

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**KHÓA LUẬN TỐT NGHIỆP**

**NGÀNH KỸ THUẬT DỮ LIỆU**

**PHÁT TRIỂN ỨNG DỤNG THEO DÕI GIÁ BÁN**

**VÀ ĐÁNH GIÁ HÀNG HÓA ONLINE**

**GVHD:** Thầy Trần Nhật Quang

**SVTH:**

Uông Thị Thanh Thủy 17133064

Trần Gia Bảo 17133002

Hoàng Thị Cẩm Tú 17133071

**KHÓA 2017 - 2021**

# **ĐH SƯ PHẠM KỸ THUẬT TP.HCM XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**KHOA CNTT Độc lập - Tự do - Hạnh Phúc**

**\*\*\*\*\*\*\* \*\*\*\*\*\*\***

**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

Họ và tên Sinh viên 1: Uông Thị Thanh Thủy MSSV 1: 17133064

Họ và tên Sinh viên 2: Trần Gia Bảo MSSV 2: 17133002

Họ và tên Sinh viên 3: Hoàng Thị Cẩm Tú MSSV 1: 17133071

Ngành: Kỹ thuật Dữ liệu

Tên đề tài: Phát triển ứng dụng theo dõi giá bán và đánh giá hàng hóa online.

Họ và tên Giáo viên hướng dẫn: TS.Trần Nhật Quang

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện: ........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................
2. Ưu điểm:

........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

1. Khuyết điểm:

........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

1. Đề nghị cho bảo vệ hay không ?
2. Đánh giá loại:
3. Điểm:

*Tp.Hồ Chí Minh, ngày tháng năm 2021*

*Giáo viên hướng dẫn*

*(Ký & ghi rõ họ tên)*

# **ĐH SƯ PHẠM KỸ THUẬT TP.HCM XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**KHOA CNTT Độc lập - Tự do - Hạnh Phúc**

**\*\*\*\*\*\*\* \*\*\*\*\*\*\***

**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN**

Họ và tên Sinh viên 1: Uông Thị Thanh Thủy MSSV 1: 17133064

Họ và tên Sinh viên 2: Trần Gia Bảo MSSV 2: 17133002

Họ và tên Sinh viên 3: Hoàng Thị Cẩm Tú MSSV 1: 17133071

Ngành: Kỹ thuật Dữ liệu

Tên đề tài: Phát triển ứng dụng theo dõi giá bán và đánh giá hàng hóa online.

Họ và tên Giáo viên phản biện: TS.Nguyễn Thành Sơn

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện: ........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................
2. Ưu điểm:

........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

1. Khuyết điểm:

........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

1. Đề nghị cho bảo vệ hay không ?
2. Đánh giá loại:
3. Điểm:

*Tp.Hồ Chí Minh, ngày tháng năm 2021*

*Giáo viên phản biện*

*(Ký & ghi rõ họ tên)*

# **LỜI CẢM ƠN**

Trong suốt quá trình học tập, nghiên cứu đề tài : “Phát triển ứng dụng theo dõi giá bán và đánh giá hàng hóa online”. Chúng em đã nhận được sự giúp đỡ, chỉ bảo nhiệt tình của các thầy, cô giáo Khoa Công nghệ thông tin - Trường Đại học Sư Phạm Kỹ Thuật thành phố Hồ Chí Minh. Chúng em xin chân thành cảm ơn quý thầy cô Khoa Công nghệ thông tin, Trường Đại học Sư Phạm Kỹ Thuật thành phố Hồ Chí Minh đã tạo điều kiện thuận lợi cho chúng em học tập và thực hiện đề tài tốt nghiệp này.

Đặt biệt, chúng em xin bày tỏ lòng biết ơn sâu sắc tới thầy Trần Nhật Quang đã trực tiếp tận tình hướng dẫn chỉ bảo chúng em trong quá trình thực hiện đề tài này. Trong quá trình nghiên cứu khóa luận tốt nghiệp ngành Kỹ thuật dữ liệu, chúng em đã nhận được sự quan tâm, giúp đỡ, hướng dẫn rất tận tình và tâm huyết của thầy. Thầy đã giúp chúng em tích lũy thêm nhiều kiến thức để có cái nhìn đúng và hoàn thiện hơn cho đề tài.

Đề tài là một sản phẩm mà nhóm chúng em đã nỗ lực thực hiện trong quá trình học tập. Do chưa có nhiều kinh nghiệm làm để tài cũng như những hạn chế về kiến thức, trong bài tiểu luận chắc chắn sẽ không tránh khỏi những thiếu sót. Rất mong nhận được sự thông cảm, góp ý của quý thầy cô để bài tiểu luận của chúng em được hoàn thiện hơn.

Chúng em xin chân thành cảm ơn!

TP. Hồ Chí Minh, tháng 07 năm 2021

Nhóm sinh viên thực hiện

Uông Thị Thanh Thủy

Trần Gia Bảo

Hoàng Thị Cẩm Tú

# Trường ĐH Sư Phạm Kỹ Thuật TP.HCM

# Khoa : CNTT

**ĐỀ CƯƠNG LUẬN VĂN TỐT NGHIỆP**

Họ và tên Sinh viên 1: Uông Thị Thanh Thủy MSSV 1: 17133064

Họ và tên Sinh viên 2: Trần Gia Bảo MSSV 2: 17133002

Họ và tên Sinh viên 3: Hoàng Thị Cẩm Tú MSSV 3: 17133071

Thời gian làm luận văn: Từ: Đến:

Ngành: Kỹ thuật Dữ liệu

Tên đề tài: Phát triển ứng dụng theo dõi giá bán và đánh giá hàng hóa online.

Họ và tên Giáo viên hướng dẫn: TS.Trần Nhật Quang

**Nhiệm vụ của luận văn:**

1. Hiểu được tầm quan trọng và cấp thiết của đề tài.
2. Hiểu được mô hình Crawl Data
3. Hiểu được các môn hình Scrapy
4. Tìm hiểu về NoSQL và MongoDB
5. Tìm hiểu và sử dụng học sâu, cũng như thuật toán RNN để giải quyết bài toán
6. Tìm hiểu về Node JS và React để phát triển ứng dụng.

**Đề cương viết luận văn:**

[**MỤC LỤC**](#_m5s19a1i8n3m)

[**PHẦN 1: MỞ ĐẦU**](#_wsknwg2estry)

[1. Tính cấp thiết của đề tài](#_yp0qibgowzs5)

[2. Mục đích của đề tài](#_wa8cda7ima74)

[3. Cách tiếp cận và phương pháp nghiên cứu](#_2552fhkgrlc9)

[3.1. Đối tượng nghiên cứu](#_3g8qdjfbrlsm)

[3.2. Phạm vi nghiên cứu](#_3uwvfzhni6z2)

[4. Kết quả dự kiến đạt được](#_rig2ztbl7da3)

[**PHẦN 2: NỘI DUNG**](#_3a46a876xy8)

[CHƯƠNG 1: TÌM HIỂU QUÁ TRÌNH CRAWL API PUBLIC, FAST API VÀ XỬ LÝ DỮ LIỆU THÔ](#_aqv11ffdixrt)

[1.1. Tổng quan về API](#_9o1rymb4m7h7)

[1.2. RESTful API](#_53m3zuqsyodh)

[1.3. Framework FAST API](#_v4umr37ywdc8)

[1.4. Một số khái niệm cơ bản](#_3s1ks05rg0cf)

[1.5. Request Body](#_1r5b6u718q2z)

[1.6. Xây dựng back-end FastAPI cơ bản](#_cpi6k0mnjfjq)

[1.7. Tạo schemas](#_sbd5oy58sabk)

[1.8. Crawl Data và xây dựng hệ Scrapy crawl data shopee](#_tv0tzpdwq9lp)

[1.9. Xây dựng các hệ thống Crawl Data](#_ehf38obindil)

[1.10. Hệ thống Crawl Data API Rating of Product Shopee](#_lv8i5e4acmz)

[CHƯƠNG 2: TÌM HIỂU, XÂY DỰNG VÀ ỨNG DỤNG HỌC SÂU VÀO PHÂN LOẠI THÔNG TIN QUAN ĐIỂM](#_60j1z12fmgec)

[2.1. Tổng quát về bài toán](#_s5zycvm2t7ay)

[2.2. Cơ sở mạng nơ-ron nhân tạo](#_nlaf9a458mr1)

[2.3. Mạng Nơron hồi quy RNN](#_2695gpkkwxrv)

[2.4. RNN cho bài toán trích xuất quan điểm của bình luận](#_2ztfv4pnchyl)

[CHƯƠNG 3: THIẾT KẾ ỨNG DỤNG WEBSITE HIỂN THỊ SẢN PHẨM](#_y255ivh6utvp)

[3.1. Phân tích yêu cầu](#_29j48hmgra55)

[3.2. Mô tả chi tiết các kiểu dữ liệu](#_cqpyezhmm59o)

[3.3. Sơ đồ và quy trình ứng dụng](#_33ak6h1rewxn)

[3.4. Thiết kế giao diện](#_qikm1i5ytjvk)

[3.5. Thiết kế phần mềm](#_3fm8f4527z8m)

[**PHẦN 3: KẾT LUẬN**](#_mzesqbz22gil)

[1. Kết quả đạt được](#_ze4zygtz1hpe)

[2. Đóng góp của đề tài](#_3u7cg0yrscwr)

[3. Hạn chế](#_538z7dvxpumx)

**KẾ HOẠCH THỰC HIỆN**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Thời gian** | **Công việc** | **Chú thích** |
| **1** | Từ 20-02-2021  Đến 28-02-2021 | Nghiên cứu đề tài |  |
| **2** | Từ 01-03-2021  Đến nay | Chuẩn bị code và crawl dữ liệu về thông tin sản phẩm cũng như bình luận của những người mua hàng trước đó.  Xử lý dữ liệu thô đã crawl về. | Dữ liệu như giá bán phải cập nhật liên tục .  Vào thứ 3 và thứ 7 hàng tuần sẽ tiến hành cập nhật. |
| **3** | Từ 05-2021  Đến 31-05-2021 | Nghiên cứu giải pháp và chọn thuật toán phù hợp để đào tạo phân loại bình luận. |  |
| **4** | Từ 01-06-2021  Đến 30-06-2021 | Cải thiện và nâng cấp mô hình đào tạo thành deep learning.  Lập trình website thể hiện thông tin sản phẩm và biểu đồ về giá, biểu đồ mức độ hài lòng của người mua. |  |
| **5** | Từ 01-07-2021  Đến 14-07-2021 | Hoàn thiện báo cáo |  |

Ngày 14 tháng 07 năm 2021

**Người viết đề cương**

Uông Thị Thanh Thủy

Trần Gia Bảo

Hoàng Thị Cẩm Tú

**Ý kiến của giáo viên hướng dẫn**

**(Ký và ghi rõ họ tên)**

# 

# MỤC LỤC

[**PHẦN 1: MỞ ĐẦU**](#_wsknwg2estry)**…………………………………………………………………….. 12**

[1. Tính cấp thiết của đề tài](#_yp0qibgowzs5)…………………………………………………………... 12

[2. Mục đích của đề tài](#_wa8cda7ima74)……………………………………………………………….. 13

[3. Cách tiếp cận và phương pháp nghiên cứu](#_2552fhkgrlc9)……………………………………….. 13

[3.1. Đối tượng nghiên cứu](#_3g8qdjfbrlsm)……………………………………………………….. 13

[3.2. Phạm vi nghiên cứu](#_3uwvfzhni6z2)…………………………………………………………. 13

[4. Kết quả dự kiến đạt được](#_rig2ztbl7da3)…………………………………………………………. 14

[**PHẦN 2: NỘI DUNG**](#_3a46a876xy8)………………………………………………………………….. **14**

[CHƯƠNG 1: TÌM HIỂU QUÁ TRÌNH CRAWL API PUBLIC, FAST API VÀ XỬ LÝ DỮ LIỆU THÔ](#_aqv11ffdixrt)………………………………………………………………….. 14

[1.1. Tổng quan về API](#_9o1rymb4m7h7)…...………………………………………………………. 14

[1.1.1. API là gì?](#_c4iketucqjlv)…...………………………………………………………….. 14

[1.1.2. Web API là gì?](#_bqget9q2i7xc)…...……………………………………………………. 14

[1.2. RESTful API](#_53m3zuqsyodh)…...……………………………………………………………. 15

[1.2.1. RESTful là gì ?](#_3wndtfbgrkd6)…...…………………………………………………… 15

[1.2.2. Status code](#_rbo2vb2vcwds)…...………………………………………………………... 16

[1.2.3. Authorization](#_6nnc4hs5zgxx)…...……………………………………………………… 16

[1.2.4. API Document](#_xcn24bdf2io2)…...…………………………………………………….. 16

[1.3. Framework FAST API](#_v4umr37ywdc8)…...………………………………………………….. 17

[1.3.1. Fast API là gì ?](#_478vw2avq63i)…...……………………………………………………. 17

[1.3.2. Cài đặt Framework Fast API](#_6c71jkm3m2ul)…...…………………………………….... 17

[1.3.3. Fast API Docs](#_mslg36d9ft1m)…...…………………………………………………….. 17

[1.4. Một số khái niệm cơ bản](#_3s1ks05rg0cf)…...……………………………………………….. 18

[1.4.1. Path Parameters](#_thsxwqm0z4e0)…...………………………………………………….. 18

[1.4.2. Query Parameters](#_cuwnakx41w79)…...………………………………………..……….. 19

[1.4.3. Optional parameters](#_z1f4am8g1pcp)…...………………………………...…………….. 20

[1.4.4. Multiple path and query parameters](#_l6acxlu332oo)…...………………………………. 21

[1.4.5. Required query parameters](#_8xm2po8v2qkb)…...……………………………………….. 21

[1.5. Request Body](#_1r5b6u718q2z)…...……………………………………..…………………….. 22

[1.5.1. Pydantic Models](#_2vyhz9g68wso)…...……………………….………………………….. 22

[1.5.2. Use model](#_9kjly425oqaz)…...………………………………...……………………….. 23

[1.5.3. Request body + path parameters](#_bcb6g0jimtlt)…...………………………………….. 24

[1.6. Xây dựng back-end FastAPI cơ bản](#_cpi6k0mnjfjq)…...………………………...………….. 25

[1.6.1. Các chức năng của back-end](#_lzh7p0drx5kc)…...………………………..…………….. 25

[1.6.2. Tạo database postgresql](#_esy109kc3xir)…...……………………….………………….. 25

[1.7. Tạo schemas](#_sbd5oy58sabk)…...……………………………………...…………………….. 28

[1.7.1. Tạo schemas cho Comment](#_fjbo9iesx212)…...………...…………………………….. 28

[1.7.1. Tạo schemas cho Rating](#_a3a4sck6d51o)……….………………...…………………….. 29

[1.7.2. Tạo router](#_km89c3g9hshq)…...…………………………...…………………………….. 29

[1.7.3. Router Comment](#_3zt7svqxewek)…...………………………………………………….. 29

[1.7.4. Router Rating](#_6hwtwed1aliv)…...……………………….…………………………….. 32

[1.7.5. Documents back-end Version 1.0](#_jvbzqd1hizva)…...………………………...……….. 34

[1.8. Crawl Data và xây dựng hệ Scrapy crawl data shopee](#_tv0tzpdwq9lp)…...……………...….. 36

[1.8.1. Api shopee](#_1t9o83i4lbfq)…...……………………………………………..………….. 36

[1.8.2.1. URL API List Products](#_47tsmp6suygi)…...………………………..…………….. 36

[1.8.2.2 URL API Public Detail Product](#_6cbnbsmjqpp5)…...…………………………….. 37

[1.8.2.3. URL API Comments Of Products](#_hndb09485tj2)…...….……………………….. 38

[1.8.3. Header API](#_1xwgikfq6bow)…...…………………………….………………………….. 38

[1.9. Xây dựng các hệ thống Crawl Data](#_ehf38obindil)…...…………………………………….. 39

[1.9.1. Hệ thống Crawl Data API Product Shopee](#_iba2u06oha22)…..……………………….. 39

[1.9.1.1. Quá trình hoạt động của hệ thống Crawl Data API Product Shopee](#_tl97ddjexlvu)……………………………………………………………………. 39

[1.9.1.2. Schedule của hệ thống Crawl Data API Product Shopee](#_g1qslr5urm4)………... 40

[1.9.2. Mô tả dữ liệu mà hệ thống Crawl Data API Product Shopee đã trích xuất được](#_22wvdpgfsb0r)…………………………………………….…………………………….. 40

[1.9.3. Deploy hệ thông Crawl Data API Product Shopee lên Heroku](#_hy6nb4mij7cx)…...…... 41

[1.9.4. Kết quả thu được](#_sc5750obb50v)………………………………...…………………….. 44

[1.10. Hệ thống Crawl Data API Rating of Product Shopee](#_lv8i5e4acmz)……………………... 44

[1.10.1. Quá trình hoạt động của hệ thống Crawl Data API Rating of Product Shopee](#_3puz18tz0l2c)………………………………………………………………………... 44

[1.10.2. Schedule của hệ thống Crawl Data API Rating of Product Shopee](#_fjd79apfew9m)….. 45

[1.10.3. Kết quả thu được](#_dhxjrzcg9gfx)……………………………………………………... 45

1.10.4. Crawl dữ liệu và Xây dựng hệ thống Scrapy crawl data Tiki………...45

[CHƯƠNG 2: TÌM HIỂU, XÂY DỰNG VÀ ỨNG DỤNG HỌC SÂU VÀO PHÂN LOẠI THÔNG TIN QUAN ĐIỂM](#_60j1z12fmgec)………………………………………………….. 46

[2.1. Tổng quát về bài toán](#_s5zycvm2t7ay)……………………………………………………….. 46

[2.1.1. Khái niệm và bài toán trích xuất thông tin bình luận](#_w50t4phn1xxq)………………….. 46

[2.1.2. Hướng tiếp cận và giải quyết bài toán](#_6c0l49dt4mot)…………………………………. 47

[2.2. Cơ sở mạng nơ-ron nhân tạo](#_nlaf9a458mr1)………………………………………………... 47

[2.2.1. Kiến trúc tổng quát mạng nơ-ron nhân tạo](#_lpkg4c70e4jg)……………………………. 47

[2.2.2. Hoạt động của mạng nơ-ron nhân tạo](#_vb956gmo0vdq)…………………………………. 48

[2.3. Mạng Nơron hồi quy RNN](#_2695gpkkwxrv)………………………………………………….. 50

[2.3.1. Huấn luyện mạng](#_182ecfs7merv)…………………………………………………….... 52

[2.3.2. Gated recurrent unit (GRU)](#_5zbu3nwx2scb).................................................................... 52

[2.3.2.1. Hoạt động của Cổng Xóa](#_q505olxetvqh)……………………………………….... 53

[2.3.2.2. Hoạt động của Cổng Cập Nhật](#_f1xtwbbudtgd)…………………………………... 53

[2.4. RNN cho bài toán trích xuất quan điểm của bình luận](#_2ztfv4pnchyl)……………………... 54

[2.4.1. Tập dữ liệu](#_2e1ng32fzvld)…...…………………………………………………….... 54

[2.4.2. Tiền xử lý dữ liệu dạng văn bản](#_fre06hdbkaz)…...…………………………………... 56

[2.4.3. Xây dựng mô hình RNN](#_hha0b8b4awrb)…...………………………………………….. 57

[2.4.4. Huấn luyện mô hình](#_bthenralrepp)…...………………………………………………. 58

[2.4.5. Một số kết quả khi model dự đoán trên dữ liệu thật](#_ozyj25pwo0ho)…………………... 59

[CHƯƠNG 3: THIẾT KẾ ỨNG DỤNG WEBSITE HIỂN THỊ SẢN PHẨM](#_y255ivh6utvp)………. 61

[3.1. Phân tích yêu cầu](#_29j48hmgra55)…...……………………………………………………….. 61

[3.2. Mô tả chi tiết các kiểu dữ liệu](#_cqpyezhmm59o)…...………………………………………….. 61

[3.3. Sơ đồ và quy trình ứng dụng](#_33ak6h1rewxn)…...………………………………………….... 62

[3.4. Thiết kế giao diện](#_qikm1i5ytjvk)…...…………………………………………………...….. 63

[3.5. Thiết kế phần mềm](#_3fm8f4527z8m)…...……………………………………………………... 66

[3.5.1. Giới thiệu ngăn xếp MERN](#_p3k5hcoznta2)…...……………………………………….[66](#_p3k5hcoznta2)

[3.5.2. Cấu trúc website](#_4vt8ndlw37jl)…...……………………………………………….….. 67

[**PHẦN 3: KẾT LUẬN**](#_mzesqbz22gil)**………………………………………………………………….. 70**

[1. Kết quả đạt được](#_ze4zygtz1hpe)…...……………………………………………………………... 70

[2. Đóng góp của đề tài](#_3u7cg0yrscwr)…...…………………………………………………………... 70

[3. Hạn chế](#_538z7dvxpumx)…...……………………………………………………………………….. 70

[**DANH MỤC TÀI LIỆU THAM KHẢO**](#_o2ixj9dm3ufh)…...……………………………..………….. **71**

# 

# DANH MỤC HÌNH ẢNH

[Hình 1.3.1 Docs của FastAPI](#_1hq1wnt44q8h)…………………………………………………………... .20

[Hình 1.4.1 Kết quả khi truyền parameter item\_id thông qua đường dẫn tại Docs](#_5lgewoeelmae)…….... 21

[Hình 1.4.2 Kết quả khi truyền param item\_id thông qua đường dẫn tại Browser Web](#_pfus8cnt4p3b).... 21

[Hình 1.4.3 Kết quả khi truyền parameter item\_id kiểu integer](#_48r1atpdfwgx)……………………….... .22

[Hình 1.4.4 Thông báo lỗi khi parameter không nhận đúng định dạng](#_gzsbgsug0qjx)………………….. 22

[Hình 1.4.5 Kết quả khi dùng parameter ở dạng key-value](#_ebot7bdrqbvv)……………………………... .23

[Hình 1.4.6 Kết quả khi dùng khai báo optional query parameters với q = None](#_5ssjsats1evl)………. .24

[Hình 1.4.7 Kết quả khi dùng khai báo optional query parameters](#_7za0sxgz6m10)…………………….... 24

[Hình 1.4.8 Kết quả khi dùng multiple path and query parameters](#_273awzs44us0)…………………….... 24

[Hình 1.5.1 Request body của Item](#_bl18zipfr3m3)………………………………………………………. 27

[Hình 1.5.2 Response bode Item](#_i592rg1zn5gt)…………………………………………………………. 27

[Hình 1.5.3 Response body của Item với parameter là item\_id](#_awnte5bk7eyq)…………………………. 28

[Bảng 1.6.1 Mô tả dữ liệu trong postgresql](#_bh5dbwold6it)……………………………………………… 28

[Hình 1.6.1 Table của Comments](#_3j0r9jqrvij)……………………………………………………….... 29

[Hình 1.6.2 Table của Ratings](#_sc8jwn99qjy)……………………………………………………………. 30

[Hình 1.8.3 Cách kiểm tra User-agent hiện tại](#_cvy64n53pw5l)…………………………………………... 42

[Hình 2.1.1 Các hướng tiếp cận giải quyết bài toán trích xuất thông tin bình luận](#_yruulanepz2h)…….... 50

[Hình 2.2.1 Quá trình xử lý thông tin của neural trong mạng ANN. Nguồn: www.semanticscholar.org](#_qfuuzid58g9d)……………………………………………………………….. 51

[Hình 2.3.1: Mô hình mạng RNN. Nguồn: https://dominhhai.github.io/](#_b6adbu8e5yhi)……………….... 53

[Hình 2.3.2: Ví dụ cách xử lý thông tin dạng chuỗi của RNN](#_dt2ujq2or7l0)………………………….... 54

[Hình 2.3.3 Mô hình trích xuất quan điểm cơ bản sử dụng RNN và softmax](#_a545glekm07b)………….... 54

[Hình 2.3.4 Cổng xóa và cổng cập nhật. Nguồn: d2l.aivivn.com](#_lbkb431mhzl5)………………………... 56

[Hình 2.3.5 Tính toán trạng thái ẩn tiềm năng trong một GRU. Nguồn: d2l.aivivn.com](#_agf66x2swqpi)... 56

[Hình 2.3.6 Tính toán trạng thái ẩn trong GRU. Nguồn: d2l.aivivn.com](#_y0m9y6gzw606)………………... 57

[Hình 2.4.1 Một số dòng trong dữ liệu đào tạo](#_jqnian65mwn4)…………………………………………... 58

[Hình 2.4.2 Thông tin về tập dữ liệu](#_sqlhh0mx0tw5)……………………………………………………... 58

[Hình 2.4.3 Số lượng các câu bình luận theo nhãn](#_e53rw8psuk21)………………………………………. 59

[Hình 2.4.4 Thống kê số bình luận được gán nhãn tích cực (pos), tiêu cực (neg), trung lập (neu)](#_9is3ravdme8f)................................................................................................................................... 59

[Hình 2.4.5 Số lượng các bình luận theo nhãn sau khi chọn lại mẫu](#_gufn1bl3tjml)…………………….. 59

[Hình 2.4.6 Thống kê số lượng bình luận theo nhãn sau khi chọn lại mẫu](#_vct2vnaj70p2)…………….... 59

[Hình 2.4.8 Thông tin của các lớp trong model](#_z69v2p170147)………………………………………….. 61

[Hình 2.4.9 Các giá trị accuracy theo từng epoch của tập train và validation](#_ozdgjyxbxeea)………….... 62

[Hình 2.4.10 Các giá trị loss theo từng epoch của tập train và validation](#_vn3j0tvicgn9)……………….. 62

[Hình 2.4.11 Một số kết quả khi chạy model trên tập dữ liệu thật](#_xlm8q1knbl56)……………………….. 63

[Hình 3.3.1 Sơ đồ quy trình ứng dụng](#_kh752511ohbi)………………………………………………….... 65

[Hình 3.4.1 Giao diện tìm kiếm và lựa chọn loại sản phẩm](#_wogizb5d3alg)…………………………….... 66

[Hình 3.4.2 Giao diện trình bày sản phẩm](#_kmyr77imqrq5)……………………………………………….. 67

[Hình 3.4.3 Biểu đồ giá của sản phẩm từ tháng 3 đến nay](#_cc28w8jbj8o0)………………………………. 68

[Hình 3.4.4 Biểu đánh giá sự hài lòng của người mua](#_wej0vokqo46q)………………………………….... 69

[Hình 3.5.1 Kiến trúc ngăn xếp MERN](#_lf9imwqbb078)………………………………………………….. 70

[Hình 3.5.2 Sơ đồ cấu trúc back-end NodeJS](#_x8qbtmxo1a9m)……………………………………………. 71

[Hình 3.5.3 Sơ đồ cấu trúc front-end](#_h1wxdknjmza8)…………………………………………………….. 72

# DANH SÁCH BẢNG

Bảng 1.7.2 Mô tả Documents của Rating…………………………………………….....38

Bảng 1.9.2 Mô tả dữ liệu mà hệ thống đã thu thập được………………………………..43

Bảng 1.10.1 Mô tả dữ liệu mà hệ thống đã thu thập được………………………………44

Bảng 1.10.4 Mô tả dữ liệu mà hệ thống Scrapy crawl data Tiki đã trích xuất được….....46

Bảng 1.10.2 Mô tả dữ liệu mà Model đã phân tích được………………………………..48

Bảng 3.2.1 Mô tả chi tiết kiểu dữ liệu…………………………………………………...26

# 

# DANH SÁCH TỪ VIẾT TẮT

[GRU: Gated recurrent unit](#_5zbu3nwx2scb)

RNN: Recurrent Neural Network

ANN: Artificial Neural Network

MERN: MongoDB Express ReactJS NodeJS

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# PHẦN 1: MỞ ĐẦU

## **1. Tính cấp thiết của đề tài**

Với sự phát triển của Internet, hiện nay có vô số các trang thương mại điện tử, cho phép bạn mua bán trao đổi các sản phẩm với nhiều hình thức khác nhau. Điều này đã đem lại rất nhiều lợi ích cho cả người bán và người mua.

Đối với người bán, bạn có thể giảm được chi phí quảng bá sản phẩm, tiếp thị cho thị trường toàn cầu, dịch vụ chăm sóc khách hàng tốt hơn, tăng doanh thu, giảm tồn kho, tăng lợi thế cạnh tranh. Đối với khách hàng, bạn có thể mua hàng ở khắp mọi nơi, nhiều thông tin hơn để khách hàng tối ưu quyết định khi mua, có thể so sánh giá ở nhiều trang thương mại điện tử khác nhau. Nhưng cũng có rất nhiều vấn đề xảy ra khi mua bán qua các trang thương mại điện tử như khó xây dựng được lòng tin với khách hàng khi họ không được tiếp xúc trực tiếp, gian lận trong thương mại điện tử, quá nhiều thông tin trên các trang thương mại điện tử làm người mua khó ra quyết định.

Nhận định được vấn đề đó, nhóm chúng em trọng tâm giải quyết bằng cách xây dựng, phát triển ứng dụng theo dõi giá bán và đánh giá hàng hóa online cũng như sử dụng website để thể hiện sự so sánh giá các sản phẩm trên trang thương mại điện tử. Mục đích để người dùng có thể nhìn vào biến động giá cộng thêm những đánh giá khách quan từ khách hàng về sản phẩm mà tin cậy và lựa chọn hàng hóa trên trang thương mại điện tử.

## **2. Mục đích của đề tài**

Giúp ta hiểu rõ về các mô hình crawl data, đặc biệt là Crawl Data Api Public và Fast Api. Trích xuất dữ liệu và lưu trữ. Xử lý ngôn ngữ tự nhiên và học máy sâu nhờ thuật toán Recurrent Neural Network (RNN) từ bình luận của khách hàng. Từ đó tạo được nguồn dữ liệu có cấu trúc đáng tin cậy. Áp dụng những kiến thức đã học để phát triển ứng dụng theo dõi giá bán và đánh giá hàng hóa online trên trang thương mại điện tử giúp người dùng ra quyết định tốt hơn khi mua hàng.

## **3. Cách tiếp cận và phương pháp nghiên cứu**

### **3.1. Đối tượng nghiên cứu**

* Mô hình Crawl Data Api Public và Fast Api
* Xử lý ngôn ngữ tự nhiên (NLP - Natural Language Processing)
* Thuật toán Recurrent Neural Network (RNN)
* NoSQL và MongoDB
* Node JS và React JS

### **3.2. Phạm vi nghiên cứu**

Gồm 04 phương pháp nghiên cứu khoa học để tiếp cận và làm rõ những vấn đề của đề tài.

Đó là các phương pháp nghiên cứu sau:

* Phương pháp phân tích và tổng hợp lý thuyết: Nghiên cứu các tài liệu khác nhau về các mô hình Crawl Data Api Public và Fast Api. Phân tích để tìm hiểu sâu sắc đối với mỗi vấn đề và tổng hợp để có cái nhìn tổng quan, đầy đủ về các vấn đề cần tìm hiểu.
* Phương pháp thực nghiệm khoa học: Chủ động tiến hành thu thập dữ liệu, tìm kiếm các công cụ quản lý trình thu thập thông tin, xử lý, lưu trữ và quản lý dữ liệu.
* Phương pháp xử lý ngôn ngữ tự nhiên và sử dụng thuật toán: Áp dụng thuật toán Recurrent Neural Network (RNN) để xử lý bình luận của khách hàng và tự học bằng cách đào tạo nhiều lần, từ đó đưa ra đánh giá đúng nhất về sản phẩm.
* Phương pháp phân tích, tổng kết kinh nghiệm: Từng bước xây dựng Website phù hợp nhất cho người dùng với độ tin cậy cao, tiện ích dễ sử dụng, dễ thao tác.

## **4. Kết quả dự kiến đạt được**

Áp dụng các kiến thức đã tìm hiểu. Phát triển ứng dụng theo dõi giá bán và đánh giá hàng hóa online để có thể cho người dùng ra quyết định mua hàng nhanh, nhưng độ tin cậy cao.

# 

# **PHẦN 2: NỘI DUNG**

## **CHƯƠNG 1: TÌM HIỂU QUÁ TRÌNH CRAWL API PUBLIC, FAST API VÀ XỬ LÝ DỮ LIỆU THÔ**

### **1.1. Tổng quan về API**

#### 1.1.1. API là gì?

API hiện nay đều tuân theo tiêu chuẩn REST và HTTP nhầm tạo sự thân thiện và dễ sử dụng. Giúp người dùng dễ dàng truy cập và dễ hiểu hơn thông qua document và version

API key: Đây là loại mã code được truyền tải bởi các chương trình máy tính gọi API để xác định chương trình, nhà phát triển hoặc người dùng nó tới trang web. Các API key được sử dụng với mục đích nhằm giới hạn, kiểm soát sử dụng API.

#### 1.1.2. Web API là gì?

Web API là một phương thức dùng để cho phép các ứng dụng khác nhau có thể giao tiếp, trao đổi dữ liệu qua lại. Dữ liệu được Web API trả lại thường ở dạng Json thông qua giao thức HTTP hoặc HTTPS.

Những điểm nổi bật của Web API:

* Web API hỗ trợ restful đầy đủ các phương thức: Get/Post/Put/Delete dữ liệu Nó giúp bạn xây dựng các HTTP service một cách rất đơn giản và nhanh chóng. Nó cũng có khả năng hỗ trợ đầy đủ các thành phần HTTP: URI, request/response/headers
* Khả năng sử dụng linh hoạt: API hoạt động như một đường ống dẫn dữ liệu, cho phép các bộ phận của công ty hoặc các công ty khác nhau chia sẻ thông tin được chọn, tránh những lạm dụng không mong muốn.
* Cập nhật thông tin thời gian thực: Với API bạn có thể truyền dữ liệu tốt hơn nhanh hơn với độ chính xác cao hơn, dịch vụ cung cấp linh hoạt hơn.
* Web API hoạt động như thế nào?
* Đầu tiên là xây dựng URL API để bên thứ ba có thể gửi request dữ liệu đến máy chủ cung cấp nội dung, dịch vụ thông qua giao thức HTTP hoặc HTTPS.
* Tại web server các ứng dụng sẽ thực hiện kiểm tra xác thực nếu có và tìm đến tài nguyên thích hợp để tạo nội dung sau đó trả về kết quả.

Ưu điểm

* Server trả về kết quả theo định dạng JSON hoặc XML thông qua giao thức HTTP/HTTPS.
* Tại nơi yêu cầu ban đầu là ứng dụng web hoặc ứng dụng di động , dữ liệu JSON/XML sẽ được phân tích để lấy dữ liệu. Sau khi có được dữ liệu cần thiết thì thực hiện tiếp các hoạt động như lưu dữ liệu xuống cơ sở dữ liệu, hoặc hiển thị dữ liệu…

Nhược điểm

* Để sử dụng hiệu quả cần có kiến thức chuyên sâu, có kinh nghiệm backend tốt
* Tốn thời gian và chi phí cho việc phát triển, nâng cấp và vận hành
* Có thể gặp vấn đề về bảo mật khi hệ thống bị tấn công nếu không giới hạn điều kiện kỹ.

### **1.2. RESTful API**

Đây là phương thức tạo API với các nguyên lý và các tổ chức nhất định. Những nguyên lý này nhằm hướng dẫn lập trình viên tạo môi trường xử lý API request được tốt nhất. Để hiểu rõ hơn về RESTful API ta cần giải thích qua các khái niệm REST, RESTful.

#### 1.2.1. RESTful là gì ?

REST (REpresentational State Transfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo giao tiếp giữa các ứng dụng. REST gửi một yêu cầu HTTP như GET, POST, DELETE, đến một URL để xử lý dữ liệu.

RESTful là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay. Chức năng quan trọng nhất của RESTful là quy định cách sử dụng các HTTP method (như GET, POST, PUT, DELETE…) và cách định dạng các URL cho ứng dụng web để quản các resource. RESTful không quy định logic code ứng dụng và không giới hạn bởi ngôn ngữ lập trình ứng dụng, bất kỳ ngôn ngữ hoặc framework nào cũng có thể sử dụng để thiết kế một RESTful API.

REST hoạt động chủ yếu dựa vào giao thức HTTP. Các hoạt động cơ bản nêu trên sẽ sử dụng những phương thức HTTP riêng.

* GET (SELECT): Trả về một Resource hoặc một danh sách Resource.
* POST (CREATE): Tạo mới một Resource.
* PUT (UPDATE): Cập nhật thông tin cho Resource.
* DELETE (DELETE): Xóa một Resource.

Những phương thức hay hoạt động này thường được gọi là CRUD tương ứng với Create, Read, Update, Delete – Tạo, Đọc, Sửa, Xóa.

#### 1.2.2. Status code

Khi chúng ta request một API nào đó thường thì sẽ có vài status code được trả về như sau:

* 200 Trả về thành công cho những phương thức GET, PUT, PATCH hoặc DELETE.
* 201 Created – Trả về khi một Resource vừa được tạo thành công.
* 204 No Content – Trả về khi Resource xoá thành công.
* 304 Not Modified – Client có thể sử dụng dữ liệu cache.
* 400 Bad Request – Request không hợp lệ
* 401 Unauthorized – Request cần có auth.
* 403 Forbidden – bị từ chối không cho phép.
* 404 Not Found – Không tìm thấy resource từ URI
* 405 Method Not Allowed – Phương thức không cho phép với user hiện tại.
* 410 Gone – Resource không còn tồn tại, Version cũ đã không còn hỗ trợ.
* 415 Unsupported Media Type – Không hỗ trợ kiểu Resource này.
* 422 Unprocessable Entity – Dữ liệu không được xác thực
* 429 Too Many Requests – Request bị từ chối do bị giới hạn

#### 1.2.3. Authorization

Hiện tại có 3 cơ chế Authorize chính:

· HTTP Basic

· JSON Web Token (JWT)

· OAuth2

Tùy thuộc vào service của bạn, mà hãy chọn loại Authorize có mức độ phù hợp, cố gắng giữ nó càng đơn giản càng tốt.

#### 1.2.4. API Document

Ai cũng biết việc viết API documents là rất cần thiết, dưới đây là một số lưu ý lúc viết docs:

* Mô tả đầy đủ về params request: gồm những params nào, datatype, require hay optional.
* Nên đưa ra các ví dụ về HTTP requests và responses với data chuẩn.
* Cập nhật Docs thường xuyên, để sát nhất với API có bất cứ thay đổi gì.
* Format, cú pháp cần phải nhất quán, mô tả rõ ràng, chính xác.

### **1.3. Framework FAST API**

#### 1.3.1. Fast API là gì ?

FastAPI là một framework mã nguồn mở dùng để xây dựng API với hiệu suất cao, khả năng mở rộng và dễ triển khai.

Fast API có những đặc điểm nổi bật sau:

* Hiệu suất cao hơn NodeJS và Go
* Do việc triển khai đơn giản nên code nhanh hơn, bugs giảm nhiều hơn.
* Tối thiểu việc lập lại code. Các tham số truyền vào có nhiều tính năng hơn.
* Đặc biệt, hiệu suất mạnh mẽ, có thể tương tác qua docs.

#### 1.3.2. Cài đặt Framework Fast API

Yêu cầu: python 3.6+

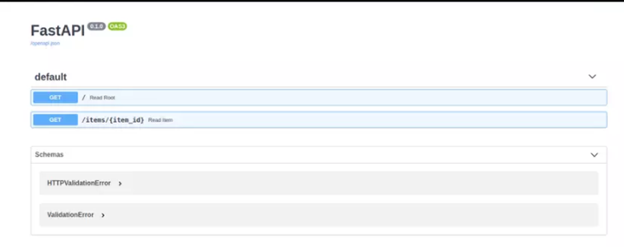
pip install fastapi

Bạn cần cài thêm ASGI server khi deploy sản phẩm như Uvicorn hoặc Hypercorn:

pip install uvicorn

#### 1.3.3. Fast API Docs

FastAPI cung cấp một giao diện dễ nhìn, dễ tương tác. Bạn cần truy cập URL như sau để mở Docs:<http://localhost:8000/docs>



##### Hình 1.3.1 Docs của FastAPI

Ngoài ra FastAPI còn có một giao diện khác là<http://localhost:8000/redoc>

### **1.4. Một số khái niệm cơ bản**

#### 1.4.1. Path Parameters

Bạn có thể truyền parameters thông qua đường dẫn như sau:

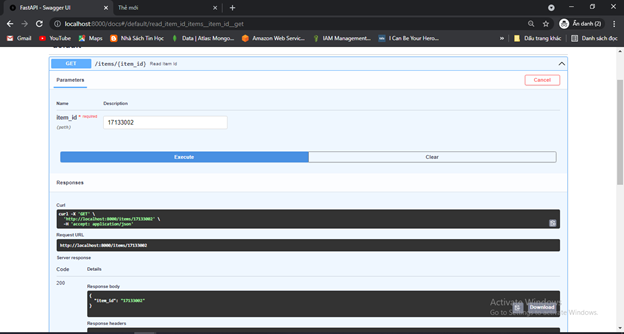
from fastapi import FastAPI

app = FastAPI()

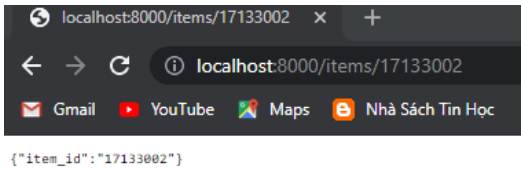
@app.get("/items/{item\_id}")

async def read\_item\_id(item\_id):

return {"item\_id": item\_id}



##### Hình 1.4.1 Kết quả khi truyền parameter item\_id thông qua đường dẫn tại Docs



##### Hình 1.4.2 Kết quả khi truyền param item\_id thông qua đường dẫn tại Browser Web

Biến item\_id trên đường dẫn URL sẽ truyền vào hàm read\_item\_id thông qua parameter item\_id

* **Path parameters with types**

Nếu cẩn thận bạn có thể khai báo định dạng của parameter như ví dụ. Khai báo kiểu dữ liệu của parameter item\_id là integer:

from fastapi import FastAPI

app = FastAPI()

@app.get("/items/{item\_id}")

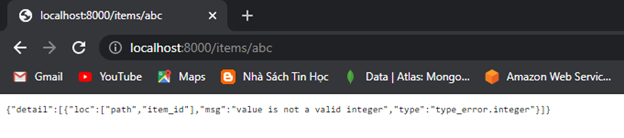
async def read\_item\_id(item\_id: int):

return {"item\_id": item\_id}

##### Hình 1.4.3 Kết quả khi truyền parameter item\_id kiểu integer

* **Data validation**

Nếu parameter nhận item\_id từ URL không đúng định dạng thì trả về thông báo. Mọi dữ liệu được validate đều dựa trên Pydantic.



##### Hình 1.4.4 Thông báo lỗi khi parameter không nhận đúng định dạng

#### 1.4.2. Query Parameters

FastAPI có hỗ trợ bạn truyền người dùng truyền parameter ở dạng key-value, nó có tên gọi là Query Parameter như ví dụ:

from fastapi import FastAPI

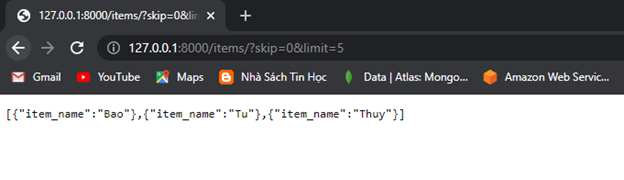
app = FastAPI()

fake\_items\_db = [{"item\_name": "Bao"}, {"item\_name": "Tu"}, {"item\_name": "Thuy"}]

@app.get("/items/")

async def read\_item(skip: int = 0, limit: int = 10):

return fake\_items\_db[skip : skip + limit]



##### Hình 1.4.5 Kết quả khi dùng parameter ở dạng key-value

#### 1.4.3. Optional parameters

Ngoài ra FastAPI cung cấp một cách khai báo optional query parameters, mặc định là None.

from typing import Optional

from fastapi import FastAPI

app = FastAPI()

@app.get("/items/{item\_id}")

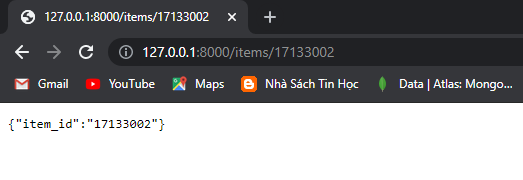
async def read\_item(item\_id: str, q: Optional[str] = None):

if q:

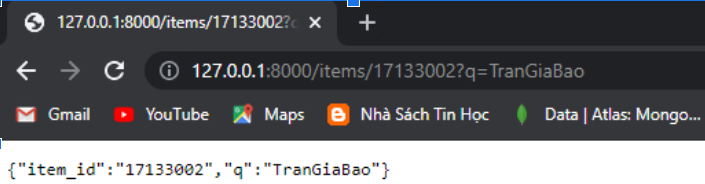
return {"item\_id": item\_id, "q": q}

return {"item\_id": item\_id}

Bạn có thể thấy trong hàm read\_item parameter item\_id sẽ nhận giá trị từ URL, và parameter q. FastAPI dùng str để nhận định format của parameter còn optional dùng để kiểm tra lỗi.



##### Hình 1.4.6 Kết quả khi dùng khai báo optional query parameters với q = None



##### Hình 1.4.7 Kết quả khi dùng khai báo optional query parameters

#### 1.4.4. Multiple path and query parameters

Với các đường dẫn lồng nhau, FastAPI biết param nào với param nào dựa trên tên param.

from fastapi import FastAPI

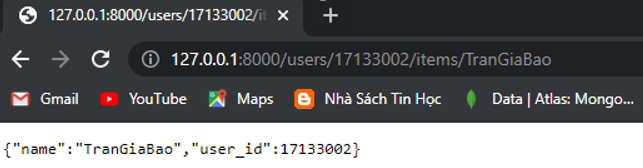
app = FastAPI()

@app.get("/users/{user\_id}/items/{item\_id}")

async def read\_user\_item(user\_id: int, item\_id: str):

item = {"name": item\_id, "user\_id": user\_id}

return item



##### Hình 1.4.8 Kết quả khi dùng multiple path and query parameters

#### 1.4.5. Required query parameters

Nếu bạn điền thiếu param trên đường dẫn URL thì sẽ xuất hiện lỗi

from fastapi import FastAPI

app = FastAPI()

@app.get("/items/{item\_id}")

async def read\_user\_item(item\_id: str, name: str):

item = {"item\_id": item\_id, "name": name}

return item

### **1.5.** **Request Body**

* Request body: người dùng gửi request từ browser đến API.
* Response body: dựa trên request, APi trả về response cho người dùng.

Để khai báo format của request body, bạn cần sử dụng Pydantic models.

#### 1.5.1. Pydantic Models

from typing import Optional

from fastapi import FastAPI

from pydantic import BaseModel # import class BaseModel của thư viện pydantic

class Item(BaseModel): # kế thừa từ class Basemodel và khai báo các biến

name: str

description: Optional[str] = None

price: float

tax: Optional[float] = None

app = FastAPI()

@app.post("/items/")

async def create\_item(item: Item): # khai báo dưới dạng parameter

return item

Dựa trên việc import Pydantic module, FastAPI hỗ trợ:

* Đọc request body dưới dạng Json.
* Chuyển đổi linh hoạt định dạng biến.
* Validate dữ liệu
* Khai báo format mặc định của request body.

#### 1.5.2. Use model

Trong hàm create\_item, bạn có thể tùy biến các biến của class Item, đơn giản như việc tính phí chịu thuế bằng cách tính tổng item.price và item.tax như sau.

from typing import Optional

from fastapi import FastAPI

from pydantic import BaseModel

class Item(BaseModel):

name: str

description: Optional[str] = None

price: float

tax: Optional[float] = None

app = FastAPI()

@app.post("/items/")

async def create\_item(item: Item):

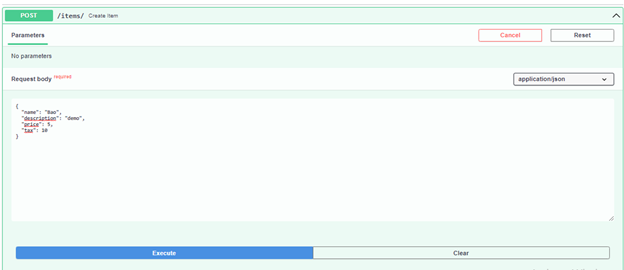
item\_dict = item.dict()

if item.tax:

price\_with\_tax = item.price + item.tax

item\_dict.update({"price\_with\_tax": price\_with\_tax})

return item\_dict



##### Hình 1.5.1 Request body của Item



##### Hình 1.5.2 Response bode Item

#### 1.5.3. Request body + path parameters

FastAPI hỗ trợ khai báo tham số URL và request body cùng lúc, framework sẽ biết tham số nào truyền từ đường dẫn và tham số nào lấy từ request

from typing import Optional

from fastapi import FastAPI

from pydantic import BaseModel

class Item(BaseModel):

name: str

description: Optional[str] = None

price: float

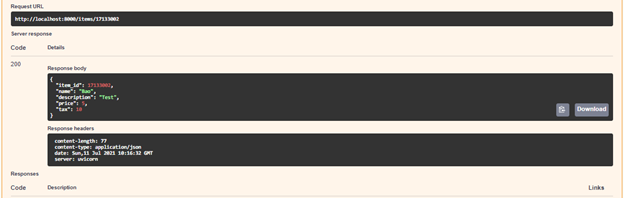
tax: Optional[float] = None

app = FastAPI()

@app.put("/items/{item\_id}")

async def create\_item(item\_id: int, item: Item):

return {"item\_id": item\_id, \*\*item.dict()}



##### Hình 1.5.3 Response body của Item với parameter là item\_id

### **1.6. Xây dựng back-end FastAPI cơ bản**

#### 1.6.1. Các chức năng của back-end

Ta sẽ xây dựng một FastAPI với các chức năng đơn giản như sau:

Đối với table comment

· Get danh sách comment theo id\_product

· Post danh sách comment theo id\_product

· Post từng comment theo id\_product

· Delete danh sách comment theo id\_product

Đối với table rating

· Get rating theo id\_product

· Post rating theo id\_product

· Delete danh sách comment theo id\_product

#### 1.6.2. Tạo database postgresql

Dùng sqlalchemy để tạo database postgresql. Bao gồm như sau:

##### Bảng 1.6.1 Mô tả dữ liệu trong postgresql

|  |  |  |
| --- | --- | --- |
| Bảng | Comments | Ratings |
| Thuộc tính | id\_comment – string  id\_product – string  comment\_product – string  rating\_comment – integer  rating\_model – integer | id\_product – string  positive\_shopee – integer  negative\_shopee – integer  neutral\_shopee – integer  positive\_model – integer  negative\_model – integer  neutral\_model – integer |

Trong đó:

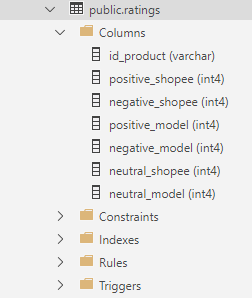
Bảng Comments bao gồm các khóa như sao:

· id\_comment là khóa chính

· id\_product là khóa phụ

##### Hình 1.6.1 Table của Comments

Bảng Ratings bao gồm khóa:

· id\_product là khóa chính

##### Hình 1.6.2 Table của Ratings

import databases

from sqlalchemy import create\_engine

import sqlalchemy

# postgres Database

DATABASE\_URL = ""

DATABASE = databases.Database(DATABASE\_URL)

metadata = sqlalchemy.MetaData()

# Tạo bảng data

dbCOMMENTS = sqlalchemy.Table(

"comments",

metadata,

sqlalchemy.Column("id\_comment", sqlalchemy.String, primary\_key=True),

sqlalchemy.Column("id\_product", sqlalchemy.String),

sqlalchemy.Column("comment\_product", sqlalchemy.String),

sqlalchemy.Column("rating\_comment", sqlalchemy.Integer),

sqlalchemy.Column("rating\_model", sqlalchemy.Integer),

)

dbRATING = sqlalchemy.Table(

"ratings",

metadata,

sqlalchemy.Column("id\_product", sqlalchemy.String, primary\_key=True),

sqlalchemy.Column("positive\_shopee", sqlalchemy.Integer),

sqlalchemy.Column("negative\_shopee", sqlalchemy.Integer),

sqlalchemy.Column("neutral\_shopee", sqlalchemy.Integer),

sqlalchemy.Column("positive\_model", sqlalchemy.Integer),

sqlalchemy.Column("negative\_model", sqlalchemy.Integer),

sqlalchemy.Column("neutral\_model", sqlalchemy.Integer),

)

engine = create\_engine(DATABASE\_URL)

metadata.create\_all(engine)

### **1.7. Tạo schemas**

#### 1.7.1. Tạo schemas cho Comment

from pydantic import BaseModel

from typing import Optional

class CommentProduct(BaseModel):

id\_comment: Optional[str] = None

id\_product: str

comment\_product: Optional[str] = None

rating\_comment: Optional[int] = None

rating\_model: Optional[int] = None

Class CommentProduct kế thừa từ class BaseModel do FastAPI hỗ trợ. Ngoại trừ id\_product thì tất cả các thuộc tính còn lại đều được nhận trước giá trị là None, tránh trường hợp không crawl được data, khi post vào sẽ bị lỗi.

#### 1.7.1. Tạo schemas cho Rating

from pydantic import BaseModel

from typing import Optional

class RatingProductInsert(BaseModel):

positive\_shopee: Optional[int] = None

negative\_shopee: Optional[int] = None

neutral\_shopee: Optional[int] = None

positive\_model: Optional[int] = None

negative\_model: Optional[int] = None

neutral\_model: Optional[int] = None

class RatingProduct(RatingProductInsert):

id\_product: str

Class RatingProductInsert được kế thừa từ class BaseModel do FastAPI hỗ trợ. Class RatingProduct được kế thừa từ class RatingProductInsert

#### 1.7.2. Tạo router

Router giúp bạn tách cách hoạt động riêng lẻ không liên quan đến nhau, để phần code của bạn được đẹp hơn dễ quản lý hơn, và có tổ chức hơn.

#### 1.7.3. Router Comment

Lấy danh sách tất cả các comment

# show all comment

@router.get("/comment", response\_model=List[CommentProduct])

async def find\_comment():

query = dbCOMMENTS.select()

return await DATABASE.fetch\_all(query)

Lấy danh sách tất cả các comment theo id\_product

# show comments theo id\_product

@router.get("/comment/{idProduct}", response\_model=List[CommentProduct])

async def find\_comment\_of\_product(idProduct: str):

query = dbCOMMENTS.select().where(dbCOMMENTS.c.id\_product == idProduct)

return await DATABASE.fetch\_all(query)

Lấy danh sách tất cả các comment theo id\_product với điều kiện rating\_comment khác với rating\_model. Điều này thể hiện sự khác biệt về trạng thái của comment.

# show comments theo id\_product với điều kiện rating shopee khác rating model

@router.get("/comment\_different/{idProduct}", response\_model=List[CommentProduct])

async def find\_comment\_different\_of\_product(idProduct: str):

query = dbCOMMENTS.select() \

.where(dbCOMMENTS.c.id\_product == idProduct) \

.where(dbCOMMENTS.c.rating\_comment != dbCOMMENTS.c.rating\_model)

return await DATABASE.fetch\_all(query)

Thêm danh sách comment vào database bảng comments dựa vào id\_product.

Nếu id\_product này đã có trong database rồi thì xóa danh sách comment đó đi và thêm vào danh sách comment mới tức update. Ngược lại thì ta chỉ thêm danh sách comment mới vào database

# input List comments vào Database

@router.post("/comment", response\_model=List[CommentProduct])

async def insert\_comment(comments: List[CommentProduct]):

query\_check = dbCOMMENTS.select().where(dbCOMMENTS.c.id\_product == str(comments[0].id\_product))

comments\_check = await DATABASE.fetch\_one(query\_check)

if comments\_check is not None:

query\_drop = dbCOMMENTS.delete().where(dbCOMMENTS.c.id\_product == str(comments[0].id\_product))

await DATABASE.execute(query\_drop)

comment\_dict = []

for comment in comments:

query = dbCOMMENTS.insert().values(

id\_comment=str(comment.id\_comment),

id\_product=str(comment.id\_product),

comment\_product=str(comment.comment\_product),

rating\_comment=int(comment.rating\_comment),

rating\_model=int(comment.rating\_model)

)

try:

await DATABASE.execute(query)

except:

pass

comment\_dict.append(comment)

return comment\_dict

Cập nhật comment dựa trên id\_comment. Nếu kiểm tra nếu thấy id\_comment này có trong database thì sẽ cập nhật lại comment này.

# update comment in id\_comment

@router.put("/comment/{idComment}", response\_model=CommentProduct)

async def update\_comment(idComment: str, comment: CommentProduct):

query = dbCOMMENTS.update() \

.where(dbCOMMENTS.c.id\_comment == idComment) \

.values(

id\_comment=str(comment.id\_comment),

id\_product=str(comment.id\_product),

comment\_product=str(comment.comment\_product),

rating\_comment=int(comment.rating\_comment),

rating\_model=int(comment.rating\_model)

)

await DATABASE.execute(query)

return {

\*\*comment.dict()

}

Xóa comment dựa trên id\_product. Nếu kiểm tra thấy trong database có id\_product thì ta sẽ xóa toàn bộ danh sách comment ấy.

# DELETE comment Database dựa vào idProduct

@router.delete("/comment/{idProduct}")

async def delete\_comment(idProduct: str):

query = dbCOMMENTS.delete().where(dbCOMMENTS.c.id\_product == idProduct)

await DATABASE.execute(query)

return {

"idProduct": idProduct,

}

#### 1.7.4. Router Rating

Lấy tất danh sách tất cả rating

# show all rating of productID

@router.get("/rating", response\_model=List[RatingProduct])

async def find\_rating():

query = dbRATING.select()

return await DATABASE.fetch\_all(query)

Lấy tất danh sách tất cả rating theo ID sản phẩm

# show rating of productID

@router.get("/rating/{idProduct}", response\_model=RatingProduct)

async def find\_rating\_of\_product(idProduct: str):

query = dbRATING.select().where(dbRATING.c.id\_product == idProduct)

return await DATABASE.fetch\_one(query)

Thêm dữ liệu rating vào database bảng ratings dựa vào id\_product. Nếu id\_product này đã có trong database rồi thì xóa dữ liệu rating đó đi và thêm dữ liệu rating mới vào. Ngược lại thì ta dữ liệu rating mới vào database

# input ratings on Database

@router.post("/rating", response\_model=RatingProduct)

async def insert\_rating(rating: RatingProduct):

query\_check = dbRATING.select().where(dbRATING.c.id\_product == str(rating.id\_product))

ratng\_check = await DATABASE.fetch\_one(query\_check)

if ratng\_check is not None:

query\_drop = dbRATING.delete().where(dbRATING.c.id\_product == str(rating.id\_product))

await DATABASE.execute(query\_drop)

query = dbRATING.insert().values(

id\_product=str(rating.id\_product),

positive\_shopee=float(rating.positive\_shopee),

negative\_shopee=float(rating.negative\_shopee),

neutral\_shopee=float(rating.neutral\_shopee),

positive\_model=float(rating.positive\_model),

negative\_model=float(rating.negative\_model),

neutral\_model=float(rating.neutral\_model),

)

await DATABASE.execute(query)

return {

\*\*rating.dict(),

}

Xóa rating dựa trên id\_product. Nếu kiểm tra thấy trong database có id\_product thì ta sẽ xóa dữ liệu rating ấy.

# DELETE rating in Database using idProduct

@router.delete("/rating/{idProduct}")

async def delete\_rating(idProduct: str):

query = dbRATING.delete().where(dbRATING.c.id\_product == idProduct)

await DATABASE.execute(query)

return {

"idProduct": idProduct,

}

#### 1.7.5. Documents back-end Version 1.0

Sau khi hoàn thành việc xây dựng một dự án FastAPI. Bạn có thể chạy dự án thông qua lệnh sau:

uvicorn main:app --reload

Hoặc chạy trên port và host đã quy định

uvicorn main:app --host 0.0.0.0 –port 8000 --reload

Sau khi chạy lệnh thì ta vào URL<http://localhost:8000/docs> để xem document của dự án mà chúng ta đã xây dựng.

##### Bảng 1.7.1 Mô tả Documents của Comment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Request | URL | Parameter | Chức năng | Kết quả |
| GET | /comment | Không có | Trả về danh sách tất cả các comment | LIST Comment |
| POST | /comment | Không có | Thêm vào database một danh sách comment | LIST Comment |
| GET | /comment/{idProduct} | idProduct | Lấy ra danh sách tất cả các comment dựa theo idProduct | LIST Comment |
| DELETE | /comment/{idProduct} | idProduct | Xóa tất cả danh sách các comment dựa theo idProduct | STRING idProduct |
| GET | /comment\_different/ {idProduct} | idProduct | Lấy ra danh sách tất cả các comment dựa theo idProduct với điều kiện rating\_model phải khác rating\_shopee | LIST Comment |
| PUT | /comment/ {idComment} | idComment | Sửa hoặc thay đổi dữ liệu của comment dựa vào idComment | DICT Comment |

##### 

##### 

##### 

##### Bảng 1.7.2 Mô tả Documents của Rating

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Request | URL | Parameter | Chức năng | Kết quả |
| GET | /rating | Không có | Trả về danh sách tất cả các rating có trong database | LIST Rating |
| POST | /rating | Không có | Thêm vào database một rating | DICT Rating |
| GET | /rating/{idProduct} | idProduct | Trả về danh sách tất cả các rating dựa vào idProduct | DICT Rating |
| DELETE | /rating/{idProduct} | idProduct | Xóa một danh sách tất cả các rating có trong database dựa vào idProduct | STRING idProduct |

### **1.8. Crawl Data và xây dựng hệ Scrapy crawl data shopee**

#### 1.8.1. Api shopee

Chúng ta truy cập vào website của Shopee: <https://shopee.vn>.

Sau đó ta nhấp tổ hợp phím Ctrl + Shift + I và chọn Network để tìm API Public của Shopee cũng như là xem cách truyền parameter vào URL API.

Dữ liệu bạn nhận được sẽ ở định dạng format Json. Điều này góp phần làm dễ dàng việc trích xuất dữ liệu từ API Public.

Sau khi tìm kiếm và phân tích thì ta tìm được các URL API Public quan trọng như sau:

· [https://shopee.vn/api/v2/search\_items/?by=relevancy&limit=50&match\_id={match\_id}&newest={newest}&order=desc&page\_type=search&version=2](https://shopee.vn/api/v2/search_items/?by=relevancy&limit=50&match_id=%7bmatch_id%7d&newest=%7bnewest%7d&order=desc&page_type=search&version=2)

·  [https://shopee.vn/api/v2/item/get?itemid={item\_id}&shopid={shop\_id}](https://shopee.vn/api/v2/item/get?itemid=%7bitem_id%7d&shopid=%7bshop_id%7d)

· https://shopee.vn/api/v2/item/get\_ratings?filter=1&flag=1&itemid={item\_id}&limit=6&offset={offset}&shopid={shopid}&type=0

**1.8.2.** **URL API Public**

##### 1.8.2.1. URL API List Products

Đây là URL lấy một số thông tin cơ bản của sản phẩm trong một danh mục dựa vào ID danh mục.

[https://shopee.vn/api/v2/search\_items/?by=relevancy&limit=50&match\_id={match\_id}&newest={newest}&order=desc&page\_type=search&version=2](https://shopee.vn/api/v2/search_items/?by=relevancy&limit=50&match_id=%7bmatch_id%7d&newest=%7bnewest%7d&order=desc&page_type=search&version=2)

Với các parameter như sau:

· match\_id là id của danh mục

· newest là số lượng sản phẩm mới nhất

Ví dụ:

Ta muốn lấy 50 sản phẩm đầu tiên của danh mục: Thời Trang Nam. Thì URL API Public là:

https://shopee.vn/api/v2/search\_items/?by=relevancy&limit=50&match\_id= 11035567&newest=0&order=desc&page\_type=search&version=2

Giải thich:

· match\_id=11035567 vì ID danh mục Thời Trang Nam là 11035567

· newest=0 vì lấy 50 sản phẩm đầu

##### 1.8.2.2 URL API Public Detail Product

Đây là URL API Public lấy tất cả thông tin chi tiết của từng sản phẩm dựa vào ID sản phẩm và ID cửa hàng

[https://shopee.vn/api/v2/item/get?itemid={item\_id}&shopid={shop\_id}](https://shopee.vn/api/v2/item/get?itemid=%7bitem_id%7d&shopid=%7bshop_id%7d)

Với các parameter như sau:

· Itemid id của sản phẩm

· Shopid id của cửa hàng

Ví dụ:

Ta muốn lấy thông tin chi tiết của sản phẩm: Bộ quần áo bảo hộ y tế, bộ đồ phòng dịch đi máy bay có kiểm định. Thì URL API Public là:

<https://shopee.vn/api/v2/item/get?itemid=3644596220&shopid=257397937>

Giải thích:

· itemid=3644596220 vì ID của sản phẩm là 3644596220

· shopid=257397937 vì ID của cửa hàng là 257397937

##### 1.8.2.3. URL API Comments Of Products

Đây là URl lấy tất cả bình luận của sản phẩm dựa vào ID sản phẩm và ID cửa hàng.

https://shopee.vn/api/v2/item/get\_ratings?filter=1&flag=1&itemid={item\_id}&limit=6&offset={offset}&shopid={shopid}&type=0

Với các parameter như sau:

· Itemid: id của sản phẩm

· Shopid: id của cửa hàng

· Offset: Phần bù

Ví dụ:

Ta muốn lấy thông tin 6 bình luận chi tiết mới nhất của sản phẩm: Bộ quần áo bảo hộ y tế, bộ đồ phòng dịch đi máy bay có kiểm định. Thì URL API Public là:

https://shopee.vn/api/v2/item/get\_ratings?filter=1&flag=1&itemid=3644596220&limit=6&offset=0shopid=257397937type=0

Giải thích:

· itemid=3644596220 vì ID của sản phẩm là 3644596220

· shopid=257397937 vì ID của cửa hàng là 257397937

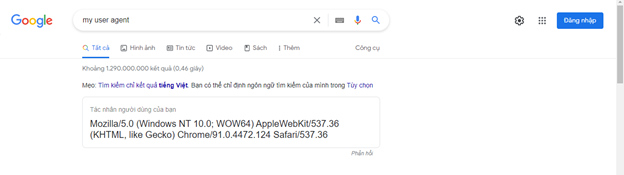
· Offset=0 vì lấy 6 bình luận đầu tiên. Nếu lấy 6 bình luận tiếp theo thì Offset=6

#### 1.8.3. Header API

Mỗi API cần đi kèm với một Header để xác nhận xem Request đó có thật sự an toàn hay không. Header rất quan trọng vì nó quyết định có hoặc không việc trả dữ liệu về từ database về cho client.

Một Header cần các parameter sau:

· User-Agent : Đây là một chuỗi nhận diện của trình duyệt Web khi gửi yêu cầu đến máy chủ Web. Khi trình duyệt của bạn truy cập vào một trang Web cụ thể, trình duyệt sẽ gửi lên một máy chủ một HTTP request trong đó có User Agent. Nội dung User Agent phụ thuộc vào trình duyệt Web mà bạn đang sử dụng. Nói chung User Agent giống như một lời giới thiệu của trình duyệt với máy chủ trước khi bắt đầu cuộc nói chuyện



##### Hình 1.8.3 Cách kiểm tra User-agent hiện tại

· If-none-match- : Khi bạn thực hiện request trực tiếp đến API public của Shopee mà không kèm If-none-match- (If-none-match- chứ không phải If-none-match), thì dữ liệu trả về đã được obfuscated gây tình trạng mất dữ liệu, dữ liệu bị mã hóa. Đây là một kiểu bảo mật dữ liệu public của Shopee.

· Cookie: Đây là một đoạn mã do trang Web mà người dùng truy cập vào tạo ra. Cookie giúp việc truy cập Web của người dùng tiện lợi và nhanh hơn. Vì không phải đăng nhập lại nhiều lần.

### **1.9. Xây dựng các hệ thống Crawl Data**

#### 1.9.1. Hệ thống Crawl Data API Product Shopee

##### 1.9.1.1. Quá trình hoạt động của hệ thống Crawl Data API Product Shopee

Sau khi có được hai URL API Public cần thiết, ta tiến hành việc trích xuất dữ liệu:

· Đầu tiên ta gửi request và header của URL API List Products đến Server Shopee, sau đó ta sẽ nhận được dữ liệu là một danh sách các sản phẩm ở định dạng Json. Ta tiến hành trích xuất hai thông tin quan trọng là Item ID ( tức ID của sản phẩm ) và Shop ID ( tức ID của cửa hàng ).

· Từ Item ID và Shop ID ta truyền vào URL API Detail Product, sau đó gửi request và header của URL API Detail Product sau khi truyền parameter. Ta nhận được dữ liệu trả về là tất cả thông tin chi tiết của sản phẩm, ta chỉ cần trích xuất thông tin mà mình cần lấy.

· Sau khi trích xuất thông tin cần thiết ở URL API Detail Product ta viết một pipeline để thêm dữ liệu vào database MongoDB.

· Pipeline được xây dựng theo phương thức không đồng bộ và real time.

· Từng dữ liệu của một sản phẩm khi được thêm vào database sẽ được kiểm tra qua item ID và shop ID, xem sản phẩm này có trong database chưa. Nếu có thì update toàn bộ dữ liệu của sản phẩm đó, riêng giá sản phẩm thì được insert vào mảng giá sản phẩm. Nếu sản phẩm chưa có trong database thì insert dữ liệu của sản phẩm vào database.

· Schedule mỗi tuần hai lần vào ngày thứ 4 và thứ 7, lúc 2h UTC

##### 1.9.1.2. Schedule của hệ thống Crawl Data API Product Shopee

· Schedule mỗi tuần hai lần vào ngày thứ 4 và thứ 7, lúc 2h UTC

· Hệ thống bắt đầu chạy từ đầu tháng 3 năm 2021

#### 1.9.2. Mô tả dữ liệu mà hệ thống Crawl Data API Product Shopee đã trích xuất được

##### Bảng 1.9.2 Mô tả dữ liệu mà hệ thống đã thu thập được

|  |  |  |
| --- | --- | --- |
| Code | Mô tả | Kiểu dữ liệu |
| catid | ID danh mục | INTEGER |
| shopid | ID cửa hàng | INTEGER |
| itemid | ID sản phẩm | INTEGER |
| name | Tên sản phẩm | STRING |
| attributes | Chi tiết sản phẩm | ARRAY |
| images | Danh sách hình sản phẩm | ARRAY |
| description | Mô tả sản phẩm | STRING |
| currency | Đơn vị tiền | STRING |
| image | Hình đại diện sản phẩm | STRING |
| shop\_location | Địa điểm cửa hàng | STRING |
| historical\_sold | Số lượng bán được | INTEGER |
| normal\_stock | Số lượng hàng tồn kho | INTEGER |
| discount | Phần trăm giảm giá | NULL/ INTEGER |
| cmt\_count | Tổng số đánh giá | INTEGER |
| liked\_count | Số lượt thích | INTEGER |
| rating\_star | Tổng điểm trên 5 SAO | DOUBLE |
| url | Liên kết đến địa chỉ Web | STRING |
| review\_price | Đánh giá giá sản phẩm | ARRAY |
| price | Giá hiện tại | INTEGER |
| price\_before\_discount | Giá trước khi giảm giá | INTEGER |
| price\_max | Giá cao nhất | INTEGER |
| price\_max\_before\_discount | Giá cao nhất ở quá khứ | INTEGER |
| price\_min | Giá thấp nhất | INTEGER |
| price\_min\_before\_discount | Giá thấp nhất ở quá khứ | INTEGER |
| DayUpdate | Thời gian crawl data | STRING |

#### 1.9.3. Deploy hệ thông Crawl Data API Product Shopee lên Heroku

· Tạo tài khoản Heroku

· Cấu hình và tạo ứng dụng mới trên Heroku thông qua cách lệnh sau:

ssh-keygen -t rsa

heroku keys:add

heroku keys

heroku create

· Heroku sẽ trả về URL như ví dụ: [https://glacial-reaches-67206.herokuapp.com](https://glacial-reaches-67206.herokuapp.com/). Sao chép URL này

· Cài đặt thêm thư viện

pip install herokuify-scrapyd

· Tạo file requirements.txt để chứa các thư viện

pip freeze > requirements.txt

· Tạo file runtime.txt với nội dung: python-3.8.5

· Tạo file scrapy.cfg với nội dung tương tự như sau:

# Automatically created by: scrapy startproject

# For more information about the [deploy] section see:

# https://scrapyd.readthedocs.io/en/latest/deploy.html

[settings]

default = SHOPEEVN.settings

[scrapyd]

application = herokuify\_scrapyd.app.application

[deploy:local]

#URL bạn nhận được trong lúc create app heroku

url = https://murmuring-plains-30059.herokuapp.com/

project = SHOPEEVN

· Tạo file periodic\_requests.py để setup schedule

import requests

import pytz

from apscheduler.schedulers.twisted import TwistedScheduler

from twisted.internet import reactor

def send\_request\_thoitrangnam():

requests.post("https://murmuring-plains-30059.herokuapp.com/schedule.json", data={

'project': 'default',

'spider' : 'thoitrangnam',

'city': 'San Francisco'

})

if \_\_name\_\_ == '\_\_main\_\_':

scheduler\_thoitrangnam = TwistedScheduler(timezone=pytz.utc)

scheduler\_thoitrangnam.add\_job(send\_request\_thoitrangnam, 'cron', day\_of\_week='2-5', hour='10', minute='00')

scheduler\_thoitrangnam.start()

reactor.run()

· Tạo file Procfile

web: scrapyd

clock: python periodic\_requests.py

· Tạo file .gitignore

\_\_pycache\_\_/

.vscode/

build/

dbs/

eggs/

project.eggs-info/

· Sử dụng GIT để deploy

git init

git add .

git commit -m "initial commit"

git push heroku master

· Nếu lỗi thì chạy lệnh sau

heroku git:remote -a glacial-reaches-67206

git push heroku master

· Chạy lệnh này để xem logs

heroku logs –tail

· Chạy lệnh này để scale clock

heroku ps:scale clock=1

#### 1.9.4. Kết quả thu được

· Danh mục hệ thống thu thập dữ liệu bao gồm: Thời Trang Nam, Thời Trang Nữ, Giày Dép Nam, Giày Dép Nữ.

· Số sản phẩm thu được đối với 1 danh mục sau mỗi là chạy là: hơn 8000 sản phẩm

· Số sản phẩm thu được đối với 4 danh mục sau mỗi là chạy là: khoảng 32000 sản phẩm

· Số sản phẩm thu được trong database đến thời điểm hiện tại:

### **1.10. Hệ thống Crawl Data API Rating of Product Shopee**

#### 1.10.1. Quá trình hoạt động của hệ thống Crawl Data API Rating of Product Shopee

Sau khi có được dữ liệu từ hệ thống **Crawl Data API Product Shopee.**

· Ta trích xuất hai thông tin là ID sản phẩm và ID cửa hàng của tất cả từ database MongoDB, để làm parameter truyền vào URL API Comments Of Product.

· Sau khi truyền parameter truyền vào URL API Comments of Product, ta gửi request cùng header đến Server Shopee.

· Ta nhận được dữ liệu trả về là một danh sách bình luận ở định dạng Json.

· Ta tiến hành việc trích xuất thông tin cần thiết từ dữ liệu trả về

##### 

##### Bảng 1.10.1 Mô tả dữ liệu mà hệ thống đã thu thập được

|  |  |  |
| --- | --- | --- |
| Code | Mô tả | Kiểu dữ liệu |
| id\_comment | ID của bình luận | STRING |
| id\_product | ID của sản phẩm | STRING |
| comment\_product | Bình luận của khách hàng đối với sản phẩm | STRING |
| rating\_comment | Xếp hạng bình luận của khách hàng đối với sản phẩm | INTEGER |

· Sau đó truyền dữ liệu đến Model Sentiment Analysis of Customer Reviews để Model phân tích. Sau khi phân tích thì ta nhận dữ liệu kết quả từ Model

##### Bảng 1.10.2 Mô tả dữ liệu mà Model đã phân tích được

|  |  |  |
| --- | --- | --- |
| Code | Mô tả | Kiểu dữ liệu |
| positive\_shopee | Trạng thái hài lòng do người dùng đánh giá | INTEGER |
| negative\_shopee | Trạng thái không hài lòng do người dùng đánh giá | INTEGER |
| neutral\_shopee | Trạng thái trung lập do người dùng đánh giá | INTEGER |
| positive\_model | Trạng thái hài lòng do model đánh giá | INTEGER |
| negative\_model | Trạng thái không hài lòng do model đánh giá | INTEGER |
| neutral\_model | Trạng thái trung lập do model đánh giá | INTEGER |

· Dùng Fast API để đưa kết quả vừa nhận được vào Database PostgreSQL, dữ liệu đã hoàn tất và sẵn sàng cho Website

#### 1.10.2. Schedule của hệ thống Crawl Data API Rating of Product Shopee

· Schedule mỗi tuần hai lần vào ngày thứ 5, lúc 2h UTC

· Hệ thống bắt đầu chạy từ đầu tháng 7 năm 2021

#### 1.10.3. Kết quả thu được

· Danh mục hệ thống thu thập dữ liệu bao gồm: Thời Trang Nam, Thời Trang Nữ, Giày Dép Nam, Giày Dép Nữ.

· Số bình luận mà hệ thống thu thập từ 1 sản phẩm là: 150/1 comment/product

· Số sản phẩm thu được đối với 1 danh mục sau mỗi là chạy là: hơn 1200000 bình luận

· Số sản phẩm thu được đối với 4 danh mục sau mỗi là chạy là: khoảng 4800000 sản phẩm

#### 1.10.4. Crawl dữ liệu và **Xây dựng hệ thống Scrapy crawl data Tiki**

Tương tự như hệ thông **Scrapy crawl data shopee**. Hệ thống này cũng hoạt động tượng tự.

Dữ liệu của hệ thống này dùng để huấn luyện Model Sentiment Analysis of Customer Reviews.

**Kết quả thu được từ hệ thống**

· Danh mục hệ thống thu thập dữ liệu bao gồm: Thời Trang Nữ, Thời Trang Nam, Baolo Vali, Giày Dép Nữ, Giày Dép Nam, Phụ Kiện, Túi Thời Trang Nữ, Túi Thời Trang Nam

· Số sản phẩm thu được đối với mỗi danh mục là: hơn 15.000 bình luận cùng với xếp hạng

· Số sản phẩm thu được đối với tất cả các danh mục là: hơn 110.000 bình luận cùng với xếp hạng

Số sản phẩm thu được trong

##### Bảng 1.10.4 Mô tả dữ liệu mà hệ thống Scrapy crawl data Tiki đã trích xuất được

|  |  |  |
| --- | --- | --- |
| Code | Mô tả | Kiểu dữ liệu |
| comment | Bình luận của khách hàng đối với sản phẩm | STRING |
| rating | Xếp hạng của khách hàng đối với sản phẩm | INTEGER |
| label | Nhãn – do xét điều kiện:  · Nếu rating bằng 4 hoặc 5 thì label là 1 tức là trạng thái hài lòng  · Nếu rating bằng 3 thì label là 0 tức là trạng thái trung lập  · Nếu rating bằng 1 hoặc 2 thì label là -1 tức là trạng thái không hài lòng | INTEGER |

## 

## **CHƯƠNG 2: TÌM HIỂU, XÂY DỰNG VÀ ỨNG DỤNG HỌC SÂU VÀO PHÂN LOẠI THÔNG TIN QUAN ĐIỂM**

### **2.1. Tổng quát về bài toán**

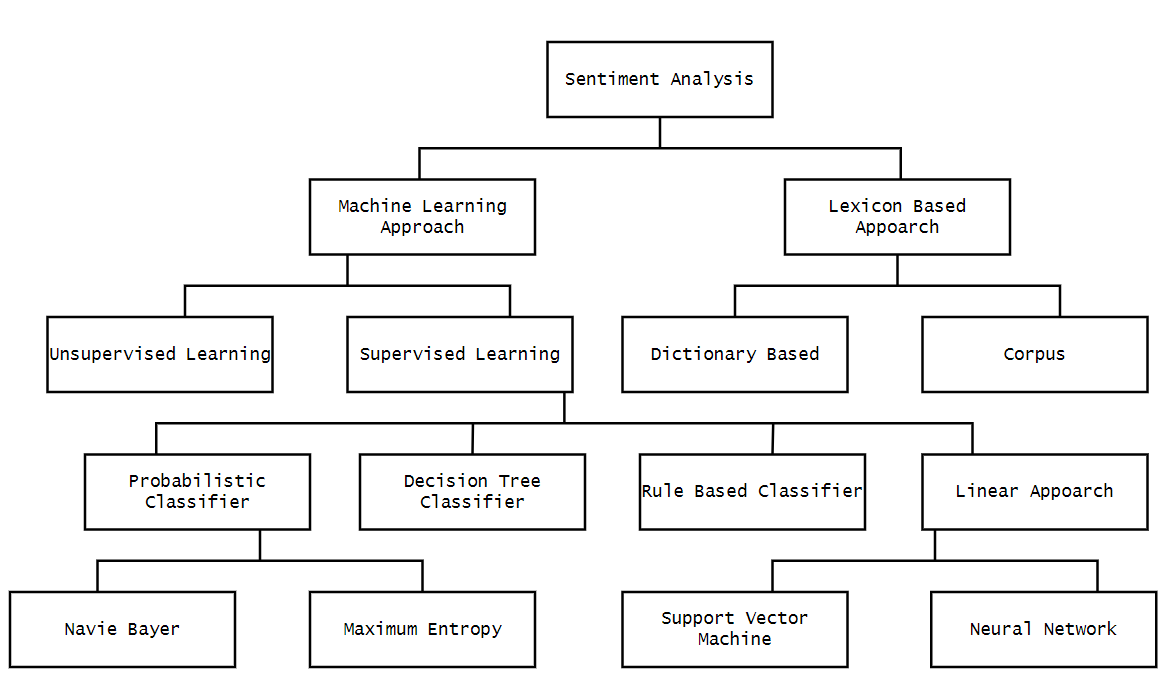
#### **2.1.1. Khái niệm và bài toán trích xuất thông tin bình luận**

Khi mua sản phẩm trên các sàn thương mại điện tử, việc để lại bình luận là quan trọng và cần thiết. Giúp cửa hàng có thể hoàn thiện dịch vụ và sản phẩm cũng như đưa ra ý kiến khách quan cho những người mua hàng sau đó đưa ra lựa chọn đúng đắn khi có ý định mua.

Bài toán trích xuất thông tin bình luận (sentiment analysis) dựa trên các thông tin phản hồi của khách hàng/người dùng trang thương mại điện tử nhằm phân loại phản hồi đó có phải tiêu cực (negative) hay tích cực (positive) cũng có khi là trung lập, không rõ ràng (neural). Dựa trên phản hồi đó có thể khẳng định được chất lượng của cửa hàng và sản phẩm. Những cửa hàng hay sản phẩm có càng nhiều phần trăm phản hồi tích cực thì càng tốt và ngược lại.

#### **2.1.2. Hướng tiếp cận và giải quyết bài toán**

Có khá nhiều hướng tiếp cận chủ đề này như dựa trên học máy hoặc dựa trên từ điển và ngữ nghĩa. Trong đó, hướng tiếp cận dựa trên học máy đang phát triển rất mạnh. Xét trên kĩ thuật học máy có giám sát có thể kể đến những thuật toán kinh điển và hiệu quả như Decision Tree, Support Vector Machine (SVM). Các thuật toán được đánh giá cao về tính đơn giản và hiệu quả trong nhiều trường hợp so với các thuật toán dựa trên mô hình mạng neural.



##### Hình 2.1.1 Các hướng tiếp cận giải quyết bài toán trích xuất thông tin bình luận

Với sự phát triển của khả năng tính toán các thuật toán mang hướng học sâu ngày càng phát triển. Báo cáo sẽ trình bày về một phương pháp dựa trên mạng neural có ưu điểm trong mô tả dữ liệu đầu vào, đó là mạng neural hồi quy RNN.

### **2.2. Cơ sở mạng nơ-ron nhân tạo**

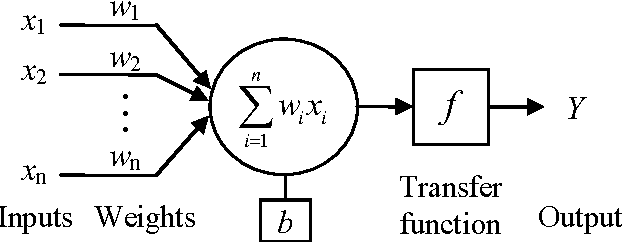
#### **2.2.1. Kiến trúc tổng quát mạng nơ-ron nhân tạo**

Mạng nơron nhân tạo (Artificial Neural Network) gọi tắt là ANN là một mô hình xử lý thông tin phỏng theo các thức xử lý thông tin của hệ thống nơron sinh học. Nó được tạo lên từ một số lượng lớn các phần tử gọi là nơron kết nối với nhau thông qua các liên kết gọi là trọng số liên kết. Mạng nơron nhân tạo thường được mô phỏng và huấn luyện từ tập mẫu. Qua quá trình huấn luyện, các trọng số liên kết sẽ được cập nhật sao cho giá trị hàm lỗi là nhỏ nhất.

Kiến trúc chung của một ANN gồm 3 thành phần đó là **Input layer**, **Hidden layer** và **Output layer**.

Trong đó, lớp ẩn (**Hidden layer**) gồm các nơron , nhận dữ liệu input từ các nơron ở lớp trước đó và chuyển đổi các input này cho các lớp xử lý tiếp theo. Trong một mạng ANN có thể có nhiều Hidden Layer.

#### **2.2.2. Hoạt động của mạng nơ-ron nhân tạo**



##### Hình 2.2.1 Quá trình xử lý thông tin của neural trong mạng ANN. Nguồn: www.semanticscholar.org

**Inputs:** Mỗi Input tương ứng với 1 đặc trưng của dữ liệu. Ví dụ như trong việc phân loại cảm xúc, mỗi từ trong một câu được coi như là một thuộc tính của dữ liệu.

**Output**: Kết quả của một ANN là một giải pháp cho một vấn đề, ví dụ như bài toán phân loại cảm xúc trên thì kết quả có thể là tích cực, tiêu cực hay trung lập.

**Connection Weights** (Trọng số liên kết): Đây là thành phần rất quan trọng của một ANN, nó thể hiện mức độ quan trọng, độ mạnh của dữ liệu đầu vào đối với quá trình xử lý thông tin chuyển đổi dữ liệu từ layer này sang layer khác. Quá trình học của ANN thực ra là quá trình điều chỉnh các trọng số (Weight) của các dữ liệu đầu vào để có được kết quả mong muốn.

**Summation Function** (Hàm tổng): Tính tổng trọng số của tất cả các input được đưa vào mỗi Nơron. Hàm tổng của một Nơron đối với n input được tính theo công thức sau:

**Độ lệch** (b): Hệ số điều chỉnh - bias

**Transfer Function** (Hàm chuyển đổi): Hàm tổng của một nơron (Summation Function) cho biết khả năng kích hoạt của nơron đó còn gọi là kích hoạt bên trong. Các nơ-ron này có thể sinh ra một output hoặc không trong mạng ANN, nói cách khác rằng có thể output của 1 Nơron có thể được chuyển đến layer tiếp theo trong mạng Nơ-ron hoặc không. Mối quan hệ giữa hàm tổng và kết quả output được thể hiện bằng hàm chuyển đổi.Việc lựa chọn **hàm chuyển đổi** có tác động lớn đến kết quả đầu ra của mạng ANN. Hàm chuyển đổi phi tuyến được sử dụng phổ biến trong mạng ANN là hoặc *sigmoid* hoặc *tanh*.

Trong đó, hàm tanh là phiên bản thay đổi tỉ lệ của sigmoid , tức là khoảng giá trị đầu ra của hàm chuyển đổi thuộc khoảng [-1, 1] thay vì [0,1] nên chúng còn gọi là hàm chuẩn hóa (Normalize Function).

Kết quả xử lý tại các nơ-ron (Output) đôi khi rất lớn, vì vậy hàm chuyển đổi được sử dụng để xử lý output này trước khi chuyển đến layer tiếp theo.

Mạng nơ-ron của chúng ta dự đoán dựa trên lan truyền thẳng (forward propagation) là các phép nhân ma trận cùng với activation function để thu được kết quả đầu ra. Nếu input x là vector 2 chiều thì ta có thể tính kết quả dự đoán 𝑦̂ bằng công thức sau:

Trong đó, là input của layer thứ 𝑖, là output của layer thứ 𝑖 sau khi áp dụng activation function. , , , là các thông số (parameters) cần tìm của mô hình mạng nơ-ron.

Huấn luyện để tìm các thông số cho mô hình tương đương với việc tìm các thông số , , , sao cho độ lỗi của mô hình đạt được là thấp nhất. Ta gọi hàm độ lỗi của mô hình là loss function.

Nếu ta có N dòng dữ liệu huấn luyện, và C nhóm phân lớp (trường hợp này là ba lớp tích cực, tiêu cực và trung lập), khi đó loss function giữa giá trị dự đoán 𝑦̂ và 𝑦 được tính như sau:

Ý nghĩa công thức trên nghĩa là: lấy tổng trên toàn bộ tập huấn luyện và cộng dồn vào hàm loss nếu kết quả phân lớp sai. Độ lệch giữa hai giá trị 𝑦̂ và 𝑦 càng lớn thì độ lỗi càng cao. Mục tiêu của chúng ta là tối thiểu hóa hàm lỗi này. Ta có thể sử dụng phương pháp gradient descent để tối tiểu hóa hàm lỗi. Có hai loại gradient descent, một loại với learning rate cố định được gọi là batch gradient descent, loại còn lại có learning rate thay đổi theo quá trình huấn luyện được gọi là SGD (stochastic gradient descent) hay minibatch gradient descent.

Gradient descent cần các gradient là các vector có được bằng cách lấy đạo hàm của loss function theo từng thông số ,,,. Để tính các gradient này, ta sử dụng thuật toán backpropagation (lan truyền ngược). Đây là cách hiệu quả để tính gradient khởi điểm từ output layer.

Áp dụng backpropagation ta có các đại lượng

### **2.3. Mạng Nơron hồi quy RNN**

Ý tưởng của RNN đó là thiết kế một Neural Network sao cho có khả năng xử lý được thông tin dạng chuỗi (sequential information), ví dụ một câu là một chuỗi gồm nhiều từ.

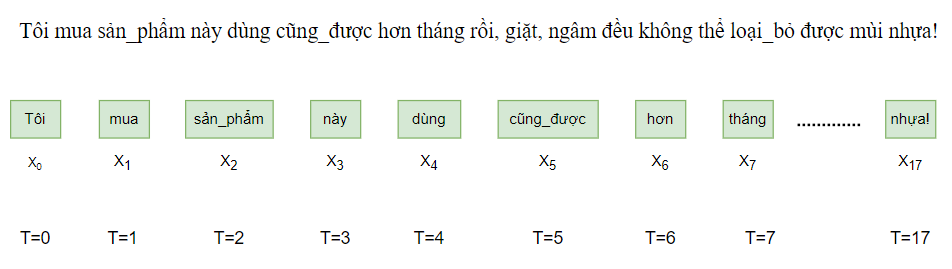
Recurrent có nghĩa là thực hiện lặp lại cùng một tác vụ cho mỗi thành phần trong chuỗi. Trong đó, kết quả đầu ra tại thời điểm hiện tại phụ thuộc vào kết quả tính toán của các thành phần ở những thời điểm trước đó.

Nói cách khác, RNN là một mô hình có trí nhớ (memory), có khả năng nhớ được thông tin đã tính toán trước đó. Không như các mô hình Neural Network truyền thống đó là thông tin đầu vào (input) hoàn toàn độc lập với thông tin đầu ra (output). Về lý thuyết, RNN có thể nhớ được thông tin của chuỗi có chiều dài bất kỳ, nhưng trong thực tế mô hình này chỉ nhớ được thông tin ở vài bước trước đó.



##### Hình 2.3.1: Mô hình mạng RNN. Nguồn: https://dominhhai.github.io/

Hình 2.3.1 là mô tả cơ bản của mạng RNN. Hàm A nhận đầu vào tại thời điểm t và đầu ra là giá trị vector ẩn . Nhận thấy, hàm A cho phép thông tin được lặp lại truyền từ một bước của mạng tới bước tiếp theo.



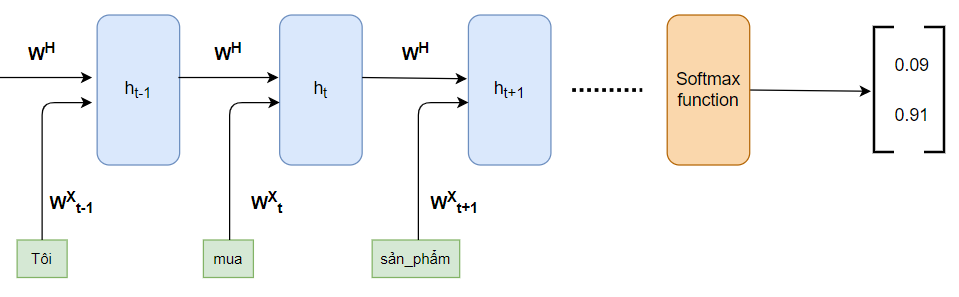
##### Hình 2.3.2: Ví dụ cách xử lý thông tin dạng chuỗi của RNN

Hình 2.3.2 cho thấy cách mô hình RNN xử lý một thông tin dạng chuỗi theo thời gian. Tại từng thời điểm t, các từ sẽ lần lượt được đưa vào mô hình. Tương ứng với mỗi mốc thời gian là một thành phần vector ẩn .

Hiểu một cách mô hình hóa, vector sẽ gói gọn và tóm tắt tất cả thông tin đã được đọc trong các bước thời gian trước đó. Trong khi đó, là vector đóng gói thông tin của một từ cụ thể được đưa vào mô hình RNN tại thời điểm t.

Ở đây, là vector mô tả từ “Tôi” được đưa vào mô hình tại thời điểm , là vector mô tả từ “mua” được đưa vào mô hình tại thời điểm . Vector trạng thái ẩn là một hàm của cả từ vựng hiện tại và vector trạng thái ẩn ở bước trước. Sigma là một hàm kích hoạt thường là một hàm sigmoid hoặc tanh.

và trong công thức là hai ma trận trọng số. Ma trận được sử dụng để nhân với vector đầu vào và ma trận trọng số nhân với vectơ trạng thái ẩn vào thời điểm trước đó. là một ma trận không thay đổi trong tất cả các bước thời gian trong khi đó là ma trận có giá trị thay đổi khác nhau cho mỗi đầu vào. Có thể thấy, giá trị của vector ẩn tại thời điểm t bị ảnh hưởng bởi giá trị của vector tại thời điểm hiện tại và giá trị của vector ẩn của trạng thái trước đó.



##### Hình 2.3.3 Mô hình trích xuất quan điểm cơ bản sử dụng RNN và softmax

Vector trạng thái ẩn tại bước cuối cùng được đưa vào hàm phân loại. Bước này thường được đặt tên là full connection. Trong đó, vector trạng thái ẩn ở bước cuối thường được nhân với mà một ma trận trọng số và đưa vào hàm softmax để đưa ra tương ứng các giá trị của lớp phân loại. Thông thường đối với bài toán trích xuất thông tin quan điểm thì ta sẽ xác định giá trị đầu ra của hàm softmax cho ba phân lớp tích cực, tiêu cực và trung lập.

,

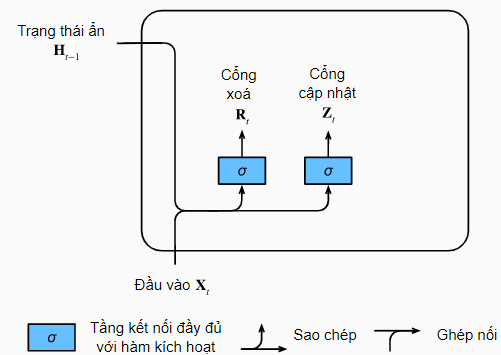
#### **2.3.1. Huấn luyện mạng**

Huấn luyện RNN tương tự như huấn luyện Neural Network truyền thống. Chúng ta cũng sử dụng đến thuật toán backpropagation (lan truyền ngược) nhưng có một chút tinh chỉnh. Đạo hàm tại mỗi đầu ra phụ thuộc không chỉ vào các tính toán tại bước đó, mà còn phụ thuộc vào các bước trước đó nữa, vì các tham số trong mạng RNN được sử dụng chung cho tất cả các bước trong mạng.

Ví dụ, để tính đạo hàm tại thời điểm t = 4, ta cần lan truyền ngược (backpropagation) 3 bước trước đó và cộng dồn các đạo hàm này lại với nhau. Kỹ thuật này gọi là Backpropagation Through Time (BPPTT). Điểm hạn chế ở đây đó là hidden layer không có trí nhớ dài hạn. Vấn đề này còn gọi là vấn đề vanishing/exploding gradient và như vậy, LSTM và GRU được sinh ra để giải quyết vấn đề này.

#### **2.3.2. Gated recurrent unit (GRU)**

GRU dùng để giải quyết vấn đề vanishing/exploding gradient của RNN tiêu chuẩn, GRU sử dụng **cổng cập nhật** (giúp mô hình xác định lượng thông tin trong quá khứ cần được chuyển đến tương lai) **và cổng xóa** (được sử dụng từ mô hình để quyết định có bao nhiêu thông tin quá khứ cần loại bỏ).



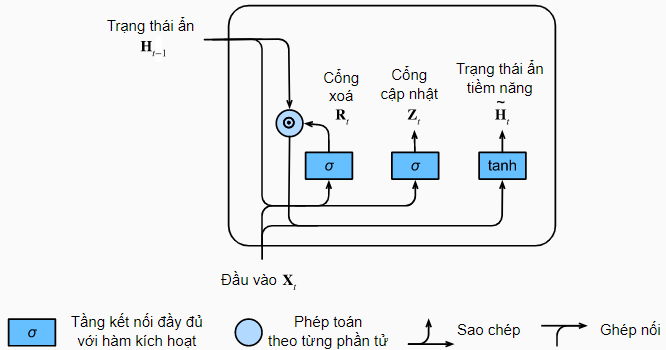
##### Hình 2.3.4 Cổng xóa và cổng cập nhật. Nguồn: d2l.aivivn.com

Tại thời điểm t, với đầu vào minibatch là (số lượng mẫu: n, số lượng đầu vào: d) và trạng thái ẩn ở thời điểm gần nhất là (số lượng trạng thái ẩn: h), cổng xóa và cổng cập nhật được tính như sau:

Ở đây, , và ,là các tham số trọng số và , là các hệ số điều chỉnh. Ta sẽ sử dụng hàm sigmoid để biến đổi các giá trị đầu vào trong khoảng (0,1).

##### **2.3.2.1. Hoạt động của Cổng Xóa**

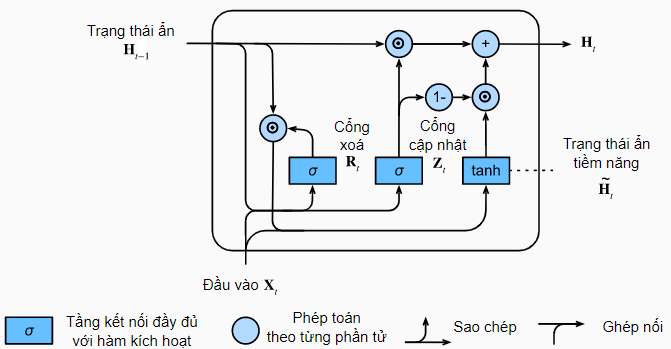
Ở đây chúng ta tích hợp cổng xóa với vectơ trạng thái ẩn để giảm ảnh hưởng của các trạng thái trước đó. Nhân với theo tửng phần tử. Nếu các phần tử trong cổng xóa có giá trị gần với 1 , kết quả sẽ giống RNN thông thường. Nếu tất cả các phần tử của cổng xóa gần với 0 , trạng thái ẩn sẽ là đầu ra của một perceptron đa tầng với đầu vào là . Và các vectơ trạng thái ẩn trước đó sẽ được đưa về lại giá trị mặc định. Nó được gọi là *trạng thái ẩn tiềm năng.*



##### Hình 2.3.5 Tính toán trạng thái ẩn tiềm năng trong một GRU. Nguồn: d2l.aivivn.com

##### **2.3.2.2. Hoạt động của Cổng Cập Nhật**

Cổng này xác định mức độ giống nhau giữa trạng thái mới và trạng thái cũ , cũng như là mức độ *trạng thái ẩn tiềm năng* được sử dụng. Cổng được sử dụng bằng cách áp dụng tổ hợp lồi giữa trạng thái cũ và *trạng thái ẩn tiềm năng.* Phương trình cập nhật:



##### Hình 2.3.6 Tính toán trạng thái ẩn trong GRU. Nguồn: d2l.aivivn.com

Nếu giá trị trong cổng cập nhật bằng 1 sẽ giữ lại trạng thái cũ. Có nghĩa là thông tin từ về cơ bản được bỏ qua, tương đương với việc bỏ qua bước thời gian t trong chuỗi phụ thuộc. Ngược lại, nếu gần giá trị 0, trạng thái ẩn sẽ gần với trạng thái ẩn tiềm năng .

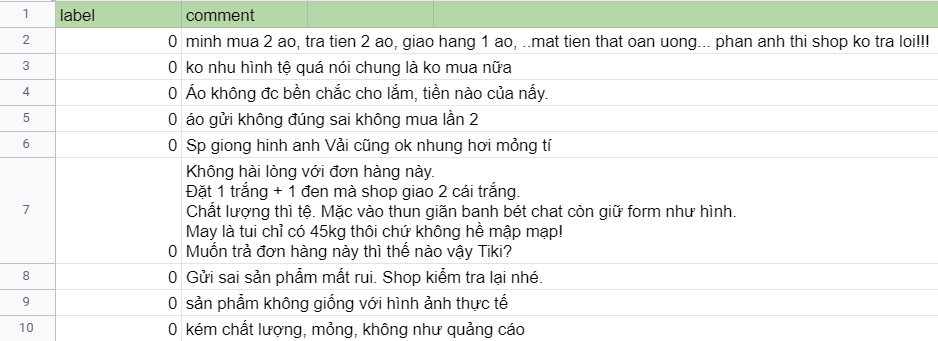
Với hai cổng trên đã giúp giải quyết quyết vấn đề tiêu biến gradient trong các mạng RNN và nắm bắt tốt hơn sự phụ thuộc xa trong chuỗi thời gian. Vậy tóm lại, GRU có hai tính chất nổi bật đó là:

* Cổng xóa giúp nắm bắt các phụ thuộc ngắn hạn trong chuỗi thời gian.
* Cổng cập nhật giúp nắm bắt các phụ thuộc dài hạn trong chuỗi thời gian.

### **2.4. RNN cho bài toán trích xuất quan điểm của bình luận**

#### **2.4.1. Tập dữ liệu**

* Bộ dữ liệu dùng để đào tạo bao gồm hơn 100.000 đánh giá của khách hàng từ các trang web thương mại điện tử về các danh mục: Quần áo, Giày dép, Túi xách, Hành lý, Đồng hồ và các phụ kiện thời trang khác.



##### Hình 2.4.1 Một số dòng trong dữ liệu đào tạo

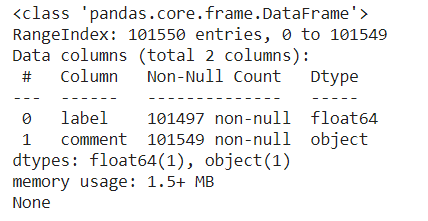
* Dữ liệu đào tạo đã được xử lý qua các bước sau:
  + Dữ liệu được thu thập thông tin bao gồm các bài đánh giá (văn bản) và xếp hạng (số nguyên 1-5 tương ứng với 1-5 sao xếp hạng).
  + Các nhãn lớp sau đó đã được gán lại theo cách thủ công.
  + Dữ liệu xếp hạng đã được chuyển đổi thành 3 nhãn lớp:

1-2 sao => lớp 0 (tiêu cực)

4-5 sao => lớp 1 (tích cực)

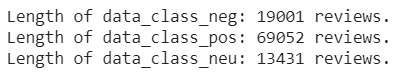
3 sao => lớp 2 (trung lập)

* Một số thống kê về dữ liệu:
  + Hình dưới cho biết về các thông tin của tập huấn luyện. Gồm hai cột, số lượng giá trị không rỗng và kiểu dữ liệu trong mỗi cột

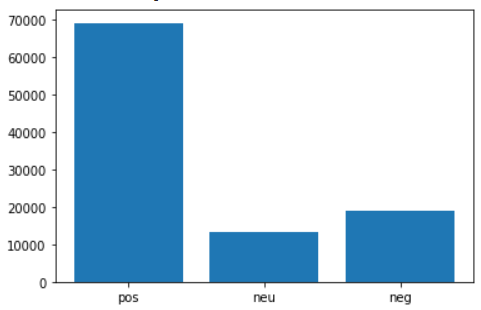


##### Hình 2.4.2 Thông tin về tập dữ liệu

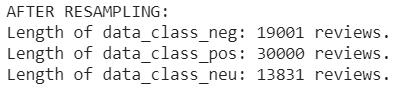
* + Thống kê số lượng các bình luận theo từng nhãn, thể hiện dưới dạng biểu đồ cho thấy được sự chênh lệch khá lớn giữa số lượng bình luận tích cực và bình luận tiêu cực, trung lập. Vì vậy ta cần phải có thêm bước chọn lại mẫu để cân bằng lại tập dữ liệu.



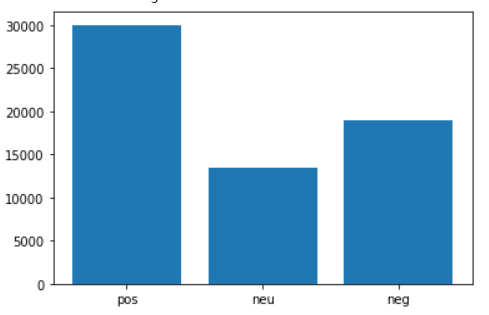
##### Hình 2.4.3 Số lượng các câu bình luận theo nhãn



##### Hình 2.4.4 Thống kê số bình luận được gán nhãn tích cực (pos), tiêu cực (neg), trung lập (neu)



##### Hình 2.4.5 Số lượng các bình luận theo nhãn sau khi chọn lại mẫu



##### Hình 2.4.6 Thống kê số lượng bình luận theo nhãn sau khi chọn lại mẫu

#### **2.4.2. Tiền xử lý dữ liệu dạng văn bản**

Dữ liệu văn bản của đồ án này là một dữ liệu chuỗi. Một câu bình luận có thể coi là một chuỗi các từ, hoặc một chuỗi các ký tự. Quá trình tiền xử lý thường bao gồm bốn bước sau:

**Bước 1:** Chuẩn hóa các từ viết hoa thành viết thường, xóa xuống dòng, thay thế chữ số và dấu chấm câu bằng dấu cách, loại bỏ những ký tự lặp, chuyển đổi bảng mã text về dạng Unicode.

**Bước 2:** Chia chuỗi thành các token trong đó một token có thể là một từ hoặc một ký tự. Với mỗi câu, chúng ta chia nó thành một danh sách các token. Một token là một điểm dữ liệu mà mô hình sẽ huấn luyện và đưa ra dự đoán từ nó. Dùng Moses Tokenizer để thực hiện việc này.

* Moses tokenizer (Koehn et al., 2007): là tokenizer đi kèm với bộ công cụ Moses. Nó tách dấu câu khỏi từ, lưu trữ các mã thông báo đặc biệt như URL hoặc ngày tháng, và chuẩn hóa các ký tự (ví dụ: dấu ngoặc kép các biến thể Unicode). Nó có thể được sử dụng với bất kỳ ngôn ngữ nào.
* Lý do chọn Moses tokenizer: Trong bài nghiên cứu “[How Much Does Tokenization Affect Neural Machine Translation?](https://www.catalyzex.com/paper/arxiv:1812.08621)" có thể thấy rằng Moses tokenizer cho kết quả tốt hơn các phương pháp còn lại.

**Bước 3:** Xây dựng một bộ từ vựng cho các token để ánh xạ chúng thành các chỉ số (index).

* Token kiểu chuỗi không phải là kiểu dữ liệu tiện lợi được sử dụng bởi các mô hình, thay vào đó chúng thường nhận dữ liệu đầu vào dưới dạng số. Bây giờ, chúng ta sẽ xây dựng một bộ từ điển, thường được gọi là bộ từ vựng (vocabulary), để ánh xạ chuỗi token thành các chỉ số bắt đầu từ 0. Để làm điều này, đầu tiên chúng ta lấy các token xuất hiện (không lặp lại) trong toàn bộ tài liệu, thường được gọi là kho ngữ liệu (corpus), và sau đó sắp xếp lại các token giảm dần dựa trên tần suất xuất hiện của chúng, và lấy vị trí của token trong từ vựng để làm chỉ số cho từ đó. Các token có tần suất xuất hiện rất ít thường được loại bỏ để giảm độ phức tạp. Một token chưa bao giờ xuất hiện trong kho ngữ liệu thì được gán chỉ số là 2. Trong bộ từ vựng, các chỉ số 0,1 và 2 là để thể hiện cho điểm đầu, cuối và token chưa biết.

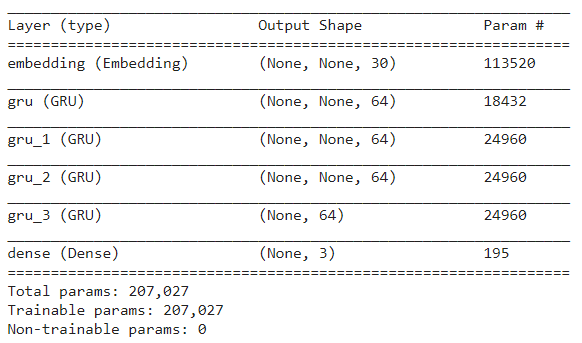
**Bước 4:** Ánh xạ tất cả các token trong dữ liệu văn bản thành các chỉ số để dễ dàng đưa vào các mô hình. Thêm các id 1 và 2 để đánh dấu bắt đầu và kết thúc câu. Ta sẽ thêm các số 0 vào cuối câu để các câu có độ dài bằng nhau.

#### 2.4.3. Xây dựng mô hình RNN

*Giới thiệu về Keras:*

Keras là một open source cho Neural Network được viết bởi ngôn ngữ Python. Nó là một library được phát triển vào năm 205 bởi Francois Chollet, là một kỹ sư nghiên cứu Deep Learning. Keras có thể sử dụng chung với các thư viện nổi tiếng như Tensorflow, CNTK, Theano. Trong phần xây dựng model dưới đây, chúng ta sử dụng model được Keras hỗ trợ đó là Sequence model.

* Model được xây dựng với 4 lớp GRU. Ở ba lớp GRU đầu chúng ta phải thiết lập tham số ‘return\_sequences=True’ để đảm bảo rằng lớp sau sẽ nhận được đúng dữ liệu đầu vào.
* Ngoài ra, sử dụng Embedding Layer là lớp đầu tiên của Model để tạo Word Embedding . Tham số truyền vào cho lớp này bao gồm vocab\_X\_size là kích thước của tập từ điển, embed\_size là kích thước của không gian vectơ mà các từ sẽ được nhúng vào. Nó xác định kích thước của các vectơ đầu ra từ lớp này cho mỗi từ.
* Embedding layer này được Keras cung cấp để có thể sử dụng cho mạng nơron trên tập dữ liệu văn bản. Ta cung cấp đầu vào cho nó là chỉ số của các từ trong câu. Lớp Embedding này sẽ được khởi tạo với các trọng số ngẫu nhiên và sẽ học các trọng số cho tập dữ liệu huấn luyện.



##### Hình 2.4.8 Thông tin của các lớp trong model

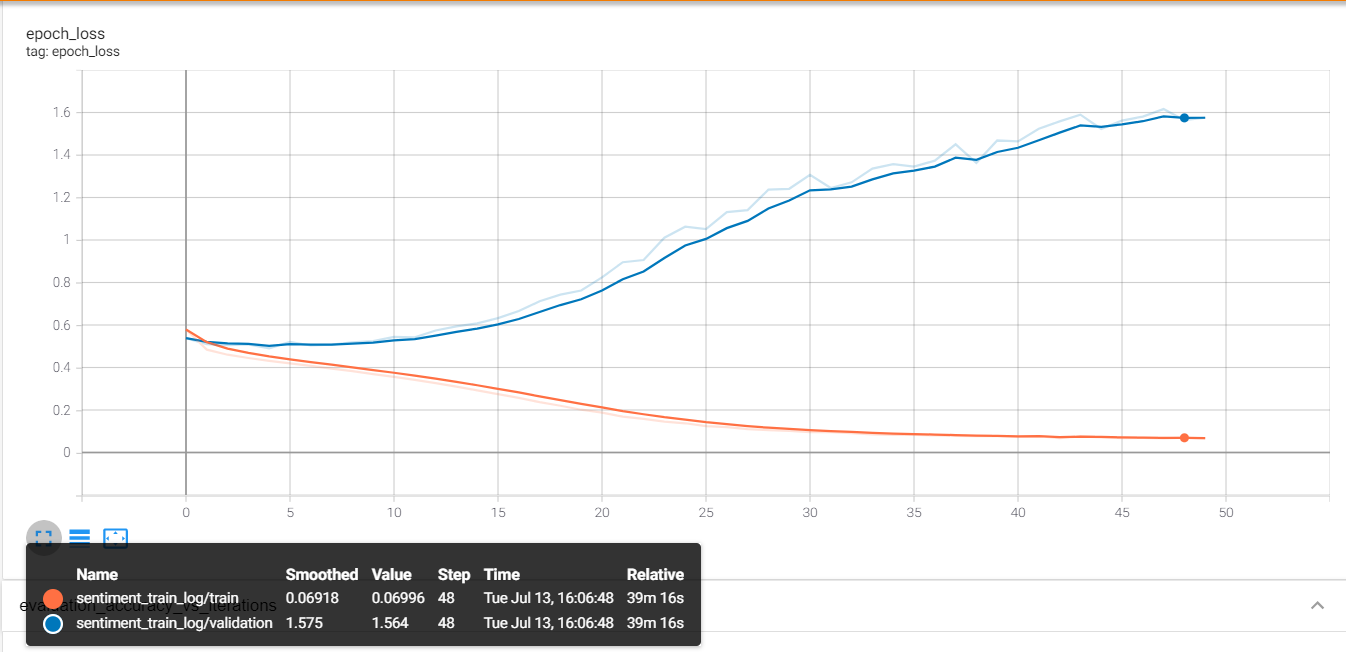
* 64 là số lượng nơron trong một lớp (Layer).
* Với lớp GRU cuối, ta thiết lập ‘return\_sequences=False’ tức là GRU chỉ trả về kết quả là của cell cuối cùng
* Dense( ): Layer này cũng như một layer neural network bình thường, với các tham số như: số class sau khi train (ở đây ta có 3 class là tích cực, tiêu cực và trung lập). Sử dụng hàm activation là softmax để để chuyển đổi giá trị thực trong các node ở output layer sang giá trị phần trăm.

#### **2.4.4. Huấn luyện mô hình**

* Sau khi định nghĩa được các layer cho model, trong Keras, ta cần compile model và định nghĩa hàm loss, optimizer và metrics.
* Để train model trong keras, chúng ta sử dụng model.fit(). Bao gồm tập data train được chia thành hai phần là data train và data validation để đưa vào huấn luyện. Tham số batch\_size thể hiện số lượng mẫu mà Mini-batch GD sử dụng cho mỗi lần cập nhật trọng số. Epoch là số lần duyệt qua hết số lượng mẫu trong tập huấn luyện.
* Hàm callbacks là một tập hợp các hàm được áp dụng ở các giai đoạn nhất định của quy trình huấn luyện. Có thể sử dụng callbacks để xem trạng thái bên trong và số liệu thống kê của mô hình trong quá trình đào tạo.



##### Hình 2.4.9 Các giá trị accuracy theo từng epoch của tập train và validation

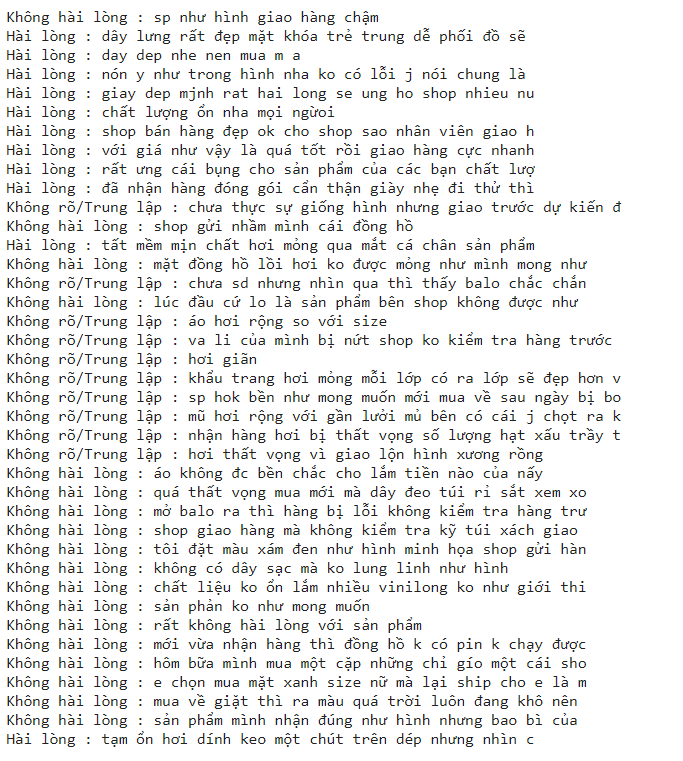


##### Hình 2.4.10 Các giá trị loss theo từng epoch của tập train và validation

* Chọn model của epoch 40 vì số accuracy cao nhất. Phần tiếp theo ta sẽ chạy thử model này trên data thật để xem độ chính xác của nó.

#### 2.4.5. Một số kết quả khi model dự đoán trên dữ liệu thật

Cho model dự đoán trên 40 câu bình luận mới, và nhận được các kết quả trả về như hình dưới. Ta có thể thấy model phân loại khá chính xác các quan điểm của các bình luận đã cho.



##### Hình 2.4.11 Một số kết quả khi chạy model trên tập dữ liệu thật

## **CHƯƠNG 3: THIẾT KẾ ỨNG DỤNG WEBSITE HIỂN THỊ SẢN PHẨM**

### **3.1. Phân tích yêu cầu**

* Biểu đồ giá sản phẩm theo thời gian thực - bắt đầu từ tháng 3 đến nay. Dựa trên biểu đồ này người mua có thể tự so sánh giá của sản phẩm xem có phải thực sự sản phẩm này đang giảm giá và thời điểm giá thấp nhất của sản phẩm.
* Biểu đồ đánh giá sự hài lòng thực sự của những người mua trước. Biểu đồ dựa trên dữ liệu được đào tạo trước đó ở chương hai.
* Hiển thị các thông tin cơ bản của sản phẩm như tên, giá, thông số, chất liệu sản phẩm và hình ảnh thực tế của sản phẩm.
* Tìm kiếm sản phẩm dựa trên các đặc điểm và thông tin của sản phẩm
* Phân loại sản phẩm mà nhóm đã cào dữ liệu. Có 4 loại sản phẩm chính là: Giày dép nam, giày dép nữ, quần áo nam, quần áo nữ.

### **3.2. Mô tả chi tiết các kiểu dữ liệu**

**Bảng 3.2.1 Mô tả chi tiết kiểu dữ liệu**

|  |  |  |
| --- | --- | --- |
| **Tên thuộc tính** | **Kiểu dữ liệu** | **Mô tả chi tiết** |
| catid | Number | Mã id của loại sản phẩm |
| itemid | Number | Mã id của sản phẩm |
| name | String | Tên sản phẩm |
| images | String | Link chuỗi hình ảnh của sản phẩm |
| description | String | Mô tả sản phẩm |
| currency | String | Mệnh giá tiền theo VND |
| image | String | Link hình ảnh đại diện của sản phẩm |
| historical\_sold | Number | Số lượng bán được của sản phẩm |
| url | String | Đường dẫn để đến trang thương mại điện tử Shopee |
| price | Number | Giá của sản phẩm |
| DayUpdate | Date | Ngày cập nhật sản phẩm gần nhất |

### **3.3. Sơ đồ và quy trình ứng dụng**

##### Hình 3.3.1 Sơ đồ quy trình ứng dụng

Người sử dụng website khi bắt đầu vào ứng dụng sẽ hiện các tính năng ở trang chủ bao gồm:

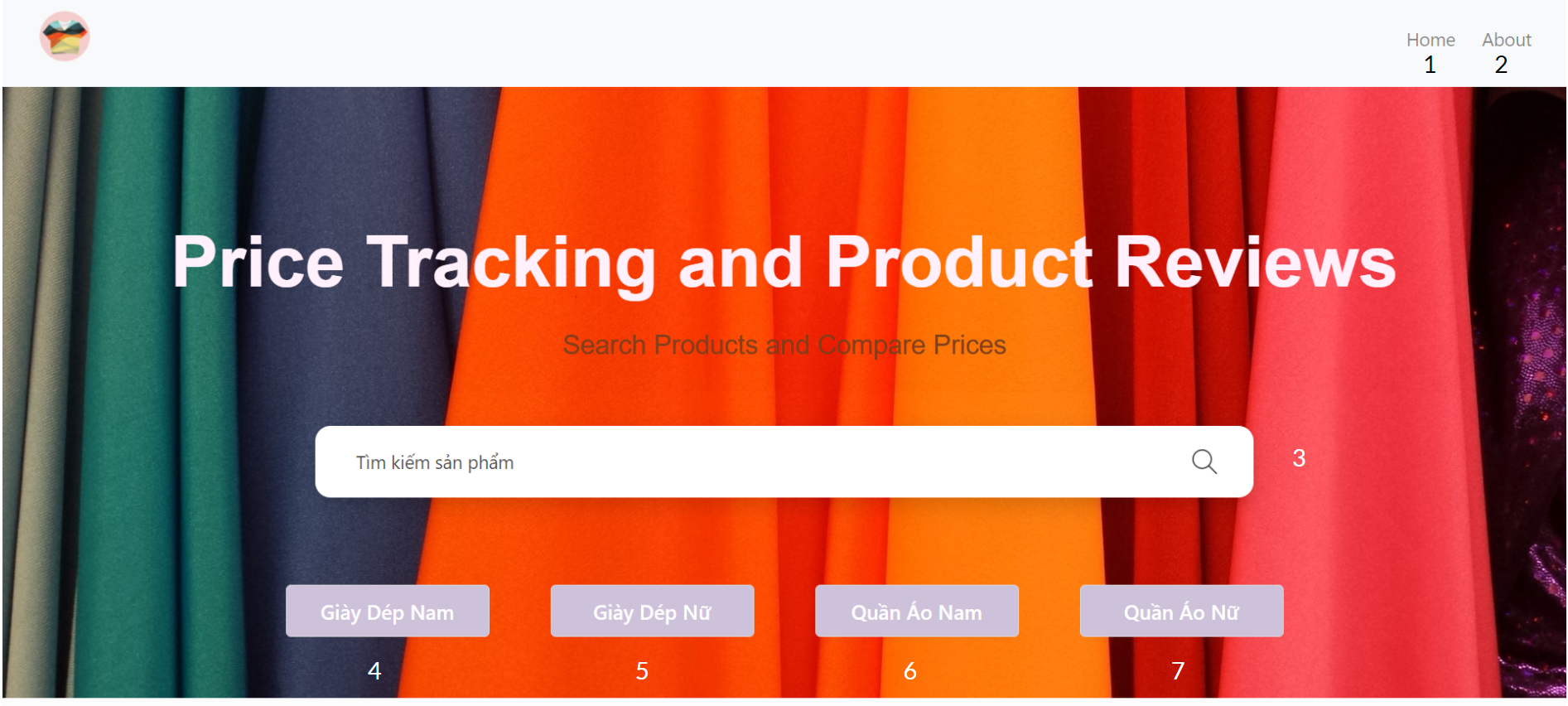
* Hộp tìm kiếm sản phẩm - chuyên tìm kiếm các sản phẩm dựa trên các thông tin đặc trưng của sản phẩm mà người tìm kiếm cung cấp.
* Loại sản phẩm gồm: Áo quần nam, áo quần nữ, giày dép nam, giày dép nữ. Khi người dùng chỉ muốn xem chuyên một thể loại hàng hóa, người dùng có thể nhấn vào các biểu nút loại sản phẩm.

Sau khi chọn các tính năng tùy theo nhu cầu người dùng, giao diện đều sẽ dẫn đến trang hiện các thông tin sản phẩm như: tên, giá và hình ảnh của sản phẩm. Người dùng muốn xem kỹ hơn về chi tiết của sản phẩm thì sẽ phải chọn vào sản phẩm ấy. Lúc đó thông tin chi tiết của sản phẩm sẽ hiện lên bao gồm:

* Tên sản phẩm
* Giá sản phẩm
* Hình ảnh sản phẩm
* Số lượng sản phẩm được bán
* Ngày cập nhật sản phẩm
* Button dẫn link đến trang web/sàn thương mại điện tử Shopee đang bán sản phẩm này để khách hàng có thể mua sản phẩm.
* Các biểu đồ về giá của sản phẩm qua thời gian và biểu đồ sự hài lòng của khách hàng/chất lượng của sản phẩm.
* Các mô tả chi tiết của sản phẩm. Phần thông tin này được cào từ phần mô tả sản phẩm trên chính cửa hàng bán hàng.

### **3.4. Thiết kế giao diện**

* Giao diện trang chủ:

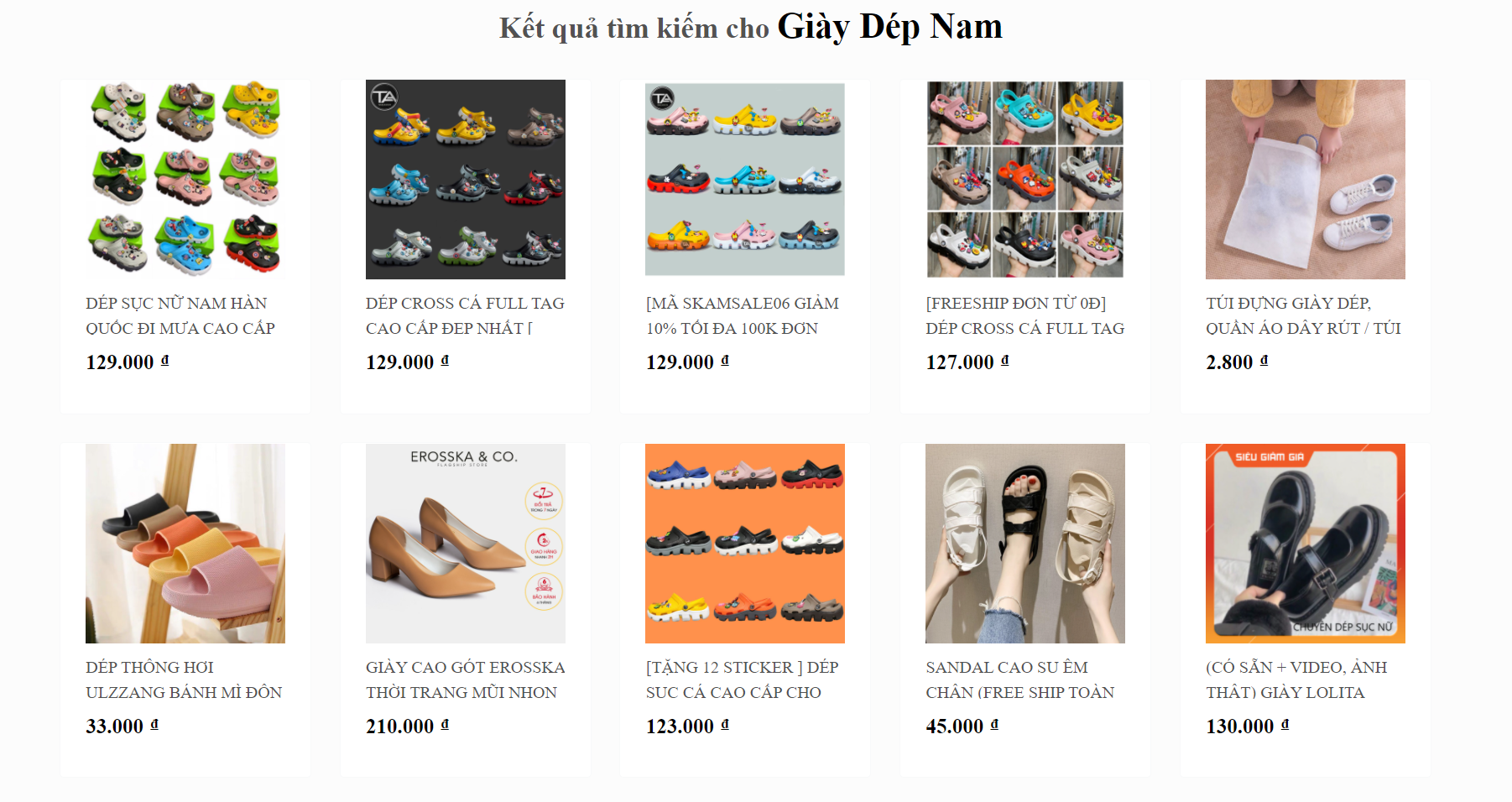


##### Hình 3.4.1 Giao diện tìm kiếm và lựa chọn loại sản phẩm

Giao diện đầu tiên khi truy ứng dụng sẽ là giao diện về tìm kiếm và các loại sản phẩm mặc định:

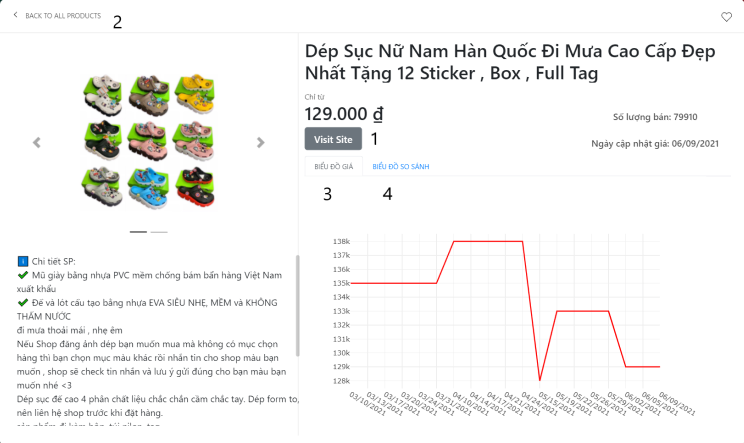
1. Home: khi bấm vào đây sẽ quay lại giao diện trang chủ như hình 3.4.1
2. About: Ở đây có đường dẫn đến trang Github có chứa code lập trình nên website này.
3. Hộp tìm kiếm: Dùng để tìm kiếm các mặt hàng dựa trên từ khóa mà người mua nhập.
4. Giày Dép Nam: Khi nhấn vào đây sẽ chuyển người mua đến giao diện sản phẩm toàn giày dép dành cho nam.
5. Giày Dép Nữ: Khi nhấn vào đây sẽ chuyển người mua đến giao diện sản phẩm toàn giày dép dành cho nữ.
6. Quần Áo Nam: Khi nhấn vào đây sẽ chuyển người mua đến giao diện sản phẩm toàn quần áo dành cho nam.
7. Quần Áo Nữ: Khi nhấn vào đây sẽ chuyển người mua đến giao diện sản phẩm toàn quần áo dành cho nữ.

* Giao diện thông tin sản phẩm:



##### Hình 3.4.2 Giao diện trình bày sản phẩm

Ở giao diện này sẽ có hình ảnh, tên và giá của các sản phẩm. Chỉ cần khách hàng muốn xem sản phẩm nào thì click chuột vào ô sản phẩm đó, khách hàng sẽ được chuyển đến giao diện mô tả chi tiết thông tin sản phẩm như hình 3.4.3. Với giao diện này sẽ có tên, giá bán, chuỗi hình ảnh thực của mặt hàng, nút bấm dẫn trực tiếp người mua đến nơi sản phẩm được bày bán trên sàn Shopee, số lượng sản phẩm bán được, ngày cập nhật lại sản phẩm.



##### Hình 3.4.3 Biểu đồ giá của sản phẩm từ tháng 3 đến nay

1. Visit Site: Nút bấm đưa người mua tới sản phẩm đó trên sàn thương mại Shopee
2. BACK TO ALL PRODUCTS: Quay lại giao diện thông tin sản phẩm như hình 3.4.2

* Giao diện biểu đồ:



##### Hình 3.4.4 Biểu đánh giá sự hài lòng của người mua

Như hình 3.4.3 và 3.4.4 thì biểu đồ được sử dụng chung với một phần giao diện thông tin sản phẩm nên sẽ sử dụng chung tính năng số “1” và “2”. Còn tính năng “3” và “4” sẽ có tác dụng:

1. BIỂU ĐỒ GIÁ: Biểu đồ trực quan giá bán của sản phẩm qua những ngày được cập nhật - bắt đầu từ tháng 3 đến nay.
2. BIỂU ĐỒ SO SÁNH: So sánh giữa rating của Shopee và dự đoán của mô hình, cũng như so sánh mức độ hài lòng của khách hàng mua sản phẩm thông qua bình luận.

### **3.5. Thiết kế phần mềm**

#### 3.5.1. Giới thiệu ngăn xếp MERN

MERN stack là bộ các công nghệ mã nguồn mở đều liên quan đến Javascript: MongoDB, ExpressJS, React, NodeJS được sử dụng để xây dựng các ứng dụng nâng cao. Bộ công cụ trên cung cấp toàn bộ các thành phần phát triển cả ở Frontend và Backend.

JavaScript là một ngôn ngữ lập trình đa nền tảng, ngôn ngữ lập trình kịch bản, hướng đối tượng. JavaScript là một ngôn ngữ nhỏ và nhẹ. Nó có thể kết nối tới các object khi nằm trong một môi trường, và cung cấp cách quản lý các object đó.

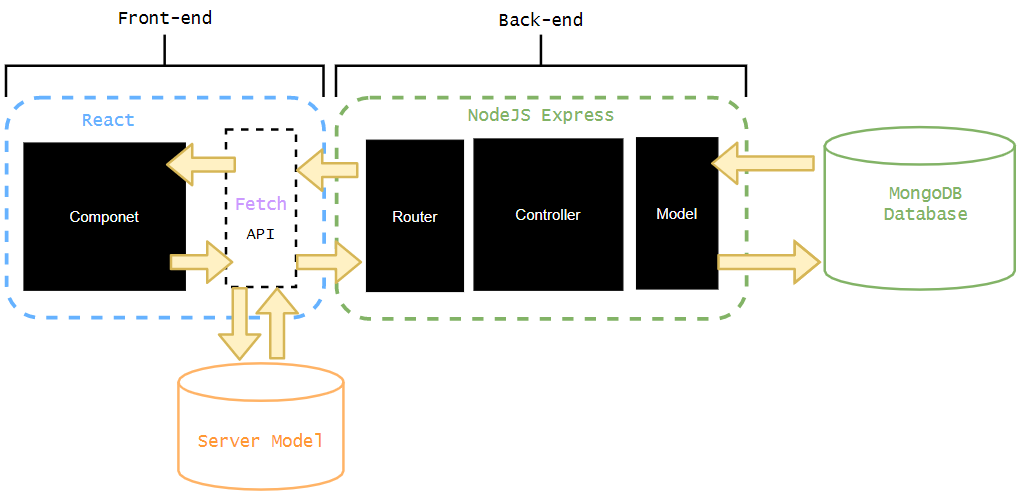
Một số ưu điểm của JavaScript:

* Không cần một compiler vì web browser có thể biên dịch nó bằng HTML
* Nó dễ học hơn các ngôn ngữ lập trình khác
* Lỗi dễ phát hiện hơn vì vậy dễ sửa hơn
* Nó có thể được gắn trên một số element của trang web hoặc event của trang web như là thông qua click chuột hoặc di chuột tới
* JS hoạt động trên nhiều trình duyệt, nền tảng
* Có thể sử dụng JavaScript để kiểm tra input và giảm thiểu việc kiểm tra thủ công khi truy xuất qua database
* Nó giúp website tương tác tốt hơn với khách truy cập
* Nó nhanh hơn và nhẹ hơn các ngôn ngữ lập trình khác

NodeJS có thể chạy trên nhiều nền tảng hệ điều hành khác nhau từ Windows cho tới Linux, OS X nên đó cũng là một lợi thế. Bên cạnh đó NodeJS còn cung cấp phong phú các thư viện ở dạng Javascript Module khác nhau giúp đơn giản hóa việc lập trình và giảm thời gian ở mức thấp nhất. Phần Core bên dưới của Nodejs được viết hầu hết bằng C++ nên cho tốc độ xử lý và hiệu năng khá cao.

ExpressJS là một Framework nhỏ nhưng linh hoạt, được xây dựng trên nền tảng của Nodejs. Nó cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile.

Thư viện JavaScript phổ biến nhất để xây dựng giao diện người dùng là React. Nó được phát triển vào năm 2013 bởi Facebook, hỗ trợ việc xây dựng những thành phần (components) UI có tính tương tác cao, có trạng thái và có thể sử dụng lại được.



##### Hình 3.5.1 Kiến trúc ngăn xếp MERN

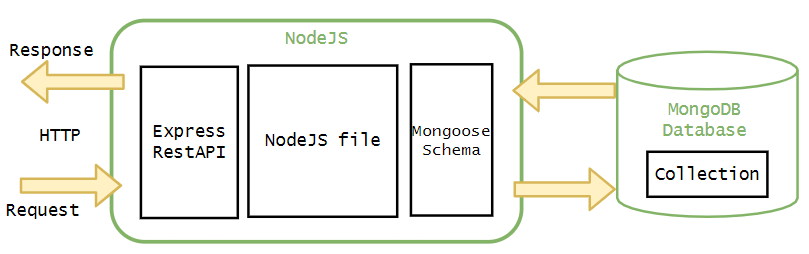
#### 3.5.2. Cấu trúc website

Như hình 3.5.1 thì cấu trúc của ứng dụng sẽ có:

* Phần thiết kế giao diện sử dụng công nghệ ReactJS
* Website API được xây dựng dựa trên nền tảng NodeJS sử dụng framework là Express
* Dữ liệu được lưu dưới dạng JSON trong MongoDB

**Back-end**:

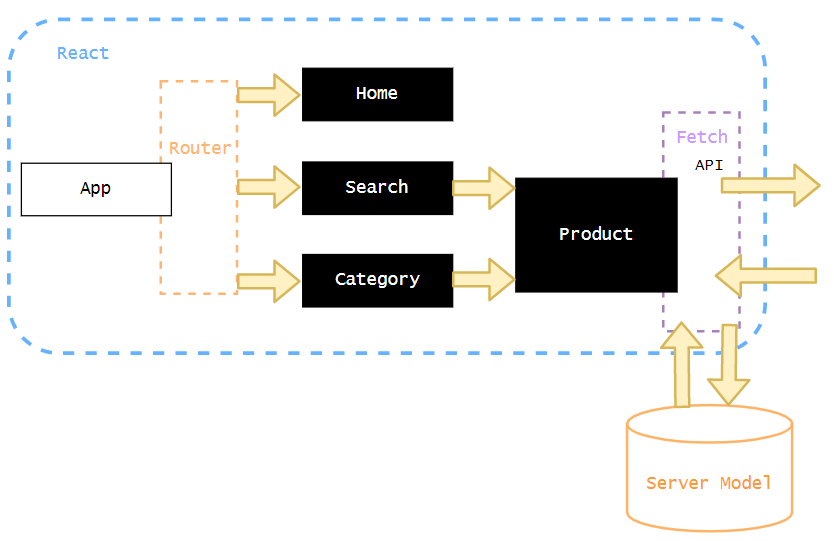
* Cấu trúc của phần back-end bao gồm các file(thư mục) xác định ***model***, ***controller*** và ***routers***.
* Node.js Express kết nối với cơ sở dữ liệu MongoDB bằng Mongoose ODM. Ta sẽ tạo Schema trong file ***model*** để định dạng dữ liệu mà ta mong muốn lấy từ database. Cụ thể trong ứng dụng này, chúng ta lấy dữ liệu từ collection ***products****.* Để việc tìm kiếm theo tên sản phẩm sau này dễ dàng hơn, ta cần chia index cho trường ***name*** (tên sản phẩm) trong Schema.
* Trong file ***routers*** ta thiết kế các router để tạo và điều hướng API cho các controller xử lý các request.
* Trong file ***controller*** sẽ nhận các request, xử lý và trả dữ liệu về cho các hành động như tìm kiếm sản phẩm hay hiển thị sản phẩm của từng danh mục.



##### Hình 3.5.2 Sơ đồ cấu trúc back-end NodeJS

**Front-end:**

* Cấu trúc front-end cơ bản bao gồm các **Component** - là những thành phần nhỏ cấu thành một ứng dụng mà ở đó chúng được tách riêng biệt và có thể sử dụng lại. Thành phần này chính là các View (phần hiển thị) để người dùng có thể hiểu hoặc tương tác.
* File **App** là một *header component* dùng chung. Nó được dùng để điều hướng tới các trang khác.
* App chia các router cho từng hành động khác nhau. Có ba hoạt động chính là **Home**, **Search**, **Category**. Nếu tương tác với một trong ba ta sẽ được dẫn tới các component chứa các hàm trả về HTML phù hợp với hành động đó.
* **Home** sẽ trả về giao diện trang chủ.
* **Search** và **Category** sẽ trả về component Product thể hiện các sản phẩm khác nhau.
* Với Product component, ở đây nó sẽ sử dụng hàm *Fetch* với API của Nodejs đã tạo để lấy dữ liệu về các sản phẩm theo từ khóa tìm kiếm (**Search**) hay các danh mục (**Category**) được chọn.
* Sau đó, các dữ liệu này được truyền tới các component nhỏ khác là MiniCard để thể hiện các sản phẩm khác nhau. (Như hình 3.4.2)
* Mỗi MiniCard sẽ dẫn tới component ProductCard để thể hiện các thông tin của sản sản phẩm (Như hình 3.4.3).
* Ngoài ra ở MiniCard cũng sử dụng hàm *Fetch* để lấy dữ liệu từ Server Model. Các dữ liệu này là các bình luận được phân loại bởi model trên chương 2, dùng để tạo ra biểu đồ so sánh các đánh giá. (Như hình



##### Hình 3.5.3 Sơ đồ cấu trúc front-end

**Database**: NoSQL - MongoDB

# 

# 

# 

# 

# 

# 

# 

# 

# **PHẦN 3: KẾT LUẬN**

## **1. Kết quả đạt được**

* Hiểu được các mô hình Crawl Data Api Public và Fast Api
* Tìm hiểu cơ bản về NoSQL và MongoDB
* Tìm hiểu về công nghệ Node JS và React JS
* Tìm hiểu phương pháp xử lý ngôn ngữ tự nhiên và thuật toán Recurrent Neural Network (RNN)

## **2. Đóng góp của đề tài**

Thực tế hiện nay có rất nhiều sản phẩm được bày bán trên các sàn thương mại điện tử với các mức giá biến động và thông số khác nhau. Vì vậy, với đề tài: **“Phát triển ứng dụng theo dõi giá bán và đánh giá hàng hóa online”** – đề tài này sẽ hỗ trợ khách hàng có thể tìm kiếm được mặt hàng chất lượng và cũng như giúp cho khách hàng canh được thời điểm mua sản phẩm có giá tốt nhất. Xử lý nhanh một khối lượng lớn về thông tin, tra cứu tìm kiếm một cách nhanh chóng với độ chính xác cao mà không cần so sánh hay bị mua hàng giả, hàng kém chất lượng, tiết kiệm được nhiều thời gian và công sức của người mua.

## **3. Hạn chế**

Do mới tiếp xúc với công nghệ như NodeJS, ReactJS, Crawl Data Api Public, Fast Api cũng như lĩnh vực học sâu và phương pháp xử lý ngôn ngữ tự nhiên nên khó tránh khỏi có lỗi xảy ra. Trang web chưa sinh động, còn đơn giản giới hạn nhiều tính năng, tính ứng dụng còn nhiều điều hạn chế. Thời gian tìm kiếm còn lâu, đánh giá bình luận còn chưa hoàn toàn chính xác.

**4. Hướng phát triển**

* Cải tiến và hoàn thiện giao diện bắt mắt hơn.
* Đưa ra các giải pháp tốt hơn.
* Cập nhật nhiều chức năng hơn.
* Code tối ưu hơn.
* Áp dụng các thuật toán để tìm kiếm sản phẩm nhanh và chính xác hơn.
* Cải tiến và đào tạo mô hình học sâu để ra kết quả chính xác hơn nữa.

# DANH MỤC TÀI LIỆU THAM KHẢO

**Uông Thị Thanh Thủy, Trần Gia Bảo, Hoàng Thị Cẩm Tú:** Tiểu luận chuyên ngành kỹ thuật dữ liệu “Tìm hiểu Web Scraping - Xây dựng Website so sánh giá hàng hóa”

[1]<https://topdev.vn/blog/api-la-gi/>

[2]<https://topdev.vn/blog/restful-api-la-gi/>

[3]h[ttps://viblo.asia/p/huong-dan-co-ban-framework-fastapi-tu-a-z-phan-1-V3m5W0oyKO7](https://viblo.asia/p/huong-dan-co-ban-framework-fastapi-tu-a-z-phan-1-V3m5W0oyKO7)

[4] <https://fastapi.tiangolo.com/>

[5]<http://lib.uet.vnu.edu.vn/bitstream/123456789/868/1/HTTT_NhuBaoVu_K21_Luan%20Van%20Thac%20si.pdf>

[6]<http://lib.uet.vnu.edu.vn/bitstream/123456789/868/1/HTTT_NhuBaoVu_K21_Luan%20Van%20Thac%20si.pdf>

[7]<http://lib.uet.vnu.edu.vn/bitstream/123456789/868/1/HTTT_NhuBaoVu_K21_Luan%20Van%20Thac%20si.pdf>

[8]<http://lib.uet.vnu.edu.vn/bitstream/123456789/868/1/HTTT_NhuBaoVu_K21_Luan%20Van%20Thac%20si.pdf>

[9] <https://d2l.aivivn.com/chapter_recurrent-modern/gru_vn.html>

[10] <https://www.catalyzex.com/paper/arxiv:1812.08621>

[11]<https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/>