
LIFESTORE

SALES REPORT



EDUARDO ALCALÁ HUERTA

01 Resume

02 Data Analysis and
Opportunity Areas

03 Code

04 Final Thoughts

01 RESUME

760,177
K

**TOTAL
REVENUE**

755,968
K

REVENUE
THIS YEAR

26,949
K

REVENUE
THIS MONTH

OUR PERFORMANCE

In recent months during the covid-19 pandemic there has been a downward trend in product sales, lagged products and accumulated inventory



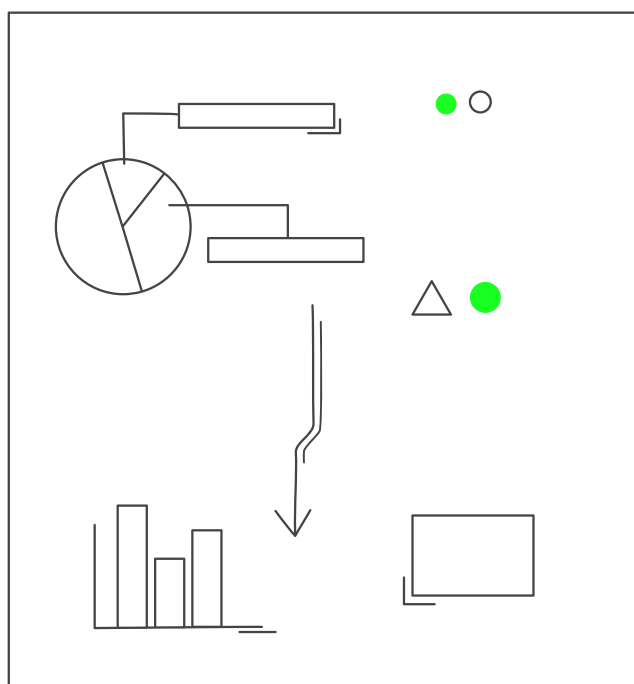
BI REPORT

This report was created in order to propose a growth strategy and attack areas of opportunity of the company

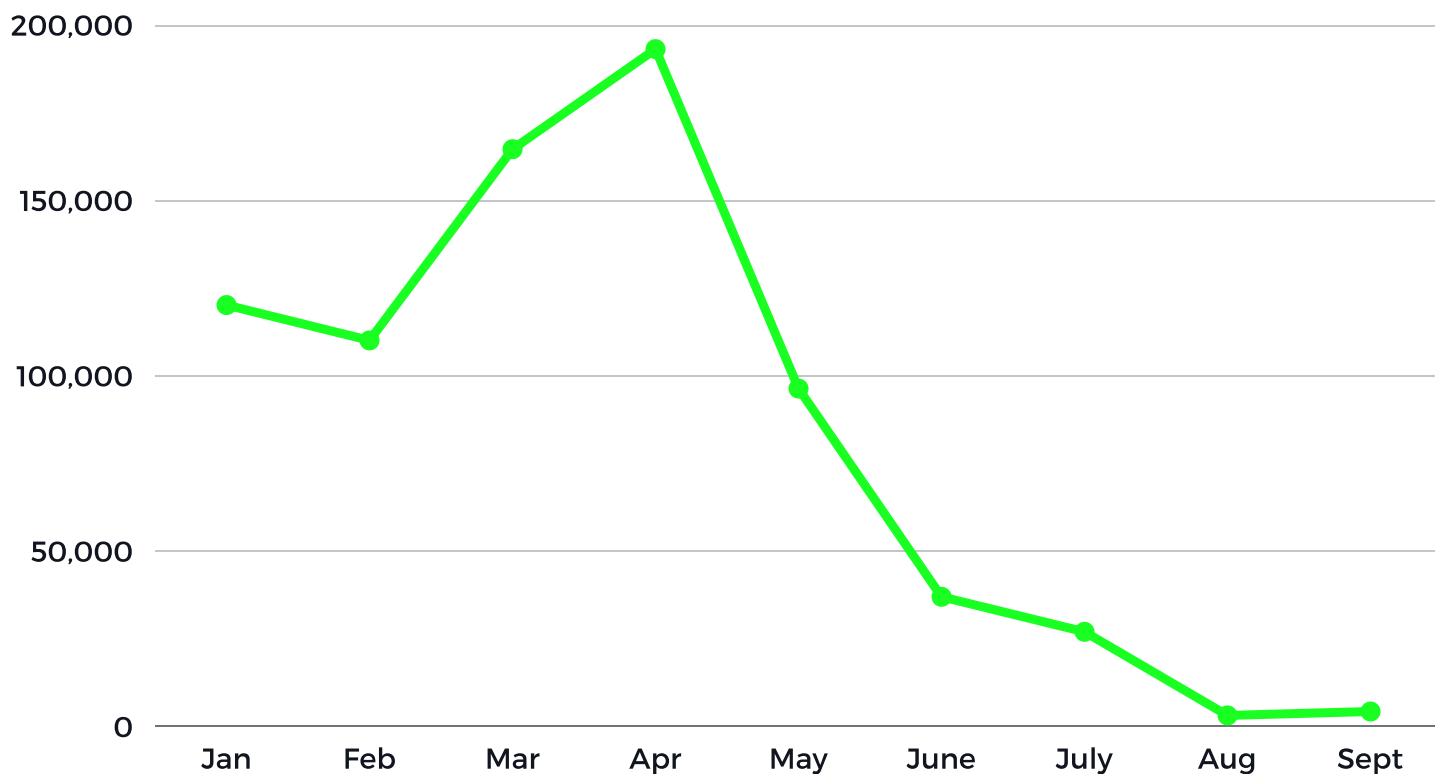
OUR STATS

Key performance indicators (KPIs), both financial and non-financial, are an important component.

Let's know some of the most important.



02 ANALYSIS



MONTHLY SALES

The forced closure of most shops by covid-19 has wiped out sales in the sector. The retail trade registered a historical decrease in its turnover of 14.3% in March compared to the same month of 2019, This has greatly affected our sales and expectations this year.

After an analysis of the sales information, the following metrics were evaluated.

Best Selling Products

This list of products comprises the largest volume of sales, so it is advisable to give them greater exposure and restocking

BEST SELLER	SALES	PRODUCT
1	50	SSD KINGSTON A400, 120GB, SATA III, 2.5', 7MM
2	42	PROCESADOR AMD RYZEN 5 2600, S-AM4, 3.40GHZ
3	20	PROCESADOR INTEL CORE I3-9100F, S-1151, 3.60GHZ
4	18	TARJETA MADRE ASROCK MICRO ATX B450M STEEL LEGEND
5	15	SSD ADATA ULTIMATE SU800, 256GB, SATA III
6	14	TARJETA MADRE ASUS MICRO ATX TUF B450M-PLUS GAMING
7	13	PROCESADOR AMD RYZEN 5 3600, S-AM4, 3.60GHZ
8	13	PROCESADOR AMD RYZEN 3 3200G CON GRÁFICOS RADEON VEGA 8
9	13	SSD XPG SX8200 PRO, 256GB, PCI EXPRESS, M.2
10	11	TARJETA DE VIDEO ASUS NVIDIA GEFORCE GTX 1660

Most Searched Products

This products attract the attention of the user, it would be a good idea to offer them at a lower price to increase sales

MOST SEARCHED	SEARCES	PRODUCT
1	263	SSD KINGSTON A400, 120GB, SATA III, 2.5', 7MM
2	107	SSD ADATA ULTIMATE SU800, 256GB, SATA III
3	55	TARJETA MADRE ASUS MICRO ATX TUF B450M-PLUS GAMING
4	41	PROCESADOR AMD RYZEN 5 2600, S-AM4, 3.40GHZ
5	35	LOGITECH AUDÍFONOS GAMER G635 7.1, ALÁMBRICO,
6	32	TV MONITOR LED 24TL520S-PU 24, HD, WIDESCREEEN
7	31	PROCESADOR AMD RYZEN 5 3600, S-AM4, 3.60GHZ
8	30	PROCESADOR INTEL CORE I3-9100F, S-1151, 3.60GHZ
9	30	SSD XPG SX8200 PRO, 256GB, PCI EXPRESS, M.2
10	27	SSD KINGSTON A2000 NVME, 1TB, PCI EXPRESS 3.0

Lowest Sales by Category

Swapping out products is not particularly difficult for dropshippers, but if we create our own goods then that's a bigger issue.

In that case, it may be better to find a way to pivot to another related niche.

SALES	CATEGORY	PRODUCT
1	AUDIFONOS	HYPERX AUDÍFONOS GAMER CLOUD FLIGHT PARA PC/PS4/PS4 PRO
1	AUDIFONOS	COUGAR AUDÍFONOS GAMER PHONTUM ESSENTIAL, ALÁMBRICO
1	AUDIFONOS	LOGITECH AUDÍFONOS GAMER G332, ALÁMBRICO
1	PANTALLAS	TV MONITOR LED 24TL520S-PU 24, HD, WIDESCREEEN
1	PANTALLAS	TCL SMART TV LED 55S425 54.6, 4K ULTRA HD
1	MEMORIAS USB	KIT MEMORIA RAM CORSAIR DOMINATOR PLATINUM DDR4
1	DISCOS DUROS	SSD CRUCIAL MX500, 1TB, SATA III
1	TARJETAS MADRE	TARJETA MADRE GIGABYTE MICRO ATX GA-H110M-DS2

Lowest Searches by Category

Users trust recommendations to buy products from a search, it is important to take an SEO approach to keywords and refine them, they must be visible, useful and understandable.

SEARCHES	CATEGORY	PRODUCT
1	AUDIFONOS	GINGA AUDÍFONOS CON MICRÓFONO GI18ADJ01BT-RO
1	BOCINAS	GHIA BOCINA PORTÁTIL BX800, BLUETOOTH, INALÁMBRICO
1	PANTALLAS	SAMSUNG SMART TV LED 43, FULL HD
1	DISCOS DUROS	SSD SAMSUNG 860 EVO, 1TB, SATA III
1	TARJETAS MADRE	TARJETA MADRE ASROCK ATX H110 PRO BTC+
1	TARJETAS MADRE	TARJETA MADRE GIGABYTE MICRO ATX Z390 M GAMING
1	TARJETAS DE VIDEO	TARJETA DE VIDEO VISIONTEK AMD RADEON HD5450
1	TARJETAS DE VIDEO	MSI GEFORCE 210, 1GB GDDR3, DVI, VGA

Best product reviews

The user trusts the products and services when they are recommended by people. Benefits can be offered to customers if they recommend us to their friends and acquaintances, generating referral marketing.

BEST REVIEWED	SCORE	PRODUCT
1	4	SSD KINGSTON A400, 120GB, SATA III, 2.5', 7MM
2	5	PROCESADOR AMD RYZEN 5 2600, S-AM4, 3.40GHZ
3	4	PROCESADOR INTEL CORE I3-9100F, S-1151, 3.60GHZ
4	5	TARJETA MADRE ASROCK MICRO ATX B450M STEEL LEGEND
5	5	LOGITECH AUDÍFONOS GAMER G635 7.1, ALÁMBRICO,
6	4	SSD ADATA ULTIMATE SU800, 256GB, SATA III
7	5	TARJETA MADRE ASUS MICRO ATX TUF B450M-PLUS GAMING
8	4	PROCESADOR AMD RYZEN 5 3600, S-AM4, 3.60GHZ
9	4	PROCESADOR AMD RYZEN 3 3200G CON GRÁFICOS RADEON VEGA 8

Worst product reviews

The offer must be improved in some way:

Offer more warranty, more and better customer service options, product improvements

In general we can change elements of the product that make it more desirable.

WORST REVIEWED	SCORE	PRODUCT
1	1	TARJETA MADRE AORUS MICRO ATX B450 AORUS
2	1	TARJETA DE VIDEO GIGABYTE AMD RADEON R7
3	1	TARJETA MADRE ASROCK ATX H110 PRO BTC+
4	2	TARJETA MADRE GIGABYTE MICRO ATX GA



760 K

Total revenue

Total income is all the income received by a company during a period of time. It is calculated multiplying the quantity sold by the price.

30%

The year 2020 and subsequent years will undoubtedly bring their own challenges and business opportunities. We are moving forward from a solid strategic position



756 K

Revenue this year

40%

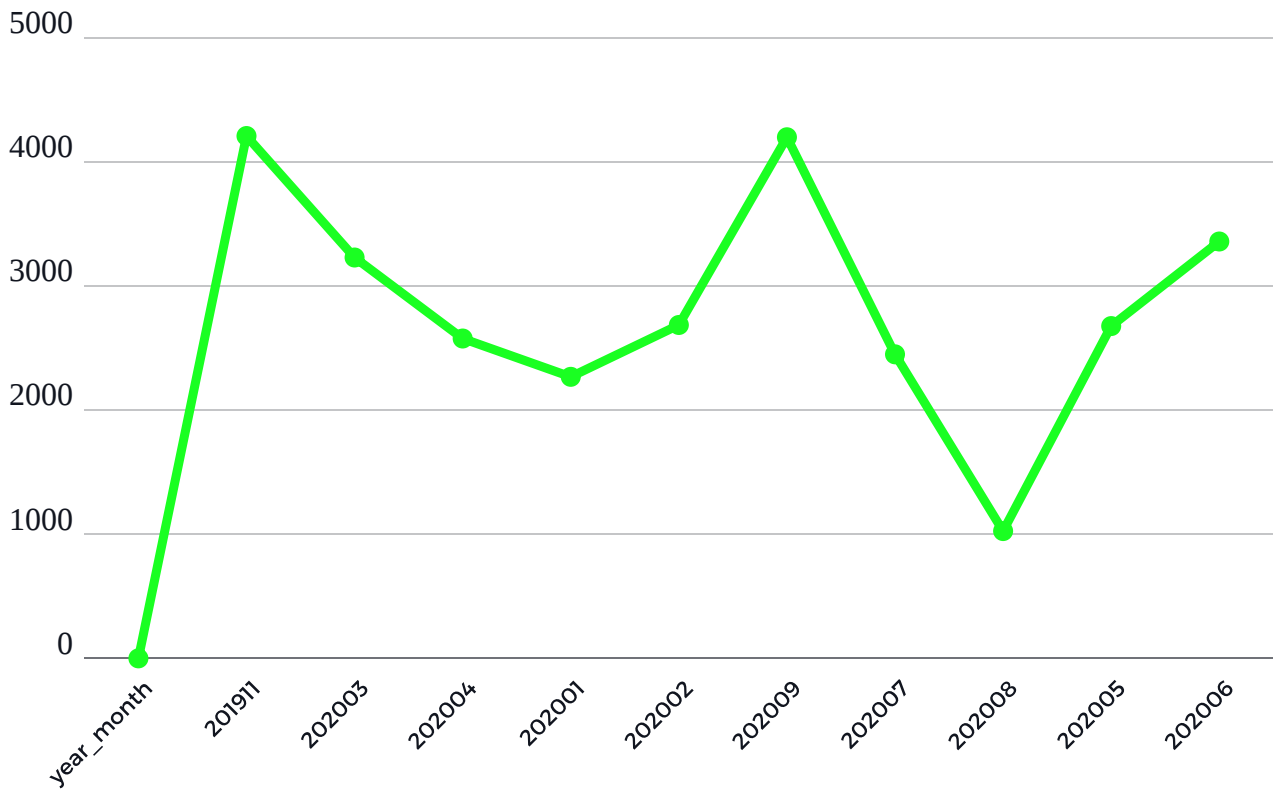
Ecommerce statistics state that 40 percent of internet users have bought products or goods online via online devices. This amounts to more than 1 billion online buyers and is projected to continuously grow.

72%

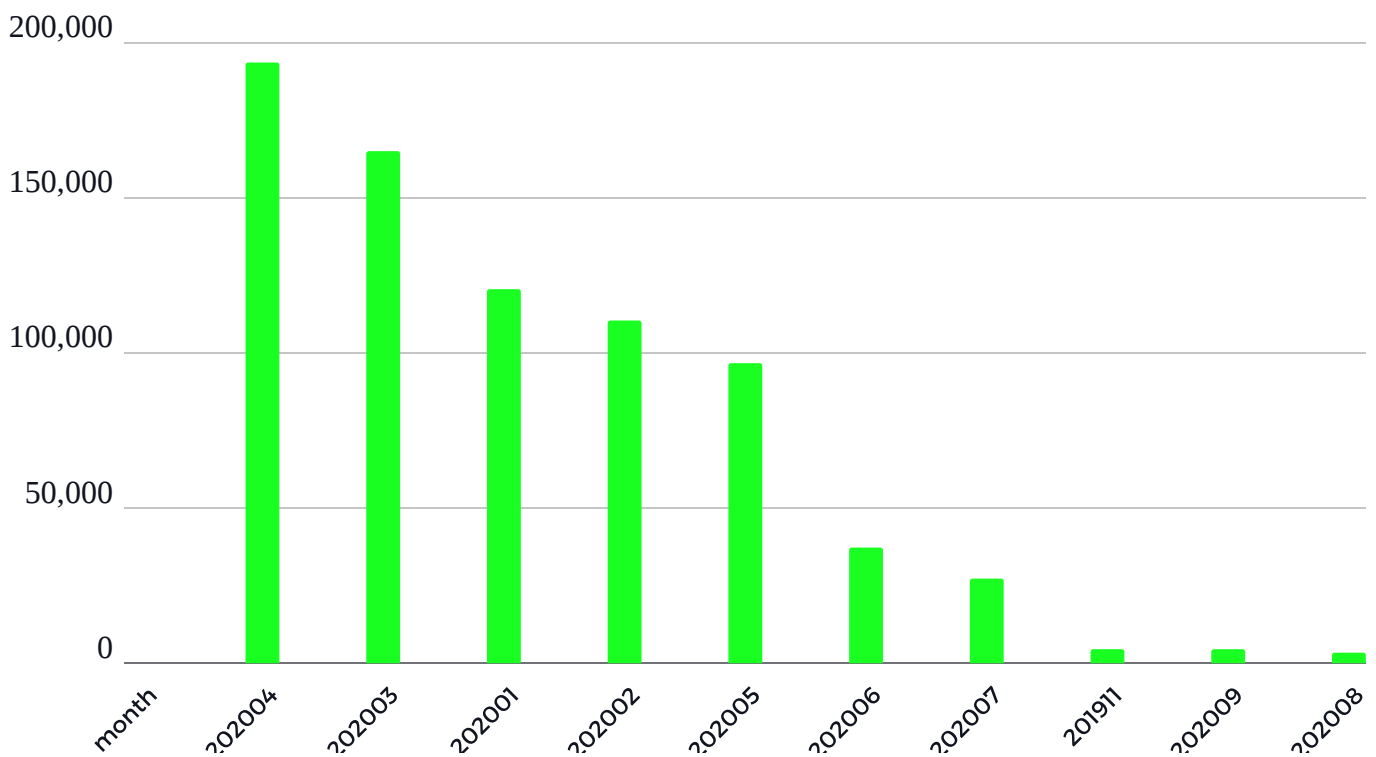
The ecommerce has grown this year and we must attack the market with an aggressive marketing strategy and a restocking of key products

LIFESTORE

AVERAGE MONTHLY SALES



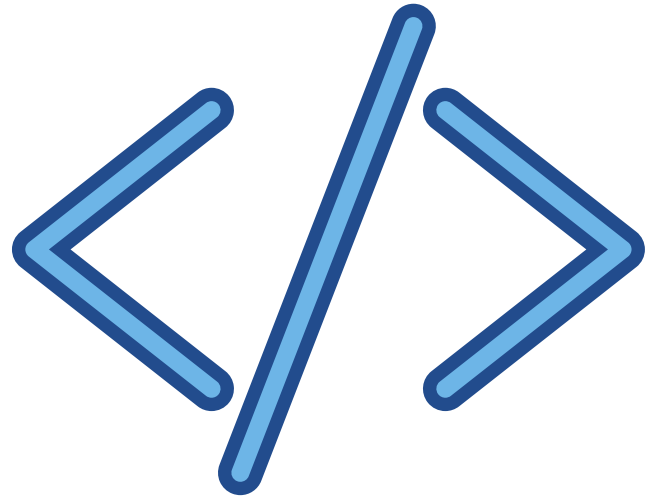
BEST SELLING MONTHS



03 CODE

Online documentation:

<https://github.com/EduardoAlcala/emtech.git>



PROJECT 1

I hope you find this program useful, it shows the different areas of sales opportunity of lifestore

If you are having issues, please let me know: eduardoalcalahuerta@gmail.com

LICENSE

The project is licensed under the GNU GPLv3 license.



```

"""
This is the LifeStore-SalesList data:
lifestore-searches = [id_search, id product]
lifestore-sales = [id_sale, id_product, score (from 1 to 5), date, refund (1 for true or 0 to
false)]
lifestore-products = [id_product, name, price, category, stock]

Default login user: eduardoalcala
Default login password: admin
"""

```

```

#Main function from an interactive prompt (login)
def main():
    login()

```

```

def login():
    #default user
    user = "1"
    #default password
    password = "1"
    print("Enter user: ")
    user_input = input()
    print("Enter password: ")
    password_input = input()
    print()
    #both fields are correct
    if user_input == user and password_input == password:
        #Clean screen
        print("\033[H\033[J")
        print("#####- Welcome back " + user + " -#####")
        menu()
    #user doesn't match
    elif user_input != user:
        print()
        print("user doesn't exist, please try again")
        login()
    #password doesn't match
    elif password_input != password:
        print()
        print("The password you entered is incorrect, please try again")
        login()

```


LIFESTORE

```
def menu():
    print()
    print("***** MAIN MENU *****")
    print()

#user interface for menu options
    option = input("""
1: Best Selling Products
2: Most searched products
3: Lowest sales by category
4: Lowest searches by category
5: Best product reviews
6: Worst product reviews
7: Total revenues
8: Average monthly sale
9: Total annual sale
10: Best selling months
11: Exit
Enter an option: """)
```

```
def best_sales():
    """
    With the sales_dictionary each element can be identified by a key
    """
    sales_dictionary = {}

    for sales_data in lifestore_sales:
        try:
            sales_dictionary[sales_data[1]] = int(sales_dictionary[sales_data[1]]) + 1
        except:
            sales_dictionary[sales_data[1]] = 1
    """
    Count the number of times the element is repeated and append it to sales_tuple
    """

    sales_counter = []
    for product_id, sale in sales_dictionary.items():
        sales_counter.append([product_id, sale])

    sales_tuple = tuple(sales_counter)
    """
    Sales tuple is sorted from higher to lower by lambda function
    """

    sales_tuple = sorted(sales_tuple, key=lambda x : x[1], reverse = True)
    print(sales_tuple[:21])
```

LIFESTORE

```
def most_searched():  
    """  
    With the search_dictionary each element can be identified by a key  
    """  
  
    search_dictionary = {}  
  
    for search_data in lifestore_searches:  
        try:  
            search_dictionary[search_data[1]] = int(search_dictionary[search_data[1]]) + 1  
        except:  
            search_dictionary[search_data[1]] = 1  
  
    """  
  
    Count the number of times the element is repeated and append it to search_tuple  
    """  
  
    search_counter = []  
    for product_id, sale in search_dictionary.items():  
        search_counter.append([product_id, sale])  
  
    search_tuple = tuple(search_counter)  
  
    search_tuple = sorted(search_tuple, key=lambda x : x[1], reverse= True)  
  
    print(search_tuple[:28])
```

```
def category_lowest_searches():  
    """  
    With the search_dictionary each element can be identified by a key  
    """  
  
    search_dictionary = {}  
  
    """  
    Get match info from sales  
    """  
  
    for sales_data in lifestore_sales:  
        try:  
            search_dictionary[sales_data[1]] = int(search_dictionary[sales_data[1]]) + 1  
        except:  
            search_dictionary[sales_data[1]] = 1  
  
    search_counter = []  
    for product_id, sale in search_dictionary.items():  
        search_counter.append([product_id, sale])  
  
    search_tuple = tuple(search_counter)  
  
    search_tuple = sorted(search_tuple, key=lambda x : x[1], reverse = True)  
  
    category_dictionary = {}  
  
    for category_data in lifestore_products:  
        if category_data[3] in category_dictionary:  
            (category_dictionary[category_data[3]]).append(category_data[0])  
        else:  
            category_dictionary[category_data[3]] = [category_data[0]]  
  
    sales_category_data = {}  
  
    for category_id, product_id in category_dictionary.items():  
        for sales_data in search_tuple:  
            if sales_data[0] in product_id:  
                if category_id in sales_category_data:  
                    (sales_category_data[category_id]).append([sales_data[0],sales_data[1]])  
                else:  
                    sales_category_data[category_id] = [[sales_data[0],sales_data[1]]]  
  
    for sort_category in sales_category_data:  
        sales_category_data[sort_category] = sorted(sales_category_data[sort_category], key=lambda x :  
x[1], reverse = True)  
  
    print(sales_category_data)
```

```

def lowest_searches():

    search_dictionary = {}

    for search_data in lifestore_searches:
        try:
            search_dictionary[search_data[1]] = int(search_dictionary[search_data[1]]) + 1
        except:
            search_dictionary[search_data[1]] = 1

    search_counter = []
    for product_id, sale in search_dictionary.items():
        search_counter.append([product_id, sale])

    search_tuple = tuple(search_counter)

    search_tuple = sorted(search_tuple, key=lambda x : x[1], reverse = True)

    category_dictionary = {}

    for category_data in lifestore_products:
        if category_data[3] in category_dictionary:
            (category_dictionary[category_data[3]]).append(category_data[0])
        else:
            category_dictionary[category_data[3]] = [category_data[0]]

    data_category_search = {}

    for category_id, product_id in category_dictionary.items():
        for search_data in search_tuple:
            if search_data[0] in product_id:
                if category_id in data_category_search:
                    (data_category_search[category_id]).append([search_data[0], search_data[1]])
                else:
                    data_category_search[category_id] = [[search_data[0], search_data[1]]]

    for sort_category in data_category_search:
        data_category_search[sort_category] = sorted(data_category_search[sort_category], key=lambda x : x[1])

    print(data_category_search)

```

```

def best_reviewed():

    sales_dictionary = {}

    for sales_data in lifestore_sales:
        if sales_data[4] == 0:
            try:
                sales_dictionary[sales_data[1]] = int(sales_dictionary[sales_data[1]]) + 1
            except:
                sales_dictionary[sales_data[1]] = 1

    sales_counter = []
    for product_id, sale in sales_dictionary.items():
        sales_counter.append([product_id, sale])

    sales_tuple = tuple(sales_counter)

    sales_tuple = sorted(sales_tuple, key=lambda x : x[1], reverse = True)

    print(sales_tuple[:20])

```

```

def worst_reviewed():
    sales_dictionary = {}

    for sale_data in lifestore_sales:
        if sale_data[4] == 0:
            try:
                sales_dictionary[sale_data[1]] = int(sales_dictionary[sale_data[1]]) + 1
            except:
                sales_dictionary[sale_data[1]] = 1

    sales_counter = []
    for product_id, sale in sales_dictionary.items():
        sales_counter.append([product_id, sale])

    sales_tuple = tuple(sales_counter)

    sales_tuple = sorted(sales_tuple, key=lambda x : x[1], reverse = True)

    print(sales_tuple[:20])

```

```

def total_revenues():
    products = []
    sales = []

    for sale in lifestore_sales:
        if sale[4] == 0:
            products.append(sale[1])

    counter = 0

    for product in products:
        if product == lifestore_products[counter][0]: sales.append(lifestore_products[counter][2])
        else:
            counter = counter + 1

    total_revenues = 0

    for sale in sales:
        total_revenues = (total_revenues + sale)

    print("Total revenues = " + str(total_revenues))

```

```

def average_monthly_sale():

    months = ['01','02','03','04','05','06','07','08','09','10','11','12']
    count = 0
    monthly_sales = [[],[],[],[],[],[],[],[],[],[],[],[],[]]
    for element in months:
        for element1 in lifestore_sales:
            temp_date = element1[3].split('/')
            #print(temp_date)
            if element == temp_date[1]:
                monthly_sales[int(element)].append(element1)

    monthly_amount = [[],[],[],[],[],[],[],[],[],[],[],[],[]]

    for element2 in monthly_sales:
        for element3 in element2:
            id_temp = element3[1]
            for element4 in lifestore_products:
                if id_temp == element4[0]:
                    monthly_amount[count].append(element4[2])
            count = count + 1

    #print(monthly_amount)

    average_month = [[],[],[],[],[],[],[],[],[],[],[],[],[]]


    count = 0
    for element5 in monthly_amount:
        total_month = 0
        #print(element5)

        sales_count = 0
        for element6 in element5:
            total_month = total_month + element6
            sales_count = sales_count + 1


        if sales_count == 0:
            average_month[count].append(0)
            sales_count = 0
        else:
            average_month[count].append(total_month/sales_count)
            count = count + 1
            sales_count = 0

    #Every element is a month
    print("Average monthly sale: " + str(average_month))

```



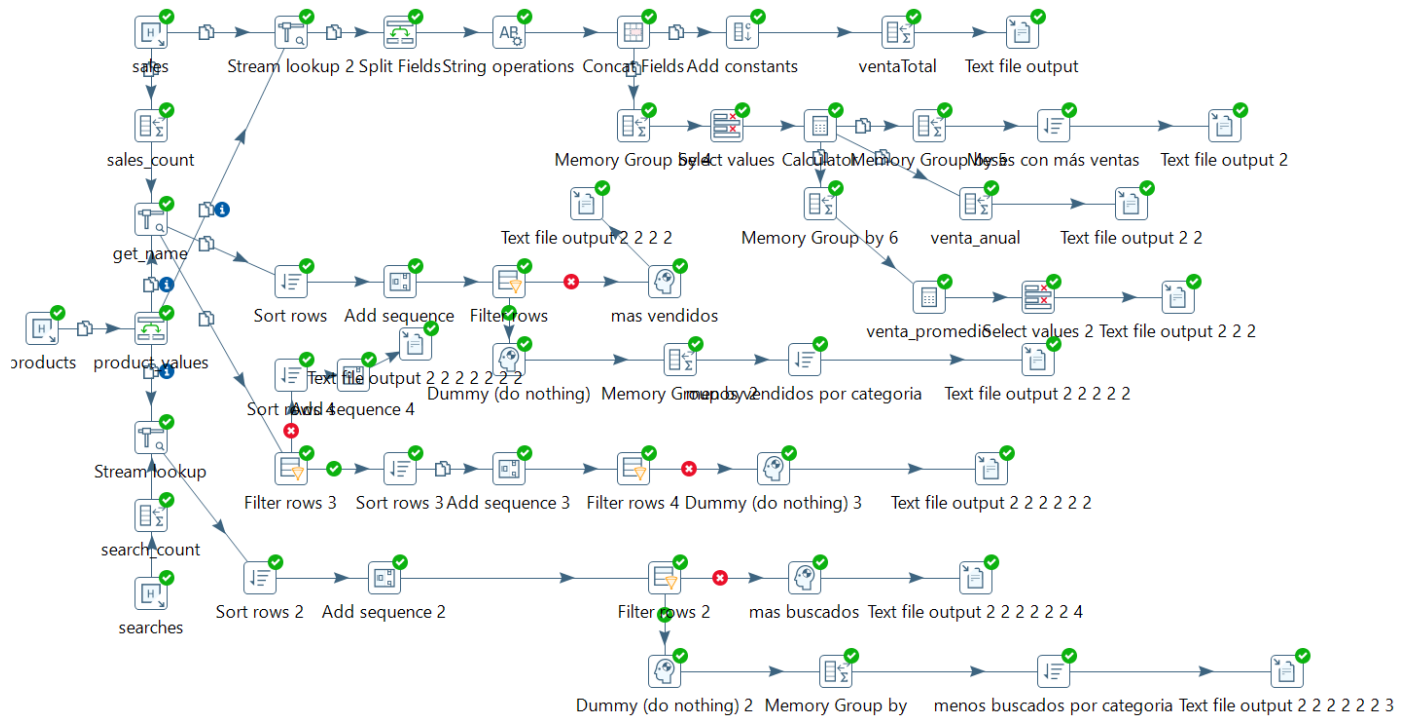
```
def total_annual_sale():  
    """  
    total_month = average_monthly_sale()  
    total_sale = 0  
    for element in total_month[1]:  
        total_sale = total_sale + len(element)  
  
    print(total_sale)  
    """  
    #this code is in development  
    print("This code is in development :")  
    pass
```



```
def best_selling_months():  
    #this code is in development  
    print("This code is in development :")  
    pass
```



Additionally I carried out an ETL process with all the specified requirements. In order to comply with all of them and always seeking to have a second way of interpreting the information, i'm attaching this file





04 FINAL THOUGHTS

It was a wonderful learning experience for me while working on this project. This project took me through the various phases of data analysis and gave me real insight into the world of data science.

The joy of working and the thrill involved while tackling the various problems and challenges gave me a feel of the data scientist's industry. It was due to this project I came to know how data analysis is made. Thank you!

