

Module M2 – Big Data et Architectures Distribuées

Introduction : Bienvenue chez StreamFlix

Vous venez d'être recruté(e) comme **Data Engineer** chez *StreamFlix*, une entreprise mondiale de streaming vidéo.

Chaque jour, des **pétabytes de données** sont générés : vidéos, logs de navigation, historiques, recommandations...

Mais le système est à bout :

- TROP de données,
- TROP d'utilisateurs,
- TROP de lenteurs...


Le CTO vous confie une mission critique :


"Tu dois refondre toute notre architecture Big Data.
Stockage, traitement, bases de données, résilience.
Le monde entier compte sur toi."

À la fin de ce cours, vous serez capable de :

- Comprendre les **principes d'une architecture distribuée**.
- Identifier les **besoins spécifiques du Big Data** (volume, vitesse, variété, etc.).
- Comparer et mettre en œuvre des **systèmes de stockage distribués**.
- Choisir des **bases de données NoSQL** adaptées à différents cas d'usage.
- Utiliser des **plateformes de calcul massif** comme Hadoop ou Spark.
- Garantir la **scalabilité** et la **tolérance aux pannes** d'un système distribué.

Les différents chapitres abordés dans ce module

 Séance	Thème Technique	Scénario narratif chez StreamFlix
Chap 1	Introduction au Big Data et à l'architecture distribuée	Pourquoi une machine ne suffit plus pour gérer la charge ?
Chap 2	Systèmes de fichiers distribués (ext4, Ceph, GPFS ...)	Comment stocker tous les films, logs et backups ?
Chap 3	Les bases de données et le NoSQL (clé-valeur, document, colonne, graphe)	Les grands principes du NoSQL
Chap 4	Focus Neo4j	Chaque techno répond-elle à un besoin spécifique ?
Chap 5	Focus MongoDB	Chaque techno répond-elle à un besoin spécifique ?

 Séance	Thème Technique	Scénario narratif chez StreamFlix
Chap 6	Focus Redis	Chaque techno répond-elle à un besoin spécifique ?
Chap 7	Les systèmes de fichiers distribués	Les systèmes de fichiers locaux au stockage objet.
Chap 8	Hadoop : MapReduce, YARN, HDFS	Premiers traitements sur un système de batch parallélisé
Chap 9	Apache Spark : RDD, DataFrame, MLlib	Une plateforme polyvalente pour le traitement parallélisé

Préparer son environnement de Travail

Les travaux pratiques du module s'appuieront sur un environnement conteneurisé sous Docker. Toutes les applications que nous utiliserons feront parties du catalogue Docker Hub et nous détaillerons dans la suite de ce README les étapes à suivre :

- 1 - Prérequis matériel
- 2 - Installer Docker Desktop
- 3 - Télécharger les images docker
- 4 - Installation de NoSQLBOOSTER

La création d'un compte Docker Hub est fortement recommandé pour les accès aux images :

<https://app.docker.com/signup>

1 - Prérequis

Même si la conteneurisation à l'avantage d'être légère, les prérequis matériels pour suivre ce module sont les suivants :

- 4Go de RAM
- 4 cores
- 5 Go d'espace disque libre

En termes de privilèges, vous devrez disposer des droits d'administrateur sur votre machine. Vous devrez disposer d'un client git ou d'un IDE disposant des fonctionnalités git.

2 - Installation de Docker desktop

Docker fournit un logiciel Docker Desktop pour déployer très facilement et rapidement des applications par le biais d'images docker. Vous pourrez télécharger Docker Desktop à l'url suivante :

<https://www.docker.com/products/docker-desktop>

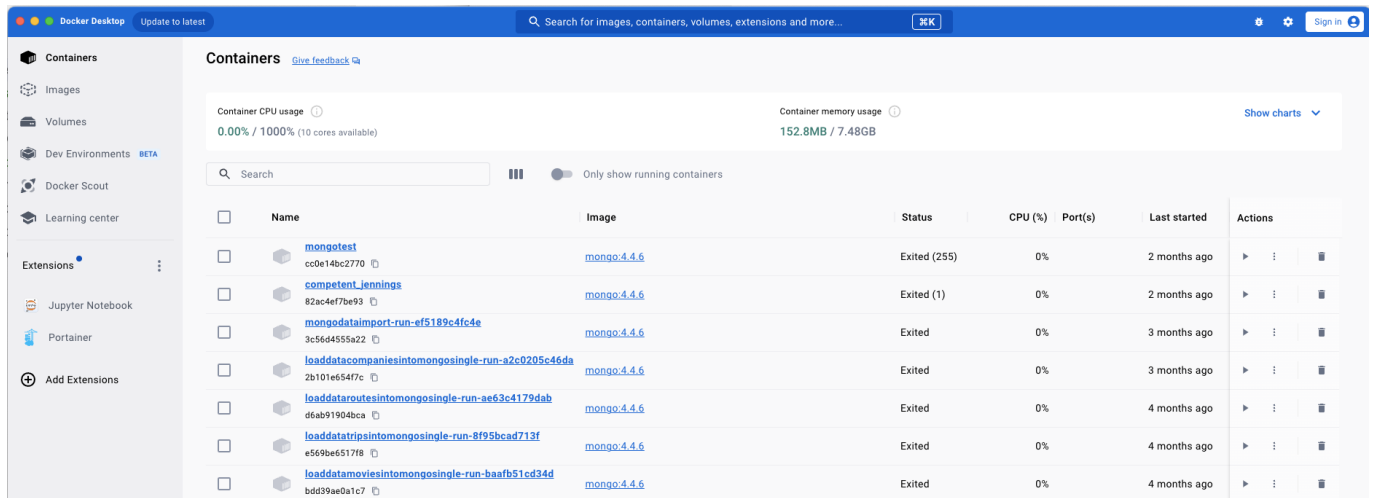
NOTE

Selon la version de votre Windows, il peut vous être demandé d'installer WSL 2.

Le tableau de bord ci-dessous s'affichera :

WARNING

Si c'est la première fois que vous installez Docker Desktop le tableau des containers sera vide mais ne vous inquiétez pas c'est que temporaire.



Pour travailler avec docker on pourrait utiliser l'interface graphique fournie par Docker Desktop mais pour des raisons de facilité et de cohérence d'environnement il est fortement recommandé d'utiliser les invites de commande.

Ouvrez un terminal ou un powershell si vous êtes sous Windows

WARNING

Si vous êtes sur Windows, ne confondez pas MSDOS et Powershell.

WARNING

Pour ceux qui sont sous Linux, il est probable que le plugin compose ne soit pas installé en même temps que docker desktop. Pour vérifier que le plugin compose est bien installé avec docker, exécutez dans un terminal la commande suivante : `docker compose --help`

Si vous n'obtenez pas l'aide de `docker compose` alors vous devrez installer le plugin. Le guide d'installation est par ici : <https://docs.docker.com/compose/install/linux/>

3 - Télécharger les images docker

Dans le répertoire de votre choix `$monrepertoire`, cloner le répertoire gitlab suivant :

```
cd $monrepertoire
git clone https://github.com/oaidel38090/BG_M2_2025.git
```

Positionner vous à présent dans le répertoire **BG_M2_2025/docker** puis télécharger l'ensemble des images docker nécessaire à la formation avec la commande suivante :

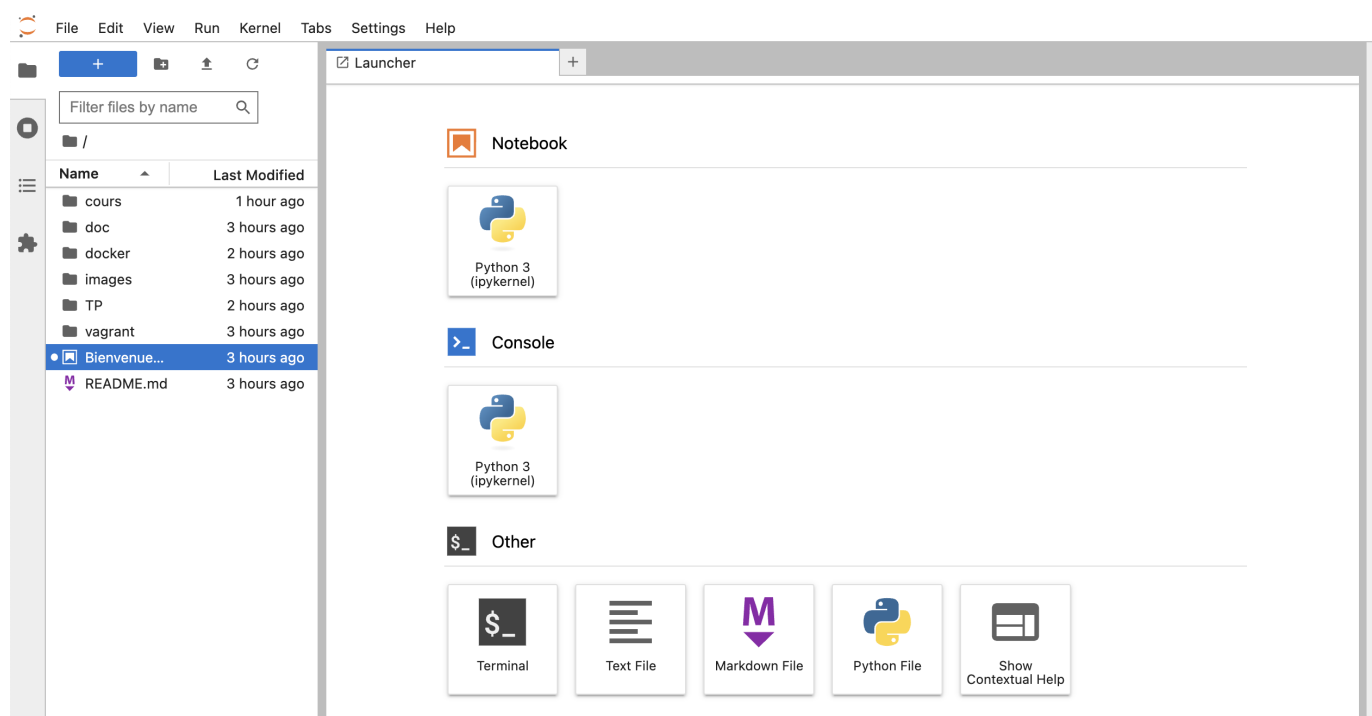
```
cd $monrepertoire/BG_M2_2025/docker
docker compose pull
```

L'ensemble des cours et TP est disponible dans le répertoire gitlab.

Pour vérifier que votre environnement est opérationnel lancer la commande suivante :

```
cd $monrepertoire/BG_M2_2025/docker
docker compose up -d jupyterlab
```

Cette commande va lancer l'application jupyterlab sur votre ordinateur et sera accessible à l'url <http://127.0.0.1:8888/lab>. La page suivante devrait apparaître :



Pour arrêter l'application jupyterlab, il vous suffit d'exécuter la commande suivante :

```
docker stop jupyterlab
```

4 - Installation de NoSQLBOOSTER

NosqlBooster est un client graphique gratuit pour faciliter les opérations sur MongoDB. Il est téléchargeable à l'url suivante :

<https://www.nosqlbooster.com/downloads>

Télécharger et installer le sur votre ordinateur.

Bravo, vous êtes à présent prêt à suivre le module.