**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN, ĐHQG-HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

A blue and white logo

Description automatically generated

**BÁO CÁO CUỐI KỲ VỀ CASE STUDY VÀ USE CASE THỰC TẾ**

**Môn học:** Xử lý dữ liệu trực tuyến

**Bộ môn:** Hệ thống thông tin

**Mã lớp:** CQ2022/1

**GVHD:** Nguyễn Trần Minh Thư

**Tên nhóm:** Nhóm 4

**Thành viên:**

Lê Đức Cường MSSV: 21120213

Nguyễn Dương Trường Sinh MSSV: 21120322

Phạm Trần Trung Hậu MSSV: 22120100

Bùi Đoàn Thuý Vy MSSV: 22120448

*Thành phố Hồ Chí Minh, ngày 23 tháng 05 năm 2025*

# Mục lục

[Mục lục 2](#_Toc198915981)

[Danh mục bảng biểu 3](#_Toc198915982)

[Danh mục hình ảnh 3](#_Toc198915983)

[I. Phát triển bài nghiên cứu Grab Food ở cấp độ sâu hơn 4](#_Toc198915984)

[I.1 Mục tiêu: 4](#_Toc198915985)

[I.2 Phát triển 4](#_Toc198915986)

[I.2.1 GENERATION – Dữ liệu được sinh ra 4](#_Toc198915987)

[I.2.2 INGESTION – Thu thập dữ liệu 5](#_Toc198915988)

[I.2.3 TRANSFORMATION 6](#_Toc198915989)

[I.2.4 SERVING 7](#_Toc198915990)

[I.2.5 STORAGE 8](#_Toc198915991)

[I.2.6 CÁC YẾU TỔ CÁC YẾU TỔ NỀN TẢNG(UNDERCURRENTS) HỖ TRỢ PIPELINE DATA ENGINEER 10](#_Toc198915992)

[II. Chia sẻ công nghệ từ Case study thực tế - Netflix 14](#_Toc198915993)

[II.1 Lý do chọn đề tài 14](#_Toc198915994)

[II.2 Vì sao Netflix thành công như vậy? 15](#_Toc198915995)

[II.3 Dữ liệu – nhân tố định hình lợi thế cạnh tranh của Netflix 16](#_Toc198915996)

[II.4 Data engineering lifecycle tại Netflix 17](#_Toc198915997)

[II.4.1 Source system 18](#_Toc198915998)

[II.4.2 Ingestion 20](#_Toc198915999)

[II.4.3 Transformation 22](#_Toc198916000)

[II.4.4 Serving 25](#_Toc198916001)

[II.5 Vai trò của AWS trong hệ thống dữ liệu của Netflix 27](#_Toc198916002)

[II.5.1 Vì sao Netflix chọn AWS? 27](#_Toc198916003)

[II.5.2 Chiến lược “Everything on AWS” nhưng vẫn giữ tính tự chủ 28](#_Toc198916004)

[II.5.3 Vai trò của AWS trong từng tầng dữ liệu 28](#_Toc198916005)

[Tài liệu tham khảo 30](#_Toc198916006)

# Danh mục bảng biểu

[Bảng 1 Phân loại dữ liệu theo cấu trúc hình ảnh 19](#_Toc198906638)

[Bảng 2 Phân loại theo mục tiêu khai thác 20](#_Toc198906639)

[Bảng 3 Một số hệ thống cốt lõi riêng của Netflix 28](#_Toc198906640)

[Bảng 4 III.5.3 Vai trò của AWS trong từng tầng dữ liệu 29](#_Toc198906641)

# Danh mục hình ảnh

[Ảnh 1 Doanh thu của Netflix và Blockbuster từ 1995 đến 2012 15](#_Toc198906728)

[Ảnh 2 Wednesday (2022) - Một bộ phim đình đám do Netflix sản xuất 16](#_Toc198906729)

[Ảnh 3 Giao diện của Netflix trên di động 17](#_Toc198906730)

[Ảnh 4 Vòng đời của Dữ liệu tại Netflix 17](#_Toc198906731)

[Ảnh 5 Keystone Pipeline 21](#_Toc198906732)

[Ảnh 6 Từ 10000 gợi ý khả thi, các thuật toán được áp dụng để đề xuất 10-40 dòng 26](#_Toc198906733)

# Phát triển bài nghiên cứu Grab Food ở cấp độ sâu hơn

## Mục tiêu:

Cải thiện hệ thống đề xuất món ăn trên GrabFood. Cụ thể, *phân tích hành vi của người dùng để dự đoán món ăn tiếp theo* mà họ có khả năng đặt và tìm hiểu lý *do tại sao người dùng từ bỏ giỏ hàng khi chưa hoàn tất đơn hàng*.

## Phát triển

### GENERATION – Dữ liệu được sinh ra

* **Nguồn dữ liệu**
* **Nguồn phát sinh dữ liệu thô từ hệ thống thực tế:**
  + Dữ liệu hành vi người dùng trên app/web GrabFood (log đặt món, nhấp chọn món, thời gian tương tác, lịch sử đơn hàng).
  + Dữ liệu giỏ hàng: trạng thái giỏ hàng (thêm món, xóa món, thời gian từ lúc thêm đến lúc hủy hoặc hoàn tất).
  + Dữ liệu tương tác bổ sung: feedback, hủy đơn, thời gian trên trang.
  + Dữ liệu người dùng (tuổi, khu vực, thiết bị, lịch sử đặt món).
* **Đặc điểm**
  + **Tính đa dạng và biến động dữ liệu**   
    Dữ liệu phát sinh từ rất nhiều kênh, không chỉ app/web mà còn có thể từ các đối tác thanh toán, đối tác giao hàng, sự kiện bên ngoài như khuyến mãi, lễ hội, thời tiết... Đây là các yếu tố rất quan trọng ảnh hưởng đến hành vi người dùng và đặc điểm đơn hàng.
  + **Khối lượng và tốc độ**   
    Số lượng sự kiện người dùng lớn, liên tục, đặc biệt vào giờ cao điểm hoặc dịp khuyến mãi, tạo áp lực cho hệ thống thu thập và xử lý.
* **Ảnh hưởng 6V** 
  + **Volume**: Hàng triệu sự kiện mỗi ngày, dữ liệu đa dạng từ nhiều kênh (app, thanh toán, giao hàng, đối tác bên ngoài).
  + **Variety** & **Variability**: Dữ liệu đa dạng về định dạng và nguồn, biến động theo mùa vụ, chương trình khuyến mãi, sự kiện.
* **Công nghệ tiêu biểu:**
  + Các hệ thống ứng dụng (Mobile App, Web App), database truyền thống, event logging system (ví dụ: Google Analytics, Firebase)
* **Phân tích công nghệ trọng tâm: Event Logging System (Google Analytics / Firebase)**
  + Ghi lại mọi hành vi người dùng trên app/web theo thời gian thực, hỗ trợ định dạng đa dạng (sự kiện, page views, tương tác)
  + Tích hợp dễ dàng, khả năng mở rộng theo người dùng và tương tác
  + Đảm bảo dữ liệu được chuẩn hóa ngay từ đầu, tạo nền tảng cho downstream ingestion

### INGESTION – Thu thập dữ liệu

* **Dữ liệu cần thu thập:**
  + **Hành vi người dùng**: click, xem món, thêm vào giỏ, mua hàng, thời gian tương tác, từ khóa tìm kiếm, thời gian sử dụng ứng dụng, thời gian xem sản phẩm,..
  + **Thông tin đơn hàng**: món đã đặt, thời gian đặt, vị trí người dùng, tổng giá trị đơn.
  + **Các hoạt động đối với giỏ hàng**: Thêm, xóa và thay đổi số lượng các món ăn hoặc các đơn hàng được thêm món nhưng không hoàn tất.
  + **Thông tin giỏ hàng bị bỏ**: thời gian rời đi, bước rời khỏi, món trong giỏ.
  + **Dữ liệu bổ sung**: thời tiết, sự kiện đặc biệt (VD: lễ, bóng đá), chương trình khuyến mãi.
  + **Thông tin thanh toán**: Phương thức thanh toán, việc sử dụng các ưu đãi.
  + **Feedback**: đánh giá món, khảo sát sau khi hủy đơn.
* Kết hợp real-time và batch ingestion

Đây là điểm then chốt để xử lý dữ liệu nhanh, phục vụ gợi ý món ăn gần như tức thì, đồng thời vẫn có dữ liệu tổng hợp dùng cho phân tích sâu.

* **Xử lý đa dạng định dạng và nguồn (Variety):**  
  Ngoài log sự kiện, còn có dữ liệu dạng JSON, CSV từ các nguồn bên ngoài (thời tiết, bản đồ), nên cần thiết kế pipeline linh hoạt để thích ứng.
* **Công cụ**:
  + **Streaming ingestion**: dùng Apache Kafka, Kinesis hoặc Pub/Sub để thu thập sự kiện theo thời gian thực.
  + **Batch ingestion**: thu thập logs hoặc đơn hàng mỗi giờ/ngày qua job ETL.
  + Tích hợp từ các **nguồn bên ngoài** như API thời tiết hoặc Google Maps để người dùng có thể theo dõi hành trình đơn hàng theo thời gian thực hoặc thanh toán (các ứng dụng như Momo, Zalopay, Smartbanking).
* **Ảnh hưởng của 6V**
  + **Velocity**: Kết hợp streaming ingestion để xử lý real-time (Kafka, Kinesis) và batch ingestion (ETL định kỳ) cho dữ liệu tổng hợp.
  + **Variety**: Phải xử lý dữ liệu đa định dạng (log, JSON, CSV, API từ bên thứ ba).
  + **Volume**: Lượng dữ liệu khổng lồ cần pipeline có khả năng mở rộng và phân tán.
* **Công nghệ tiêu biểu**
  + Apache Kafka (streaming), AWS Kinesis, Google Pub/Sub, Apache NiFi (điều phối dữ liệu), ETL tools (Airflow, Talend)
* **Phân tích công nghệ trọng tâm: Apache Kafka**
  + Nền tảng streaming data phân tán, hỗ trợ thu thập dữ liệu real-time, độ trễ thấp
  + Cho phép xây dựng pipeline ingest dữ liệu theo mô hình publish-subscribe, dễ mở rộng theo khối lượng dữ liệu lớn
  + Hỗ trợ giữ lại dữ liệu theo thời gian để xử lý lại khi cần
  + **Liên quan 6V:** Giải quyết tốt **Velocity**, **Volume**, **Variety** do khả năng xử lý luồng sự kiện lớn, đa dạng
  + Có thể kết hợp với hệ thống batch ETL để cân bằng giữa real-time và phân tích sâu

### TRANSFORMATION

* **Tiền xử lý và biến đổi dữ liệu**
* Xử lý dữ liệu thô thành dữ liệu hữu ích
* Làm sạch dữ liệu: loại bỏ null, format lại timestamp
* Tính toán các feature như:

o Tần suất đặt món

o Thời gian người dùng rời khỏi giỏ hàng

o Tổng tiền trung bình trong giỏ hàng

* Chuẩn hóa tên các món ăn để việc phân loại và gợi ý món ăn được dễ dàng hơn ( ví dụ “cơm tấm” hay “cơm sườn” được xem như cùng một món)
* Tổng hợp các hành vi người dùng theo từng phiên hoạt động của họ (từ khi mở ứng dụng  đến lúc giao hàng).
* Tính toán các thông số về việc bỏ giỏ hàng hoặc thời gian trung bình hoàn thành đơn  hàng để đo lường độ hiệu quả của hệ thống gợi ý và cải thiện đề xuất.
* **Chú trọng Feature Engineering đa chiều**

Không chỉ tính toán tần suất hay thời gian bỏ giỏ mà còn các feature phức tạp hơn như: xu hướng món ăn theo mùa, mối liên hệ giữa các món ăn được đặt chung (market basket analysis), hoặc phân tích ngữ nghĩa từ khóa tìm kiếm bằng NLP.

* **Xử lý dữ liệu bất thường**

Phát hiện và xử lý các giá trị sai lệch (ví dụ GPS bị lệch, phản hồi không hợp lệ) để tránh gây nhiễu mô hình.

* **Phiên người dùng(Sessionization)**

Tổng hợp toàn bộ hành vi trong một phiên giúp hiểu rõ ngữ cảnh đặt món hoặc bỏ giỏ.

* **6V áp dụng**
  + Veracity: Tập trung làm sạch dữ liệu, chuẩn hóa timestamp, loại bỏ outlier.
  + Variability: Tính toán feature cần thích ứng với hành vi người dùng thay đổi theo thời gian.
  + Volume: Xử lý dữ liệu khối lượng lớn để tổng hợp thành các bảng feature cho mô hình.
* **Công nghệ tiêu biểu**
  + Apache Spark, Apache Flink, dbt (data build tool), Apache Beam, Airflow (workflow orchestration)
* **Phân tích công nghệ trọng tâm**
  + Framework xử lý dữ liệu phân tán tốc độ cao, hỗ trợ batch và streaming
  + Hỗ trợ lập trình đa ngôn ngữ (Scala, Python, Java), dễ tích hợp trong hệ sinh thái big data
  + Có khả năng xử lý data cleansing, feature engineering với hiệu năng tốt trên dữ liệu lớn
  + **Liên quan 6V:** Đáp ứng tốt **Volume** và **Veracity** với khả năng xử lý dữ liệu sạch, chuẩn hóa, loại bỏ nhiễu

### SERVING

* **Phục vụ dữ liệu và mô hình**
* Làm cho dữ liệu được sử dụng cho hệ thống downstream (phía sử dụng)
* **Analytics:**
  + Phân nhóm người dùng theo hành vi (clustering: người thường đặt 1 món, người hay xem nhưng không mua...).
  + Phân tích dữ liệu theo từng phiên hoạt động của người dùng.
  + Trực quan hoá dữ liệu theo dạng phễu để tìm các điểm gây cản trở trong quá trình hoàn tất đơn hàng.
  + Phân tích thời gian để xác định khi nào người dùng hay bỏ giỏ hàng nhất (giai đoạn chọn món, thanh toán hay vận chuyển).
  + Xây dựng tính năng đầu vào cho hệ thống gợi ý:
* Món ăn phổ biến
* Món ăn gần đây nhất
* Giờ ăn quen thuộc
* Nhà hàng thường đặt
* **Mô hình máy học:**
  + Dùng Collaborative filtering models để gợi ý các món ăn theo từng thời điểm trong ngày hoặc các món ăn được đặt cùng nhau, dự đoán thời gian người dùng đặt hàng lại và hiển thị thông báo ứng dụng phù hợp.
  + Sử dụng Clustering algorithms để gom nhóm những người dùng có hành vi sử dụng giống nhau.
  + Áp dụng các Classification models để dự đoán các khả năng người dùng bỏ giỏ hàng.
  + Sử dụng xử lý ngôn ngữ tự nhiên (NLP) để phân tích các từ khoá mà người dùng tìm kiếm để gợi ý món ăn phù hợp.
* **Reverse ETL**
  + Liên tục cập nhật lịch sử đặt hàng để phân tích và gợi ý món cho những lần đặt hàng kế tiếp.
  + Đồng bộ hoá dữ liệu người dùng theo từng nhóm để cải thiện hệ thống gợi ý món.
  + Cung cấp các gợi ý cá nhân hoá thông qua thông báo ứng dụng hoặc kênh thông tin như email cá nhân.
* **Áp dụng 6V**
  + **Value**: Tạo ra các mô hình gợi ý món ăn cá nhân hóa, dự đoán bỏ giỏ hàng chính xác.
  + **Velocity**: Phục vụ dữ liệu và dự đoán gần real-time để tăng trải nghiệm người dùng.
  + **Visibility**: Dashboard giám sát hiệu quả đề xuất và funnel bỏ giỏ hàng.
* **Công nghệ tiêu biểu**

TensorFlow, PyTorch (mô hình ML/AI), MLflow (quản lý lifecycle mô hình), Kubernetes + Docker (triển khai mô hình), Apache Superset/Tableau (dashboard, BI)

* **Phân tích công nghệ trọng tâm: TensorFlow**
  + Framework mã nguồn mở phổ biến cho xây dựng và huấn luyện các mô hình machine learning/deep learning
  + Hỗ trợ đa dạng mô hình (collaborative filtering, RNN, CNN, NLP) phục vụ đề xuất món ăn và phân tích bỏ giỏ
  + Tích hợp dễ dàng với hệ thống phục vụ real-time, hỗ trợ huấn luyện phân tán, fine-tuning mô hình
  + **Liên quan 6V:** Góp phần tạo ra **Value** thông qua mô hình đề xuất chính xác và dự đoán bỏ giỏ hàng hiệu quả

### STORAGE

* **Lưu trữ dữ liệu**
* Dữ liệu bán cấu trúc: như event logs, clickstream, lịch sử tương tác người dùng, dữ liệu giỏ hàng chưa hoàn tất… được lưu trữ trong Data Lake để đảm bảo tính linh hoạt và chi phí thấp. Các nền tảng phổ biến:
  + AWS S3
  + Google Cloud Storage (GCS)
* Dữ liệu quan hệ: bao gồm thông tin đơn hàng, người dùng, nhà hàng, thanh toán… được lưu trữ trong Data Warehouse nhằm phục vụ truy vấn nhanh, phân tích chuyên sâu và báo cáo chính xác. Công nghệ thường dùng:
  + Google BigQuery
  + Snowflake
  + Amazon Redshift
* Dữ liệu cache real-time: các thông tin được truy cập thường xuyên hoặc cần phản hồi nhanh, như lịch sử đặt món gần đây, các đề xuất cá nhân hóa được lưu trong bộ nhớ đệm (cache) nhằm giảm độ trễ, tăng trải nghiệm người dùng. Các công cụ cache:
  + Redis
  + Memcached
* **Tối ưu lưu trữ cho truy vấn nhanh (Velocity) và dung lượng lớn (Volume):**
* Kết hợp Data Lake và Data Warehouse giúp cân bằng giữa:
  + Chi phí lưu trữ: Data Lake có chi phí thấp, lưu trữ linh hoạt cho dữ liệu thô và bán cấu trúc
  + Hiệu năng truy vấn: Data Warehouse tối ưu cho truy vấn quan hệ, báo cáo và phân tích phức tạp
* Tối ưu lưu trữ, xử lý dữ liệu lớn theo mô hình phân tán, cho phép mở rộng theo nhu cầu gia tăng người dùng và dữ liệu
* **Caching Layer:**
  + Sử dụng Redis hoặc Memcached để lưu trữ tạm thời các kết quả truy vấn phổ biến và các dữ liệu đề xuất nhanh
  + Cache được cấu hình tự động làm mới hoặc hết hạn (TTL - time to live) để đảm bảo dữ liệu kịp thời và chính xác
  + Việc sử dụng cache giảm thiểu số truy vấn trực tiếp đến Data Warehouse, giảm tải hệ thống và tăng khả năng phục vụ số lượng lớn người dùng đồng thời
* **Partitioning & Indexing:**
  + Áp dụng partitioning theo các tiêu chí như:
    - Khu vực địa lý (ví dụ: thành phố, quận huyện)
    - Thời gian (ngày, tháng, quý)
  + Partitioning giúp giảm phạm vi dữ liệu khi truy vấn, tăng tốc độ xử lý và giảm chi phí
  + Thiết lập indexing trên các trường thường xuyên sử dụng để truy vấn (ví dụ: user\_id, order\_id, món ăn) nhằm tăng hiệu quả tìm kiếm và lọc dữ liệu
  + Cơ chế partition và indexing cũng hỗ trợ hiệu quả cho các công cụ phân tích thời gian thực và batch
* **6V áp dụng:**
  + Volume & Variety: Kết hợp Data Lake (AWS S3, GCS) lưu trữ dữ liệu bán cấu trúc và Data Warehouse (BigQuery, Snowflake) lưu trữ dữ liệu quan hệ.
  + Velocity: Caching layer (Redis, Memcached) để giảm độ trễ truy vấn gợi ý.
* **Công nghệ tiêu biểu:**
  + Data Lake: AWS S3, Google Cloud Storage
  + Data Warehouse: Snowflake, BigQuery, Amazon Redshift
  + Cache: Redis, Memcached
* **Phân tích công nghệ trọng tâm: Snowflake (Data Warehouse)**
  + Dịch vụ data warehouse đám mây cung cấp khả năng lưu trữ dữ liệu lớn, khả năng xử lý truy vấn mạnh mẽ, tự động mở rộng
  + Hỗ trợ phân vùng, indexing tự động, giúp truy vấn dữ liệu nhanh trên dữ liệu quan hệ
  + Tích hợp dễ dàng với nhiều công cụ phân tích và ETL
  + **Liên quan 6V:** Đáp ứng **Volume**, **Velocity**, **Variety** với khả năng quản lý dữ liệu quan hệ và bán cấu trúc

### CÁC YẾU TỔ CÁC YẾU TỔ NỀN TẢNG(UNDERCURRENTS) HỖ TRỢ PIPELINE DATA ENGINEER

#### Security – Bảo mật dữ liệu

**Vai trò và tầm quan trọng:**

* Bảo vệ dữ liệu cá nhân, giao dịch của người dùng khỏi các truy cập trái phép và tấn công mạng.
* Tuân thủ các quy định bảo vệ dữ liệu như GDPR, PCI DSS, và luật bảo mật dữ liệu Việt Nam.
* Xây dựng lòng tin người dùng, tránh rủi ro pháp lý và tài chính do rò rỉ dữ liệu.

**Ứng dụng thực tế trong pipeline**

* **Tại Source System:** Mã hóa dữ liệu đầu vào (data encryption), xác thực người dùng và thiết bị trước khi thu thập dữ liệu.
* **Trong Data Ingestion:** Sử dụng kết nối an toàn (TLS/SSL) khi truyền tải dữ liệu qua Kafka, Pub/Sub.
* **Lưu trữ dữ liệu:** Mã hóa dữ liệu tại rest (encryption at rest) trên Data Lake, Data Warehouse; phân quyền truy cập nghiêm ngặt qua IAM (Identity and Access Management).
* **Giám sát bảo mật:** Thiết lập cảnh báo bất thường (anomaly detection) trên luồng dữ liệu và truy cập hệ thống.

**Thách thức và giải pháp:**

* **Thách thức:** Quản lý quyền truy cập phức tạp do nhiều bộ phận và hệ thống tham gia.
* **Giải pháp:** Áp dụng mô hình least privilege (quyền tối thiểu cần thiết), audit logs để truy vết.

#### Data Management – Quản lý dữ liệu

**Vai trò và tầm quan trọng**

* Đảm bảo dữ liệu đầy đủ, chính xác, nhất quán và dễ truy xuất trong toàn bộ pipeline.
* Quản lý metadata để định nghĩa, ghi chú ý nghĩa, nguồn gốc dữ liệu giúp tăng khả năng tái sử dụng và giảm lỗi.

**Ứng dụng thực tế**

* Data Catalog: Sử dụng các công cụ như Apache Atlas, AWS Glue Data Catalog để quản lý metadata.
* Versioning: Áp dụng quản lý phiên bản dữ liệu, đặc biệt với dữ liệu huấn luyện mô hình ML để audit và tái tạo kết quả.
* Data Governance: Thiết lập quy trình phê duyệt, kiểm soát thay đổi dữ liệu.
* Kiểm soát chất lượng dữ liệu: Sử dụng các tool như Great Expectations để tự động kiểm tra data quality trong pipeline.

**Thách thức và giải pháp**

* Thách thức: Dữ liệu đến từ nhiều nguồn, dễ bị trùng lặp hoặc thiếu nhất quán.
* Giải pháp: Áp dụng quy trình chuẩn hóa, validation data ở các bước ingestion và transformation.

#### DataOps – Vận hành dữ liệu

**Vai trò và tầm quan trọng**

* Tự động hóa phát triển, kiểm thử, triển khai và giám sát pipeline và mô hình dữ liệu.
* Tăng tốc độ đưa sản phẩm ra thị trường, giảm lỗi do con người.

**Ứng dụng thực tế**

* Thiết lập CI/CD pipelines cho ETL và mô hình ML bằng Jenkins, GitLab CI, hoặc CircleCI.
* Giám sát pipeline tự động với các công cụ như Prometheus, Grafana để cảnh báo lỗi kịp thời.
* Sử dụng containerization (Docker) và orchestration (Kubernetes) để đảm bảo môi trường nhất quán và mở rộng linh hoạt.

**Thách thức và giải pháp**

* Thách thức: Đảm bảo pipeline phức tạp hoạt động ổn định trong môi trường đa dịch vụ.
* Giải pháp: Áp dụng monitoring toàn diện, retry mechanism, tự động rollback khi phát hiện lỗi.

#### Data Architecture – Kiến trúc dữ liệu

**Vai trò và tầm quan trọng**

* Thiết kế hệ thống dữ liệu linh hoạt, mở rộng, tối ưu chi phí và hiệu năng.
* Hỗ trợ dễ dàng mở rộng theo sự tăng trưởng của người dùng và dữ liệu.

**Ứng dụng thực tế**

* **Xây dựng kiến trúc phân lớp rõ ràng:**
  + Source Layer – thu thập dữ liệu thô
  + Ingestion Layer – xử lý đầu vào realtime và batch
  + Storage Layer – data lake và data warehouse
  + Processing Layer – transformation, feature engineering
  + Serving Layer – cung cấp dữ liệu cho downstream, mô hình ML
* **Phân tách rõ ràng giữa compute và storage (Snowflake, BigQuery) để tối ưu chi phí**
* **Thiết kế partitioning, indexing phù hợp để tăng tốc truy vấn và phân tích.**

**Thách thức và giải pháp**

* Thách thức: Giữ cân bằng giữa hiệu suất, chi phí và độ phức tạp hệ thống.
* Giải pháp: Đánh giá định kỳ kiến trúc, sử dụng kiến trúc microservices, event-driven để tăng tính linh hoạt.

#### Orchestration – Điều phối luồng công việc

**Vai trò và tầm quan trọng**

* Quản lý luồng công việc ETL, xử lý dữ liệu và mô hình đảm bảo thứ tự chạy, tái thực thi khi lỗi.
* Giúp tự động hóa pipeline, giảm thiểu thao tác thủ công.

**Ứng dụng thực tế**

* Sử dụng Apache Airflow, Prefect hoặc Luigi để thiết kế workflow cho ingestion, transformation, serving.
* Tích hợp giám sát trạng thái, alert khi pipeline gặp lỗi hoặc chậm trễ.
* Hỗ trợ chạy theo lịch, theo trigger sự kiện, và xử lý phụ thuộc phức tạp.

**Thách thức và giải pháp**

* Thách thức: Xử lý luồng công việc phức tạp với nhiều bước và phụ thuộc đa chiều.
* Giải pháp: Thiết kế workflow rõ ràng, sử dụng DAG (directed acyclic graph) để quản lý phụ thuộc.

#### Software Engineering – Kỹ thuật phần mềm

**Vai trò và tầm quan trọng**

* Xây dựng code, script và mô hình có cấu trúc, dễ bảo trì và mở rộng.
* Áp dụng kiểm thử, quản lý phiên bản, tài liệu rõ ràng để đảm bảo chất lượng.

**Ứng dụng thực tế**

* Áp dụng modularization, viết function, class tái sử dụng trong các pipeline.
* Viết unit test, integration test cho pipeline ETL và mô hình ML (pytest, unittest).
* Quản lý code bằng Git, áp dụng code review, pull request để đảm bảo chất lượng.
* Container hóa ứng dụng bằng Docker để đảm bảo môi trường chạy nhất quán.

**Thách thức và giải pháp**

* Thách thức: Đội nhóm lớn phát triển code gây ra xung đột và khó quản lý.
* Giải pháp: Áp dụng quy trình Agile, DevOps, code review nghiêm ngặt, và CI/CD.

# Chia sẻ công nghệ từ Case study thực tế - Netflix

## Lý do chọn đề tài

- Netflix là hình mẫu tiêu biểu của một công ty vận hành dựa trên dữ liệu (data-driven), phù hợp với yêu cầu seminar của giảng viên.

- Câu chuyện giữa Netflix và Blockbuster gợi nhớ đến Google và Yahoo – một chủ đề gần gũi, dễ tiếp cận, thu hút sự quan tâm.

- Bài học giá trị từ câu chuyện kinh điển về đổi mới và thích nghi:

A graph with a price tag and a price tag

AI-generated content may be incorrect.

Ảnh 1 Doanh thu của Netflix và Blockbuster từ 1995 đến 2012

* Cuối thập niên 1990, Netflix chỉ là một startup cho thuê DVD qua bưu điện, trong khi Blockbuster là “đế chế” thống trị thị trường với hàng nghìn cửa hàng.
* Năm 2000, do gặp khó khăn tài chính, Netflix từng đề nghị bán mình cho Blockbuster với giá 50 triệu USD, nhưng bị CEO John Antioco từ chối, coi đây là "ý tưởng nhỏ bé, thiếu thực tế".
* Bị từ chối, Netflix kiên trì đổi mới, chuyển sang mô hình streaming và vươn lên trở thành đế chế công nghệ truyền thông toàn cầu.
* Ngược lại, Blockbuster không kịp thích nghi với thay đổi hành vi người dùng và tuyên bố phá sản vào năm 2010.
* Chủ đề mang tính thời sự, hàm chứa nhiều bài học quý giá về chiến lược, công nghệ và tư duy chuyển đổi số.

## Vì sao Netflix thành công như vậy?

- Một trong những yếu tố then chốt tạo nên thành công của Netflix là chiến lược vận hành dựa trên dữ liệu (data-driven). Từ năm 2006, Netflix đã định hướng "data-driven by design", và đến năm 2013, dữ liệu trở thành nền tảng trong mọi quyết định chiến lược – từ sản phẩm, nội dung đến tiếp thị và trải nghiệm người dùng.

- Netflix khai thác dữ liệu không chỉ để phản ánh hành vi người dùng, mà còn để dự đoán nhu cầu và tối ưu hóa nội dung. Các chiến dịch kinh doanh được triển khai dựa trên insights thu được qua phân tích dữ liệu quy mô lớn. Theo thống kê từ hệ thống data pipeline, mỗi ngày Netflix xử lý khoảng **500 tỷ sự kiện**, tương đương **1,3 PB dữ liệu**. Trong giờ cao điểm, hệ thống tiếp nhận khoảng **8 triệu sự kiện/giây**, tương đương **24 GB/s**.

- Với năng lực phân tích dữ liệu vượt trội, Netflix không chỉ nâng cao hiệu quả vận hành mà còn tạo lợi thế cạnh tranh bền vững trong ngành công nghiệp nội dung số.

## Dữ liệu – nhân tố định hình lợi thế cạnh tranh của Netflix

- Dữ liệu không chỉ là công cụ hỗ trợ ra quyết định tại Netflix, mà còn là yếu tố cốt lõi tạo nên tính độc đáo và nâng cao vị thế cạnh tranh của doanh nghiệp trong ngành công nghiệp nội dung số. Netflix đã tích hợp dữ liệu vào hầu hết các hoạt động then chốt, qua đó tạo ra ba năng lực nổi bật:

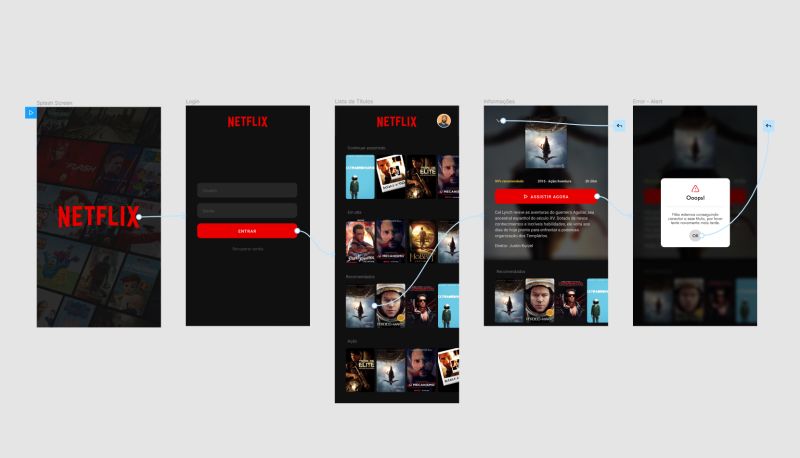
**+ Cá nhân hoá nội dung**: Hệ thống đề xuất dựa trên machine learning và hành vi người dùng giúp Netflix phân phối nội dung phù hợp với sở thích cá nhân hoá của từng tài khoản. Theo Business Insider (2012), khoảng **75–80% lượng nội dung người dùng xem** đến từ các đề xuất được hệ thống cá nhân hoá đưa ra – minh chứng rõ ràng cho hiệu quả của mô hình khai thác dữ liệu.

**+ Dự báo xu hướng và phát triển nội dung gốc**: Dữ liệu từ hành vi xem và phản hồi người dùng trên toàn cầu giúp Netflix xác định nhu cầu thị trường tiềm năng. Nhờ đó, các sản phẩm không chỉ thành công về mặt nội dung mà còn đáp ứng tốt thị hiếu đa dạng tại nhiều khu vực.



Ảnh 2 Wednesday (2022) - Một bộ phim đình đám do Netflix sản xuất

**+ Tối ưu hóa trải nghiệm người dùng (UX)**: Từ tốc độ tải nội dung, thiết kế giao diện đến thuật toán tự động điều chỉnh chất lượng video theo băng thông – tất cả đều được tối ưu hóa dựa trên phân tích dữ liệu thời gian thực, nhằm đảm bảo trải nghiệm mượt mà và cá nhân hóa.



Ảnh 3 Giao diện của Netflix trên di động

- Từ ba năng lực trên, có thể khẳng định rằng dữ liệu là nền tảng xuyên suốt trong chiến lược phát triển của Netflix. **Bài luận sẽ tiếp cận Netflix như một case study điển hình, trong đó dữ liệu được sử dụng đa mục tiêu: từ gợi ý nội dung, phát triển sản phẩm đến tối ưu vận hành – góp phần tạo nên lợi thế cạnh tranh bền vững.**

## Data engineering lifecycle tại Netflix

A screenshot of a computer

AI-generated content may be incorrect.

Ảnh 4 Vòng đời của Dữ liệu tại Netflix

- Netflix vận hành một vòng đời dữ liệu hiện đại, bao gồm bốn giai đoạn chính: Ingestion, Transformation, Storage, và Serving – đảm bảo luồng dữ liệu được xử lý hiệu quả, phục vụ đa mục tiêu từ vận hành đến phân tích.

+ Ingestion: Dữ liệu được thu thập liên tục từ hệ thống microservices (sự kiện người dùng, log hệ thống, tương tác nội dung) và các nguồn bên ngoài như Google Drive hoặc Workday. Hệ thống ingestion sử dụng Apache Kafka, Keystone và kiến trúc MAAS (Metrics as a Service) để truyền tải dữ liệu theo thời gian thực với độ trễ thấp, sẵn sàng cho xử lý tiếp theo.

+ Transformation: Dữ liệu được xử lý tại tầng SPaaS (Stream Processing as a Service) với công cụ chính là Apache Flink và hệ thống phân tích streaming nội bộ Mantis. Đồng thời, các tác vụ batch được thực hiện qua Apache Spark, Presto, và Hadoop, nhằm tổng hợp, chuẩn hóa và làm giàu dữ liệu theo lịch trình định kỳ.

+ Storage: Dữ liệu sau xử lý được lưu trữ tại nhiều hệ thống tùy theo mục đích sử dụng, bao gồm: Amazon S3 (data lake), Cassandra, MySQL, DynamoDB, ElasticSearch và EVCache. Kiến trúc lưu trữ phân tán giúp đảm bảo hiệu suất cao, tính mở rộng và tính sẵn sàng.

+ Serving: Tại tầng truy cập, dữ liệu được phân phối cho nhiều mục tiêu sử dụng: truy vấn tương tác (bằng Presto, BigQuery), trực quan hóa trên dashboard, phân tích dữ liệu bằng Jupyter Notebooks, hoặc cung cấp qua REST API phục vụ hệ thống downstream và sản phẩm cá nhân hóa.

- Kiến trúc này cho phép Netflix tối ưu toàn bộ vòng đời dữ liệu theo thời gian thực và theo lô, từ đó hỗ trợ mạnh mẽ cho các chiến lược cá nhân hóa nội dung, phát triển sản phẩm và tối ưu trải nghiệm người dùng.

### Source system

#### Những nguồn dữ liệu

- Dữ liệu hành vi người dùng :

+ Lịch sử xem: Các chương trình và phim bạn đã xem, thời lượng xem, và thời điểm xem trong ngày.

+ Tương tác: Đánh giá, lượt thích/không thích, và hành vi tìm kiếm, chia sẻ.

+ Hành vi người dùng tương tự: So sánh với người dùng có sở thích giống bạn.

+ Thiết bị và ngôn ngữ: Loại thiết bị sử dụng và ngôn ngữ ưa thích.

+ Thể loại yêu thích: Ai đang xem thể loại nào, mức độ thường xuyên.

+ Tình huống bỏ phim giữa chừng: Khi nào thì người xem ngừng xem và lý do có thể là gì.

- Phân tích xu hướng bên ngoài:

+ Mạng xã hội & diễn đàn: Theo dõi chủ đề đang "hot" trên Twitter, TikTok, YouTube, Reddit, Facebook Groups...

+ Google Trends: Xác định xu hướng tìm kiếm theo khu vực và thời gian.

+ Trang đánh giá: Thu thập phản hồi và xếp hạng từ IMDb, Rotten Tomatoes, Letterboxd.

+ Diễn đàn điện ảnh: Nắm bắt thảo luận, ý kiến từ cộng đồng yêu phim.

- Dữ liệu từ đối tác & thị trường:

+ Báo cáo nghiên cứu ngành: Từ Nielsen, Statista, McKinsey, PwC...

+ Thông tin từ chuỗi cung ứng nội dung: Nhà sản xuất, phân phối, rạp chiếu phim

#### Phân loại dữ liệu

* + - * 1. Phân loại theo cấu trúc dữ liệu

Bảng 1 Phân loại dữ liệu theo cấu trúc hình ảnh

|  |  |
| --- | --- |
| Loại dữ liệu | Nguồn dữ liệu cụ thể |
| **Có cấu trúc** | - Lịch sử xem: tên phim, thời lượng, thời điểm xem  - Hành vi tương tác: đánh giá, lượt thích, tìm kiếm  - Thiết bị & ngôn ngữ: loại thiết bị, ngôn ngữ ưa thích  - Thể loại yêu thích: thể loại xem phổ biến, tần suất  - Tình huống bỏ phim giữa chừng: timestamp, phần/phút bị bỏ  - Hành vi người dùng tương tự: nhóm người dùng theo hành vi |
| **Bán cấu trúc** | - Google Trends: dữ liệu JSON/XML từ API - Trang đánh giá (IMDb, Rotten Tomatoes...): dữ liệu HTML/JSON - Thông tin chuỗi cung ứng nội dung: metadata từ nhà sản xuất, lịch phát hành |
| **Phi cấu trúc** | - Mạng xã hội & diễn đàn: bài viết, comment, hashtag  - Diễn đàn điện ảnh: thảo luận tự do, phản hồi  - Báo cáo nghiên cứu ngành (Nielsen, PwC...): file PDF, Word, PowerPoint |

#### Phân loại theo mục tiêu khai thác

Bảng 2 Phân loại theo mục tiêu khai thác

|  |  |
| --- | --- |
| Mục tiêu | Nguồn dữ liệu cụ thể |
| **Hệ thống đề xuất cá nhân hóa** | - Lịch sử xem  - Tương tác người dùng  - Thiết bị và ngôn ngữ  - Thể loại yêu thích  - Hành vi người dùng tương tự |
| **Nắm bắt nhu cầu thị trường để sản xuất phim, đầu tư** | - Mạng xã hội & diễn đàn  - Google Trends  - Trang đánh giá  - Diễn đàn điện ảnh  - Báo cáo nghiên cứu ngành |
| **Tối ưu hóa trải nghiệm người dùng** | - Dữ liệu thiết bị  - Tình huống bỏ phim giữa chừng  - Hành vi tìm kiếm và phản hồi  - Báo cáo lỗi người dùng |

### Ingestion

- Khi người dùng tương tác với nền tảng Netflix—chẳng hạn như chọn một bộ phim, tìm kiếm nội dung, hoặc thực hiện thao tác tua nhanh—ứng dụng phía **người dùng** (frontend) sẽ gửi yêu cầu đến hệ thống phía **máy chủ** (backend) thông qua các giao thức **API** như REST hoặc GraphQL.

- GraphQL là một ngôn ngữ truy vấn dữ liệu được phát triển bởi Facebook, cho phép phía client định nghĩa chính xác dữ liệu mình cần. Khác với REST, GraphQL giúp giảm số lượng endpoint và tránh truyền tải dữ liệu dư thừa, từ đó tối ưu hiệu suất mạng và trải nghiệm người dùng.

- Sau khi nhận được yêu cầu, backend của Netflix sẽ xử lý và điều phối việc phân phối nội dung thông qua các dịch vụ **streaming** chuyên dụng, đảm bảo việc phát video diễn ra mượt mà. Đồng thời, toàn bộ dữ liệu phát sinh từ tương tác của người dùng (như hành vi play, pause, tìm kiếm, đánh giá) cũng được ghi lại để phục vụ cho phân tích và cải thiện dịch vụ.

- Trong hệ thống pipeline của Netflix, giai đoạn ingestion (tiếp nhận dữ liệu) đóng vai trò trung tâm trong việc thu thập và điều phối dữ liệu từ các nguồn đầu vào đến các hệ thống xử lý downstream. Dữ liệu sự kiện được các microservices thu thập và đẩy vào **Apache Kafka**, một nền tảng xử lý luồng dữ liệu theo thời gian thực.

- Đặc biệt, Netflix phát triển một framework ingestion nội bộ có tên **Keystone Pipeline**. Đây là hệ thống chuẩn hóa luồng dữ liệu sự kiện, giúp tích hợp linh hoạt nhiều nguồn dữ liệu khác nhau. Keystone Pipeline được thiết kế để có khả năng mở rộng cao, đảm bảo độ tin cậy và hiệu quả khi xử lý hàng tỷ sự kiện mỗi ngày.

A diagram of a system

AI-generated content may be incorrect.

Ảnh 5 Keystone Pipeline

- Dữ liệu sau khi được tiếp nhận sẽ đi qua cụm **Fronting Kafka**, đóng vai trò như một tầng đệm (buffer layer). Từ đây, dữ liệu được định tuyến đến các cụm **Kafka consumer** khác nhau, phục vụ cho các hệ thống xử lý phía sau như Apache Flink (xử lý streaming) hoặc Apache Spark (xử lý batch).

=> Cách tiếp cận này cho phép Netflix duy trì khả năng xử lý đồng thời hàng trăm tỷ sự kiện mỗi ngày, đồng thời cân bằng giữa xử lý thời gian thực (real-time) và phân tích dữ liệu theo lô (batch), góp phần tối ưu hiệu quả vận hành và cá nhân hóa trải nghiệm người dùng.

### Transformation

- Chuyển đổi dữ liệu (**Data Transformation**) là giai đoạn trung tâm trong chu trình xử lý dữ liệu tại Netflix. Mục tiêu chính là biến đổi dữ liệu thô — thu thập từ các tương tác người dùng như xem, tìm kiếm, đánh giá — thành dữ liệu sạch, có cấu trúc và mang giá trị sử dụng.

- Netflix triển khai chuyển đổi dữ liệu theo hai hướng: **streaming** (thời gian thực) và **batch** (xử lý định kỳ), tùy thuộc vào yêu cầu độ trễ và mục tiêu phân tích.

#### Streaming Processing – Xử lý dữ liệu thời gian thực

**a) Công cụ sử dụng**

- Netflix kết hợp công cụ mã nguồn mở với nền tảng phát triển nội bộ:

**+ Apache Flink**: Framework xử lý dữ liệu phân tán real-time, mã nguồn mở.

**+ Mantis** *(phát triển bởi Netflix)*: Hệ thống streaming riêng dùng cho giám sát (observability), phát hiện bất thường và cảnh báo thời gian thực.

**b) Các bước xử lý chính (sử dụng Flink hoặc Mantis)**

- **Filter** – Loại bỏ sự kiện không cần thiết. Ví dụ: Bỏ sự kiện heartbeat không chứa hành vi.

- **Enrichment** – Bổ sung metadata từ hệ thống nội dung (tên phim, thể loại, quốc gia…).

- **Join** – Kết hợp với dữ liệu hành vi người dùng. Ví dụ: Gắn thông tin phim vào log “play” từ client.

- **Aggregatio**n – Tổng hợp theo cửa sổ thời gian. Ví dụ: Đếm số lượt xem 5 phút/lần.

- **Format** – Chuẩn hóa schema đầu ra (Parquet, JSON...).

**c) Đầu ra và ứng dụng**

- Sau khi dữ liệu streaming được xử lý bởi các pipeline (Apache Flink hoặc Mantis), nó được ghi vào nhiều hệ thống lưu trữ khác nhau, tùy theo mục tiêu sử dụng.

- **Apache Druid**:

+ Dữ liệu sau xử lý được ghi trực tiếp vào Druid thông qua các connector (như Kafka-Druid connector hoặc HTTP ingestion spec). Druid lưu dữ liệu dạng cột (columnar format) và hỗ trợ chỉ mục inverted index, cho phép thực hiện các truy vấn OLAP thời gian thực với độ trễ rất thấp.

+ Ứng dụng: Giám sát số lượt xem, lỗi phát sinh theo thời gian thực trên dashboard nội bộ.

- **Apache Iceberg**:

+ Dữ liệu đã chuẩn hóa và enriched được ghi vào bảng Iceberg trên HDFS hoặc S3 bằng cách sử dụng các writer tích hợp từ Flink hoặc Spark. Iceberg hỗ trợ schema evolution và snapshot isolation, cho phép truy xuất dữ liệu chính xác tại thời điểm bất kỳ.

+ Ứng dụng: Lưu trữ dữ liệu dạng bảng phục vụ các hệ thống phân tích, như Presto hoặc Spark SQL.

- **Amazon S3**:

+ Dữ liệu gốc hoặc intermediate (định dạng JSON, Parquet hoặc Avro) được ghi vào bucket S3 theo cấu trúc phân vùng (partitioned folder structure). Dữ liệu tại đây là nguồn đầu vào cho các pipeline batch hoặc dùng để audit, replay khi cần.

+ Ứng dụng: Là nơi lưu trữ bền vững cho tất cả dữ liệu chưa cần truy cập tức thì.

**d) Ví dụ thực tiễn:**

- Một pipeline Flink xử lý các sự kiện người dùng như “pause”, “rewind” hay “exit”. Dữ liệu sau xử lý được:

+ Ghi vào Apache Druid để hiển thị ngay trên dashboard thời gian thực cho đội nội dung.

+ Đồng thời ghi vào Iceberg và S3 để dùng cho phân tích theo lô: xác định những đoạn video khiến người xem dừng nhiều, từ đó cung cấp phản hồi định lượng cho đội sáng tạo nội dung nhằm chỉnh sửa, cắt gọn hoặc cải tiến kịch bản.

#### Batch Processing – Xử lý dữ liệu theo lô (ETL)

**a) Công cụ sử dụng**

- Netflix dùng kết hợp nền tảng mã nguồn mở và công cụ nội bộ:

**+ Apache Spark**: Xử lý dữ liệu lớn theo batch.

**+ Presto**: Truy vấn SQL phân tán trên data lake.

**+ Hadoop**: Lưu trữ và xử lý dữ liệu phân tán.

**+ Meson** *(phát triển bởi Netflix)*: Hệ thống điều phối workflow dạng DAG, quản lý và tự động hóa pipeline batch.

**b) Các bước**

**+ Cleaning** – Loại bỏ bản ghi lỗi, thiếu, trùng.

**+ Normalization** – Chuẩn hóa định danh nội dung, format thời gian, schema.

**+ Join & Aggregate** – Kết hợp logs hành vi, metadata phim, kết quả hệ thống đề xuất.

**+ Advanced Transformation** – Phân cụm người dùng, trích xuất đặc trưng cho mô hình ML.

**c) Đầu ra và ứng dụng**

- Sau khi dữ liệu được xử lý thông qua pipeline batch (với Apache Spark, Presto...), kết quả sẽ được ghi vào các hệ thống lưu trữ và phân tích theo cách sau:

**- Apache Iceberg**  
+ Dữ liệu được ghi vào bảng Iceberg theo định dạng cột (Parquet hoặc Avro) thông qua Spark hoặc Presto. Iceberg hỗ trợ ghi dữ liệu với khả năng quản lý version, snapshot isolation và schema evolution. Các bảng thường được lưu trên HDFS hoặc S3.  
*+ Ứng dụng*: Truy vấn phân tích quy mô lớn (BigQuery-like) phục vụ phân tích hành vi người dùng, hiệu suất chiến dịch marketing, hoặc hiệu quả A/B testing.

**- Amazon Redshift**  
+ Dữ liệu sau khi được xử lý và làm sạch sẽ được nạp (load) vào Redshift thông qua các công cụ như AWS Glue, Apache Spark hoặc dịch vụ ETL nội bộ. Việc này thường sử dụng kết nối JDBC hoặc thông qua Amazon Redshift COPY command từ S3.  
*+ Ứng dụng*: Hỗ trợ tạo báo cáo BI định kỳ, phân tích tương tác người dùng và xây dựng dashboard hiệu suất theo thời gian.

**- Công cụ BI như Tableau, Looker hoặc Power BI**  
+ Các công cụ này kết nối trực tiếp với Redshift hoặc truy vấn dữ liệu trong Iceberg qua Presto. Dữ liệu sau khi load có thể được trực quan hóa thành biểu đồ, bảng tổng hợp hoặc bản đồ nhiệt (heatmap).  
*+ Ứng dụng*: Giúp nhà quản lý, nhà sản xuất nội dung hoặc đội marketing đưa ra quyết định dựa trên dữ liệu: xác định nội dung ưa thích, phân tích churn rate, hoặc hiệu quả của các chiến dịch cá nhân hóa.

**d) Ví dụ**: Pipeline Spark + Presto phân tích kết quả A/B test của tính năng đề xuất mới để đo ảnh hưởng tới tỷ lệ giữ chân người dùng.

### ****Serving****

Hệ thống **Serving** tại Netflix là nơi chuyển hóa dữ liệu và mô hình học máy thành giá trị thực tế: cung cấp nội dung cá nhân hóa, định hướng chiến lược nội dung, và liên tục cải tiến trải nghiệm người dùng (UX). Dưới đây là ba trụ cột chính:

#### Hệ thống đề xuất nội dung cá nhân hóa

- Netflix vận hành một trong những hệ thống đề xuất tiên tiến nhất thế giới, sử dụng các mô hình **machine learning** được huấn luyện trên data pipeline khổng lồ để dự đoán sở thích cá nhân của từng người dùng. Hệ thống này gồm nhiều tầng xử lý:

**+ Page Generation System**: Tạo giao diện homepage động, riêng biệt cho từng tài khoản dựa trên hành vi, vị trí địa lý và lịch sử xem.

**+ Ranking Models**: Xếp hạng nội dung bằng cách phân tích các đặc trưng như: thời lượng xem, đánh giá, thể loại ưa thích, tốc độ bỏ xem (*bounce rate*).

**+ Bandit Algorithms**: Áp dụng mô hình *multi-armed bandit* để cân bằng giữa việc **giới thiệu nội dung phổ biến** (exploitation) và **thử nghiệm phim mới** (exploration), nhờ đó Netflix phát hiện được các nội dung tiềm năng mới một cách nhanh chóng và hiệu quả.

- Ví dụ: Người dùng thường xem phim khoa học viễn tưởng sẽ được ưu tiên đề xuất các series như Stranger Things, nhưng cũng được thử nghiệm giới thiệu nội dung mới thuộc thể loại gần kề như Black Mirror.

A screen with a yellow arrow

AI-generated content may be incorrect.

Ảnh 6 Từ 10000 gợi ý khả thi, các thuật toán được áp dụng để đề xuất 10-40 dòng

#### Khai thác dữ liệu để định hướng sản xuất nội dung

- Không chỉ là nền tảng phân phối, Netflix còn sử dụng dữ liệu để **ra quyết định sáng tạo nội dung**:

+ Kết hợp giữa dữ liệu nội bộ (sở thích, hành vi người dùng) và dữ liệu bên ngoài (xu hướng trên mạng xã hội, diễn đàn, IMDb, TikTok).

+ Áp dụng **data science** để:

* Dự đoán thể loại hoặc mô-típ nội dung đang lên xu hướng.
* Lên kế hoạch **thời điểm phát hành tối ưu** (ví dụ: phát hành phim tình cảm trước Valentine).
* Chọn **diễn viên, đạo diễn** phù hợp với từng phân khúc người xem để tối đa hóa lượt xem.

- Ví dụ: Bộ phim The Irishman được Netflix đầu tư mạnh mẽ nhờ các phân tích cho thấy nhu cầu cao đối với phim hình sự kinh điển, cùng sức hút từ đạo diễn Martin Scorsese.

#### Tối ưu hóa trải nghiệm người dùng

- Netflix liên tục thử nghiệm và tinh chỉnh giao diện người dùng để tối ưu hóa sự hài lòng và tăng thời gian sử dụng dịch vụ:

+ Giao diện được **tùy biến theo thiết bị**, quốc gia, và hành vi sử dụng.

**+ A/B Testing** được triển khai quy mô lớn và liên tục nhằm kiểm tra hiệu quả của từng thay đổi nhỏ, như:

* Vị trí nút "Play"
* Kiểu thumbnail động
* Hiển thị preview khi rê chuột

+ Netflix sử dụng hệ thống **giám sát hiệu suất và lỗi theo thời gian thực**, giúp nhanh chóng phát hiện và xử lý các vấn đề ảnh hưởng đến trải nghiệm xem phim.

- Ví dụ: Hệ thống **Mantis** – một nền tảng observability nội bộ – giúp Netflix phát hiện bất thường UX (ví dụ: video bị dừng, app bị crash) trong thời gian gần như thực và chủ động tối ưu hóa dịch vụ.

## Vai trò của AWS trong hệ thống dữ liệu của Netflix

Netflix là một trong những doanh nghiệp tiên phong và thành công nhất trong việc chuyển đổi toàn bộ hạ tầng CNTT lên đám mây (cloud migration). Quá trình này bắt đầu vào năm 2008 và hoàn tất vào năm 2016, đánh dấu một trong những case study tiêu biểu nhất về chuyển dịch lên cloud trong ngành công nghệ.

### Vì sao Netflix chọn AWS?

Netflix lựa chọn Amazon Web Services (AWS) không chỉ vì lý do kỹ thuật, mà còn vì chiến lược phát triển lâu dài.

#### Khả năng mở rộng linh hoạt (Elasticity)

- AWS cho phép Netflix tự động điều chỉnh tài nguyên theo nhu cầu thực tế, đặc biệt trong các trường hợp:

**+ Giờ cao điểm**: khi hàng triệu người dùng đồng loạt truy cập.

**+ Ra mắt phim bom tấn**: như *Stranger Things* hoặc *The Witcher*, nhu cầu tăng đột biến → hệ thống vẫn vận hành ổn định nhờ auto-scaling của AWS.

- Dịch vụ tiêu biểu: **EC2 Auto Scaling**, **Elastic Load Balancing (ELB)**

#### Tính sẵn sàng cao (High Availability)

- AWS cung cấp kiến trúc đa vùng (multi-region architecture), giúp Netflix duy trì dịch vụ liên tục, không gián đoạn, kể cả khi một vùng gặp sự cố.

- Ví dụ: Nếu có lỗi tại cụm máy chủ ở Bắc Virginia, hệ thống vẫn hoạt động bình thường nhờ dữ liệu và dịch vụ được phân tán tại Oregon hoặc châu Âu.

- Dịch vụ hỗ trợ: Amazon Route 53, S3 Cross-Region Replication

#### Hệ sinh thái dịch vụ đa dạng, tích hợp dễ dàng

- AWS cung cấp hầu hết các thành phần mà một nền tảng dữ liệu lớn cần: từ lưu trữ, xử lý dữ liệu thời gian thực, đến các công cụ học máy và bảo mật.

- Netflix có thể triển khai nhanh chóng:

**+ Amazon S3** để lưu trữ dữ liệu lớn

**+ Amazon EMR** cho xử lý dữ liệu batch

**+ Amazon SageMaker** cho đào tạo mô hình học máy

**+ AWS IAM & KMS** cho bảo mật

### Chiến lược “Everything on AWS” nhưng vẫn giữ tính tự chủ

- Dù gần như toàn bộ hạ tầng chạy trên AWS, Netflix **không phụ thuộc hoàn toàn** mà chủ động phát triển các hệ thống cốt lõi riêng, nhằm duy trì tính linh hoạt và tránh rơi vào "vendor lock-in".

Bảng 3 Một số hệ thống cốt lõi riêng của Netflix

|  |  |
| --- | --- |
| Phần mềm | Vai trò |
| Mantis | Hệ thống quan sát dữ liệu thời gian thực |
| EVCache | Lưu đệm dữ liệu phân tán hiệu suất cao |
| Atlas | Monitoring và thu thập số liệu hệ thống |
| Zuul | API Gateway điều phối lưu lượng |
| Spinnaker | Triển khai phần mềm CI/CD, tự phát triển và open source |

- Nhờ vậy, Netflix tận dụng được hạ tầng mạnh mẽ từ AWS, nhưng vẫn giữ quyền kiểm soát kiến trúc phần mềm và khả năng tùy biến cao theo nhu cầu riêng.

### Vai trò của AWS trong từng tầng dữ liệu

Bảng 4 III.5.3 Vai trò của AWS trong từng tầng dữ liệu

|  |  |
| --- | --- |
| Tầng dữ liệu | Vai trò của AWS |
| Ingestion | Dữ liệu log ban đầu được lưu vào Amazon S3 (dạng dump tạm thời). Các dịch vụ thu thập dữ liệu chạy trên EC2 để linh hoạt mở rộng. |
| Storage | S3 đóng vai trò lưu trữ chính; hệ thống bảng dữ liệu Iceberg được triển khai trực tiếp trên S3. |
| Serving | Các công cụ như Redshift, Druid, DynamoDB phục vụ nhu cầu phân tích nhanh (BI) và truy vấn dữ liệu học máy (ML). |

# Tài liệu tham khảo

1. Nguyễn, T. M. T. (n.d.). *Slide, môn học ODAP*. SlideShare. <https://www.slideshare.net/slideshow/netflix-keystone-pipeline-at-big-data-bootcamp-santa-clara-nov-2015/55167725>
2. Netflix Technology Blog. (n.d.). *Evolution of the Netflix data pipeline*. Medium. https://netflixtechblog.com/evolution-of-the-netflix-data-pipeline-5336f8579b57
3. Business Insider. (2012, June 19). *Netflix's recommendation engine drives 75% of viewer activity*. <https://www.businessinsider.com/netflixs-recommendation-engine-drives-75-of-viewership-2012>
4. Infographic Journal. (n.d.). *Netflix vs. Blockbuster*. <https://infographicjournal.com/netflix-vs-blockbuster>
5. Amazon Web Services. (n.d.). *Netflix case study*. <https://aws.amazon.com/vi/solutions/case-studies/netflix-case-study/>
6. Netflix Technology Blog. (2019, February 12). *Cloud efficiency at Netflix*. <https://netflixtechblog.com/cloud-efficiency-at-netflix-f2a142955f83>
7. Netflix Technology Blog. (n.d.). *Netflix’s EVCache*. https://netflixtechblog.com/tagged/evcache
8. Netflix Technology Blog. (n.d.). *Netflix search with Elasticsearch*. https://netflixtechblog.com/netflix-search-how-we-rebuilt-it-and-what-we-learned-along-the-way-a9f709643f0b
9. Netflix Technology Blog. (n.d.). *Apache Flink at Netflix*. https://netflixtechblog.com/apache-flink-at-netflix-bbe4645c3cb9
10. Netflix Technology Blog. (n.d.). *Mantis: Netflix’s real-time stream processing platform*. https://netflixtechblog.com/mantis-ecosystem-update-and-roadmap-e3b7b79de3a0
11. Netflix Technology Blog. (n.d.). *Spark ETL at Netflix: Challenges and solutions*. https://netflixtechblog.com/spark-etl-at-netflix-4b4d60c3b6d
12. Netflix Technology Blog. (n.d.). *Data engineering at Netflix: Data mesh and beyond*. https://netflixtechblog.com/data-engineering-at-netflix-data-mesh-and-beyond-4c440b4bb80c
13. Netflix Technology Blog. (n.d.). *Meson: A unified data orchestration system*. https://netflixtechblog.com/meson-a-unified-data-orchestration-system-at-netflix-7fae9f1e27e4
14. Netflix Technology Blog. (n.d.). *Introducing Fresto: Federated REST operations at Netflix scale*. https://netflixtechblog.com/introducing-fresto-federated-rest-operations-at-netflix-scale-62a723ecac6a