INTRODUCTION-PROJECT OVERVIEW

- **Objective**: Scrape car details from Car24.
- **Steps:**

Set Up Environment: Install necessary libraries (requests, BeautifulSoup, pandas).

Inspect Website: Identify HTML tags and classes for target data.

Write Scraper: Fetch and parse HTML, extract car details.

Save Data: Organize and export data to CSV.

- **Outcome**: Successfully extracted and saved car data, enhancing web scraping and data manipulation skills.

METHODOLOGIES

- **Definition**: Web scraping is the automated process of extracting data from website.
- **Data Sources:** Data is gathered from Cars24, focusing on Ford cars available in Mumbai and Delhi, ensuring relevant and localized insights.
- **Tools Used**: Common tools include Python libraries like requests and BeautifulSoup.
- **Applications**: Used for data mining, market research, price monitoring, and more.
- **Challenges**: Includes handling dynamic content, managing request limits, and adhering to legal and ethical guidelines.

INPUT CODE

```
import pandas as pd
from bs4 import BeautifulSoup
import requests

url = 'https://www.cars24.com/buy-used-car?f=make%3A%3D%3Aford&sort=bestmatch&serveWarrantyCount=true&gaId=81591696.1720131812&storeCityId=2378'

mumbai_cars = requests.get(url)

soup = BeautifulSoup(mumbai_cars.content, 'html.parser')

model = soup.find_all('div', class_ = '_2YB7p')

len(model)

models = []
fuel_type = []
kms_used = []
car_type = []
service = []
emi = []
price = []
location = []

for i in model:
    models.append(i.find('h3',class_ = '_11dVb').text),
    kms_used.append(i.find_all('li')[0].text),
    fuel_type.append(i.find_all('li')[2].text),
    car_type.append(i.find_all('li')[4].text),
    service.append(i.find('span', class_ = '_3JoYA').text),
    emi.append(i.find('span', class_ = '_2O0yU').text),
    price.append(i.find('strong', class_ = '_3RL-I').text),
    location.append('Mumbai')


url1 = 'https://www.cars24.com/buy-used-car?f=make%3A%3D%3Aford&sort=bestmatch&serveWarrantyCount=true&gaId=81591696.1720131812&storeCityId=2'

delhi_cars = requests.get(url1)

soup1 = BeautifulSoup(delhi_cars.content, 'html.parser')

model1 = soup1.find_all('div', class_ = '_2YB7p')

len(model1)

for i in model1:
    models.append(i.find('h3',class_ = '_11dVb').text),
    kms_used.append(i.find_all('li')[0].text),
    fuel_type.append(i.find_all('li')[2].text),
    car_type.append(i.find_all('li')[4].text),
    service.append(i.find('span', class_ = '_3JoYA').text),
    emi.append(i.find('span', class_ = '_2O0yU').text),
    price.append(i.find('strong', class_ = '_3RL-I').text),
    location.append('New Delhi')

dict1 = {'Year': models, 'KM':kms_used, 'Fuel': fuel_type, 'Transmission': car_type, 'Feature':service, 'EMI_per_month':emi, 'Price':price, 'Location':location}

data = pd.DataFrame(dict1)
data.head()
data.to_csv('Cars24_web_scrap.csv', index = False)
data.shape
data.info()
data.isna().sum()
data.describe()
data.sample(5)
cars = pd.DataFrame()
cars = data[['KM','Year','Fuel','Transmission','Price']]

cars.KM = cars.KM.apply(lambda x: x.split()[0])
cars.Year = cars.Year.apply(lambda x: x.split()[0])
```

INPUT CODE

```
cars.KM = cars.KM.apply(lambda x: x.split()[0])
cars.Year = cars.Year.apply(lambda x: x.split()[0])
cars.Price = cars.Price.apply(lambda x: x.split()[0])

cars.KM = cars.KM.str.replace(',','')
cars.Price = cars.Price.str.replace('₹', '')

cars[['KM','Year','Price']] = cars[['KM','Year','Price']].astype(float)

import matplotlib.pyplot as plt
import seaborn as sns

cars.sample(3)plt.title('Histogram of Kilometers driven')
sns.histplot(cars.KM, bins=10, kde=True)

plt.title("Car Counts by Year")
sns.countplot(data = cars, x='Year')

sns.countplot(data = cars, x='Transmission')

cars.groupby(['Transmission','Fuel']).mean()

plt.title("Fuel Type Distribution")
cars.Fuel.value_counts().plot(kind='pie',autopct='%1.1f"%%')

sns.scatterplot(data = cars, x='KM', y='Price', hue='Transmission')
```

OUTPUT

```
cars.head(3)
```

	KM	Year	Fuel	Transmission	Price
0	27,102	2017	Petrol	Manual	₹5.87
1	45,779	2015	Petrol	Manual	₹4.27
2	79,659	2017	Diesel	Manual	₹6.52

```
cars.dtypes
```

```
KM              object
Year            object
Fuel            object
Transmission    object
Price           object
dtype: object
```

- The data on the left showws the details of cars as scrapped from the cars_24 website. The code implementation can be viewed on the previous slides.
- The scrapped shows about the Km, year of manufacturing, fuel type, transmission, price of cars listed on that.
- The lower output shows the data type of each object that we have scrapped earlier.

DATA ANALYSIS

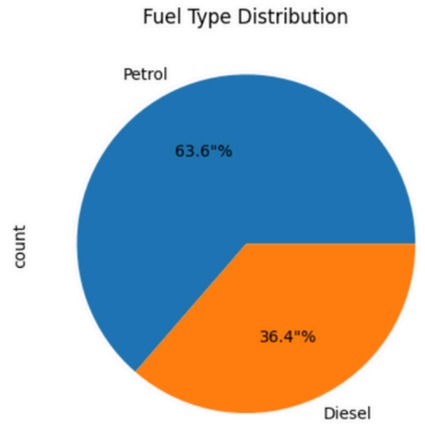
- **Distribution plot** visually assess the distribution of sample data by comparing the empirical distribution of the data with the theoretical values expected from a specified distribution.

- A **scatter plot** (aka scatter chart, scatter graph) uses dots to represent values for two different numeric variables. The position of each dot on the horizontal and vertical axis indicates values for an individual data point. Scatter plots are used to observe relationships between variables.

Distribution Plot

```
[34]: plt.title("Fuel Type Distribution")
      cars.Fuel.value_counts().plot(kind='pie',autopct='%1.1f"%%')
```

```
[34]: <Axes: title={'center': 'Fuel Type Distribution'}, ylabel='count'>
```

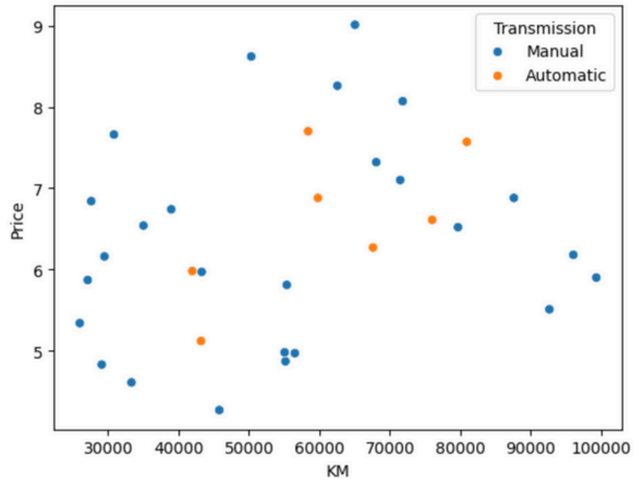



```
182]: # As we can see above that the majority of cars, precisely(61.5%) belongs to "petrol" when it comes to fuel
```

Scatter Plot

```
[35]: sns.scatterplot(data = cars, x='KM', y='Price', hue='Transmission')
```

```
[35]: <Axes: xlabel='KM', ylabel='Price'>
```



DATA ANALYSIS

- A **histogram** is a graph that shows the frequency of numerical data using rectangles. The height of a rectangle (the vertical axis) represents the distribution frequency of a variable (the amount, or how often that variable appears)

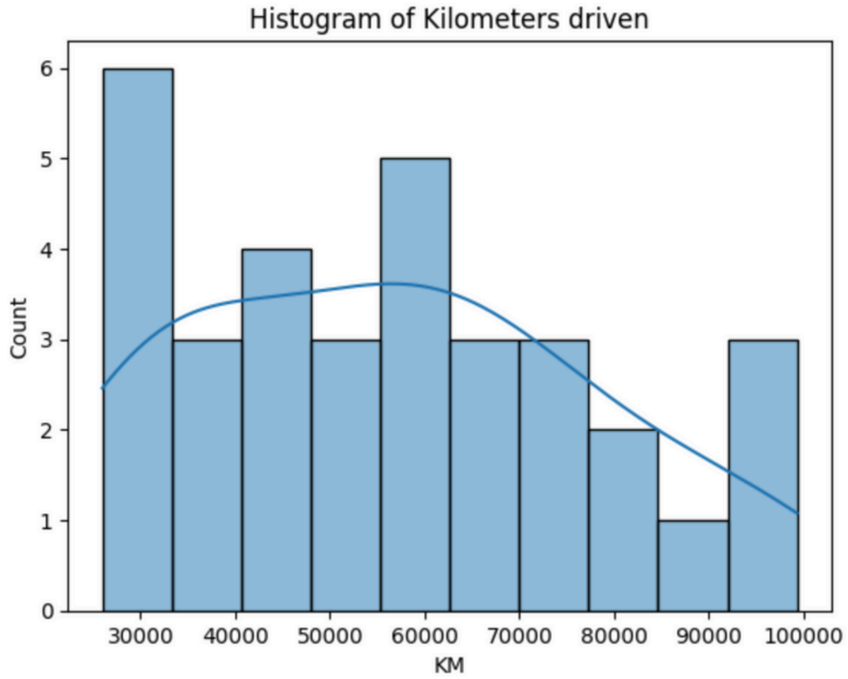
```
cars.sample(3)
```

	KM	Year	Fuel	Transmission	Price
16	99291.0	2018.0	Petrol	Manual	5.90
22	75985.0	2018.0	Petrol	Automatic	6.61
32	80913.0	2018.0	Petrol	Automatic	7.57

```
# Histogramplot

plt.title('Histogram of Kilometers driven')
sns.histplot(cars.KM, bins=10, kde=True)
```

```
<Axes: title={'center': 'Histogram of Kilometers driven'}, xlabel='KM', ylabel='Count'>
```

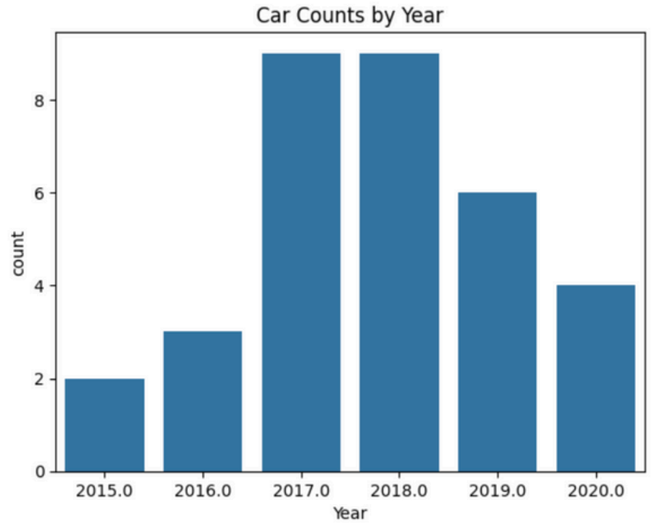


DATA ANALYSIS

- **Count Plot -** Show the counts of observations in each categorical bin using bars. A count plot can be thought of as a histogram across a categorical, instead of quantitative, variable. The basic API and options are identical to those for barplot() , so you can compare counts across nested variables.

```
plt.title("Car Counts by Year")
sns.countplot(data = cars, x='Year')
```

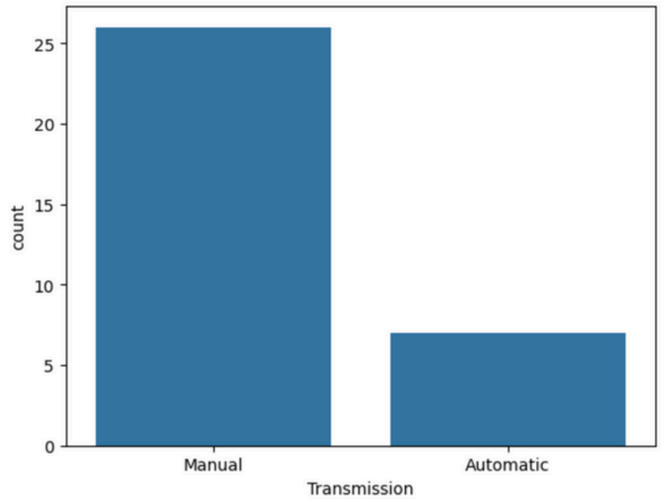
```
<Axes: title={'center': 'Car Counts by Year'}, xlabel='Year', ylabel='count'>
```




```
# From the result we could see that the cars manufactured in 2017 was present in our dataset more whereas 2019
```

```
sns.countplot(data = cars, x='Transmission')
```

```
<Axes: xlabel='Transmission', ylabel='count'>
```



EXPERIENCE GAINED

- **TECHNICAL SKILLS :** Working on a web scraping project to extract car details from Car24 has been an enriching experience. We gained practical skills in Python libraries such as requests for fetching web content and BeautifulSoup for parsing HTML. This project required a deep dive into HTML structure, enhancing our ability to inspect and identify the necessary tags and classes to extract data accurately.
- **ANALYTICAL SKILLS:** Analyzing the scraped data provided valuable insights into pricing trends and popular Ford models in Mumbai and Delhi. By identifying patterns and anomalies, we could draw meaningful conclusions about market dynamics and consumer preferences. This analytical approach helped us in recognizing the factors influencing car prices and demand, enabling data-driven decision-making.
- **PROJECT MANAGEMENT:** Managing the web scraping project from start to finish honed our project management skills. We developed project management skills by defining scope, setting timelines, ensuring data quality, and adapting to challenges. This experience emphasized the significance of planning, monitoring, and communication for successful project delivery.

TECHNICAL CHALLENGES

- **DATA QUALITY** - In our project, a key data quality issue was that all data were in object format, causing significant problems. Before any meaningful analysis could be conducted, we had to convert numerical columns from object to int and float data type.
- **FEATURE EXTRACTION** - We extracted year of manufacture from the car_model column as on site there wasn't any specified detail provided for that.

TIME MANAGEMENT STRATEGIES

- **Efficient Scripting** -We wrote optimized scripts to reduce scraping time and improve data extraction efficiency.
- **Parallel Processing** - We implemented parallel processing techniques to accelerate the scraping process by running multiple scripts simultaneously.
- **Data Storage Optimization** - We used efficient data storage solutions, such as relational databases, to minimize storage overhead and facilitate data analysis.

CONCLUSION

SUMMARY: The analysis of Ford cars listed on Cars24 in Mumbai and Delhi revealed key insights into the market. The dataset consisted of information on kilometers driven, year of manufacture, fuel type, transmission, and price. These variables provided a comprehensive understanding of the current state of Ford cars in these major cities.

KEY - TAKEAWAYS: This project provides valuable insights into market trends and consumer behavior by analyzing the demand and pricing patterns of Ford cars in Mumbai and Delhi. Understanding these trends helps in making data-driven decisions, improving inventory management, and tailoring marketing strategies to meet consumer preferences.