

ĐẠI HỌC HUẾ
TRƯỜNG ĐẠI HỌC KHOA HỌC
KHOA CÔNG NGHỆ THÔNG TIN



TIỂU LUẬN
DỮ LIỆU LỚN

PHÂN TÍCH MÔ HÌNH CHUYỂN TUYẾN
(TRANSIT PATTERNS) VỚI APACHE NIFI

Sinh viên thực hiện: **NGUYỄN HOÀI NAM**

Tên học phần: **DỮ LIỆU LỚN - NHÓM 1**

Mã học phần: **2023-2024.1.TIN4523.001**

Giáo viên hướng dẫn: **T.S HỒ QUỐC DŨNG**

Huế, 12 - 2023

Mục lục

LỜI MỞ ĐẦU	1
1 GIỚI THIỆU	2
2 PHƯƠNG PHÁP THỰC HIỆN	8
2.1 Cài đặt NiFi	10
2.2 Xây dựng NiFi Process Group để mô phỏng NextBus API	12
2.3 Xây dựng NiFi Process Group để phân tích các sự kiện chuyển tuyến (Transit Events)	17
2.4 Xây dựng NiFi Process Group để xác thực dữ liệu GeoEnriched Data .	22
2.5 Xây dựng NiFi Process Group để lưu trữ dữ liệu dưới dạng JSON . .	29
2.6 Tích hợp API NextBus để lấy nguồn cấp dữ liệu trực tiếp về phương tiện công cộng	34
3 THẢO LUẬN VÀ NHẬN XÉT	38
3.1 Tổng quan	38
3.2 Giải quyết vấn đề	40
3.2.1 Làm sao để NiFi giải quyết dữ liệu từ nhiều nguồn khác nhau .	40
3.2.2 NiFi khi làm việc với các sensor	41
3.2.3 Thu thập dữ liệu tự động từ API	42
3.2.4 Ưu điểm và nhược điểm	43
4 KẾT LUẬN	44

Danh sách hình vẽ

1.1	Ảnh chụp màn hình giao diện người dùng web của Apache NiFi.	2
1.2	Ảnh hiển thị trực quan vị trí của từng cơ chế.	3
1.3	Hình ảnh minh họa vị trí của từng tùy chọn trên cửa sổ thêm bộ xử lý.	4
1.4	Hình ảnh minh họa các tab cấu hình.	5
1.5	Hình ảnh minh họa các kết nối và mối quan hệ.	6
1.6	Hình ảnh minh họa kiến trúc của NiFi.	7
2.1	Giao diện đăng nhập.	10
2.2	Hình ảnh tìm kiếm tài khoản và mật khẩu.	11
2.3	Giao diện HTML của NiFi sau khi đăng nhập thành công.	11
2.4	Hình ảnh trực quan về dữ liệu chuyển tuyến mà bạn sẽ nhập vào NiFi.	12
2.5	Hình ảnh chỉnh sửa Processor GetFile.	13
2.6	Hình ảnh thư mục chứa dữ liệu đầu vào	13
2.7	Hình ảnh chỉnh sửa Processor UnpackContent.	14
2.8	Hình ảnh chỉnh sửa Processor UnpackContent.	14
2.9	Hình ảnh chỉnh sửa Processor UnpackContent.	15
2.10	Hình ảnh chỉnh sửa Processor GetFile.	15
2.11	Hình ảnh thư mục đầu ra.	16
2.12	Hình ảnh NiFi DataFlow của SimulateXmlTransitEvents.	16
2.13	Hình ảnh chỉnh sửa Processor EvaluateXPath.	17
2.14	Hình ảnh chỉnh sửa Processor SplitXml.	18
2.15	Hình ảnh chỉnh sửa Processor ExtractTransitObservations.	19
2.16	Hình ảnh mối quan hệ giữa SimulateXmlTransitEvents và ParseTransitEvents.	19
2.17	Hình ảnh Data Nifi Flow.	20
2.18	Hình ảnh chạy SimulateXmlTransitEvents và ParseTransitEvents.	21

2.19	Hình ảnh các giá trị được ánh xạ tới tên thuộc tính của chúng.	21
2.20	Hình ảnh API Key lấy từ Google Places API.	22
2.21	Hình ảnh chỉnh sửa Processor RouteOnAttribute.	23
2.22	Hình ảnh chỉnh sửa Processor InvokeHTTP.	24
2.23	Hình ảnh chỉnh sửa Processor EvaluateJsonPath.	25
2.24	Hình ảnh chỉnh sửa Processor RouteOnAttribute.	26
2.25	Hình ảnh mối quan hệ giữa ParseTransitEvents và ValidateGeoEnrichedTransitData.	26
2.26	Hình ảnh Data Nifi Flow.	27
2.27	Hình ảnh Data Provenance cho thuộc tính FlowFile (khóa/giá trị): thành phố chứa San Francisco.	28
2.28	Hình ảnh Data Provenance cho thuộc tính FlowFile (khóa/giá trị): Neighbors nearby chứa ["Saint Francis Wood","West Portal"]	28
2.29	Hình ảnh chỉnh sửa Processor AttributesToJson.	29
2.30	Hình ảnh chỉnh sửa Processor MergeContent.	30
2.31	Hình ảnh chỉnh sửa Processor UpdateAttribute.	30
2.32	Hình ảnh chỉnh sửa Processor PutFile.	31
2.33	Hình ảnh thư mục đầu ra.	31
2.34	Hình ảnh mối quan hệ giữa ValidateGeoEnrichedTransitData và StoreDataAsJSONToDisk.	32
2.35	Hình ảnh Data Nifi Flow.	32
2.36	Hình ảnh NiFi Data Provenance Window.	33
2.37	Hình ảnh Provenance Event Window.	33
2.38	Hình ảnh View FlowFile JSON Content.	33
2.39	Hình ảnh chỉnh sửa Processor GetHTTP.	34
2.40	Hình ảnh chỉnh sửa PutFile.	35
2.41	Hình ảnh thư mục đầu ra.	35
2.42	Hình ảnh chỉnh sửa IngestNextBusXMLData.	36
2.43	Hình ảnh số lượng FlowFiles trong hàng chờ.	36
2.44	Hình ảnh giá trị mặc định số lượng đối tượng hàng đợi có thể chứa.	37
2.45	Hình ảnh Nifi DataFlow.	37

LỜI MỞ ĐẦU

Trong hệ sinh thái của Big Data và xử lý dữ liệu phân tán, NiFi là một công cụ không thể thiếu, giúp bạn dễ dàng di chuyển, chuyển đổi và quản lý dữ liệu từ nhiều nguồn đến đích một cách hiệu quả.

Khi đối mặt với bài toán phân tích mô hình chuyển tuyến, NiFi trở thành một cầu nối mạnh mẽ, giúp ta thu thập, xử lý và chuyển tải dữ liệu một cách linh hoạt. Ta có thể tận dụng các processors tích hợp để kết nối với các nguồn dữ liệu đa dạng và thực hiện các xử lý phức tạp như lọc, ánh xạ và biến đổi dữ liệu.

Nhờ vào giao diện đồ họa thân thiện, NiFi giúp dễ dàng cấu hình và theo dõi quy trình dữ liệu, giảm độ phức tạp trong việc quản lý luồng dữ liệu. Ta có thể tùy chỉnh quy trình làm việc của mình và triển khai chúng một cách linh hoạt trên môi trường phân tán.

Ngoài ra, Apache NiFi là nền tảng tích hợp đầu tiên giải quyết các thách thức thời gian thực trong việc thu thập và vận chuyển dữ liệu từ nhiều nguồn, đồng thời cung cấp lệnh tương tác và kiểm soát các luồng trực tiếp với nguồn gốc dữ liệu đầy đủ và tự động. NiFi cung cấp cơ chế thu thập dữ liệu, xử lý sự kiện đơn giản, vận chuyển và phân phối được thiết kế để đáp ứng các luồng dữ liệu đa dạng được tạo ra bởi một thế giới gồm con người, hệ thống và vạn vật được kết nối.

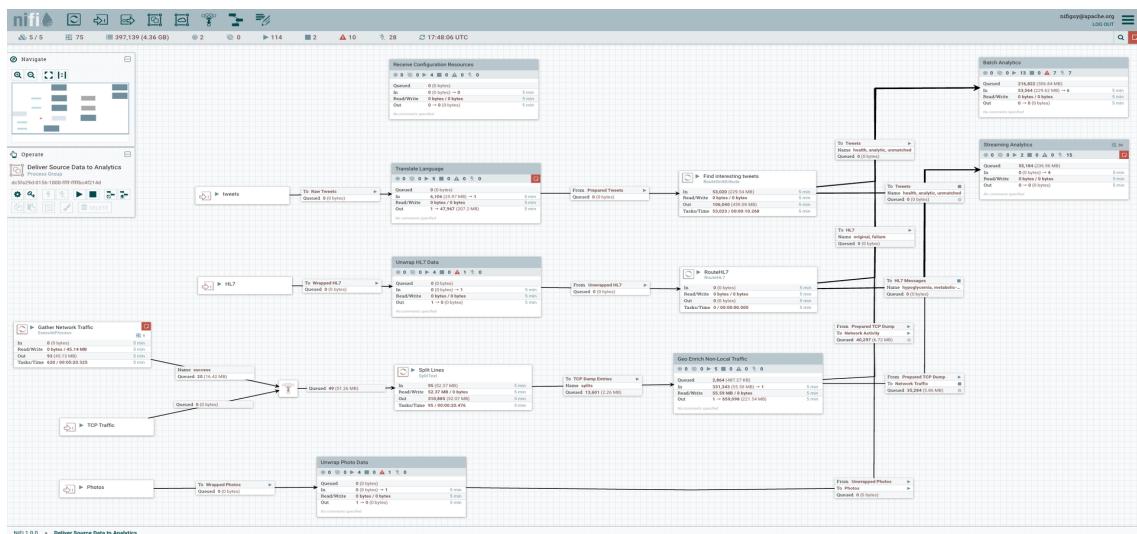
Trong bài báo cáo tiểu luận cuối kì này, một loạt các bước cài đặt được thực hiện và phân tích kĩ càng để xây dựng một hệ thống NiFi được sử dụng để xử lý luồng dữ liệu giữa các cảm biến (sensor), dịch vụ web (NextBus và Google Place API), các vị trí khác nhau và hệ thống tệp cục bộ của sandbox-team.

Chương 1

GIỚI THIỆU

Apache NiFi là một dự án phần mềm của Apache Software Foundation được thiết kế để tự động hóa luồng dữ liệu giữa các hệ thống phần mềm ((Databases, Sensors, Data Lakes, Data Platforms) [1].

Apache NiFi có nguồn gốc từ NSA Technology Transfer Program vào mùa thu năm 2014. NiFi trở thành Dự án Apache chính thức vào tháng 7 năm 2015. NiFi đã được phát triển được 8 năm. NiFi được xây dựng với ý tưởng giúp mọi người dễ dàng tự động hóa và quản lý dữ liệu đang chuyển động hơn mà không cần phải viết nhiều dòng mã. Do đó, giao diện người dùng đi kèm với các thành phần luồng dữ liệu có thể được thả vào biểu đồ và kết nối với nhau. NiFi cũng được tạo ra để giải quyết nhiều thách thức về chuyển động dữ liệu, chẳng hạn như luồng dữ liệu đa chiều, nhập dữ liệu từ bất kỳ nguồn dữ liệu nào, phân phối dữ liệu với mức độ bảo mật và quản trị cần thiết.

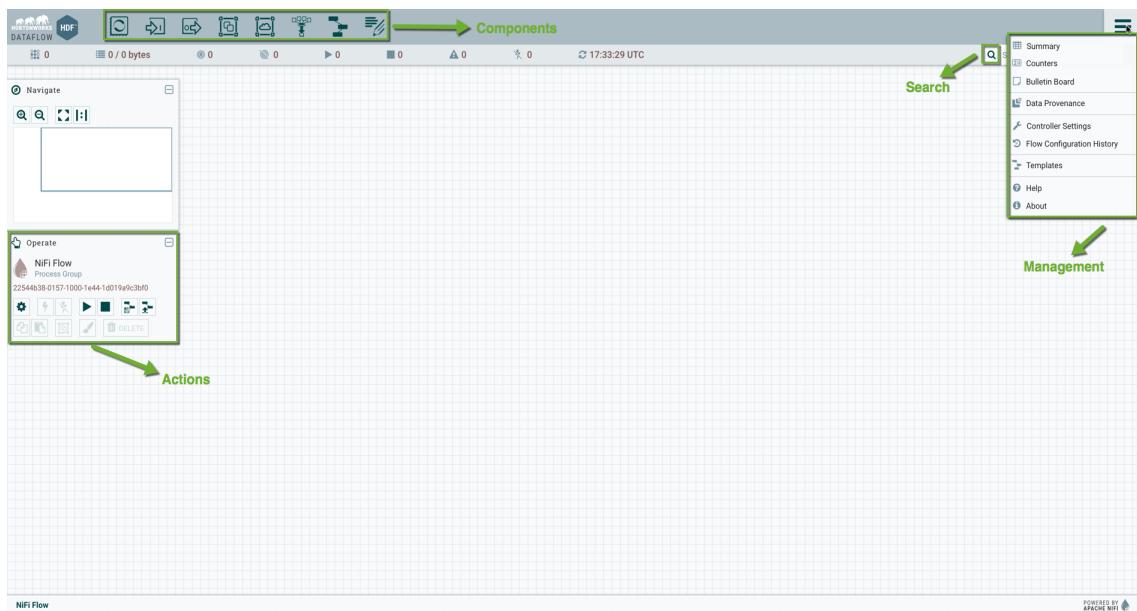


Hình 1.1: Ảnh chụp màn hình giao diện người dùng web của Apache NiFi.

Đầu tiên, hãy nhìn qua hình 1.1 để có thể nắm bắt được một hệ thống NiFi [2].

Khi NiFi được truy cập bởi những người dùng chạy NiFi từ Hortonworks DataFlow (HDF) hoặc từ máy cục bộ của họ, giao diện người dùng NiFi xuất hiện trên màn hình. Giao diện người dùng (UI) là nơi các luồng dữ liệu sẽ được phát triển. Nó bao gồm một khung vẽ và các cơ chế để xây dựng, trực quan hóa, giám sát, chỉnh sửa và quản lý các luồng dữ liệu của chúng tôi trong các hướng dẫn (Hình 1.2).

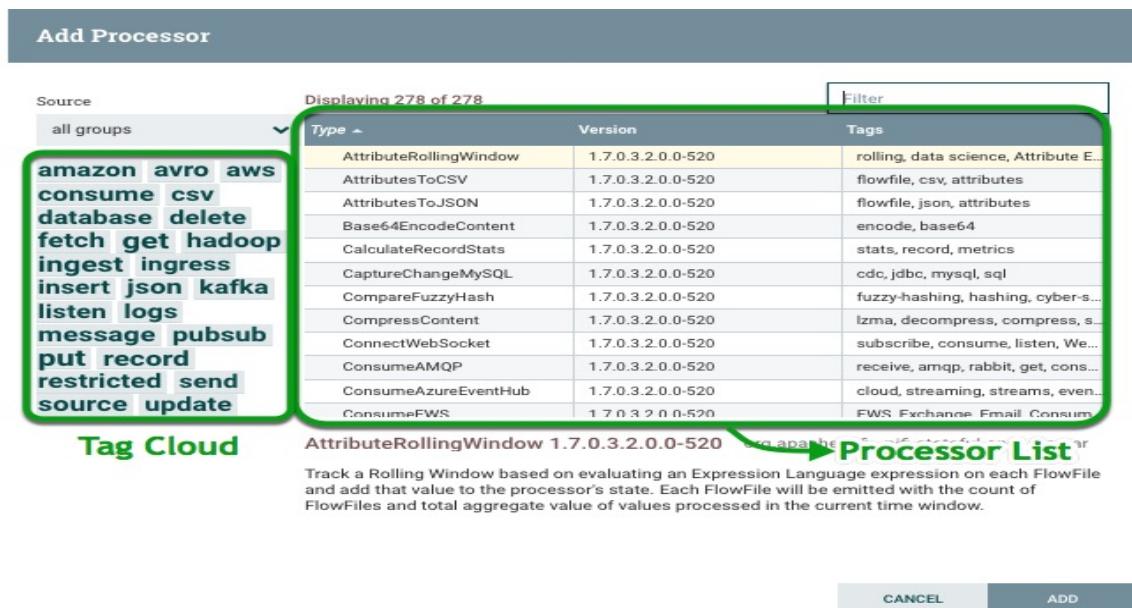
- Thanh công cụ thành phần (The components toolbar) chứa tất cả các công cụ để xây dựng luồng dữ liệu.
- Thanh công cụ hành động (The actions toolbar) bao gồm các nút thao tác các thành phần trên khung vẽ.
- Thanh công cụ quản lý (The management toolbar) có các nút để DFM quản lý luồng và quản trị viên NiFi để quản lý quyền truy cập của người dùng và thuộc tính hệ thống.
- Thanh công cụ tìm kiếm (The search toolbar) cho phép người dùng tìm kiếm bất kỳ thành phần nào trong luồng dữ liệu.



Hình 1.2: Ảnh hiển thị trực quan vị trí của từng cơ chế.

Mỗi luồng dữ liệu đều yêu cầu một bộ xử lý (processor). Hãy xem qua cửa sổ thêm bộ xử lý bằng hình 1.3. Có 3 tùy chọn để tìm bộ xử lý mong muốn:

- Danh sách bộ xử lý (The processor list) chứa gần 190 mục kèm theo mô tả cho từng bộ xử lý.
- Danh sách thẻ (The tag cloud) giảm danh sách theo danh mục, nếu biết bộ xử lý mong muốn của dự án liên quan đến trường hợp sử dụng cụ thể nào điều này sẽ giúp chọn thẻ và tìm bộ xử lý thích hợp nhanh hơn.
- Thanh bộ lọc tìm kiếm (The filter bar) bộ xử lý dựa trên từ khóa đã nhập.

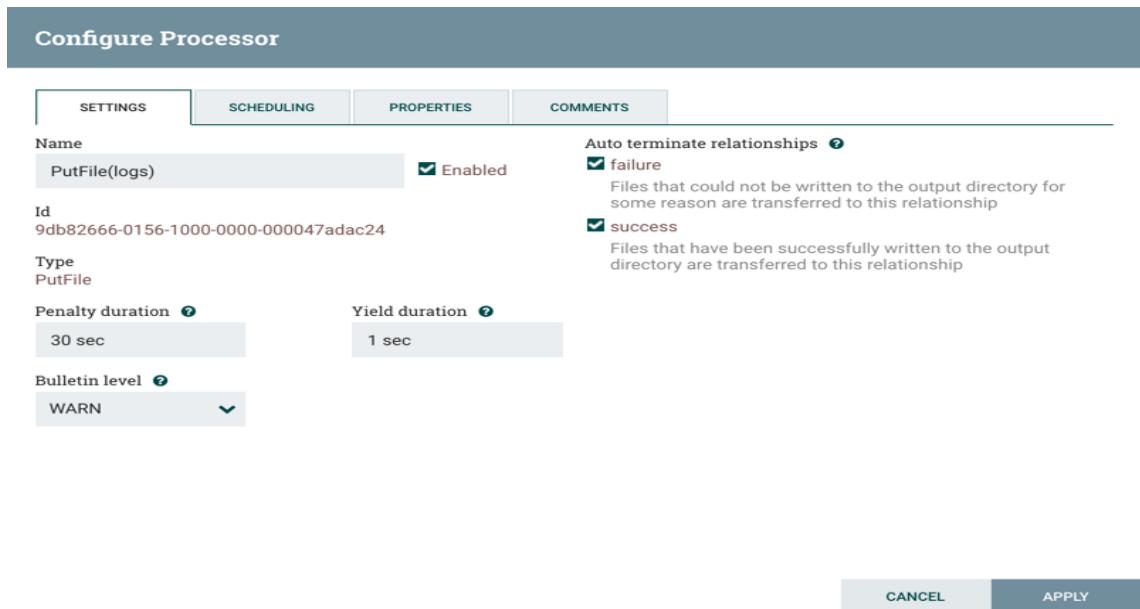


Hình 1.3: Hình ảnh minh họa vị trí của từng tùy chọn trên cửa sổ thêm bộ xử lý.

Khi thêm từng bộ xử lý vào luồng dữ liệu, phải đảm bảo chúng được cấu hình đúng. Người quản lý DataFlow điều hướng xung quanh 4 tab cấu hình để kiểm soát hành vi cụ thể của bộ xử lý và hướng dẫn bộ xử lý cách xử lý dữ liệu đang truyền (Hình 1.4). Trong bài báo cáo này, ta sẽ dành phần lớn thời gian để sửa đổi các thuộc tính.

- Tab Cài đặt (The settings tab) cho phép người dùng thay đổi tên bộ xử lý, xác định mối quan hệ và bao gồm nhiều thông số khác nhau.
- Tab Lập lịch (The scheduling tab) ảnh hưởng đến cách bộ xử lý được lên lịch chạy.
- Tab Thuộc tính (The properties) ảnh hưởng đến hành vi cụ thể của bộ xử lý.

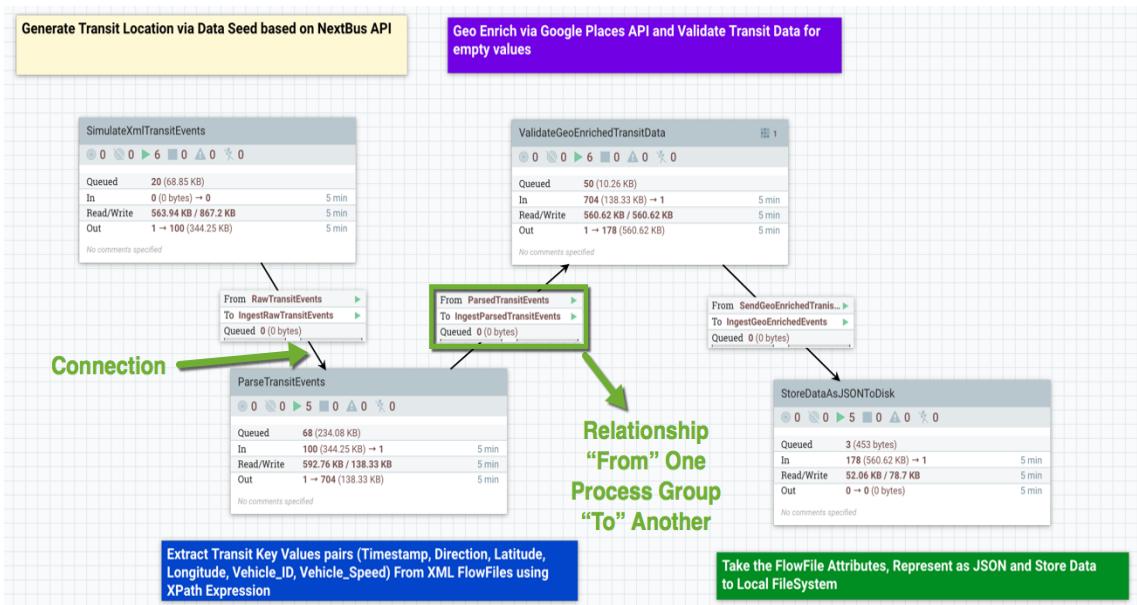
- Tab Nhận xét (The comments tab) cung cấp một nơi để DFM đưa vào thông tin hữu ích về trường hợp sử dụng của bộ xử lý.



Hình 1.4: Hình ảnh minh họa các tab cấu hình.

Khi mỗi cấu hình bộ xử lý được hoàn thành, chúng ta phải kết nối nó với một thành phần khác. Kết nối (connection) là mối liên kết giữa các bộ xử lý (hoặc các thành phần) có chứa ít nhất một mối quan hệ.

Người dùng chọn mối quan hệ và dựa trên kết quả xử lý sẽ xác định dữ liệu được định tuyến ở đâu. Bộ xử lý có thể không có hoặc có nhiều mối quan hệ tự động chấm dứt. Nếu kết quả xử lý của FlowFile là đúng đắn với bộ xử lý có mối quan hệ gắn liền với chính nó thì FlowFile sẽ bị xóa khỏi luồng (Hình 1.5).



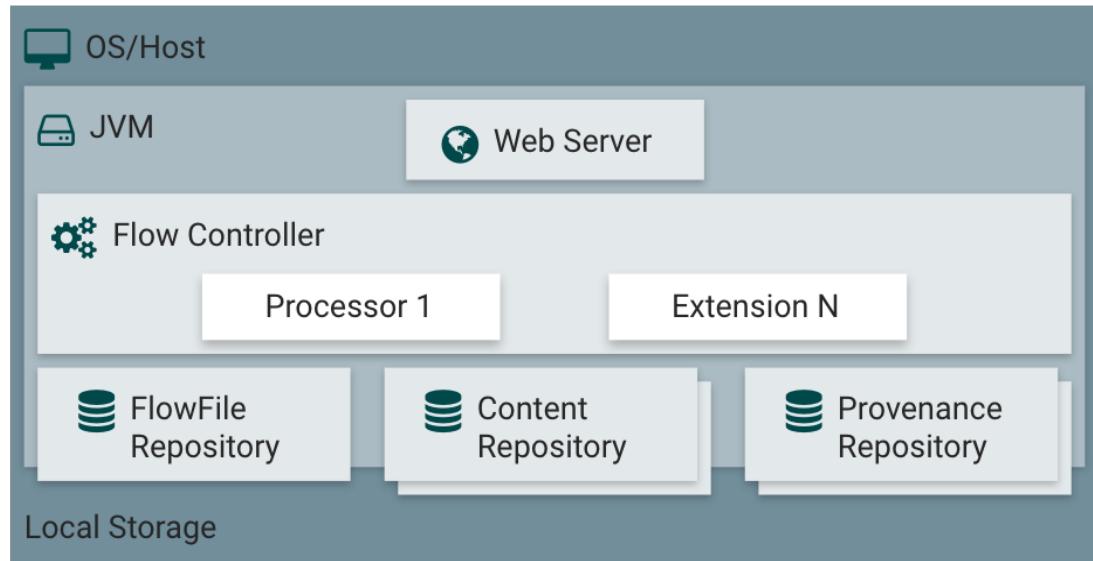
Hình 1.5: Hình ảnh minh họa các kết nối và mối quan hệ.

Như vậy, ta rút được các khái niệm cốt lõi trong NiFi như bảng 1.1 dưới đây.

Bảng 1.1: Khái niệm cốt lõi của NiFi.

Thuật ngữ NiFi	Mô tả
FlowFile	Dữ liệu được đưa vào NiFi sẽ di chuyển trong hệ thống. Dữ liệu này chứa các thuộc tính và có thể chứa nội dung.
Processor	Công cụ lấy dữ liệu từ các nguồn bên ngoài, thực hiện các hành động trên các thuộc tính và nội dung của FlowFiles và xuất bản dữ liệu ra nguồn bên ngoài.
Connection	Liên kết giữa các bộ xử lý có chứa hàng đợi và các mối quan hệ ánh hưởng đến việc định tuyến dữ liệu.
Back Pressure	Ngăn hệ thống khỏi bị tràn dữ liệu bằng cách kiểm soát số lượng hoặc kích thước dữ liệu của FlowFiles có thể được lưu trữ trong hàng đợi.
Flow Controller	Hoạt động như một Nhà môi giới để tạo điều kiện thuận lợi cho việc trao đổi FlowFiles giữa các bộ xử lý
Process Group	Cho phép tạo các thành phần mới dựa trên thành phần của bộ xử lý, kênh, v.v.
Data Provenance	Lịch sử các hành động xảy ra trên dữ liệu khi nó di chuyển trong suốt luồng. Cho phép người dùng kiểm tra dữ liệu từ bất kỳ bộ xử lý hoặc thành phần nào trong khi FlowFiles di chuyển trong suốt luồng dữ liệu.
Controller Service	Gói tham số cấu hình và mã thực hiện tác vụ nào đó ở chế độ nền: Cho bộ xử lý bản ghi biết cách diễn giải dữ liệu, thiết lập tham số kết nối với các dịch vụ bên ngoài (cơ sở dữ liệu, API), gửi số liệu thống kê về NiFi đến dịch vụ giám sát và chia sẻ trạng thái với các dịch vụ bộ nhớ đám mây.

Hãy cùng tìm hiểu sâu hơn về kiến trúc cho phép NiFi hoạt động rất tốt khi xây dựng DataFlows nhằm giải quyết các trường hợp sử dụng khác nhau (Hình 1.6).



Hình 1.6: Hình ảnh minh họa kiến trúc của NiFi.

Có thể nhận thấy rằng, NiFi thực thi trong Máy ảo Java (JVM) nằm trong hệ điều hành máy chủ (OS/Host). Trong JVM, ta có:

- Có thể nhận thấy rằng, NiFi thực thi trong Máy ảo Java (JVM) nằm trong Hệ điều hành máy chủ (OS/Host). Trong JVM, ta có:
 - Web Server - cho phép truy cập Giao diện người dùng của NiFi từ trình duyệt web.
 - Bộ điều khiển luồng hoạt động (The Flow Controller) hoạt động như bộ não, nó cung cấp các luồng cho các tiện ích mở rộng (bộ xử lý tùy chỉnh) và theo dõi tất cả các hoạt động được thực hiện bởi các tiện ích mở rộng.
 - Kho lưu trữ FlowFile (FlowFile Repository) là khu vực NiFi theo dõi tất cả các cập nhật trạng thái liên quan đến FlowFiles khi chúng di chuyển khắp DataFlow.
 - Kho lưu trữ nội dung (Content Repository) là vị trí chứa các byte nội dung của FlowFiles.
 - Kho lưu trữ xuất xứ (Provenance Repository) bao gồm tất cả dữ liệu sự kiện.

Chương 2

PHƯƠNG PHÁP THỰC HIỆN

Trong bài báo cáo này, giả sử tình huống rằng ban quy hoạch thành phố đang đánh giá nhu cầu về đường cao tốc mới, để cải thiện phân tích giao thông, người quy hoạch thành phố muốn tận dụng dữ liệu thời gian thực để hiểu sâu hơn về mô hình giao thông. NiFi đã được chọn để tích hợp dữ liệu thời gian thực này, làm việc với dữ liệu của MUNI Transit San Francisco agency, được thu thập từ Nguồn cấp dữ liệu trực tiếp XML của NextBus API, xử lý vị trí, tốc độ của phương tiện và các biến khác.

Dữ liệu: https://github.com/xvanausloos/hdp_data_tutorials/blob/master/tutorials/hdf/analyze-transit-patterns-with-apache-nifi/assets/transit_data_seed.zip

Sau khi dữ liệu được nhập, với tư cách là người quản lý DataFlow (DataFlow Manager - DFM), ta sẽ tạo 2 nhóm quy trình hoặc phần của luồng dữ liệu để xử lý một mục đích cụ thể trong quá trình tiền xử lý dữ liệu:

- Một nhóm quy trình (Process group) là một bộ xử lý phức tạp bao gồm nhiều bộ xử lý (processor). Ta sẽ tạo nhóm quy trình Data Management để theo dõi và nhận phản hồi về trạng thái hiện tại của NiFi DataFlow. Ta sẽ sử dụng bản tin (bulletins) để khắc phục sự cố trong luồng dữ liệu. Bản tin được đặt trên bộ xử lý và thanh công cụ quản lý, chúng cung cấp mèo sử dụng công cụ về thời gian, mức độ nghiêm trọng và thông điệp của cảnh báo.
- Trong khi dữ liệu được quản lý, ta sẽ tạo nhóm quy trình Data Enrichment để nâng cao, tinh chỉnh và cải thiện chất lượng dữ liệu nhằm làm cho dữ liệu trở nên có ý nghĩa và có giá trị đối với người dùng. Ta sẽ sử dụng NiFi để làm phong phú dữ liệu thời gian thực về mặt địa lý nhằm hiển thị các địa điểm lân cận khi địa

điểm thay đổi.

Các bước cài đặt được thực hiện dựa theo hướng dẫn ở hai đường liên kết dưới đây:

Link 1: https://github.com/xvanausloos/hdp_data_tutorials/tree/master/tutorials/hdf/analyze-transit-patterns-with-apache-nifi

Link 2: <https://github.com/apache/nifi>

Bao gồm 6 bước:

- Đầu tiên, cài đặt NiFi.
- Bước 2, xây dựng một NiFi Process Group để mô phỏng NextBus API.
- Bước 3, xây dựng một NiFi Process Group để phân tích các sự kiện chuyển tuyến (Transit Events).
- Bước 4, xây dựng một NiFi Process Group để xác thực dữ liệu GeoEnriched Data.
- Bước 5, xây dựng một NiFi Process Group để lưu trữ dữ liệu dưới dạng JSON,
- Cuối cùng, tích hợp API NextBus để lấy nguồn cấp dữ liệu trực tiếp về phương tiện công cộng.

2.1 Cài đặt NiFi

Có khá nhiều cách để cài đặt giao diện HTML UI Nifi, trong đó có:

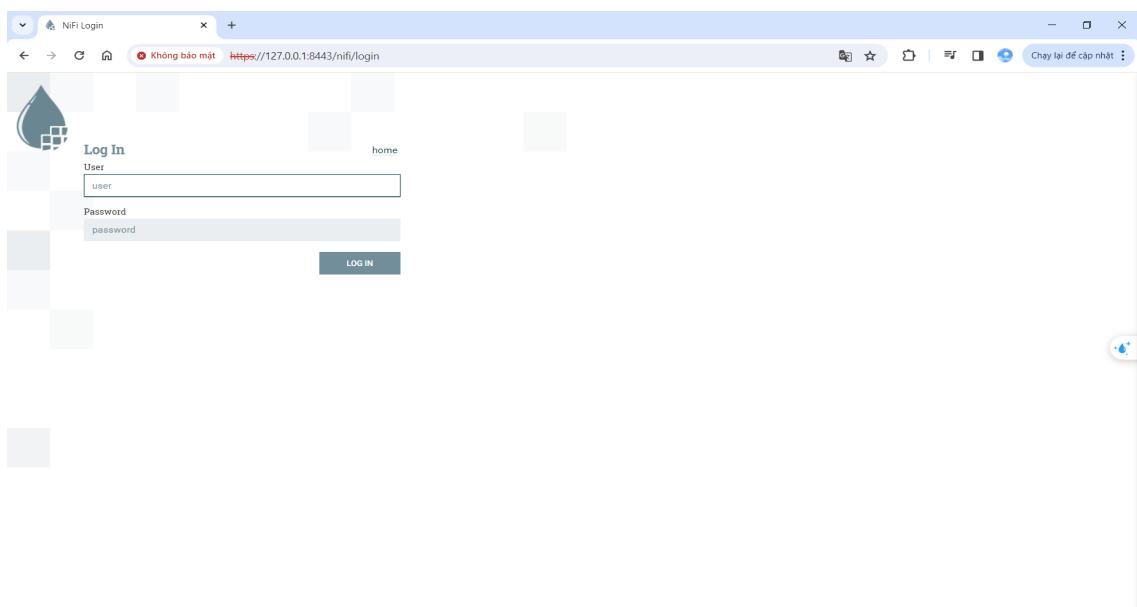
- Truy cập giao diện người dùng HTML Nifi thông qua Ambari Dashboard.
- Khởi chạy giao diện người dùng HTML Nifi từ HDF Splash Quick Links.

Tuy nhiên, để tiết kiệm được dung lượng, cũng như tối ưu thời gian thực hiện dự án, ta tiến hành thực hiện tải Nifi theo hướng dẫn của 1 trong 2 đường liên kết dưới đây:

<https://www.udemy.com/course/apache-nifi-the-beginner-guide/?referralCode=3FCCF7F1DF404032EE74>

https://www.youtube.com/watch?v=ydhcxBNJ6Fs&list=PL55symSEWBbMBSnNW_Aboh2TpYkNIFMgb&index=6

Sau khi cài đặt, ta được giao diện Nifi như hình 2.1:



Hình 2.1: Giao diện đăng nhập.

Lúc này, ta cần vào file nifi-app.log để tìm kiếm tài khoản và mật khẩu như hình

2.2:

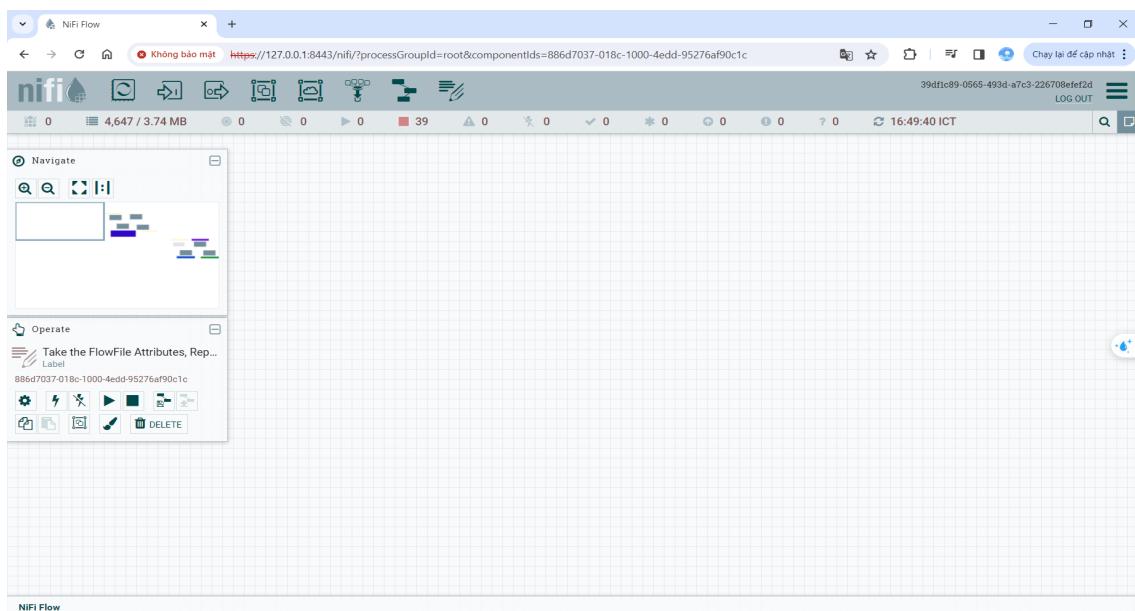
```
2023-12-19 23:41:03,539 INFO [main] o.apache.nifi.controller.FlowController Creating Provenance Repository
2023-12-19 23:41:03,593 INFO [main] o.a.n.p.store.WriteAheadStorePartition After recovering .\provenance_r...
2023-12-19 23:41:03,596 INFO [main] o.a.n.p.index.lucene.LuceneEventIndex Will avoid re-indexing Provenanc...
2023-12-19 23:41:03,596 INFO [main] o.apache.nifi.controller.FlowController Creating Content Repository [o...
2023-12-19 23:41:03,604 INFO [main] o.a.n.c.repository.FileSystemRepository Maximum Threshold for Containe...
2023-12-19 23:41:03,605 INFO [main] o.a.n.c.repository.FileSystemRepository Initializing FileSystemReposit...
2023-12-19 23:41:04,042 INFO [main] org.wali.MinimalLockingWriteAheadLog org.wali.MinimalLockingWriteAhead...
2023-12-19 23:41:04,042 INFO [main] org.wali.MinimalLockingWriteAheadLog Successfully recovered 0 records ...
2023-12-19 23:41:04,061 INFO [main] org.wali.MinimalLockingWriteAheadLog org.wali.MinimalLockingWriteAhead...
2023-12-19 23:41:04,276 INFO [main] o.apache.nifi.controller.FlowController Not enabling RAW Socket Site-to-S...
2023-12-19 23:41:06,346 INFO [main] o.s.s.web.DefaultSecurityFilterChain Will secure any request with [org...
2023-12-19 23:41:06,470 INFO [main] o.a.n.a.s.u.SingleUserLoginIdentityProvider

Generated Username [39df1c89-0565-493d-a7c3-226708efef2d]
Generated Password [grofnEiGDbBq7XFetpMS90Ct6o9y3mYr]

2023-12-19 23:41:06,471 INFO [main] o.a.n.a.s.u.SingleUserLoginIdentityProvider Run the following command ...
2023-12-19 23:41:06,742 INFO [main] o.a.n.a.s.u.SingleUserLoginIdentityProvider Updating Login Identity Pr...
2023-12-19 23:41:07,114 INFO [main] o.a.n.w.c.ApplicationStartupContextListener Starting Flow Controller...
2023-12-19 23:41:07,140 INFO [main] o.apache.nifi.controller.FlowController Successfully synchronized cont...
2023-12-19 23:41:07,275 INFO [main] o.a.n.wali.SequentialAccessWriteAheadLog Recovering records from Write...
2023-12-19 23:41:07,276 INFO [main] o.a.n.wali.SequentialAccessWriteAheadLog No Snapshot File to recover fi...
2023-12-19 23:41:07,277 INFO [main] o.a.n.wali.SequentialAccessWriteAheadLog Successfully recovered 0 recor...
2023-12-19 23:41:07,291 INFO [main] o.a.n.wali.SequentialAccessWriteAheadLog Checkpointed Write-Ahead Log ...
2023-12-19 23:41:07,292 INFO [main] o.a.n.c.r.WriteAheadFlowFileRepository Successfully restored 0 FlowFil...
2023-12-19 23:41:07,399 INFO [main] o.apache.nifi.controller.FlowController Performed initial validation o...
```

Hình 2.2: Hình ảnh tìm kiếm tài khoản và mật khẩu.

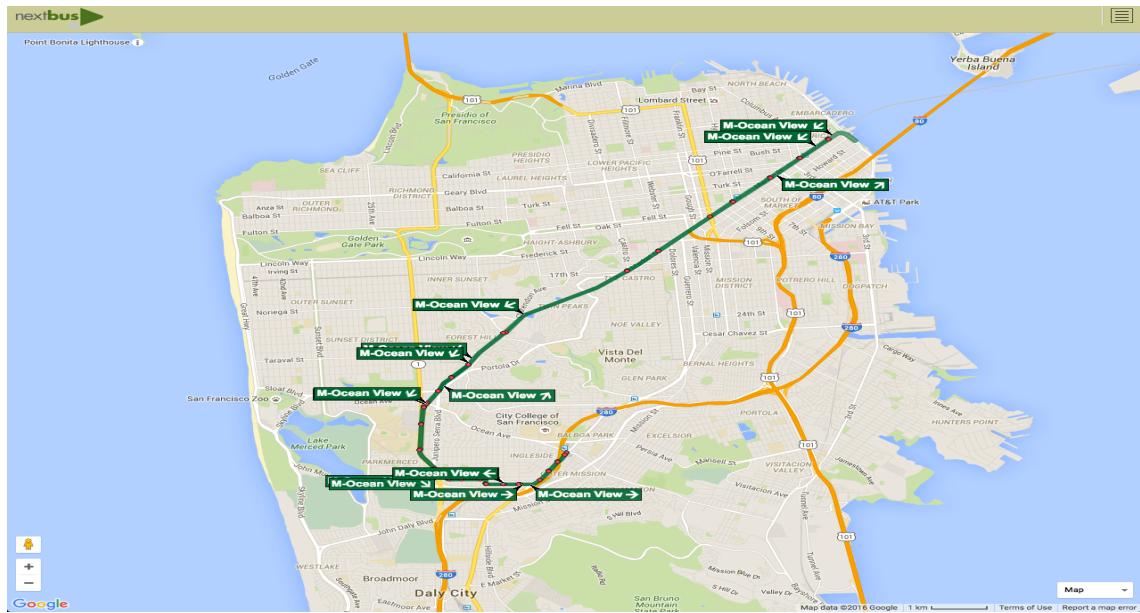
Và cuối cùng, ta được giao diện HTML của NiFi như hình 2.3:



Hình 2.3: Giao diện HTML của NiFi sau khi đăng nhập thành công.

2.2 Xây dựng NiFi Process Group để mô phỏng NextBus API

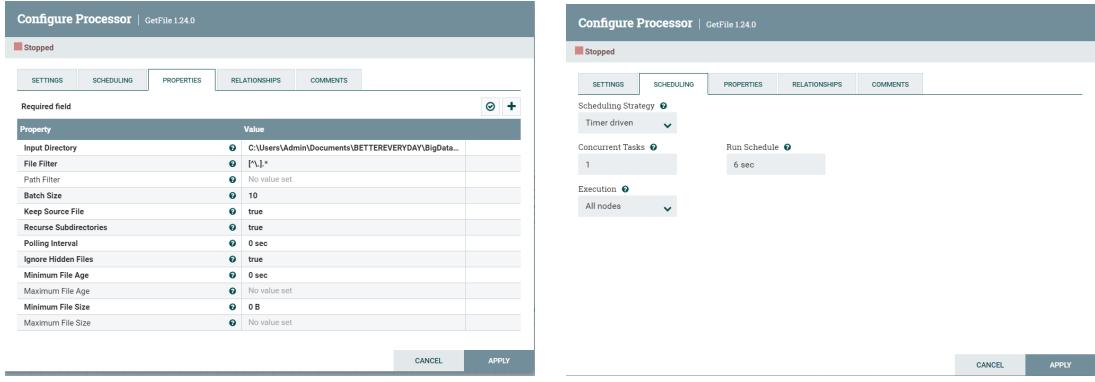
Bây giờ, ta cần xây dựng NiFi DataFlow và đóng gói nó thành một nhóm quy trình (process group) để mô phỏng nguồn cấp dữ liệu chuyển tuyến (transit) API NextBus và kiểm tra quá trình tạo dữ liệu từ trình mô phỏng (Hình 2.4).



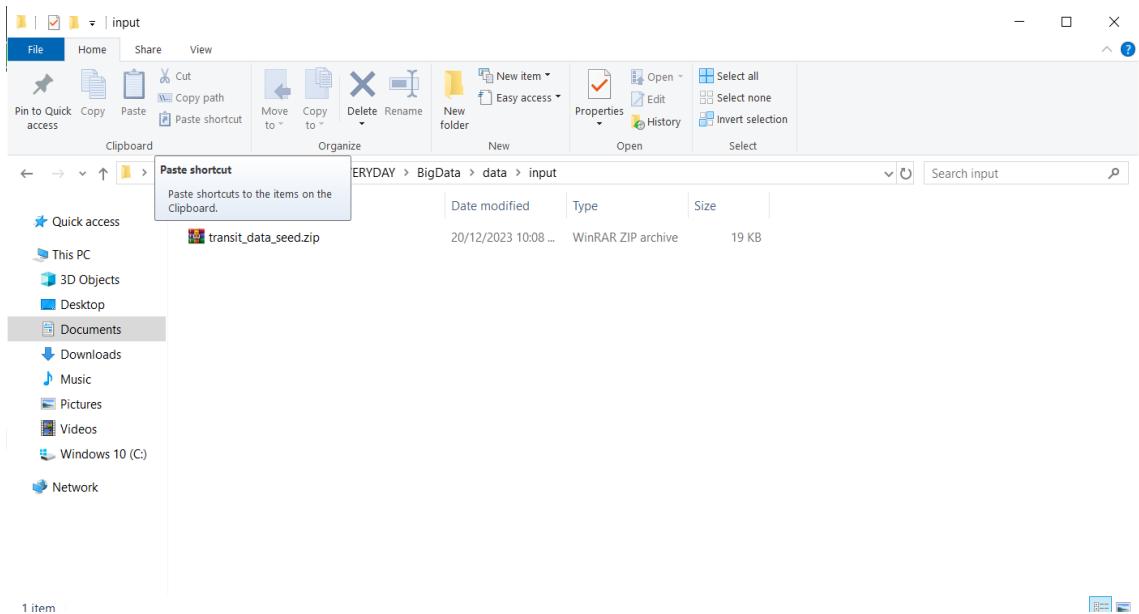
Hình 2.4: Hình ảnh trực quan về dữ liệu chuyển tuyến mà bạn sẽ nhập vào NiFi.

Đầu tiên, chúng ta cần tạo một Label và kéo dài khoảng 24 ô vuông, sau đó, ta đặt tên nó là Generate Transit Location via Data Seed based on NextBus API và chuyển cỡ chữ thành 18px.

- Tiếp theo, thực hiện tạo một Process Group, đặt tên là SimulateXmlTransitEvents.
- Bước 2, vào trong SimulateXmlTransitEvents, tiến hành thêm một processor GetFile. Processor GetFile có nhiệm vụ tạo FlowFiles từ các tập tin trong một thư mục. NiFi sẽ bỏ qua các tệp mà nó không có quyền đọc. Tiến hành chỉnh sửa processor GetFile như hình 2.5. Lưu ý, Input Directory cần là thư mục có dữ liệu đầu vào ở dạng file zip để thực hiện processor UnpackContent ở phía sau (Hình 2.6).

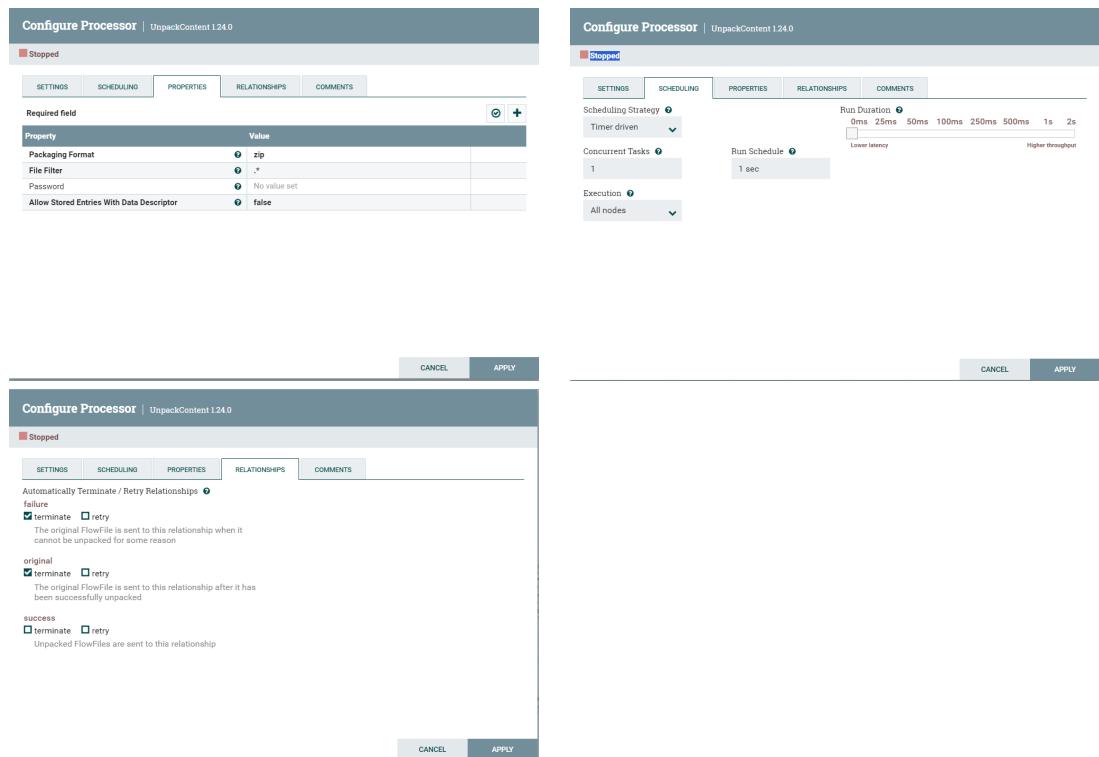


Hình 2.5: Hình ảnh chỉnh sửa Processor GetFile.



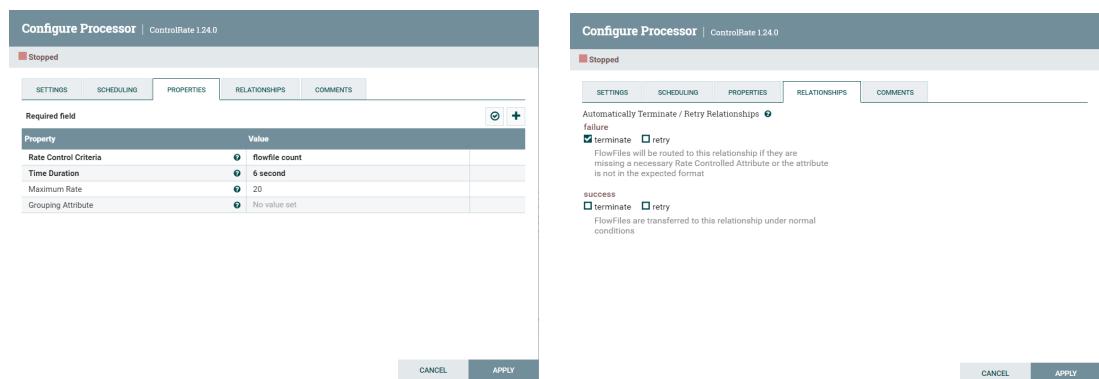
Hình 2.6: Hình ảnh thư mục chứa dữ liệu đầu vào

- Bước 3, thêm processor UnpackContent để giải nén dữ liệu ở dạng zip. Processor UnpackContent có nhiệm vụ giải nén file dữ liệu ở dạng Packaging Format (ở trong bài này là dạng zip). Sau đó, tiến hành tạo mối quan hệ (connection) giữa processor GetFile và UnpackContent. Sau đó, ta chỉnh sửa processor UnpackContent theo hình 2.7. (Lưu ý, chỉnh sửa Run Schedule nghĩa là bộ xử lý thực thi một tác vụ cứ sau 1 giây và tránh áp lực ngược xuồng bộ xử lý tiếp theo).



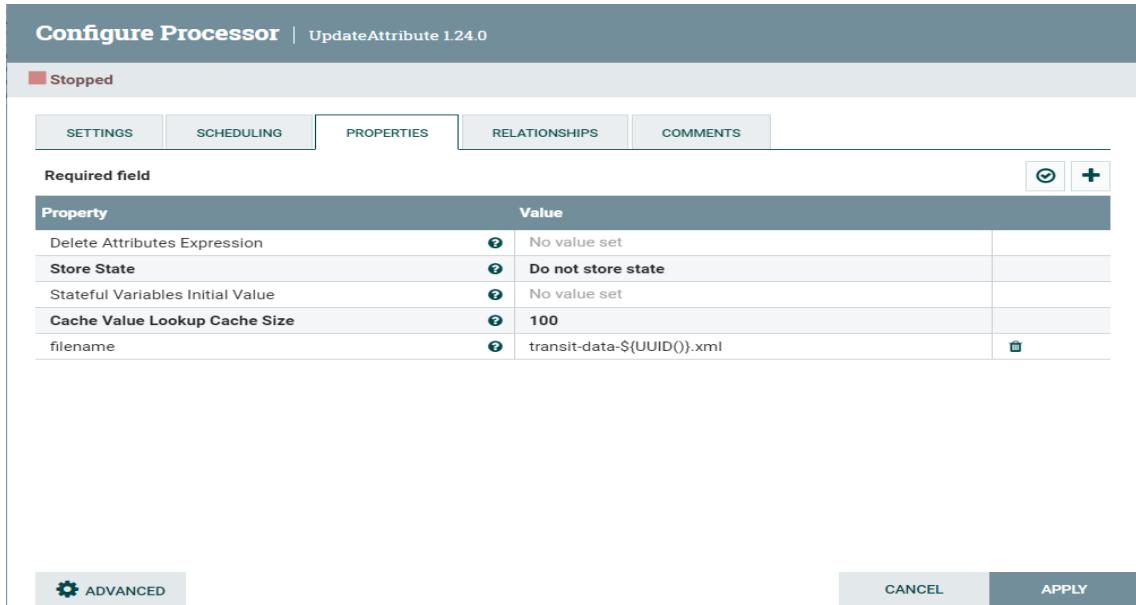
Hình 2.7: Hình ảnh chỉnh sửa Processor UnpackContent.

- Bước 4, thêm processor ControlRate. Sau đó, tiến hành tạo mối quan hệ giữa UnpackContent và ControlRate. Tiếp theo, chỉnh sửa processor Controlrate như hình 2.8. (Lưu ý, Rate Control Criteria có công việc hướng dẫn bộ xử lý đếm số lượng FlowFiles trước khi quá trình truyền diễn ra; Maximum Rate có công việc hướng dẫn bộ xử lý truyền 20 FlowFiles cùng một lúc; Time Duration có công việc làm cho nó chỉ có 20 FlowFiles sẽ truyền qua bộ xử lý này cứ sau 6 giây).



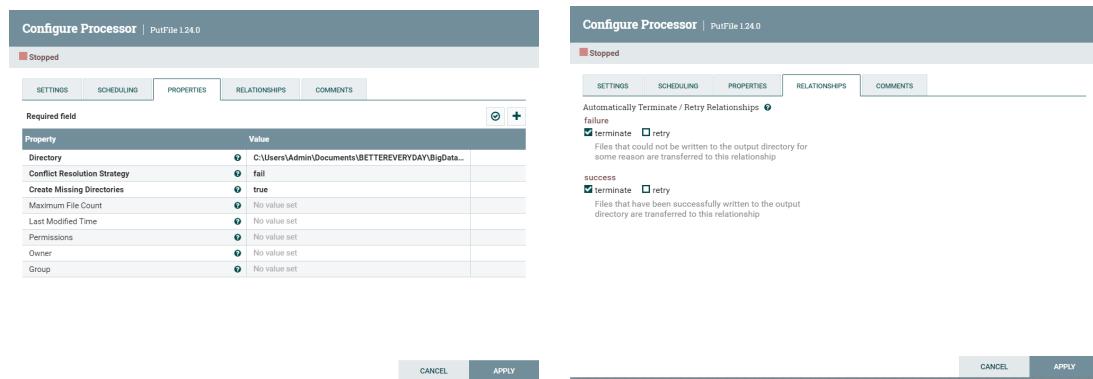
Hình 2.8: Hình ảnh chỉnh sửa Processor UnpackContent.

- Bước 5, thêm processor UpdateAttribute. Sau đó, tạo mối quan hệ giữa ControlRate và UpdateAttribute. Sau đó, chỉnh sửa processor UpdateAttribute như hình 2.9.

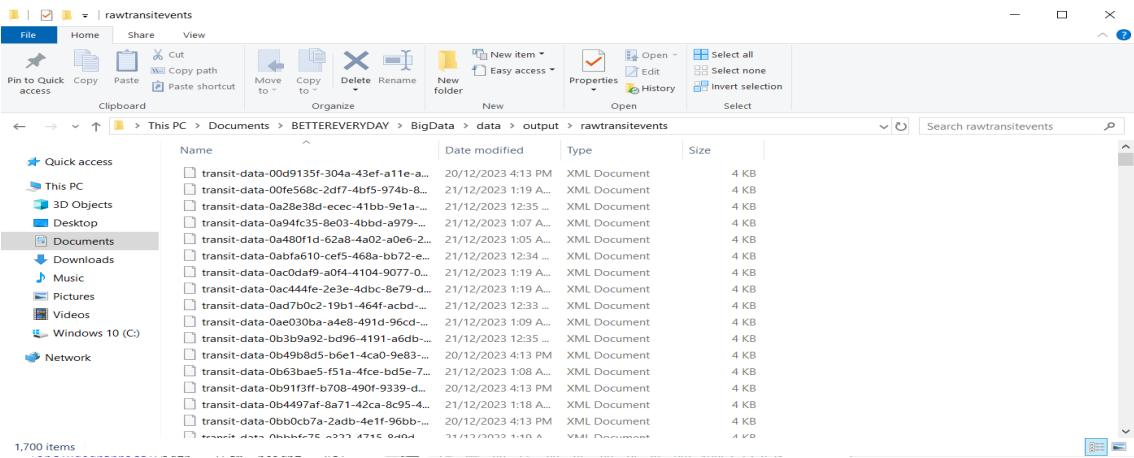


Hình 2.9: Hình ảnh chỉnh sửa Processor UnpackContent.

- Bước 6, thêm processor PutFile. Sau đó, chỉnh sửa PutFile như hình 2.10 (Lưu ý, ở Dicectory chính là thư mục đầu ra của dữ liệu như hình 2.11.)



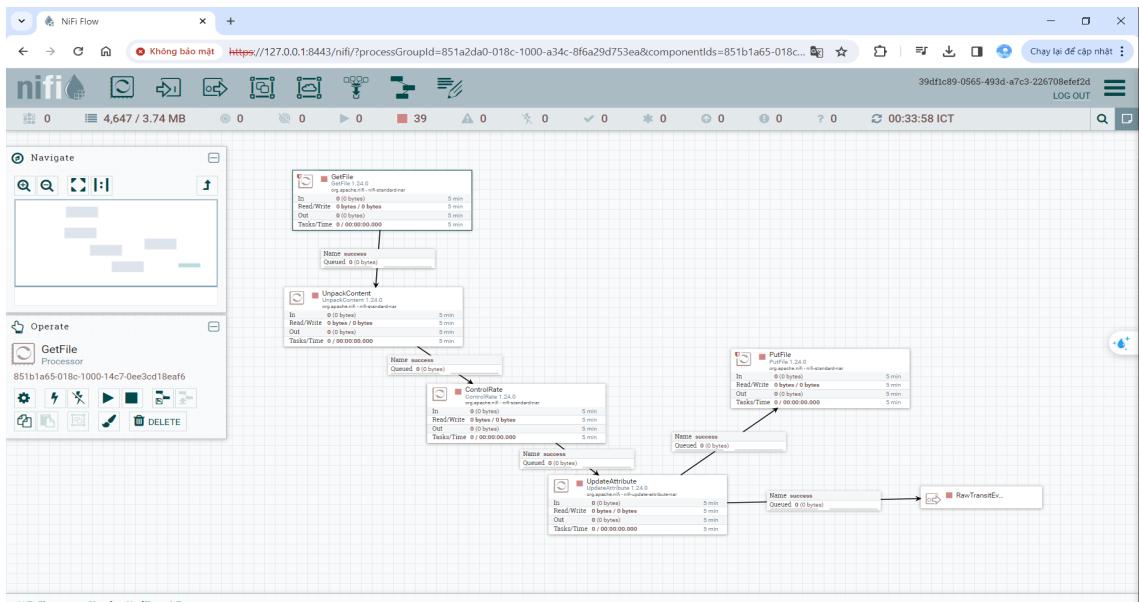
Hình 2.10: Hình ảnh chỉnh sửa Processor GetFile.



Hình 2.11: Hình ảnh thư mục đầu ra.

- Cuối cùng, thêm Output Port, đặt tên là RawTransitEvents. Sau đó, tạo mối quan hệ giữa UpdateAttribute với RawTransitEvents.

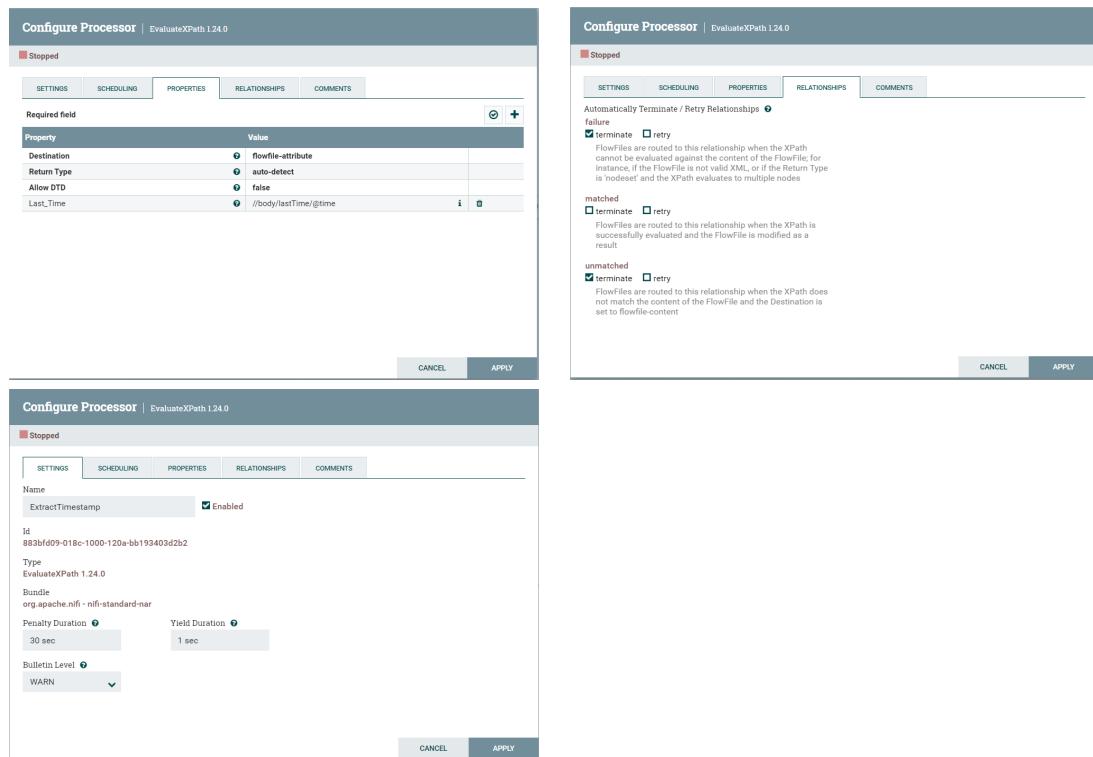
Như vậy, sau khi hoàn tất các bước, ta được NiFi DataFlow như hình 2.12. Tóm tắt lại, ta vừa xây dựng nhóm quy trình NiFi SimulateXmlTransitEvents để sao chép API NextBus, tạo ra dữ liệu chuyển tuyến cho hành khách. GetFile để nhận dữ liệu chuyển tuyến; UnpackContent đã giải nén tệp zip dữ liệu chuyển tuyến và định tuyến dữ liệu đến phần còn lại của luồng; ControlRate kiểm soát tốc độ phân phối mỗi FlowFile cho các thành phần còn lại của luồng; Output Port cho phép phân tán FlowFiles sang nhóm quy trình tiếp theo mà ta sẽ xây dựng trong bước tiếp theo tên ParseTransitEvents.



Hình 2.12: Hình ảnh NiFi DataFlow của SimulateXmlTransitEvents.

2.3 Xây dựng NiFi Process Group để phân tích các sự kiện chuyển tuyến (Transit Events)

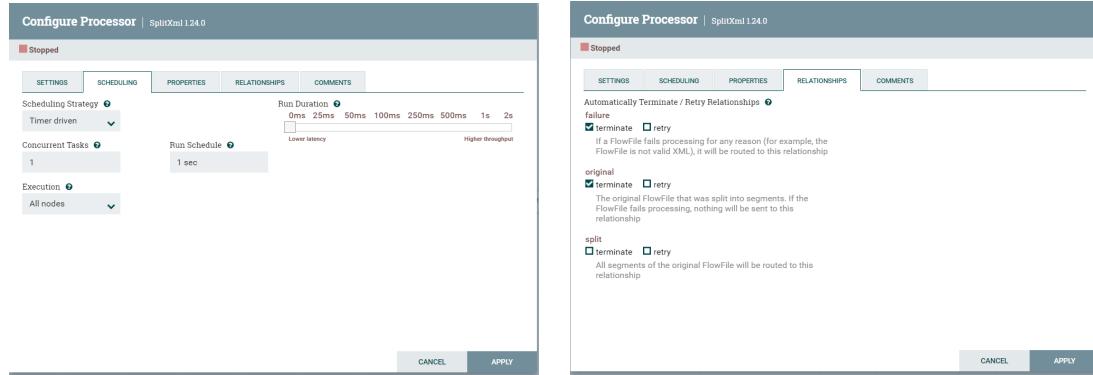
- Thực hiện tạo một Process Group, đặt tên là ParseTransitEvents.
- Bước 2, vào trong ParseTransitEvents, tiến hành thêm một Input Port, đặt tên IngestRawTransitEvents. Input Port sẽ nhận dữ liệu từ Output Port của process group trước.
- Bước 3, thêm processor EvaluateXPath, đổi tên thành ExtractTimestamp. Processor EvaluateXPath có nhiệm vụ trích xuất dấu thời gian của lần cập nhật cuối cùng cho dữ liệu vị trí phương tiện được trả về từ mỗi FlowFile. Sau đó, tiến hành tạo mối quan hệ (connection) giữa Input Port và processor EvaluateXPath. Sau đó, ta chỉnh sửa processor EvaluateXPath theo hình 2.13. (Lưu ý, Destination là kết quả từ việc đánh giá XPath được lưu trữ trong thuộc tính FlowFile; LastTime là một thuộc tính FlowFile và biểu thức XPath truy xuất giá trị của nút thời gian trong tệp XML).



Hình 2.13: Hình ảnh chỉnh sửa Processor EvaluateXPath.

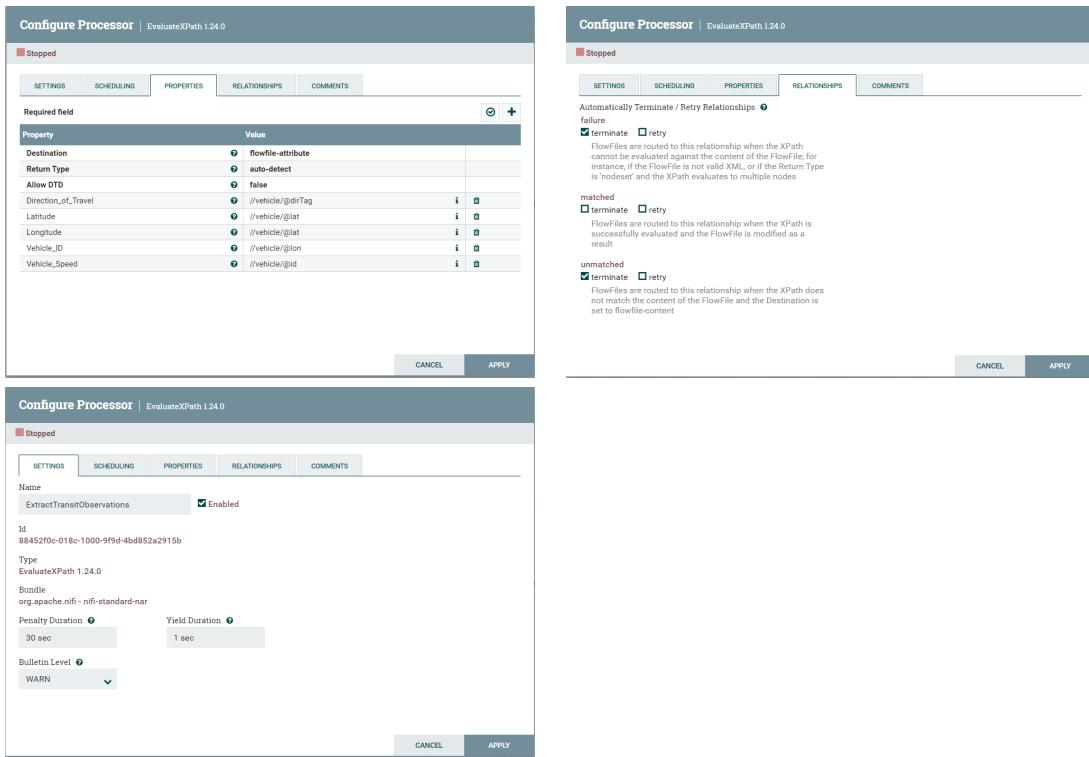
- Bước 4, thêm processor SplitXml. Processor SplitXml có công việc chia các phần tử con của cha thành các FlowFiles riêng biệt. Vì phương tiện là phần tử

con trong tệp xml của chúng tôi nên mỗi phần tử phương tiện mới được lưu trữ riêng biệt. Sau đó, tiến hành tạo mối quan hệ giữa EvaluateXPath và SplitXml. Tiếp theo, chỉnh sửa processor SplitXml như hình 2.14. (Lưu ý, Run Schedule nghĩa là bộ xử lí thực hiện một tác vụ cứ sau 1 giây).



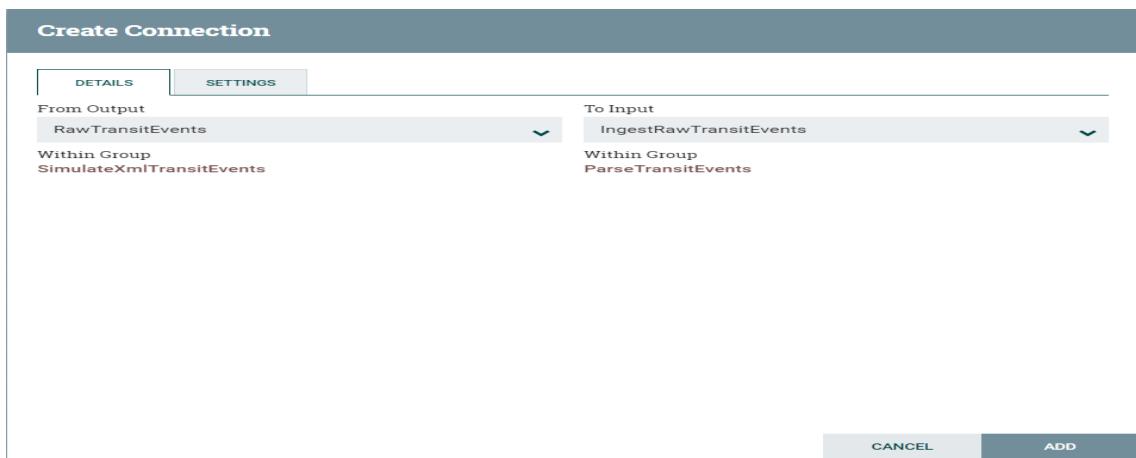
Hình 2.14: Hình ảnh chỉnh sửa Processor SplitXml.

- Bước 5, thêm processor EvaluateXPath. Sau đó, tạo mối quan hệ giữa SplitXml và EvaluateXPath. Sau đó, chỉnh sửa processor EvaluateXPath như hình 2.15. Đổi tên processor này thành ExtractTransitObservations. (Lưu ý, Destination được đặt thành thuộc tính FlowFile vì kết quả của các giá trị từ biểu thức XPath cần được lưu trữ trong thuộc tính FlowFile; năm thuộc tính mỗi biểu thị dữ liệu liên quan đến quan sát chuyển tuyến được liên kết với dấu thời gian được trích xuất trước đó.)



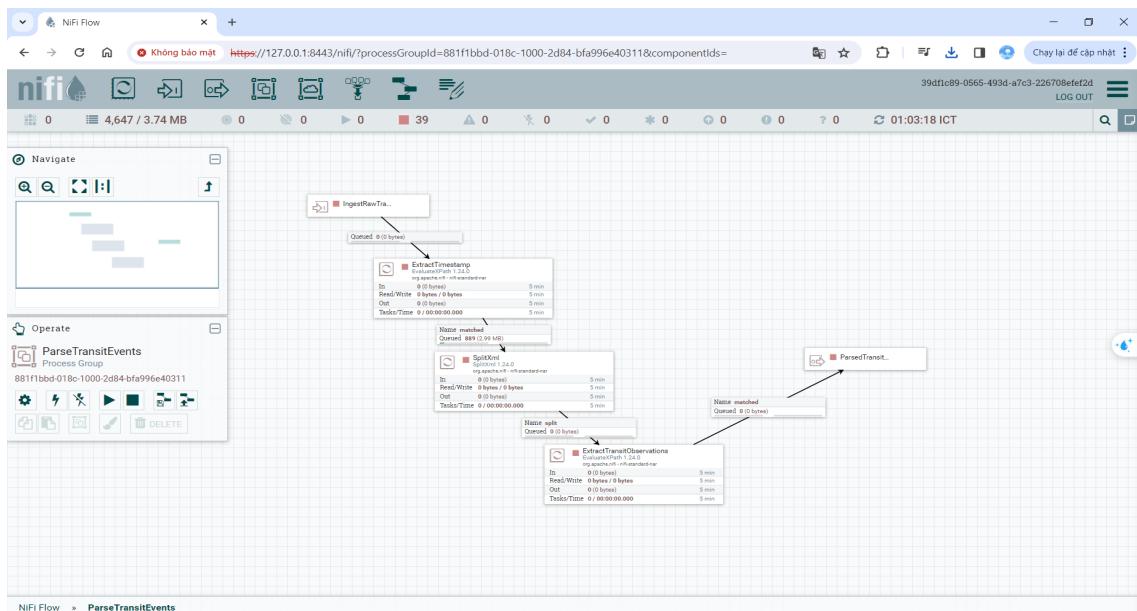
Hình 2.15: Hình ảnh chỉnh sửa Processor ExtractTransitObservations.

- Bước 6, thêm Output Port, đặt tên là ParsedTransitEvents. Sau đó, tạo mối quan hệ giữa ExtractTransitObservations với ParsedTransitEvents.
- Cuối cùng, thực hiện kết nối giữa Process Group SimulateXmlTransitEvents với ParseTransitEvents (Hình 2.16).



Hình 2.16: Hình ảnh mối quan hệ giữa SimulateXmlTransitEvents và ParseTransitEvents.

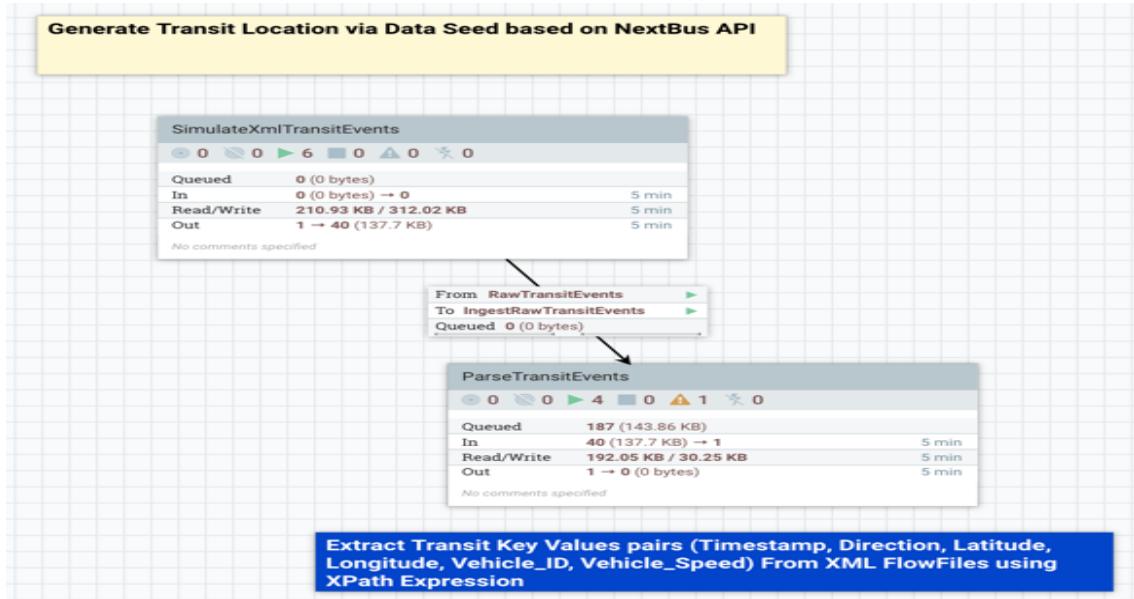
Như vậy, sau khi hoàn tất các bước, ta được NiFi DataFlow như hình 2.17. Tóm tắt lại, ta vừa xây dựng một nhóm quy trình NiFi ParseTransitEvents để phân tích nội dung XML và trích xuất các quan sát chuyển tiếp thành các thuộc tính FlowFile. Input Port lấy dữ liệu từ SimulateXmlTransitEvents, đi vào bộ xử lý ExtractTimestamp để lấy dấu thời gian cho hoạt động quan sát phương tiện và thêm dấu thời gian đó làm thuộc tính FlowFile. Các hàng nội dung FlowFile sau đó được chia thành nhiều FlowFiles thông qua bộ xử lý SplitXml. Các bản ghi FlowFile đơn lẻ này được định tuyến đến một bộ xử lý ExtractTransitObservations khác để trích xuất các quan sát vận chuyển cho nhiều phương tiện vận chuyển có dấu thời gian đó từ trước đó. Dữ liệu này với các thuộc tính FlowFile mới được định tuyến đến phần còn lại của luồng thông qua Output Port.



Hình 2.17: Hình ảnh Data Nifi Flow.

Sau khi đã hoàn thành, ta cần xác minh ParseTransitEvents trích xuất các giá trị từ SimulateXmlTransitEvents bằng các bước sau:

- Đầu tiên, Giữ phím Shift, chọn vào SimulateXmlTransitEvents và ParseTransitEvents, sau đó chọn Start như hình 2.18.



Hình 2.18: Hình ảnh chạy SimulateXmlTransitEvents và ParseTransitEvents.

- Tiếp theo, vào ParseTransitEvents, chọn ExtractTransitObservations, sau đó chọn View Data Provenance. Ta sẽ quan sát các giá trị được ánh xạ tới tên thuộc tính của chúng như hình 2.19.

Hình 2.19: Hình ảnh các giá trị được ánh xạ tới tên thuộc tính của chúng.

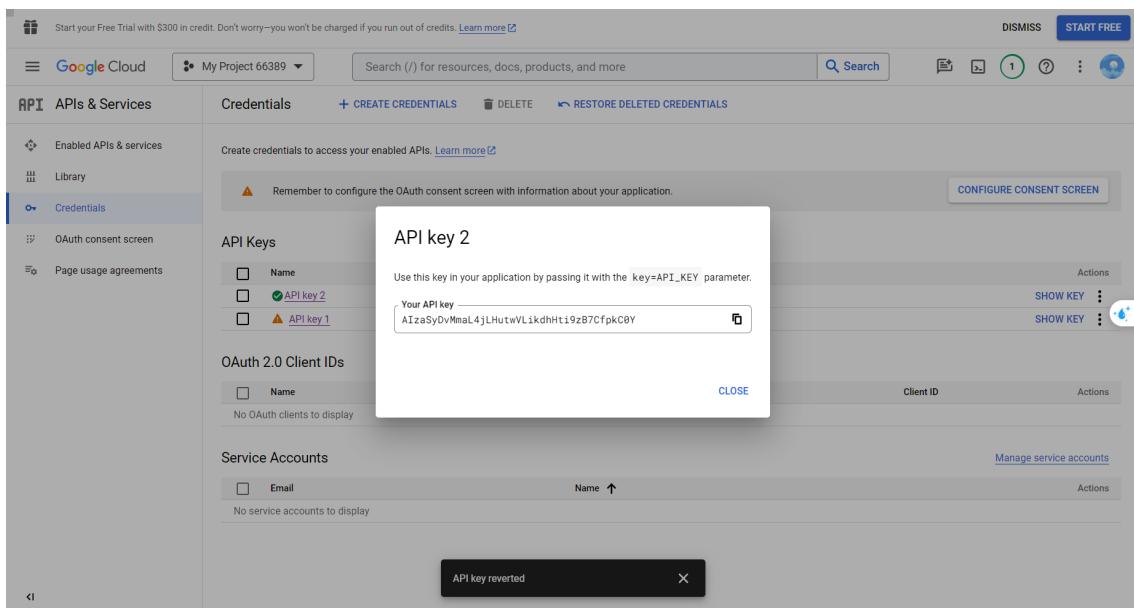
2.4 Xây dựng NiFi Process Group để xác thực dữ liệu GeoEnriched Data

Với dữ liệu chuyển tuyến được lấy từ trình mô phỏng API NextBus, dữ liệu này hiển thị vị trí ở dạng vĩ độ và kinh độ, nhưng không biểu thị những thông tin chi tiết có ý nghĩa hơn như các vùng lân cận mà các tuyến đường chuyển tuyến đi qua. Ta sẽ thêm khả năng đó vào luồng NiFi bằng cách tích hợp Google Places API.

- Thực hiện tạo một Process Group, đặt tên là ValidateGeoEnrichedTransitData.
- Bước 1, tiến hành lấy khóa API cho Bộ xử lý InvokeHTTP của NiFi theo hướng dẫn của đường liên kết:

Link: <https://developers.google.com/maps/documentation/places/web-service/get-api-key#console>

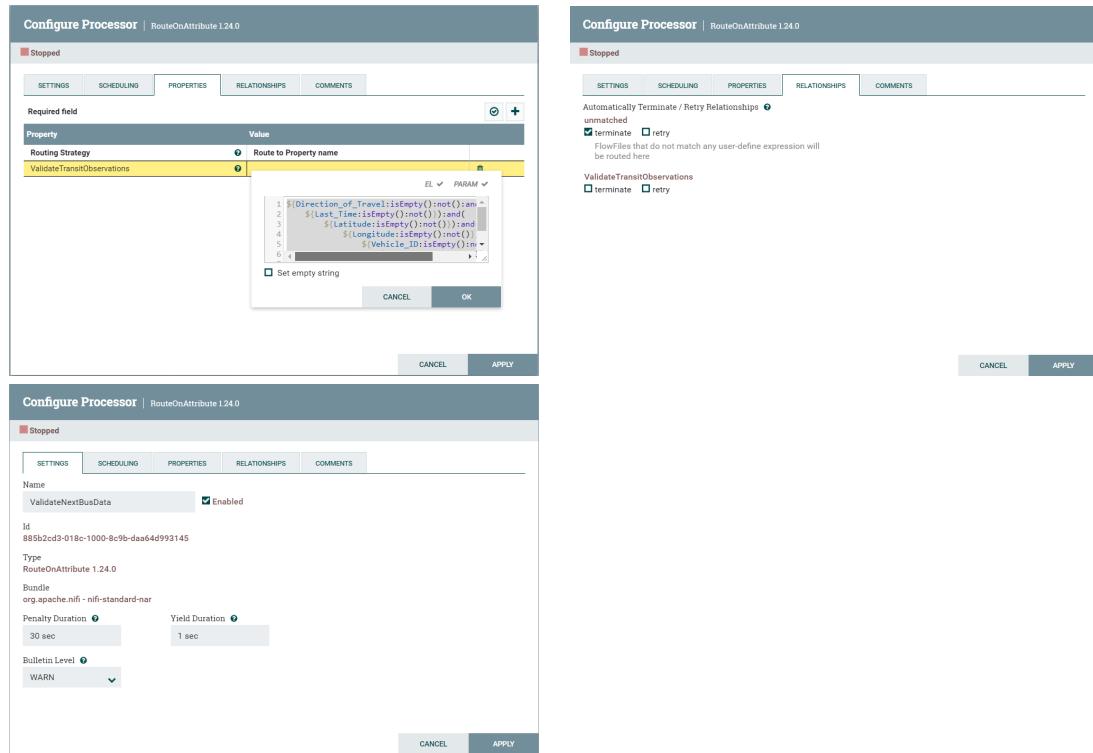
Sau khi hoàn thành sẽ có được API Key như hình 2.20



Hình 2.20: Hình ảnh API Key lấy từ Google Places API.

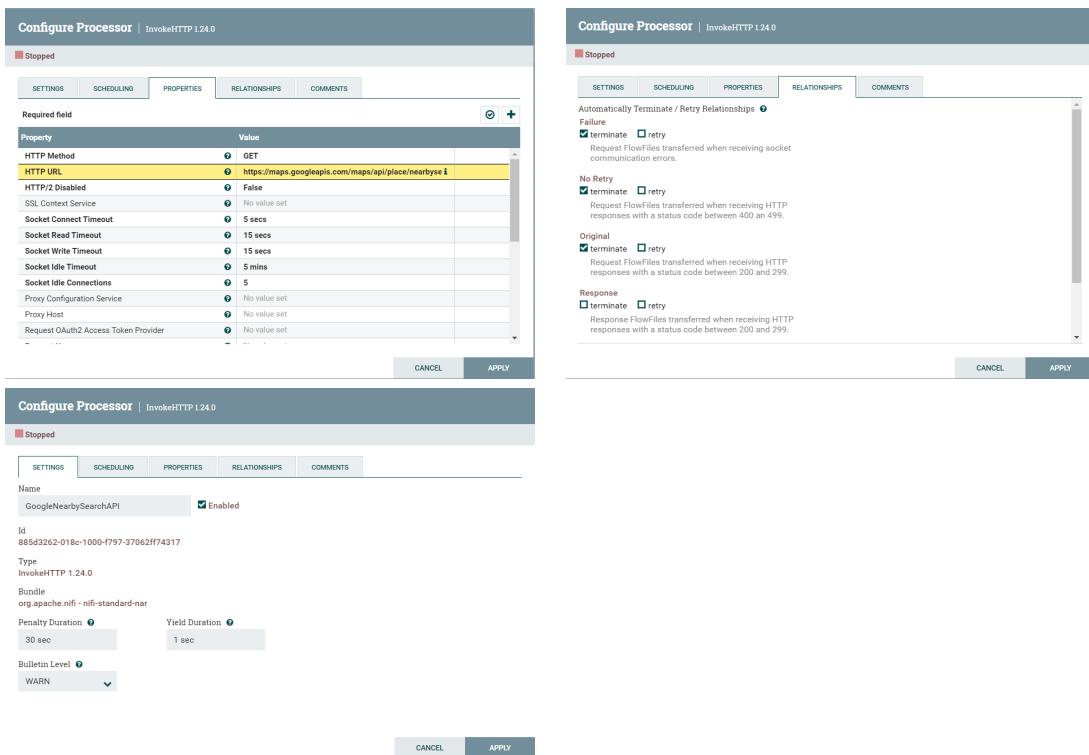
- Bước 2, vào trong ValidateGeoEnrichedTransitData, tiến hành thêm một Input Port, đặt tên IngestParsedTransitEvents. Input Port sẽ nhận dữ liệu từ Output Port của process group trước.
- Bước 3, thêm processor RouteOnAttribute, đổi tên thành ValidateNextBusData. Processor RouteOnAttribute có nhiệm vụ kiểm tra dữ liệu Trình mô phỏng NextBus bằng cách chỉ định tuyến FlowFiles nếu thuộc tính của chúng chứa

dữ liệu quan sát chuyến tuyến (Direction of Travel, Last Time, Latitude, Longitude, Vehicle ID, Vehicle Speed). Sau đó, tiến hành tạo mối quan hệ (connection) giữa Input Port và processor RouteOnAttribute. Sau đó, ta chỉnh sửa processor RouteOnAttribute theo hình 2.21. (Lưu ý, Filter Attributes sử dụng các giá trị khóa Thuộc tính FlowFile thu được từ Biểu thức XPath để lọc bất kỳ FlowFiles nào có ít nhất một giá trị Thuộc tính trống hoặc giá trị thuộc tính tốc độ bằng 0. Ngược lại, FlowFiles được chuyển cho các bộ xử lý còn lại.)



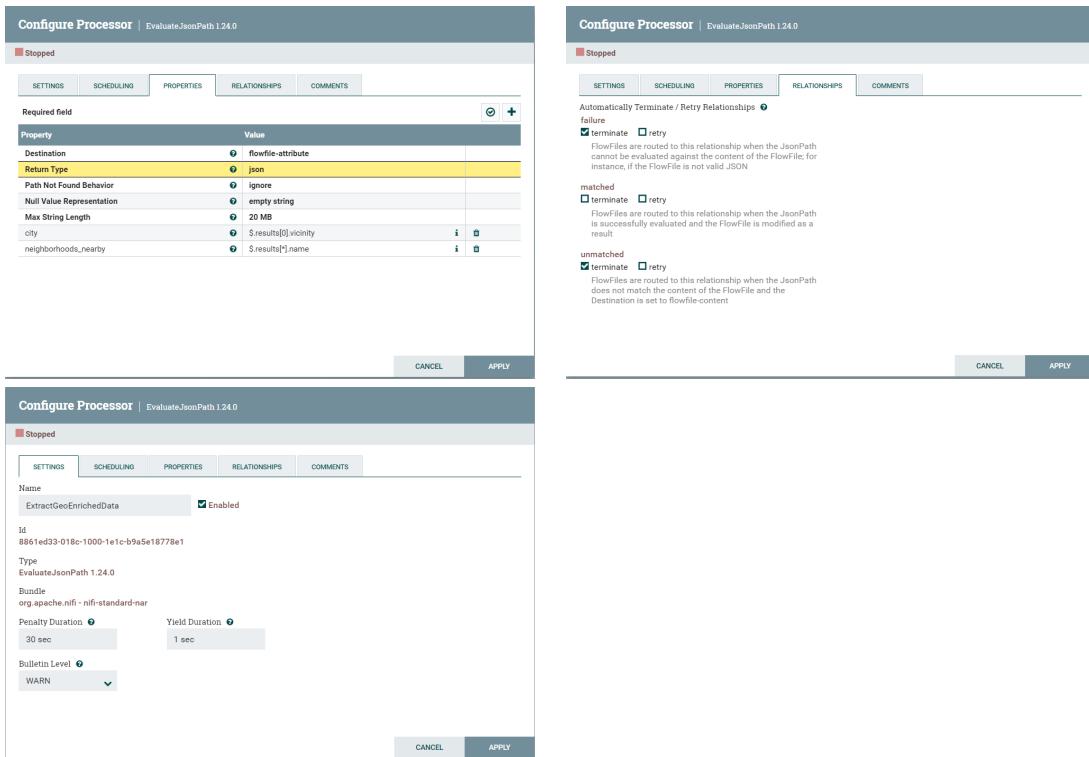
Hình 2.21: Hình ảnh chỉnh sửa Processor RouteOnAttribute.

- Bước 4, thêm processor InvokeHTTP. Đổi tên processor này thành GoogleNearbySearchAPI. Processor InvokeHTTP gửi lệnh gọi nghỉ tới Google Places API để lấy dữ liệu địa lý phong phú cho vị trí chuyến tuyến. Sau đó, tiến hành tạo mối quan hệ giữa RouteOnAttribute và InvokeHTTP. Tiếp theo, chỉnh sửa processor InvokeHTTP như hình 2.22. (Lưu ý, Remote URL có nhiệm vụ kết nối với URL HTTP mà chúng tôi đã tạo bằng Google Places API và cung cấp dữ liệu đó vào luồng dữ liệu. Lưu ý rằng ta sử dụng hai biểu thức NiFi cho tham số vị trí. Điều này là do hai giá trị đó thay đổi khi FlowFiles mới đi qua bộ xử lý này.)



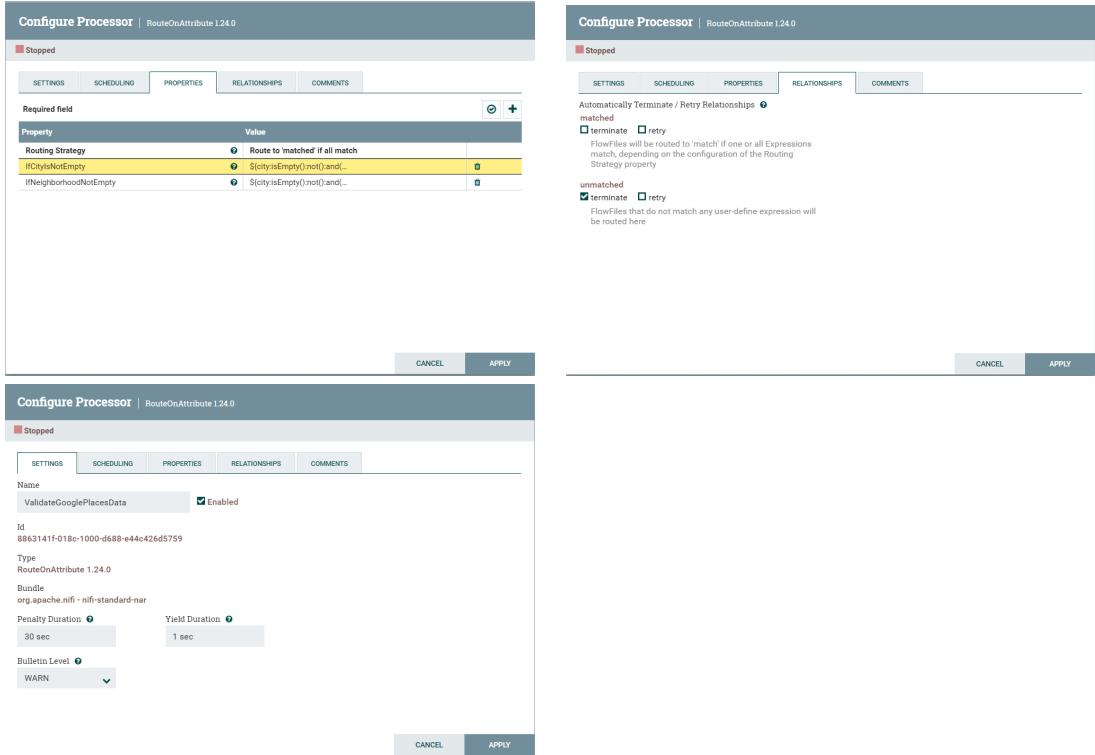
Hình 2.22: Hình ảnh chỉnh sửa Processor InvokeHTTP.

- Bước 5, thêm processor EvaluateJsonPath. Sau đó, tạo mối quan hệ giữa InvokeHTTP và EvaluateJsonPath. Sau đó, chỉnh sửa processor EvaluateJsonPath như hình 2.23. Đổi tên processor này thành ExtractGeoEnrichedData. (Lưu ý, Destination là kết quả từ Đánh giá đường dẫn JSON được lưu trữ trong thuộc tính FlowFile.; hai thuộc tính mỗi cái giữ một giá trị được sử dụng trong điều kiện lọc ngôn ngữ Biểu thức NiFi trong bộ xử lý tiếp theo.)



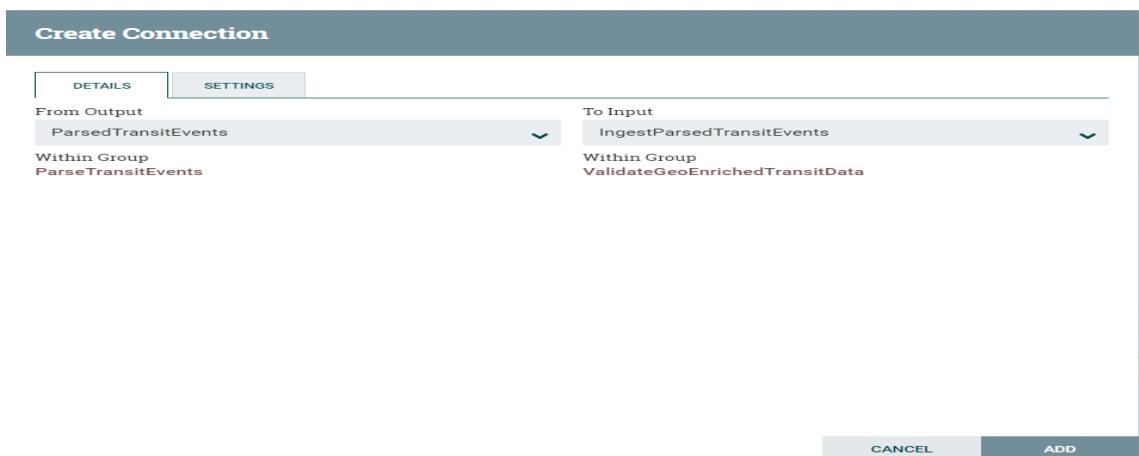
Hình 2.23: Hình ảnh chỉnh sửa Processor EvaluateJsonPath.

- Bước 6, thêm processor RouteOnAttribute. Sau đó, tạo mối quan hệ giữa ExtractGeoEnrichedData và RouteOnAttribute. Sau đó, chỉnh sửa processor RouteOnAttribute như hình 2.24. Đổi tên processor này thành ValidateGooglePlacesData. (Lưu ý, ValidateGooglePlacesData sử dụng các giá trị Thuộc tính FlowFile thu được từ Biểu thức đường dẫn JSON để lọc ra bất kỳ FlowFiles nào có ít nhất một giá trị Thuộc tính trống. Mặt khác, FlowFiles sẽ được chuyển đến các bộ xử lý còn lại.)



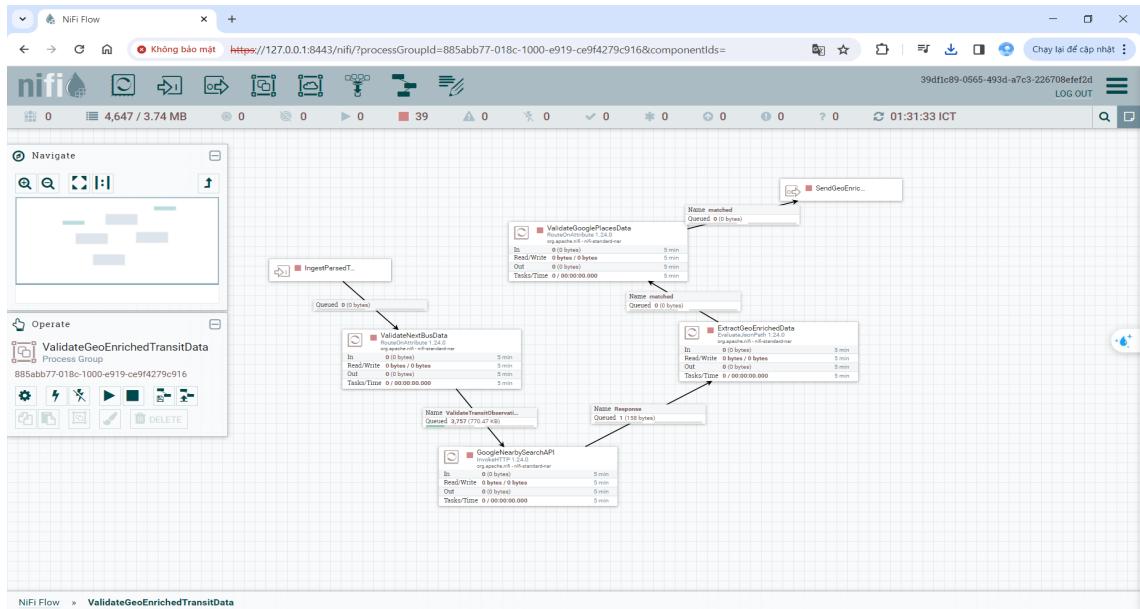
Hình 2.24: Hình ảnh chỉnh sửa Processor RouteOnAttribute.

- Bước 7, thêm Output Port, đặt tên là SendGeoEnrichedTransitEvents. Sau đó, tạo mối quan hệ giữa ValidateGooglePlacesData với SendGeoEnrichedTransitEvents.
- Cuối cùng, thực hiện kết nối giữa Process Group ParseTransitEvents với ValidateGeoEnrichedTransitData (Hình 2.25).



Hình 2.25: Hình ảnh mối quan hệ giữa ParseTransitEvents và ValidateGeoEnrichedTransitData.

Như vậy, sau khi hoàn tất các bước, ta được NiFi DataFlow như hình 2.26. Tóm tắt lại, ta đã học cách sử dụng InvokeHTTP để truy cập vị trí địa lý của các địa điểm lân cận bằng API tìm kiếm của Google Places. Ta đã học cách thêm các biến biểu thức NiFi vào thuộc tính RemoteURL của InvokeHTTP để các giá trị cho vĩ độ và kinh độ liên tục thay đổi trong URL khi FlowFiles mới đi qua bộ xử lý này. Ta đã học cách sử dụng EvaluJsonPath tương tự như EvaluXPath, ngoại trừ Biểu thức JSON được sử dụng để trích xuất các phần tử JSON (neighborhoods nearby và city) từ cấu trúc JSON. Bây giờ bạn đã biết cách kết hợp API bên ngoài vào NiFi để nâng cao hơn nữa luồng dữ liệu.



Hình 2.26: Hình ảnh Data Nifi Flow.

Sau khi đã hoàn thành, ta cần xác minh Xác minh dữ liệu GeoEnriched được định tuyến bởi ValidateGooglePlacesData là hợp lệ. Bên trong ValidateGeoEnriched-TransitData, ta sẽ kiểm tra bộ xử lý ValidateGooglePlacesData, ta sẽ xem xét các luồng đi qua bộ xử lý này và phân tích các thuộc tính của chúng. Cần phải đảm bảo rằng thuộc tính key city và Neighbors nearby của mỗi flowfile không có giá trị trống tương ứng. Thuộc tính FlowFile phải có định dạng khóa/giá trị (Hình 2.27 và Hình 2.28).

RouteNearbyNeighborhoods							
Displaying 100 of 1,003 (3.09 MB)							
Position	UUID	Filename	File Size	Queued Duration	Lineage Duration	Penalized	Node
0	1	88cbfaa7-14d8-4561-b017-8d580e52bea4	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
1	2	70b33bde-7773-4945-b4f4-79168b73fbac	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
2	3	8401303d-989f-43f4-8381-c18ff0adafdc	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
3	4	2ef110f1-5606-45bb-a0f2-5c4d232c0b01	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
4	5	efc59308-a72e-4ecf-95fe-c7d556fb98a4	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
5	6	5f750dd4-2600-47d5-a321-f30fecc2a02f	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
6	7	74345a03-9468-4d84-82da-25321659873	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
7	8	0355303e-66d7-4574-a0a8-93570788c5b	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
8	9	d65b6101-e055-458e-9e9d-56469af22f4	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
9	10	6e917467-8795-4273-9902-31846e3d3074	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
10	11	2877eda5-f959-4021-9c4d-42c5ae07b20	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
11	12	a30f8858-f414-4e5f-93ca-35227e6fc0cc	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
12	13	a75b20ed-625c-4635-a1fe-dc18c09498a	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
13	14	9c2539c-0e38-4477-00ba-af0e54d77f	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
14	15	e3e7730b-af62-440b-af73-97ea100400e	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
15	16	1d7a9e4b-1ef1-432b-ba41-574b5055de8a1	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
16	17	76b3a3db-fe11-4990-a70b-2f05ab8d7319	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
17	18	5b627558-9094-47e6-9154-ae98933002a	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
18	19	8c700ec6-871f-4c3b-8a47-d10ac21d158	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
19	20	4e6b6608b-1e03-449b-8fb0-927d9b6556	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
20	21	c1a958c2-8429-4567-80e9-53050f1514	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
21	22	10a9f116-6203-4ee8-8a81-95b6b62bc2b2	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
22	23	f69539f7-082a-4b20-9059-27586452cd	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
23	24	ab024649-2cf4-4f15-e182-123800573686	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
24	25	5eb1f4a8e1e814-4147-941c-7c2af4744d	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:52.357		sandbox-hdf.hortonworks.com:9090
25	26	b0c49160-2729-42en-a1c0-c2981d144da	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:46.604		sandbox-hdf.hortonworks.com:9090
26	27	5a179440-1000-413e-8158-2b466d832299	transit-data-0c212571-6092-49d9-b645-4d32...	3.16 KB	00:44:47.346		sandbox-hdf.hortonworks.com:9090
27	28	45b89732-5094-4257-8745-8c30edf1785	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:47.126		sandbox-hdf.hortonworks.com:9090
28	29	82d2aa07-8501-40e5-88a6-1b167b601389	transit-data-0c212571-6092-49d9-b645-4d32...	3.15 KB	00:44:46.959		sandbox-hdf.hortonworks.com:9090

Last updated: 11:40:43 UTC

Hình 2.27: Hình ảnh Data Provenance cho thuộc tính FlowFile (khóa/giá trị): thành phố chứa San Francisco.

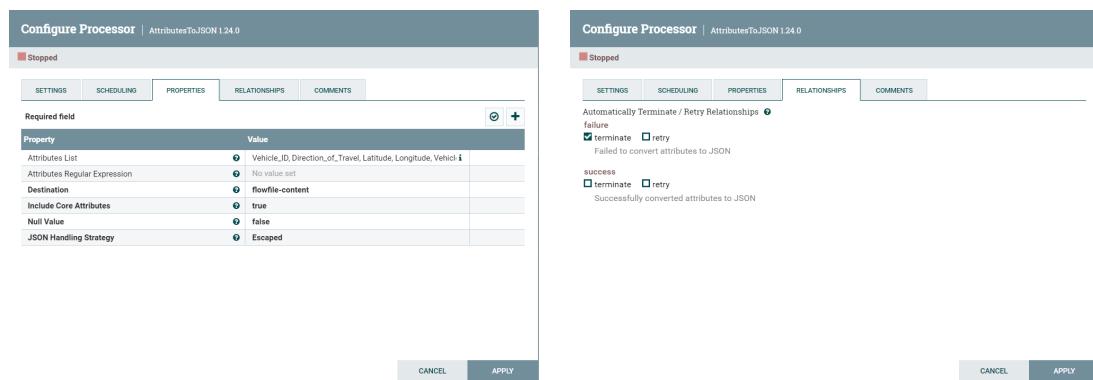
FlowFile	
DETAILS	ATTRIBUTES
Attribute Values	
<pre> invokehttp.tx.id 8cf597fb-5ce1-4eef-ab1b-4fa0c5896d2f mime.type application/json; charset=UTF-8 neighborhoo..._nearby ["Saint Francis Wood","West Portal"] path / segment.original.filename transit-data-0c212571-6092-49d9-b645-4d32d094712a.xml uuid 88cbfaa7-14d8-4561-b017-8d580e52bea4 </pre>	

Hình 2.28: Hình ảnh Data Provenance cho thuộc tính FlowFile (khóa/giá trị): Neighbors nearby chứa ["Saint Francis Wood","West Portal"]

2.5 Xây dựng NiFi Process Group để lưu trữ dữ liệu dưới dạng JSON

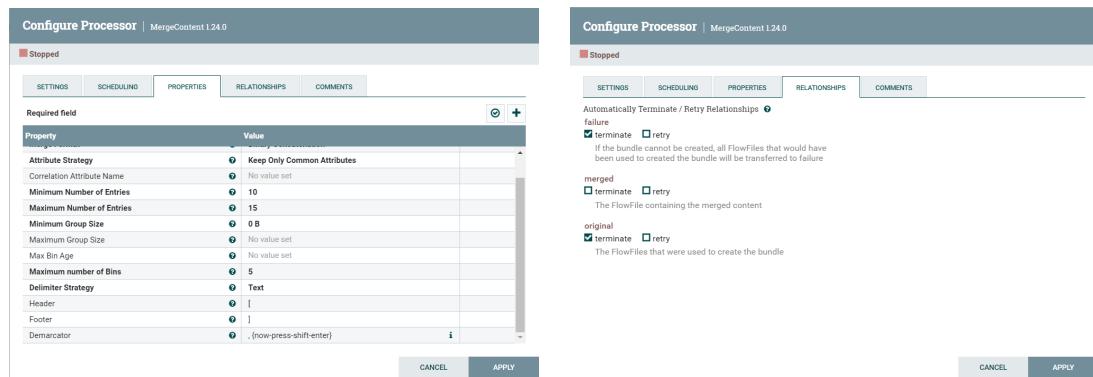
Ở phần này, ta sẽ xây dựng một nhóm quy trình mới có tên StoreTransitEventsAsJSONToDisk để lấy các thuộc tính FlowFile, ghi chúng vào nội dung của FlowFile mới dưới dạng biểu diễn JSON. Sau đó, bạn sẽ lưu trữ các FlowFiles này vào hệ thống tệp cục bộ.

- Thực hiện tạo một Process Group, đặt tên là StoreDataAsJSONToDisk.
- Bước 2, vào trong StoreDataAsJSONToDisk, tiến hành thêm một Input Port, đặt tên IngestGeoEnrichedEvents. Input Port sẽ nhận dữ liệu từ Output Port của process group trước.
- Bước 3, thêm processor AttributesToJson. Processor AttributesToJson có nhiệm vụ tạo ra một biểu diễn JSON của các thuộc tính được trích xuất từ FlowFiles và chuyển đổi XML sang định dạng JSON ít thuộc tính này hơn. Sau đó, tiến hành tạo mối quan hệ (connection) giữa Input Port và processor AttributesToJson. Sau đó, ta chỉnh sửa processor AttributesToJson theo hình 2.29. (Lưu ý, Attributes List lấy các tham số thuộc tính FlowFile và trình bày chúng ở định dạng JSON; Destination lưu trữ đầu ra dưới dạng nội dung trong FlowFile)



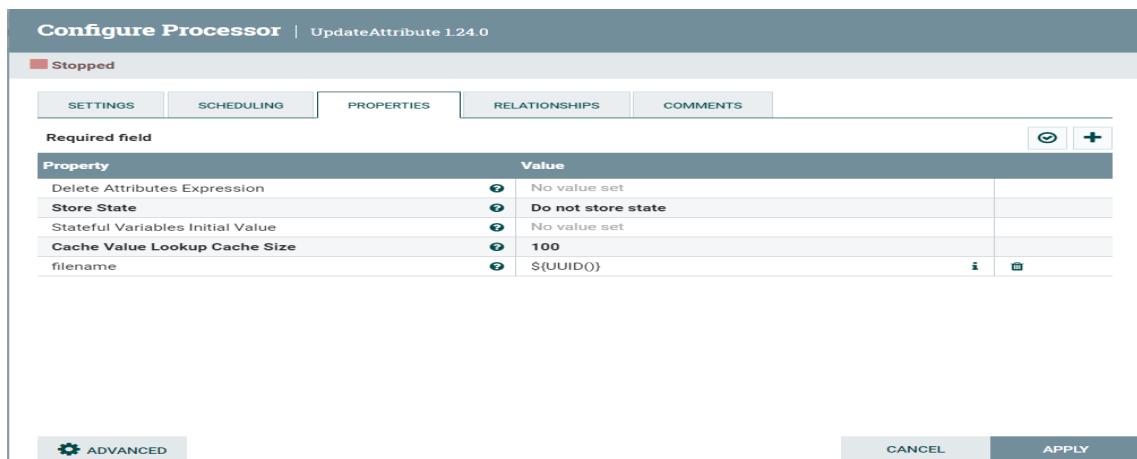
Hình 2.29: Hình ảnh chỉnh sửa Processor AttributesToJson.

- Bước 4, thêm processor MergeContent. Sau đó, tiến hành tạo mối quan hệ giữa AttributesToJson và MergeContent. Tiếp theo, chỉnh sửa processor MergeContent như hình 2.30. (Lưu ý, Minimum Number of Entries là lấy ít nhất số lượng FlowFiles được chỉ định, sau đó hợp nhất chúng thành 1 FlowFile; Maximum Number of Entries là nhận không quá số lượng FlowFiles được chỉ định, sau đó hợp nhất chúng; Delimiter Strategy có nhiệm vụ chỉ định rằng Header, Footer và Demarcator đặt các điều kiện định dạng cho văn bản trong tệp)



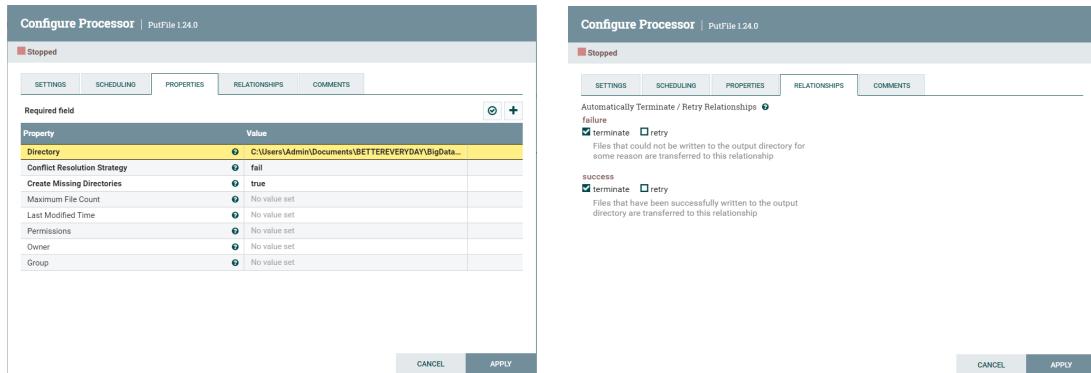
Hình 2.30: Hình ảnh chỉnh sửa Processor MergeContent.

- Bước 5, thêm processor UpdateAttribute. Processor UpdateAttribute có nhiệm vụ cập nhật tên thuộc tính cho mỗi FlowFile. Sau đó, tạo mối quan hệ giữa MergeContent và UpdateAttribute. Sau đó, chỉnh sửa processor UpdateAttribute như hình 2.31.

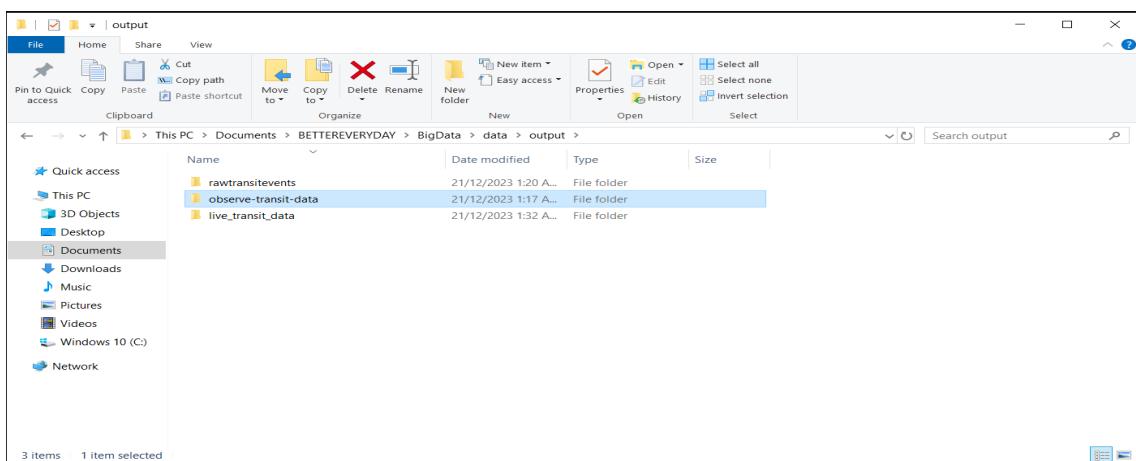


Hình 2.31: Hình ảnh chỉnh sửa Processor UpdateAttribute.

- Bước 6, thêm processor PutFile . Sau đó, tạo mối quan hệ giữa UpdateAttribute và PutFile. Sau đó, chỉnh sửa processor PutFile như hình 2.32. Sau khi hoàn tất, ta có hình 2.33 là hình ảnh thư mục đầu ra.

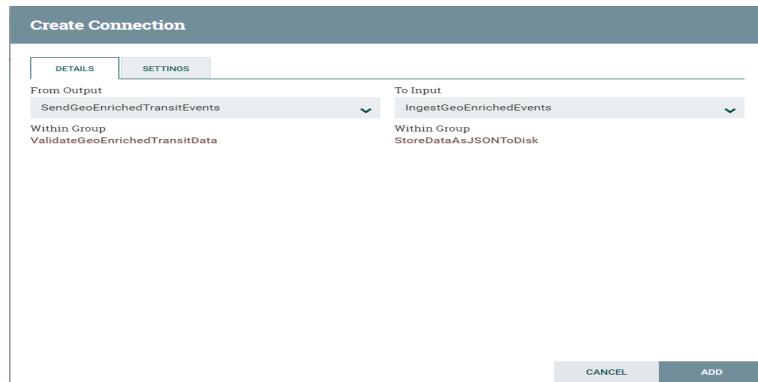


Hình 2.32: Hình ảnh chỉnh sửa Processor PutFile.



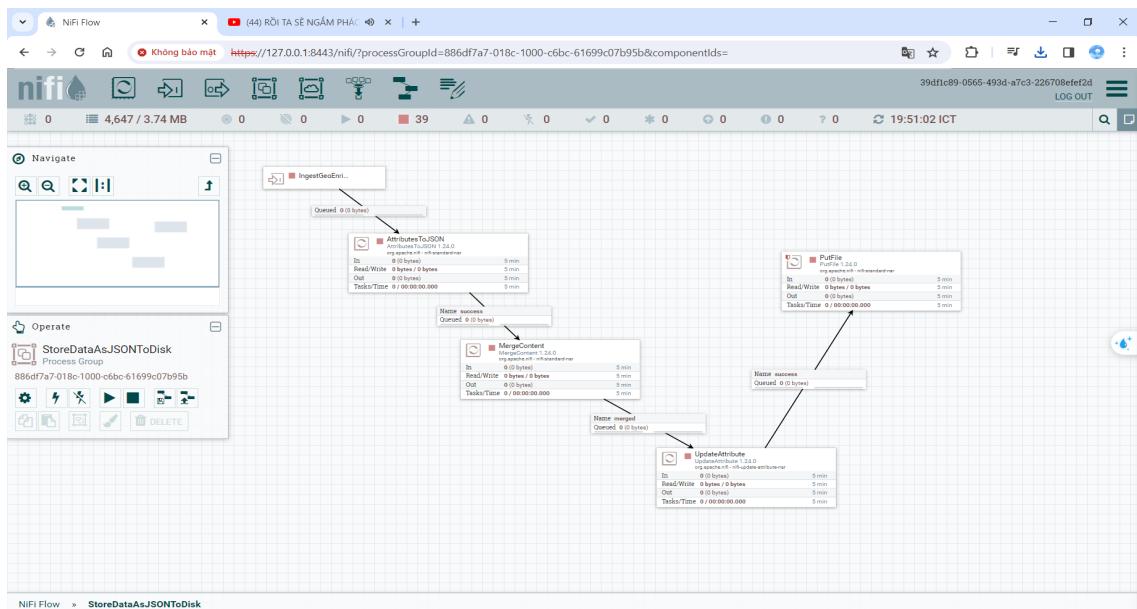
Hình 2.33: Hình ảnh thư mục đầu ra.

- Cuối cùng, thực hiện kết nối giữa Process Group ValidateGeoEnrichedTransitData với StoreDataAsJSONToDisk (Hình 2.34).



Hình 2.34: Hình ảnh mối quan hệ giữa ValidateGeoEnrichedTransitData và StoreDataAsJSONToDisk.

Như vậy, sau khi hoàn tất các bước, ta được NiFi DataFlow như hình 2.35. Tóm tắt lại, ta đã học cách lấy các cặp thuộc tính (khóa/giá trị) FlowFile và biểu diễn chúng dưới dạng định dạng JSON. Sau đó, bạn đã sử dụng MergeContent để kết hợp nhiều FlowFiles lại với nhau nhằm tạo ra một FlowFile lớn hơn có nhiều bản ghi. UpdateAttribution được sử dụng để đảm bảo không có FlowFiles nào có tên trùng lặp. PutFile đã được thêm vào luồng để lưu trữ dữ liệu JSON vào hệ thống tệp cục bộ.



Hình 2.35: Hình ảnh Data Nifi Flow.

Sau khi đã hoàn thành, ta kiểm tra dữ liệu NiFi's Data Provenance (Hình 2.36 và Hình 2.37). Cuối cùng, NiFi cung cấp cho người dùng tùy chọn xem dữ liệu ở nhiều định dạng, ta sẽ xem nó ở định dạng ban đầu như hình 2.38.

NiFi Data Provenance

Displaying 13 of 13
Oldest event available: 08/18/2016 06:13:29 PDT

Provenance Events

Showing the events that match the specified query. Clear search

Filter by component n... ▾

Date/Time	Type	FlowFile UUID	Size	Component Name	Component Type
08/18/2016 11:45:13.920 PDT	DROP	11c08e22-a3d4-b8e8-bcd13e9e7e044c	2.1 KB	PutFile	PutFile
08/18/2016 11:44:56.898 PDT	DROP	8fc1eadc-21d6-4bcd-95a3-5c5c85903497	2.25 KB	PutFile	PutFile
08/18/2016 11:44:39.873 PDT	DROP	98d6731d-6fc8-4190-bc29-edf4e44f1f14	2.25 KB	PutFile	PutFile
08/18/2016 11:44:27.849 PDT	DROP	1497b775-11a7-43a7-906a-4c7789954f16	2.09 KB	PutFile	PutFile
08/18/2016 11:44:10.832 PDT	DROP	6df1587b-c25d-4abc-b5c6-068f51bd9af	2.09 KB	PutFile	PutFile
08/18/2016 11:43:53.811 PDT	DROP	b1533d6-91ef-4aa4-be94-da25460601f	2.09 KB	PutFile	PutFile
08/18/2016 11:43:31.782 PDT	DROP	c02d30eb-6149-432d-8975-667324ba8ee	2.09 KB	PutFile	PutFile
08/18/2016 11:43:14.757 PDT	DROP	8f582865-9829-4469-bc21-71f87601080e	2.1 KB	PutFile	PutFile
08/18/2016 11:42:58.740 PDT	DROP	89f92492-29c4-4736-8385-d2f5c7ab812	2.1 KB	PutFile	PutFile
08/18/2016 11:42:41.715 PDT	DROP	66953c31-1389-4774-b5c6-9f548105e0c	2.1 KB	PutFile	PutFile
08/18/2016 11:42:24.698 PDT	DROP	8412e9b7-71a6-479a-ba51-daz2b2271057	2.1 KB	PutFile	PutFile
08/18/2016 11:42:07.675 PDT	DROP	27387031-7756-4848-b80b-2be0feab99c	2.25 KB	PutFile	PutFile
08/18/2016 11:41:50.652 PDT	DROP	87179c6-b524-a0a8-850c-6ee0cb88e6c7	2.25 KB	PutFile	PutFile

View Event

Hình 2.36: Hình ảnh NiFi Data Provenance Window.

Provenance Event

DETAILS **ATTRIBUTES** **CONTENT**

Input Claim Container default Section 52 Identifier 1473856810600-2100 Offset 0 Size 2.1 KB	Output Claim Container default Section 52 Identifier 1473856810600-2100 Offset 0 Size 2.1 KB
Download	View
Replay Connection Id 229ee29f-6023-4092-89b0-69005bba8d71	Download

OK

Hình 2.37: Hình ảnh Provenance Event Window.

Filename: traffic-location-data-020931875.xml
Content Type: application/json

```

1  [{"Last_Time": "1465504739812", "Vehicle_Speed": "", "Latitude": "37.71428", "Vehicle_ID": "1466", "Longitude": "-122.46386", "Direction_of_Travel": "N_0_FOO"},  
2  {"Last_Time": "1465504739812", "Vehicle_Speed": "1", "Latitude": "37.74028", "Vehicle_ID": "1497", "Longitude": "-122.46386", "Direction_of_Travel": "N_I_FOO"},  
3  {"Last_Time": "1465504739812", "Vehicle_Speed": "25", "Latitude": "37.73833", "Vehicle_ID": "1501", "Longitude": "-122.46888", "Direction_of_Travel": "N_I_FOO"},  
4  {"Last_Time": "1465504739812", "Vehicle_Speed": "31", "Latitude": "37.7388", "Vehicle_ID": "1454", "Longitude": "-122.46794", "Direction_of_Travel": "N_I_FOO"},  
5  {"Last_Time": "1465504739812", "Vehicle_Speed": "9", "Latitude": "37.71416", "Vehicle_ID": "1448", "Longitude": "-122.46269", "Direction_of_Travel": "N_I_FOO"},  
6  {"Last_Time": "1465504739812", "Vehicle_Speed": "33", "Latitude": "37.71402", "Vehicle_ID": "1440", "Longitude": "-122.46375", "Direction_of_Travel": "N_I_FOO"},  
7  {"Last_Time": "1465504739812", "Vehicle_Speed": "16", "Latitude": "37.71428", "Vehicle_ID": "1464", "Longitude": "-122.46311", "Direction_of_Travel": "N_0_FOO"},  
8  {"Last_Time": "1465504743839", "Vehicle_Speed": "31", "Latitude": "37.7389", "Vehicle_ID": "1454", "Longitude": "-122.46796", "Direction_of_Travel": "N_I_FOO"},  
9  {"Last_Time": "1465504743839", "Vehicle_Speed": "33", "Latitude": "37.71402", "Vehicle_ID": "1440", "Longitude": "-122.46375", "Direction_of_Travel": "N_I_FOO"},  
10 {"Last_Time": "1465504743839", "Vehicle_Speed": "12", "Latitude": "37.71428", "Vehicle_ID": "1501", "Longitude": "-122.46888", "Direction_of_Travel": "N_I_FOO"},  
11 {"Last_Time": "1465504743839", "Vehicle_Speed": "1", "Latitude": "37.74028", "Vehicle_ID": "1497", "Longitude": "-122.46386", "Direction_of_Travel": "N_I_FOO"},  
12 {"Last_Time": "1465504743839", "Vehicle_Speed": "18", "Latitude": "37.71428", "Vehicle_ID": "1464", "Longitude": "-122.46311", "Direction_of_Travel": "N_0_FOO"},  
13 {"Last_Time": "1465504743839", "Vehicle_Speed": "7", "Latitude": "37.71428", "Vehicle_ID": "1466", "Longitude": "-122.46386", "Direction_of_Travel": "N_0_FOO"},  
14 {"Last_Time": "1465504743839", "Vehicle_Speed": "18", "Latitude": "37.7143", "Vehicle_ID": "1448", "Longitude": "-122.46485", "Direction_of_Travel": "N_I_FOO"}]

```

Hình 2.38: Hình ảnh View FlowFile JSON Content.

2.6 Tích hợp API NextBus để lấy nguồn cấp dữ liệu trực tiếp về phương tiện công cộng

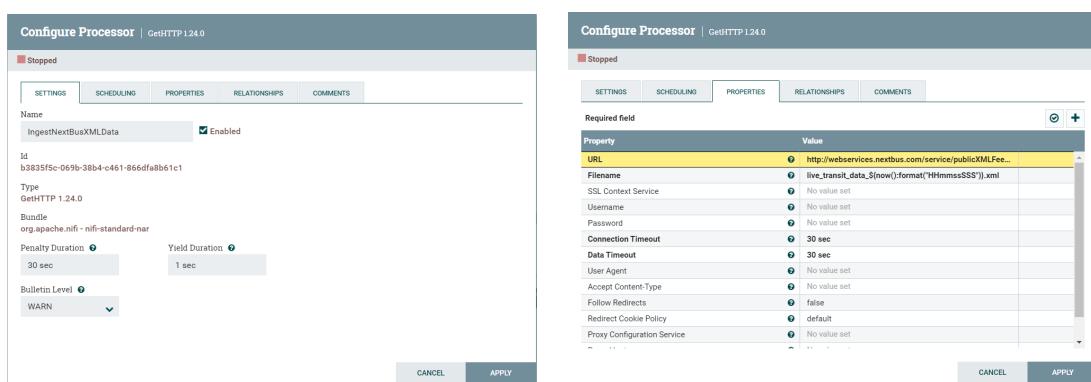
Ở phần này, ta sẽ học cách thực hiện các lệnh gọi nghỉ dựa trên API NextBus để truy xuất dữ liệu chuyển tuyến. Ta sẽ thay thế dữ liệu của Nhóm quy trình SimulateXmlTransitEvents bằng một bộ xử lý mới lấy dữ liệu phát trực tiếp từ Cơ quan San Francisco Muni trên tuyến OceanView vào NiFi DataFlow.

- NextBus Live Feed cung cấp cho công chúng thông tin trực tiếp về thông tin hành khách, chẳng hạn như thông tin vị trí xe, thời gian dự đoán về các phương tiện chuyển tuyến, tuyến đường của các phương tiện và các cơ quan khác nhau (San Francisco Muni, Unitrans City of Davis, v.v.). Tiến hành tạo URL cho processor GetHTTP ở bước 2:

Link: [http://webservices.nextbus.com/service/publicXMLFeed?
command=vehicleLocations&a=sf-muni&r=M&t=0](http://webservices.nextbus.com/service/publicXMLFeed?command=vehicleLocations&a=sf-muni&r=M&t=0)

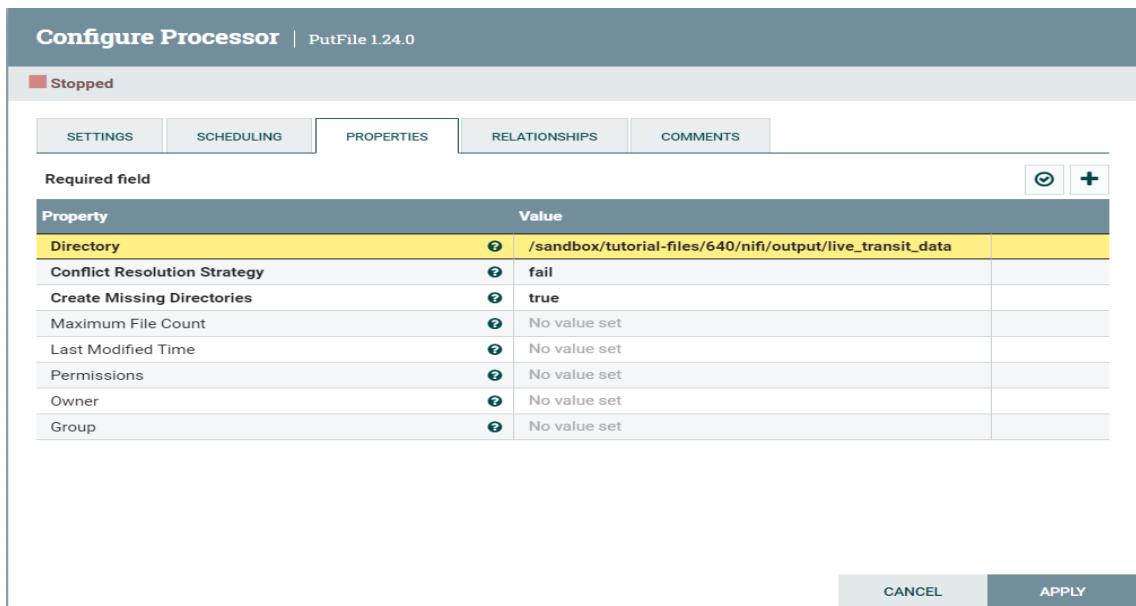
Trong đó: commands: command = vehicleLocations; agency: a=sf-muni; route: r=M; time: t=0.

- Bước 2, ta sẽ thay thế SimulateXmlTransitEvents với processor GetHTTP. Processor GetHTTP có nhiệm vụ nhập dữ liệu quan sát chuyển tuyến theo thời gian thực từ API NextBus. Sau đó, chỉnh sửa processor GetHTTP như hình 2.39. Đổi tên processor GetHTTP thành IngestNextBusXMLData.

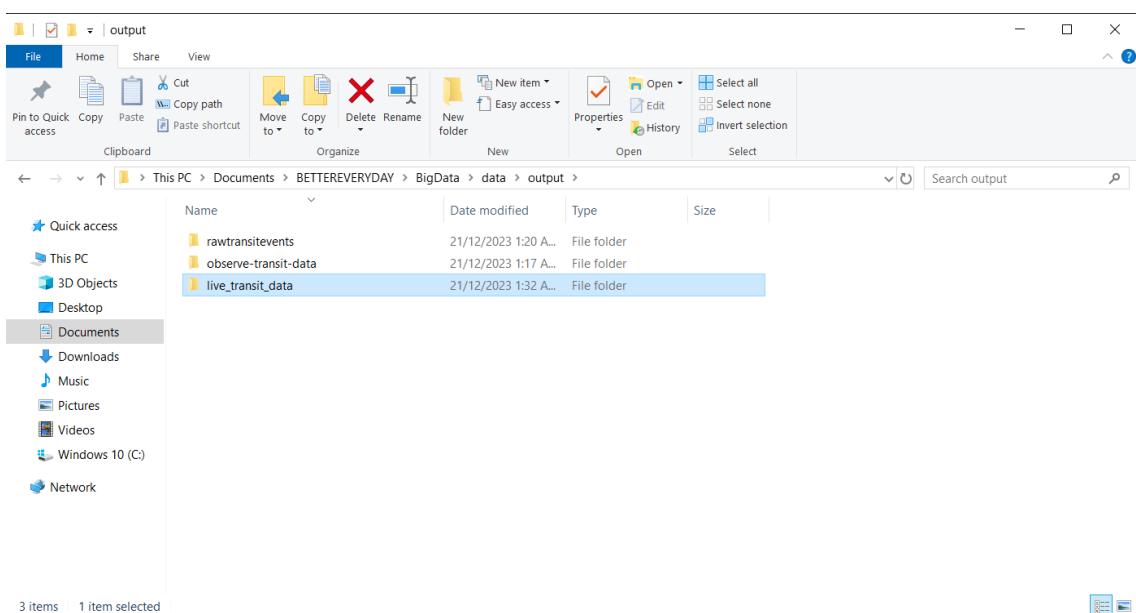


Hình 2.39: Hình ảnh chỉnh sửa Processor GetHTTP.

- Bước 3, sửa đổi PutFile trong StoreDataAsJSONToDisk như hình 2.40. Sau khi hoàn tất, thư mục đầu ra như hình 2.41.

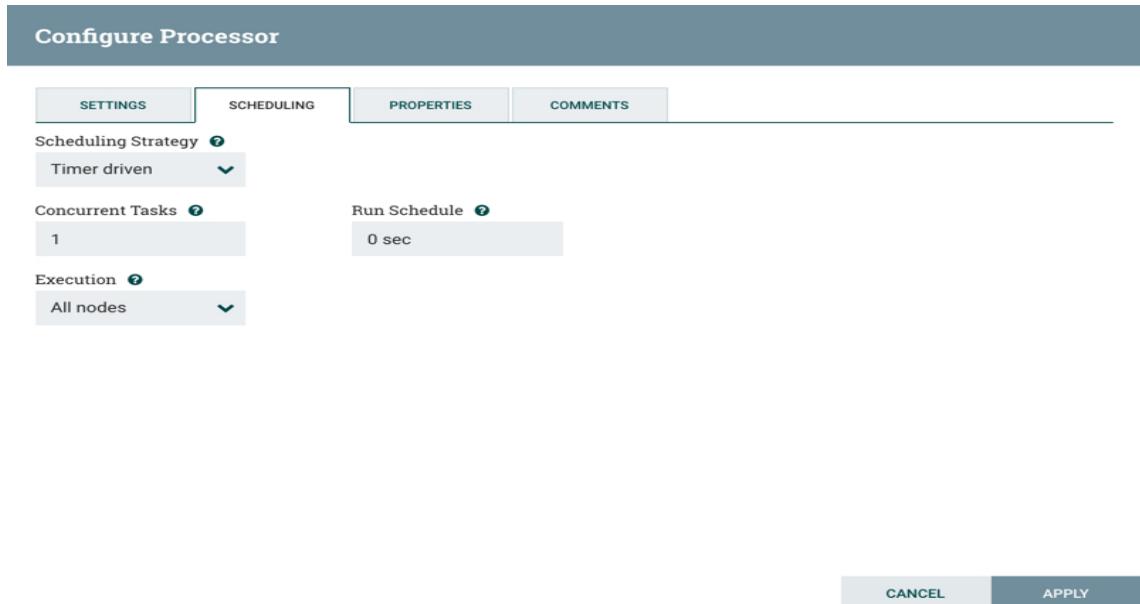


Hình 2.40: Hình ảnh chỉnh sửa PutFile.

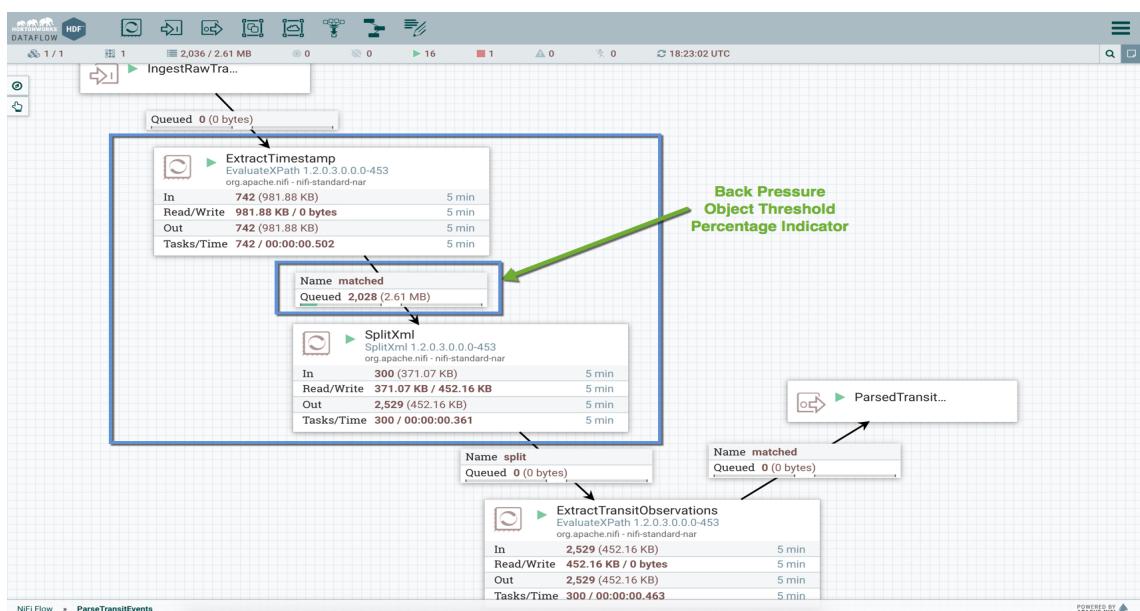


Hình 2.41: Hình ảnh thư mục đầu ra.

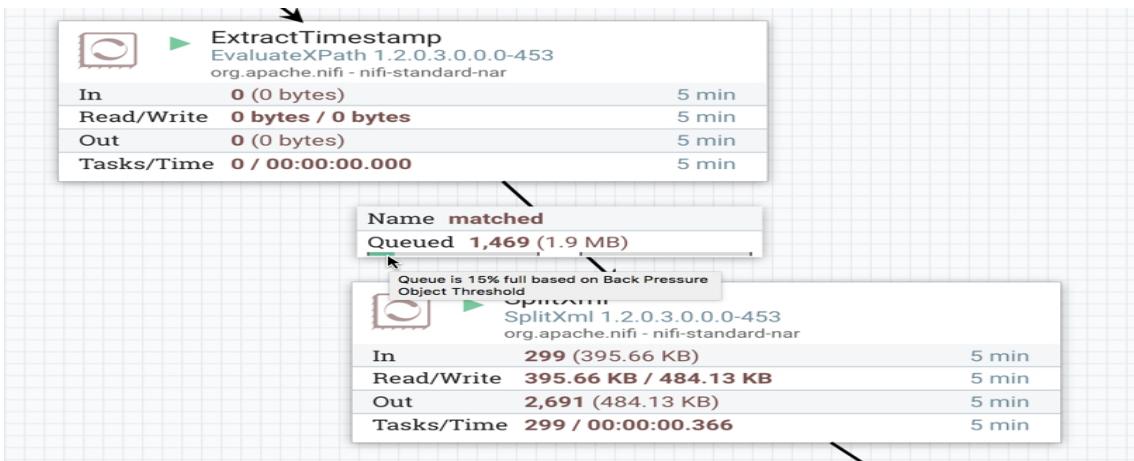
- Bước 4, ta cần biết được giá trị Back Pressure, giá trị cho phép chúng ta chỉ định lượng dữ liệu được phép tồn tại trong hàng đợi trước khi thành phần là nguồn kết nối không còn được lên lịch để chạy. Tiến hành chỉnh sửa IngestNextBusXMLData như hình 2.42. Sau đó, vào ParseTransitEvents, ta sẽ thấy hàng đợi đang trở nên rất lớn như hình 2.43. Theo mặc định, hàng đợi có thể chứa 10.000 đối tượng hoặc FlowFiles như hình 2.44. Như vậy, tùy thuộc vào ứng dụng của bạn, bạn có thể cần sửa đổi các giá trị này để có nhiều hay ít dữ liệu được điền vào hàng đợi.



Hình 2.42: Hình ảnh chỉnh sửa IngestNextBusXMLData.

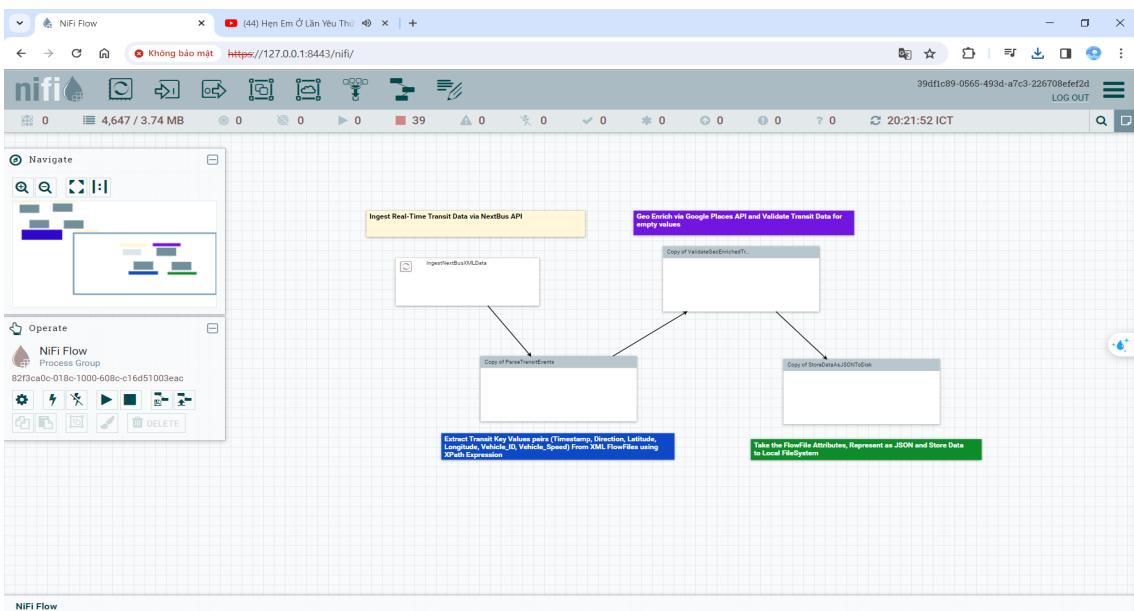


Hình 2.43: Hình ảnh số lượng FlowFiles trong hàng chờ.



Hình 2.44: Hình ảnh giá trị mặc định số lượng đối tượng hàng đợi có thể chứa.

Như vậy, sau khi hoàn tất các bước, ta được NiFi DataFlow như hình 2.45. Tóm tắt lại, ta đã học cách sử dụng API của NextBus để kết nối với Nguồn cấp dữ liệu trực tiếp XML của họ để lấy dữ liệu vị trí xe. Ngoài ra, ta cũng đã học cách sử dụng bộ xử lý GetHTTP để nhập luồng trực tiếp từ NextBus San Francisco Muni vào NiFi.



Hình 2.45: Hình ảnh NiFi DataFlow.

Chương 3

THẢO LUẬN VÀ NHẬN XÉT

3.1 Tổng quan

Như vậy, ta đã hoàn thiện việc xây dựng, hãy xem lại tổng quan NiFi DataFlow ta có những gì (Hình 2.45):

Đầu tiên, GetHTTP nhập dữ liệu quan sát chuyển tuyến theo thời gian thực từ API NextBus.

Thứ hai, ParseTransitEvents:

- Input Port nhập dữ liệu từ IngestNextBusXMLData.
- ExtractTimestamp trích xuất dấu thời gian của lần cập nhật cuối cùng cho dữ liệu vị trí phương tiện được trả về từ mỗi FlowFile.
- SplitXML chia các phần tử con của phần tử cha thành các FlowFiles riêng biệt. Vì phương tiện là phần tử con trong tệp xml của chúng tôi nên mỗi phần tử phương tiện mới được lưu trữ riêng biệt.
- ExtractTransitObservations trích xuất các thuộc tính: vehicle id, direction, latitude, longitude và speed từ FlowFile.
- Output Port xuất dữ liệu với thuộc tính FlowFile (khóa/giá trị) mới cho phần còn lại của luồng.

Thứ ba, ValidateGooglePlacesData:

- Input Port nhập dữ liệu từ ParseTransitEvents.
- ValidateNextBusData kiểm tra dữ liệu Trình mô phỏng NextBus bằng cách chỉ

định tuyến FlowFiles nếu thuộc tính của chúng chứa dữ liệu quan sát chuyển tuyến (Direction of Travel, Last Time, Latitude, Longitude, Vehicle ID, Vehicle Speed).

- InvokeHTTP gửi lệnh gọi nghỉ tới Google Places API để lấy dữ liệu geo enriched data cho vị trí chuyển tuyến.
- EvaluateJSONPath phân tích nội dung luồng cho city and neighborhoods nearby
- ValidateGooglePlacesData kiểm tra dữ liệu Google Places mới bằng cách chỉ định tuyến FlowFiles nếu thuộc tính của chúng chứa dữ liệu geo enriched data (city, neighborhoods nearby)
- Output Port xuất dữ liệu với thuộc tính FlowFile (khóa/giá trị) mới cho phần còn lại của luồng.

Thứ tư, StoreTransitEventsAsJSONToDisk:

- Input Port nhập dữ liệu từ ValidateGooglePlacesData.
- AttributionToJSON tạo biểu diễn JSON của các thuộc tính được trích xuất từ FlowFiles và chuyển đổi định dạng XML sang định dạng JSON ít thuộc tính hơn.
- MergeContent hợp nhất một nhóm JSON FlowFiles lại với nhau dựa trên một số FlowFiles và đóng gói chúng thành một FlowFile duy nhất.
- UpdateAttribution cập nhật tên thuộc tính cho mỗi FlowFile.
- PutFile ghi nội dung của FlowFile vào một thư mục mong muốn trên hệ thống tệp cục bộ.

3.2 Giải quyết vấn đề

3.2.1 Làm sao để NiFi giải quyết dữ liệu từ nhiều nguồn khác nhau

Apache NiFi là một công cụ mạnh mẽ được thiết kế để quản lý và chuyển đổi dữ liệu từ nhiều nguồn khác nhau. NiFi hỗ trợ nhiều nguồn dữ liệu khác nhau bao gồm các hệ thống lưu trữ như HDFS, Amazon S3, cơ sở dữ liệu như MySQL, PostgreSQL, và nhiều nguồn dữ liệu khác như HTTP, FTP, Kafka, Twitter, và nhiều loại khác. Theo như trong bài báo cáo này là dữ liệu ở HTTP và từ tệp local.

Trong Nifi, mô hình lập trình dựa trên luồng (flow-based programming), trong đó các quy trình (processors) được kết nối để tạo thành các luồng xử lý dữ liệu. Các processor có thể được cấu hình để đọc và ghi dữ liệu từ các nguồn cụ thể. (Ví dụ như trong bài cáo cáo này có GetHTTP, GetFile) Ngoài ra, NiFi cung cấp các processor cho việc biến đổi (transform) và bổ sung thông tin (enrich) vào dữ liệu. Điều này cho phép bạn thực hiện các thay đổi, lọc, và làm phong phú dữ liệu dựa trên các điều kiện và luật do bạn đặt ra. (Ví dụ như UpdateAttribute, SplitXml, EvaluateJsonPath).

Hơn nữa, NiFi cho phép bạn định tuyến dữ liệu dựa trên nhiều tiêu chí như nội dung, nguồn, đích, hoặc điều kiện khác. Bạn có thể xác định các quy tắc định tuyến linh hoạt để điều chỉnh cách dữ liệu được chuyển đến và xử lý. (Như Input Port, Output Port, các connection).

Cuối cùng, NiFi cung cấp tính năng chịu lỗi (fault tolerance) bằng cách theo dõi sự hoạt động của các luồng xử lý và có khả năng tự động chuyển đến các nguồn dự phòng khi cần thiết (Ví dụ như ControlRate,...), bên cạnh đó, NiFi hỗ trợ các tính năng bảo mật như SSL, đăng nhập, và quản lý quyền để đảm bảo an toàn cho quá trình xử lý dữ liệu.

3.2.2 NiFi khi làm việc với các sensor

Khi làm việc với dữ liệu từ các cảm biến (sensors) và muốn xử lý dữ liệu thời gian thực bằng Apache NiFi, có một số điểm cần xem xét:

- Chọn các processors được tối ưu hóa cho việc xử lý dữ liệu thời gian thực như EvaluateJsonPath để trích xuất thông tin quan trọng từ dữ liệu đầu vào.
- Đảm bảo rằng bạn đã cấu hình các bộ đệm (buffers) và hàng đợi (queues) một cách phù hợp để giảm độ trễ và đảm bảo tính thời gian thực của quá trình xử lý dữ liệu. (Ví dụ trong bài báo cáo này có Back Pressure).
- Sử dụng chế độ xử lý định kỳ (scheduling) để đảm bảo rằng các processors được kích hoạt đúng thời điểm và có thể xử lý dữ liệu một cách đều đặn.
- Sử dụng các processors có khả năng xử lý dữ liệu thời gian thực, ví dụ như các processors xử lý sự kiện (event-driven), để đáp ứng nhanh chóng với dữ liệu đến từ cảm biến. (Ví dụ trong bài báo cáo này có RouteOnAttribute cho phép bạn xác định các điều kiện (conditions) dựa trên thuộc tính của sự kiện và định tuyến (route) dữ liệu theo các điều kiện đó)
- Nếu dữ liệu từ cảm biến tăng lên, xem xét vấn đề về việc mở rộng (scaling) hệ thống của bạn để đảm bảo khả năng xử lý đủ mạnh.
- Xem xét kích thước của dữ liệu và định dạng truyền tải để giảm bớt tải cho hệ thống và tăng cường khả năng xử lý thời gian thực.

3.2.3 Thu thập dữ liệu tự động từ API

Để thu thập dữ liệu tự động từ các API như API của X, Google, NextBus, Apache NiFi có thể được cấu hình sử dụng các processors và controller services liên quan đến HTTP, REST, và quản lý Credential để thực hiện nhiệm vụ này. Dưới đây là một quy trình cơ bản để xử lý việc này:

- Sử dụng processor GetHTTP để gửi các yêu cầu HTTP GET đến API của X, Google hoặc bất kỳ API nào bạn muốn thu thập dữ liệu từ.
- Thiết lập các Controller Service như StandardSSLContextService nếu API yêu cầu kết nối an toàn (HTTPS).
- Sử dụng các processor khác như EvaluateJsonPath hoặc EvaluateXPath để trích xuất thông tin cần thiết từ dữ liệu JSON hoặc XML nhận được từ API.
- Sử dụng các processors như PutFile để lưu trữ dữ liệu xuống ổ đĩa, hoặc sử dụng các processors khác phù hợp với nguồn lưu trữ bạn muốn sử dụng.
- Nếu bạn muốn tiếp tục xử lý dữ liệu đã thu thập, bạn có thể kết nối các processors khác như PutKafka để đưa dữ liệu vào một topic Kafka, PutHDFS để đưa vào Hadoop Distributed File System (HDFS), hoặc bất kỳ lưu trữ khác phù hợp với yêu cầu của bạn.

Lưu ý rằng việc xử lý API cần phải tuân thủ các quy tắc và điều kiện của API đó, bao gồm cách xác thực, giới hạn tần suất yêu cầu, và các thách thức bảo mật khác. Nếu API yêu cầu xác thực, bạn cần cung cấp thông tin xác thực hợp lệ thông qua các Controller Service.

3.2.4 Ưu điểm và nhược điểm

Ưu điểm:

- **Dễ triển khai và sử dụng:** Giao diện người dùng đồ họa giúp người dùng tạo, cấu hình và theo dõi các quy trình dễ dàng.
- **Linh hoạt và mở rộng:** Hỗ trợ nhiều nguồn và đích dữ liệu khác nhau, từ cơ sở dữ liệu đến hệ thống lưu trữ đám mây.
- **Quản lý độ an toàn:** Hỗ trợ các tính năng bảo mật như xác thực, ủy quyền và mã hóa dữ liệu.
- **Kiến trúc linh hoạt:** Thiết kế với kiến trúc mô-đun, dễ dàng tích hợp với các ứng dụng và dịch vụ khác.
- **Quản lý lỗi và giám sát:** Cung cấp công cụ để giám sát và xử lý lỗi hiệu quả, giúp đảm bảo tính ổn định của quy trình dữ liệu.
- **Hỗ trợ chuỗi công việc (workflow):** Cho phép xây dựng các chuỗi công việc phức tạp để xử lý và chuyển đổi dữ liệu theo quy trình đã định.

Nhược điểm:

- **Hiệu suất có thể giảm khi xử lý dữ liệu lớn:** Trong một số trường hợp, hiệu suất của Apache NiFi có thể bị ảnh hưởng khi xử lý lượng dữ liệu lớn.
- **Yêu cầu tài nguyên:** Cần một lượng tài nguyên đáng kể để triển khai và chạy Apache NiFi, đặc biệt là khi xử lý lượng dữ liệu lớn.
- **Khả năng tích hợp với một số công nghệ hạn chế:** Mặc dù Apache NiFi hỗ trợ nhiều nguồn và đích dữ liệu, nhưng có thể có những công nghệ cụ thể mà nó không tích hợp tốt, ví dụ như các công nghệ không hỗ trợ giao thức chuẩn, không hỗ trợ định dạng dữ liệu (JSON, XML, CSV, Avro, Parquet), ...
- **Khả năng mở rộng có thể gặp hạn chế:** Trong một số tình huống, việc mở rộng có thể đòi hỏi sự quản lý cẩn thận và cấu hình.
- **Chưa phải lúc nào cũng thích hợp cho mọi ứng dụng:** Apache NiFi được tạo ra để giải quyết những vấn đề cụ thể, và không phải là giải pháp tối ưu cho mọi loại ứng dụng.

Chương 4

KẾT LUẬN

Trong bài báo cáo này, chúng ta đã xây dựng thành công một NiFi DataFlow để giải quyết tình huống đặt ra ban đầu. Ngoài ra, chúng ta còn đạt được những kiến thức như là: Hiểu các nguyên tắc cơ bản của Apache NiFi; Giới thiệu giao diện người dùng HTML của NiFi; Giới thiệu cấu hình, mối quan hệ, nguồn gốc dữ liệu và tài liệu của NiFi; Tạo luồng dữ liệu; Kết hợp API vào luồng dữ liệu NiFi; Tìm hiểu về các mảng NiFi; Tạo nhóm quy trình (Process Group).

Với sự phát triển của công nghệ, Apache NiFi đã trở thành một dự án quan trọng trong lĩnh vực quản lý và chuyển đổi dữ liệu. Nó được sử dụng rộng rãi trong các hệ thống xử lý dữ liệu lớn, IoT, và các ngữ cảnh khác đòi hỏi xử lý dữ liệu linh hoạt và hiệu quả.

Do thời gian và khả năng có hạn nên không thể tránh khỏi sai sót. Vì vậy em mong thầy có thể nhận xét và đưa ra ý kiến để em hoàn thiện hơn bài báo cáo của mình. Cuối cùng, nhóm xin chân thành cảm ơn thầy **Hồ Quốc Dũng** đã hướng dẫn em trong suốt thời gian làm báo cáo này.

Tài liệu tham khảo

- [1] sandbox team. Analyze transit patterns with apache nifi. 1998.
- [2] Wikipedia. Apache nifi. 2023.