**UI-клієнти для Apache Kafka**

*Підписуйтеся на [Telegram-канал «DOU #tech»](http://telegram.me/dou_tech" \t "_blank), щоб не пропустити нові технічні статті.*

Усім привіт. Я Сергій Моренець, розробник, викладач, спікер та технічний письменник. Хочу поділитися своїм досвідом роботи з такою цікавою темою як візуалізація та керування даними в Apache Kafka. Ми розглядаємо Kafka у [деяких наших тренінгах](https://it-simulator.com/" \l "/portfolio/event/19/apache-kafka-dlya-java-razrabotchikov). У цій статті я хотів би торкнутися теми UI клієнтів для цієї технології. Apache Kafka — це не тiльки система для обміну повідомленнями, а й сховище для них (як, наприклад, база даних). Нам (і розробникам, і DevOps) потрібно якимось чином контролювати дані, які там знаходяться, а також моніторити внутрішній стан кластера. Зазвичай розробники СУБД надають повноцінне GUI для своїх баз даних, наприклад MySQL Workbench, pgAdmin або MongoDB Compass. У самій Apache Kafka як open-source проекті для цього є лише засоби командного рядка. Тому давайте розглянемо додаткові засоби для візуалізації, які пропонує індустрія:

1. Confluent Control Center
2. Kowl
3. Akhq
4. Kafdrop
5. Kafka Topics UI

Одна з важливих вимог до таких систем — можливість інтеграції з різними сервісами Kafka:

* Schema Registry
* Kafka Connect
* ksqlDB
* Kafka Streams

Ще одна критично важлива функціональність — підтримка та розпізнавання (бажано, автоматичне) найбільш популярних форматів повідомлень для Kafka:

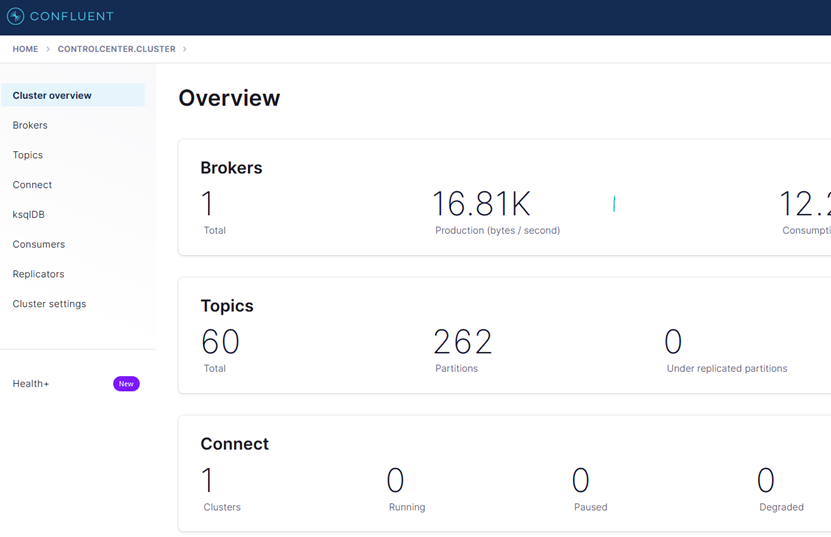
* JSON
* Avro
* MsgPack
* Protobuf
* XML

**Confluent Control Center**

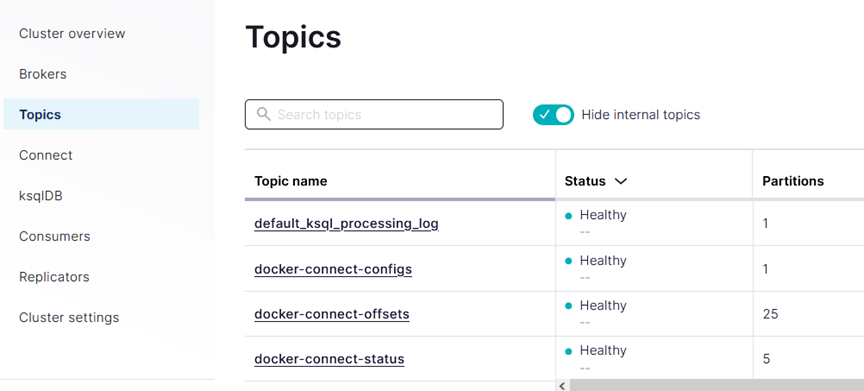
І найпотужніша система з усіх розглянутих [— Control Center](https://docs.confluent.io/platform/current/control-center/index.html), який розробляється компанією Confluent і в свою чергу є частиною [Confluent Platform](https://docs.confluent.io/platform/current/platform.html" \t "_blank). Його головний плюс — він розробляється тією самою організацією, як і сама Kafka, а мінус — те, що поширюється під комерційною ліцензією.

**Confluent Control Center** — вебдодаток, який може працювати всередині контейнера Docker. Він дозволяє інтегруватися з усіма вищезгаданими Kafka сервісами і може працювати у двох режимах. У нормальному режимі ви можете моніторити, зберігати та аналізувати метрики роботи Kafka. Для цього необхідно на всіх клієнтських програмах (споживачі та відправники даних) встановити так звані [metrics interceptors](https://docs.confluent.io/platform/current/control-center/installation/clients.html" \t "_blank). Вони будуть моніторити роботу клієнтів та надсилати метрики до Control Center. У спрощеному (*Reduced Infrastructure*) режимі ця функціональність відключена для зниження навантаження на сервери та зменшення обсягу використовуваного простору.

На головній сторінці є доступ до всіх сервісів екосистеми Kafka:



Включаючи і роботу з topics — перегляд, створення та видалення:

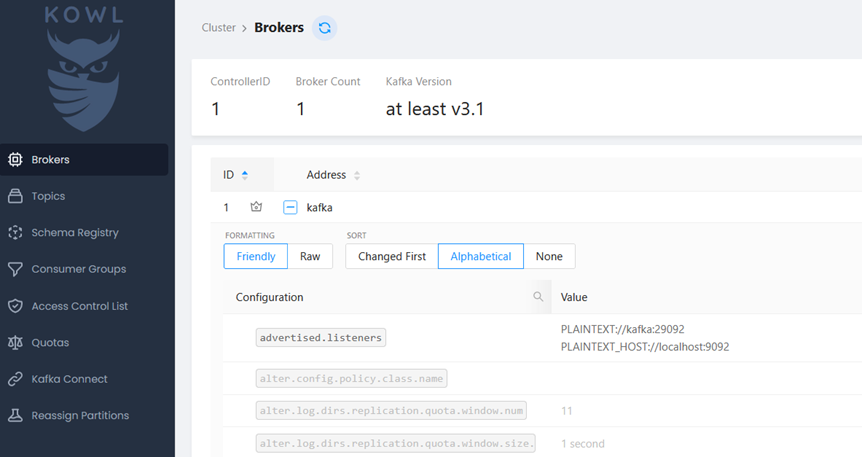


По суті, Control Center — дуже потужна система керування Kafka кластером, що вимагає складної конфігурації і вимоглива до ресурсів. Якщо вам потрібен більш простий «переглядач» даних, то community пропонує інші варіанти.

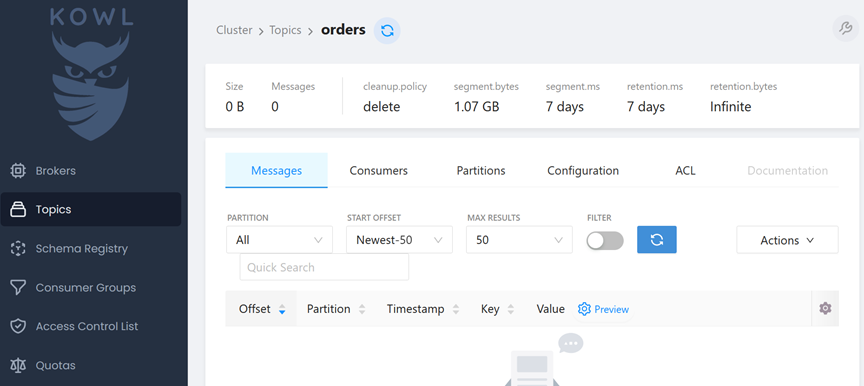
**Kowl**

[Kowl](https://github.com/redpanda-data/kowl) (раніше називався Kafka Owl) — безкоштовний вебдодаток, що розробляється з 2019 року на Go (серверна частина) та React (фронт-енд), який є частиною складнішої платформи [Redpanda](https://redpanda.com/" \t "_blank). Спочатку його автором була компанія CloudHut, яка у 2022 році була поглинена Redpanda. Kowl дозволяє інтегруватися з Schema Registry та Kafka Connect та переглядати найважливішу інформацію про Kafka кластер:

* Topics
* Messages
* Consumer Groups



На жаль, це лише браузер даних, і ви не можете створювати нові topics або відправляти повідомлення до них. Але, з іншого боку, ви можете використовувати додаткові фільтри та спеціальний формат запитів для пошуку повідомлень.

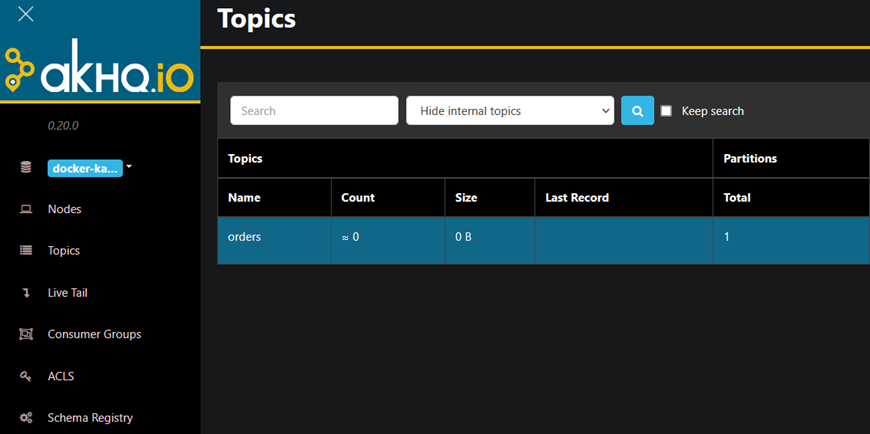


Великим плюсом цієї технології є розпізнавання найпопулярніших бінарних форматів, що використовуються в повідомленнях, включаючи Avro, MsgPack та Protobuf. Якщо ж вам додатково потрібна автентифікація та авторизація (наприклад, через OAuth), це доступно тільки в платній версії **Kowl Business**.

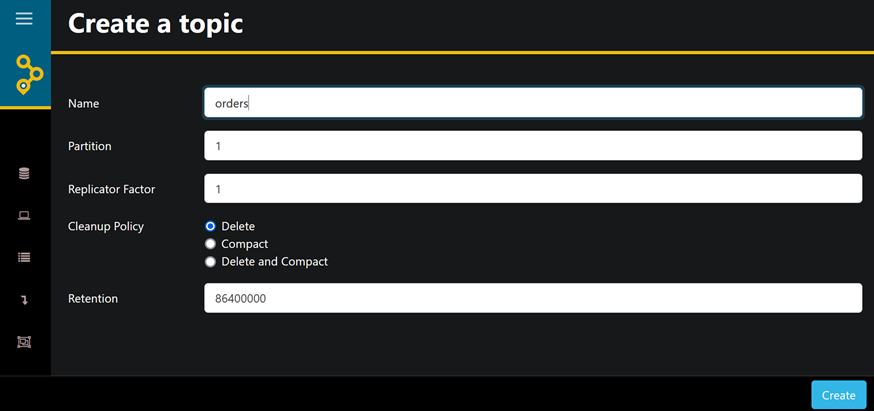
**Akhq**

[Akhq](https://github.com/tchiotludo/akhq) (раніше він називався KafkaHQ) — ще один безкоштовний вебклієнт, схожий на Kowl, але вже написаний на Micronaut (серверна частина) та React (фронт-енд). Досить молодий проєкт (розробка стартувала у 2018), який є частиною великої платформи з побудови складних workflow [Kestra](https://github.com/kestra-io/kestra" \t "_blank). Якщо підключити Schema Registry, то Akhq може десеріалізувати повідомлення у форматах Avro і Protobuf. На жаль, ksqlDB не підтримується.

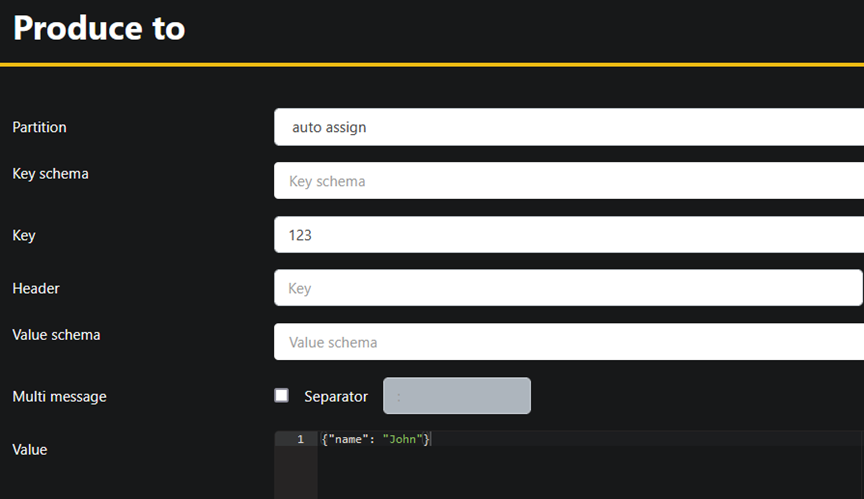
Ось як виглядає його головна сторінка:



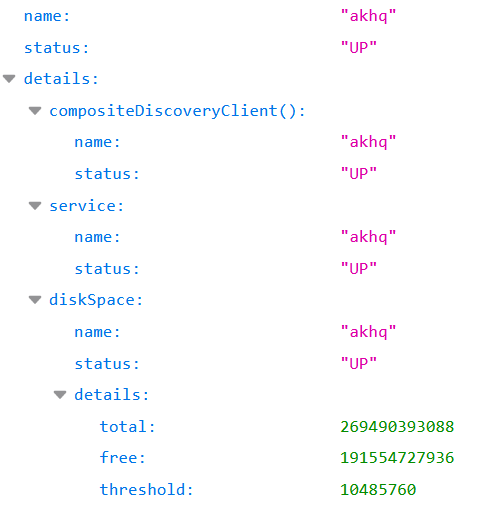
Але на відміну від Kowl, тут ви можете створювати нові topics:



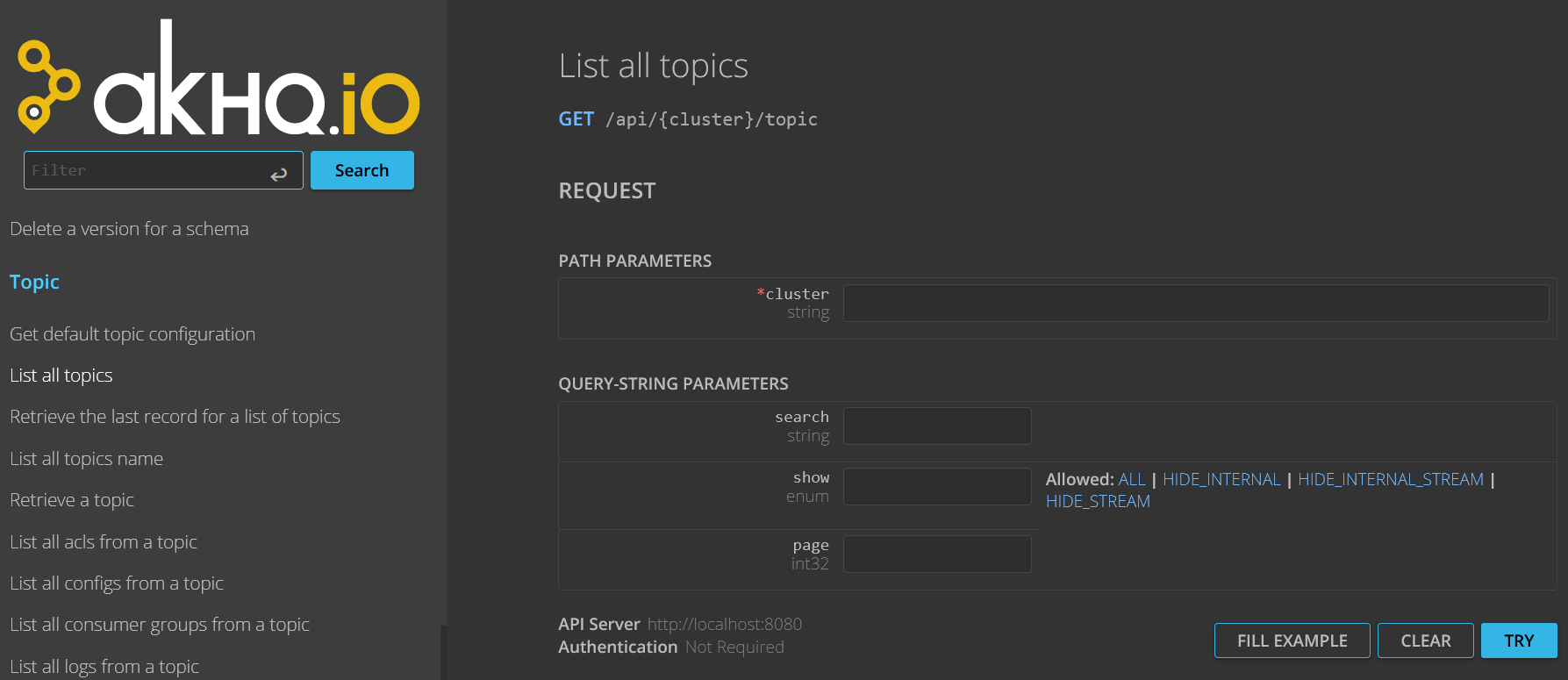
І надсилати туди повідомлення:



Плюсом є те, що Akhq підтримує автентифікацію (JWT, LDAP, AWS MSK) та базові можливості моніторингу (метрики, логування, стан життєдіяльності програми):



Але це ще не все. Розробники Akhq створили публічний API (нехай і експериментальний), і тепер можна отримати всю інформацію про кластер Kakfa через цей REST API:

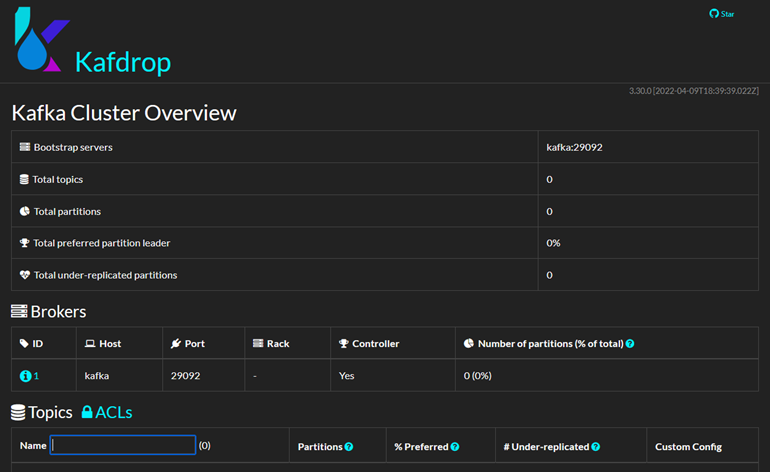


Akhq став єдиним із цих 5 аналізованих проектів, де довелося повозитися з конфігурацією, причому не його, а Kafka. У ній є дві важливі властивості — listeners та «**advertised listeners**». Друга властивість дуже важлива для клієнтів, оскільки вони використовуватимуть її для підключення. Так ось, якщо вказати значення властивості як «*PLAINTEXT://broker:29092, PLAINTEXT\_HOST://localhost:9092*», то Akhq використовує останнє значення зі списку як адресу брокера. Звичайно, це не буде працювати, якщо Akhq і Kafka знаходяться на різних серверах (наприклад, ми їх стартуємо через Docker Compose). Тому довелося змінити для Akhq значення просто на PLAINTEXT://broker:29092.

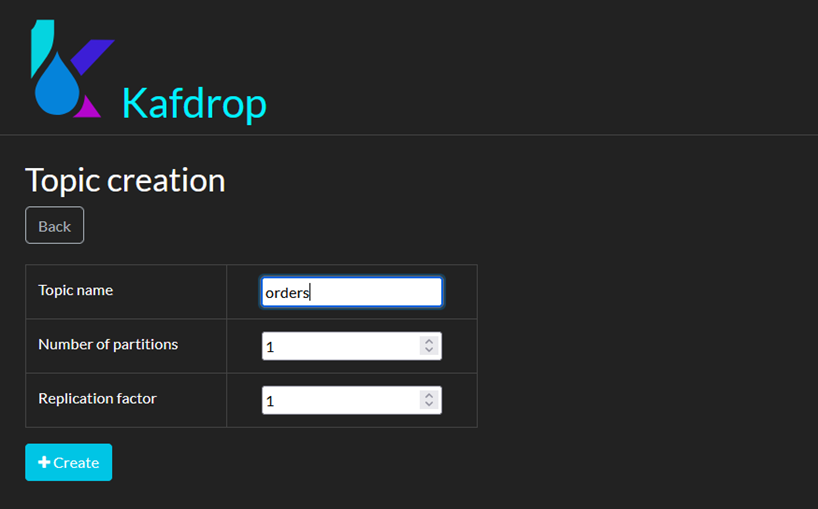
**Kafdrop**

[Kafdrop](https://github.com/obsidiandynamics/kafdrop) — безкоштовний UI-клієнт від компанії [Obsidian Dynamics](http://obsidiandynamics.com/" \t "_blank). Цікаво, що ця компанія не тільки пише комерційні проєкти для своїх клієнтів, але й активно вкладається в open-sourcing. Kafdrop розробляється з 2019 року і написаний на Spring Boot (серверна частина) та Freemarker, jQuery та Boostrap (фронт-енд).

Цей проект пропонує найбільші можливості роботи з Kafka без інтеграції з ksqlDB і Kafka Connect/Streams. Але ви можете переглянути поточну інформацію по Kafka кластеру:



І легко створити topic:



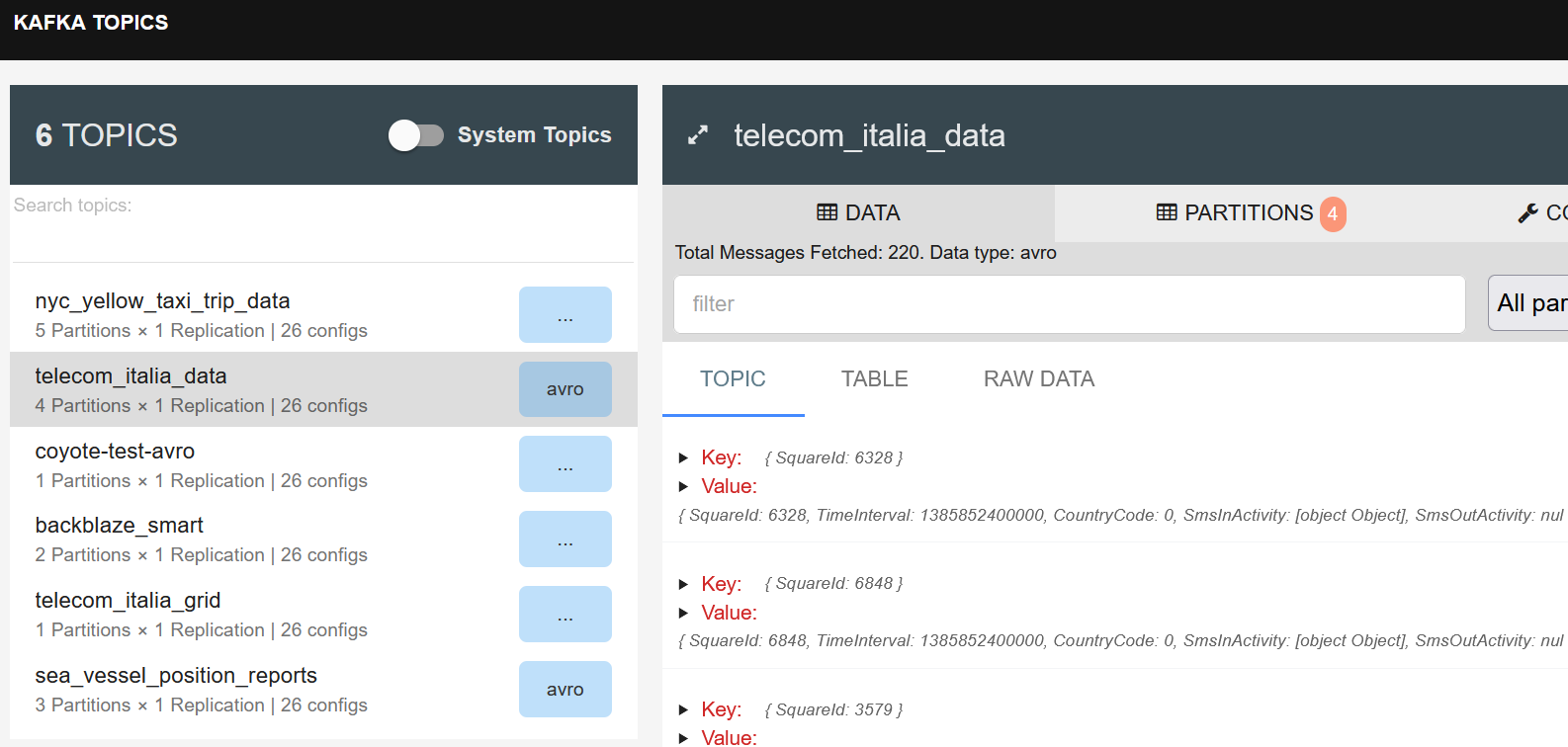
Kafdrop також підтримує інтеграцію з Schema Registry і дозволяє розпізнавати повідомлення у форматі Protobuf.

**Kafka Topics UI**

[Kafka Topics UI](https://github.com/lensesio/kafka-topics-ui) — найстарiший з вебклієнтів для Kafka, розроблений лондонським стартапом [Lenses](https://lenses.io/" \t "_blank). Ця компанія відома своєю streaming platform на основі Kafka, а також безліччю безкоштовних UI клієнтів для Kafka сервісів. Kafka Topics UI написав на AngularJS і розроблявся з 2016 по 2019 рік. На жаль, зараз розробка зупинена, можливо, через те, що AngularJS втратив свою популярність після появи Angular 2.x.

Особливість цієї технології в тому, що тут немає серверної частини, а його клієнтська частина не містить Kafka клієнта, як усі попередні проєкти. Натомість Kafka Topics UI використовує ще один цікавий проєкт — [Kafka REST Proxy](https://github.com/confluentinc/kafka-rest" \t "_blank) вiд Confluent. Цей сервіс дозволяє вимагати або створювати дані в Kafka кластері через простий і зрозумілий REST API. Іноді це єдиний варіант підключення до Kafka, якщо, наприклад, для вашої мови програмування або платформи немає Kafka клієнта.

Тому якщо ви хочете використовувати цей додаток, вам доведеться встановити та запустити Kafka REST Proxy. Сам Kafka Topics UI дуже простий у використанні і пропонує лише перегляд topics/повідомлень:



Kafka Topics UI може розпізнавати формати JSON/Avro, кодування Base64, [але не підтримує Protobuf](https://github.com/lensesio/kafka-topics-ui/issues/137).

З мінусів цього проекту — так як він перестав підтримуватися, то його не можна використовувати з останніми версіями Kafka REST Proxy, тому що там поміняли API.

**Висновки**

Яку з цих технологій варто вибрати? Звичайно, у цій статті описані не всі UI tools, та про них усіх і неможливо було розповісти одночасно. Наприклад, є така популярна утиліта [Offset Explorer](https://www.kafkatool.com/" \t "_blank). Але для неї немає офіційного Docker-образу, а в цьому огляді вибрано ті проєкти, для яких є свій Docker-образ, що спрощує їх використання та тестування. Поєднує їх те, що вони не є самостійними проєктами, а сплановані як вебклієнти для платформ обробки даних на основі Kafka (крім Kafdrop). Всі вони тією чи іншою мірою дозволяють переглядати дані в Kafka кластері.

Найбільш простим і невибагливим до ресурсів є **Kafka Topics UI**, але він з 2020 року не підтримується і вимагає Kafka REST Proxy, причому певних версій. Confluent Control Center найбільш функціональний, але поширюється тільки для комерційного використання. Його головний плюс — інтеграція з іншими Kafka сервісами та можливість використання у Confluent Platform/Confluent Cloud.

Kowl не дозволяє модифікувати дані в Kafka кластері, так що він підходить тільки якщо вам потрібен read-only доступ. Ну і у виборі між Akhq і Kafdrop я вибрав Akhq, виходячи з тієї функціональності, що він пропонує.