## Contexte et objectif

Ce projet vise à mettre en place un pipeline complet pour la gestion, le traitement, la transformation et la visualisation de données issues de l'**Open Data**. Les données utilisées proviennent du site data.gouv.fr, et concernent les **stations de taxi**. Le projet repose sur une architecture distribuée utilisant Kafka, des services Spring Boot, un système de base de données, et une application Flutter pour la visualisation sur une carte Google Maps.

## 1. Données Open Data

### 1.1. Source des données

Les données brutes ont été récupérées depuis data.gouv.fr, une plateforme fournissant des jeux de données publiques en France. Ces données contiennent des informations sur les stations de taxi, notamment :

- Identifiant de la station.
- Nom de la station.
- Adresse.
- Code INSEE.
- Nombre d'emplacements.
- Latitude et longitude.
- Statut de la station (active ou inactive).

### 1.2. Nettoyage et préparation des données

Avant leur ingestion dans le pipeline, les données ont été nettoyées et préparées à l'aide d'**OpenRefine** :

- **Correction des erreurs** dans les champs (orthographe, format des adresses, etc.).
- **Suppression des doublons** pour éviter les redondances.
- **Validation des coordonnées GPS** pour garantir une précision sur la carte.
- **Exportation** du fichier final au format **CSV**.

## 2. Service Producteur Kafka

Le **service producteur Kafka** a pour rôle de lire les données nettoyées (au format CSV) et de les envoyer dans un **topic Kafka** nommé taxi-stations.raw. Voici les étapes principales :

1. **Lecture des Données** : Utilisation de la bibliothèque OpenCSV pour parser le fichier CSV.
2. **Transformation en JSON** : Conversion de chaque ligne du CSV en objet JSON.
3. **Envoi vers Kafka** : Les messages JSON sont envoyés au topic taxi-stations.raw via un producteur Kafka.

- **Technologies Utilisées**
  - Spring Boot.
  - Kafka Producer API.
  - OpenCSV pour le parsing du fichier.

## 3. Service Consommateur Kafka

Le **service consommateur Kafka** est un composant clé qui consomme les messages JSON envoyés par le producteur et les stocke dans une base de données pour une utilisation ultérieure.

### 3.1. Fonctionnement

1. **Consommation Kafka** : Lecture des messages JSON depuis le topic taxi-stations.raw.

2. **Stockage en base de bonnées** :
    - **MySQL** est utilisé comme base de données principale.
    - En cas d'indisponibilité de MySQL, le service bascule automatiquement vers une base **H2** persistante grâce à une logique dans l'application.
3. **Exposition des Données** :
    - Une API RESTful expose les données stockées dans la base.

### 3.2. Gestion des bases de données

1. **Configuration Multiple** :
    - Un fichier application.properties global est utilisé par défaut.
    - Des fichiers spécifiques (application-mysql.properties et application-h2.properties) configurent respectivement MySQL et H2.
2. **Détection Automatique** :
    - Le service teste la disponibilité de MySQL au démarrage.
    - Si MySQL est indisponible, il bascule automatiquement sur H2.

### 3.3. API RESTful

L'API RESTful permet :

- **Récupération de toutes les stations**.
- **Recherche par identifiant**.
- **Pagination des résultats** pour optimiser les requêtes.

## 4. Service Kafka Streams

Le **service Kafka Streams** est un composant autonome qui traite les données brutes issues du topic taxi-stations.raw. Son rôle est de transformer, regrouper et enrichir les données avant de les produire dans de nouveaux topics Kafka.

### 4.1. Fonctionnalités

1. **Regroupement par Statut** :
    - Compte le nombre de stations en fonction de leur statut (active ou inactive).
    - Produit les résultats dans le topic stations.grouped.by.status.
2. **Filtrage pour Paris** :
    - Filtre les stations situées à Paris à partir des champs address et latitude/longitude.
    - Produit les résultats filtrés dans le topic paris.stations.

3. **Création automatique des topics** :
    - Les topics nécessaires (taxi-stations.raw, stations.grouped.by.status, paris.stations) sont créés dynamiquement au démarrage grâce à l'API Kafka Admin.

### 4.2. Technologies utilisées

- Kafka Streams API.
- Jackson pour le parsing JSON.
- Kafka AdminClient pour la gestion des topics.

## 5. Application Flutter

Une **application mobile Flutter** a été développée pour visualiser les données exposées par le consommateur via son API RESTful. Cette application affiche les stations de taxi sur une carte interactive.

### 5.1. Fonctionnalités

1. **Récupération des Données** :
    - L'application consomme les endpoints de l'API pour récupérer les données des stations.
2. **Affichage sur Google Maps** :
    - Les stations sont affichées sur une carte Google Maps avec des marqueurs interactifs.
3. **Recherche** :
    - L'utilisateur peut rechercher une station spécifique par son nom ou son adresse.
4. **Navigation** :
    - Possibilité d'obtenir des itinéraires vers une station sélectionnée.

### 5.2. Technologies utilisées

- Flutter pour le développement mobile.
- Google Maps API pour l'intégration cartographique.
- http pour consommer les endpoints REST.

## 6. Architecture Globale

**Schéma :**

- **Producteur Kafka** : Topic d'entrée : taxi-stations.raw.
- **Kafka Streams** : Lit taxi-stations.raw et produit dans :
    - stations.grouped.by.status (regroupement par statut).

    - paris.stations (stations filtrées pour Paris).
- **Consommateur Kafka** :
    - Lit stations.grouped.by.status et paris.stations.
    - Stocke dans MySQL (ou H2 si indisponible).
    - Expose une API RESTful.
- **Application Flutter** :
    - Consomme l'API pour afficher les données sur Google Maps.

## Conclusion

Ce projet illustre un pipeline complet, allant de l'ingestion et de la transformation des données brutes à leur visualisation interactive. Les principaux atouts du projet incluent :

- **Utilisation de données publiques Open Data**.
- **Architecture distribuée** basée sur Kafka.
- **Gestion de la persistance** avec basculement automatique entre MySQL et H2.
- **Flexibilité et scalabilité** grâce à Kafka Streams.
- **Accessibilité** via une application mobile intuitive.

Le projet démontre une intégration réussie de technologies modernes et répond efficacement aux besoins de gestion et de visualisation de données en temps réel.

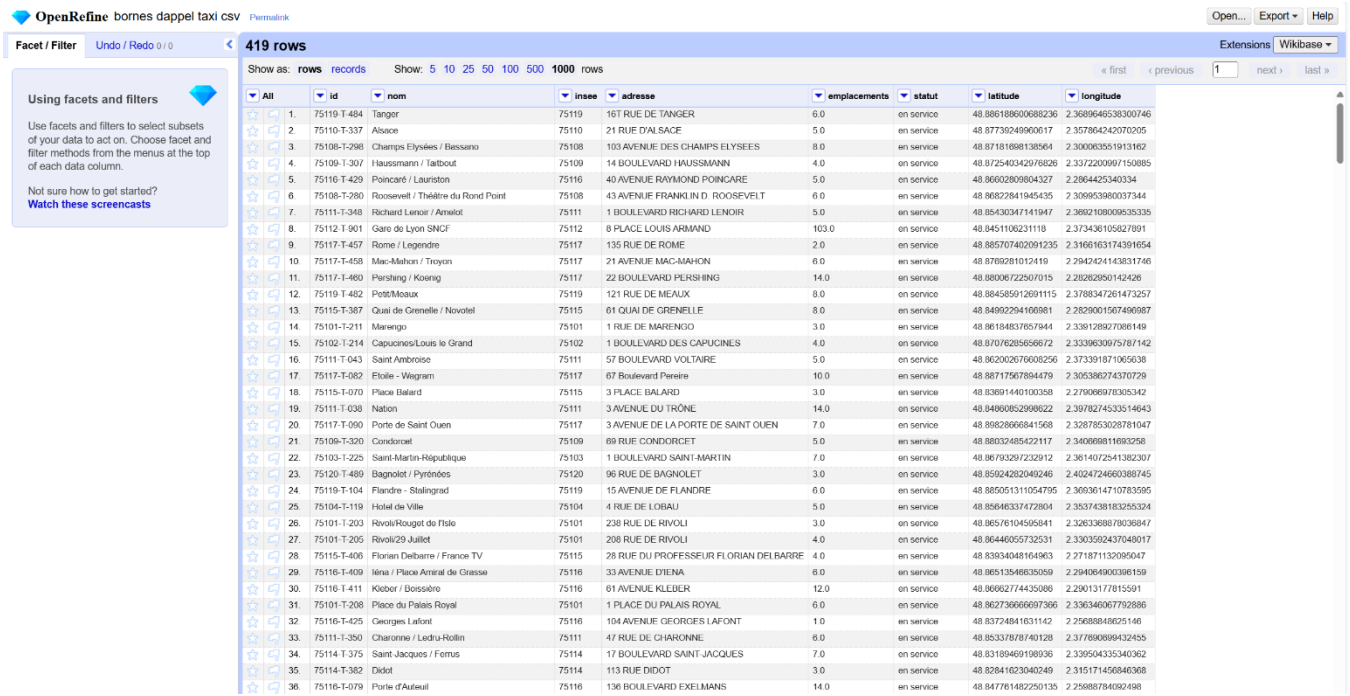
## Annexe 1 : Console de démarrage du serveur Kafka, Zookeeper, consumer console

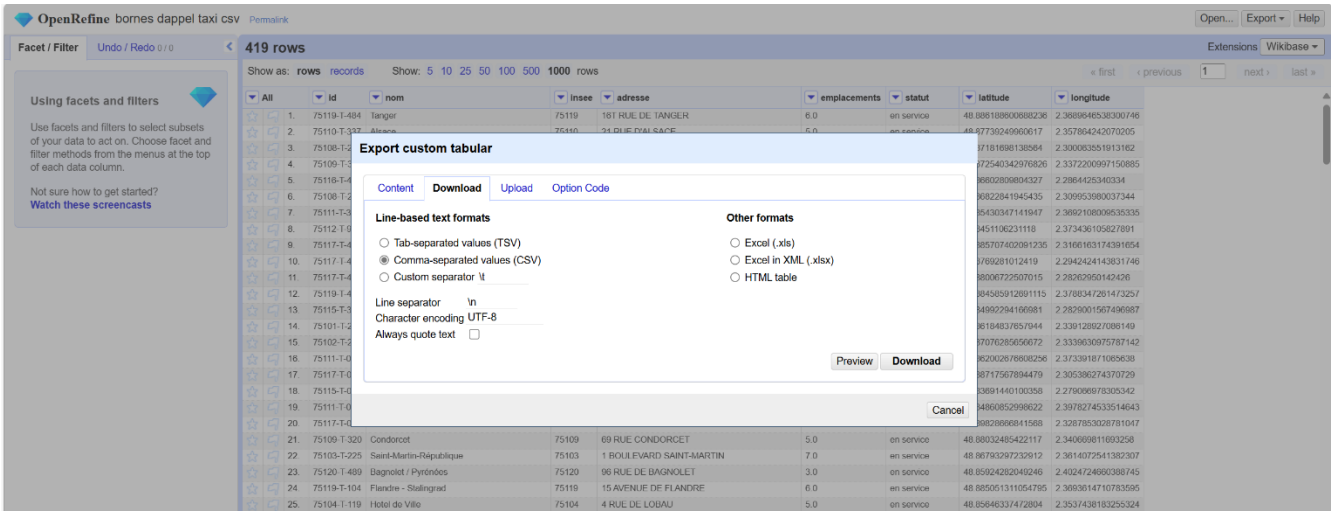
```
c:\kafka>.\bin\windows\kafka-server-start.bat .\config\server.properties
[2025-01-19 09:07:54,110] INFO Registered kafka:type=kafka.Log4jController MBean (kafka.utils.Log4jControllerRegistratio
n$)
[2025-01-19 09:07:54,282] INFO Setting -D jdk.tls.rejectClientInitiatedRenegotiation=true to disable client-initiated TL
S renegotiation (org.apache.zookeeper.common.X509Util)
[2025-01-19 09:07:54,361] INFO starting (kafka.server.KafkaServer)
[2025-01-19 09:07:54,361] INFO Connecting to zookeeper on localhost:2181 (kafka.server.KafkaServer)
[2025-01-19 09:07:54,383] INFO [ZooKeeperClient Kafka server] Initializing a new session to localhost:2181. (kafka.zooke
eper.ZooKeeperClient)
[2025-01-19 09:07:59,364] INFO Client environment:zookeeper.version=3.8.4-9316c2a7a97e1666d8f4593f34dd6fc36ecc436c, buil
t on 2024-02-12 22:16 UTC (org.apache.zookeeper.ZooKeeper)
[2025-01-19 09:07:59,365] INFO Client environment:host.name=SAVED (org.apache.zookeeper.ZooKeeper)
[2025-01-19 09:07:59,365] INFO Client environment:java.version=23.0.1 (org.apache.zookeeper.ZooKeeper)
[2025-01-19 09:07:59,365] INFO Client environment:java.vendor=Oracle Corporation (org.apache.zookeeper.ZooKeeper)
[2025-01-19 09:07:59,365] INFO Client environment:java.home=C:\Program Files\Java\jdk-23 (org.apache.zookeeper.ZooKeeper
)
[2025-01-19 09:07:59,365] INFO Client environment:java.class.path=c:\kafka\libs\activation-1.1.1.jar;c:\kafka\libs\aopal
liance-repackaged-2.6.1.jar;c:\kafka\libs\argparse4j-0.7.0.jar;c:\kafka\libs\audience-annotations-0.12.0.jar;c:\kafka\li
bs\caffeine-2.9.3.jar;c:\kafka\libs\commons-beanutils-1.9.4.jar;c:\kafka\libs\commons-cli-1.4.jar;c:\kafka\libs\commons-
collections-3.2.2.jar;c:\kafka\libs\commons-digester-2.1.jar;c:\kafka\libs\commons-io-2.14.0.jar;c:\kafka\libs\commons-l
ang3-3.12.0.jar;c:\kafka\libs\commons-logging-1.2.jar;c:\kafka\libs\commons-validator-1.7.jar;c:\kafka\libs\connect-api-
3.9.0.jar;c:\kafka\libs\connect-basic-auth-extension-3.9.0.jar;c:\kafka\libs\connect-file-3.9.0.jar;c:\kafka\libs\connec
t-json-3.9.0.jar;c:\kafka\libs\connect-mirror-3.9.0.jar;c:\kafka\libs\connect-mirror-client-3.9.0.jar;c:\kafka\libs\conn
ect-runtime-3.9.0.jar;c:\kafka\libs\connect-transforms-3.9.0.jar;c:\kafka\libs\error_prone_annotations-2.10.0.jar;c:\kaf
ka\libs\hk2-api-2.6.1.jar;c:\kafka\libs\hk2-locator-2.6.1.jar;c:\kafka\libs\hk2-utils-2.6.1.jar;c:\kafka\libs\jackson-an
```

```
der)
[2025-01-19 09:07:35,417] INFO zookeeper.DigestAuthenticationProvider.enabled = true (org.apache.zookeeper.server.auth.D
igestAuthenticationProvider)
[2025-01-19 09:07:35,419] INFO zookeeper.snapshot.trust.empty : false (org.apache.zookeeper.server.persistence.FileTxnSn
apLog)
[2025-01-19 09:07:35,427] INFO  (org.apache.zookeeper.server.ZooKeeperServer)
[2025-01-19 09:07:35,427] INFO   ______                  _                                          (org.apache.zookeep
er.server.ZooKeeperServer)
[2025-01-19 09:07:35,427] INFO  |___  /                 | |                                         (org.apache.zookeep
er.server.ZooKeeperServer)
[2025-01-19 09:07:35,427] INFO     / /    ___     ___   | | __   ___    ___    _ __     ___    _ __    (org.apache.zookeep
er.server.ZooKeeperServer)
[2025-01-19 09:07:35,428] INFO    / /    / _ \   / _ \  | |/ /  / _ \  / _ \ | '_ \   / _ \ | '__| (org.apache.zookeeper
.server.ZooKeeperServer)
[2025-01-19 09:07:35,428] INFO   / /__  | (_) | | (_) | |   <  |  __/ |  __/ | |_) | |  __/ | |     (org.apache.zookeepe
r.server.ZooKeeperServer)
[2025-01-19 09:07:35,428] INFO  /_____|  \___/   \___/  |_|\_\  \___|  \___| | .__/   \___| |_| (org.apache.zookeeper.se
rver.ZooKeeperServer)
[2025-01-19 09:07:35,428] INFO                                               | |                  (org.apache.zookee
per.server.ZooKeeperServer)
[2025-01-19 09:07:35,428] INFO                                               |_|                  (org.apache.zookee
per.server.ZooKeeperServer)
[2025-01-19 09:07:35,428] INFO  (org.apache.zookeeper.server.ZooKeeperServer)
[2025-01-19 09:07:40,377] INFO Server environment:zookeeper.version=3.8.4-9316c2a7a97e1666d8f4593f34dd6fc36ecc436c, buil
t on 2024-02-12 22:16 UTC (org.apache.zookeeper.server.ZooKeeperServer)
[2025-01-19 09:07:40,377] INFO Server environment:host.name=SAVED (org.apache.zookeeper.server.ZooKeeperServer)
```

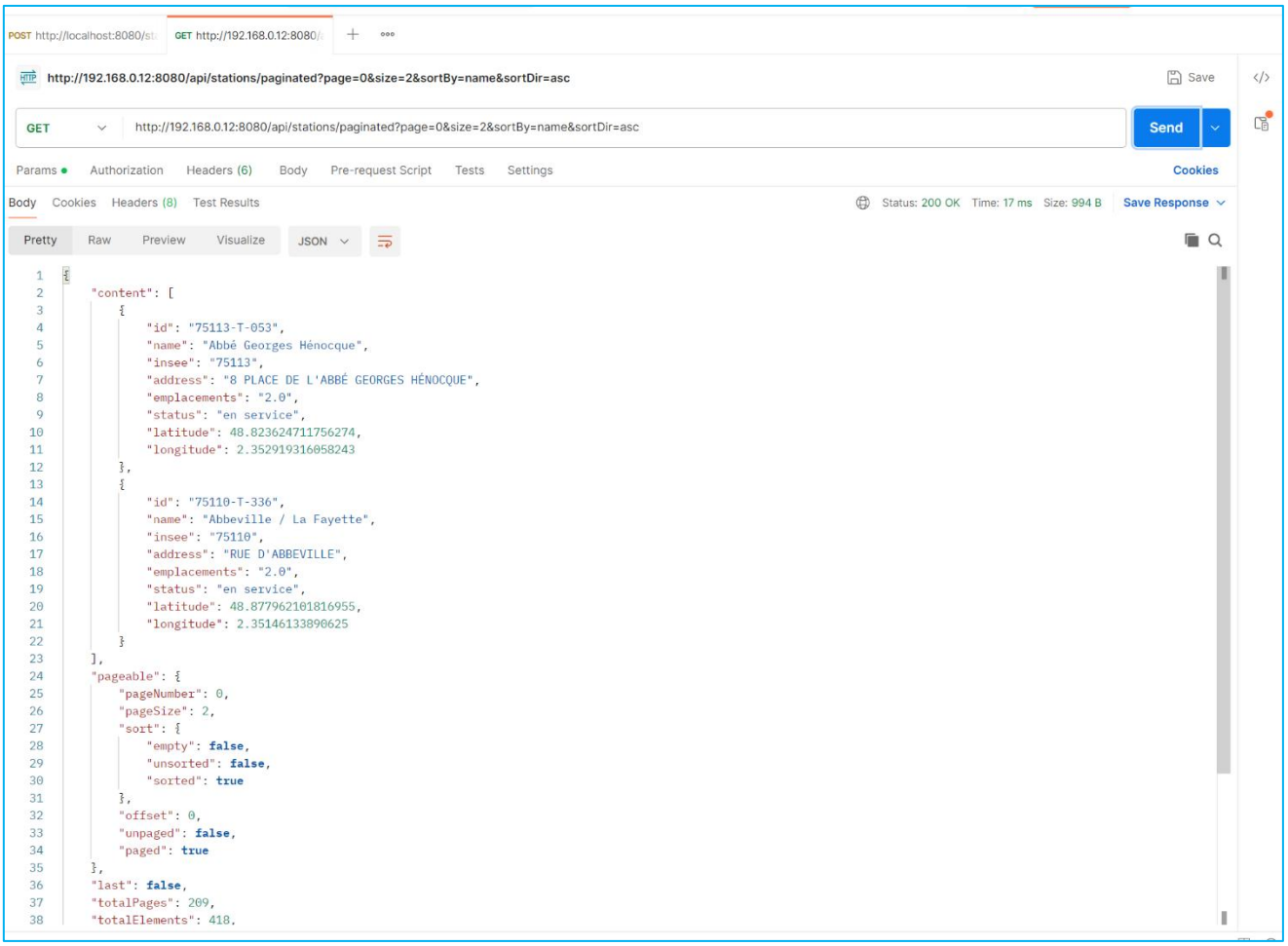
```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

Installez la dernière version de PowerShell pour de nouvelles fonctionnalités et améliorations ! https://aka.ms/PSWindows

PS C:\Users\DAVS-LAB> cd C:\kafka\
PS C:\kafka> .\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic taxi-stations.raw
{"id":"75119-T-484","name":"Tanger","insee":"75119","address":"16T RUE DE TANGER","emplacements":"6.0","status":"en service","latitude":48.886188600688236,"longitude":2.3689646538
300746}
{"id":"75110-T-337","name":"Alsace","insee":"75110","address":"21 RUE D'ALSACE","emplacements":"5.0","status":"en service","latitude":48.877392499606617,"longitude":2.3578642420702
05}
{"id":"75108-T-298","name":"Champs Elys├®es / Bassano","insee":"75108","address":"103 AVENUE DES CHAMPS ELYSEES","emplacements":"8.0","status":"en service","latitude":48.871816981
38564,"longitude":2.300063551913162}
{"id":"75109-T-307","name":"Haussmann / Taitbout","insee":"75109","address":"14 BOULEVARD HAUSSMANN","emplacements":"4.0","status":"en service","latitude":48.872540342976826,"long
itude":2.3372200997150885}
{"id":"75116-T-429","name":"Poincar├® / Lauriston","insee":"75116","address":"40 AVENUE RAYMOND POINCARE","emplacements":"5.0","status":"en service","latitude":48.86602809804327,"
longitude":2.2864425340334}
{"id":"75108-T-280","name":"Roosevelt / Th├®├ótre du Rond Point","insee":"75108","address":"43 AVENUE FRANKLIN D. ROOSEVELT","emplacements":"6.0","status":"en service","latitude":
48.86822841945435,"longitude":2.309953980037344}
{"id":"75111-T-348","name":"Richard Lenoir / Amelot","insee":"75111","address":"1 BOULEVARD RICHARD LENOIR","emplacements":"5.0","status":"en service","latitude":48.85430347141947
,"longitude":2.3692108009535335}
{"id":"75112-T-901","name":"Gare de Lyon SNCF","insee":"75112","address":"8 PLACE LOUIS ARMAND","emplacements":"103.0","status":"en service","latitude":48.8451106231118,"longitude
":2.3734361058278891}
{"id":"75117-T-457","name":"Rome / Legendre","insee":"75117","address":"135 RUE DE ROME","emplacements":"2.0","status":"en service","latitude":48.885707402091235,"longitude":2.316
6163174391654}
{"id":"75117-T-458","name":"Mac-Mahon / Troyon","insee":"75117","address":"21 AVENUE MAC-MAHON","emplacements":"6.0","status":"en service","latitude":48.8769281012419,"longitude":
2.2942424143831746}
{"id":"75117-T-460","name":"Pershing / Koenig","insee":"75117","address":"22 BOULEVARD PERSHING","emplacements":"14.0","status":"en service","latitude":48.88006722507015,"longitud
e":2.28262950142426}
{"id":"75119-T-482","name":"Petit/Meaux","insee":"75119","address":"121 RUE DE MEAUX","emplacements":"8.0","status":"en service","latitude":48.884585912691115,"longitude":2.378834
7261473257}
{"id":"75115-T-387","name":"Quai de Grenelle / Novotel","insee":"75115","address":"61 QUAI DE GRENELLE","emplacements":"8.0","status":"en service","latitude":48.84992294166981,"lo
ngitude":2.2829001567496987}
{"id":"75101-T-211","name":"Marengo","insee":"75101","address":"1 RUE DE MARENGO","emplacements":"3.0","status":"en service","latitude":48.86184837657944,"longitude":2.33912892708
6149}
{"id":"75102-T-214","name":"Capucines/Louis le Grand","insee":"75102","address":"1 BOULEVARD DES CAPUCINES","emplacements":"4.0","status":"en service","latitude":48.87076285656672
,"longitude":2.3339630975787142}
{"id":"75111-T-043","name":"Saint Ambroise","insee":"75111","address":"57 BOULEVARD VOLTAIRE","emplacements":"5.0","status":"en service","latitude":48.862002676608256,"longitude":
2.373391871065638}
{"id":"75117-T-082","name":"Etoile - Wagram","insee":"75117","address":"67 Boulevard Pereire","emplacements":"10.0","status":"en service","latitude":48.887175678944479,"longitude":
2.30538627437072 9}
{"id":"75115-T-070","name":"Place Balard","insee":"75115","address":"3 PLACE BALARD","emplacements":"3.0","status":"en service","latitude":48.83691440100358,"longitude":2.27906697
8305342}
{"id":"75111-T-038","name":"Nation","insee":"75111","address":"3 AVENUE DU TR├öNE","emplacements":"14.0","status":"en service","latitude":48.84860852998622,"longitude":2.397827453
3514643}
{"id":"75117-T-090","name":"Porte de Saint Ouen","insee":"75117","address":"3 AVENUE DE LA PORTE DE SAINT OUEN","emplacements":"7.0","status":"en service","latitude":48.8982866684
1568,"longitude":2.3287853028781047}
{"id":"75109-T-320","name":"Condorcet","insee":"75109","address":"69 RUE CONDORCET","emplacements":"5.0","status":"en service","latitude":48.88032485422117,"longitude":2.340669811
693258}
```

# Annexe 2 : OpenRefine

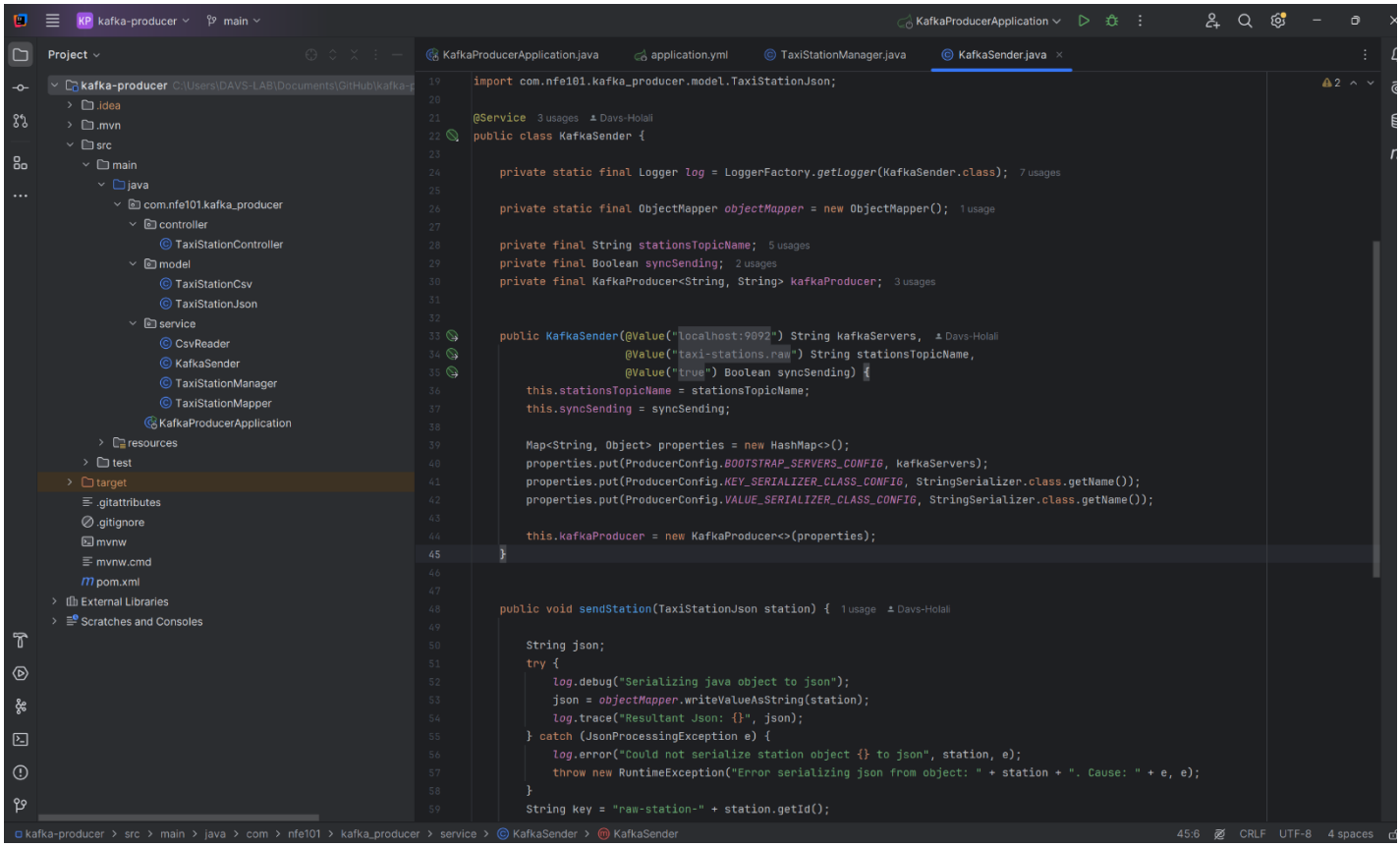


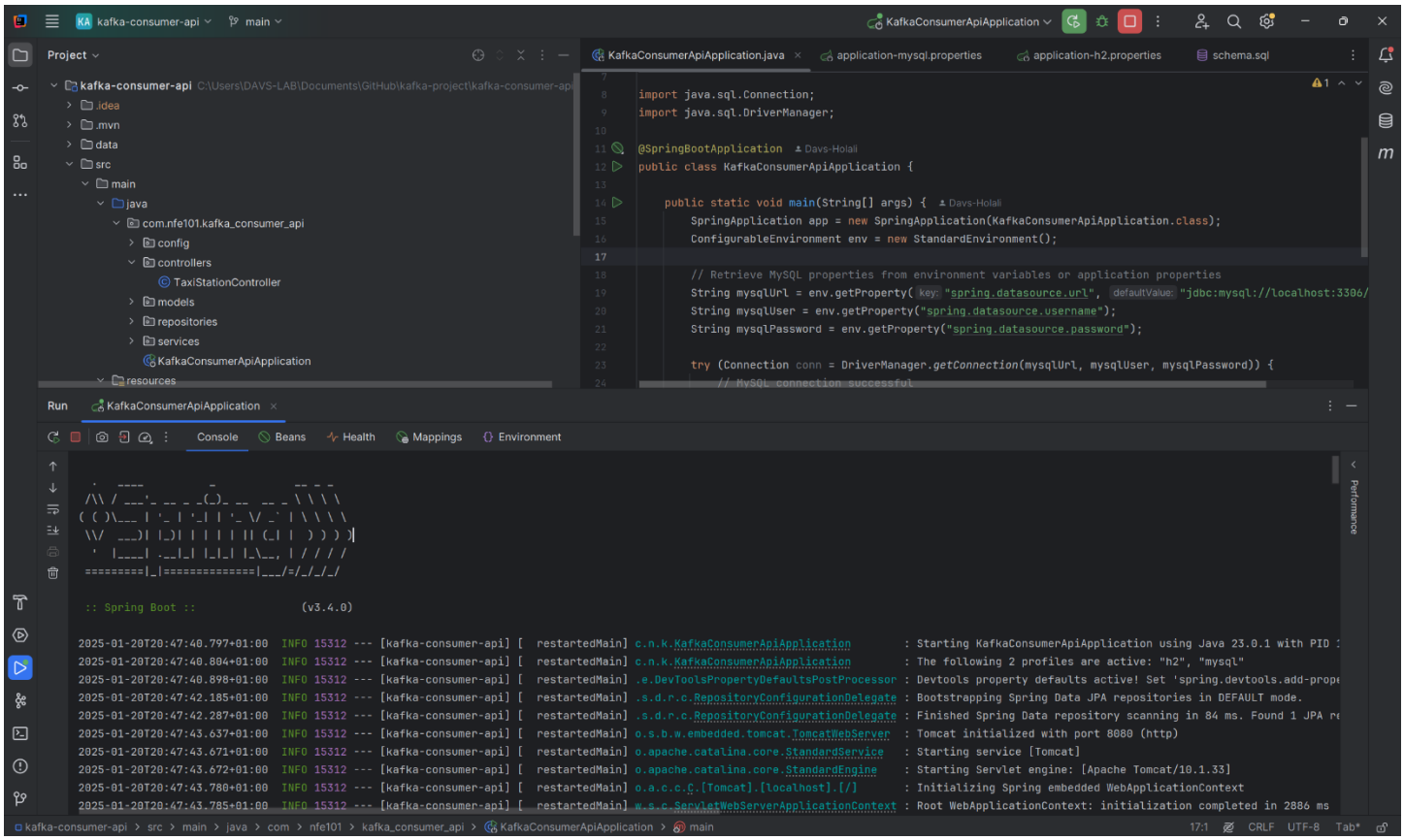
## Annexe 3 : Requête API via Postman



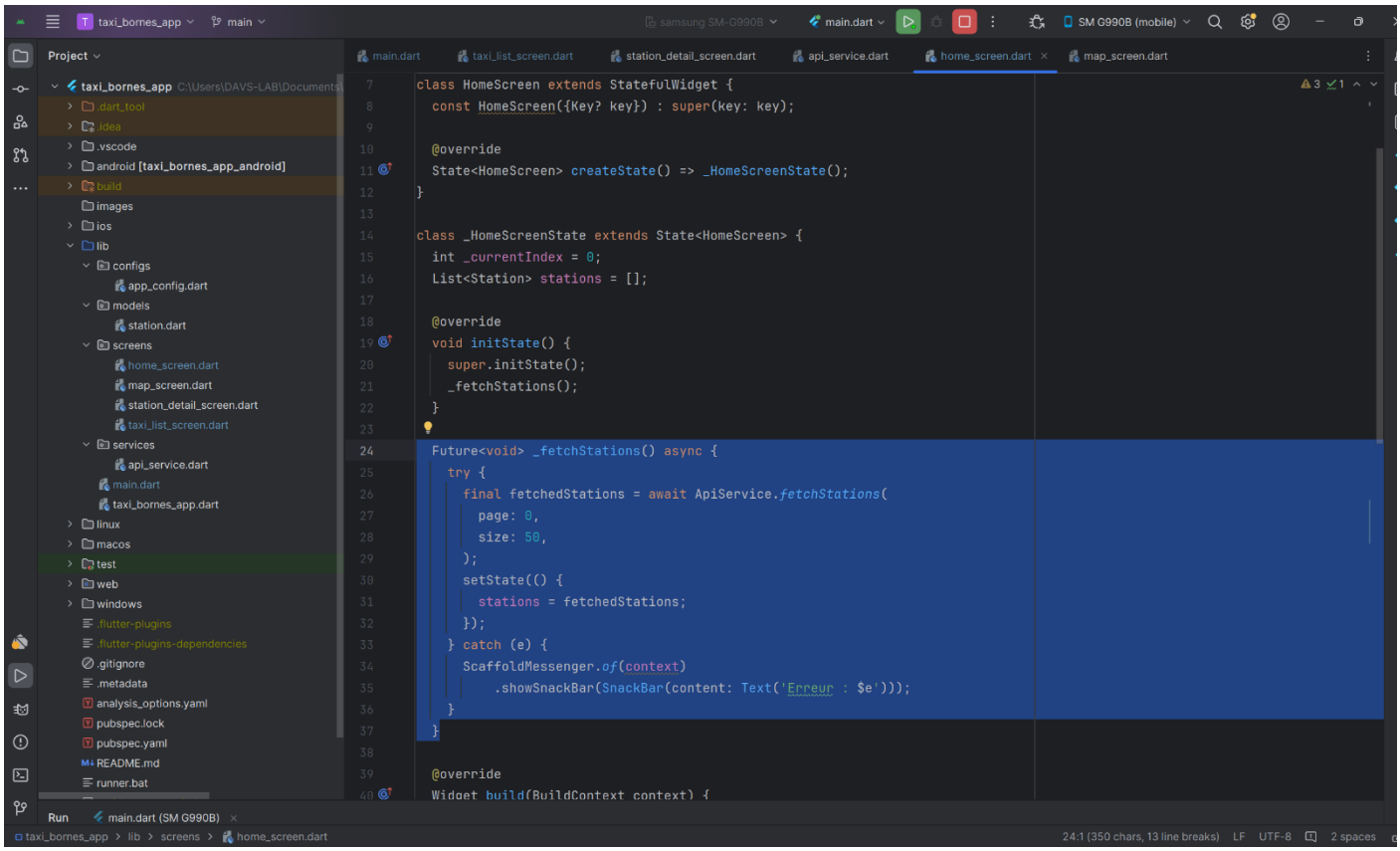

## Annexe 4 : Service Kafka Producer



## Annexe 5 : Service Kafka consumer and API




## Annexe 6 : Application flutter (extrait de codes)



## Annexe 7: Interface UI de l'application mobile flutter