Składowanie danych w systemach Big Data

DOKUMENTACJA PROJEKTU

Zespół: *Problem, Plan, Problems*
w składzie:

Przemysław Chojecki, Paweł Morgen, Paulina Przybyłek

Opis źródeł danych wejściowych

W tym rozdziale opisane są źródła danych, które były wykorzystane w projekcie. Dostęp do wszystkich wykorzystanych źródeł jest nieodpłatny.

Free News API

To API jest udostępnione przez newscatcherapi (https://newscatcherapi.com) jako darmowe do wykorzystania niekomercyjnego. Dane udostępniane są z myślą o niezależnych deweloperach, hakerach-amatorach, studentach oraz data scientist'ach.

Istnieją dwa rodzaje użytkowników - podstawowy (Basic) oraz uczestnik (Contributor). Dostęp do obu jest darmowy, jednak o status uczestnik trzeba się dodatkowo starać. Niestety, autorom niniejszej pracy nie udało się zdobyć statusu uczestnika mimo wypełniania formularzy oraz wysyłania mili. Dlatego rozwiązanie powstało w oparciu o użytkownika podstawowego. Z tej perspektywy napisana będzie reszta niniejszego raportu.

API udostępnia użytkownikom dostęp do informacji o artykułach nie starszych niż siedem dni. Informacje, jakie udostępnia to m.in. tytułu, autor, pierwsze pięćset znaków treści artykułu (nazywane dalej "podsumowaniem artykułu"), informacji czy artykuł zawiera opinie autora, nazwę konta wydawnictwa w serwisie Twitter.

W celu uzyskania dostępu do wyżej wymienionych danych należało formułować zapytania REST z odpowiednim tekstem, którego użytkownik szuka w artykule (dalej nazywany kwerendą, ang. query).

Niestety, to API nie jest idealne, co nie dziwi biorąc pod uwagę jego darmowość. Jedną kwestią jest utrudniony dostęp do API ze względu na problem w skomunikowaniu się z administratorami API w celu uzyskania statusu uczestnika. Drugą kwestią jest jakość uzyskiwanych danych. W czasie pracy nad projektem odkryto, że często artykułom brakuje informacji o nazwie konta wydawnictwa w serwisie Twitter, a zdarza się, że jest ona podana błędna. Poza tym, czasem treść artykułu (podsumowanie) nie zawiera odpowiedniej treści artykułu, co sugeruje, że program przeszukujący internet i zbierający informacje dla tego API nie działa idealnie.

Jednakże napotkane problemy nie uniemożliwiły korzystania z tego źródła danych.

Twitter API

To API jest udostępniane przez właściciela popularnego serwisu typu medium społecznościowe, Twitter (twitter.com).

Właściciel platformy udostępnia API dla swojego serwisu, aby udostępnić deweloperom z całego świata łatwą integrację swoich produktów z serwisem Twitter.

Twitter API ma wiele wersji i poziomów dostępu - niektóre odpłatne. W tym projekcie wykorzystano darmową wersję dostępu o numerze v1.1. Z tej perspektywy napisana będzie reszta niniejszego raportu.

API serwisu Twitter wykorzystano na dwa różne sposoby. Jednym z nich było wykorzystanie dostępu do informacji o Tweetach nie starszych niż siedem dni, a drugim dostępu do informacji o użytkownikach serwisu.

Pobierano informacje o tweetach na temat danego artykułu. Pobierano również informacje o użytkownikach Twittera - wydawnictwach publikujących artykuły.

W celu uzyskania dostępu do wyżej wymienionych danych należało formułować zapytania REST z odpowiednim tekstem, którego użytkownik szuka, bądź nazwą poszukiwanego użytkownika serwisu.

W czasie pracy nad projektem nie natrafiono na żadne problemy dotyczące tego źródła danych poza problemem integracyjnym szerzej opisanym w sekcji NiFi.

Opis architektury rozwiązania

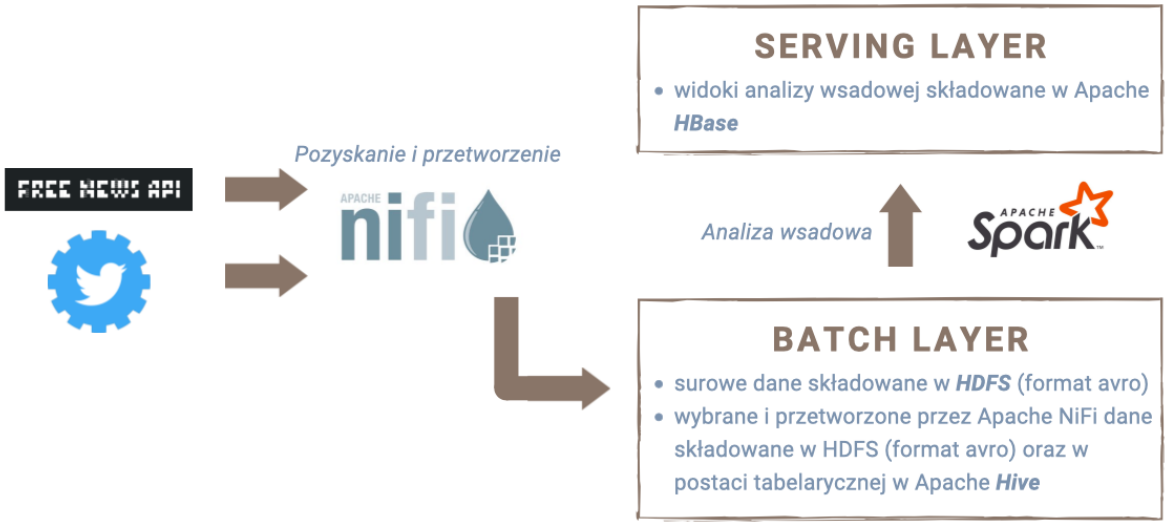
W tym rozdziale opisano wykonane rozwiązanie. Rozwiązanie było zaprojektowane z myślą o dużym wolumenie danych mimo, iż źródła danych w użytych wersjach nie pozwalały na zdobycie takowych.

Atchutekturę zaprojektowano z myślą o zapewnianiu najwyższej jakość rozwiązania oraz prędkość przetwarzania potencjalnie dużych wolumenów danych. Z tego powodu zastosowana architektura wzorowana była na architekturze lambda. Do pełni implementacji architektury lambda brakuje jedynie implementacji Speed Layer, o czym więcej informacji znajduje się w rozdziale Możliwe kierunki rozwoju.

Dostarczono zatem warstwy Batch Layer oraz Serving Layer.

Diagram architektury - jak to działa

Na poniższym obrazku zaprezentowany jest schemat rozwiązania. Widnieje na nim schematyczna reprezentacja zaimplementowanego rozwiązania.



Na połączeniu warstw ze źródłami danych znajduje się Apache NiFi, który dokonuje autoryzacji w obu ze źródeł danych. Zapisuje on surowe dane do HDFS w formacie AVRO. Surowe dane są także przetwarzane, a wyniki tego przetworzenia są zapisywane w dwóch miejscach - jako pliki na HDFS w formacie AVRO, ale również w formie tabel Hive.

Analiza na tabelach Hive wykonywana jest przez Apache Spark, a wynik tej analizy zapisywany jest w Apache HBase w warstwie Serving Layer w formie dwóch tabel.

Poniżej opisano szczegóły każdego z fragmentów architektury. Szczegółowe informacje na temat wykorzystanych źródeł danych znajdują się w rozdziale Opis źródeł danych wejściowych.

NiFi

Apache NiFi został użyty w celu połączenia źródeł danych z warstwami dostępowymi. Został on wybrany ze względu na swoją popularność w tego typu rozwiązaniach oraz ze względu na skalowalność i wysoką wydajność rozwiązań, które z niego korzystają.

Flow w Apache NiFi został zaprojektowany w taki sposób, aby być prosty w zrozumieniu oraz w edycji. Aby ewentualne zmiany architektury w przyszłości były możliwie szybkie do wdrożenia nawet przez zespół nie biorący udział w tworzeniu tego rozwiązania.

NiFi został skonfigurowany tak, aby regularnie, raz w tygodniu pobierać nowe dane ze źródeł danych, gdyż w użytych konfiguracjach udostępniają one swoje zasoby na siedem dni w przeszłość. To rozwiązanie zapewnia dostęp do wszystkich możliwych danych, używając minimalnych potrzebnych zasobów komputerowych.

Pobrane surowe dane są zamieniane na format AVRO i zapisywane na HDFS.

Pobrane surowe dane są przetwarzane, a na końcu tego procesu ustrukturyzowane do postaci trzech tabeli. Tabele te są zapisywane w dwóch niezależnych systemach - jako surowe pliki typu AVRO oraz jako tabele w Apache Hive.

HDFS

Do składowania danych użyto systemu HDFS przede wszystkim ze względu na jego skalowalność. Pozwala on bowiem na wygodne składowanie dużej ilości danych. Na dodatek system ten jest popularny i pozostałe używane w tym projekcie systemy (NiFi, Spark) natywnie się z nim komunikują.

Użyty system HDFS jest łatwo skalowalny wszerz oraz zapewnia replikację danych.

Hive

Dane w warstwie batchowej są składowane w 3 tabelkach Hive'owych: *articles, tweets* oraz *publishers*.

Tabela *articles*

Tabela *articles* przechowuje informacje o zebranych artykułach, jak tytuł, autorzy czy temat. Jej struktura jest zaprezentowana poniżej:

Kolumna	**Typ**	**Opis**
id	string	Identyfikator artykułu od Free News API
published_date	string	Data opublikowania artykułu
title	string	Tytuł artykułu
author	string	Nazwisko autora artykułu
topic	string	Temat artykułu (przypisany przez Free News API)
country	string	Kraj którego dotyczy artykuł
language	string	Język, w którym artykuł został opublikowany
is_opinion	boolean	Czy artykuł został oznaczony jako opinia przez wydawcę
querry	string	Zapytanie jakiego użyto do pobrania artykułu z FreeNewsAPI
summary	string	Pierwsze 500 znaków artykułu
my_timestamp	bigint	Timestamp

Poniżej przykładowe wiersze z tabeli *articles* otrzymane w wyniku zapytania w beelinie. Dla lepszej czytelności kolumna *summary* została pominięta.

```
0: jdbc:hive2://localhost:10000/> SELECT id,published_date,title,author,topic,country,language,is_opinion,querry,my_timestamp FROM articles LIMIT 5;
+----------------------------------+-----------------+----------------------------------------------------+----------------------+--------+----------+-----------+------------+---------+---------------+
|                id                | published_date  |                       title                        |        author        | topic  | country  | language  | is_opinion | querry  |  my_timestamp  |
+----------------------------------+-----------------+----------------------------------------------------+----------------------+--------+----------+-----------+------------+---------+---------------+
| cf8049c89481f1a9cb26bf867a719aaf | 2022-01-15      | Fire breaks out at NJ chemical plant 'The worst that I've ever seen' | NULL                 | news   | US       | en        | false      | the     | 1642861434487  |
| 76c60ec14b9966bc5f496980259aa0cb | 2022-01-15      | Sharecare Inc                                      | Kwhen Finance Editors | news   | US       | en        | false      | the     | 1642861434491  |
| c0081c32b23deb6cd13dafba8b446865 | 2022-01-20      | Missouri Alert System Warns Citizens To Be On The Lookout For The Joker  107.9 YYD | Scott Stevens        | news   | US       | en        | false      | the     | 1642861
| 69e7a8738b65546e0b8f7d35d4f3806b | 2022-01-21      | yoboo Launched New Nursing Pad and Entered Southeast Asian Market | NULL                 | news   | US       | en        | false      | the     | 1642861434485  |
| 1a63ca699e79a7003dbad985See89005 | 2022-01-17      | Scottish government in line for near-£700m payday after windfarm auction | Jillian Ambrose      | sport  | GB       | en        | false      | the     | 1642861434492  |
+----------------------------------+-----------------+----------------------------------------------------+----------------------+--------+----------+-----------+------------+---------+---------------+
```

Tabela *tweets*

Tabela *tweets* przechowuje dane o wpisach na Twitterze na temat artykułów z tabeli *articles*. Jej struktura jest zaprezentowana poniżej:

Kolumna	**Typ**	**Opis**
id	string	Identyfikator; połączenie article_id oraz tweet_id
article_id	string	Identyfikator artykułu wspomnianego w tweecie
tweet_id	bigint	Identyfikator dostarczony przez Twittera
tweet_text	string	Przeczyszczony tekst tweeta
my_timestamp	bigint	Timestamp

Poniżej przykładowe wiersze z tabeli *tweets* otrzymane w wyniku zapytania w beelinie.

Tabela *publishers*

Tabela publishers zawiera dane o wydawcach artykułów z tabeli *articles*. Część przechowywanych danych zmienia się w czasie (ilość followersów, ilość tweetów autorstwa wydawcy). Jej struktura jest zaprezentowana poniżej:

Kolumna	**Typ**	**Opis**
id	string	Identyfikator
article_id	string	Identyfikator artykułu opublikowanego przez tego wydawcę
twitter_id	bigint	Identyfikator konta wydawcy na Twitterze
twitter_account	string	Nazwa konta wydawcy na Twitterze
publisher_name	string	Nazwa wydawcy
location	string	Lokalizacja wydawcy
followers_count	int	Ilość followersów wydawcy
list_count	int	Ilość list twitterowych, jakie wydawca utworzył na swoim koncie
number_of_tweets	int	Ilość tweetów z konta wydawcy

Poniżej przykładowe wiersze z tabeli *publishers* otrzymane w wyniku zapytania w beelinie.

```
0: jdbc:hive2://localhost:10000/> SELECT * FROM publishers LIMIT 4;
```

publishers.id	publishers.article_id	publishers.twitter_id	publishers.twitter_account	publishers.publisher_name	publishers.location	publishers.followers_count	publishers.list_count	publishers.number_of_tweets
1ba05ecbdcc6dd62a1294d853f648088_7113672	1ba05ecbdcc6dd62a1294d853f648088	7113672	businesswire	Business Wire	San Francisco, CA	71904	2430	30538
d31c64b2e9f9bc5642c3489c40db9dfd_23059499	d31c64b2e9f9bc5642c3489c40db9dfd	23059499	SeekingAlpha	Seeking Alpha	New York, NY	195554	4866	857498
44ea89b26a20d0c92a5121d3bb1328f2_18639734	44ea89b26a20d0c92a5121d3bb1328f2	18639734	Nasdaq	Nasdaq	NULL	709686	7708	57096
43aef9df782175af5bccec5212786fd1_19380829	43aef9df782175af5bccec5212786fd1	19380829	Yahoo	Yahoo	Sunnyvale CA	1418298	8387	110708

```
4 rows selected (0.321 seconds)
```

Spark

Dane przetworzone przez NiFi zostają poddane przykładowej analizie za pomocą narzędzia Apache Spark. W toku pracy za pomocą biblioteki *spark-nlp* zostaje wyznaczony sentyment tytułu dla każdego artykułu. Rezultatem analizy są dwie tabele agregujące dane odpowiednio po wydawcach i tematach. Dla obu tabel zostają wyznaczone miarki: sumy artykułów, procentu artykułów z pozytywnym i negatywnym sentymentem oraz sumy tweetów na temat artykułów od danego wydawnictwa/na dany temat. Ponadto dla wydawnictw jest zawarta informacja o procencie artykułów będących opinią oraz o liczbie followersów.

HBase

Wyniki analizy w Sparku są zapisywane do plików tymczasowych, które następnie są podnoszone przez NiFia i ładowane do HBasowych tabel: *publishers* oraz *topics*.
Obie tabele mają po 3 kolumn families: *name*, *article_stats* oraz *twitter_stats*. W rodzinie *name* znajdują się odpowiednio nazwa tematu lub wydawcy; część miarek związana z twitterem (ilość tweetów oraz followersów w przypadku wydawców) związana jest z *twitter_stats*, a pozostałe miarki - z *article_stats*.

Przykładowe wiersze z tabelek HBasowych zaprezentowano na screenshotach poniżej. Wczytano je za pomocą Pythonowej biblioteki *happybase* do obiektów klasy *pd.DataFrame* (struktura tych tabel na to pozwalała).

```
In [66]:  print(topics.head())
```

```
  article_stats:articles_with_negative_sentiment_fraction      timestamp  \
0                                                 0.0      1642947921065
1                                               0.341      1642947921598
2                                               0.357      1642947921631
3                                                 0.2      1642947921270
4                                               0.274      1642947921199

  article_stats:articles_with_positive_sentiment_fraction article_stats:total_published_articles      name:topic  \
0                                                 1.0                                     1        beauty
1                                               0.339                                   528      business
2                                               0.393                                   168     economics
3                                               0.425                                    40        energy
4                                                0.46                                   113  entertainment

  twitter_stats:tweets_mentioning_articles_sum
0                                            0
1                                         6279
2                                          619
3                                          296
4                                          447
```

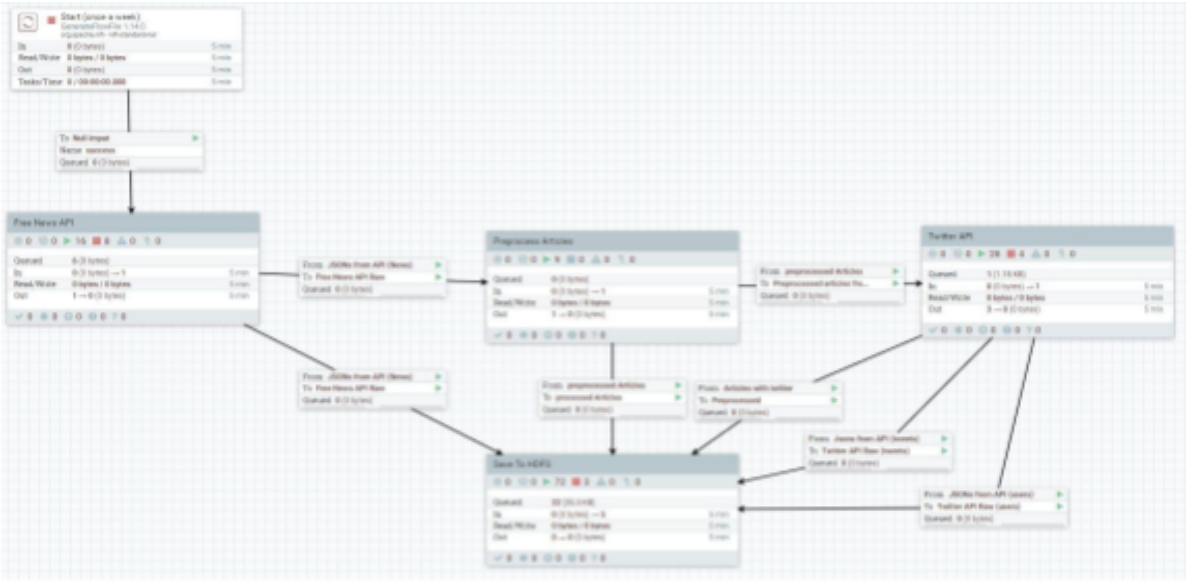
```
In [63]: 1 print(publishers.head())

  article_stats:articles_with_negative_sentiment_fraction      timestamp  \
0                                                0.0       1642947801727
1                                                0.0       1642947801727
2                                                0.8       1642947801439
3                                                1.0       1642947801727
4                                                0.0       1642947801010

  article_stats:articles_with_positive_sentiment_fraction article_stats:published_opinions_fraction  \
0                                                0.0                                           0.0
1                                                0.0                                           0.0
2                                                0.2                                           0.0
3                                                0.0                                           0.0
4                                                0.5                                           0.0

  article_stats:total_published_articles article_stats:total_published_opinions name:publisher_name  \
0                                      1                                       0      Harpreet Singh
1                                      1                                       0      Cormac McQuinn
2                                      5                                       0           Slashdot
3                                      1                                       0         Jon Fingas
4                                      2                                       0                TNW

  twitter_stats:publisher_followers twitter_stats:tweets_mentioning_articles_sum
0                              9035                                           10
1                              4670                                           52
2                            301818                                          488
3                              9236                                          200
4                           1671036                                           98
```

Testy funkcjonalne

W tym rozdziale opisane są przeprowadzone testy rozwiązania pozwalające stwierdzić poprawność jego działania.

Pobranie i przetworzenie danych, zapis do HDFS

Poniżej przedstawione jest przykładowe przejście całego procesu ładowania, przetwarzania oraz zapisu danych.



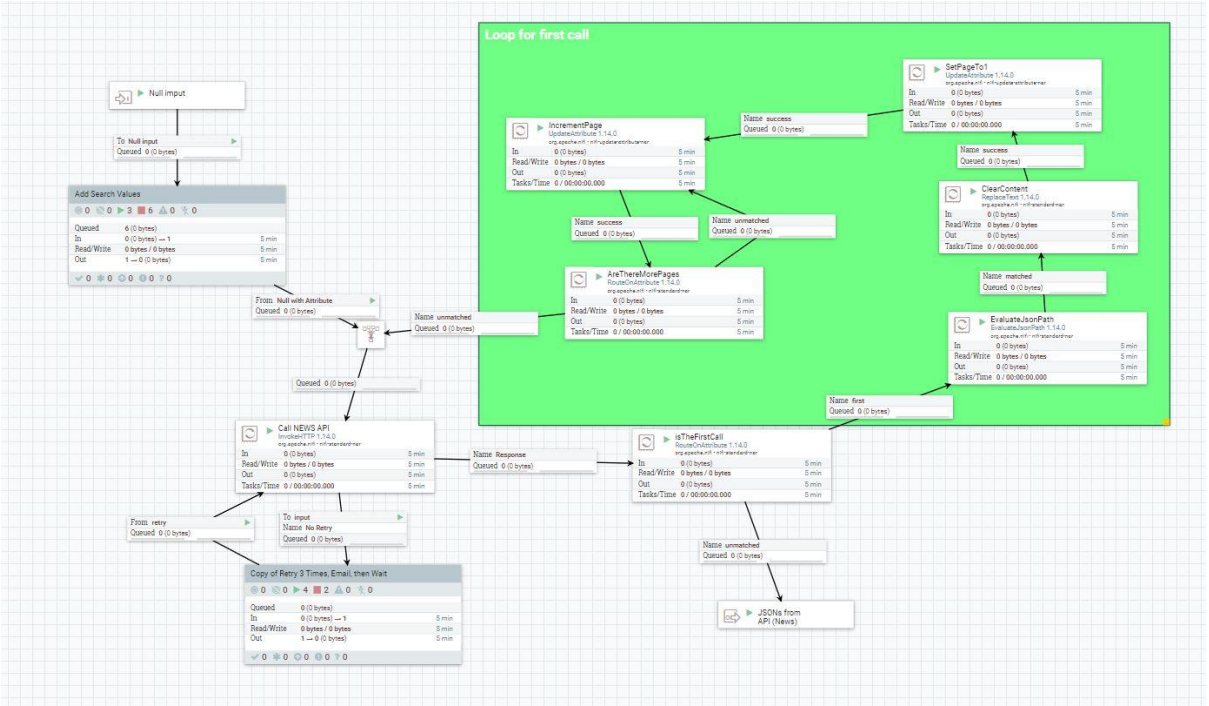
Przykładowe wywołanie flow w NiFi.

Wszystkie pobrane jak i przetworzone dane spływają do grupy zapisującej.

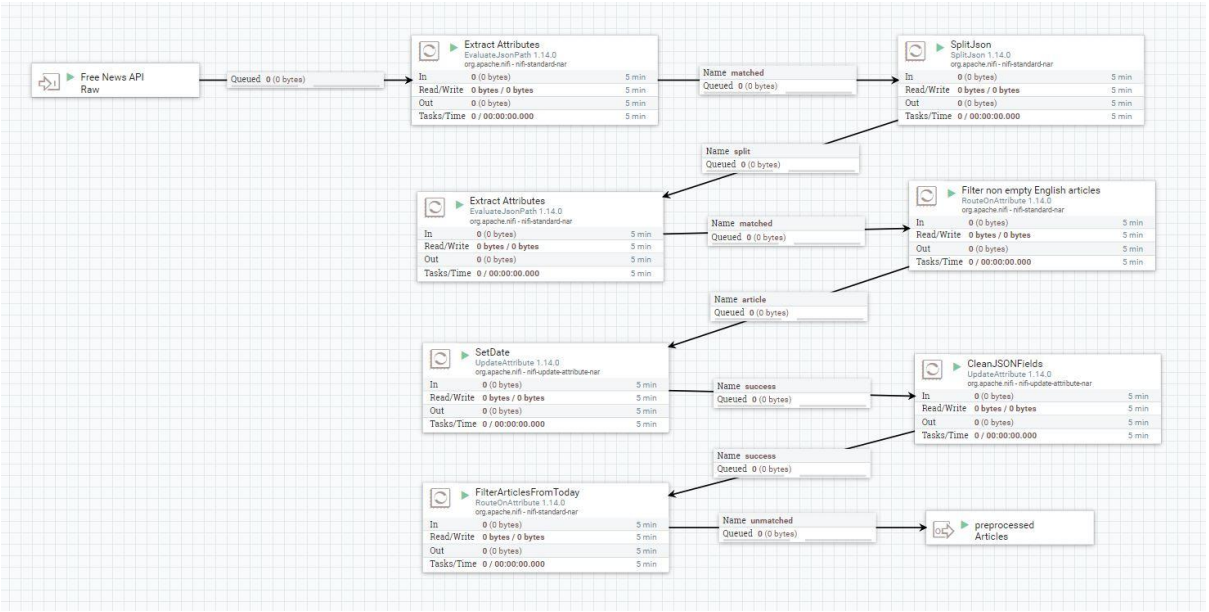
Widzimy po prawej, że jeden z artykułów jest w kolejce. Oznacza to, że w czasie pobierania danych z Twittera nastąpił błąd.

Widzimy na dole, że 22 flowfile oczekują w kolejce. Jest to spowodowane tym, że oczekują one, aż zbierze się ich większa ilość aby zostać wspólnie zapisane na HDFS.

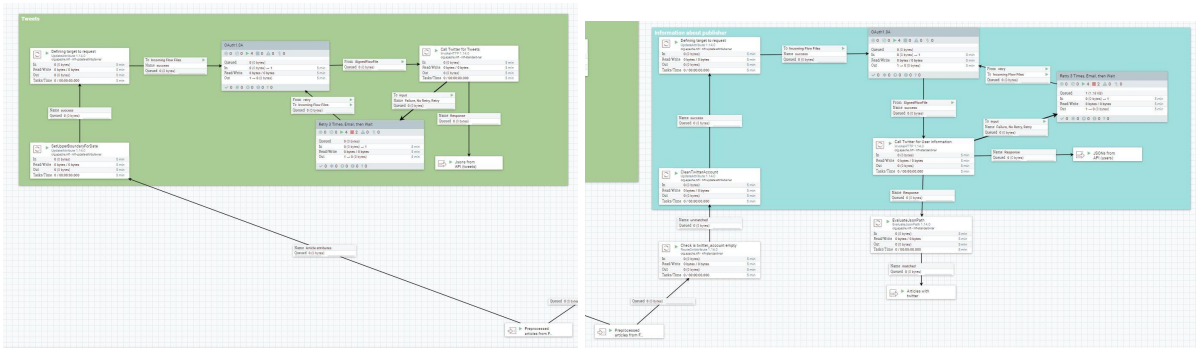
W pierwszej części pobierane są artykuły z Free News API. Pierwsze zapytanie dostarcza informacji o tym ile zapytań należy wykonać, a następnie na zielono zaznaczono pętlę, która dostarcza flowfiles w liczbie równej ilości zapytań do Free News API, które są potrzebne do pobrania wszystkich danych.



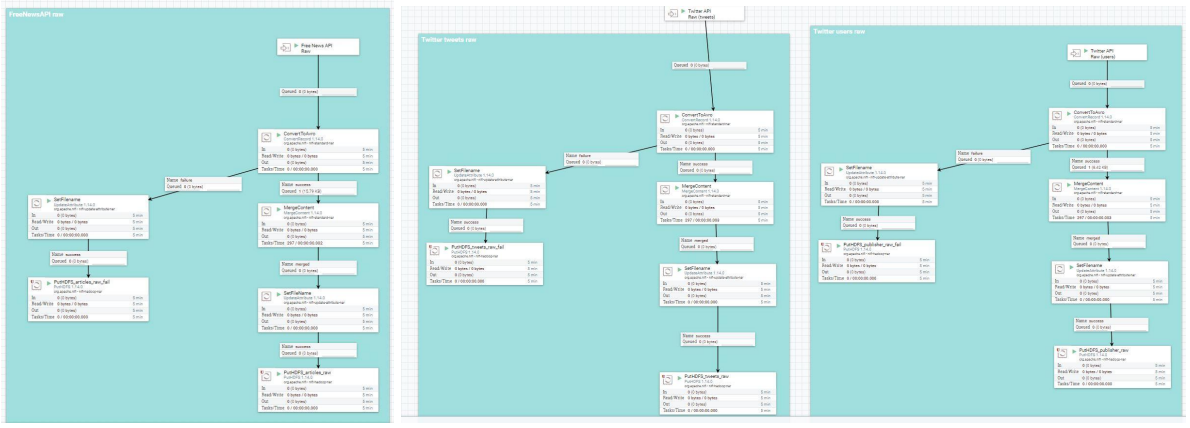
Następnie pobrane surowe dane w formie JSON idą w dwa miejsca: do zapisu surowych danych oraz do przetworzenia.
Przetworzenie danych polega na wyciągnięciu z JSONa informacji, które będą składowane.



Następnie przetworzone artykuły trafiają w dwa miejsca. Do zapisu do HDFS oraz do modułu wywołującego Twitter API.
W module Twitter API flowfiles z przetworzonymi artykułami trafiają w dwa miejsca - do modułu pobierającego treści tweetów (zielone tło) oraz do modułu zbierającego informacje o wydawnictwie (turkusowe/niebieskie tło).



Z obu tych modułów pobrane dane trafiają w dwa miejsca - do modułu zapisującego surowe dane oraz do modułu przetwarzającego.
Wszystkie trzy moduły zapisujące mają turkusowe/niebieskie tło:



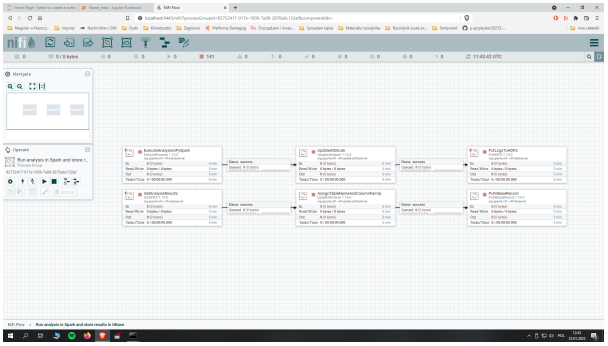
Moduły strukturyzujące dane do formy tabl, a potem zapisujące je do HDFS oraz Hive, są na żółtym tle.

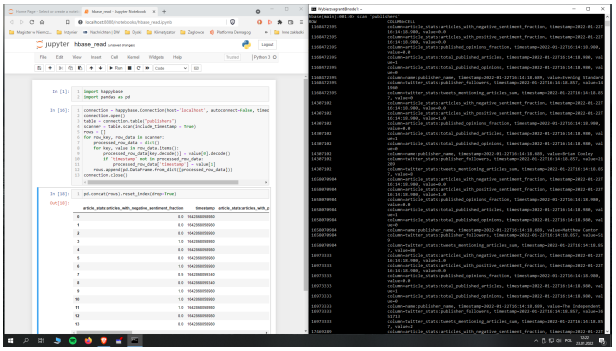
Każdy z tych modułów ma procesor “LOADING FROM BACK UP” umożliwiający załadowanie danych z tak zwanego back-up’u bez potrzeby przetwarzania surowych danych całym NiFi-flow.

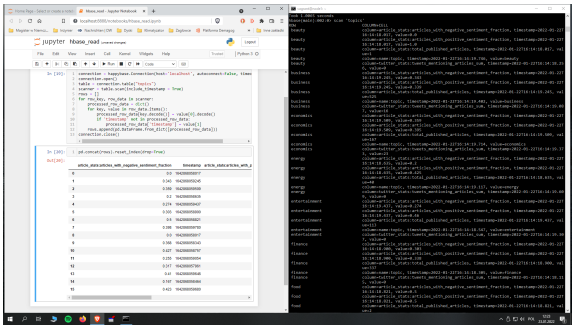
Wykonanie analizy i ładowanie do HBase’a

Stan “przed”

NiFi oczekuje na wydanie polecenia. Zwróćmy uwagę na timestampy w tabelkach HBase’owych - są one z dnia 22.01.2022.

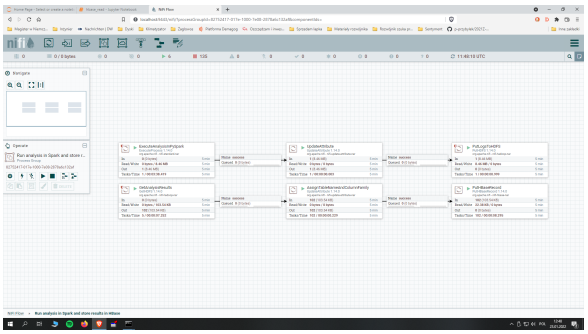


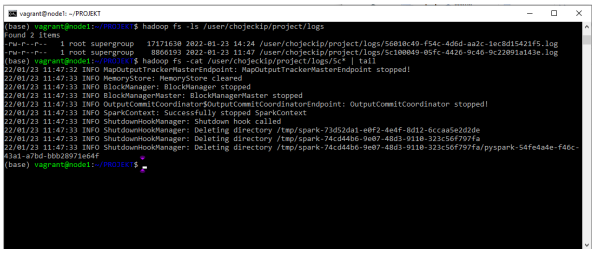


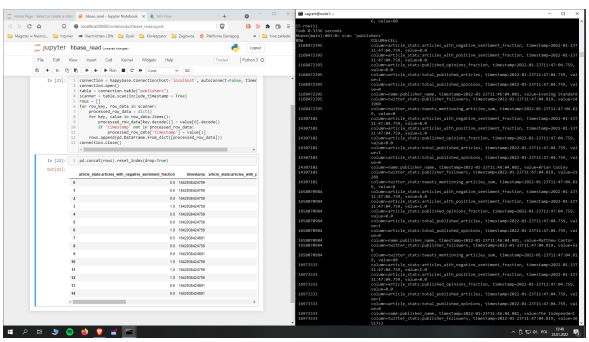


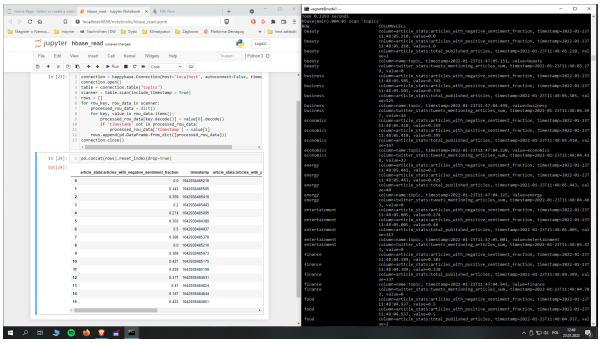
Stan “po”

Pojedyńczy Flowfile przeszedł przez część *analityczną*. Jego zawartość to napisy, które zostały skierowane na konsolę podczas wywołania Sparka. Zostały one zapisane do pliku .log, którego ostatnie 5 linijek podejrzano w konsoli.
W tabelkach HBase’owych zmienił się timestamp (wskazuje na 23.01.2022).. Pozostałe dane się zmieniły, ponieważ analiza była odpalana na tych samych danych wejściowych.









Możliwe kierunki rozwoju

W toku pracy wąskim gardłem okazało się Twitter API. W wersji darmowej bardzo szybko przekroczono dopuszczalny limit i trzeba było znacznie ograniczyć ilość zapytań na minutę. Przy zastosowaniach praktycznych korzystanie z płatnych wersji API twitterowego oraz z innego źródła API z artykułami o większej przepustowości jest niezbędne. Możliwym jest również wzbogacenie projektu o dodatkowe źródła danych, jak na przykład API New York Timesa czy Guardiana.

Analiza w Sparku jest dość podstawowa. Takie były założenia projektu - skupiono się na składowaniu i zautomatyzowaniu przetwarzania danych wejściowych. Przeanalizowanie tekstów tweetów metodami NLP na pewno podniosłoby wartość biznesową całości rozwiązania.

Cenne również byłoby wprowadzenie automatycznej wizualizacji danych oraz generowania raportów, np. za pomocą narzędzia PowerBI. Dane - nawet przeprocesowane - w obecnej formie są trudno dostępne i kompletnie niezrozumiałe dla użytkownika bez zaawansowanych kompetencji technicznych.

Do pełni rozwiązania lambda brakuje w tym rozwiązaniu warstwy Speed Layer. Dodanie jej na pewno zwiększyłoby wartość biznesową projektu. Budując wyjście dla warstwy Speed Layer można by wykorzystać popularny i powszechnie stosowany system Apache Kafka, dla którego nie było miejsca w zastosowanej w tym projekcie rozwiązania.