Business Objective

A C2B firm is in the process of creating an Online platform where the consumers can sell their used Laptops. This platform should host an automated mechanism that can suggest to a consumer the realistic price of their used laptop when the required details (different features of the laptop) are provided.

Importing required libraries

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
os.chdir(r'C:\Users\V.SaranReddy\OneDrive\Desktop')
```

In [3]:

```
os.listdir()
```

Out[3]:

```
['Airtel Money.xlsx',
 'all csv files',
 'dataset (1).xlsx',
 'desktop.ini',
 'Eclipse IDE for Java Developers - 2021-03.lnk',
 'educational docs',
 'goto meeting screenshots',
 'GoToMeeting 000.png',
 'images',
 'index.html',
 'index2.html',
 'index3.html',
 'index4.html',
 'index5.html',
 'Jupyter Notebook (anaconda3).lnk',
 'Lecture 01 Material-20220306.zip',
 'Microsoft Teams.lnk',
 'module 3',
 'module2 lectures',
 'New Text Document.txt',
 'notepad module3.txt',
 'Outlook (PWA).lnk',
 'Recharge.xlsx',
 'salesforce files',
 'Saran Advanced.doc',
 'Spyder (anaconda3).lnk',
 'Tharun resume.docx',
 'UG 3-2 R19 EEE Syllabus (2).pdf',
 'VELURU SARAN -educational docs.pdf',
 'WhatsApp Desktop.lnk',
 'Zoom.lnk',
 '[Dataset 1]supermarket_sales.csv',
 '~$RD0004.tmp']
```

Reading Dataset

In [4]:

```
df=pd.read_excel('dataset (1).xlsx')
df.head()
```

Out[4]:

	laptop_ID	Company	Product	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight
0	1	Apple	MacBook Pro	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37kg
1	2	Apple	Macbook Air	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34kg
2	3	HP	250 G6	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8GB	256GB SSD	Intel HD Graphics 620	No OS	1.86kg
3	4	Apple	MacBook Pro	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16GB	512GB SSD	AMD Radeon Pro 455	macOS	1.83kg
4	5	Apple	MacBook Pro	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8GB	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37kg

In [5]:

```
df.shape
```

Out[5]:

```
(1303, 13)
```

In [6]:

```
df.columns
```

Out[6]:

```
Index(['laptop_ID', 'Company', 'Product', 'TypeName', 'Inches',
       'ScreenResolution', 'Cpu', 'Ram', 'Memory', 'Gpu', 'OpSys', 'Weight',
       'Price_euros'],
      dtype='object')
```

In [ ]:

```

```

In [ ]:

```

```

Checking Missing values and Data type of variables

In [7]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   laptop_ID         1303 non-null   int64 
```

```
 1   Company           1303 non-null   object
 2   Product           1303 non-null   object
 3   TypeName          1303 non-null   object
 4   Inches            1303 non-null   float64
 5   ScreenResolution  1303 non-null   object
 6   Cpu               1303 non-null   object
 7   Ram               1303 non-null   object
 8   Memory            1303 non-null   object
 9   Gpu               1303 non-null   object
 10  OpSys             1303 non-null   object
 11  Weight            1303 non-null   object
 12  Price_euros       1303 non-null   float64
dtypes: float64(2), int64(1), object(10)
memory usage: 132.5+ KB
```

In [8]:

```
df.isnull().sum()
```

Out[8]:

```
laptop_ID           0
Company             0
Product             0
TypeName            0
Inches              0
ScreenResolution    0
Cpu                 0
Ram                 0
Memory              0
Gpu                 0
OpSys               0
Weight              0
Price_euros         0
dtype: int64
```

There are no missing values in the data and variables Ram and Weight read as object type due to presence of units GB and kg

Checking duplicated rows

In [9]:

```
df.duplicated().sum()
```

Out[9]:

```
0
```

There are no duplicate rows in the data

Removing Unwanted columns

In [10]:

```
df['Product'].unique()
```

Out[10]:

```
array(['MacBook Pro', 'Macbook Air', '250 G6', 'Aspire 3',
       'ZenBook UX430UN', 'Swift 3', 'Inspiron 3567', 'MacBook 12"',
       'IdeaPad 320-15IKB', 'XPS 13', 'Vivobook E200HA',
       'Legion Y520-15IKBN', '255 G6', 'Inspiron 5379',
       '15-BS101nv (i7-8550U/8GB/256GB/FHD/W10)', 'MacBook Air',
       'Inspiron 5570', 'Latitude 5590', 'ProBook 470', 'LapBook 15.6"',
       'E402WA-GA010T (E2-6110/2GB/32GB/W10)',
       '17-ak001nv (A6-9220/4GB/500GB/Radeon', 'IdeaPad 120S-14IAP',
       'Inspiron 5770', 'ProBook 450',
```

'X540UA-DM186 (i3-6006U/4GB/1TB/FHD/Linux)', 'Inspiron 7577',
'X542UQ-GO005 (i5-7200U/8GB/1TB/GeForce', 'Aspire A515-51G',
'Inspiron 7773', 'IdeaPad 320-15ISK', 'Rog Strix',
'X751NV-TY001T (N4200/4GB/1TB/GeForce', 'Yoga Book', 'ProBook 430',
'Inspiron 3576', '15-bs002nv (i3-6006U/4GB/128GB/FHD/W10)',
'VivoBook Max', 'GS73VR 7RG',
'X541UA-DM1897 (i3-6006U/4GB/256GB/FHD/Linux)', 'Vostro 5471',
'IdeaPad 520S-14IKB',
'UX410UA-GV350T (i5-8250U/8GB/256GB/FHD/W10)', 'ZenBook Pro',
'Stream 14-AX040wm', 'V310-15ISK (i5-7200U/4GB/1TB/FHD/W10)',
'FX753VE-GC093 (i7-7700HQ/12GB/1TB/GeForce', 'Surface Laptop',
'Inspiron 5370', 'GL72M 7RDX', 'Aspire E5-475',
'FX503VD-E4022T (i7-7700HQ/8GB/1TB/GeForce', 'IdeaPad 320-15IKBN',
'Aspire A515-51G-32MX', 'ProBook 440', 'IdeaPad 320-15AST',
'Pavilion 15-CK000nv', 'FX503VM-E4007T (i7-7700HQ/16GB/1TB',
'FX550IK-DM018T (FX-9830P/8GB/1TB/Radeon', 'Aspire 5',
'Probook 430', 'Zenbook UX430UA', 'Spin 5',
'X541UV-DM1439T (i3-7100U/6GB/256GB/GeForce', 'Omen 15-ce007nv',
'15-bs017nv (i7-7500U/8GB/256GB/Radeon',
'15-bw000nv (E2-9000e/4GB/500GB/Radeon', 'Envy 13-ad009n',
'Pavilion 14-BK001nv', 'Ideapad 310-15ISK',
'UX430UQ-GV209R (i7-7500U/8GB/256GB/GeForce', 'GP62M 7REX',
'Thinkpad T470', 'VivoBook S15', 'ThinkPad Yoga', 'Probook 440',
'Spectre x360', 'Inspiron 7570',
'X705UV-BX074T (i3-6006U/4GB/1TB/GeForce', 'Spin 3', 'GS63VR 7RG',
'Probook 470', 'E402WA-GA007T (E2-6110/4GB/64GB/W10',
'Inspiron 5567', 'Aspire A515-51G-37JS',
'15-BS078nr (i7-7500U/8GB/1TB/W10)',
'V110-15IAP (N3350/4GB/1TB/No',
'FX753VD-GC086T (i5-7300HQ/8GB/1TB', 'Envy 13-AD007nv',
'ThinkPad E480', 'Satellite Pro', 'ZenBook UX430UA',
'EliteBook Folio', 'X541NA (N3350/4GB/1TB/FHD/W10)', 'GE72MVR 7RG',
'Aspire A315-51', 'Inspiron 5577', 'Inspiron 7567',
'V110-15IKB (i5-7200U/4GB/128GB/W10)', 'GE73VR 7RE',
'EliteBook 840', '15-BS103nv (i5-8250U/6GB/256GB/Radeon',
'Yoga 520-14IKB', 'ZenBook Flip', 'Inspiron 5579',
'X555BP-XX180T (A9-9420/4GB/1TB/Radeon', 'Aspire A517-51G',
'Aspire A315-31', 'GE63VR 7RE', 'MateBook X',
'17-bs001nv (i5-7200U/6GB/2TB/Radeon', 'GT80S 6QF-074US',
'V310-15IKB (i5-7200U/8GB/1TB', 'Yoga 920-13IKB', 'Mi Notebook',
'XPS 15', 'Swift 7', 'Thinkpad Yoga',
'K147 (N3350/4GB/32GB/FHD/W10)', 'IdeaPad 320-17IKBR', 'Blade Pro',
'Omen 17-W295', 'V110-15ISK (i5-6200U/4GB/128GB/W10)',
'Aspire E5-576G', 'Legion Y720-15IKB', 'Precision 7520',
'Aspire 7', 'ROG GL703VD-GC028T',
'15-bs018nq (i3-6006U/4GB/500GB/FHD/No', 'IdeaPad 320-17IKB',
'Latitude 5490', 'Portege Z30-C-16L', 'Alienware 17',
'Vivobook X541UV-DM1217T', 'K756UX-T4340T (i5-7200U/8GB/500GB',
'ZBook 15u', 'Pro P2540UA-XO0198T',
'15-rb013nv (E2-9000e/4GB/500GB/W10)', 'Vostro 5468', 'Aspire R7',
'X555QG-DM242T (A10-9620P/4GB/1TB', 'ROG G703VI-E5062T',
'Nitro AN515-51', 'VivoBook Pro',
'F756UX-T4201D (i7-7500U/8GB/128GB', 'Yoga 910-13IKB',
'15-bs015dx (i5-7200U/8GB/1TB/W10)', 'Rog G701VIK-BA060T',
'ROG G752VSK-GC493T', 'X505BP-BR019T (A9-9420/4GB/1TB/Radeon',
'Vostro 5370', '15-BW094nd (A6-9220/8GB/128GB/W10)',
'Envy 17-U275cl', 'GT73EVR 7RE', 'Yoga 720-15IKB', 'Vostro 3568',
'V330-15IKB (i7-8550U/8GB/256GB/FHD/W10)', 'ThinkPad X1',
'IdeaPad 320-17ISK', 'Ideapad 320-15IKBN',
'SP315-51 (i7-7500U/12GB/1TB/FHD/W10)', 'Thinkpad T570',
'Chromebook C910-C2ST',
'FX753VD-GC071T (i7-7700HQ/8GB/1TB/GeForce',
'17-BS037cl (i3-6006U/8GB/1TB/W10)',
'V330-15IKB (i5-8250U/8GB/256GB/FHD/W10)', 'Aspire A715-71G',
'Precision 7720', 'IdeaPad 310-15ABR', 'ZenBook UX530UQ-PRO',
'VivoBook S14', 'Rog GL702VS-GC095T',
'GL553VE-FY082T (i7-7700HQ/8GB/1TB', 'IdeaPad 320-15IAP',
'EliteBook x360', 'IdeaPad 720S-13IKB', 'GE63VR 7RF',
'ES1-523-84K7 (A8-7410/8GB/256GB/FHD/W10)', 'VivoBook Flip',
'ThinkPad 13', 'ProBook 640', 'TravelMate B', 'Elitebook 840',
'ZenBook UX410UA-GV183T', 'Aspire E5-575', 'Elitebook 820',
'GL72M 7REX', 'UX510UX-CN269T (i7-7500U/8GB/256GB',

'V310-15ISK (i3-6006U/4GB/1TB/FHD/W10)',
'FX553VD-FY647T (i7-7700HQ/8GB/256GB/GeForce', 'Elitebook 850',
'X541NA (N3350/4GB/1TB/Linux)', 'Inspiron 3552',
'IdeaPad 320-15ABR', 'Stream 14-AX001nv', 'GP72MVR 7RFX',
'Zbook 15', 'Tecra A50-C-21G', 'Latitude 7480',
'Zenbook UX410UA-GV027T', '15-AY023na (N3710/8GB/2TB/W10)',
'Elitebook 1040', 'IdeaPad 110-17ACL',
'15-bw003nv (A9-Series-9420/4GB/256GB/FHD/W10)', 'Yoga 11e',
'VivoBook E403NA', 'Omen 17-w212nv',
'V310-15ISK (i3-6006U/4GB/128GB/FHD/No', 'ROG Strix',
'IdeaPad 720S-14IKB', 'Zenbook Flip', 'Thinkpad X1',
'Ideapad 510S-13IKB', 'Precision 3510', 'Precision 5520',
'Rog GL753VD-GC042T', 'Rog GL753VE-GC070T', 'Leopard GP72M',
'15-BW004nv (A9-9420/4GB/256GB/Radeon', 'ThinkPad E580',
'ThinkPad L470', 'Precision M5520',
'FX753VD-GC461T (i7-7700HQ/16GB/1TB', 'GE73VR 7RF', 'Zenbook 3',
'Portege Z30-C-16P', 'Lenovo IdeaPad', 'ThinkPad P51',
'Thinkpad T470p', '15-BS028nv (i3-6006U/4GB/1TB/Radeon',
'Latitude 3380', 'EliteBook 1040', 'LapBook 12.3', 'ProBook 650',
'X542UQ-DM117 (i3-7100U/8GB/1TB/GeForce', 'Latitude 5480',
'Omen 17-w207nv', 'FlexBook Edge', 'Chromebook 3', 'Thinkpad 13',
'IdeaPad 320s-14IKB', 'Thinkpad P51',
'15-ra044nv (N3060/4GB/500GB/W10)', 'Pixelbook (Core',
'ThinkPad T470s', 'ThinkPad X270', 'Omen 15-AX205na',
'Aspire ES1-572', 'Precision 3520', 'GV62 7RD-1686NL',
'15-bs024nv (i5-7200U/8GB/128GB/W10)', 'ThinkPad T470',
'Inspiron 3168', '17-BS092ND (i3-6006U/8GB/256GB/W10)',
'Pro P2540UA-AB51', 'IdeaPad 510s-14IKB',
'X541NA-PD1003Y (N4200/4GB/500GB/W10)', 'Omen 17-an006nv',
'Thinkpad T460s', 'Latitude 7390', 'Latitude E5470',
'Portege X30-D-10J', 'Lapbook 15,6', 'ThinkPad E570',
'Thinkpad X270', 'Zenbook UX390UA', 'Thinkpad E570',
'Portege X30-D-10L', 'Rog G752VL-UH71T', 'Thinkpad X260',
'Ideapad 520-15IKBR', 'ThinkPad L570', 'VivoBook E201NA',
'15-BS026nv (i5-7200U/8GB/256GB/Radeon', 'IdeaPad 320-14IAP',
'Chromebook N23', 'ZenBook UX510UX-CN211T', 'Aspire A515-51G-59QF',
'Envy 13-AB002nv', 'Vostro 5568', 'VivoBook E12',
'15-bs190od (i5-8250U/4GB/1TB/W10)', 'ROG Zephyrus', 'Probook 450',
'FX753VE-GC155T (i7-7700HQ/16GB/1TB', 'Spectre X360',
'Latitude 5580', 'Zenbook UX510UW-FI095T', 'SmartBook Edge',
'Omen 15-ce006nv', 'Thinkpad E470', 'Envy 13-AB020nr',
'VivoBook X540YA-XX519T', 'ThinkPad E470',
'V310-15ISK (i5-6200U/4GB/1TB/FHD/No', 'ThinkPad T570',
'17-X047na (i3-6006U/8GB/1TB/W10)',
'A541NA-GO342 (N3350/4GB/500GB/Linux)', 'SmartBook 130',
'15-bw007nv (A10-9620P/6GB/128GB/Radeon', 'Spin SP111-31',
'V330-15IKB (i3-7130U/4GB/128GB/FHD/W10)', 'EliteBook 1030',
'Thinkpad P71', 'FX553VD-DM627T (i5-7300HQ/8GB/1TB',
'Lifebook A557', 'ZBook 17', '14-am079na (N3710/8GB/2TB/W10)',
'15-cd005nv (A9-9420/6GB/256GB/Radeon',
'V330-15IKB (i5-8250U/4GB/500GB/FHD/W10)', 'SmartBook 141',
'Tecra X40-D-10H', 'IdeaPad Y910-17ISK', 'GT73VR Titan',
'Chromebook 11', 'GT80S 6QE', 'Omen 17-AN010nv',
'Ideapad 320-15IKBR', 'TP501UA-CJ131T (i5-7200U/8GB/1TB/W10)',
'Inspiron 3179', 'Notebook Odyssey',
'V320-17ISK (i3-6006U/4GB/500GB/FHD/No', 'IdeaPad 110-15ISK',
'Latitude 5289', 'EliteBook 850', 'Aspire 1', 'Laptop MSI',
'GS63VR 7RF', 'Tecra Z50-C-144', 'IdeaPad 310-15IKB',
'Swift SF114-31-P5HY', 'Inspiron 7559',
'FX753VD-GC007T (i7-7700HQ/8GB/1TB', 'GT62VR 7RE',
'CB5-132T-C9KK (N3160/4GB/32GB/Chrome', 'LifeBook A557',
'SmartBook 140', 'Q304UA-BHI5T11 (i5-7200U/6GB/1TB/FHD/W10)',
'ZenBook 3', 'V330-15IKB (i5-8250U/4GB/256GB/FHD/W10)',
'Ideapad 320-15ISK', 'X541NA-GO414T (N3350/8GB/1TB/W10)',
'IdeaPad 100S-14IBR', '17-AK091ND (A9-9420/8GB/1TB/W10)',
'ROG GL553VE-FY022', 'Extensa EX2540', 'Portege Z30-C-16J',
'ROG G701VI', 'A715-71G-59DH (i5-7300HQ/8GB/1TB/GeForce',
'GL62M 7REX', 'Tecra A50-D-11M', 'IdeaPad Y700-15ISK',
'Latitude E7470', 'Ideapad 320-15IAP',
'15-ay047nv (i3-6006U/6GB/1TB/Radeon', 'GP72VR Leopard',
'Latitude 3580', '15-bs012nv (i7-7500U/8GB/1TB/Radeon',
'Tecra Z50-D-10E', 'V310-15ISK (i5-7200U/8GB/1TB',

'Yoga 720-13IKB', 'Pavilion X360', 'GP62 7RDX', 'Chromebook X360',
'Gram 15Z975', 'Aspire VX5-591G', 'GV62M 7RD',
'L502NA-GO052T (N3350/4GB/128GB/W10)', 'Alienware 15',
'17-bs000nv I3', 'Yoga 730', '17-Y002nv (A10-9600P/6GB/2TB/Radeon',
'V110-15ISK (3855U/4GB/500GB/W10)', 'Chromebook 14',
'IdeaPad 520s-14IKB', 'TravelMate B117-M', 'Chromebook Flip',
'Portege Z30T-C-133', '15-bs011nv (i7-7500U/4GB/500GB/Radeon',
'V310-15IKB (i5-7200U/4GB/1TB/FHD/W10)',
'V310-15ISK (i3-6006U/4GB/500GB/No', 'ThinkPad P51s',
'Thinkpad T460p', '17-ak002nv (A10-9620P/6GB/2TB/Radeon',
'110-15ACL (A6-7310/4GB/500GB/W10)', 'Smartbook 142',
'V310-15IKB (i5-7200U/4GB/1TB/No', 'Inspiron 5378',
'15-BW037na (A9-9420/4GB/1TB/Radeon', 'Predator 17',
'15-BW091ND (A9-9420/6GB/1TB', 'Extensa EX2540-58KR',
'V310-15IKB (i7-7500U/4GB/1TB/FHD/W10)', 'ZBook 15',
'Inspiron 7560', 'Tecra X40-D-10G', 'Flex 5', 'Thinkpad P51s',
'Notebook 9', 'Zbook 17', 'N23 (N3060/4GB/128GB/W10)',
'X550VX-XX015D (i5-6300HQ/4GB/1TB/GeForce', 'Thinkpad T460',
'Pro P2540UA-XO0192R', 'Yoga 900-13ISK',
'15-cb003na (i5-7300HQ/8GB/1TB', 'Latitude 7280',
'Zenbook UX330UA-AH5Q', 'TravelMate P238-M',
'X751NV-TY001 (N4200/4GB/1TB/GeForce', 'Tecra A40-C-1E5',
'EliteBook 820', 'Q524UQ-BHI7T15 (i7-7500U/12GB/2TB/GeForce',
'Thinkpad P50', 'Vivobook Max', 'Rog G752VS-BA171T',
'Tecra Z40-C-161', 'IdeaPad 110-15IBR', 'GS43VR 7RE',
'GL62M (i5-7300HQ/8GB/1TB', 'Predator G9-793',
'FX502VM-DM560T (i7-7700HQ/8GB/1TB', 'K146 (N3350/4GB/32GB/W10)',
'Yoga 510-15IKB', 'R417NA-RS01 (N3350/4GB/32GB/W10)',
'Pro P2540UA-XS51', 'Latitude 3180',
'15-ba043na (A12-9700P/8GB/2TB/W10)', 'Omen 17-an012dx',
'Thinkpad T470s', 'Blade Stealth', 'Latitude 3480',
'V110-15ISK (i3-6006U/4GB/500GB/W10)', 'Tecra X40-D-10Z',
'GL62M 7RD', 'Rog GL702VS-BA023T', 'N42-20 Chromebook',
'R558UA-DM966T (i5-7200U/8GB/128GB/FHD/W10)', 'Rog GL702VM-GC017T',
'ZenBook UX310UQ-GL026T', 'Rog GL502VM-DS74', 'Inspiron 5767',
'ThinkPad T470p', 'K556UR-DM621T (i7-7500U/8GB/256GB/GeForce',
'X541NA (N4200/4GB/1TB/W10)', 'Inspiron 5368', 'Portege X30-D-10X',
'Portégé Z30-C-188',
'TMX349-G2-M-50FS (i5-7200U/8GB/256GB/FHD/W10)', 'Tecra A50-D-11D',
'X541NA-GO121 (N4200/4GB/1TB/Linux)', 'Pavilion x360',
'VivoBook L402NA', 'IdeaPad 510-15ISK', 'Rog GL753VD-GC082T',
'Chromebook C731-C78G', 'Probook 640', 'Envy x360',
'GS73VR Stealth', 'Portege X30-D-10V',
'G701VO-IH74K (i7-6820HK/32GB/2x', 'Gram 15Z970',
'Chromebook CB5-571-C1DZ', 'Gram 14Z970', 'Elitebook Folio',
'IdeaPad 510-15IKB', 'GE72VR 6RF', 'Envy 13-AB077cl',
'Tecra Z50-C-140', 'Probook 650', 'Tecra Z40-C-12X',
'GP62M Leopard', 'Omen 17-W006na',
'X751SV-TY001T (N3710/4GB/1TB/GeForce', 'TravelMate P259-G2',
'Tecra A50-C-1ZV', 'Yoga 700-11ISK', 'IdeaPad Y700-15ACZ',
'Insprion 5767', 'ZBook Studio', 'Portege Z30-C-1CW',
'ProBook x360', 'Chromebook C738T-C2EJ', 'Portege Z30-C-16Z',
'Aspire F5-573G-510L', 'Portege X20W-D-10V', 'Tecra A40-C-1DF',
'ThinkPad T460', 'Q534UX-BHI7T19 (i7-7500U/16GB/2TB',
'15-bs053od (i7-7500U/6GB/1TB/W10)', 'Rog GL753VE-DS74',
'Inspiron 7579', 'Portege Z30-C-1CV', 'LifeBook A556',
'Tecra A40-C-1KF', '15-bs005nv (i3-6006U/4GB/1TB',
'V110-15IAP (N3350/4GB/128GB/No', 'ThinkPad T560',
'ZenBook UX310UA-FB485T', 'Spectre 13-V111dx', 'Aspire ES1-533',
'Rog GL553VE-DS74', 'Nitro 5', 'ENVY -', 'Portege Z30-C-16H',
'Portege A30-C-1CZ', 'ThinkPad P70', 'Tecra Z40-C-12Z',
'Inspiron 5568', 'Portégé Z30-C-16K', 'Spectre 13-V100nv',
'Latitude E5570', 'Tecra Z40-C-136', 'Yoga 500-15ISK',
'V142 (X5-Z8350/2GB/32GB/W10)', 'Tecra A50-C-218', 'Thinkpad L560',
'GT72S Dominator', 'IdeaPad Y900-17ISK', 'Chromebook C202SA',
'Noteb Pav', 'Inspiron 5578', '250 G5', 'Aspire ES1-523',
'Inspiron 7378', 'GT62VR 6RD', 'Rog G752VL-GC088D', 'GS63VR 6RF',
'ROG G701VO', 'Latitude 3570', 'IdeaPad 300-17ISK',
'Ideapad 700-15ISK', 'GT72VR Dominator',
'V110-15ISK (i5-6200U/4GB/500GB/W10)', 'Yoga 900S-12ISK',
'Chromebook 13', 'Rog GL702VM-GC354T', 'Aspire F5-573G',
'GS70 Stealth', 'G752VY-GC162T (i7-6700HQ/16GB/1TB',

```
       'Latitude E5270', 'Chromebook 15', 'GE72 Apache',
       '15-bw011nv (A6-9220/4GB/1TB/FHD/W10)', 'Rog GL552VW-CN470T',
       'Vostro 3559', 'V110-15ISK (i3-6006U/4GB/128GB/W10)',
       'Spectre Pro', 'Portege X30-D-10K', 'Rog GL752VW-T4308T',
       'V131 (X5-Z8350/4GB/32GB/FHD/W10)', 'Omen -',
       '15-bs078cl (i7-7500U/8GB/2TB/W10)', 'ThinkPad P40',
       'L403NA-GA013TS (N3350/4GB/32GB/W10)', 'IdeaPad 500-15ISK',
       'GP62M 7RDX', 'V110-15ISK (i3-6006U/4GB/1TB/No',
       '15-BA015wm (E2-7110/4GB/500GB/W10)',
       'B51-80 (i5-6200U/8GB/1TB/Radeon',
       '15-bw002nv (A6-9220/4GB/256GB/Radeon', 'GP72M 7REX',
       'ThinkPad T460s', 'B51-80 (i5-6200U/8GB/1008GB/Radeon',
       'GS40 Phantom', 'Pavilion 15-cb003nv', 'IdeaPad 310-15ISK',
       '250 G4', '320-15ISK (i3-6006U/4GB/1TB/GeForce',
       'Stream 14-AX000nv', 'PL60 7RD',
       'X553SA-XX021T (N3050/4GB/500GB/W10)',
       'V110-15ISK (i5-6200U/4GB/500GB/No',
       'UX410UA-GV097T (i3-7100U/4GB/256GB/FHD/W10)',
       'B51-80 (i7-6500U/4GB/1008GB/FHD/W7)', 'GS60 Ghost',
       'Pavilion 15-BC000nv', 'Rog GL552VW-DM201T', 'Chromebook Plus',
       'Pavilion Power', 'V110-15ISK (i3-6006U/4GB/1TB/Radeon',
       'Rog G752VY-GC229T', 'GS73VR 7RF',
       'FX502VM-DM105T (i7-6700HQ/8GB/1TB/GeForce',
       '15-bs025nv (i5-7200U/8GB/256GB/W10)', 'Aspire E5-774G',
       'FX502VM-AS73 (i7-7700HQ/16GB/1TB',
       'C740-C9QX (3205U/2GB/32GB/Chrome', 'E5 774G',
       'SP714-51 (i7-7Y75/8GB/256GB/FHD/W10)', 'Thinkpad T560',
       'GP62MVR 6RF', '15-bw009nv (A12-9720P/6GB/1TB/Radeon',
       'Latitude E7270', 'X540SA-RBPDN09 (N3710/4GB/1TB/W10)',
       'GL62M 7RDX', 'GE72VR Apache',
       '15-bs023nv (i3-6006U/4GB/1TB/FHD/W10)', 'GL62 6QF',
       'ZenBook UX310UA-WB71', 'Inspiron 7779', 'Rog GL553VE-FY052T',
       'Rog GL502VS', 'V510-15IKB (i5-7200U/8GB/256GB/FHD/No',
       'ThinkPad L460', 'X541NA-GO020T (N3350/4GB/1TB/W10)',
       'Rog G752VT-GC073T', 'B51-80 (i7-6500U/8GB/1008GB/Radeon',
       'GE62 Apache', 'Yoga 500-14IBD', 'ZenBook UX305CA-UBM1',
       'Aspire ES1-531', 'Pavilion 15-AW003nv', 'Stream 11-Y000na',
       'X556UJ-XO044T (i7-6500U/4GB/500GB/GeForce', 'Yoga 500-14ISK',
       '15-AC110nv (i7-6500U/6GB/1TB/Radeon',
       'X553SA-XX031T (N3050/4GB/500GB/W10)'], dtype=object)
```

In [11]:

```
df.laptop_ID.unique()
```

Out[11]:

```
array([   1,    2,    3, ..., 1318, 1319, 1320], dtype=int64)
```

In [12]:

```
df.drop(['laptop_ID','Product'],axis=1,inplace=True)
```

In [13]:

```
df.head(3)
```

Out[13]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37kg	1339.69
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34kg	898.94
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8GB	256GB SSD	Intel HD Graphics 620	No OS	1.86kg	575.00

In [ ]:

```
```

Data Pre-processing

Converting Ram variable into integer type and Weight variable into float type

In [14]:

```
df['Ram']=df['Ram'].str.replace('GB','')
df['Weight']=df['Weight'].str.replace('kg','')
```

In [15]:

```
df.head()
```

Out[15]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	1339.69
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	898.94
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	575.00
3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	2537.45
4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	1803.60

In [16]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Company           1303 non-null   object 
 1   TypeName          1303 non-null   object 
 2   Inches            1303 non-null   float64
 3   ScreenResolution  1303 non-null   object 
 4   Cpu               1303 non-null   object 
 5   Ram               1303 non-null   object 
 6   Memory            1303 non-null   object 
 7   Gpu               1303 non-null   object 
 8   OpSys             1303 non-null   object 
 9   Weight            1303 non-null   object 
 10  Price_euros       1303 non-null   float64
dtypes: float64(2), object(9)
memory usage: 112.1+ KB
```

In [17]:

```
df['Ram']=df.Ram.astype('int64')
```

```
df['Weight']=df.Weight.astype('float64')
```

In [18]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Company           1303 non-null   object 
 1   TypeName          1303 non-null   object 
 2   Inches            1303 non-null   float64
 3   ScreenResolution  1303 non-null   object 
 4   Cpu               1303 non-null   object 
 5   Ram               1303 non-null   int64  
 6   Memory            1303 non-null   object 
 7   Gpu               1303 non-null   object 
 8   OpSys             1303 non-null   object 
 9   Weight            1303 non-null   float64
 10  Price_euros       1303 non-null   float64
dtypes: float64(3), int64(1), object(7)
memory usage: 112.1+ KB
```

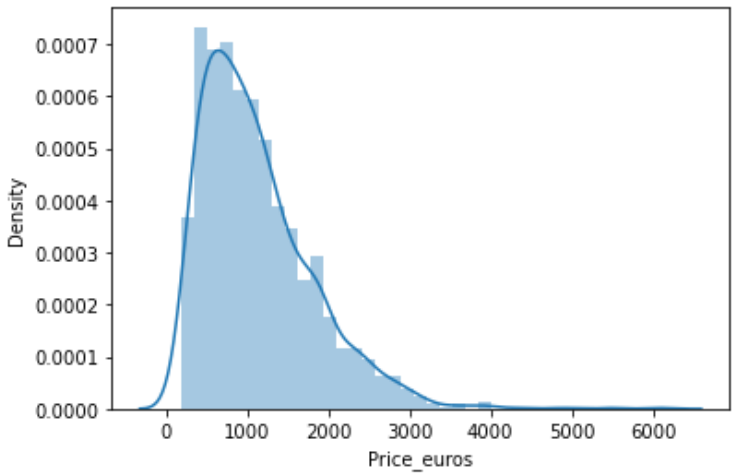
In [ ]:

```

```

Exploratory Data Analysis

In [19]:

```
sns.distplot(df['Price_euros']);
```

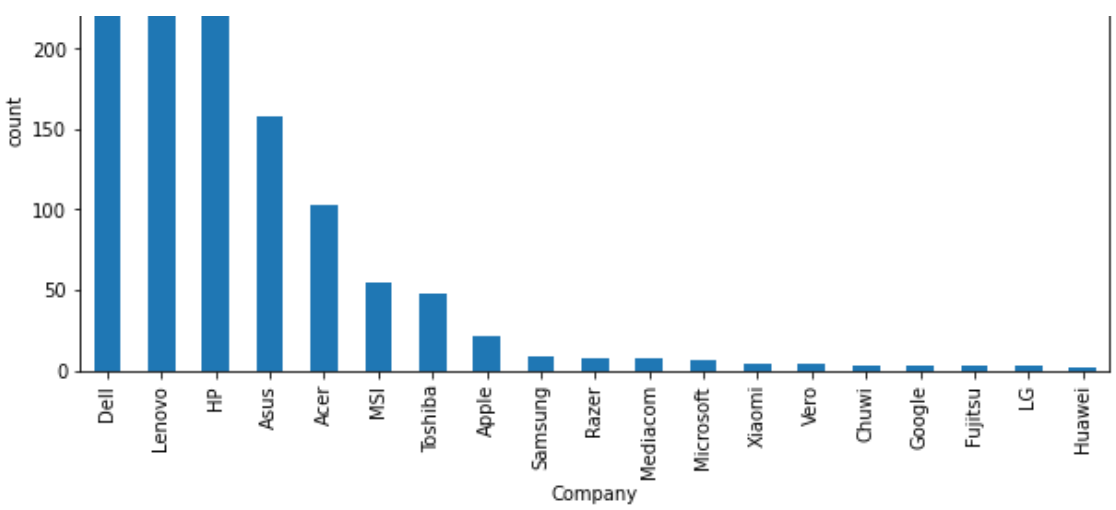


We observe that Price_euros variable is right-skewed

Analysis on Company variable

In [20]:

```
plt.figure(figsize=(10,5))
plt.xlabel('Company')
plt.ylabel('count')
df.Company.value_counts().plot(kind='bar');
```



In [21]:

```
df.Company.value_counts(normalize=True)
```

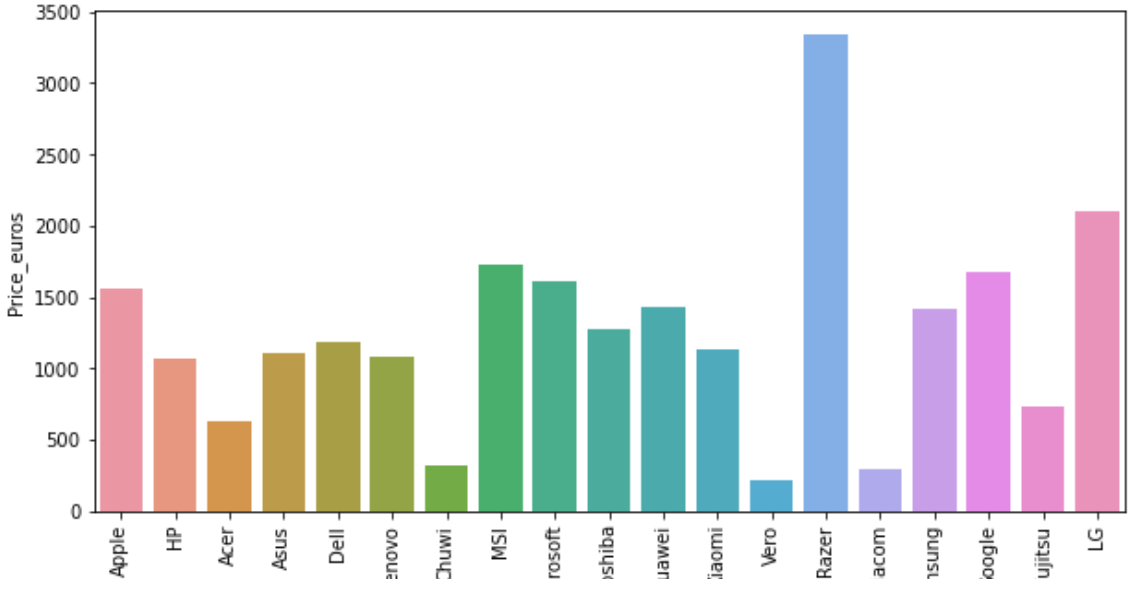
Out[21]:

```
Dell          0.227936
Lenovo        0.227936
HP            0.210284
Asus          0.121259
Acer          0.079048
MSI           0.041443
Toshiba       0.036838
Apple         0.016117
Samsung       0.006907
Razer         0.005372
Mediacom      0.005372
Microsoft     0.004605
Xiaomi        0.003070
Vero          0.003070
Chuwi         0.002302
Google        0.002302
Fujitsu       0.002302
LG            0.002302
Huawei        0.001535
Name: Company, dtype: float64
```

We have Lenovo, Dell and Hp products more in the data

In [22]:

```
plt.figure(figsize=(10,5))
sns.barplot(data=df,x='Company',y='Price_euros',ci=None);
plt.xticks(rotation='vertical');
```



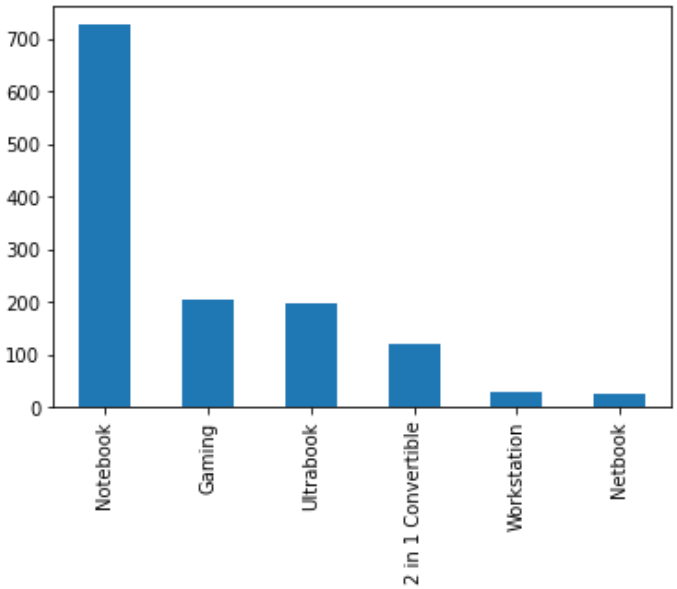
Razer and LG company laptops sold for higher price

Vero, Chuwi and Mediacom company laptops sold for lower price

Aanlysis on TypeName variable

In [23]:

```
df['TypeName'].value_counts().plot(kind='bar');
```



In [24]:

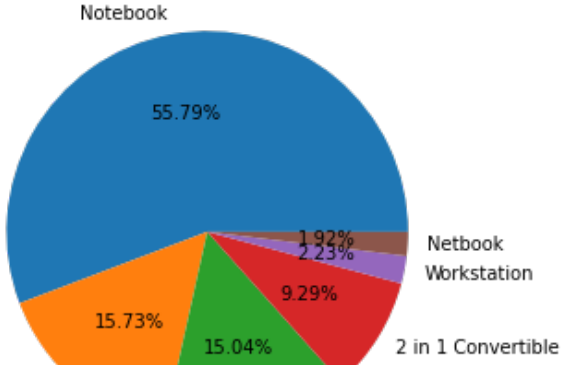
```
q=df.TypeName.value_counts(normalize=True)
q
```

Out[24]:

```
Notebook              0.557943
Gaming                0.157329
Ultrabook             0.150422
2 in 1 Convertible    0.092863
Workstation           0.022256
Netbook               0.019186
Name: TypeName, dtype: float64
```

In [25]:

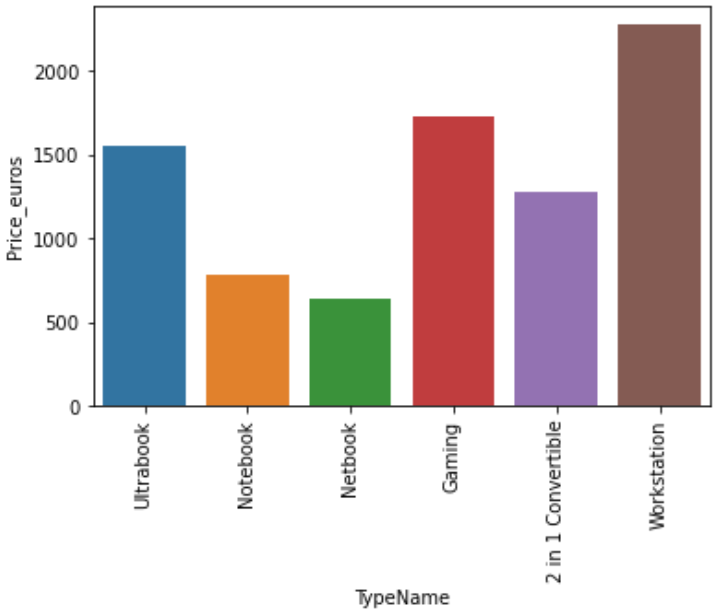
```
plt.figure(figsize=(5,5))
plt.pie(q,labels=['Notebook','Gaming','Ultrabook','2 in 1 Convertible','Workstation','Net
book'],autopct='%1.2f%%')
plt.show()
```



Notebook laptops have been sold more and Netbook are less sold

In [26]:

```
sns.barplot(data=df,x='TypeName',y='Price_euros',ci=None);
plt.xticks(rotation='vertical');
```



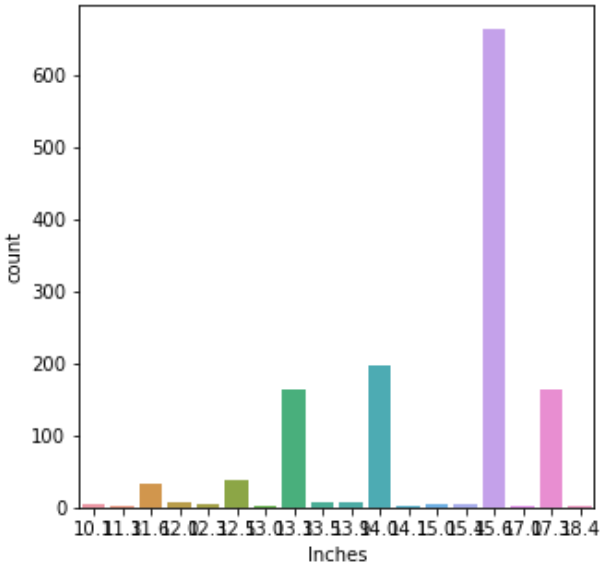
Workstation, Gaming and Ultrabook type laptops are sold for more price

Netbook type laptops are sold for less price

Analysis on Inches variable

In [27]:

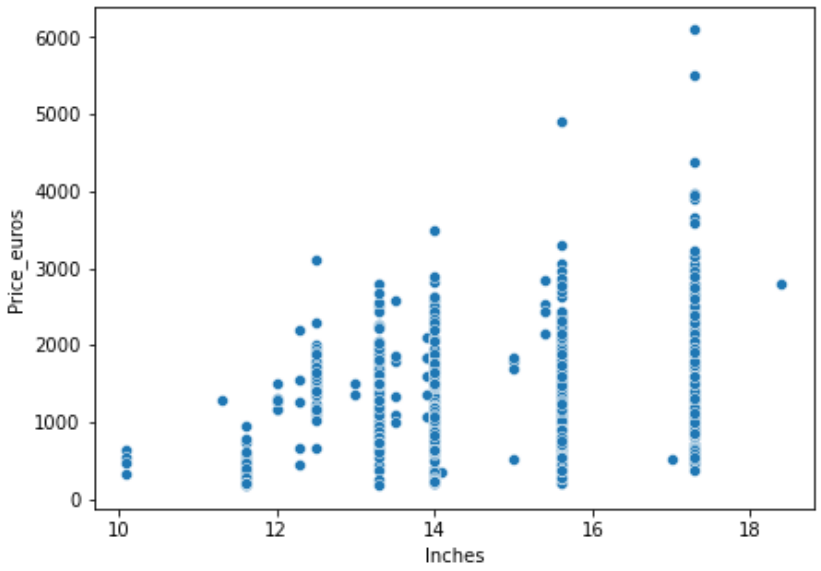
```
plt.figure(figsize=(5,5))
sns.countplot(data=df,x='Inches');
```



15.6 inch laptops are more in the data

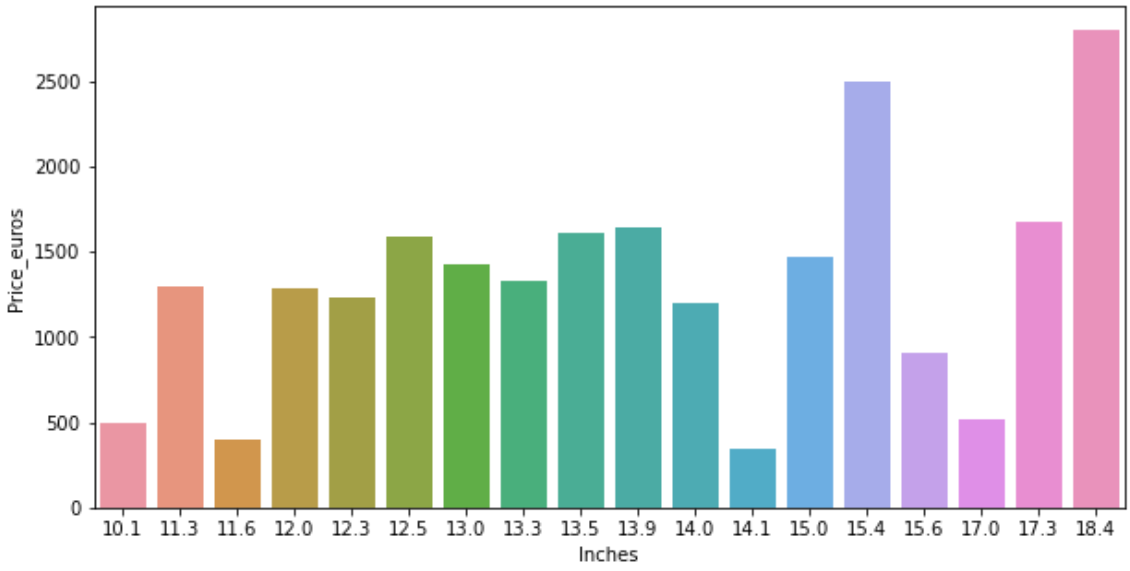
In [28]:

```
plt.figure(figsize=(7,5))
sns.scatterplot(data=df,x='Inches',y='Price_euros');
```



In [29]:

```
plt.figure(figsize=(10,5))
sns.barplot(data=df,x='Inches',y='Price_euros',ci=None);
```

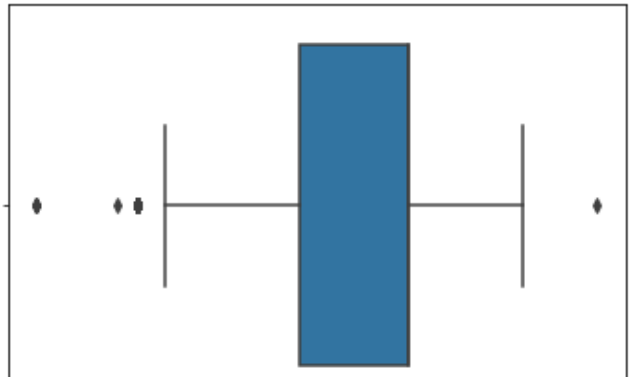


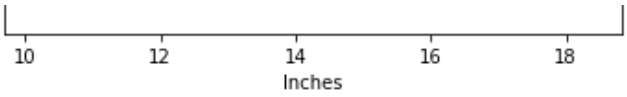
18.4 inch and 15.4 inch laptops are sold for higher price

11.6,14.1 and 10.1 inch laptops are sold for least price

In [30]:

```
sns.boxplot(data=df,x='Inches');
```





Analysis on ScreenResolution variable

In [31]:

```
df.ScreenResolution.value_counts()
```

Out[31]:

```
Full HD 1920x1080                                   507
1366x768                                            281
IPS Panel Full HD 1920x1080                         230
IPS Panel Full HD / Touchscreen 1920x1080            53
Full HD / Touchscreen 1920x1080                      47
1600x900                                             23
Touchscreen 1366x768                                 16
Quad HD+ / Touchscreen 3200x1800                     15
IPS Panel 4K Ultra HD 3840x2160                      12
IPS Panel 4K Ultra HD / Touchscreen 3840x2160        11
4K Ultra HD / Touchscreen 3840x2160                  10
4K Ultra HD 3840x2160                                 7
Touchscreen 2560x1440                                 7
IPS Panel 1366x768                                    7
IPS Panel Quad HD+ / Touchscreen 3200x1800            6
IPS Panel Retina Display 2560x1600                    6
IPS Panel Retina Display 2304x1440                    6
Touchscreen 2256x1504                                 6
IPS Panel Touchscreen 2560x1440                       5
IPS Panel Retina Display 2880x1800                    4
IPS Panel Touchscreen 1920x1200                       4
1440x900                                              4
IPS Panel 2560x1440                                   4
IPS Panel Quad HD+ 2560x1440                          3
Quad HD+ 3200x1800                                    3
1920x1080                                             3
Touchscreen 2400x1600                                 3
2560x1440                                             3
IPS Panel Touchscreen 1366x768                        3
IPS Panel Touchscreen / 4K Ultra HD 3840x2160         2
IPS Panel Full HD 2160x1440                           2
IPS Panel Quad HD+ 3200x1800                          2
IPS Panel Retina Display 2736x1824                    1
IPS Panel Full HD 1920x1200                           1
IPS Panel Full HD 2560x1440                           1
IPS Panel Full HD 1366x768                            1
Touchscreen / Full HD 1920x1080                       1
Touchscreen / Quad HD+ 3200x1800                      1
Touchscreen / 4K Ultra HD 3840x2160                   1
IPS Panel Touchscreen 2400x1600                       1
Name: ScreenResolution, dtype: int64
```

In [32]:

```
df['ScreenResolution'].apply(lambda x:1 if 'Touchscreen' in x else 0)
```

Out[32]:

```
0       0
1       0
2       0
3       0
4       0
       ..
1298    1
1299    1
1300    0
1301    0
1302    0
```

```
Name: ScreenResolution, Length: 1303, dtype: int64
```

Creating a new variable for laptops which are Touchscreen

In [33]:

```
df['Touchscreen']=df['ScreenResolution'].apply(lambda x:1 if 'Touchscreen' in x else 0)
```

In [34]:

```
df.tail()
```

Out[34]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros	To
1298	Lenovo	2 in 1 Convertible	14.0	IPS Panel Full HD / Touchscreen 1920x1080	Intel Core i7 6500U 2.5GHz	4	128GB SSD	Intel HD Graphics 520	Windows 10	1.80	638.0	
1299	Lenovo	2 in 1 Convertible	13.3	IPS Panel Quad HD+ / Touchscreen 3200x1800	Intel Core i7 6500U 2.5GHz	16	512GB SSD	Intel HD Graphics 520	Windows 10	1.30	1499.0	
1300	Lenovo	Notebook	14.0	1366x768	Intel Celeron Dual Core N3050 1.6GHz	2	64GB Flash Storage	Intel HD Graphics	Windows 10	1.50	229.0	
1301	HP	Notebook	15.6	1366x768	Intel Core i7 6500U 2.5GHz	6	1TB HDD	AMD Radeon R5 M330	Windows 10	2.19	764.0	
1302	Asus	Notebook	15.6	1366x768	Intel Celeron Dual Core N3050 1.6GHz	4	500GB HDD	Intel HD Graphics	Windows 10	2.20	369.0	

In [35]:

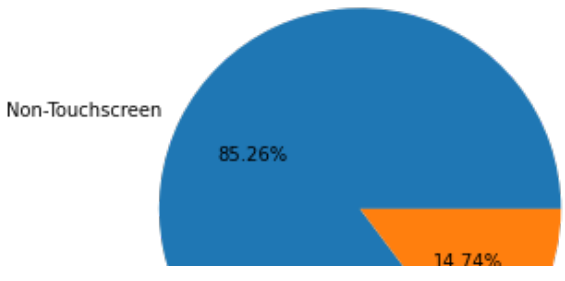
```
r=df.Touchscreen.value_counts(normalize=True)
r
```

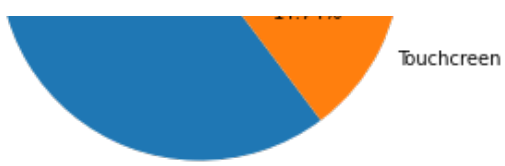
Out[35]:

```
0    0.852648
1    0.147352
Name: Touchscreen, dtype: float64
```

In [36]:

```
plt.figure(figsize=(7,5))
plt.pie(r,labels=['Non-Touchscreen','Touchcreen'],autopct='%1.2f%%')
plt.show()
```

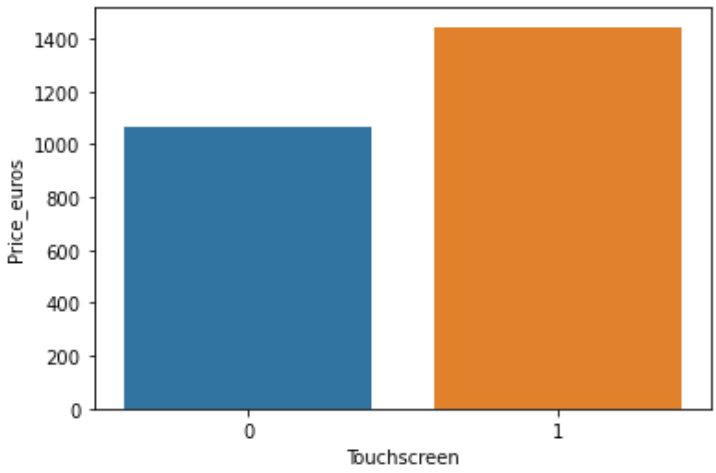




Only 14% of the laptops in the data are Touchscreen

In [37]:

```
sns.barplot(data=df,x='Touchscreen',y='Price_euros',ci=None);
```



Touch screen laptops are slightly more cost than other laptops

Creating a new variable for laptops which have IPS display

In [38]:

```
df['IPS']=df['ScreenResolution'].apply(lambda x:1 if 'IPS' in x else 0)
```

In [39]:

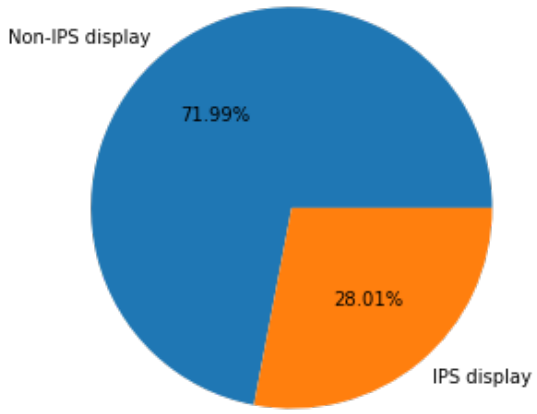
```
df.head()
```

Out[39]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscr
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	898.94	
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	575.00	
3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	2537.45	
4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	1803.60	

In [40]:

```
s=df.IPS.value_counts(normalize=True)
plt.figure(figsize=(7,5))
plt.pie(s,labels=['Non-IPS display','IPS display'],autopct='%1.2f%%')
plt.show()
```



Only 28% of the laptops in the data have IPS display

In [41]:

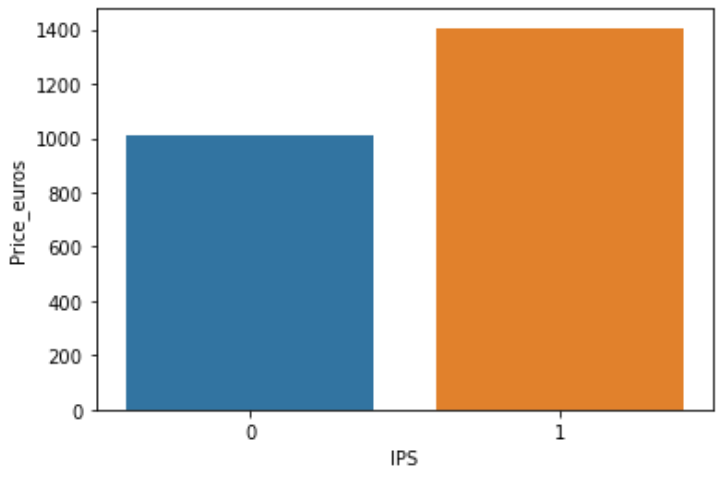
```
s
```

Out[41]:

```
0    0.719877
1    0.280123
Name: IPS, dtype: float64
```

In [42]:

```
sns.barplot(data=df,x='IPS',y='Price_euros',ci=None);
```



IPS laptops are slightly more cost than other laptops

In [43]:

```
df['ScreenResolution'].str.split('x',expand=True)
```

Out[43]:

	0	1
0	IPS Panel Retina Display 2560	1600
1	1440	900

	0	1
2	Full HD 1920	1080
3	IPS Panel Retina Display 2880	1800
4	IPS Panel Retina Display 2560	1600
...	...	...
1298	IPS Panel Full HD / Touchscreen 1920	1080
1299	IPS Panel Quad HD+ / Touchscreen 3200	1800
1300	1366	768
1301	1366	768
1302	1366	768

1303 rows × 2 columns

In [44]:

```
Res=df['ScreenResolution'].str.split('x',expand=True)
```

Creating two new variables x_res and y_res

In [45]:

```
df['x_res']=Res[0]
df['y_res']=Res[1]
```

In [46]:

```
df.head()
```

Out[46]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscr
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	898.94	
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	575.00	
3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	2537.45	
4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	1803.60	

In [47]:

```
df['x_res'].str.replace(',','').str.findall(r'(\d+\.?\d+)').apply(lambda x:x[0])
```

Out[47]:

```
0       2560
1       1440
```

```
2        1920
3        2880
4        2560
         ...
1298     1920
1299     3200
1300     1366
1301     1366
1302     1366
Name: x_res, Length: 1303, dtype: object
```

In [48]:

```
df['x_res']=df['x_res'].str.replace(',','').str.findall(r'(\d+\.?\d+)').apply(lambda x:x[0])
```

In [49]:

```
df.head()
```

Out[49]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscr
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	898.94	
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	575.00	
3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	2537.45	
4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	1803.60	

In [50]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Company           1303 non-null   object 
 1   TypeName          1303 non-null   object 
 2   Inches            1303 non-null   float64
 3   ScreenResolution  1303 non-null   object 
 4   Cpu               1303 non-null   object 
 5   Ram               1303 non-null   int64  
 6   Memory            1303 non-null   object 
 7   Gpu               1303 non-null   object 
 8   OpSys             1303 non-null   object 
 9   Weight            1303 non-null   float64
 10  Price_euros       1303 non-null   float64
 11  Touchscreen       1303 non-null   int64  
 12  IPS               1303 non-null   int64  
 13  x_res             1303 non-null   object 
 14  y_res             1303 non-null   object 
dtypes: float64(3), int64(3), object(9)
memory usage: 152.8+ KB
```

memory usage: 152.8+ KB

In [51]:

```
df['x_res']=df['x_res'].astype('int64')
df['y_res']=df['y_res'].astype('int64')
```

In [52]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Company           1303 non-null   object 
 1   TypeName          1303 non-null   object 
 2   Inches            1303 non-null   float64
 3   ScreenResolution  1303 non-null   object 
 4   Cpu               1303 non-null   object 
 5   Ram               1303 non-null   int64  
 6   Memory            1303 non-null   object 
 7   Gpu               1303 non-null   object 
 8   OpSys             1303 non-null   object 
 9   Weight            1303 non-null   float64
 10  Price_euros       1303 non-null   float64
 11  Touchscreen       1303 non-null   int64  
 12  IPS               1303 non-null   int64  
 13  x_res             1303 non-null   int64  
 14  y_res             1303 non-null   int64  
dtypes: float64(3), int64(5), object(7)
memory usage: 152.8+ KB
```

In [53]:

```
df.corr()['Price_euros']
```

Out[53]:

```
Inches          0.068197
Ram             0.743007
Weight          0.210370
Price_euros     1.000000
Touchscreen     0.191226
IPS             0.252208
x_res           0.556529
y_res           0.552809
Name: Price_euros, dtype: float64
```

Inches have very less correlation with price hence we will make a new column Pixels per inches(PPI)

we can eliminate resolution column by replacing it with PPI.

Creating a new variable PPI from x_res, y_res and Inches variables

In [54]:

```
df['PPI']=((df['x_res']**2) + (df['y_res']**2))**0.5/df['Inches']
```

In [55]:

```
df['PPI']=df['PPI'].astype('float64')
```

In [56]:

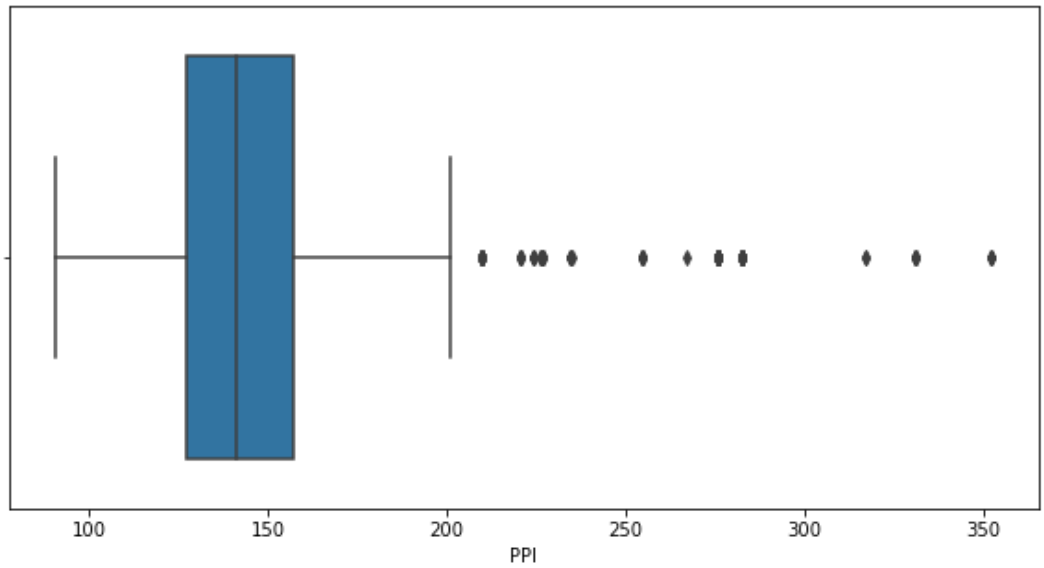
```
df['PPI']
```

Out[56]:

```
0       226.983005
1       127.677940
2       141.211998
3       220.534624
4       226.983005
           ...
1298    157.350512
1299    276.053530
1300    111.935204
1301    100.454670
1302    100.454670
Name: PPI, Length: 1303, dtype: float64
```

In [57]:

```
plt.figure(figsize=(10,5))
sns.boxplot(data=df,x='PPI');
```



In [58]:

```
df.PPI.describe([0.88,0.85,0.90,0.95,0.99])
```

Out[58]:

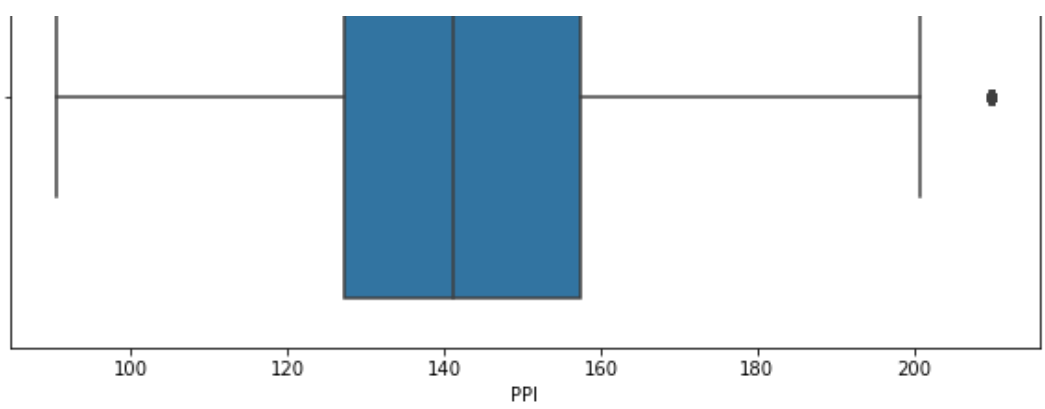
```
count    1303.000000
mean      146.635987
std        43.121345
min        90.583402
50%       141.211998
85%       165.632118
88%       165.632118
90%       176.232574
95%       254.671349
99%       282.423996
max       352.465147
Name: PPI, dtype: float64
```

In [59]:

```
df['PPI']=np.where(df['PPI']>210,210,df['PPI'])
```

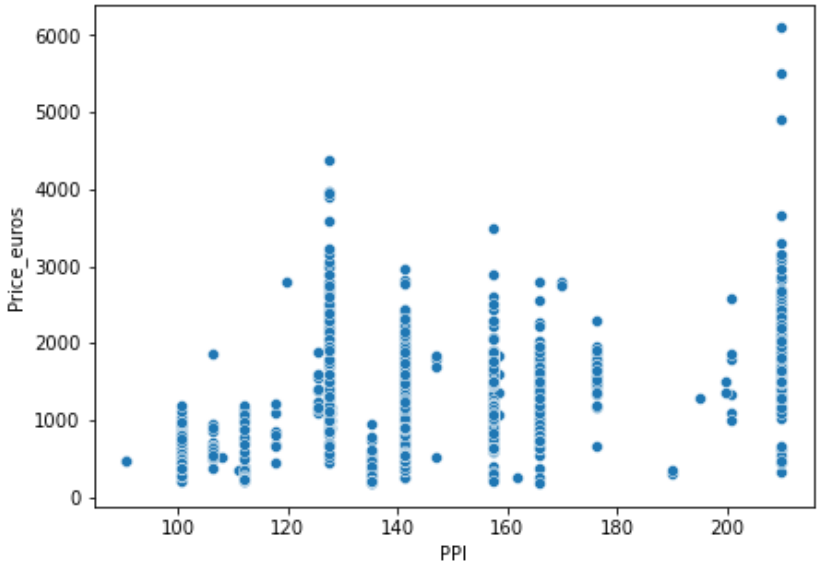
In [60]:

```
plt.figure(figsize=(10,5))
sns.boxplot(data=df,x='PPI');
```



In [61]:

```
plt.figure(figsize=(7,5))
sns.scatterplot(df['PPI'],df['Price_euros']);
```



In [62]:

```
df.corr()['Price_euros']
```

Out[62]:

```
Inches          0.068197
Ram             0.743007
Weight          0.210370
Price_euros     1.000000
Touchscreen     0.191226
IPS             0.252208
x_res           0.556529
y_res           0.552809
PPI             0.495019
Name: Price_euros, dtype: float64
```

In [63]:

```
df.drop('ScreenResolution',axis=1,inplace=True)
```

In [64]:

```
df.head()
```

Out[64]:

	Company	TypeName	Inches	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	x_res	y_
0	Apple	Ultrabook	13.3	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	0	1	2560	1

	Company	TypeName	Inches	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	x_res	y_
1	Apple	Ultrabook	13.3	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	898.94	0	0	1440	
2	HP	Notebook	15.6	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	575.00	0	0	1920	1
3	Apple	Ultrabook	15.4	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	2537.45	0	1	2880	1
4	Apple	Ultrabook	13.3	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	1803.60	0	1	2560	1

In [65]:

```
df.drop(['Inches','x_res','y_res'],inplace=True,axis=1)
```

In [66]:

```
df.corr()['Price_euros']
```

Out[66]:

```
Ram            0.743007
Weight         0.210370
Price_euros    1.000000
Touchscreen    0.191226
IPS            0.252208
PPI            0.495019
Name: Price_euros, dtype: float64
```

In [67]:

```
df.head(3)
```

Out[67]:

	Company	TypeName	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI
0	Apple	Ultrabook	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	0	1	210.000000
1	Apple	Ultrabook	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	898.94	0	0	127.677940
2	HP	Notebook	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	575.00	0	0	141.211998

Analysis on Cpu variable

In [68]:

```
df['Cpu'].value_counts()
```

Out[68]:

```
Intel Core i5 7200U 2.5GHz       190
Intel Core i7 7700HQ 2.8GHz      146
Intel Core i7 7500U 2.7GHz       134
Intel Core i7 8550U 1.8GHz        73
Intel Core i5 8250U 1.6GHz        72
                                ...
```

```
Intel Core M M3-6Y30 0.9GHz        1
AMD A9-Series 9420 2.9GHz          1
Intel Core i3 6006U 2.2GHz         1
AMD A6-Series 7310 2GHz            1
Intel Xeon E3-1535M v6 3.1GHz      1
Name: Cpu, Length: 118, dtype: int64
```

In [69]:

```
df['Cpu'].apply(lambda x:' '.join(x.split()[0:3]))
```

Out[69]:

```
0             Intel Core i5
1             Intel Core i5
2             Intel Core i5
3             Intel Core i7
4             Intel Core i5
                ...
1298          Intel Core i7
1299          Intel Core i7
1300    Intel Celeron Dual
1301          Intel Core i7
1302    Intel Celeron Dual
Name: Cpu, Length: 1303, dtype: object
```

In [70]:

```
df['Cpu_brand']=df['Cpu'].apply(lambda x:' '.join(x.split()[0:3]))
```

In [71]:

```
df.head()
```

Out[71]:

	Company	TypeName	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_
0	Apple	Ultrabook	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	0	1	210.000000	Inte
1	Apple	Ultrabook	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	898.94	0	0	127.677940	Inte
2	HP	Notebook	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	575.00	0	0	141.211998	Inte
3	Apple	Ultrabook	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	2537.45	0	1	210.000000	Inte
4	Apple	Ultrabook	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	1803.60	0	1	210.000000	Inte

In [72]:

```
def processor(x):
    if x=='Intel Core i7' or x=='Intel Core i5' or x=='Intel Core i3':
        return x
    else:
        if x.split()[0]=='Intel':
            return 'Other Intel Processor'
        else:
            return 'AMD Processor'
```

In [73]:

```
df['Cpu_brand']=df['Cpu_brand'].apply(processor)
```

In [74]:

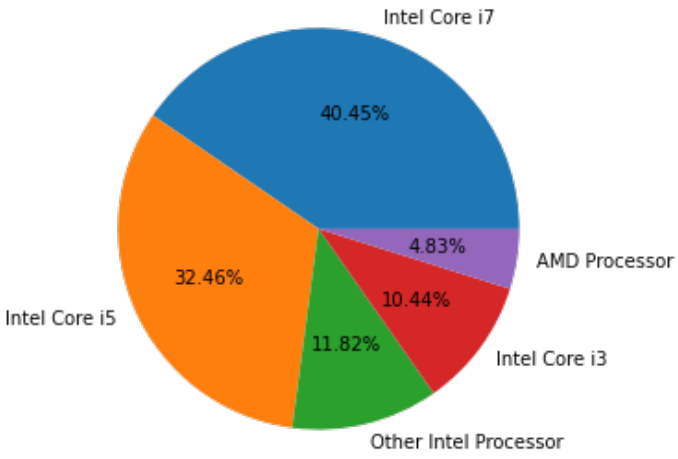
```
df.tail()
```

Out[74]:

	Company	TypeName	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI
1298	Lenovo	2 in 1 Convertible	Intel Core i7 6500U 2.5GHz	4	128GB SSD	Intel HD Graphics 520	Windows 10	1.80	638.0	1	1	157.350512
1299	Lenovo	2 in 1 Convertible	Intel Core i7 6500U 2.5GHz	16	512GB SSD	Intel HD Graphics 520	Windows 10	1.30	1499.0	1	1	210.000000
1300	Lenovo	Notebook	Intel Celeron Dual Core N3050 1.6GHz	2	64GB Flash Storage	Intel HD Graphics	Windows 10	1.50	229.0	0	0	111.935204
1301	HP	Notebook	Intel Core i7 6500U 2.5GHz	6	1TB HDD	AMD Radeon R5 M330	Windows 10	2.19	764.0	0	0	100.454670
1302	Asus	Notebook	Intel Celeron Dual Core N3050 1.6GHz	4	500GB HDD	Intel HD Graphics	Windows 10	2.20	369.0	0	0	100.454670

In [75]:

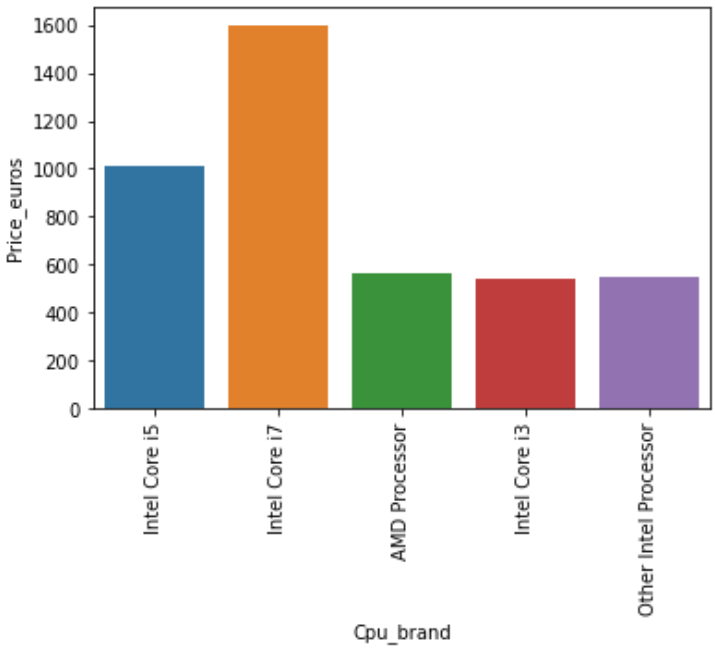
```
j=df.Cpu_brand.value_counts(normalize=True)
plt.figure(figsize=(15,5))
plt.pie(j,labels=['Intel Core i7','Intel Core i5','Other Intel Processor','Intel Core i3'
,'AMD Processor'],autopct='%1.2f%%');
```



Intel i7 and i5 laptops are more in the given data

In [76]:

```
sns.barplot(x=df['Cpu_brand'],y=df['Price_euros'],ci=None);
plt.xticks(rotation='vertical');
```



Intel core i7 and i5 laptops are sold for high price

All the other processors are sold for a similar lower price

In [77]:

```
df.head()
```

Out[77]:

	Company	TypeName	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_
0	Apple	Ultrabook	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	0	1	210.000000	Inte
1	Apple	Ultrabook	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	898.94	0	0	127.677940	Inte
2	HP	Notebook	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	575.00	0	0	141.211998	Inte
3	Apple	Ultrabook	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	2537.45	0	1	210.000000	Inte
4	Apple	Ultrabook	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	1803.60	0	1	210.000000	Inte

In [78]:

```
df.drop(['Cpu'],axis=1,inplace=True)
```

In [79]:

```
df.tail()
```

Out[79]:

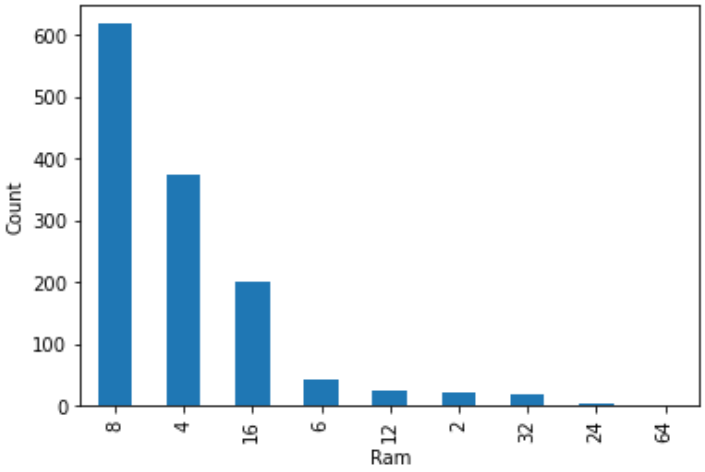
	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_bra
1298	Lenovo	2 in 1	4	128GB	Intel HD Graphics	Windows	1.80	638.0	1	1	157.350512	Intel C

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_br
1299	Lenovo	2 in 1 Convertible	16	512GB SSD	Intel HD Graphics 520	Windows 10	1.30	1499.0	1	1	210.000000	Intel C
1300	Lenovo	Notebook	2	64GB Flash Storage	Intel HD Graphics	Windows 10	1.50	229.0	0	0	111.935204	Other I Proces
1301	HP	Notebook	6	1TB HDD	AMD Radeon R5 M330	Windows 10	2.19	764.0	0	0	100.454670	Intel C
1302	Asus	Notebook	4	500GB HDD	Intel HD Graphics	Windows 10	2.20	369.0	0	0	100.454670	Other I Proces

Analysis on Ram variable

In [80]:

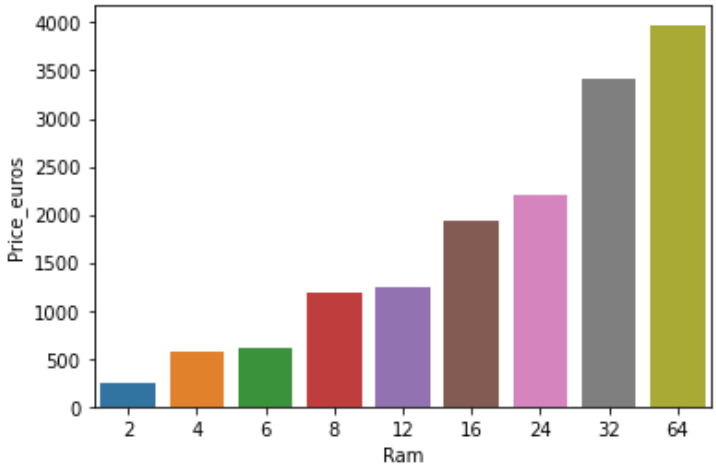
```
plt.xlabel('Ram')
plt.ylabel('Count')
df['Ram'].value_counts().plot(kind='bar');
```



8 and 4GB ram laptops are more in the data

In [81]:

```
sns.barplot(data=df,x='Ram',y='Price_euros',ci=None);
```



Laptops with 64 and 32GB Ram are sold for higher price

Laptops with 2,4 and 4GB Ram are sold for lower price

Price of Laptop is increasing with Ram size

Analysis on Memory variable

In [82]:

```
df.Memory.value_counts()
```

Out[82]:

```
256GB SSD                        412
1TB HDD                          223
500GB HDD                        132
512GB SSD                        118
128GB SSD +  1TB HDD              94
128GB SSD                         76
256GB SSD +  1TB HDD              73
32GB Flash Storage                38
2TB HDD                           16
64GB Flash Storage                15
512GB SSD +  1TB HDD              14
1TB SSD                           14
256GB SSD +  2TB HDD              10
1.0TB Hybrid                       9
256GB Flash Storage                8
16GB Flash Storage                 7
32GB SSD                           6
180GB SSD                          5
128GB Flash Storage                4
512GB SSD +  2TB HDD               3
16GB SSD                           3
512GB Flash Storage                2
1TB SSD +  1TB HDD                 2
256GB SSD +  500GB HDD             2
128GB SSD +  2TB HDD               2
256GB SSD +  256GB SSD             2
512GB SSD +  256GB SSD             1
512GB SSD +  512GB SSD             1
64GB Flash Storage +  1TB HDD      1
1TB HDD +  1TB HDD                 1
32GB HDD                           1
64GB SSD                           1
128GB HDD                          1
240GB SSD                          1
8GB SSD                            1
508GB Hybrid                       1
1.0TB HDD                          1
512GB SSD +  1.0TB Hybrid          1
256GB SSD +  1.0TB Hybrid          1
Name: Memory, dtype: int64
```

Creating four columns which are HDD, SSD, Hybrid and Flash_storage

In [83]:

```
df.head()
```

Out[83]:

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_brand
0	Apple	Ultrabook	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	0	1	210.000000	Intel Core i5
1	Apple	Ultrabook	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	898.94	0	0	127.677940	Intel Core i5

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_brand
2	HP	Notebook	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	575.00	0	0	141.211998	Intel Core i5
3	Apple	Ultrabook	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	2537.45	0	1	210.000000	Intel Core i7
4	Apple	Ultrabook	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	1803.60	0	1	210.000000	Intel Core i5

In [84]:

```
df['Memory'].unique()
```

Out[84]:

```
array(['128GB SSD', '128GB Flash Storage', '256GB SSD', '512GB SSD',
       '500GB HDD', '256GB Flash Storage', '1TB HDD',
       '32GB Flash Storage', '128GB SSD +  1TB HDD',
       '256GB SSD +  256GB SSD', '64GB Flash Storage',
       '256GB SSD +  1TB HDD', '256GB SSD +  2TB HDD', '32GB SSD',
       '2TB HDD', '64GB SSD', '1.0TB Hybrid', '512GB SSD +  1TB HDD',
       '1TB SSD', '256GB SSD +  500GB HDD', '128GB SSD +  2TB HDD',
       '512GB SSD +  512GB SSD', '16GB SSD', '16GB Flash Storage',
       '512GB SSD +  256GB SSD', '512GB SSD +  2TB HDD',
       '64GB Flash Storage +  1TB HDD', '180GB SSD', '1TB HDD +  1TB HDD',
       '32GB HDD', '1TB SSD +  1TB HDD', '512GB Flash Storage',
       '128GB HDD', '240GB SSD', '8GB SSD', '508GB Hybrid', '1.0TB HDD',
       '512GB SSD +  1.0TB Hybrid', '256GB SSD +  1.0TB Hybrid'],
      dtype=object)
```

In [85]:

```
df['Memory1']=df['Memory'].str.split('+')
df['Memory1'].sample(5)
```

Out[85]:

```
306               [256GB SSD]
1120    [32GB Flash Storage]
526                 [1TB HDD]
183               [128GB SSD]
1295                [1TB HDD]
Name: Memory1, dtype: object
```

In [86]:

```
def SSD(text):
    res=0
    for i in text:
        if 'SSD' in i:
            res=i.strip()
            if 'TB' in res:
                res=res[:-6]
                return int((res))*1000
            else:
                res=res[:-6]
                return int(float(res))
    return res
df['SSD']=df['Memory1'].apply(SSD)
df['SSD'].value_counts()
```

Out[86]:

```
256     500
0       460
128     172
512     138
1000     16
32        6
180       5
```

```
16          3
64          1
240         1
8           1
Name: SSD, dtype: int64
```

In [87]:

```
df.sample(5)
```

Out[87]:

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_bran
80	Dell	Ultrabook	8	256GB SSD	Intel UHD Graphics 620	Windows 10	1.88	1096.16	0	1	141.211998	Intel Cor i
993	Lenovo	Notebook	8	256GB SSD	Intel HD Graphics 520	Windows 10	2.30	1349.00	0	0	141.211998	Intel Cor i
349	Dell	Ultrabook	8	1TB HDD	AMD Radeon 530	Windows 10	1.90	663.00	0	0	141.211998	Intel Cor i
861	HP	Notebook	8	256GB SSD	AMD Radeon R7 M465	Windows 10	1.84	1349.00	0	0	141.211998	Intel Cor i
720	Lenovo	Ultrabook	8	512GB SSD	Intel HD Graphics 520	Windows 10	1.17	1686.64	0	1	209.800683	Intel Cor i

In [88]:

```
def HDD(text):
    l=len(text)
    res=0
    for i in text:
        if 'HDD' in i:
            res=i.strip()
            if 'TB' in res:
                res=res[:-6]
                return int(float(res))*1000
            else:
                res=res[:-6]
                return int(float(res))
    return res
df['HDD']=df['Memory1'].apply(HDD)
df['HDD'].value_counts()
```

Out[88]:

```
0       727
1000    409
500     134
2000     31
32        1
128       1
Name: HDD, dtype: int64
```

In [89]:

```
df.sample(10)
```

Out[89]:

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_bra
321	Lenovo	Notebook	4	1TB HDD	Nvidia GeForce	Windows 10	2.79	589.00	0	0	106.113062	Intel C

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_br
411	Lenovo	Ultrabook	8	512GB SSD	Nvidia GeForce GT 940MX	Windows 10	1.70	1943.00	0	1	209.800683	Intel C
728	Dell	Notebook	8	1TB HDD	AMD Radeon R7 M445	Linux	2.32	589.52	0	0	100.454670	Intel C
512	Dell	Notebook	8	1TB HDD	Intel HD Graphics 620	Windows 10	2.18	836.00	0	0	141.211998	Intel C
951	Dell	Ultrabook	8	256GB SSD	Intel HD Graphics 620	Windows 10	1.36	1775.00	0	0	157.350512	Intel C
622	Dell	Notebook	4	1TB HDD	AMD Radeon 530	Windows 10	2.20	776.00	0	0	141.211998	Intel C
461	Acer	Netbook	4	128GB SSD	Intel HD Graphics 400	Windows 10	1.40	435.00	0	0	135.094211	Other I Proces
1167	Lenovo	Notebook	4	1TB HDD	Nvidia GeForce 920MX	Windows 10	2.20	468.00	0	0	141.211998	Intel C
175	Dell	Notebook	8	1TB HDD	Intel HD Graphics 620	Windows 10	2.30	459.00	0	0	100.454670	Intel C
1020	Dell	2 in 1 Convertible	4	1TB HDD	Intel HD Graphics 520	Windows 10	2.08	795.99	1	1	141.211998	Intel C

In [90]:

```
def Flash(text):
    l=len(text)
    res=0
    for i in text:
        if 'Flash Storage' in i:
            res=i.strip()
            if 'TB' in res:
                res=res[:-16]
                return int(float(res))*1000
            else:
                res=res[:-16]
                return int(float(res))
    return res
df['Flash Storage']=df['Memory1'].apply(Flash)
df['Flash Storage'].value_counts()
```

Out[90]:

```
0      1228
32       38
64       16
256       8
16        7
128       4
512       2
Name: Flash Storage, dtype: int64
```

In [91]:

```
df.head()
```

Out[91]:

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_brand	M

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_brand
0	Apple	Ultrabook	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	0	1	210.000000	Intel Core i5
1	Apple	Ultrabook	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	898.94	0	0	127.677940	Intel Core i5
2	HP	Notebook	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	575.00	0	0	141.211998	Intel Core i5
3	Apple	Ultrabook	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	2537.45	0	1	210.000000	Intel Core i7
4	Apple	Ultrabook	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	1803.60	0	1	210.000000	Intel Core i5

In [92]:

```
def Hybrid(text):
    l=len(text)
    res=0
    for i in text:
        if 'Hybrid' in i:
            res=i.strip()
            if 'TB' in res:
                res=res[:-9]
                return int(float(res))*1000
            else:
                res=res[:-9]
                return int(float(res))
    return res
df['Hybrid']=df['Memory1'].apply(Hybrid)
df['Hybrid'].value_counts()
```

Out[92]:

```
0       1291
1000      11
508        1
Name: Hybrid, dtype: int64
```

In [93]:

```
df.sample(10)
```

Out[93]:

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_br
1149	Lenovo	2 in 1 Convertible	8	256GB SSD	Intel HD Graphics 520	Windows 10	1.27	2339.00	1	1	209.800683	Intel C
1178	MSI	Gaming	16	128GB SSD + 1TB HDD	Nvidia GeForce GTX 970M	Windows 10	1.91	2153.37	0	0	141.211998	Intel C
480	Dell	Notebook	8	512GB SSD	Nvidia GeForce 940MX	Windows 10	2.16	1262.00	0	1	141.211998	Intel C
174	HP	Notebook	8	256GB SSD	Nvidia GeForce 930MX	Windows 10	2.50	923.00	0	0	127.335675	Intel C
797	Dell	Notebook	8	128GB	Intel HD Graphics	Windows	1.95	810.00	0	0	100.454670	Intel C

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_bra
1094	HP	Netbook	4	128GB SSD	Intel HD Graphics 520	Windows 7	2.40	1599.00	0	0	125.367428	Intel C
826	Asus	2 in 1 Convertible	8	256GB SSD	Intel HD Graphics 620	Windows 10	1.10	1358.00	1	0	165.632118	Intel C
364	Lenovo	Notebook	8	256GB SSD	Nvidia GeForce 920MX	No OS	2.20	499.00	0	0	100.454670	Intel C
465	Asus	Notebook	4	500GB HDD	Intel HD Graphics 500	Windows 10	2.00	304.00	0	0	100.454670	Other I Proces
289	Lenovo	Notebook	8	1TB HDD	Nvidia GeForce 940MX	No OS	2.20	659.01	0	0	141.211998	Intel C

In [94]:

```
df[df['Memory'] =='1.0TB Hybrid']
```

Out[94]:

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_bra
151	Dell	Gaming	8	1.0TB Hybrid	Nvidia GeForce GTX 1050	Windows 10	2.62	899.00	0	0	141.211998	Intel Co
1010	Dell	Gaming	8	1.0TB Hybrid	Nvidia GeForce GTX 1050	Windows 10	2.65	949.00	0	0	141.211998	Intel Co
1135	Lenovo	Notebook	16	1.0TB Hybrid	AMD Radeon R7 M360	Windows 10	2.50	1099.00	0	0	141.211998	Intel Co
1158	Lenovo	Notebook	8	1.0TB Hybrid	AMD Radeon R5 M330	Windows 10	2.50	788.49	0	0	141.211998	Intel Co
1176	Lenovo	Notebook	4	1.0TB Hybrid	Intel HD Graphics 520	Windows 7	2.32	825.00	0	0	141.211998	Intel Co
1258	Lenovo	Notebook	8	1.0TB Hybrid	AMD Radeon R5 M330	Windows 7	2.32	895.00	0	0	141.211998	Intel Co
1266	HP	Notebook	6	1.0TB Hybrid	AMD Radeon R7 M440	Windows 10	2.04	549.99	0	0	141.211998	AM Process
1280	HP	Notebook	6	1.0TB Hybrid	AMD Radeon R7 M440	Windows 10	2.04	549.99	0	0	141.211998	AM Process
1294	HP	Notebook	6	1.0TB Hybrid	AMD Radeon R7 M440	Windows 10	2.04	549.99	0	0	141.211998	AM Process

In [95]:

```
df.corr()['Price_euros']
```

Out[95]:

```
Ram               0.743007
Weight            0.210370
```

```
Weight            0.210370
Price_euros       1.000000
Touchscreen       0.191226
IPS               0.252208
PPI               0.495019
SSD               0.676202
HDD              -0.095672
Flash Storage    -0.040511
Hybrid            0.007989
Name: Price_euros, dtype: float64
```

There is strong correlation between the variables Ram and price ,SSD and price,PPI and price.

There is weak correlation for the variables HDD,flash storage,Hybrid.

In [96]:

```
df.drop(['Flash Storage','Hybrid','HDD'],axis=1,inplace=True)
```

In [97]:

```
df.drop(['Memory1','Memory'],axis=1,inplace=True)
```

In [98]:

```
df.head()
```

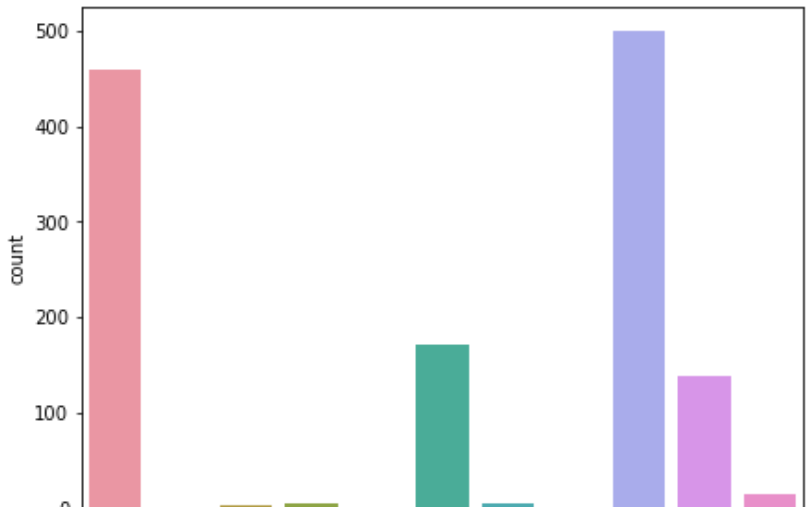
Out[98]:

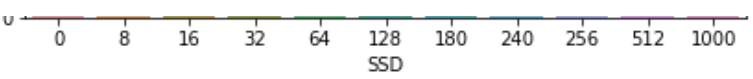
	Company	TypeName	Ram	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_brand	SSD
0	Apple	Ultrabook	8	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	0	1	210.000000	Intel Core i5	128
1	Apple	Ultrabook	8	Intel HD Graphics 6000	macOS	1.34	898.94	0	0	127.677940	Intel Core i5	0
2	HP	Notebook	8	Intel HD Graphics 620	No OS	1.86	575.00	0	0	141.211998	Intel Core i5	256
3	Apple	Ultrabook	16	AMD Radeon Pro 455	macOS	1.83	2537.45	0	1	210.000000	Intel Core i7	512
4	Apple	Ultrabook	8	Intel Iris Plus Graphics 650	macOS	1.37	1803.60	0	1	210.000000	Intel Core i5	256

Analysis on SSD variable

In [99]:

```
plt.figure(figsize=(7,5))
sns.countplot(data=df,x='SSD');
```

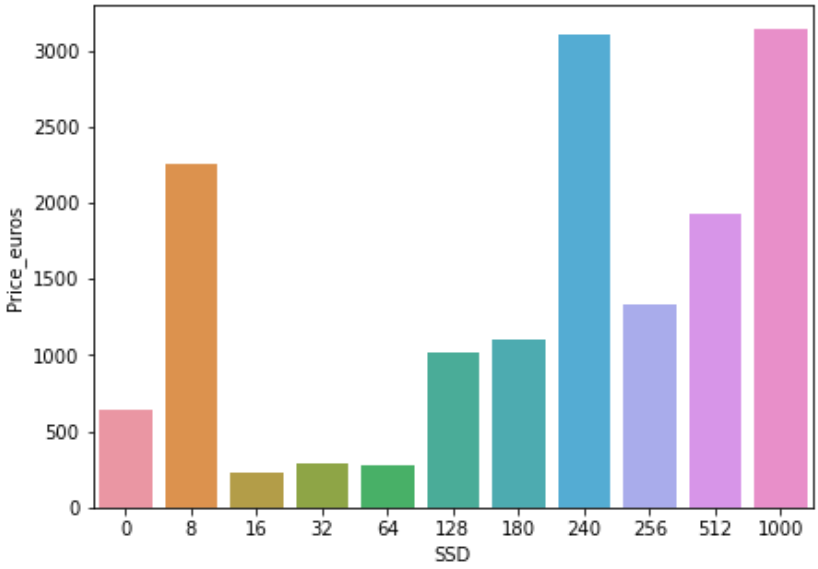




Laptops with 256GB SSD and No SSD are more in the given data

In [100]:

```
plt.figure(figsize=(7,5))
sns.barplot(data=df,x='SSD',y='Price_euros',ci=None);
```



Laptops with 1TB and 240GB SSD are high priced laptops

Analysis on Gpu variable

In [101]:

```
df.Gpu.value_counts()
```

Out[101]:

```
Intel HD Graphics 620      281
Intel HD Graphics 520      185
Intel UHD Graphics 620      68
Nvidia GeForce GTX 1050     66
Nvidia GeForce GTX 1060     48
                          ...
AMD Radeon R5 520             1
AMD Radeon R7                 1
Intel HD Graphics 540         1
AMD Radeon 540                1
ARM Mali T860 MP4             1
Name: Gpu, Length: 110, dtype: int64
```

In [102]:

```
df['Gpu'].apply(lambda x:x.split()[0] )
```

Out[102]:

```
0       Intel
1       Intel
2       Intel
3         AMD
4       Intel
        ...
1298    Intel
1299    Intel
1300    Intel
1301      AMD
```

```
1302     Intel
Name: Gpu, Length: 1303, dtype: object
```

Creating a new variable Gpu_brand

In [103]:

```
df['Gpu_brand']=df['Gpu'].apply(lambda x:x.split()[0])
```

In [104]:

```
df.head()
```

Out[104]:

	Company	TypeName	Ram	Gpu	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_brand	SSD	Gpu_
0	Apple	Ultrabook	8	Intel Iris Plus Graphics 640	macOS	1.37	1339.69	0	1	210.000000	Intel Core i5	128	
1	Apple	Ultrabook	8	Intel HD Graphics 6000	macOS	1.34	898.94	0	0	127.677940	Intel Core i5	0	
2	HP	Notebook	8	Intel HD Graphics 620	No OS	1.86	575.00	0	0	141.211998	Intel Core i5	256	
3	Apple	Ultrabook	16	AMD Radeon Pro 455	macOS	1.83	2537.45	0	1	210.000000	Intel Core i7	512	
4	Apple	Ultrabook	8	Intel Iris Plus Graphics 650	macOS	1.37	1803.60	0	1	210.000000	Intel Core i5	256	

In [105]:

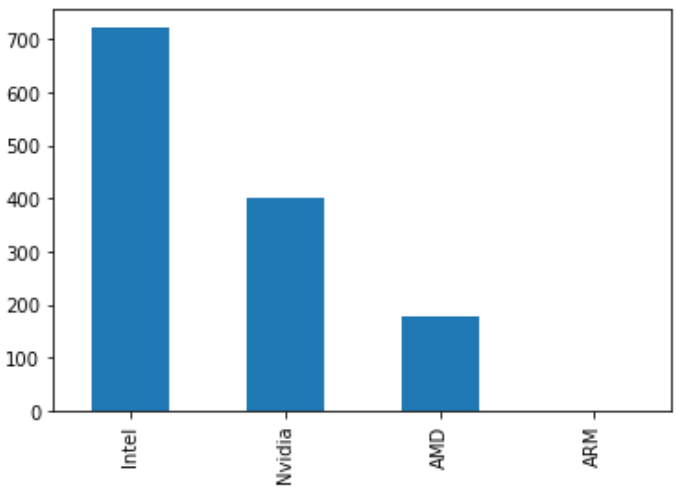
```
df.Gpu_brand.value_counts()
```

Out[105]:

```
Intel     722
Nvidia    400
AMD       180
ARM         1
Name: Gpu_brand, dtype: int64
```

In [106]:

```
df.Gpu_brand.value_counts().plot(kind='bar');
```



In [107]:

```
df=df[df['Gpu_brand']!='ARM']
```

In [108]:

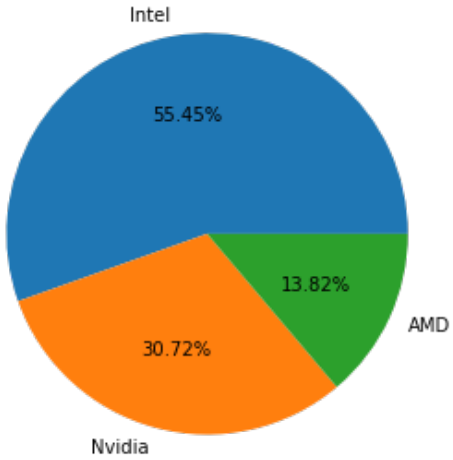
```
df.Gpu_brand.value_counts()
```

Out[108]:

```
Intel     722
Nvidia    400
AMD       180
Name: Gpu_brand, dtype: int64
```

In [109]:

```
k=df['Gpu_brand'].value_counts(normalize=True)
plt.figure(figsize=(15,5))
plt.pie(k,labels=['Intel','Nvidia','AMD'],autopct='%1.2f%%');
```

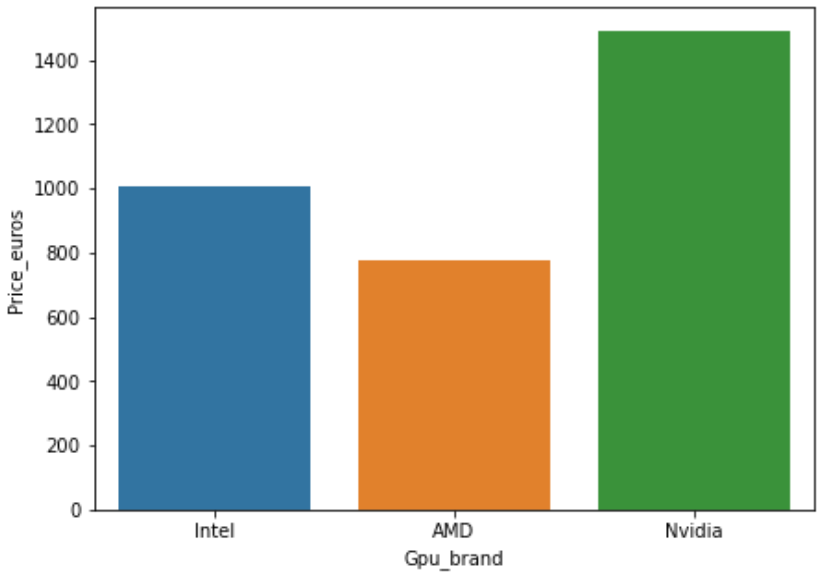


Almost 55% of the laptops have Intel graphic processor

Only 13% of the laptops have AMD graphic processor

In [110]:

```
plt.figure(figsize=(7,5))
sns.barplot(df['Gpu_brand'],df['Price_euros'],ci=None);
```



Intel is the most used Gpu among the data

AMD is the least used Gpu among the data

In [111]:

```
df.drop('Gpu',axis=1,inplace=True)
```

In [112]:

```
df.head()
```

Out[112]:

	Company	TypeName	Ram	OpSys	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_brand	SSD	Gpu_brand
0	Apple	Ultrabook	8	macOS	1.37	1339.69	0	1	210.000000	Intel Core i5	128	Intel
1	Apple	Ultrabook	8	macOS	1.34	898.94	0	0	127.677940	Intel Core i5	0	Intel
2	HP	Notebook	8	No OS	1.86	575.00	0	0	141.211998	Intel Core i5	256	Intel
3	Apple	Ultrabook	16	macOS	1.83	2537.45	0	1	210.000000	Intel Core i7	512	AMD
4	Apple	Ultrabook	8	macOS	1.37	1803.60	0	1	210.000000	Intel Core i5	256	Intel

Analysis on OpSys variable

In [113]:

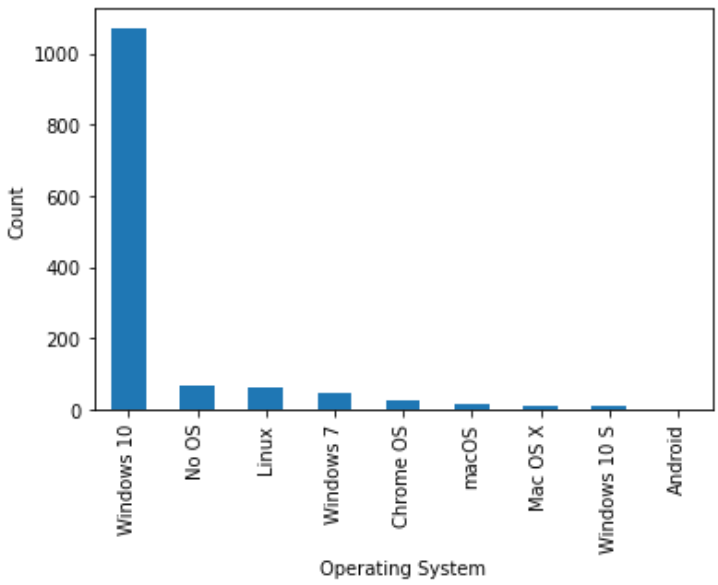
```
df.OpSys.value_counts()
```

Out[113]:

```
Windows 10      1072
No OS             66
Linux             62
Windows 7         45
Chrome OS         26
macOS             13
Mac OS X           8
Windows 10 S       8
Android            2
Name: OpSys, dtype: int64
```

In [114]:

```
plt.xlabel('Operating System')
plt.ylabel('Count')
df.OpSys.value_counts().plot(kind='bar');
```

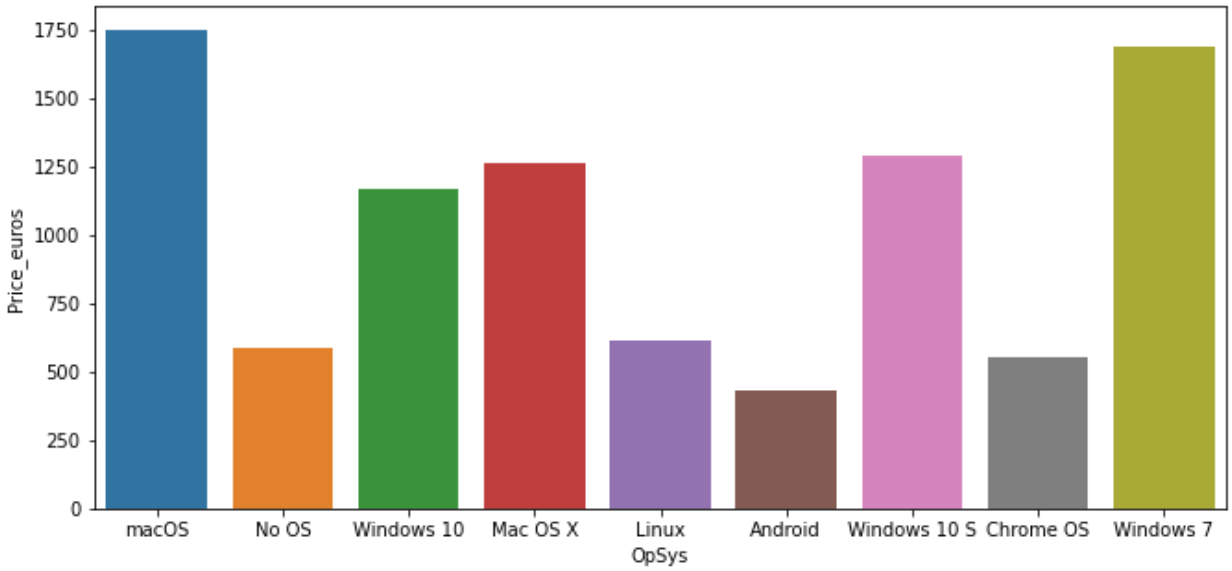


Windows 10 is most used Operating system in the data

Android ,Windows 10 S and Mac OS X are least used Operating systems in the data

In [115]:

```
plt.figure(figsize=(11,5))
sns.barplot(data=df,x='OpSys',y='Price_euros',ci=None);
```



1.The Laptops whose operating system is MacOS have high price 1750 euros followed by the laptops of operating system Windows 7.

2.The laptops having windows 10 S and mac OS X have almost similar price.

In [116]:

```
def os(x):
    if x=='Windows 10' or x=='Windows 7' or x=='Windows 10 S':
        return 'Windows'
    elif x=='macOS' or x=='Mac OS X':
        return 'Mac'
    else:
        return 'Others/No OS/Linux'
```

Creating new variable OS

In [117]:

```
df['OS']=df['OpSys'].apply(os)
```

In [118]:

```
df.OS.value_counts()
```

Out[118]:

```
Windows               1125
Others/No OS/Linux     156
Mac                     21
Name: OS, dtype: int64
```

In [119]:

```
t=df.OS.value_counts(normalize=True)
t
```

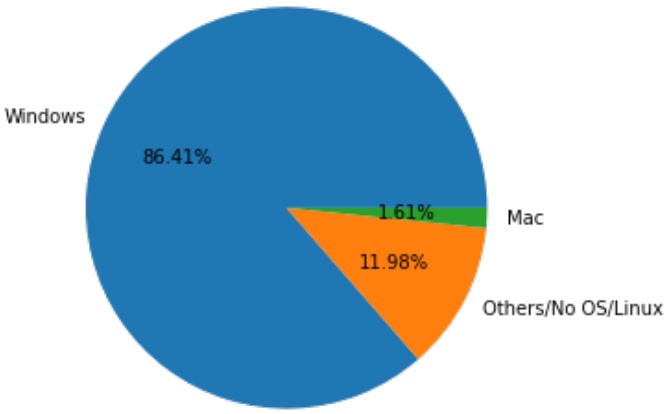
Out[119]:

```
Windows               0.864055
Others/No OS/Linux    0.119816
```

```
Others/No OS/Linux    0.119816
Mac                   0.016129
Name: OS, dtype: float64
```

In [120]:

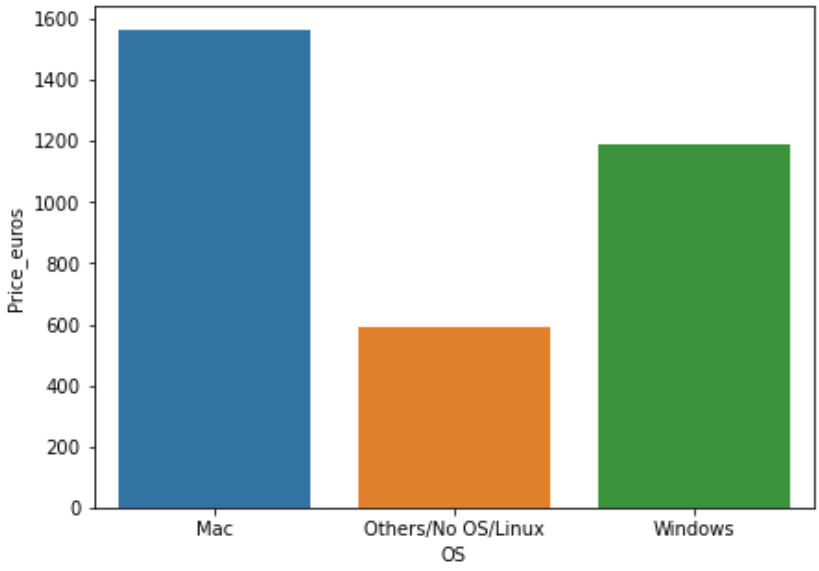
```
t=df['OS'].value_counts(normalize=True)
plt.figure(figsize=(15,5))
plt.pie(t,labels=['Windows','Others/No OS/Linux ','Mac'],autopct='%1.2f%%');
```



Almost 86% of the laptops have Windows Operating system

In [121]:

```
plt.figure(figsize=(7,5))
sns.barplot(df['OS'],df['Price_euros'],ci=None);
```



Laptops with Mac operating system are more priced laptops

In [122]:

```
df.drop('OpSys',axis=1,inplace=True)
```

In [123]:

```
df.sample(5)
```

Out[123]:

	Company	TypeName	Ram	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_brand	SSD	Gpu_brand	C
125	HP	Notebook	6	1.58	650.00	0	1	157.350512	Intel Core	256	Nvidia	Windows

	Company	TypeName	Ram	Weight	Price_euros	Touchscreen	IPS	PPI	Cpu_brand	SSD	Gpu_brand	O
105	HP	Notebook	6	1.58	659.00	0	1	157.350512	i5	256	Nvidia	Windov
1237	Dell	Notebook	4	2.24	737.00	0	0	100.454670	Intel Core i5	0	Intel	Windov
665	Toshiba	Notebook	4	2.10	498.00	0	0	100.454670	Intel Core i3	128	Intel	Windov
1281	Dell	Notebook	8	2.30	805.99	0	0	100.454670	Intel Core i7	0	AMD	Others/N OS/Linu
16	Dell	Notebook	8	2.20	745.00	0	0	141.211998	Intel Core i7	256	AMD	Windov

Unique values percentage

In [124]:

```
df.describe([0.01,0.05,0.95,0.99,0.25,0.75]).round(2)
```

Out[124]:

	Ram	Weight	Price_euros	Touchscreen	IPS	PPI	SSD
count	1302.00	1302.00	1302.00	1302.00	1302.00	1302.00	1302.00
mean	8.39	2.04	1124.04	0.15	0.28	142.06	182.89
std	5.09	0.67	699.16	0.35	0.45	30.01	184.82
min	2.00	0.69	174.00	0.00	0.00	90.58	0.00
1%	2.00	0.97	229.00	0.00	0.00	100.45	0.00
5%	4.00	1.17	309.00	0.00	0.00	100.45	0.00
25%	4.00	1.50	599.00	0.00	0.00	127.34	0.00
50%	8.00	2.04	978.00	0.00	0.00	141.21	256.00
75%	8.00	2.30	1488.44	0.00	1.00	157.35	256.00
95%	16.00	3.20	2448.80	1.00	1.00	210.00	512.00
99%	32.00	4.42	3148.98	1.00	1.00	210.00	1000.00
max	64.00	4.70	6099.00	1.00	1.00	210.00	1000.00

In [125]:

```
for i in df.columns:
    print(i,'------>',round((df[i].nunique()/df.shape[0])*100,3))
```

```
Company ------> 1.459
TypeName ------> 0.461
Ram ------> 0.691
Weight ------> 13.134
Price_euros ------> 60.753
Touchscreen ------> 0.154
IPS ------> 0.154
PPI ------> 1.997
Cpu_brand ------> 0.384
SSD ------> 0.845
Gpu_brand ------> 0.23
OS ------> 0.23
```

In [126]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1302 entries, 0 to 1302
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
```

```
 0   Company       1302 non-null   object
 1   TypeName      1302 non-null   object
 2   Ram           1302 non-null   int64
 3   Weight        1302 non-null   float64
 4   Price_euros   1302 non-null   float64
 5   Touchscreen   1302 non-null   int64
 6   IPS           1302 non-null   int64
 7   PPI           1302 non-null   float64
 8   Cpu_brand     1302 non-null   object
 9   SSD           1302 non-null   int64
 10  Gpu_brand     1302 non-null   object
 11  OS            1302 non-null   object
dtypes: float64(3), int64(4), object(5)
memory usage: 164.5+ KB
```

In [127]:

```
df.describe([0.01,0.05,0.25,0.75,0.95,0.99])
```

Out[127]:

	Ram	Weight	Price_euros	Touchscreen	IPS	PPI	SSD
count	1302.000000	1302.000000	1302.000000	1302.000000	1302.00000	1302.000000	1302.000000
mean	8.385561	2.039416	1124.043894	0.146697	0.27957	142.059061	182.890937
std	5.085166	0.665274	699.158856	0.353940	0.44896	30.014002	184.823639
min	2.000000	0.690000	174.000000	0.000000	0.00000	90.583402	0.000000
1%	2.000000	0.970000	229.000000	0.000000	0.00000	100.454670	0.000000
5%	4.000000	1.170000	309.000000	0.000000	0.00000	100.454670	0.000000
25%	4.000000	1.500000	599.000000	0.000000	0.00000	127.335675	0.000000
50%	8.000000	2.040000	978.000000	0.000000	0.00000	141.211998	256.000000
75%	8.000000	2.300000	1488.435000	0.000000	1.00000	157.350512	256.000000
95%	16.000000	3.200000	2448.800000	1.000000	1.00000	210.000000	512.000000
99%	32.000000	4.420000	3148.983700	1.000000	1.00000	210.000000	1000.000000
max	64.000000	4.700000	6099.000000	1.000000	1.00000	210.000000	1000.000000

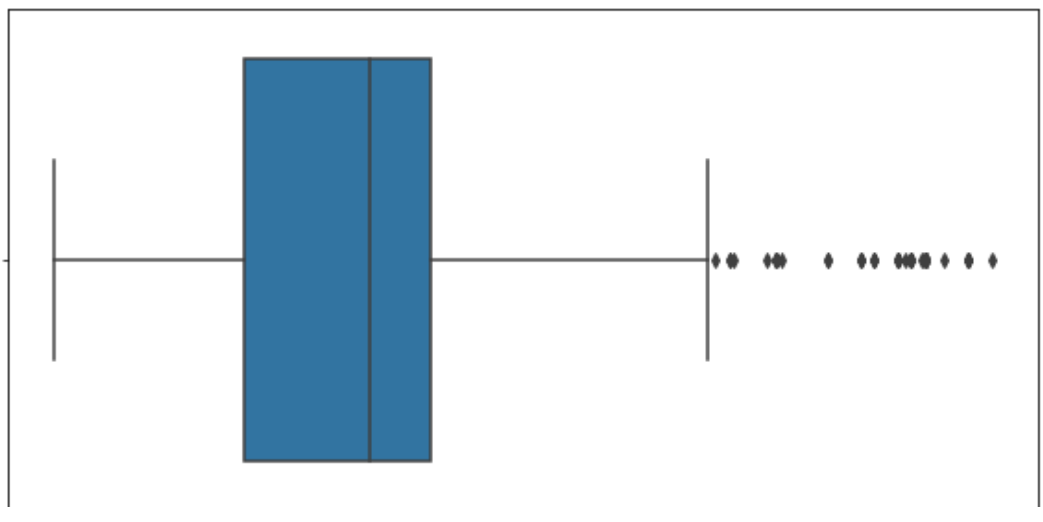
In [ ]:

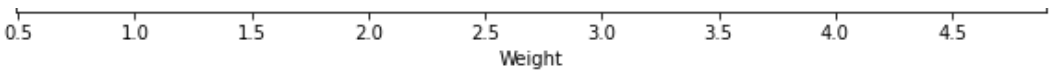
```

```

Analysis on Weight variable

In [128]:

```
plt.figure(figsize=(10,5))
sns.boxplot(data=df,x='Weight');
```





Removing outliers by using capping method

In [129]:

```
df['Weight']=np.where(df['Weight']>3.5,3.5,df['Weight'])
```

In [130]:

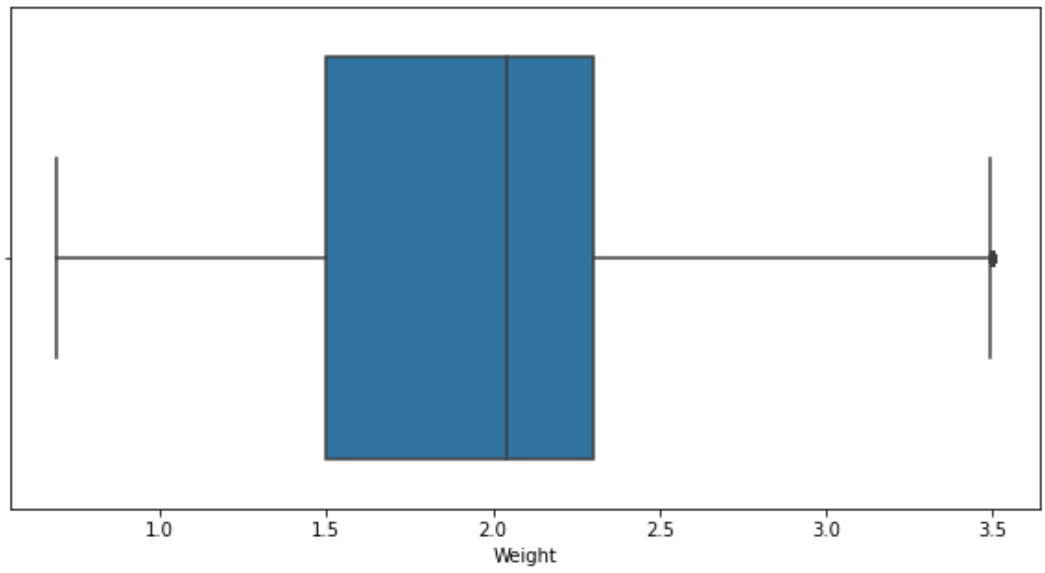
```
df.describe([0.99,0.95,0.96])
```

Out[130]:

	Ram	Weight	Price_euros	Touchscreen	IPS	PPI	SSD
count	1302.000000	1302.000000	1302.000000	1302.000000	1302.00000	1302.000000	1302.000000
mean	8.385561	2.014585	1124.043894	0.146697	0.27957	142.059061	182.890937
std	5.085166	0.590207	699.158856	0.353940	0.44896	30.014002	184.823639
min	2.000000	0.690000	174.000000	0.000000	0.00000	90.583402	0.000000
50%	8.000000	2.040000	978.000000	0.000000	0.00000	141.211998	256.000000
95%	16.000000	3.200000	2448.800000	1.000000	1.00000	210.000000	512.000000
96%	16.000000	3.350000	2536.312000	1.000000	1.00000	210.000000	512.000000
99%	32.000000	3.500000	3148.983700	1.000000	1.00000	210.000000	1000.000000
max	64.000000	3.500000	6099.000000	1.000000	1.00000	210.000000	1000.000000

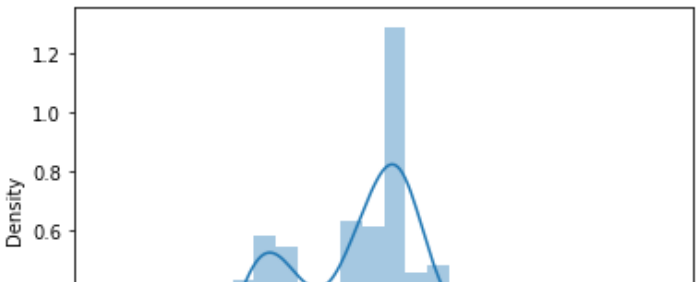
In [131]:

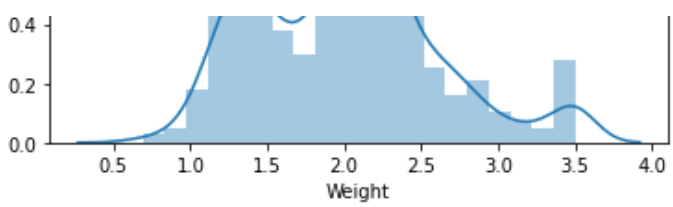
```
plt.figure(figsize=(10,5))
sns.boxplot(data=df,x='Weight');
```



In [132]:

```
sns.distplot(df.Weight);
```

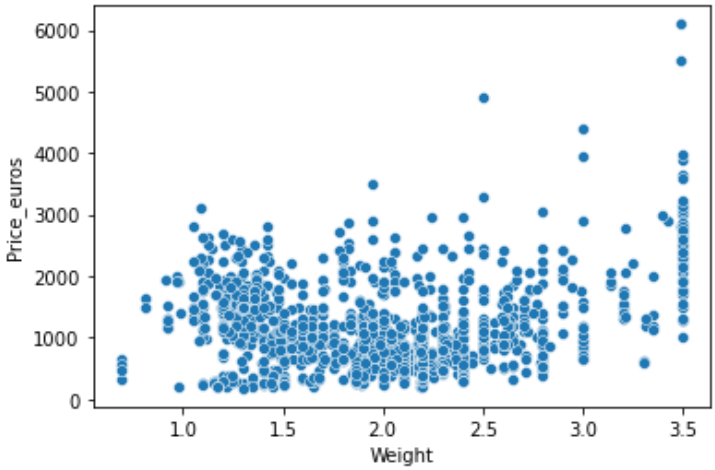




Laptops among 1, 2 and 3 kg weight are more in the data

In [133]:

```
sns.scatterplot(data=df,x='Weight',y='Price_euros');
```



Laptops among 1, 2 and 3 kg weight are of similar price ie., 0-2500 euros

In [134]:

```
df.corr()['Price_euros']
```

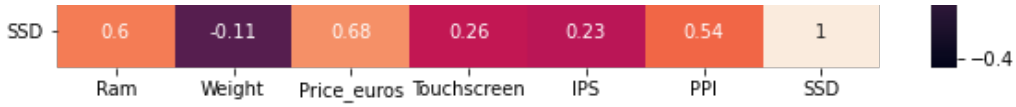
Out[134]:

```
Ram             0.742905
Weight          0.146498
Price_euros     1.000000
Touchscreen     0.192917
IPS             0.253320
PPI             0.497232
SSD             0.676066
Name: Price_euros, dtype: float64
```

Multicollinearity matrix of all the variables

In [135]:

```
plt.figure(figsize=(10,5))
sns.heatmap(df.corr(),annot=True);
```



VIF values for all variables

In [136]:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
import statsmodels.api as sm
```

In [137]:

```
X=df[['Ram','Weight','Touchscreen','IPS','PPI','SSD']]
X=sm.add_constant(X)
vif_data=pd.DataFrame({'variables':X.columns[1:],'VIF':[variance_inflation_factor(X.valu
es,i+1) for i in range(len(X.columns[1:]))]})
vif_data
```

Out[137]:

	variables	VIF
0	Ram	2.215408
1	Weight	1.835519
2	Touchscreen	1.274273
3	IPS	1.161644
4	PPI	2.137900
5	SSD	2.082202

In [138]:

```
df.describe([0.01,0.05,0.25,0.50,0.75,0.95,0.99])
```

Out[138]:

	Ram	Weight	Price_euros	Touchscreen	IPS	PPI	SSD
count	1302.000000	1302.000000	1302.000000	1302.000000	1302.00000	1302.000000	1302.000000
mean	8.385561	2.014585	1124.043894	0.146697	0.27957	142.059061	182.890937
std	5.085166	0.590207	699.158856	0.353940	0.44896	30.014002	184.823639
min	2.000000	0.690000	174.000000	0.000000	0.00000	90.583402	0.000000
1%	2.000000	0.970000	229.000000	0.000000	0.00000	100.454670	0.000000
5%	4.000000	1.170000	309.000000	0.000000	0.00000	100.454670	0.000000
25%	4.000000	1.500000	599.000000	0.000000	0.00000	127.335675	0.000000
50%	8.000000	2.040000	978.000000	0.000000	0.00000	141.211998	256.000000
75%	8.000000	2.300000	1488.435000	0.000000	1.00000	157.350512	256.000000
95%	16.000000	3.200000	2448.800000	1.000000	1.00000	210.000000	512.000000
99%	32.000000	3.500000	3148.983700	1.000000	1.00000	210.000000	1000.000000
max	64.000000	3.500000	6099.000000	1.000000	1.00000	210.000000	1000.000000

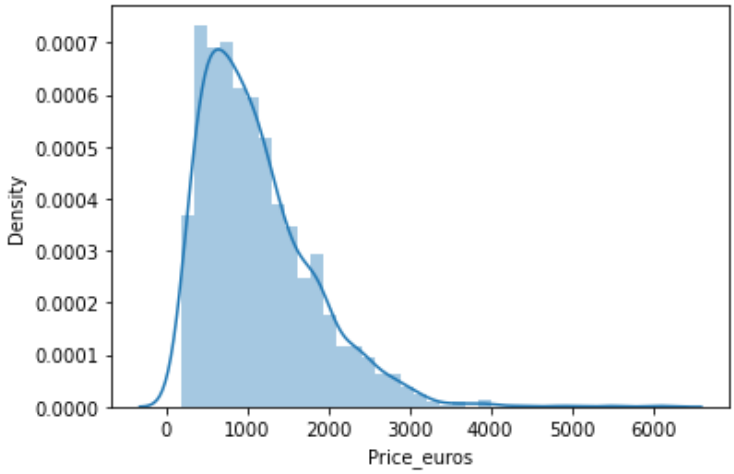
In [ ]:

```

```

Model Buliding

In [139]:

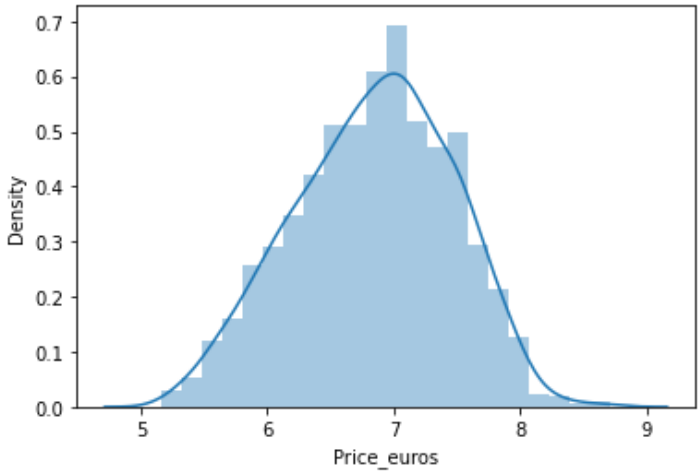
```
sns.distplot(df['Price_euros']);
```



Changing the distibution of Price_euros from right-skewed to normal distribution

In [140]:

```
sns.distplot(np.log(df['Price_euros']));
```



In [141]:

```
X=df.drop('Price_euros',axis=1)
y=np.log(df['Price_euros'])
```

In [142]:

```
X
```

Out[142]:

	Company	TypeName	Ram	Weight	Touchscreen	IPS	PPI	Cpu_brand	SSD	Gpu_brand	OS
0	Apple	Ultrabook	8	1.37	0	1	210.000000	Intel Core i5	128	Intel	Mac
1	Apple	Ultrabook	8	1.34	0	0	127.677940	Intel Core i5	0	Intel	Mac
2	HP	Notebook	8	1.86	0	0	141.211998	Intel Core i5	256	Intel	Others/No OS/Linux
3	Apple	Ultrabook	16	1.83	0	1	210.000000	Intel Core i7	512	AMD	Mac
4	Apple	Ultrabook	8	1.37	0	1	210.000000	Intel Core i5	256	Intel	Mac
...	...	...	...	...	...	...	...	...	...	...	...
1298	Lenovo	2 in 1 Convertible	4	1.80	1	1	157.350512	Intel Core i7	128	Intel	Windows
1299	Lenovo	2 in 1 Convertible	16	1.30	1	1	210.000000	Intel Core i7	512	Intel	Windows

	Company	TypeName	Ram	Weight	Touchscreen	IPS	PPI	Cpu_brand	SSD	Gpu_brand	OS
1300	Lenovo	Notebook	2	1.50	0	0	111.935204	Other Intel Processor	0	Intel	Windows
1301	HP	Notebook	6	2.19	0	0	100.454670	Intel Core i7	0	AMD	Windows
1302	Asus	Notebook	4	2.20	0	0	100.454670	Other Intel Processor	0	Intel	Windows

1302 rows × 11 columns

In [143]:

```
y
```

Out[143]:

```
0       7.200194
1       6.801216
2       6.354370
3       7.838915
4       7.497540
          ...
1298    6.458338
1299    7.312553
1300    5.433722
1301    6.638568
1302    5.910797
Name: Price_euros, Length: 1302, dtype: float64
```

Importing all the required metrics from Sci-kit learn

In [144]:

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import r2_score,mean_absolute_error
```

In [145]:

```
obj_cols=list(df.select_dtypes(include='object'))
```

In [146]:

```
obj_cols
```

Out[146]:

```
['Company', 'TypeName', 'Cpu_brand', 'Gpu_brand', 'OS']
```

Creating dummy variables for all categorical features

In [147]:

```
pd.get_dummies(data=df,columns=obj_cols)
```

Out[147]:

	Ram	Weight	Price_euros	Touchscreen	IPS	PPI	SSD	Company_Acer	Company_Apple	Company_Asus	...	C
0	8	1.37	1339.69	0	1	210.000000	128	0	1	0	...	
1	8	1.34	898.94	0	0	127.677940	0	0	1	0	...	
2	8	1.86	575.00	0	0	141.211998	256	0	0	0	...	
3	16	1.83	2537.45	0	1	210.000000	512	0	1	0	...	
4	8	1.37	1803.60	0	1	210.000000	256	0	1	0	...	
...	...	...	...	...	...	...	...	...	...	...	...	

	Ram	Weight	Price_euros	Touchscreen	IPS	PPI	SSD	Company_Acer	Company_Apple	Company_Asus	...	C
1298	4	1.80	638.00	1	1	157.350512	128	0	0	0	...	
1299	16	1.30	1499.00	1	1	210.000000	512	0	0	0	...	
1300	2	1.50	229.00	0	0	111.935204	0	0	0	0	...	
1301	6	2.19	764.00	0	0	100.454670	0	0	0	0	...	
1302	4	2.20	369.00	0	0	100.454670	0	0	0	1	...	

1302 rows × 43 columns

In [148]:

```
DF=pd.get_dummies(data=df,columns=obj_cols)
```

In [149]:

```
X=DF.drop('Price_euros',axis=1)
y=np.log(DF['Price_euros'])
```

In [150]:

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=42)
```

In [151]:

```
tree_model=DecisionTreeRegressor()
tree_model.fit(X_train,y_train)
y_pred =tree_model.predict(X_test)
```

In [152]:

```
print(r2_score(y_test,y_pred))
```

```
0.858755345839113
```

In [153]:

```
mae=mean_absolute_error(y_test,y_pred)
mae
```

Out[153]:

```
0.17266841211576753
```

In [154]:

```
#Original Mean absolute error
original_mae=np.exp(mae)
original_mae
```

Out[154]:

```
1.1884719568386144
```

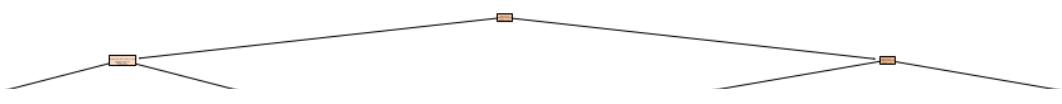
In [155]:

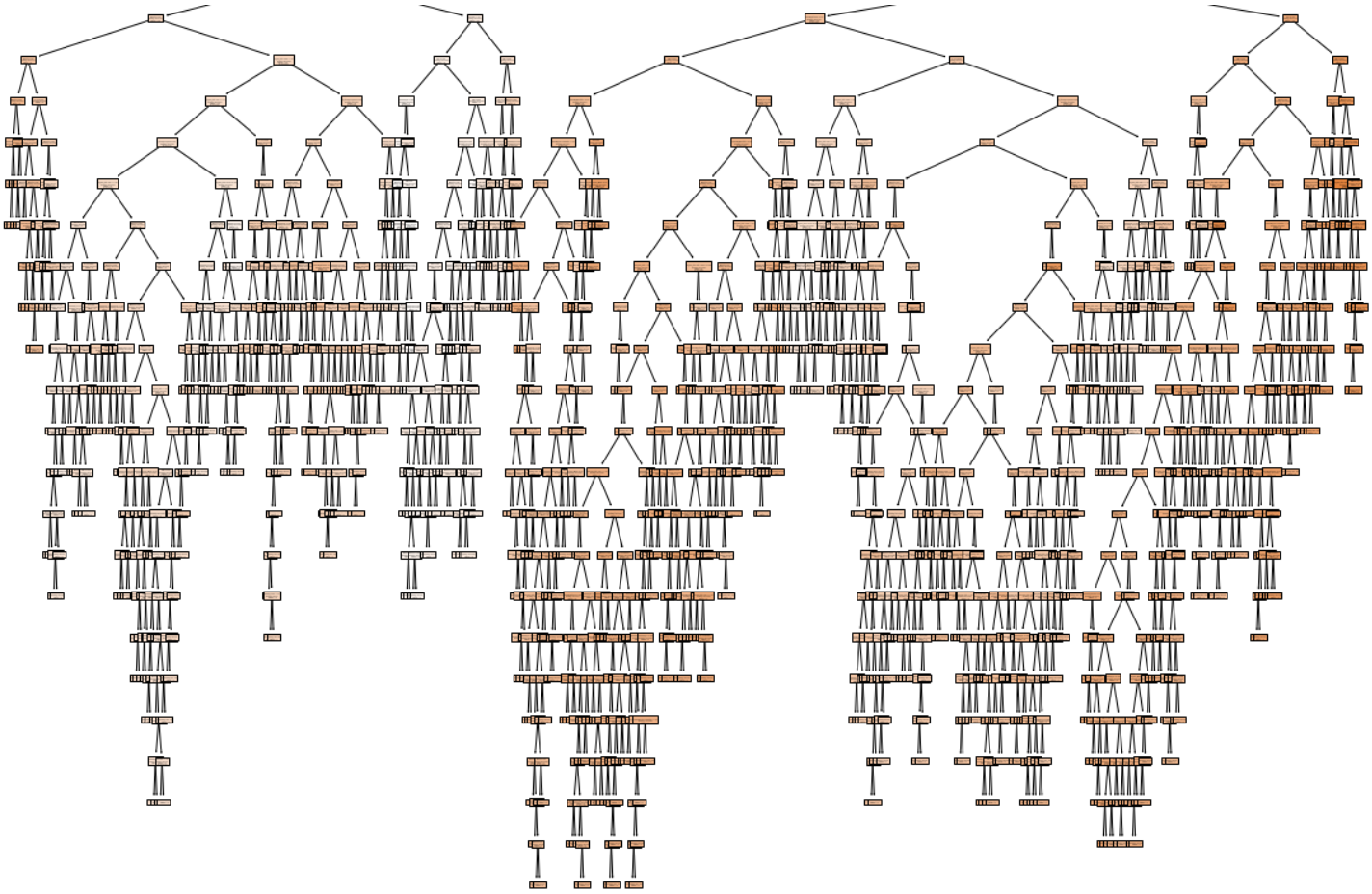
```
from sklearn import tree
```

Plotting Decision Tree Regression model

In [156]:

```
plt.figure(figsize=(20,15))
tree.plot_tree(tree_model,filled=True,feature_names=X_train.columns);
```





Comparing Train and Test performances`

In [157]:

```
def compare_train_test(model):
    y_train_predicted = model.predict(X_train)
    y_test_predicted = model.predict(X_test)
    print("Training-model Accuracy:",round(model.score(X_train,y_train)*100,2))
    print("Testing-model Accuracy:",round(model.score(X_test,y_test)*100,2))
    print('Mean Absolute error of testing data:',np.exp(mean_absolute_error(y_test,y_pred
)))
    print(np.sqrt(mean_absolute_error(y_test,y_pred)))
```

In [158]:

```
compare_train_test(tree_model)
```

```
Training-model Accuracy: 99.46
Testing-model Accuracy: 85.88
Mean Absolute error of testing data: 1.1884719568386144
0.41553388804737396
```

Pruning Decision Tree Regression model

In [159]:

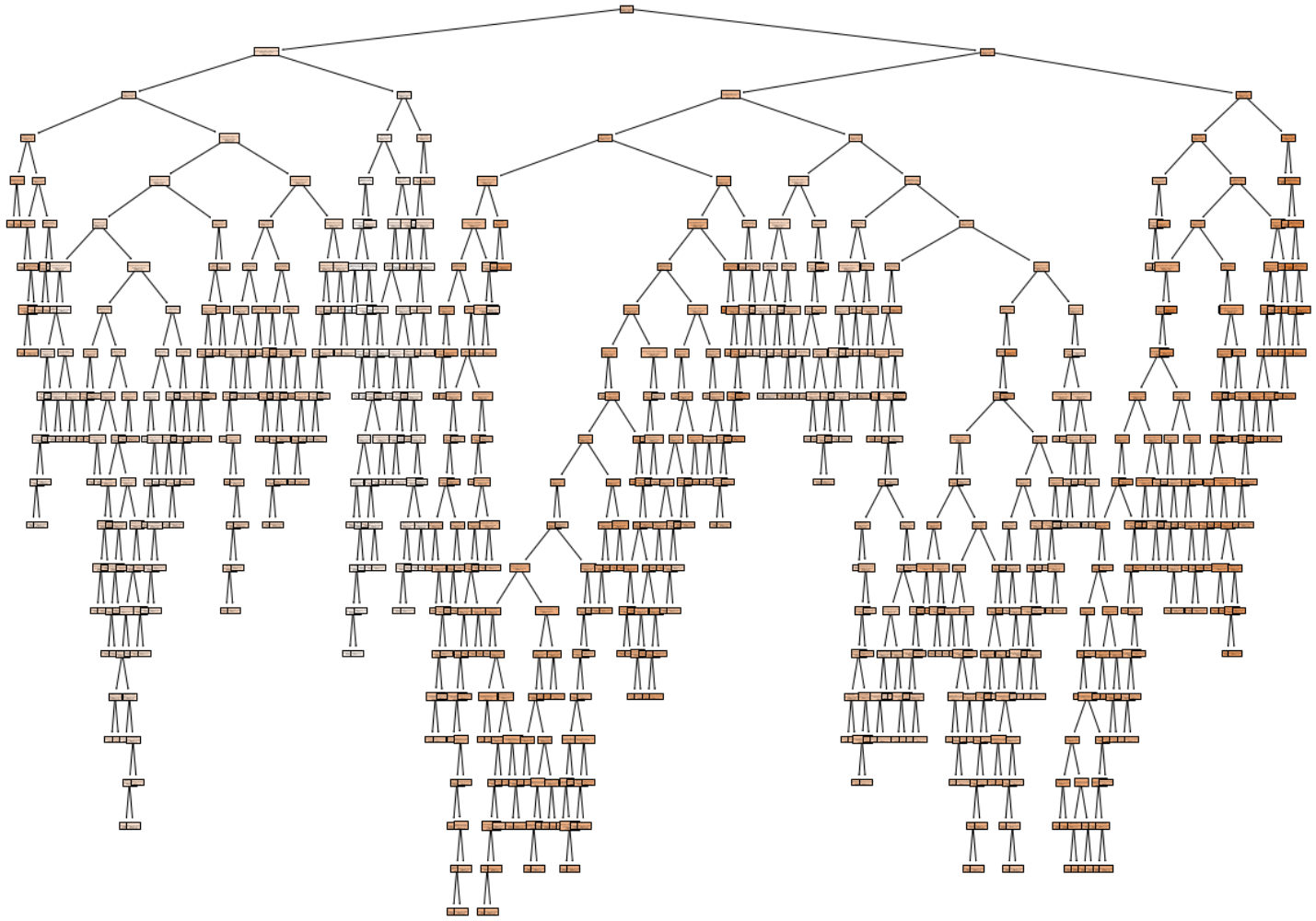
```
pruned_model=DecisionTreeRegressor(random_state=1,min_samples_split=5)
pruned_model.fit(X_train,y_train)
```

Out[159]:

```
DecisionTreeRegressor(min_samples_split=5, random_state=1)
```

In [160]:

```
plt.figure(figsize=(20,15))
tree.plot_tree(pruned_model,filled=True,feature_names=X_train.columns);
```



In [161]:

```
compare_train_test(pruned_model)
```

```
Training-model Accuracy: 97.8
Testing-model Accuracy: 84.04
Mean Absolute error of testing data: 1.1884719568386144
0.41553388804737396
```

In [ ]:

```

```

Hyperparameter tuning using GridSearch

In [162]:

```
from sklearn.model_selection import GridSearchCV
```

In [163]:

```
variable_options = {'max_depth':range(1,15),'min_samples_split': range(1,60)}
```

In [164]:

```
gs=GridSearchCV(tree_model,variable_options,scoring='r2')
gs.fit(X_train,y_train)
gs.best_params_
```

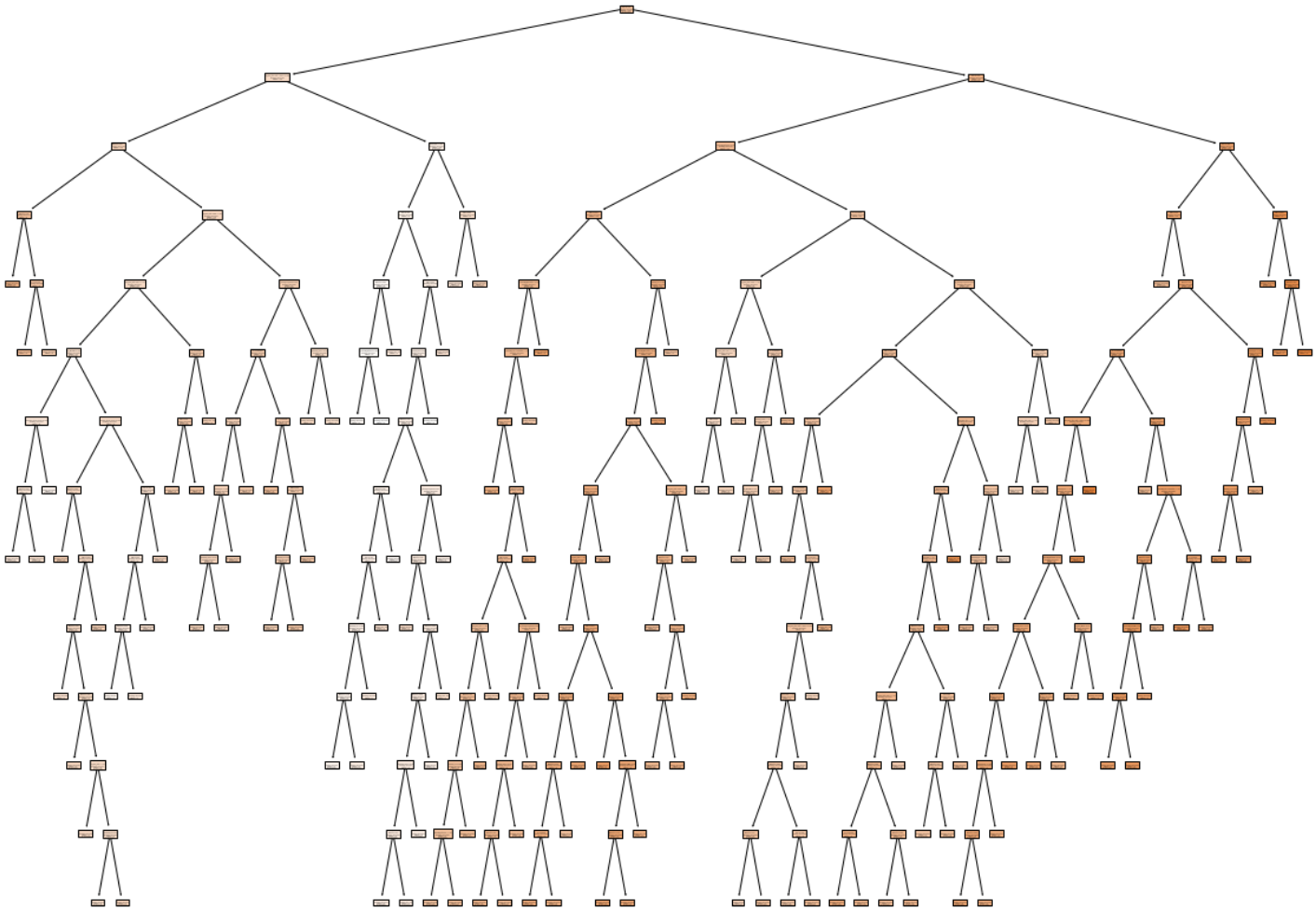
Out[164]:

```
{'max_depth': 10, 'min_samples_split': 15}
```

By using hyperparameter values plotting a Decision Tree

In [165]:

```
tree_model_new=DecisionTreeRegressor(max_depth=13,min_samples_split=13)
tree_model_new.fit(X_train,y_train)
plt.figure(figsize=(20,15))
tree.plot_tree(tree_model_new,filled=True,feature_names=X_train.columns);
```



In [166]:

```
compare_train_test(tree_model_new)
```

```
Training-model Accuracy: 92.99
Testing-model Accuracy: 85.05
Mean Absolute error of testing data: 1.1884719568386144
0.41553388804737396
```

K Fold Cross Validation

In [167]:

```
from sklearn.model_selection import KFold
kfold_validation=KFold(10)
```

In [168]:

```
from sklearn.model_selection import cross_val_score
print(cross_val_score(tree_model_new,X,y,cv=kfold_validation))
print(np.mean(cross_val_score(tree_model_new,X,y,cv=kfold_validation)))
```

```
[0.70464344 0.83529978 0.80426057 0.81257613 0.84892331 0.86040274
 0.81902698 0.7786219  0.77338606 0.8665143 ]
0.8120054820214225
```

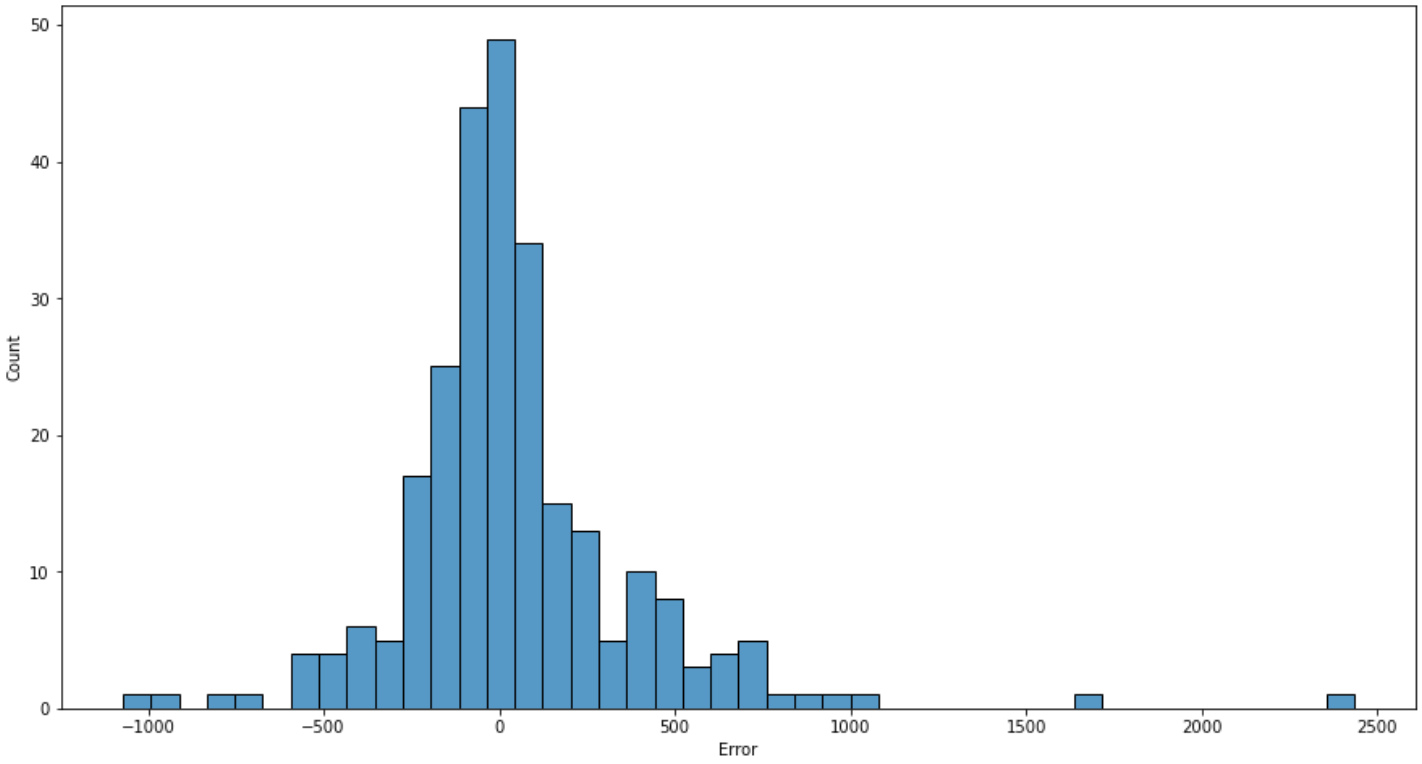
Residual deviance

In [169]:

```
y_predicted=tree_model_new.predict(X_test)
NEW_DF=pd.DataFrame({'Actual':np.exp((y_test)),'Predicted':np.exp(y_predicted)})
NEW_DF['Error']=NEW_DF['Actual']-NEW_DF['Predicted']
```

In [170]:

```
plt.figure(figsize=(15,8))
sns.histplot(data=NEW_DF,x='Error');
```



Distribution observerd for residual deviance is normal distribution so our model is performing well without any pattern

Function for a new data point

In [171]:

```
def com_name(name):
    ac,app,asu,chu,de,fuj,goog,hp,hua,lg,le,msi,mcom,mic,ra,sam,tos,ver,xia=[0 for i in
range(19)]
    if name=='Dell':
        de=1
    elif name=='Lenovo':
        le=1
    elif name=='HP':
        h=1
    elif name=='Asus':
        asu=1
    elif name=='Acer':
        ac=1
    elif name=='MSI':
        msi=1
    elif name=='Toshiba':
        tos=1
    elif name=='Apple':
        app=1
    elif name=='Samsung':
        sam=1
    elif name=='Razer':
        ra=1
```

```
    return ac,app,asu,chu,de,fuj,goog,hp,hua,lg,le,msi,mcom,mic,ra,sam,tos,ver,xia
def display(dis_type):
    t,i=0,0
    if dis_type=='Touchscreen':
        t=1
    elif dis_type=='IPS':
        i=1
    return t,i
def type_name(typ):
    co,ga,net,no,ul,wo=0,0,0,0,0,0
    if typ=='Notebook':
        no=1
    elif typ=='Gaming':
        ga=1
    elif typ=='Ultrabook':
        ul=1
    elif typ=='2 in 1':
        co=1
    elif typ=='Workstation':
        wo=1
    elif typ=='Netbook':
        net=1
    return   co,ga,net,no,ul,wo
def cpu_brand(g_name):
    amd,i3,i5,i7,other=0,0,0,0,0
    if b_name=='AMD':
        amd=1
    elif b_name=='i3':
        i3=1
    elif b_name=='i5':
        i5=1
    elif b_name=='i7':
        i7=1
    else:
        other=1
    return amd,i3,i5,i7,other
def gpu_brand(g_name):
    amd,intel,nvidia=0,0,0
    if g_name=='AMD':
        amd=1
    elif b_name=='Intel':
        intel=1
    elif b_name=='Nvidia':
        nvidia=1
    return amd,intel,nvidia
def os_type(os):
    wi,mac,oth=0,0,0
    if os=='Windows':
        wi=1
    elif os=='Mac':
        mac=1
    else:
        oth=1
    return wi,mac,oth
def price_predictor(data):
    log_p=tree_model_new.predict([data])
    price_euros=np.exp(log_p)
    return price_euros
data=[]
print("Enter the Details of the Laptop")
name=input("Company name(Dell/Lenovo/HP/Asus/Acer/MSI/Toshiba/Apple/Samsung/Razer/others)
:")
ac,app,asu,chu,de,fuj,goog,hp,hua,lg,le,msi,mcom,mic,ra,sam,tos,ver,xia=com_name(name)
print("Display")
dis=input("Display_Type(IPS/Touchscreen):")
t,i=display(dis)
print("Type_Name")
t_name=input("Laptop type name(Notebook/Netbook/Gaming/Ultrabook/2 in 1/Workstation):")
co,ga,net,no,ul,wo=type_name(t_name)
Ram=int(input("RAM(in GB): "))
Weight=float(input("Weight(in kg): "))
x_res=int(input("X Resultion:"))
```

```
y_res=int(input("y Resolution:"))
inc=float(input("Inches:"))
PPI=(((x_res*2)+(y_res*2))*(1/2))/inc
SSD=int(input("SSD(in GB):"))
print("CPU")
b_name=input("CPU brand name(AMD/Intel-i3/i5/i7/Others):")
b_amd,i3,i5,i7,other=cpu_brand(b_name)
print("GPU")
g_name=input("GPU brand name(AMD/Intel/Nvidia):")
g_amd,intel,nvidia=gpu_brand(g_name)
print("Operating System")
os=input("Name of Operating System(Windows/Mac/others):")
wi,mac,oth=os_type(os)
data=[Ram,Weight,t,i,PPI,SSD,ac,app,asu,chu,de,fuj,goog,hp,hua,lg,le,msi,mcom,mic,ra,sam
,tos,ver,xia,co,ga,net,no,ul,wo,b_amd,i3,i5,i7,other,g_amd,intel,nvidia,wi,mac,oth]
print("Predicted_price(in euros):",*price_predictor(data))
```

```
Enter the Details of the Laptop
Company name(Dell/Lenovo/HP/Asus/Acer/MSI/Toshiba/Apple/Samsung/Razer/others):Dell
Display
Display_Type(IPS/Touchscreen):Touchscreen
Type_Name
Laptop type name(Notebook/Netbook/Gaming/Ultrabook/2 in 1/Workstation):Notebook
RAM(in GB): 128
Weight(in kg): 2.6
X Resultion:1990
y Resolution:1980
Inches:4.7
SSD(in GB):256
CPU
CPU brand name(AMD/Intel-i3/i5/i7/Others):i5
GPU
GPU brand name(AMD/Intel/Nvidia):Intel
Operating System
Name of Operating System(Windows/Mac/others):Mac
Predicted_price(in euros): 2340.9551905831977
```

In [ ]: