

# COMPTE RENDU

## TP3



CHAIMAE EL MATTARI

304

# ÉTAPE 1 – « VIVE LES PETITS PRODUCTEURS »

## • Modification de la configuration :

- J'ai effectué la modification nécessaire dans le fichier server.js à la ligne 24, où la fonction getStringMessage() était utilisée avec un paramètre qui n'utilisait pas le champ NUMBER\_WORD du fichier .env.
- J'ai réalisé les modifications nécessaires dans le fichier config.js et dans server.js pour intégrer le champ NUMBER\_WORD dans la configuration.

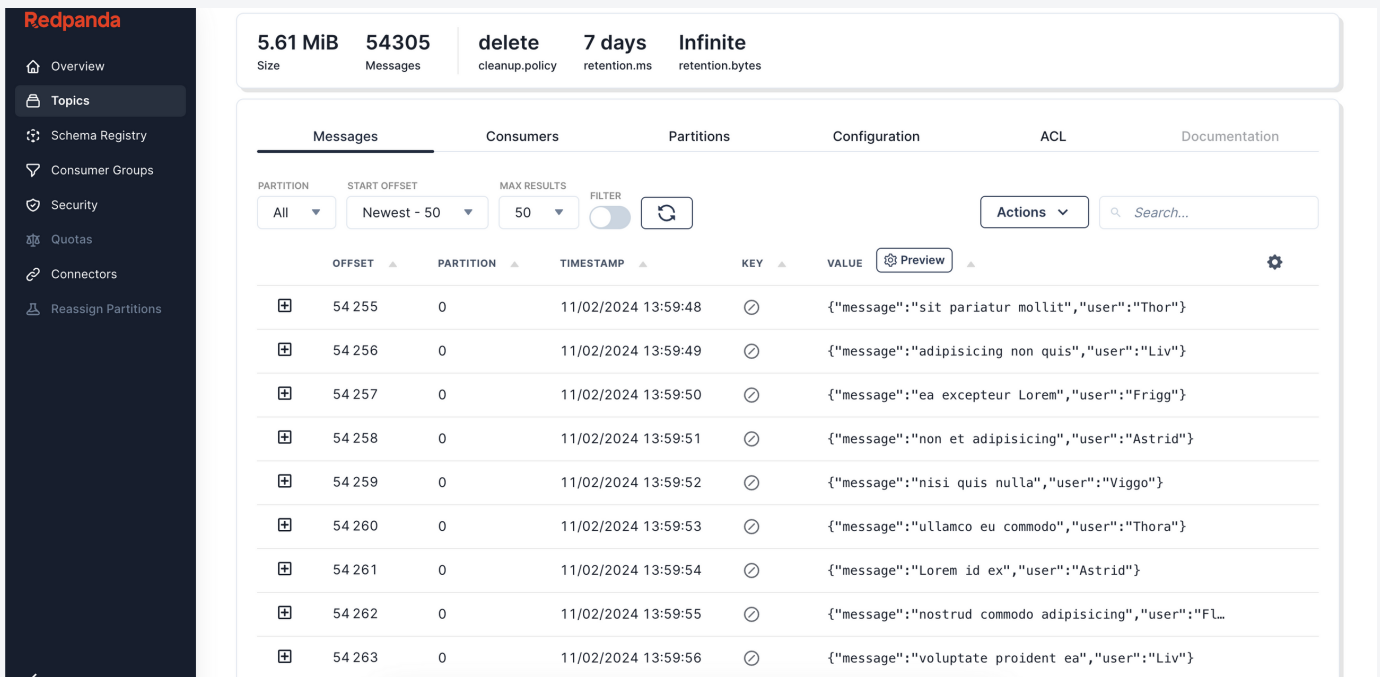
## • Construction et démarrage du conteneur Docker :

- J'ai réussi à construire l'image Docker du producteur en utilisant le fichier Dockerfile.
- Le conteneur a été lancé, et j'ai observé des messages de débogage dans la console du conteneur

## • Difficultés :

J'ai rencontré des difficultés lors de la création et du lancement du conteneur Docker, avec des erreurs de connexion au broker (ECONNREFUSED 127.0.0.1:19092).

Malheureusement, malgré mes efforts, ces erreurs persistaient, entraînant une impossibilité de connexion au broker.



The screenshot displays the Redpanda web interface. On the left is a dark sidebar with navigation links: Overview, Topics, Schema Registry, Consumer Groups, Security, Quotas, Connectors, and Reassign Partitions. The main panel shows the 'Messages' tab for a specific topic. At the top, summary statistics are provided: 5.61 MiB Size, 54305 Messages, delete cleanup.policy, 7 days retention.ms, and Infinite retention.bytes. Below this, a table lists individual messages with columns for PARTITION, START OFFSET, MAX RESULTS, FILTER, and a table of message details (OFFSET, PARTITION, TIMESTAMP, KEY, VALUE). The messages are JSON objects containing 'message' and 'user' fields. A 'Preview' button is visible next to the VALUE column header.

PARTITION	START OFFSET	MAX RESULTS	FILTER
All	Newest - 50	50	<input type="checkbox"/>

OFFSET	PARTITION	TIMESTAMP	KEY	VALUE
54 255	0	11/02/2024 13:59:48	🔗	{"message":"sit pariatur mollit","user":"Thor"}
54 256	0	11/02/2024 13:59:49	🔗	{"message":"adipisicing non quis","user":"Liv"}
54 257	0	11/02/2024 13:59:50	🔗	{"message":"ea excepteur Lorem","user":"Frigg"}
54 258	0	11/02/2024 13:59:51	🔗	{"message":"non et adipisicing","user":"Astrid"}
54 259	0	11/02/2024 13:59:52	🔗	{"message":"nisi quis nulla","user":"Viggo"}
54 260	0	11/02/2024 13:59:53	🔗	{"message":"ullamco eu commodo","user":"Thora"}
54 261	0	11/02/2024 13:59:54	🔗	{"message":"Lorem id ex","user":"Astrid"}
54 262	0	11/02/2024 13:59:55	🔗	{"message":"nostrud commodo adipisicing","user":"FL..."}
54 263	0	11/02/2024 13:59:56	🔗	{"message":"voluptate proident ea","user":"Liv"}

Console redpanda avec les différents messages

## ÉTAPE 2 – STAR DE CINÉMA : « 26L / 100 KM » - QUI SUIS-JE ?

### • Création du Projet :

- J'ai créé un nouveau projet distinct pour le consommateur.

### • Fonction de Connexion :

- Dans le fichier consommateur.js, j'ai mis en place la fonction connexion() qui utilise la méthode connect() de l'objet consommateur de Kafka.js. Cette fonction assure une connexion réussie au broker RedPanda et à l'abonnement du consommateur au Topic spécifié. La console affiche un message confirmant la connexion.

### • Identification de la Fonction de Consommation :

- En consultant la documentation de [Kafka.js](#), j'ai identifié la méthode run() comme celle permettant au consommateur de traiter chaque message.

### • Affichage du Contenu des Messages :

- J'ai utilisé la méthode run() de Kafka.js pour implémenter la fonction subscribeAndRun() qui consomme chaque message du Topic. Le contenu de chaque message est ensuite affiché dans la console

### • Conversion du Timestamp :

- J'ai ajouté la fonction formatTimestamp() pour formater le timestamp de chaque message en un format lisible.

### • Difficulté :

- J'ai rencontré une difficulté lors du lancement du consommateur, provoquée par une tentative d'abonnement au Topic pendant que le consommateur était déjà en cours d'exécution. Pour résoudre cela, j'aurais besoin d'ajuster le flux d'exécution pour garantir que la souscription n'a lieu qu'après une connexion réussie.

```
{
  key: null,
  value: '{"message":"ex velit est","user":"Astrid"}',
  headers: {}
}

{
  key: null,
  value: '{"message":"reprehenderit nulla eiusmod","user":"Thora"}',
  headers: {}
}

{
  key: null,
  value: '{"message":"non minim et","user":"Thor"}',
  headers: {}
}

{
  key: null,
  value: '{"message":"veniam veniam dolore","user":"Frigg"}',
  headers: {}
}

{
  key: null,
  value: '{"message":"est enim nostrud","user":"Thora"}',
  headers: {}
}

{
  key: null,
  value: '{"message":"anim elit inure","user":"Harald"}',
  headers: {}
}
```

Cluster > Consumer Groups > **my-consumer-group** ⓘ ↺

✓ **Stable**  
State

**1**  
Assigned Partitions

**consumer**  
Protocol Type

**0**  
Coordinator ID

**0**  
Total Lag

## ÉTAPE 3 –« ARE YOU REDIS »?

- J'ai réussi à découper les messages en mots grâce à la fonction split, ce qui me permet d'identifier chaque mot individuel et de le traiter séparément.
- Concernant Redis, j'ai récupéré avec succès le projet depuis GitHub, ce qui m'a permis de comprendre comment incrémenter manuellement les compteurs associés à chaque mot. Cependant, bien que j'aie pu faire fonctionner le projet récupéré pour l'incrémementation à la main, je n'ai pas encore réussi à intégrer harmonieusement cette logique avec mon service consommateur.

```
{ key: null, word: 'mollit',"user":"Astrid"}', headers: {} }
{ key: null, word: '{"message":"anim', headers: {} }
{ key: null, word: 'reprehenderit', headers: {} }
{ key: null, word: 'proident',"user":"Astrid"}', headers: {} }
{ key: null, word: '{"message":"ullamco', headers: {} }
{ key: null, word: 'ullamco', headers: {} }
{ key: null, word: 'anim',"user":"Thora"}', headers: {} }
{ key: null, word: '{"message":"sint', headers: {} }
{ key: null, word: 'ipsum', headers: {} }
{ key: null, word: 'ipsum',"user":"Frigg"}', headers: {} }
{ key: null, word: '{"message":"commodo', headers: {} }
{ key: null, word: 'magna', headers: {} }
{ key: null, word: 'qui',"user":"Viggo"}', headers: {} }
p_exo2 > src > redpanda > JS consommateur.js > startConsumer() > eachM
```

### Occurrence Des Mots

Nom du mot	Nombre d'occurrence
chaimae	6
bonjour	6