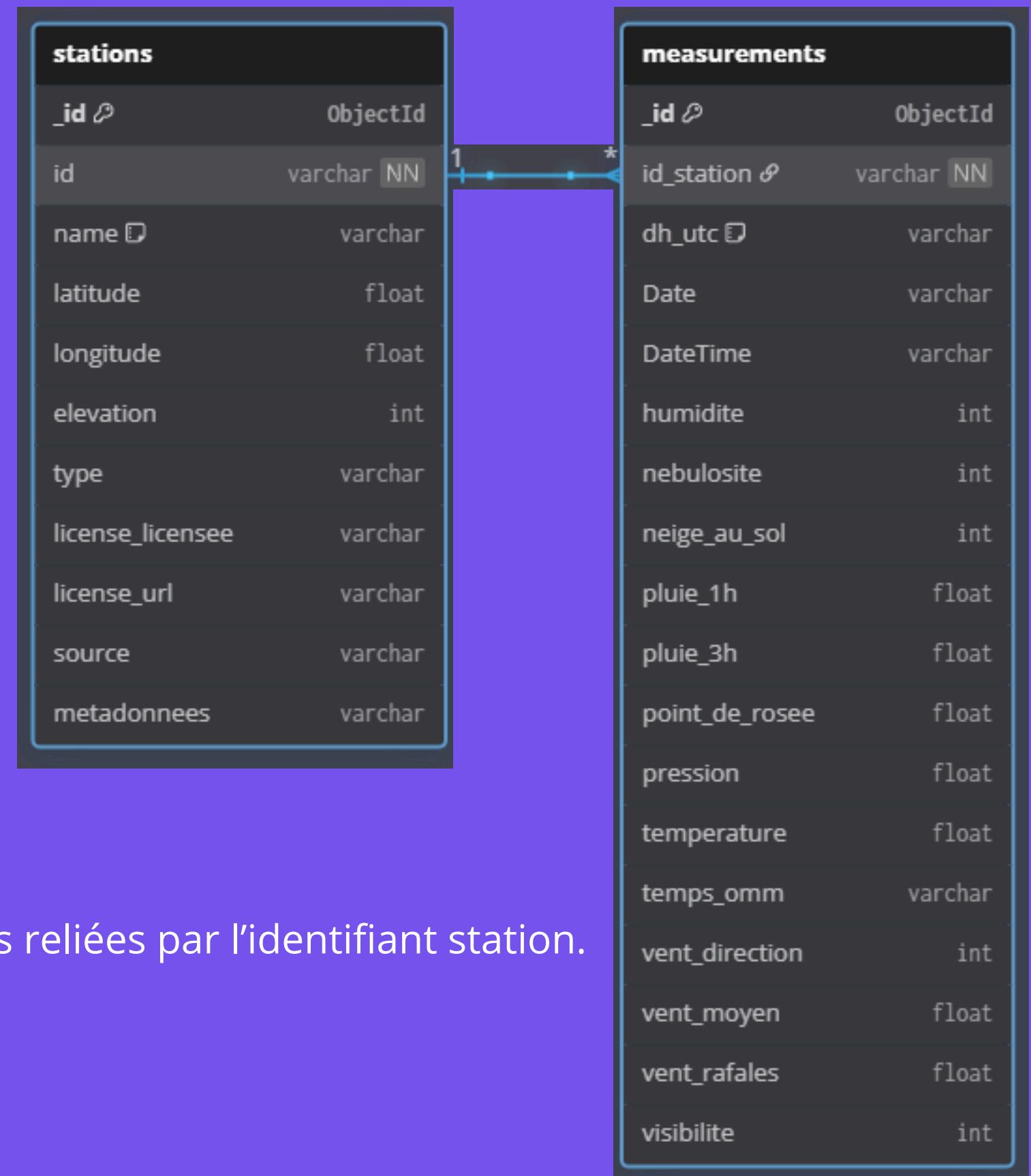
Logigramme du script migrate_to_mongo.py



ROADMAP DE LA MISSION

1. Contexte & objectifs
2. Démarche technique
3. Schéma de l'architecture & logigramme
- 4. Schéma de la BDD**
5. Connecteur Airbyte
6. Reporting
7. Déploiement AWS

Schéma de la base de données



Deux collections reliées par l'identifiant station.

Données importées dans les collections MongoDB

```
weather_db> db.stations.find()
[
  {
    _id: ObjectId('68feaf4885c646fea8b462f0'),
    id: '00052',
    elevation: 16,
    latitude: 50.689,
    license: {
      license: 'CC BY',
      url: 'https://creativecommons.org/licenses/by/2.0/fr/',
      source: 'infoclimat.fr',
      metadonnees: 'https://www.infoclimat.fr/stations/metadonnees.php?id=00052'
    },
    longitude: 2.877,
    name: 'Armentières',
    type: 'static'
  },
  {
    _id: ObjectId('68feaf4885c646fea8b462f1'),
    id: '000R5',
    elevation: 17,
    latitude: 50.968,
    license: {
      license: 'CC BY',
      url: 'https://creativecommons.org/licenses/by/2.0/fr/',
      source: 'infoclimat.fr',
      metadonnees: 'https://www.infoclimat.fr/stations/metadonnees.php?id=000R5'
    },
    longitude: 2.441,
    name: 'Bergues',
    type: 'static'
  },
  {
    _id: ObjectId('68feaf4885c646fea8b462f2'),
    id: '07015',
    elevation: 47,
    latitude: 50.575
  }
]
```

collection station
(métadonnées)

```
weather_db> db.measurements.find()
[
  {
    _id: ObjectId('68feaf4985c646fea8b462f6'),
    id_station: '07015',
    dh_utc: '2024-10-05 00:00:00',
    Date: '2024-10-05',
    DateTime: '2024-10-05 02:00:00',
    humidite: 89,
    nebulosite: null,
    neige_au_sol: null,
    pluie_1h: 0,
    pluie_3h: 0,
    point_de_rosee: 5.9,
    pression: 1020.7,
    temperature: 7.6,
    temps_omm: null,
    vent_direction: 90,
    vent_moyen: 3.6,
    vent_rafales: 7.2,
    visibilite: 6000
  },
  {
    _id: ObjectId('68feaf4985c646fea8b462f7'),
    id_station: '07015',
    dh_utc: '2024-10-05 01:00:00',
    Date: '2024-10-05',
    DateTime: '2024-10-05 03:00:00',
    humidite: 92,
    nebulosite: null,
    neige_au_sol: null,
    pluie_1h: 0,
    pluie_3h: null,
    point_de_rosee: 6.3,
    pression: 1020.6,
    temperature: 7.5,
    temps_omm: null,
    vent_direction: 30,
    vent_moyen: 3.6,
    vent_rafales: 7.2,
    visibilite: 7000
  },
  {
    _id: ObjectId('68feaf4985c646fea8b462f8'),
    id_station: '07015',
    dh_utc: '2024-10-05 02:00:00',
    Date: '2024-10-05',
    DateTime: '2024-10-05 04:00:00',
    humidite: 95,
    nebulosite: null,
    neige_au_sol: null,
    pluie_1h: 0,
    pluie_3h: null,
    point_de_rosee: 6.8,
    pression: 1020.5,
    temperature: 7.4,
    temps_omm: null,
    vent_direction: 60,
    vent_moyen: 3.5,
    vent_rafales: 7.1,
    visibilite: 7500
  }
]
```

collection measurements
(mesures horaires)

Descriptions des données

Champ	Type	Description
<code>_id</code>	ObjectId	Identifiant unique généré par MongoDB
<code>id</code>	String	Identifiant officiel de la station (ex. : "00052")
<code>name</code>	String	Nom de la station (ex. : "Armentières")
<code>latitude</code>	Float	Latitude géographique de la station
<code>longitude</code>	Float	Longitude géographique de la station
<code>elevation</code>	Integer	Altitude de la station en mètres
<code>type</code>	String	Type de station (ex. : "static", "amateur", etc.)
<code>source</code>	String	Source des données (ex. : "infoclimat.fr")
<code>metadonnees</code>	String	URL vers les métadonnées de la station
<code>license</code>	Objet	Détail de la licence d'utilisation
<code>license.license</code>	String	Type de licence (ex. : "CC BY")
<code>license.url</code>	String	Lien vers le texte complet de la licence

collection stations

Champ	Type	Description
<code>_id</code>	ObjectId	Identifiant unique MongoDB
<code>id_station</code>	String	Identifiant de la station émettrice de la mesure
<code>dh_utc</code>	String	Date/heure UTC au format ISO
<code>Date</code>	String	Date locale de la mesure (AAAA-MM-JJ)
<code>DateTime</code>	String	Date et heure locale (Europe/Paris)
<code>temperature</code>	Float	Température en °C
<code>pression</code>	Float	Pression atmosphérique en hPa
<code>humidite</code>	Float	Humidité relative en %
<code>point_de_rosee</code>	Float	Température du point de rosée en °C
<code>visibilite</code>	Float	Visibilité horizontale en mètres
<code>vent_moyen</code>	Float	Vitesse moyenne du vent en m/s
<code>vent_rafales</code>	Float	Rafales maximales du vent en m/s
<code>vent_direction</code>	Integer	Direction du vent en degrés (0–360°)
<code>pluie_1h</code>	Float	Cumul de pluie sur 1 heure (mm)
<code>pluie_3h</code>	Float	Cumul de pluie sur 3 heures (mm)
<code>neige_ae_sol</code>	Float	Épaisseur de neige au sol (cm)
<code>nebulosite</code>	Float	Couverture nuageuse en %
<code>temps_omm</code>	String	Code OMM du temps observé

collections measurements

ROADMAP DE LA MISSION

1. Contexte & objectifs
2. Démarche technique
3. Schéma de l'architecture & logigramme
4. Schéma de la BDD
5. **Connecteur Airbyte**
6. Reporting
7. Déploiement AWS

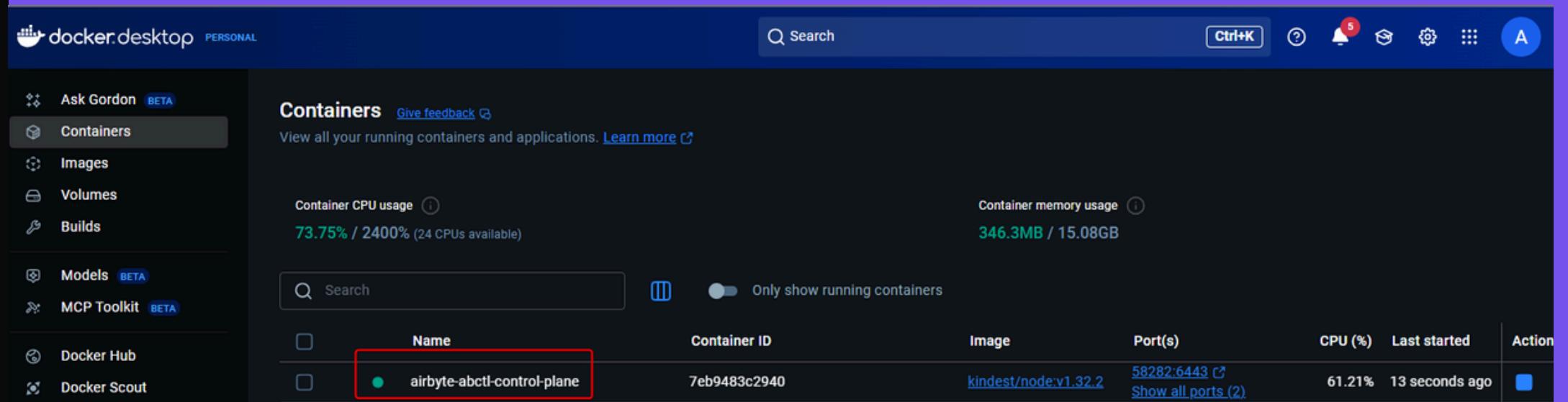
Installation locale d'Airbyte avec Docker

```
version: v0.30.2
PS C:\Program Files\abctl-v0.30.2-windows-amd64> abctl local install
INFO Using Kubernetes provider:
Provider: kind
Kubeconfig: C:\Users\antois\airbyte\abctl\abctl.kubeconfig
Context: kind-airbyte-abctl
SUCCESS Found Docker installation: version 28.3.2
INFO No existing cluster found, cluster 'airbyte-abctl' will be created
SUCCESS Port 8000 appears to be available
SUCCESS Cluster 'airbyte-abctl' created
INFO Patching image airbyte/db:1.7.0-17
INFO Pulling image airbyte/async-profiler:2.0.0
INFO Pulling image airbyte/bootloader:2.0.0
INFO Pulling image airbyte/connector-builder-server:2.0.0
INFO Pulling image airbyte/connector-sidecar:2.0.0
INFO Pulling image airbyte/container-orchestrator:2.0.0
INFO Pulling image airbyte/cron:2.0.0
INFO Pulling image airbyte/db:1.7.0-17
INFO Pulling image airbyte/server:2.0.0
INFO Pulling image airbyte/utils:2.0.0
INFO Pulling image airbyte/worker:2.0.0
INFO Pulling image airbyte/workload-api-server:2.0.0
INFO Pulling image airbyte/workload-init-container:2.0.0
INFO Pulling image airbyte/workload-launcher:2.0.0
INFO Pulling image temporalio/auto-setup:1.27.2
INFO Namespace 'airbyte-abctl' created
INFO Persistent volume 'airbyte-local-pv' created
INFO Persistent volume claim 'airbyte-storage-pvc' created
INFO Persistent volume 'airbyte-volume-db' created
INFO Persistent volume claim 'airbyte-volume-db-airbyte-db-0' created
INFO Starting Helm Chart installation of 'airbyte/airbyte' (version: 2.0.18)
INFO Installed Helm Chart airbyte/airbyte:
  Name: airbyte-abctl
  Namespace: airbyte-abctl
  Version: 2.0.18
  AppVersion: 2.0.0
  Release: 1
INFO Starting Helm Chart installation of 'nginx/ingress-nginx' (version: 4.13.3)
INFO Installed Helm Chart nginx/ingress-nginx:
  Name: ingress-nginx
  Namespace: ingress-nginx
  Version: 4.13.3
  AppVersion: 1.13.3
  Release: 1
INFO No existing Ingress found, creating one
INFO Ingress created
INFO Launched web-browser successfully for http://localhost:8000
INFO Airbyte installation complete.
INFO A password may be required to login. The password can be found by running
the command abctl local credentials
PS C:\Program Files\abctl-v0.30.2-windows-amd64>
```

installation terminée et lancement sur localhost:8000

```
PS C:\Program Files\abctl-v0.30.2-windows-amd64> abctl version
INFO Thanks for using Airbyte!
Anonymous usage reporting is currently enabled. For mo
version: v0.30.2
```

vérification de la version (v0.30.2)

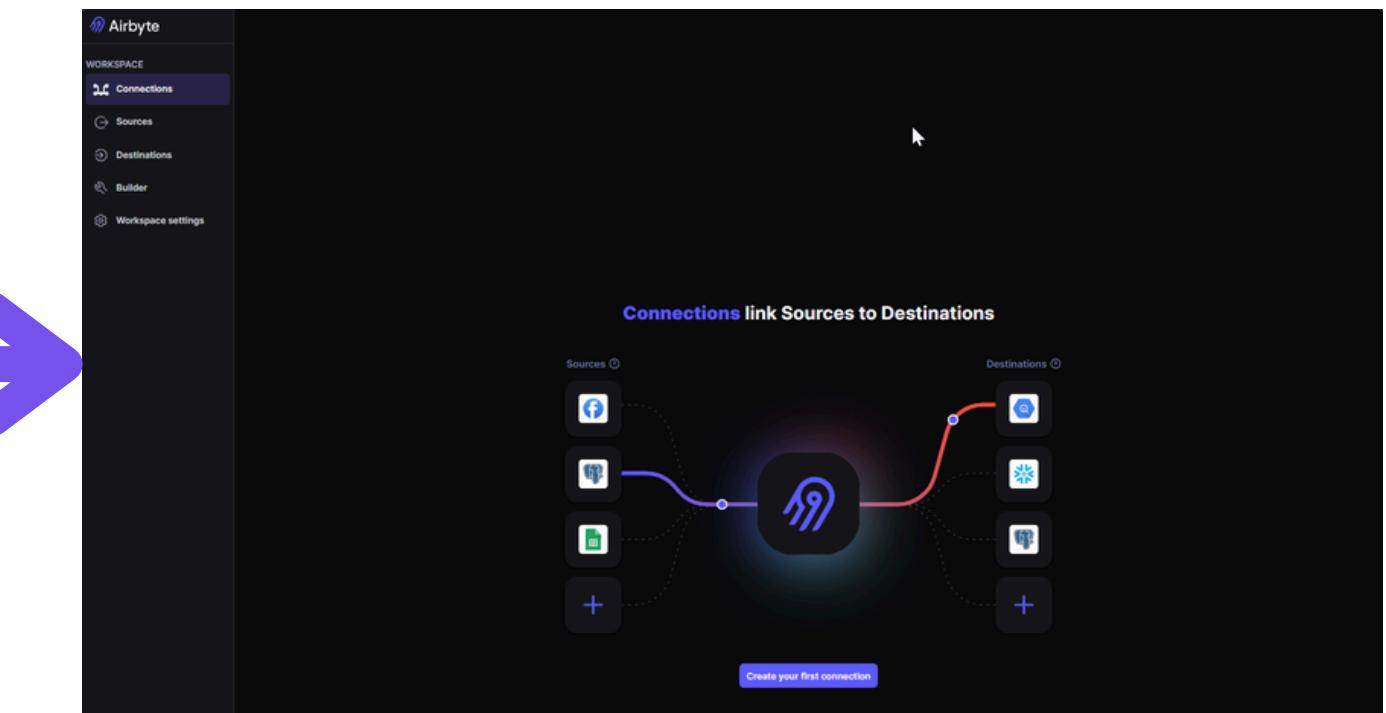


conteneur actif dans Docker

Connexion à Airbyte et accès à l'interface

```
PS C:\Program Files\abctl-v0.30.2-windows-amd64> abctl local credentials
INFO    Using Kubernetes provider:
        Provider: kind
        Kubeconfig: C:\Users\antoi\.airbyte\abctl\abctl.kubeconfig
        Context: kind-airbyte-abctl
SUCCESS  Retrieving your credentials from 'airbyte-auth-secrets'
INFO    Credentials:
        Email: antoi[REDACTED]gmail.com
        Password: tf2[REDACTED]szx
        Client-Id: fc5c[REDACTED]45a1c
        Client-Secret: Dor[REDACTED]2oq
PS C:\Program Files\abctl-v0.30.2-windows-amd64>
```

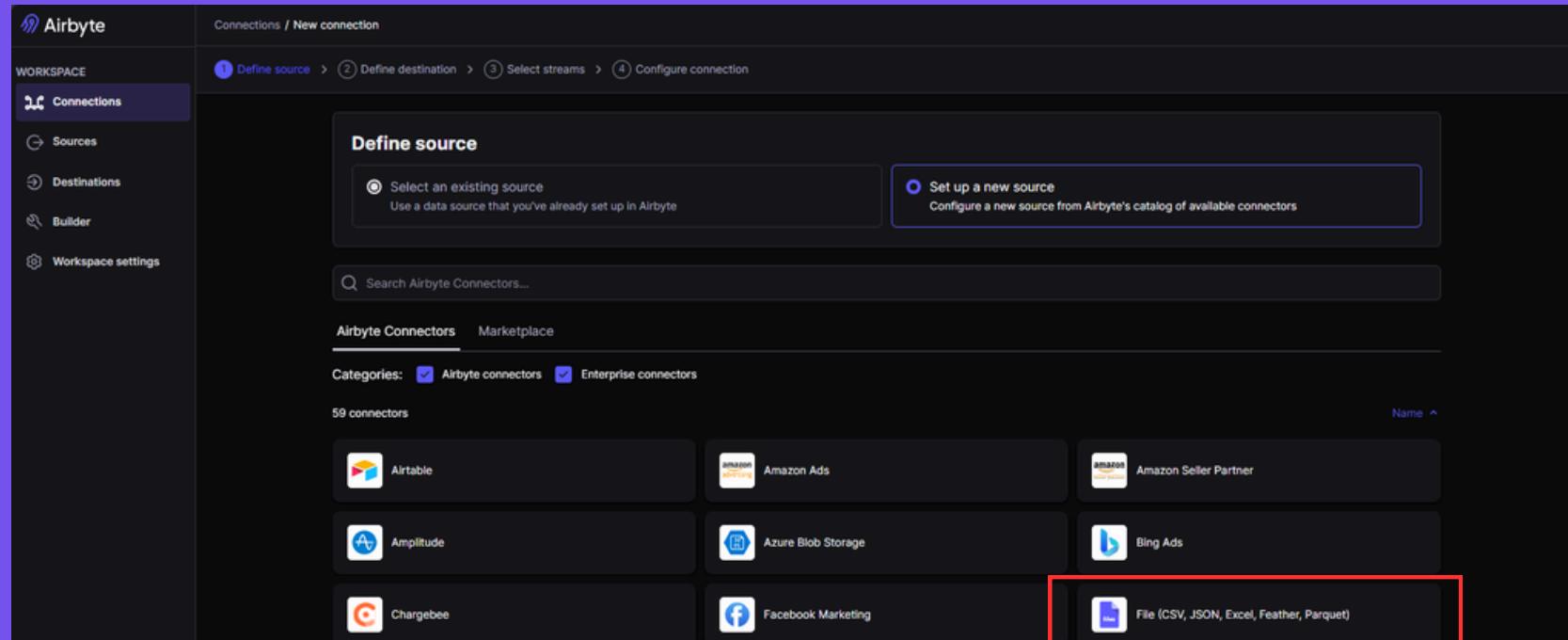
récupération des identifiants



interface principale d'Airbyte (accueil)

Création des sources de données

2 fichiers XLSX + 1 fichier JSON



Airbyte Connections / New connection

WORKSPACE

Connections

Sources

Destinations

Builder

Workspace settings

Define source

Select an existing source

Set up a new source

Search Airbyte Connectors...

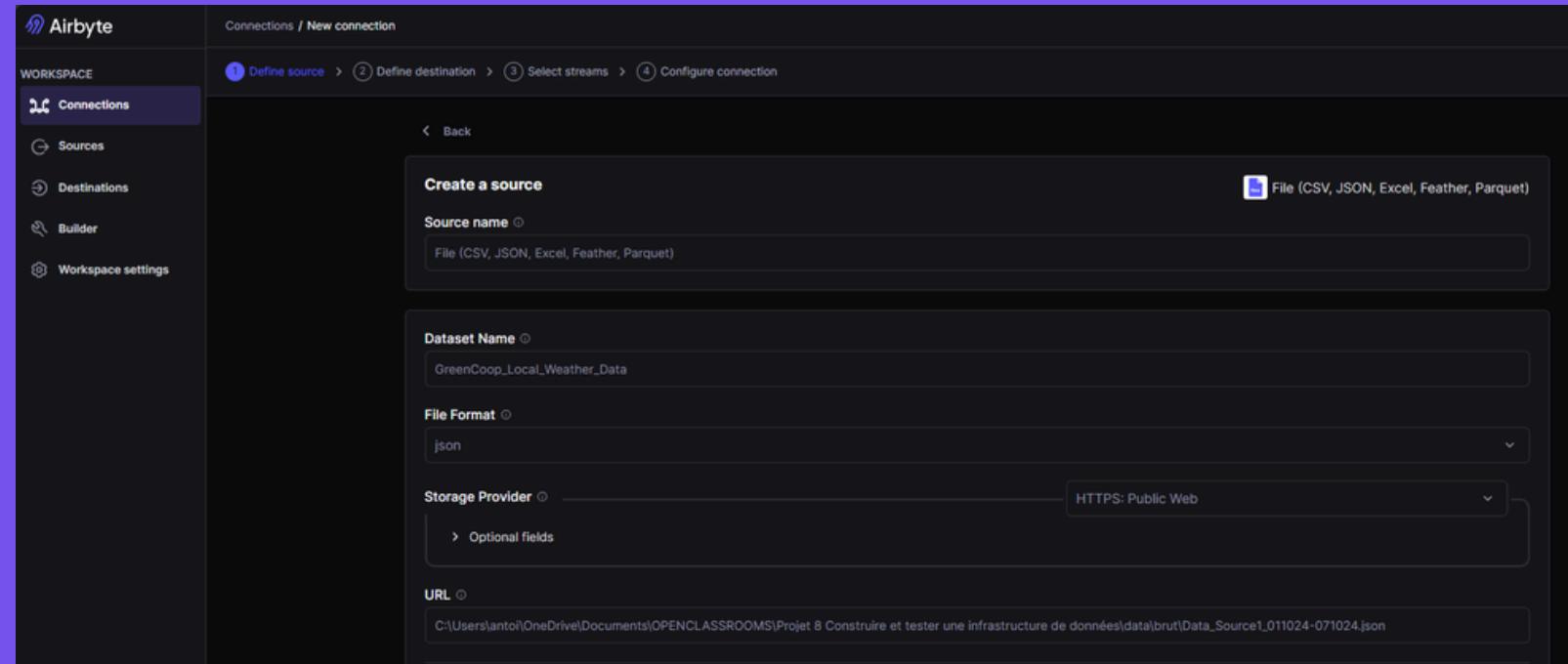
Airbyte Connectors Marketplace

Categories: Airbyte connectors Enterprise connectors

59 connectors

File (CSV, JSON, Excel, Feather, Parquet)

liste des connecteurs disponibles



Airbyte Connections / New connection

WORKSPACE

Sources

Destinations

Builder

Workspace settings

Create a source

Source name

File (CSV, JSON, Excel, Feather, Parquet)

Dataset Name

GreenCoop_LocalWeather_Data

File Format

json

Storage Provider

HTTPS: Public Web

Optional fields

URL

C:\Users\anto\OneDrive\Documents\OPENCLASSROOMS\Projet 8 Construire et tester une infrastructure de données\data\brut\Data_Source1_011024-071024.json

configuration de la source locale

NAME	SOURCE NAME	DESTINATION NAME	FREQUENCY	TAGS	LAST SYNC	ENABLED
excel Ichtegem BE avec date → S3	excel Ichtegem BE avec date	S3	Manual		il y a 12 jours	<input checked="" type="checkbox"/>
excel la madeleine avec date → S3	excel la madeleine avec date	S3	Manual		il y a 12 jours	<input checked="" type="checkbox"/>
Source Fichier Météo Générique JSON ...	Source Fichier Météo Générique JSON	S3	Manual		il y a 12 jours	<input checked="" type="checkbox"/>

3 sources BRUTS à AWS S3

ROADMAP DE LA MISSION

1. Contexte & objectifs
2. Démarche technique
3. Schéma de l'architecture & logigramme
4. Schéma de la BDD
5. Connecteur Airbyte
6. **Reporting**
7. Déploiement AWS

Reporting qualité – données avant / après migration

```
(.venv) PS C:\Users\antoi\OneDrive\Documents\OPENCLASSROOMS\Projet 8 Construire et tester une
infrastructure de données\src> python ./check_data_integrity.py

===== PROFIL AVANT MIGRATION (fichier mongo_ready_measurements.json) =====
Lignes : 4952
Colonnes : ['id_station', 'dh_utc', 'Date', 'DateTime', 'temperature', 'pression', 'humidite',
, 'point_de_rosee', 'visibilite', 'vent_moyen', 'vent_rafales', 'vent_direction', 'pluie_1h',
, 'pluie_3h', 'neige_au_sol', 'nebulosite', 'temps_omm']

Types de colonnes :
id_station          object
dh_utc              object
Date                object
DateTime            object
temperature         float64
pression             float64
humidite             float64
point_de_rosee       float64
visibilite           float64
vent_moyen           float64
vent_rafales         float64
vent_direction       float64
pluie_1h             float64
pluie_3h             float64
neige_au_sol         object
nebulosite           float64
temps_omm             object
dtype: object

Valeurs manquantes (nb et %) :
- id_station      : 0 manquants ( 0.0 %)
- dh_utc          : 2 manquants ( 0.0 %)
- Date            : 2 manquants ( 0.0 %)
- DateTime         : 2 manquants ( 0.0 %)
- temperature     : 2 manquants ( 0.0 %)
- pression         : 2 manquants ( 0.0 %)
- humidite         : 2 manquants ( 0.0 %)
- point_de_rosee   : 2 manquants ( 0.0 %)
- visibilite       : 4892 manquants (98.8 %)
- vent_moyen       : 2 manquants ( 0.0 %)
- vent_rafales     : 1085 manquants (21.9 %)
- vent_direction   : 3832 manquants (77.4 %)
- pluie_1h          : 902 manquants (18.2 %)
- pluie_3h          : 1054 manquants (21.3 %)
- neige_au_sol      : 4952 manquants (100.0 %)
- nebulosite        : 4927 manquants (99.5 %)
- temps_omm         : 4943 manquants (99.8 %)

Doublons sur ['id_station', 'dh_utc'] : 0
```

AVANT



```
===== PROFIL APRÈS MIGRATION (MongoDB weather_db.measurements) =====
Lignes : 4952
Colonnes : ['id_station', 'dh_utc', 'Date', 'DateTime', 'temperature', 'humidite', 'nebulosite', 'neige_au_s
ol', 'pluie_1h', 'pluie_3h', 'point_de_rosee', 'pression', 'temperature', 'temps_omm', 'vent_
direction', 'vent_moyen', 'vent_rafales', 'visibilite']

Types de colonnes :
id_station          object
dh_utc              object
Date                object
DateTime            object
humidite             float64
nebulosite           float64
neige_au_sol         object
pluie_1h             float64
pluie_3h             float64
temps_omm             object
temps_omm             object
vent_direction       float64
temps_omm             object
temps_omm             object
vent_direction       float64
temps_omm             object
vent_direction       float64
vent_moyen           float64
vent_rafales         float64
visibilite           float64
dtype: object

Valeurs manquantes (nb et %) :
- id_station      : 0 manquants ( 0.0 %)
- dh_utc          : 2 manquants ( 0.0 %)
- Date            : 2 manquants ( 0.0 %)
- DateTime         : 2 manquants ( 0.0 %)
- humidite         : 2 manquants ( 0.0 %)
- nebulosite        : 4927 manquants (99.5 %)
- neige_au_sol      : 4952 manquants (100.0 %)
- pluie_1h          : 902 manquants (18.2 %)
- pluie_3h          : 1054 manquants (21.3 %)
- point_de_rosee    : 2 manquants ( 0.0 %)
- pression           : 2 manquants ( 0.0 %)
- temperature        : 2 manquants ( 0.0 %)
- temps_omm          : 4943 manquants (99.8 %)
- vent_direction     : 3832 manquants (77.4 %)
- vent_moyen         : 2 manquants ( 0.0 %)
- vent_rafales        : 1085 manquants (21.9 %)
- visibilite          : 4892 manquants (98.8 %)

Doublons sur ['id_station', 'dh_utc'] : 0
```

APRÈS

Reporting qualité – données avant / après migration

Les contrôles d'intégrité montrent une correspondance totale entre les données sources et les données stockées dans MongoDB.

```
===== COMPARAISON AVANT / APRÈS =====
Colonnes communes (17) : ['Date', 'DateTime', 'dh_utc', 'humidite', 'id_station', 'nebulosite',
                           'neige_au_sol', 'pluie_1h', 'pluie_3h', 'point_de_rosee', 'pression', 'temperature', 'temp
                           s_omm', 'vent_direction', 'vent_moyen', 'vent_rafales', 'visibilite']

Lignes source : 4952
Lignes MongoDB : 4952
```

Reporting sur le temps d'accessibilité aux données

```
(.venv) PS C:\Users\antoi\OneDrive\Documents\OPENCLASSROOMS\Projet 8 Construire et tester une infrastructure de données\src> python .\bench_mongo_latency.py
[1/15] 204 docs en 78.1 ms
[2/15] 204 docs en 72.6 ms
[3/15] 204 docs en 76.3 ms
[4/15] 204 docs en 75.1 ms
[5/15] 204 docs en 71.9 ms
[6/15] 204 docs en 76.0 ms
[7/15] 204 docs en 77.5 ms
[8/15] 204 docs en 89.1 ms
[9/15] 204 docs en 75.8 ms
[10/15] 204 docs en 72.4 ms
[11/15] 204 docs en 68.8 ms
[12/15] 204 docs en 69.3 ms
[13/15] 204 docs en 71.8 ms
[14/15] 204 docs en 71.8 ms
[15/15] 204 docs en 71.8 ms

== RÉSUMÉ LATENCE ==
Docs (dernière requête) : 204
moyenne: 74.6 ms | médiane: 72.6 ms | p95: 89.1 ms | min: 68.8 ms | max: 89.1 ms
Fichier écrit : latency_report_20251030-180826.csv
```

run,ms
1,78.126
2,72.616
3,76.349
4,75.129
5,71.890
6,76.047
7,77.455
8,89.086
9,75.793
10,72.448
11,68.836
12,69.297
13,71.781
14,71.768
15,71.839
17

Résultats clés :

- 15 séries de lectures (204 documents par série)
- Latence moyenne : 74,6 ms
- Médiane : 72,6 ms
- Min : 68,8 ms / Max : 89,1 ms
- Aucune anomalie détectée

La base MongoDB affiche une excellente stabilité et un temps de réponse homogène sur l'ensemble des requêtes.

ROADMAP DE LA MISSION

1. Contexte & objectifs
2. Démarche technique
3. Schéma de l'architecture & logigramme
4. Schéma de la BDD
5. Connecteur Airbyte
6. Reporting
- 7. Déploiement AWS**

Les différents services utilisés d'AWS

Amazon S3 > Compartiments > amzn-s3-mongodb-airbyte > brut-sources/

S3

brut-sources/

Copier l'URI S3

Objets Propriétés

Objets (3)

Rechercher des objets en fonction du préfixe

Afficher les versions

Objet

Nom Type Dernière modification Taille Classe de stockage

Nom	Type	Dernière modification	Taille	Classe de stockage
Data_Source1_011024-071024.json	json	23 Oct 2025 11:10:53 AM CEST	516.2 Ko	Standard
Weather Underground - Ichtegem, BE.xlsx	xlsx	23 Oct 2025 11:10:53 AM CEST	133.0 Ko	Standard
Weather+Underground+-+La+Madeleine,+FR.xlsx	xlsx	23 Oct 2025 11:10:53 AM CEST	131.4 Ko	Standard

Actions ▾ Copier un dossier Charger

Amazon Elastic Container Service > Clusters > aws_ecs_cluster_mongodb_task

Elastic Container Service

aws_ecs_cluster_mongodb_task

ARN: arn:aws:ecs:eu-north-1:257641256173:cluster/aws_ecs_cluster_mongodb_task Status: Active CloudWatch monitoring: Default Registered container instances: -

Services Tasks

Draining Active Pending Running

Services Tasks Infrastructure Updated Metrics Scheduled tasks Configuration Event history Tags

Services (1) Info Last updated: November 5, 2025, 19:04 (UTC+1:00) Manage tags Update Delete service Create

Filter services by value Filter launch type Any launch type Filter scheduling strategy Any scheduling strategy

Service name	ARN	Status	Schedu...	Launch...	Task de...	Deployments and tasks	Last deployment	Created at
mongo-service	arn:aws:ecs:eu-r...	Active	REPLICA	-	mongo-ta...	1/1 Tasks running	Completed View	6 days ago

24

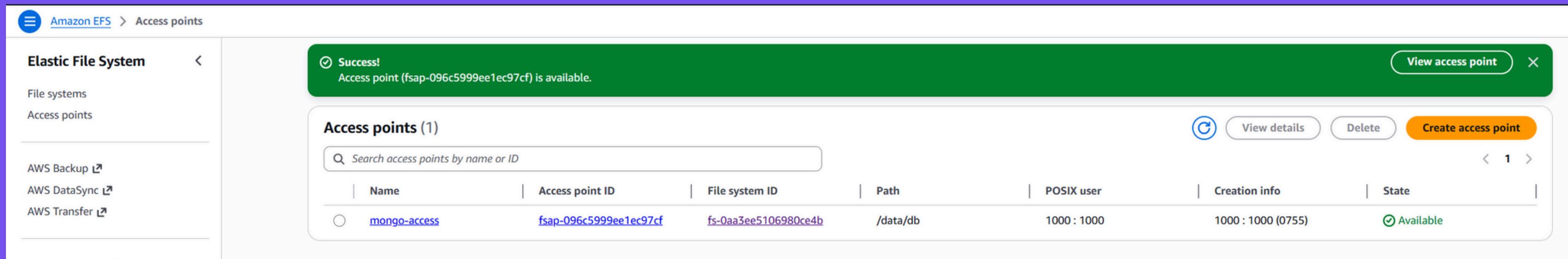
Déploiement MongoDB sur AWS ECS

The screenshot shows the AWS Elastic Container Service (ECS) console. The top navigation bar includes links for 'Amazon Elastic Container Service', 'Clusters', 'aws_ecs_cluster_mongodb_task', and 'Services'. The main title 'Elastic Container Service' is displayed with a subtitle 'aws_ecs_cluster_mongodb_task'. A green success message box states 'mongo-service has been deployed successfully.' Below this, the 'Cluster overview' section shows the ARN (arn:aws:ecs:eu-north-1:257641256173:cluster/aws_ecs_cluster_mongodb_task), Status (Active), CloudWatch monitoring (Default), and Registered container instances (none). The 'Services' tab is selected, showing one active service named 'mongo-service'. The 'Tasks' section indicates 1 active task and 1 pending task. The 'Metrics' tab is also visible. At the bottom, the 'Services' table lists the deployed service with details like ARN, Status (Active), and Last deployment (3 minutes ago).

Service name	ARN	Status	Scheduled tasks	Launch...	Task de...	Deployments and tasks	Last deployment	Created at
mongo-service	arn:aws:ecs:eu-r	Active	REPLICA	-	mongo-ta...	1/1 Tasks running	Completed View	3 minutes ago

Le service mongo-service a été déployé avec succès sur le cluster aws_ecs_cluster_mongodb_task. Le conteneur est actif et 1 tâche est en cours d'exécution, confirmant la disponibilité du service MongoDB dans l'environnement AWS.

Création du volume partagé – Amazon EFS



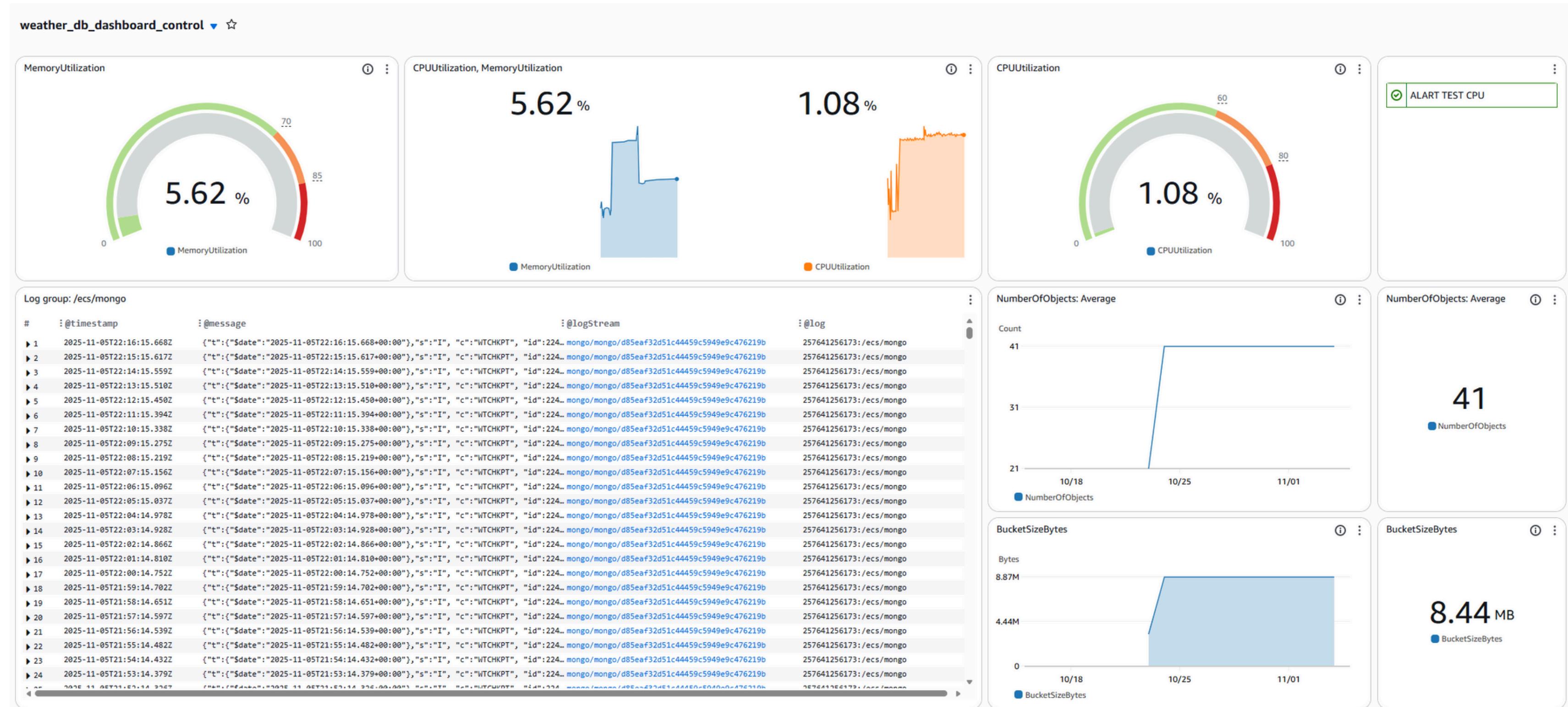
The screenshot shows the AWS EFS Access points interface. On the left, there's a sidebar with 'Elastic File System' selected. The main area has a green success message: 'Success! Access point (fsap-096c5999ee1ec97cf) is available.' Below this, a table lists the access point 'mongo-access' with details: Access point ID is fsap-096c5999ee1ec97cf, File system ID is fs-0aa3ee5106980ce4b, Path is /data/db, POSIX user is 1000 : 1000, Creation info is 1000 : 1000 (0755), and State is Available.

Name	Access point ID	File system ID	Path	POSIX user	Creation info	State
mongo-access	fsap-096c5999ee1ec97cf	fs-0aa3ee5106980ce4b	/data/db	1000 : 1000	1000 : 1000 (0755)	Available

Le point d'accès mongo-access a été créé avec succès dans le système de fichiers EFS.

Il permet le montage persistant du dossier /data/db utilisé par MongoDB pour stocker durablement les données du conteneur.

Supervision de MongoDB sur AWS CloudWatch



Suivi en temps réel des performances du conteneur MongoDB (ECS)

MERCI