Best Gadget Finder

Abstract

The World Wide Web consists of an interlinked network of information, which is presented through websites to the users. World Wide Web has significantly changed the way we share, collect, and publish data. The amount of presented information grows constantly. As the world entered the era of big data, the need for its storage and analysis also grew. One of the biggest challenges in our lifestyle over the period of last decade is online shopping. COVID have further added a lot to this ever-growing industry of e-commerce. There are different ecommerce websites present at time, but while buying a product user or customer has to compare its price, rating, reviews across many different websites. Therefore, there is a need to create a platform where customer can compare a single product across other websites. The above-mentioned problem can be nullified to some extent by the proposed system. The execution is performed with the help of web scraping for data acquisition followed by data cleaning, data mapping, data storage, Flask API and user interface. This project provides the overall idea of web scraping technology, its working and the effects of this technology.

Sources:

amazon

Flipkart snapdeal

Keywords

Web scraping, Data mapping, Name matching, Elastic Search, Rest API, UI design.

1. Introduction

Customer faces problem while comparing the product on different platforms at the time of buying. Data analysis is the method of extracting solutions to the problems via interpretation of data. The analysis process comprises of discovering problems, resolve the accessibility of suitable data, determining which method can help in finding the solution to the interesting problem and convey the result.

The process through which we can create a comparative UI, involves certain steps that has to be mandatorily followed to get the desired output. The steps are as follows, first we need to scrape the data from Amazon, Flipkart, Snapdeal. The acquired data needs to be cleaned as the dataset may have duplicate, incorrect or incomplete entries. Data from various sources need to be standardized such as similar column names, adding or removing extra columns.

Finally, this data is then merged and save as a csv file. This process is called as data mapping. To store, search and analyse big volume of data Elasticsearch is used. Now, API will be created using Flask framework. For this data analysis, python is used as a programming language.

2. Problem statement and objectives

2.1 Problem statement

Many different e-commerce websites are available on internet. A customer needs to compare a product over many different websites while shopping. User wants to tally the product price, specifications, reviews, rating across other applications. In addition, this process is time consuming and tedious. Therefore, there is a need to help users compare products at a single platform.

2.2 Objectives

- To make use of modern techniques to create a comparative UI.
- To compare prices of gadgets across various e-commerce platforms.
- To provide average rating and availability of product.
- To gather reviews summary aggregated from various platforms.

3. Work done

3.1 Working principle

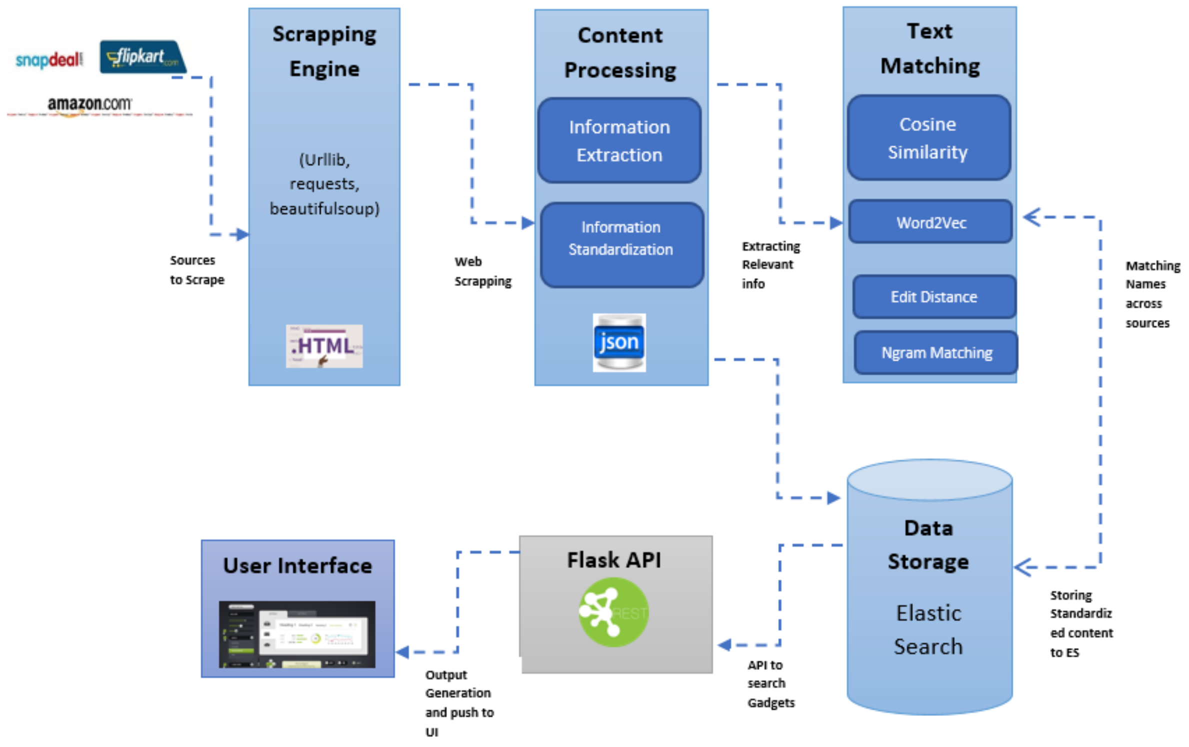


Fig 3.1 Detailed architecture design

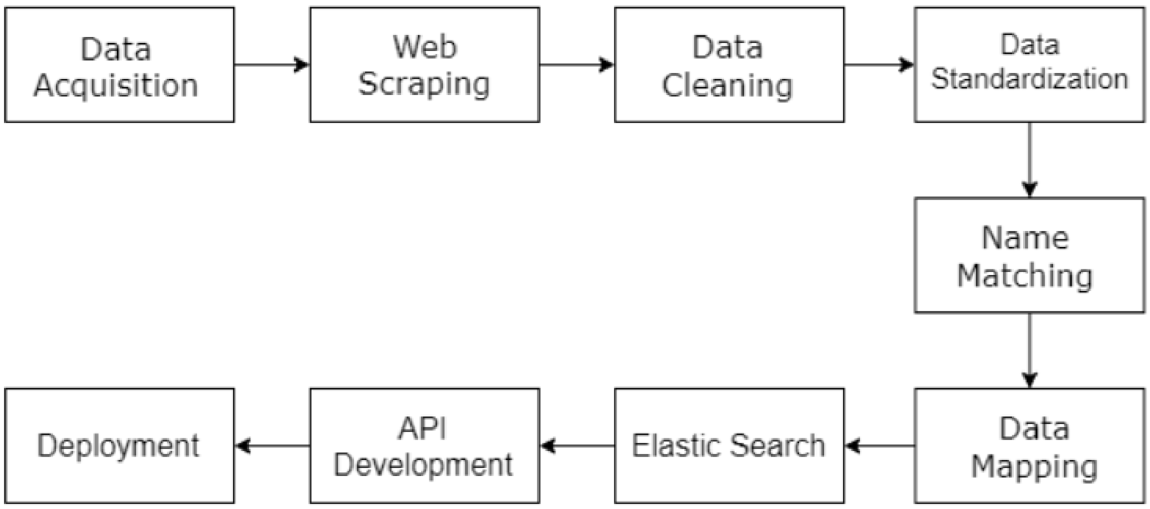
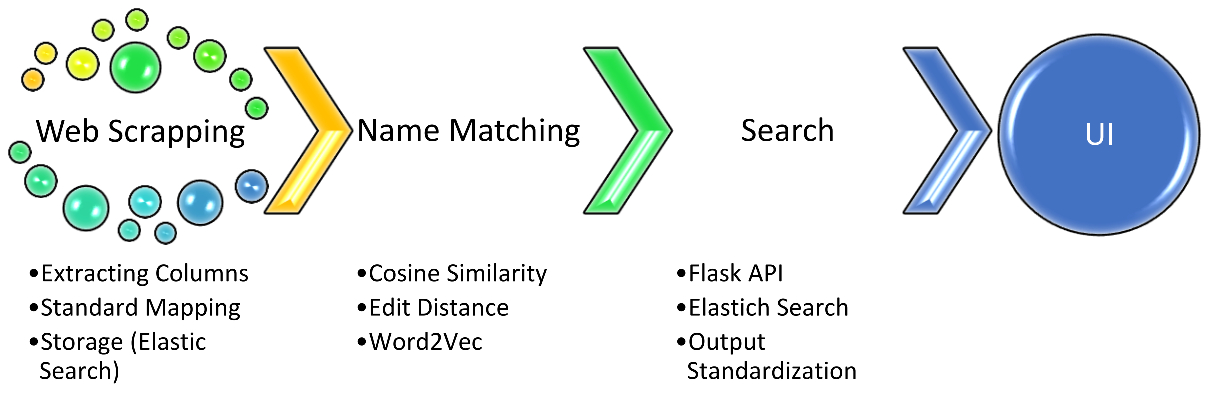


Fig 3.2 Overall flow of project

The detail working of each block explained below:



A) Data Acquisition

First step is to acquire the real time data from mentioned 3 websites. There can be many techniques to gather data such as open source platforms, APIs, RSS feeds or Web Scraping. We will be using Web Scraping for data collection process. Further we will have steps like Data cleaning, Standardization, Mapping, and Storage.

B) Web Scraping

Web scraping is the process of extracting information and data from a website, transforming the information on a webpage into structured data for further analysis. Web scraping is also known as web harvesting or web data extraction.

Need of web scraping:

With the overwhelming data available on the internet, web scraping has become the essential approach to aggregating Big Datasets.

Steps to scrape the website:

In this project, we scraped the mobile products data from Amazon, Flipkart and Snapdeal by using Python programming language.

For web scraping there are a few different libraries to consider, including:

- Beautiful Soup – It is a Python package for parsing HTML and XML documents. It creates a parse tree for parsed pages that can be used to extract data from HTML, which is useful for web scraping.
- Requests – The requests library is used for making HTTP requests in Python.
- Selenium - Selenium is a powerful tool for controlling web browsers through programs and performing browser automation. It is functional for all browsers, works on all major OS and its scripts are written in various languages.

Process of web scraping

i. Inspect the webpage

To know which elements that we need to target in the python code, we need to first inspect the web page. To gather data from Amazon we inspected the page by right clicking on the element of interest and select inspect. This brings up the HTML code where we can see the element that each field is contained within. Since the data is stored in a table, it was straight forward to scrape with just a few lines of code. As the results span over many pages, we loop over all the pages to gather all information. When inspecting the page, it is easy to see a pattern in the html. The results are contained in rows within the table.

ii. Parse the webpage html using Beautiful Soup

The first step is to import the libraries which are required in web scraping. As mentioned above, BeautifulSoup helps in handling the html. The next library we imported is urllib which makes the connection to the webpage. Finally, we will be writing the output to a csv so, we also imported csv library. The next step is to define the URL that we will be scraping. We then made the connection to the webpage and we can parse the html using BeautifulSoup, storing the object in the variable ‘soup’. We can print the soup variable at this stage which should return the full parsed html of the webpage we have requested.

iii. Search for html elements

As all of the results are contained within a table, we search the soup object for the table using the find method. We then found each row within the table using the find all method.

There are six columns in the table containing: Name, sales price, Original price, Rating, Product link, Image link.

iv. Looping through elements and saving variables

In python, it is useful to append the results to a list to then write the data to a file. We declared the list and set the headers of the csv before the loop. The next step is to loop over the results, process the data and append to rows which can be written to a csv. Some of this data however needs further cleaning to remove unwanted characters or extract further information.

Source	**Amazon**	**Flipkart**	**Snapdeal**
Total products	1889	985	54
Execution time	5 mins	4 mins	1 min

C) Data cleaning

Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. We have removed duplicate columns, some caps and comma operators from the obtained database to make it a relational database. For data cleaning, we have installed Pandas which is a software library written for the Python programming language for data manipulation and analysis. In addition, we have removed ^ sign from the sales price and original price by using string replace operator.

D) Data Standardization

User should get the information of availability of the product while shopping. To satisfy this requirement we have added a column named 'availability' by defining two functions. These functions return as available or sold out. In addition to this we have created a new column of unique ID for each product with the help of for loop. Customer may search a product name in different styles. Like phone brand with its color, phone brand with its storage. Our system should be able to provide the results by identifying these keywords.

<u>Example:</u> Product - REDMI 9i (Midnight Black, 64 GB)

<u>Synonyms</u>: REDMI 9i | REDMI 9i Midnight Black | REDMI 9i 64 GB

Therefore, we have created synonyms for every product by using rpartition method. This process is then followed by creating a dataframe and merging them into a list.

E) Name matching

As we had three different source files with the same product but with different way of naming the products. We used FuzzyWuzzy python library to match two strings and get a score form the FuzzyWuzzy Library. The way it works is it matches the string for common patterns and name and gives a score based on their similarity. After we had the algorithm for name matching, we could move on to the data mapping where based on the name matching, we can pull all the records on the same file. We used Pandas to get the data in dataframe and then match if with the FuzzyWuzzy algorithm to figure out the score and match the data as per the score if it was greater than 0.85. The FuzzyWuzzy library is built on top of difflib library; python-Levenshtein is used for speed. Therefore, it is one of the best ways for string matching in python.

Levenshtein distance

It is a metric to measure how apart are two sequences of words. It is also possible to calculate the Levenshtein distance. This can be done using the formula:

$$\frac{(|a| + |b|) - lev_{a,b}(i, j)}{|a| + |b|}$$

First Name	Second Name	Similarity ratio
Redmi 9A Nature Green 2GB RAM 32GB Storage	Redmi 9A Nature Green 32 GB	88
Samsung Galaxy M11 Metallic Blue 4GB RAM	SAMSUNG Galaxy M11 Metallic Blue 32 GB	91
Oppo A31 Fantasy White 6GB RAM	OPPO A31 Fantasy White 64 GB	90
Nokia 105 Single SIM Black	Nokia 105	100
Redmi Note 10 Aqua Green 6GB RAM	REDMI Note 10 Aqua Green 128 GB	87

Here is the example of two strings are matching. For more info refer this FuzzyWuzzy documentation https://pypi.org/project/fuzzywuzzy/

F) Data Mapping

We needed to collect all the data from three different files match the names and aggregate the data in one record for all three different sources. All the products are mapped based on their name from all three sources. This is done so we can have a row of data from where we can identify the product and its price and get the relevant information for the product in the same row. We considered Amazon as our base as it had the most number of mobile phones and we ran the name matching algorithm to match the records and add it to the new columns that where added for other sources. In case the data was not present, it was left blank. For individual products we ran the name matching this time around with Snapdeal as the base and the Flipkart as the base. Merging the data and then removing duplicate entries to get all the exclusive products we might have missed in the other source. We saved the data in csv format as it can be than imported easily by other systems.

Column names of a final dataset:

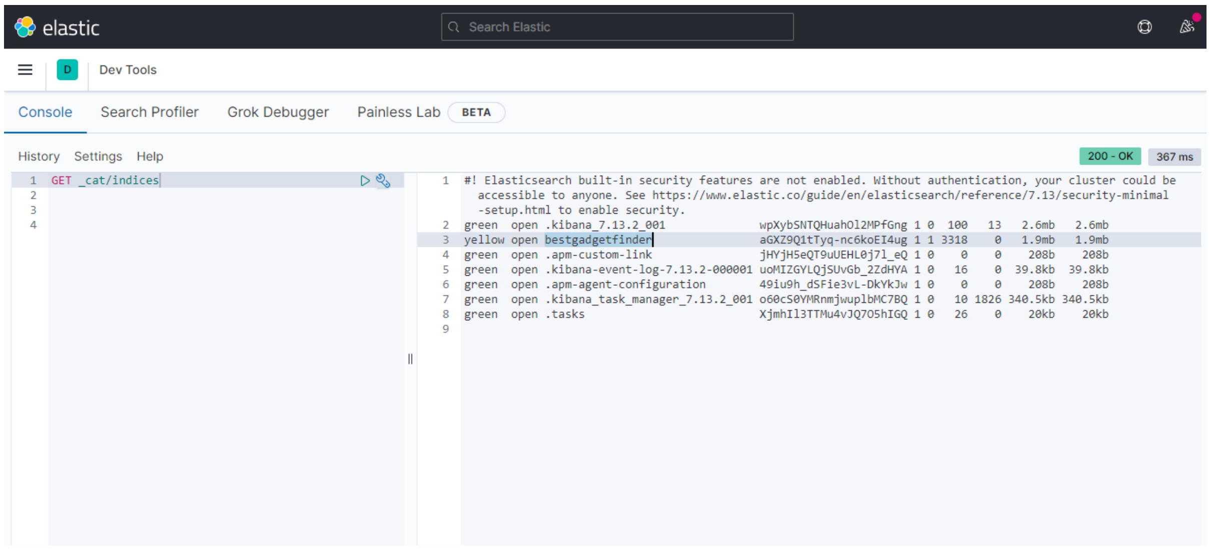
```
[ 'Cleaned_Name', 'New_Name', 'Amazon_Name', 'Flipkart_Name',
    'Snapdeal_Name', 'Amazon_SalesPrice', 'Flipkart_SalesPrice',
    'Snapdeal_SalesPrice', 'Amazon_OriginalPrice', 'Flipkart_OriginalPrice',
    'Snapdeal_OriginalPrice', 'Amazon_Rating', 'Flipkart_Rating',
    'Snapdeal_Rating', 'Amazon_ProductLink', 'Flipkart_ProductLink',
    'Snapdeal_ProductLink', 'Amazon_ImageLink', 'Flipkart_ImageLink',
    'Snapdeal_ImageLink']
```

G) Elastic Search

Elastic Search is a NO SQL database, which is scalable and can give results in seconds as well as it can be used to filter searches and full text match and provides results in a json format. We had to install the Elasticsearch and Kibana services on the system. We then needed to check and see if it is communicating with Python. We used the library Elasticsearch for python as a client for the database. We created an instance of the Elasticsearch and indexed the filed in elastic search. Once the data was present, we created a query to get the matching result from the database. The query would be used in the API when it would connect with the Elasticsearch database. We used Kibana to view the dataset which is stored in elastic search database.

Kibana enables you to give shape to your data and navigate the Elastic Stack. With Kibana, you can:

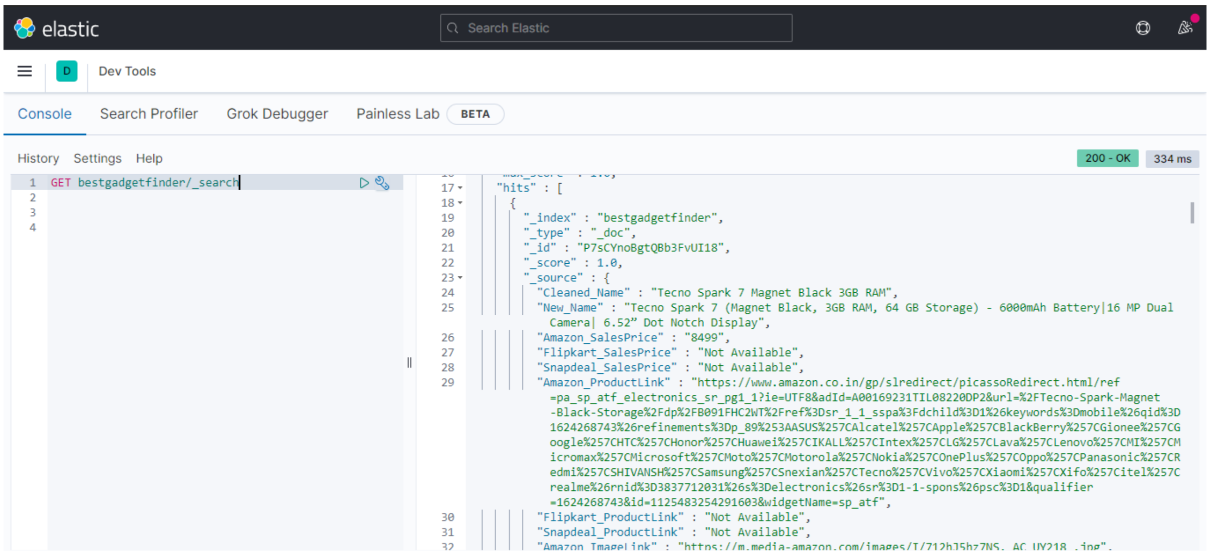
- **Search, observe, and protect.** From discovering documents to analyzing logs to finding security vulnerabilities, Kibana is your portal for accessing these capabilities and more.
- **Visualize and analyze your data.** Search for hidden insights, visualize what you've found in charts, gauges, maps and more, and combine them in a dashboard.
- **Manage, monitor, and secure the Elastic Stack.** Manage your indices and ingest pipelines, monitor the health of your Elastic Stack cluster, and control which users have access to which features.



```
GET _cat/indices
```

```
#! Elasticsearch built-in security features are not enabled. Without authentication, your cluster could be
  accessible to anyone. See https://www.elastic.co/guide/en/elasticsearch/reference/7.13/security-minimal
  -setup.html to enable security.
green  open .kibana_7.13.2_001                wpXybSNTQHuahOl2MPfGng 1 0  100   13   2.6mb   2.6mb
yellow open bestgadgetfinder                  aGXZ9Q1tTyq-nc6koEI4ug 1 1 3318    0   1.9mb   1.9mb
green  open .apm-custom-link                  jHYjH5eQT9uUEHL0j7l_eQ 1 0    0    0    208b    208b
green  open .kibana-event-log-7.13.2-000001 uoMIZGYLQjSUvGb_2ZdHYA 1 0   16    0  39.8kb  39.8kb
green  open .apm-agent-configuration          49iu9h_dSFie3vL-DkYkJw 1 0    0    0    208b    208b
green  open .kibana_task_manager_7.13.2_001 o60cS0YMRnmjwuplbMC7BQ 1 0   10 1826 340.5kb 340.5kb
green  open .tasks                            XjmhIl3TTMu4vJQ7O5hIGQ 1 0   26    0    20kb    20kb
```

Fig 3.3 Check indices in Kibana



```
GET bestgadgetfinder/_search
```

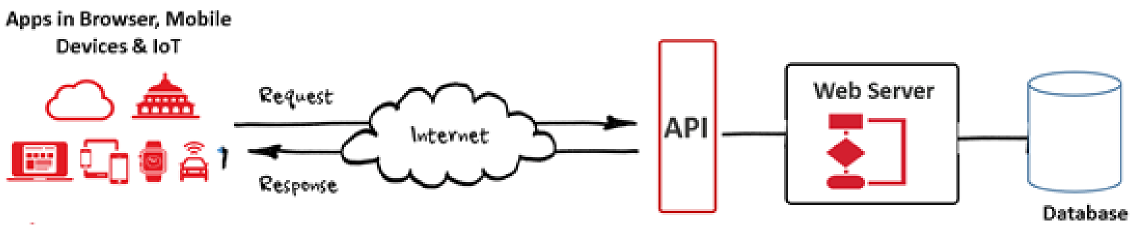
```
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "bestgadgetfinder",
        "_type" : "_doc",
        "_id" : "P7sCYnoBgtQBb3FvUI18",
        "_score" : 1.0,
        "_source" : {
          "Cleaned_Name" : "Tecno Spark 7 Magnet Black 3GB RAM",
          "New_Name" : "Tecno Spark 7 (Magnet Black, 3GB RAM, 64 GB Storage) - 6000mAh Battery|16 MP Dual
            Camera| 6.52" Dot Notch Display",
          "Amazon_SalesPrice" : "8499",
          "Flipkart_SalesPrice" : "Not Available",
          "Snapdeal_SalesPrice" : "Not Available",
          "Amazon_ProductLink" : "https://www.amazon.co.in/gp/slredirect/picassoRedirect.html/ref
            =pa_sp_atf_electronics_sr_pg1_1?ie=UTF8&adId=A00169231TIL08220DP2&url=%2FTecno-Spark-Magnet
            -Black-Storage%2Fdp%2FB091FHC2WT%2Fref%3Dsr_1_1_sspa%3Fdchild%3D1%26keywords%3Dmobile%26qid%3D
            1624268743%26refinements%3Dp_89%253AASUS%257CAlcatel%257CApple%257CBlackBerry%257CGionee%257CG
            oogle%257CHTC%257CHonor%257CHuawei%257CIKALL%257CIntex%257CLG%257CLava%257CLenovo%257CMI%257CM
            icromax%257CMicrosoft%257CMoto%257CMotorola%257CNokia%257COnePlus%257COppo%257CPanasonic%257CR
            edmi%257CSHIVANSH%257CSamsung%257CSnexian%257CTecno%257CVivo%257CXiaomi%257CXifo%257Citel%257C
            realme%26rnid%3D3837712031%26s%3Delectronics%26sr%3D1-1-spons%26psc%3D1&qualifier
            =1624268743&id=1125483254291603&widgetName=sp_atf",
          "Flipkart_ProductLink" : "Not Available",
          "Snapdeal_ProductLink" : "Not Available",
          "Amazon_ImageLink" : "https://m.media-amazon.com/images/I/712hJ5hz7NS._AC_UY218_.jpg",
```

Fig 3.4 Searching for data

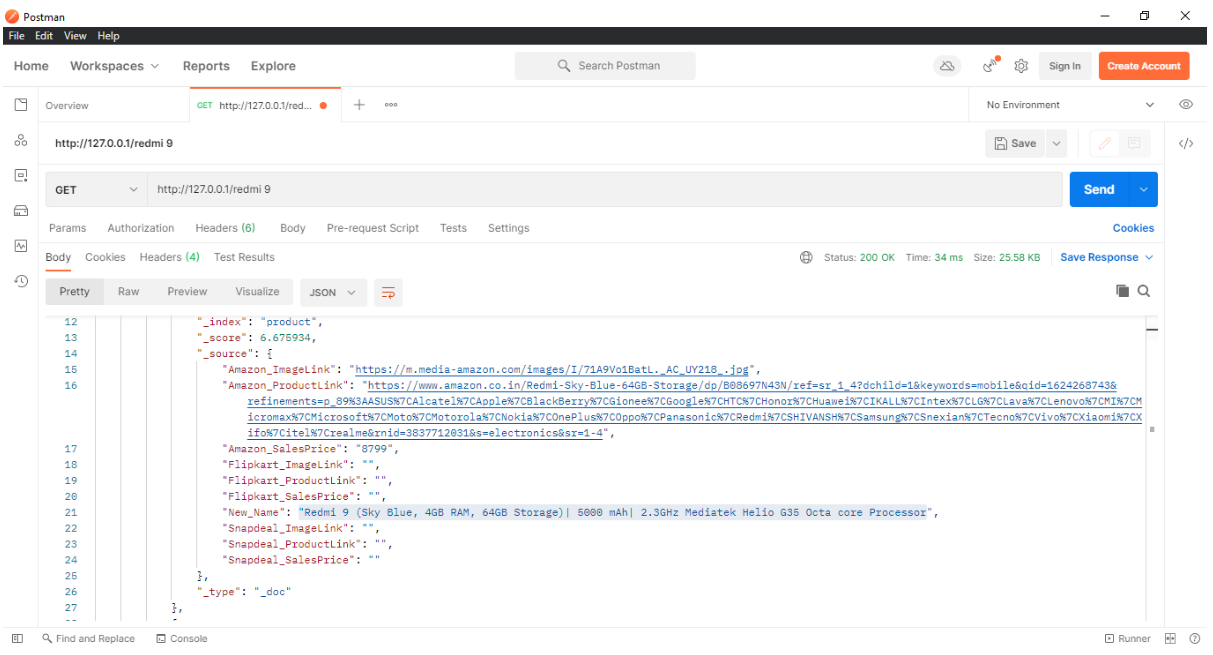
H) API development

API stands for Application Programming Interface. In basic terms, APIs are a set of functions and procedures that allow for the creation of applications that access data and features of other applications, services, or operating systems.

Good APIs make it easier to develop a computer program by providing all the building blocks, which are then put together by the programmer.



Flask is a web framework for python, meaning that it provides functionality for building web applications, including managing HTTP requests and rendering templates. For API, we required to import flask, jsonify. Jsonify is a function in Flask's flask.json module. Jsonify serializes data JSON format. The two most common requests are GET, which pulls data from a server and POST, which pushes data from a server. After running the code, we can see a http link, we copied this link and gave to a postman (is an API and development tool which helps to build, test and modify APIs. It has the ability to make various types of HTTP requests, saving environments for later use, converting the API to code for various languages like JavaScript, python).


Postman
File Edit View Help
Home Workspaces Reports Explore
Search Postman
Sign In Create Account
Overview
No Environment
Save
Send
Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies
Body Cookies Headers (4) Test Results
Save Response
Pretty Raw Preview Visualize JSON
Find and Replace Console Runner

http://127.0.0.1/redmi 9

GET http://127.0.0.1/redmi 9

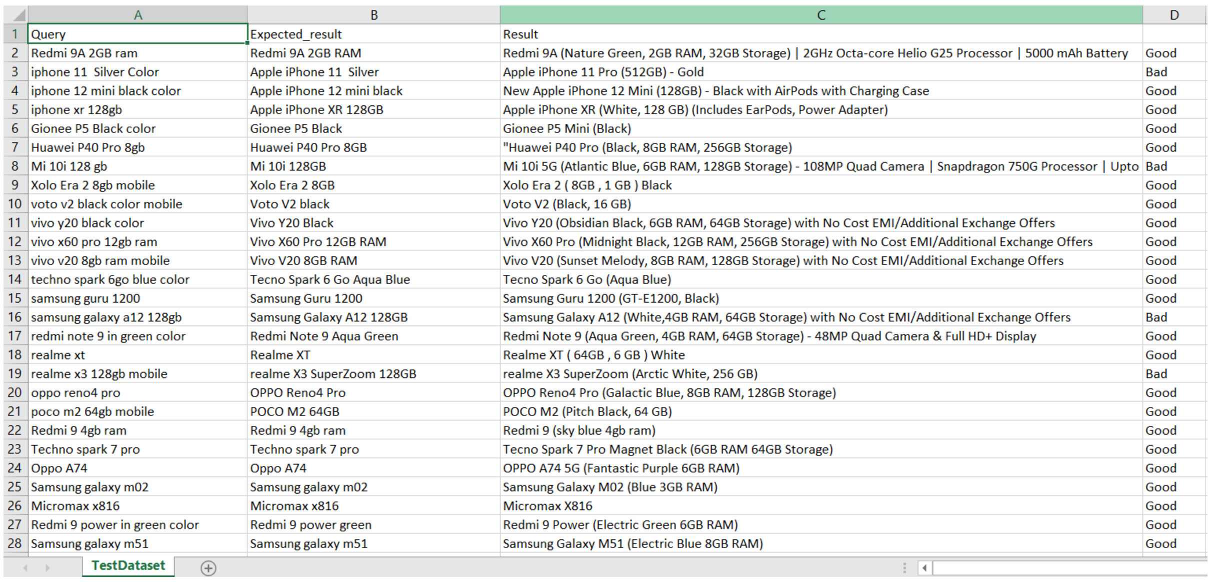
Status: 200 OK Time: 34 ms Size: 25.58 KB

```
"_index": "product",
"_score": 6.675934,
"_source": {
    "Amazon_ImageLink": "https://m.media-amazon.com/images/I/71A9Vo1BatL._AC_UY218_.jpg",
    "Amazon_ProductLink": "https://www.amazon.co.in/Redmi-Sky-Blue-64GB-Storage/dp/B08697N43N/ref=sr_1_4?dchild=1&keywords=mobile&qid=1624268743&
        refinements=p_89%3AASUS%7CAlcatel%7CApple%7CBlackBerry%7CGionee%7CGoogle%7CHTC%7CHonor%7CHuawei%7CIKALL%7CIntex%7CLG%7CLava%7CLenovo%7CMI%7CM
        icromax%7CMicrosoft%7CMoto%7CMotorola%7CNokia%7COnePlus%7COppo%7CPanasonic%7CRedmi%7CSHIVANSH%7CSamsung%7CSnexian%7CTecno%7CVivo%7CXiaomi%7CX
        ifo%7Citel%7Crealme&rnid=3837712031&s=electronics&sr=1-4",
    "Amazon_SalesPrice": "8799",
    "Flipkart_ImageLink": "",
    "Flipkart_ProductLink": "",
    "Flipkart_SalesPrice": "",
    "New_Name": "Redmi 9 (Sky Blue, 4GB RAM, 64GB Storage)| 5000 mAh| 2.3GHz Mediatek Helio G35 Octa core Processor",
    "Snapdeal_ImageLink": "",
    "Snapdeal_ProductLink": "",
    "Snapdeal_SalesPrice": ""
},
"_type": "_doc"
},
```

Fig 3.5 GET request in Postman

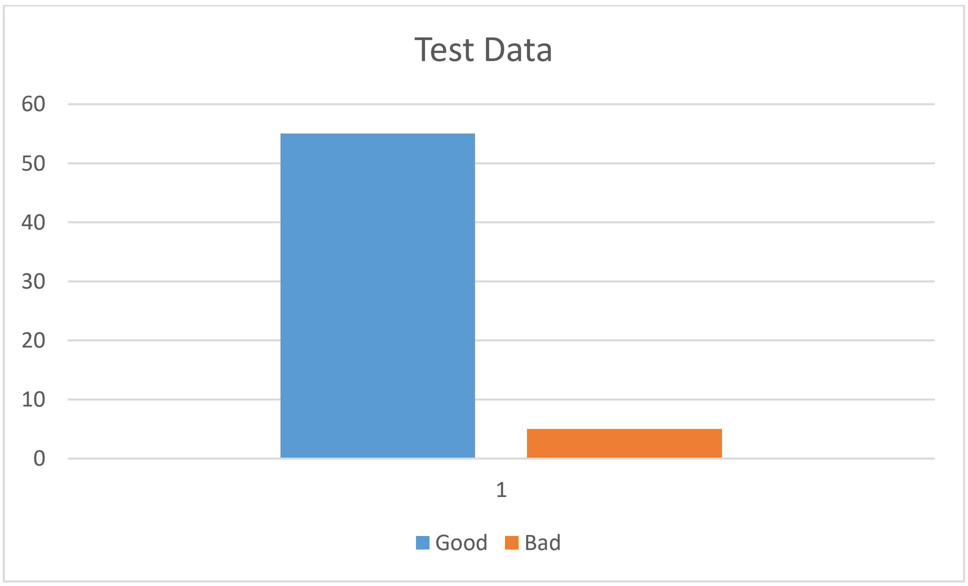
I) Data testing

It is the input given to a program during test execution. It represents data that affects or affected by software execution while testing. We have tabulated our test results as query, expected results, actual results and positive or negative result. It gave us 95-96% good results and 1-2% bad results. So, we can conclude that data testing is successful.



	A	B	C	D
1	Query	Expected_result	Result	
2	Redmi 9A 2GB ram	Redmi 9A 2GB RAM	Redmi 9A (Nature Green, 2GB RAM, 32GB Storage) | 2GHz Octa-core Helio G25 Processor | 5000 mAh Battery	Good
3	iphone 11 Silver Color	Apple iPhone 11 Silver	Apple iPhone 11 Pro (512GB) - Gold	Bad
4	iphone 12 mini black color	Apple iPhone 12 mini black	New Apple iPhone 12 Mini (128GB) - Black with AirPods with Charging Case	Good
5	iphone xr 128gb	Apple iPhone XR 128GB	Apple iPhone XR (White, 128 GB) (Includes EarPods, Power Adapter)	Good
6	Gionee P5 Black color	Gionee P5 Black	Gionee P5 Mini (Black)	Good
7	Huawei P40 Pro 8gb	Huawei P40 Pro 8GB	"Huawei P40 Pro (Black, 8GB RAM, 256GB Storage)	Good
8	Mi 10i 128 gb	Mi 10i 128GB	Mi 10i 5G (Atlantic Blue, 6GB RAM, 128GB Storage) - 108MP Quad Camera | Snapdragon 750G Processor | Upto	Bad
9	Xolo Era 2 8gb mobile	Xolo Era 2 8GB	Xolo Era 2 (8GB , 1 GB) Black	Good
10	voto v2 black color mobile	Voto V2 black	Voto V2 (Black, 16 GB)	Good
11	vivo y20 black color	Vivo Y20 Black	Vivo Y20 (Obsidian Black, 6GB RAM, 64GB Storage) with No Cost EMI/Additional Exchange Offers	Good
12	vivo x60 pro 12gb ram	Vivo X60 Pro 12GB RAM	Vivo X60 Pro (Midnight Black, 12GB RAM, 256GB Storage) with No Cost EMI/Additional Exchange Offers	Good
13	vivo v20 8gb ram mobile	Vivo V20 8GB RAM	Vivo V20 (Sunset Melody, 8GB RAM, 128GB Storage) with No Cost EMI/Additional Exchange Offers	Good
14	techno spark 6go blue color	Tecno Spark 6 Go Aqua Blue	Tecno Spark 6 Go (Aqua Blue)	Good
15	samsung guru 1200	Samsung Guru 1200	Samsung Guru 1200 (GT-E1200, Black)	Good
16	samsung galaxy a12 128gb	Samsung Galaxy A12 128GB	Samsung Galaxy A12 (White,4GB RAM, 64GB Storage) with No Cost EMI/Additional Exchange Offers	Bad
17	redmi note 9 in green color	Redmi Note 9 Aqua Green	Redmi Note 9 (Aqua Green, 4GB RAM, 64GB Storage) - 48MP Quad Camera & Full HD+ Display	Good
18	realme xt	Realme XT	Realme XT (64GB , 6 GB) White	Good
19	realme x3 128gb mobile	realme X3 SuperZoom 128GB	realme X3 SuperZoom (Arctic White, 256 GB)	Bad
20	oppo reno4 pro	OPPO Reno4 Pro	OPPO Reno4 Pro (Galactic Blue, 8GB RAM, 128GB Storage)	Good
21	poco m2 64gb mobile	POCO M2 64GB	POCO M2 (Pitch Black, 64 GB)	Good
22	Redmi 9 4gb ram	Redmi 9 4gb ram	Redmi 9 (sky blue 4gb ram)	Good
23	Techno spark 7 pro	Techno spark 7 pro	Tecno Spark 7 Pro Magnet Black (6GB RAM 64GB Storage)	Good
24	Oppo A74	Oppo A74	OPPO A74 5G (Fantastic Purple 6GB RAM)	Good
25	Samsung galaxy m02	Samsung galaxy m02	Samsung Galaxy M02 (Blue 3GB RAM)	Good
26	Micromax x816	Micromax x816	Micromax X816	Good
27	Redmi 9 power in green color	Redmi 9 power green	Redmi 9 Power (Electric Green 6GB RAM)	Good
28	Samsung galaxy m51	Samsung galaxy m51	Samsung Galaxy M51 (Electric Blue 8GB RAM)	Good

Fig 3.6 Test data

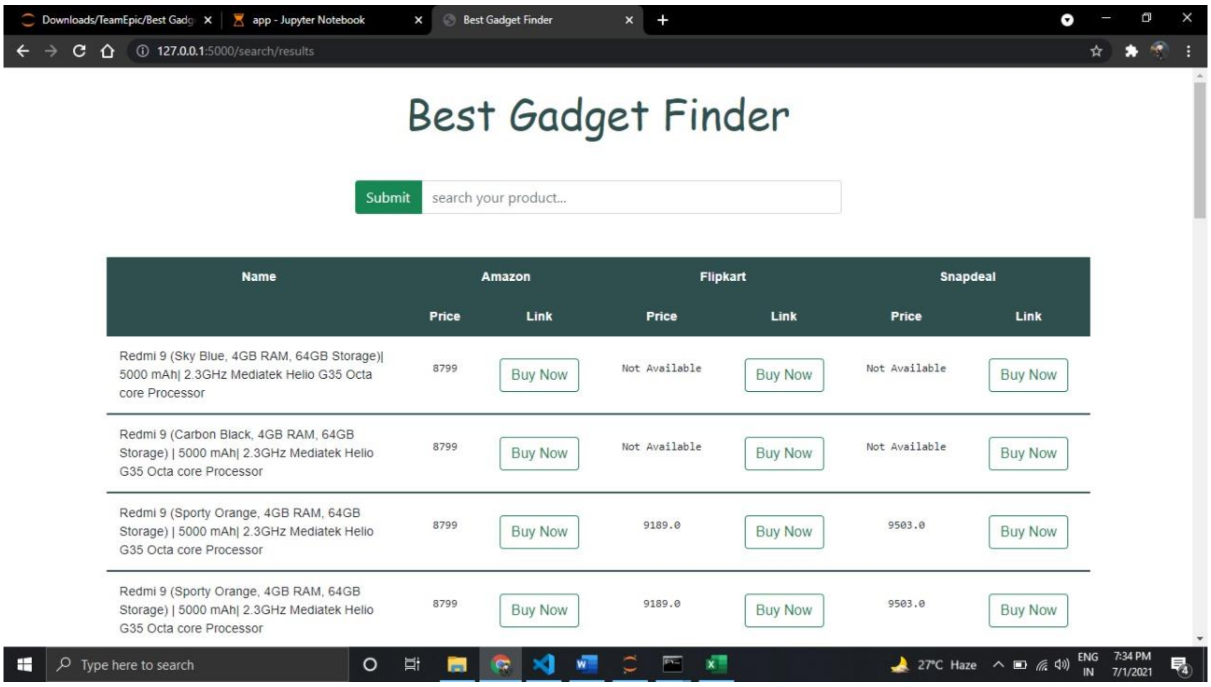


J) UI Development

In frontend we have created a website with the help of HTML, CSS and bootstrap framework. Bootstrap is a powerful toolkit - a collection of HTML, CSS, and JavaScript tools for creating and building web pages and web applications. In Backend, we have used flask framework to build web-application. It fetch the data from elastic search to website. As a final result we can see a website with three sources Amazon, Flipkart and Snapdeal User can compare product price and accordingly make decision. Also there is a button provided like buy now, by clicking on it user will automatically get redirected to the product link and can buy the desired product.

K) Deployment

We have created a wireframe design using the software Figma. In backend we have used flask API which is used for managing http requests. Here we used fetch API function to get the data. And used this data to create a frontend. In frontend we have created a website with the help of HTML, CSS and bootstrap framework. Bootstrap is a powerful toolkit - a collection of HTML, CSS, and JavaScript tools for creating and building web pages and web applications. As a final result we can see a website with three sources Amazon Flipkart and Snapdeal. Also, there is a button provided like buy now, by clicking on it user will automatically get redirected to the product link and can buy the desired product.



Best Gadget Finder

Name	Amazon		Flipkart		Snapdeal	
	Price	Link	Price	Link	Price	Link
Redmi 9 (Sky Blue, 4GB RAM, 64GB Storage)| 5000 mAh| 2.3GHz Mediatek Helio G35 Octa core Processor	8799	Buy Now	Not Available	Buy Now	Not Available	Buy Now
Redmi 9 (Carbon Black, 4GB RAM, 64GB Storage) | 5000 mAh| 2.3GHz Mediatek Helio G35 Octa core Processor	8799	Buy Now	Not Available	Buy Now	Not Available	Buy Now
Redmi 9 (Sporty Orange, 4GB RAM, 64GB Storage) | 5000 mAh| 2.3GHz Mediatek Helio G35 Octa core Processor	8799	Buy Now	9189.0	Buy Now	9503.0	Buy Now
Redmi 9 (Sporty Orange, 4GB RAM, 64GB Storage) | 5000 mAh| 2.3GHz Mediatek Helio G35 Octa core Processor	8799	Buy Now	9189.0	Buy Now	9503.0	Buy Now

Fig 3.7 Web App

4. Technology stack

Element	**Description**
Python	Programming language
BeautifulSoup	Python library used for web scraping
Requests	Python library used for making HTTP requests
Selenium	Used to automate web application testing
Pandas	Python library used for data manipulation & analysis
Elasticsearch	Full-text search and analytics engine
Kibana	Used for visualizing Elasticsearch data
Postman	API client helps to build, test and modify APIs
Flask	Web framework for python. Used for making web applications and HTTP request management
Figma	UI design tool
HTML, CSS	Used in website designing

5. Conclusion

- We were able to collect data from online sources and could pin point data that needs to be extracted. This technique can be used to extract any type of data from the internet.
- We were able to store the data in a csv file to consume later which can be used to create data files.
- We had to perform lot of data cleanup and data mapping which helps us in building and maintain datasets.
- We were able to extract synonyms and relevant data that are more articulate.
- We were able to use elastic search to search, get the data indexed, and then query the data as per our needs.
- We built an API to communicate with the NO SQL database.
- We were able to test the data with Postman and check if we were able to get the data from the database.

6. Future scope

- We can try and answer a lot of questions for which data is present online we can extract data, classify data, clean the data and then perform logical calculations to get more articulate data and try to answer a specific question.
- We can keep tabs of metrics and update them daily wherein we would have to get the data from multiple sources clean the data and update it to the dataset regularly with a script run regularly.
- We can create a website offering the service of comparing multiple products when scaled for quantity and add more features when we can try to scale for quality.
- We can add data like aggregate reviews, individual reviews, or even the image of the product.
- We can get data from various coupon websites, and add that data to factor the price. This helps potential buyers to get more offer and take informed decisions.

References

Web Scraping :

https://www.geeksforgeeks.org/implementing-web-scraping-python-beautiful-soup/

https://www.youtube.com/watch?v=uufDGjTuq34

https://www.youtube.com/watch?v=_AeudsbKYG8

FuzzyWuzzy :

https://pypi.org/project/fuzzywuzzy/

https://www.youtube.com/watch?v=6i5bHQZ-c5U

https://youtu.be/SoZ1CVU2DdE

https://www.datacamp.com/community/tutorials/fuzzy-string-python

Elastic Search :

https://www.youtube.com/watch?v=C-JKcMM6IXE

https://www.youtube.com/watch?v=QALZGdADb5k

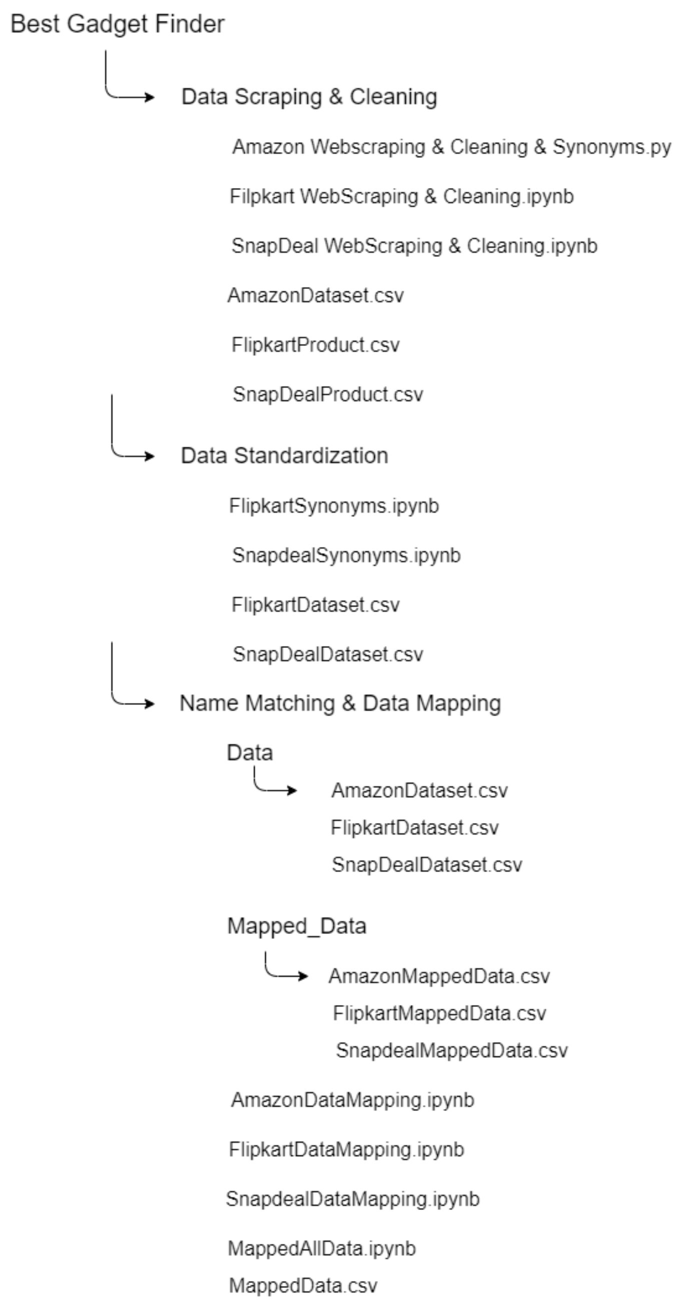
API development:

https://programminghistorian.org/en/lessons/creating-apis-with-python-and-flask

https://www.youtube.com/watch?v=Jzv3G5iDLvw

https://youtu.be/3drtMrcfssk

Directory structure



```
Best Gadget Finder
    └→ Data Scraping & Cleaning
           Amazon Webscraping & Cleaning & Synonyms.py
           Filpkart WebScraping & Cleaning.ipynb
           SnapDeal WebScraping & Cleaning.ipynb
           AmazonDataset.csv
           FlipkartProduct.csv
           SnapDealProduct.csv
    └→ Data Standardization
           FlipkartSynonyms.ipynb
           SnapdealSynonyms.ipynb
           FlipkartDataset.csv
           SnapDealDataset.csv
    └→ Name Matching & Data Mapping
           Data
             └→ AmazonDataset.csv
                FlipkartDataset.csv
                SnapDealDataset.csv
           Mapped_Data
             └→ AmazonMappedData.csv
                FlipkartMappedData.csv
                SnapdealMappedData.csv
           AmazonDataMapping.ipynb
           FlipkartDataMapping.ipynb
           SnapdealDataMapping.ipynb
           MappedAllData.ipynb
           MappedData.csv
```

```
└──> ElasticSearch & Rest API
         MappedData.csv
         DataPreparation.ipynb
         FinalProductDataset.csv
         ElasticSearch and RestAPI.ipynb
└──> Data Testing
         TestDataset.csv
└──> UI Development
         templates
             └──> result.html
                  search.html
         app.ipynb
```