In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

importing the dataset

In [2]:
```
df = pd.read_csv('ecommerce_customer_data_large.csv')
```

In [3]:
```
df
```

Out[3]:

	Customer ID	Purchase Date	Product Category	Product Price	Quantity	Total Purchase Amount	Payment Method	Customer Age	Returns	Customer Name
0	44605	2023-05-03 21:30:02	Home	177	1	2427	PayPal	31	1.0	John Rivera
1	44605	2021-05-16 13:57:44	Electronics	174	3	2448	PayPal	31	1.0	John Rivera
2	44605	2020-07-13 06:16:57	Books	413	1	2345	Credit Card	31	1.0	John Rivera
3	44605	2023-01-17 13:14:36	Electronics	396	3	937	Cash	31	0.0	John Rivera
4	44605	2021-05-01 11:29:27	Books	259	4	2598	PayPal	31	1.0	John Rivera
...	...	...	...	...	...	...	...	...	...	...
249995	33807	2023-01-24 12:32:18	Home	436	1	3664	Cash	63	0.0	Gabriel Williams
249996	20455	2021-06-04 05:45:25	Electronics	233	1	4374	Credit Card	66	1.0	Barry Foster
249997	28055	2022-11-10 17:11:57	Electronics	441	5	5296	Cash	63	NaN	Lisa Johnson
249998	15023	2021-06-27 14:42:12	Electronics	44	2	2517	Cash	64	1.0	Melissa Fernandez
249999	4148	2020-09-07 05:12:19	Home	307	5	3634	Cash	32	0.0	Angela Norton

250000 rows × 13 columns

info about datas

In [4]:
```
df.describe()
```

Out[4]:

	Customer ID	Product Price	Quantity	Total Purchase Amount	Customer Age	Returns	Ag
count	250000.000000	250000.000000	250000.000000	250000.000000	250000.000000	202618.000000	250000.0000
mean	25017.632092	254.742724	3.004936	2725.385196	43.798276	0.500824	43.7982
std	14412.515718	141.738104	1.414737	1442.576095	15.364915	0.500001	15.3649
min	1.000000	10.000000	1.000000	100.000000	18.000000	0.000000	18.0000
25%	12590.000000	132.000000	2.000000	1476.000000	30.000000	0.000000	30.0000
50%	25011.000000	255.000000	3.000000	2725.000000	44.000000	1.000000	44.0000
75%	37441.250000	377.000000	4.000000	3975.000000	57.000000	1.000000	57.0000
max	50000.000000	500.000000	5.000000	5350.000000	70.000000	1.000000	70.0000

In [5]:
```
df.info() #in the returns columns we see the there are missing values
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250000 entries, 0 to 249999
Data columns (total 13 columns):
 #   Column                  Non-Null Count   Dtype  
---  ------                  --------------   -----  
 0   Customer ID             250000 non-null  int64  
 1   Purchase Date           250000 non-null  object 
 2   Product Category        250000 non-null  object 
 3   Product Price           250000 non-null  int64  
 4   Quantity                250000 non-null  int64  
 5   Total Purchase Amount   250000 non-null  int64  
 6   Payment Method          250000 non-null  object 
 7   Customer Age            250000 non-null  int64  
 8   Returns                 202618 non-null  float64
 9   Customer Name           250000 non-null  object 
 10  Age                     250000 non-null  int64  
 11  Gender                  250000 non-null  object 
 12  Churn                   250000 non-null  int64  
dtypes: float64(1), int64(7), object(5)
memory usage: 24.8+ MB
```

Data wringling

In [6]:
```
df['Returns'].fillna('na',inplace = True)
df
```

Out[6]:

	Customer ID	Purchase Date	Product Category	Product Price	Quantity	Total Purchase Amount	Payment Method	Customer Age	Returns	Customer Name
0	44605	2023-05-03 21:30:02	Home	177	1	2427	PayPal	31	1.0	John Rivera
1	44605	2021-05-16 13:57:44	Electronics	174	3	2448	PayPal	31	1.0	John Rivera
2	44605	2020-07-13 06:16:57	Books	413	1	2345	Credit Card	31	1.0	John Rivera
3	44605	2023-01-17 13:14:36	Electronics	396	3	937	Cash	31	0.0	John Rivera
4	44605	2021-05-01 11:29:27	Books	259	4	2598	PayPal	31	1.0	John Rivera
...	...	...	...	...	...	...	...	...	...	...
249995	33807	2023-01-24 12:32:18	Home	436	1	3664	Cash	63	0.0	Gabriel Williams
249996	20455	2021-06-04 05:45:25	Electronics	233	1	4374	Credit Card	66	1.0	Barry Foster
249997	28055	2022-11-10 17:11:57	Electronics	441	5	5296	Cash	63	na	Lisa Johnson
249998	15023	2021-06-27 14:42:12	Electronics	44	2	2517	Cash	64	1.0	Melissa Fernandez
249999	4148	2020-09-07 05:12:19	Home	307	5	3634	Cash	32	0.0	Angela Norton

250000 rows × 13 columns

In [7]:
```
df.isna()
```

Out[7]:

	Customer ID	Purchase Date	Product Category	Product Price	Quantity	Total Purchase Amount	Payment Method	Customer Age	Returns	Customer Name
0	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...
249995	False	False	False	False	False	False	False	False	False	False
249996	False	False	False	False	False	False	False	False	False	False
249997	False	False	False	False	False	False	False	False	False	False
249998	False	False	False	False	False	False	False	False	False	False
249999	False	False	False	False	False	False	False	False	False	False

250000 rows × 13 columns

In [8]:
```
df[df.duplicated()] #there no duplicate values
```

Out[8]:

Customer ID	Purchase Date	Product Category	Product Price	Quantity	Total Purchase Amount	Payment Method	Customer Age	Returns	Customer Name	Age	C

In [9]:
```
df1 = pd.read_excel('Customer Call List.xlsx') #so to check whether the duplicate functi
```

In [10]:
```
df1
```

Out[10]:

	CustomerID	First_Name	Last_Name	Phone_Number	Address	Paying Customer	Do_Not_Contact	Not_Useful_
0	1001	Frodo	Baggins	123-545-5421	123 Shire Lane, Shire	Yes	No	
1	1002	Abed	Nadir	123/643/9775	93 West Main Street	No	Yes	
2	1003	Walter	/White	7066950392	298 Drugs Driveway	N	NaN	
3	1004	Dwight	Schrute	123-543-2345	980 Paper Avenue, Pennsylvania, 18503	Yes	Y	
4	1005	Jon	Snow	876|678|3469	123 Dragons Road	Y	No	
5	1006	Ron	Swanson	304-762-2467	768 City Parkway	Yes	Yes	
6	1007	Jeff	Winger	NaN	1209 South Street	No	No	
7	1008	Sherlock	Holmes	876|678|3469	98 Clue Drive	N	No	
8	1009	Gandalf	NaN	N/a	123 Middle Earth	Yes	NaN	
9	1010	Peter	Parker	123-545-5421	25th Main Street, New York	Yes	No	
10	1011	Samwise	Gamgee	NaN	612 Shire Lane, Shire	Yes	No	
11	1012	Harry	...Potter	7066950392	2394 Hogwarts Avenue	Y	NaN	
12	1013	Don	Draper	123-543-2345	2039 Main Street	Yes	N	
13	1014	Leslie	Knope	876|678|3469	343 City Parkway	Yes	No	
14	1015	Toby	Flenderson_	304-762-2467	214 HR Avenue	N	No	
15	1016	Ron	Weasley	123-545-5421	2395 Hogwarts Avenue	No	N	
16	1017	Michael	Scott	123/643/9775	121 Paper Avenue, Pennsylvania	Yes	No	
17	1018	Clark	Kent	7066950392	3498 Super Lane	Y	NaN	
18	1019	Creed	Braton	N/a	N/a	N/a	Yes	
19	1020	Anakin	Skywalker	876|678|3469	910 Tatooine Road, Tatooine	Yes	N	
20	1020	Anakin	Skywalker	876|678|3469	910 Tatooine Road, Tatooine	Yes	N	

In [11]:
```
df1[df1.duplicated()] #it works
```

Out[11]:

	CustomerID	First_Name	Last_Name	Phone_Number	Address	Paying Customer	Do_Not_Contact	Not_Useful_Colu
20	1020	Anakin	Skywalker	876|678|3469	910 Tatooine Road, Tatooine	Yes	N	T

In [12]:
```
# now the total purchase amount does not match the price and quantity columns so lets ch
df
```

Out[12]:

	Customer ID	Purchase Date	Product Category	Product Price	Quantity	Total Purchase Amount	Payment Method	Customer Age	Returns	Customer Name
0	44605	2023-05-03 21:30:02	Home	177	1	2427	PayPal	31	1.0	John Rivera
1	44605	2021-05-16 13:57:44	Electronics	174	3	2448	PayPal	31	1.0	John Rivera
2	44605	2020-07-13 06:16:57	Books	413	1	2345	Credit Card	31	1.0	John Rivera
3	44605	2023-01-17 13:14:36	Electronics	396	3	937	Cash	31	0.0	John Rivera
4	44605	2021-05-01 11:29:27	Books	259	4	2598	PayPal	31	1.0	John Rivera
...	...	...	...	...	...	...	...	...	...	...
249995	33807	2023-01-24 12:32:18	Home	436	1	3664	Cash	63	0.0	Gabriel Williams
249996	20455	2021-06-04 05:45:25	Electronics	233	1	4374	Credit Card	66	1.0	Barry Foster
249997	28055	2022-11-10 17:11:57	Electronics	441	5	5296	Cash	63	na	Lisa Johnson
249998	15023	2021-06-27 14:42:12	Electronics	44	2	2517	Cash	64	1.0	Melissa Fernandez
249999	4148	2020-09-07 05:12:19	Home	307	5	3634	Cash	32	0.0	Angela Norton

250000 rows × 13 columns

In [13]:
```
df['Total Amount Purchased']  = df['Product Price'] * df['Quantity']
```

In [14]:
```
df #now the amount has been corrected so lets drop the total purchase amount
```

Out[14]:

	Customer ID	Purchase Date	Product Category	Product Price	Quantity	Total Purchase Amount	Payment Method	Customer Age	Returns	Customer Name
0	44605	2023-05-03 21:30:02	Home	177	1	2427	PayPal	31	1.0	John Rivera
1	44605	2021-05-16 13:57:44	Electronics	174	3	2448	PayPal	31	1.0	John Rivera
2	44605	2020-07-13 06:16:57	Books	413	1	2345	Credit Card	31	1.0	John Rivera
3	44605	2023-01-17 13:14:36	Electronics	396	3	937	Cash	31	0.0	John Rivera
4	44605	2021-05-01 11:29:27	Books	259	4	2598	PayPal	31	1.0	John Rivera
...	...	...	...	...	...	...	...	...	...	...
249995	33807	2023-01-24 12:32:18	Home	436	1	3664	Cash	63	0.0	Gabriel Williams
249996	20455	2021-06-04 05:45:25	Electronics	233	1	4374	Credit Card	66	1.0	Barry Foster
249997	28055	2022-11-10 17:11:57	Electronics	441	5	5296	Cash	63	na	Lisa Johnson
249998	15023	2021-06-27 14:42:12	Electronics	44	2	2517	Cash	64	1.0	Melissa Fernandez
249999	4148	2020-09-07 05:12:19	Home	307	5	3634	Cash	32	0.0	Angela Norton

250000 rows × 14 columns

In [15]:
```
# customer age columns is the same as age column so lets drop the purchase amount and cu

df.drop(columns = ['Total Purchase Amount','Customer Age'], inplace = True)
df
```

Out[15]:

	Customer ID	Purchase Date	Product Category	Product Price	Quantity	Payment Method	Returns	Customer Name	Age	Gender	Churn
0	44605	2023-05-03 21:30:02	Home	177	1	PayPal	1.0	John Rivera	31	Female	0
1	44605	2021-05-16 13:57:44	Electronics	174	3	PayPal	1.0	John Rivera	31	Female	0
2	44605	2020-07-13 06:16:57	Books	413	1	Credit Card	1.0	John Rivera	31	Female	0
3	44605	2023-01-17 13:14:36	Electronics	396	3	Cash	0.0	John Rivera	31	Female	0
4	44605	2021-05-01 11:29:27	Books	259	4	PayPal	1.0	John Rivera	31	Female	0
...	...	...	...	...	...	...	...	...	...	...	...
249995	33807	2023-01-24 12:32:18	Home	436	1	Cash	0.0	Gabriel Williams	63	Male	0
249996	20455	2021-06-04 05:45:25	Electronics	233	1	Credit Card	1.0	Barry Foster	66	Female	0
249997	28055	2022-11-10 17:11:57	Electronics	441	5	Cash	na	Lisa Johnson	63	Female	0
249998	15023	2021-06-27 14:42:12	Electronics	44	2	Cash	1.0	Melissa Fernandez	64	Male	0
249999	4148	2020-09-07 05:12:19	Home	307	5	Cash	0.0	Angela Norton	32	Male	0

250000 rows × 12 columns

In [16]:
```
# now lets set the date to starting


df.sort_values('Purchase Date', inplace = True)
df
```

Out[16]:

	Customer ID	Purchase Date	Product Category	Product Price	Quantity	Payment Method	Returns	Customer Name	Age	Gender	Churn
231894	11789	2020-01-01 00:07:26	Clothing	426	4	Cash	1.0	Matthew Davis	45	Male	0
159409	48592	2020-01-01 00:11:40	Clothing	160	4	Cash	0.0	Tina Phillips	49	Male	0
56927	30486	2020-01-01 00:15:47	Clothing	230	4	Credit Card	na	Lance Colon	35	Male	1
228275	25222	2020-01-01 00:24:27	Clothing	17	1	Cash	1.0	John Delgado	34	Female	0
181085	6380	2020-01-01 00:28:45	Home	363	5	PayPal	0.0	Ashlee Johnson	54	Female	1
...	...	...	...	...	...	...	...	...	...	...	...
118223	28694	2023-09-13 18:02:42	Home	437	2	Credit Card	1.0	Jennifer Nichols	25	Female	0
149965	37904	2023-09-13 18:16:46	Home	168	3	Cash	0.0	Darren Summers	40	Male	0
107000	7982	2023-09-13 18:33:30	Home	28	4	PayPal	0.0	Jessica Clark	68	Female	0
125841	37031	2023-09-13 18:37:07	Books	240	3	Credit Card	0.0	Julia Campbell	25	Female	0
5163	43799	2023-09-13 18:42:49	Clothing	43	2	PayPal	0.0	Jessica Jones	66	Female	0

250000 rows × 12 columns

In [17]:
```
# the index is messed so lets reset the index

df.reset_index(drop = True) #drop is to drop the previous index in df
```

Out[17]:

	Customer ID	Purchase Date	Product Category	Product Price	Quantity	Payment Method	Returns	Customer Name	Age	Gender	Churn
0	11789	2020-01-01 00:07:26	Clothing	426	4	Cash	1.0	Matthew Davis	45	Male	0
1	48592	2020-01-01 00:11:40	Clothing	160	4	Cash	0.0	Tina Phillips	49	Male	0
2	30486	2020-01-01 00:15:47	Clothing	230	4	Credit Card	na	Lance Colon	35	Male	1
3	25222	2020-01-01 00:24:27	Clothing	17	1	Cash	1.0	John Delgado	34	Female	0
4	6380	2020-01-01 00:28:45	Home	363	5	PayPal	0.0	Ashlee Johnson	54	Female	1
...	...	...	...	...	...	...	...	...	...	...	...
249995	28694	2023-09-13 18:02:42	Home	437	2	Credit Card	1.0	Jennifer Nichols	25	Female	0
249996	37904	2023-09-13 18:16:46	Home	168	3	Cash	0.0	Darren Summers	40	Male	0
249997	7982	2023-09-13 18:33:30	Home	28	4	PayPal	0.0	Jessica Clark	68	Female	0
249998	37031	2023-09-13 18:37:07	Books	240	3	Credit Card	0.0	Julia Campbell	25	Female	0
249999	43799	2023-09-13 18:42:49	Clothing	43	2	PayPal	0.0	Jessica Jones	66	Female	0

250000 rows × 12 columns

In [18]:
```
#so lets create a two columns year and month so that it will help in the visualization l

df['Purchase Date'] = pd.to_datetime(df['Purchase Date']) #changing the datatype to date

df['year'] = df['Purchase Date'].dt.year
df['month'] = df['Purchase Date'].dt.month
df
```

Out[18]:

	Customer ID	Purchase Date	Product Category	Product Price	Quantity	Payment Method	Returns	Customer Name	Age	Gender	Churn
231894	11789	2020-01-01 00:07:26	Clothing	426	4	Cash	1.0	Matthew Davis	45	Male	0
159409	48592	2020-01-01 00:11:40	Clothing	160	4	Cash	0.0	Tina Phillips	49	Male	0
56927	30486	2020-01-01 00:15:47	Clothing	230	4	Credit Card	na	Lance Colon	35	Male	1
228275	25222	2020-01-01 00:24:27	Clothing	17	1	Cash	1.0	John Delgado	34	Female	0
181085	6380	2020-01-01 00:28:45	Home	363	5	PayPal	0.0	Ashlee Johnson	54	Female	1
...	...	...	...	...	...	...	...	...	...	...	...
118223	28694	2023-09-13 18:02:42	Home	437	2	Credit Card	1.0	Jennifer Nichols	25	Female	0
149965	37904	2023-09-13 18:16:46	Home	168	3	Cash	0.0	Darren Summers	40	Male	0
107000	7982	2023-09-13 18:33:30	Home	28	4	PayPal	0.0	Jessica Clark	68	Female	0
125841	37031	2023-09-13 18:37:07	Books	240	3	Credit Card	0.0	Julia Campbell	25	Female	0
5163	43799	2023-09-13 18:42:49	Clothing	43	2	PayPal	0.0	Jessica Jones	66	Female	0

250000 rows × 14 columns

In [19]:
```
# first we have to create a new dataframe with columns

one = ['Customer ID','Customer Name','Age','Gender','Purchase Date','Product Category','
```

In [20]:
```
new = df[one]
```

In [21]:
```
#now the new dataframe has been created

new
```

Out[21]:

	Customer ID	Customer Name	Age	Gender	Purchase Date	Product Category	Product Price	Quantity	Total Amount Purchased	Payment Method	Retu
231894	11789	Matthew Davis	45	Male	2020-01-01 00:07:26	Clothing	426	4	1704	Cash	
159409	48592	Tina Phillips	49	Male	2020-01-01 00:11:40	Clothing	160	4	640	Cash	
56927	30486	Lance Colon	35	Male	2020-01-01 00:15:47	Clothing	230	4	920	Credit Card	
228275	25222	John Delgado	34	Female	2020-01-01 00:24:27	Clothing	17	1	17	Cash	
181085	6380	Ashlee Johnson	54	Female	2020-01-01 00:28:45	Home	363	5	1815	PayPal	
...	...	...	...	...	...	...	...	...	...	...	...
118223	28694	Jennifer Nichols	25	Female	2023-09-13 18:02:42	Home	437	2	874	Credit Card	
149965	37904	Darren Summers	40	Male	2023-09-13 18:16:46	Home	168	3	504	Cash	
107000	7982	Jessica Clark	68	Female	2023-09-13 18:33:30	Home	28	4	112	PayPal	
125841	37031	Julia Campbell	25	Female	2023-09-13 18:37:07	Books	240	3	720	Credit Card	
5163	43799	Jessica Jones	66	Female	2023-09-13 18:42:49	Clothing	43	2	86	PayPal	

250000 rows × 14 columns

In [22]:
```
new.reset_index(drop = True, inplace = True) #so the index has been reset
```

In [23]:
```
new
```

Out[23]:

	Customer ID	Customer Name	Age	Gender	Purchase Date	Product Category	Product Price	Quantity	Total Amount Purchased	Payment Method	Retu
0	11789	Matthew Davis	45	Male	2020-01-01 00:07:26	Clothing	426	4	1704	Cash	
1	48592	Tina Phillips	49	Male	2020-01-01 00:11:40	Clothing	160	4	640	Cash	
2	30486	Lance Colon	35	Male	2020-01-01 00:15:47	Clothing	230	4	920	Credit Card	
3	25222	John Delgado	34	Female	2020-01-01 00:24:27	Clothing	17	1	17	Cash	
4	6380	Ashlee Johnson	54	Female	2020-01-01 00:28:45	Home	363	5	1815	PayPal	
...	...	...	...	...	...	...	...	...	...	...	...
249995	28694	Jennifer Nichols	25	Female	2023-09-13 18:02:42	Home	437	2	874	Credit Card	
249996	37904	Darren Summers	40	Male	2023-09-13 18:16:46	Home	168	3	504	Cash	
249997	7982	Jessica Clark	68	Female	2023-09-13 18:33:30	Home	28	4	112	PayPal	
249998	37031	Julia Campbell	25	Female	2023-09-13 18:37:07	Books	240	3	720	Credit Card	
249999	43799	Jessica Jones	66	Female	2023-09-13 18:42:49	Clothing	43	2	86	PayPal	

250000 rows × 14 columns

In [24]:
```
new['Returns'].replace('na',0,inplace = True)
new
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_2364\2003245549.py:1: SettingWithCopyWarnin
g:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  new['Returns'].replace('na',0,inplace = True)
```

Out[24]:

	Customer ID	Customer Name	Age	Gender	Purchase Date	Product Category	Product Price	Quantity	Total Amount Purchased	Payment Method	Retu
0	11789	Matthew Davis	45	Male	2020-01-01 00:07:26	Clothing	426	4	1704	Cash	
1	48592	Tina Phillips	49	Male	2020-01-01 00:11:40	Clothing	160	4	640	Cash	
2	30486	Lance Colon	35	Male	2020-01-01 00:15:47	Clothing	230	4	920	Credit Card	
3	25222	John Delgado	34	Female	2020-01-01 00:24:27	Clothing	17	1	17	Cash	
4	6380	Ashlee Johnson	54	Female	2020-01-01 00:28:45	Home	363	5	1815	PayPal	
...	...	...	...	...	...	...	...	...	...	...	
249995	28694	Jennifer Nichols	25	Female	2023-09-13 18:02:42	Home	437	2	874	Credit Card	
249996	37904	Darren Summers	40	Male	2023-09-13 18:16:46	Home	168	3	504	Cash	
249997	7982	Jessica Clark	68	Female	2023-09-13 18:33:30	Home	28	4	112	PayPal	
249998	37031	Julia Campbell	25	Female	2023-09-13 18:37:07	Books	240	3	720	Credit Card	
249999	43799	Jessica Jones	66	Female	2023-09-13 18:42:49	Clothing	43	2	86	PayPal	

250000 rows × 14 columns

In [25]:
```
#year_sales = new[new['Product Category','year','Total Amount Purchased']]

year_sales = new.groupby(['year']).agg({'Total Amount Purchased':['sum']})
year_sales
```

Out[25]:

	Total Amount Purchased
	sum
year	
2020	51931213
2021	51594786
2022	51774966
2023	36187700

In [26]:
```
month_sales = new.groupby(['month','year']).agg({'Total Amount Purchased':['sum']})
month_sales
```

Out[26]:

		Total Amount Purchased
		sum
month	year	
1	2020	4327930
	2021	4269385
	2022	4366436
	2023	4360942
2	2020	4108909
	2021	4020191
	2022	3932018
	2023	3906545
3	2020	4377942
	2021	4315246
	2022	4425448
	2023	4463750
4	2020	4243816
	2021	4231634
	2022	4405934
	2023	4195885
5	2020	4395282
	2021	4459474
	2022	4367214
	2023	4515550
6	2020	4264438
	2021	4270987
	2022	4234254
	2023	4195255
7	2020	4444778
	2021	4374904
	2022	4542030
	2023	4287025
8	2020	4527160
	2021	4501859
	2022	4306440
	2023	4443714
9	2020	4231220
	2021	4243240
	2022	4203367
	2023	1819034
10	2020	4433953

		Total Amount Purchased
		sum
month	year	
	2021	4289552
	2022	4271126
11	2020	4241670
	2021	4153361
	2022	4335945
12	2020	4334115
	2021	4464953
	2022	4384754

In [36]:
```
pd.reset_option('display.max_rows')
```

In [28]:
```
product_year_sales = new.groupby(['Product Category','year']).agg({'Total Amount Purchas
year_sales
```

Out[28]:

	Total Amount Purchased
	sum
year	
2020	51931213
2021	51594786
2022	51774966
2023	36187700

In [37]:
```
product_month_sales = new.groupby(['Product Category','month','year']).agg({'Total Amoun
product_month_sales
```

Out[37]:

			Total Amount Purchased
			sum
Product Category	**month**	**year**	
Books	**1**	**2020**	1092777
		2021	1073124
		2022	1077693
		2023	1093680
	2	**2020**	1040295
...	**...**	**...**	...
Home	**11**	**2021**	1088477
		2022	1053528
	12	**2020**	1061610
		2021	1116257
		2022	1089876

180 rows × 1 columns

In [ ]:
```
new['Returns'].replace('na','N/A',inplace = True)
new
```

In [ ]:
```
# so the data has been cleaned and processed now analysis of the data begins
```

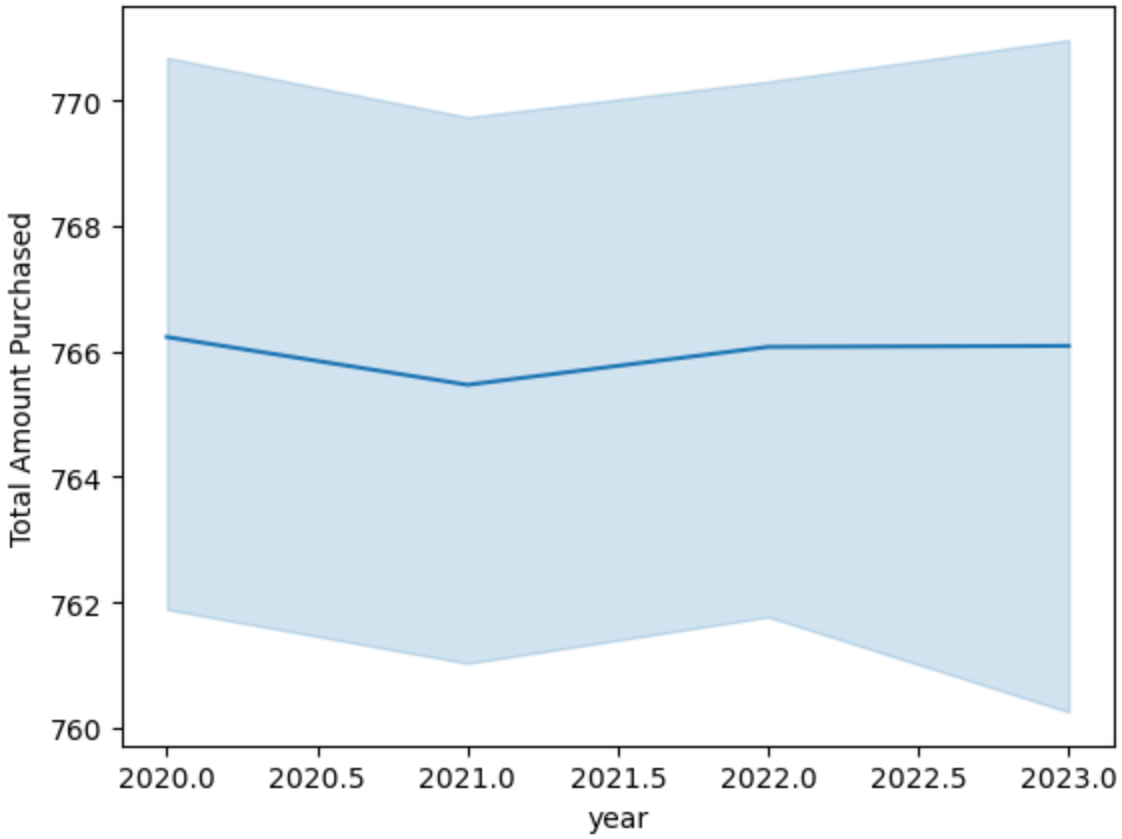
Data visualization

In [ ]:
```
new
```

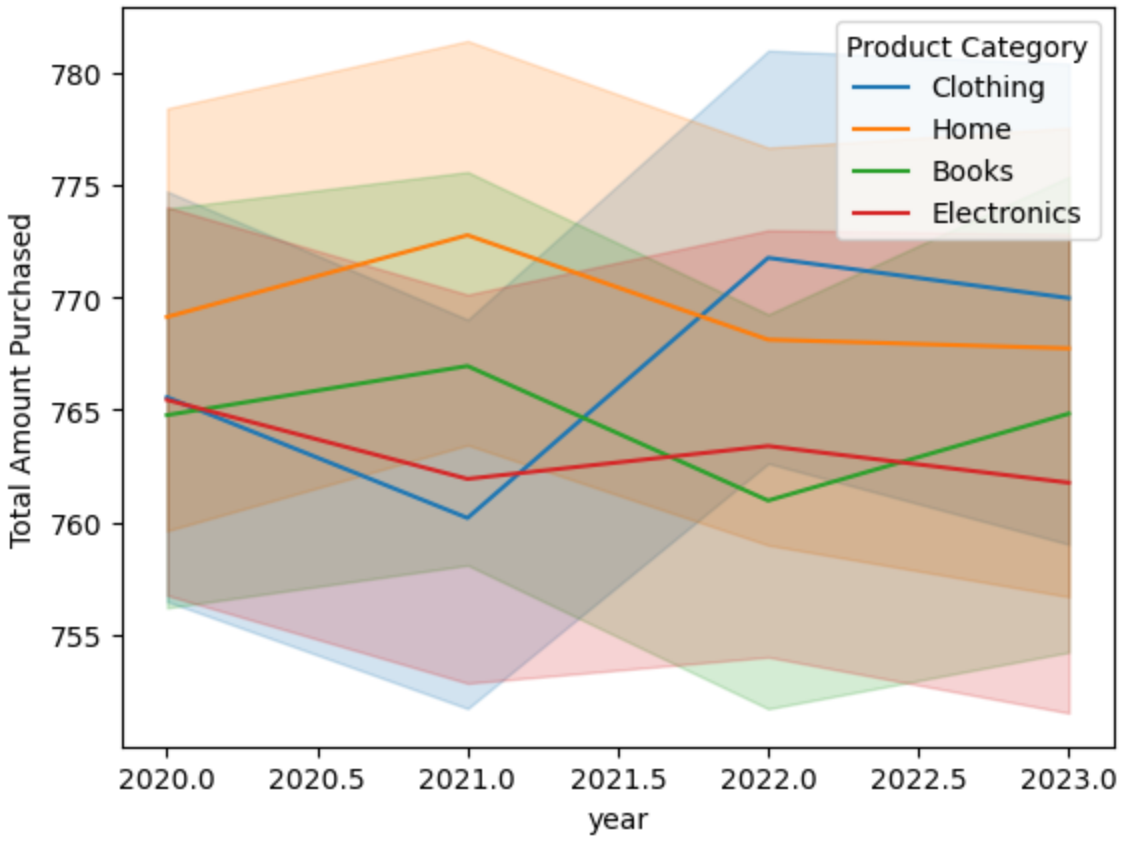
In [ ]:
```
#### so first viz
year and total purcahse *
month and total purchase *
Gender and product category *
gender and total purchase *
gender and payment *
payment and total purchase *
returns and product category
churn and gender *
churn and year *
chrun and month *
age and product categroy
age and payment ####
```

In [ ]:
```
sns.set(style = 'whitegrid')
```

In [40]:
```
sns.lineplot(x = 'year', y = 'Total Amount Purchased', data = new)
plt.show()
```



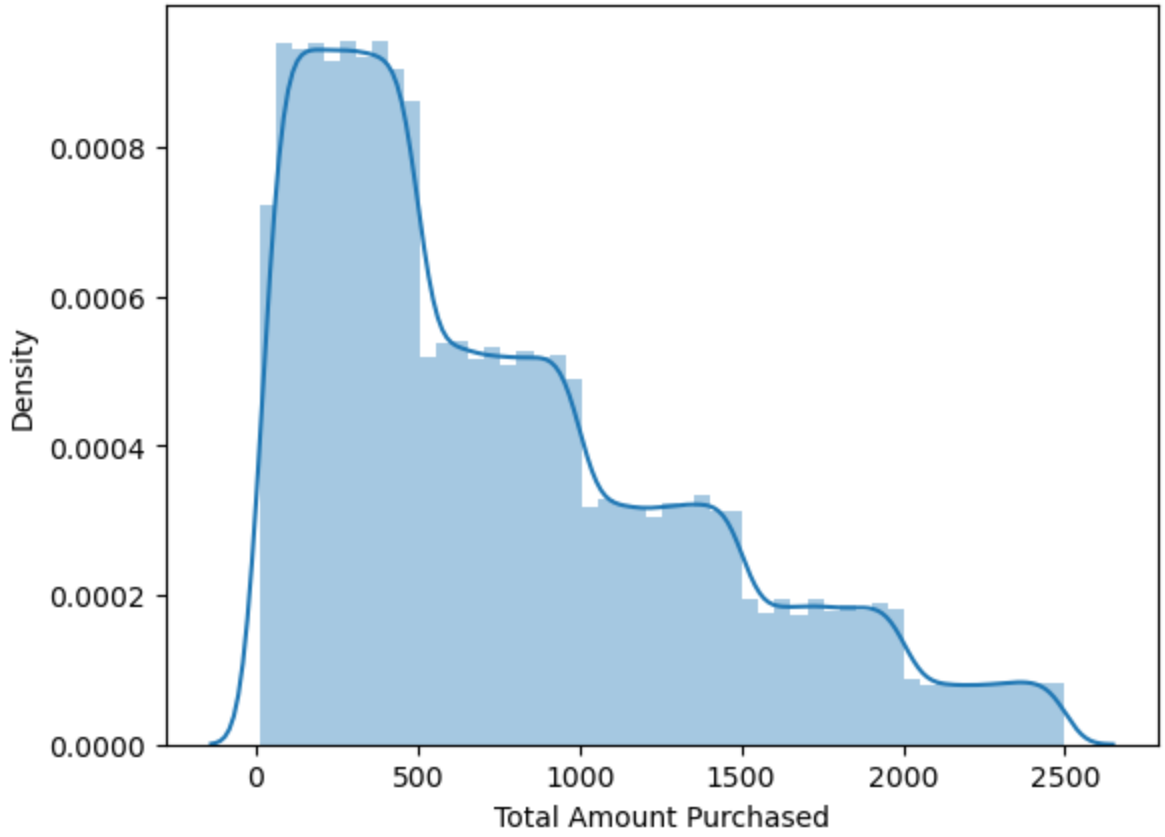
In [39]:
```
sns.lineplot(x = 'year', y = 'Total Amount Purchased',hue = 'Product Category',markers =
plt.show()
```



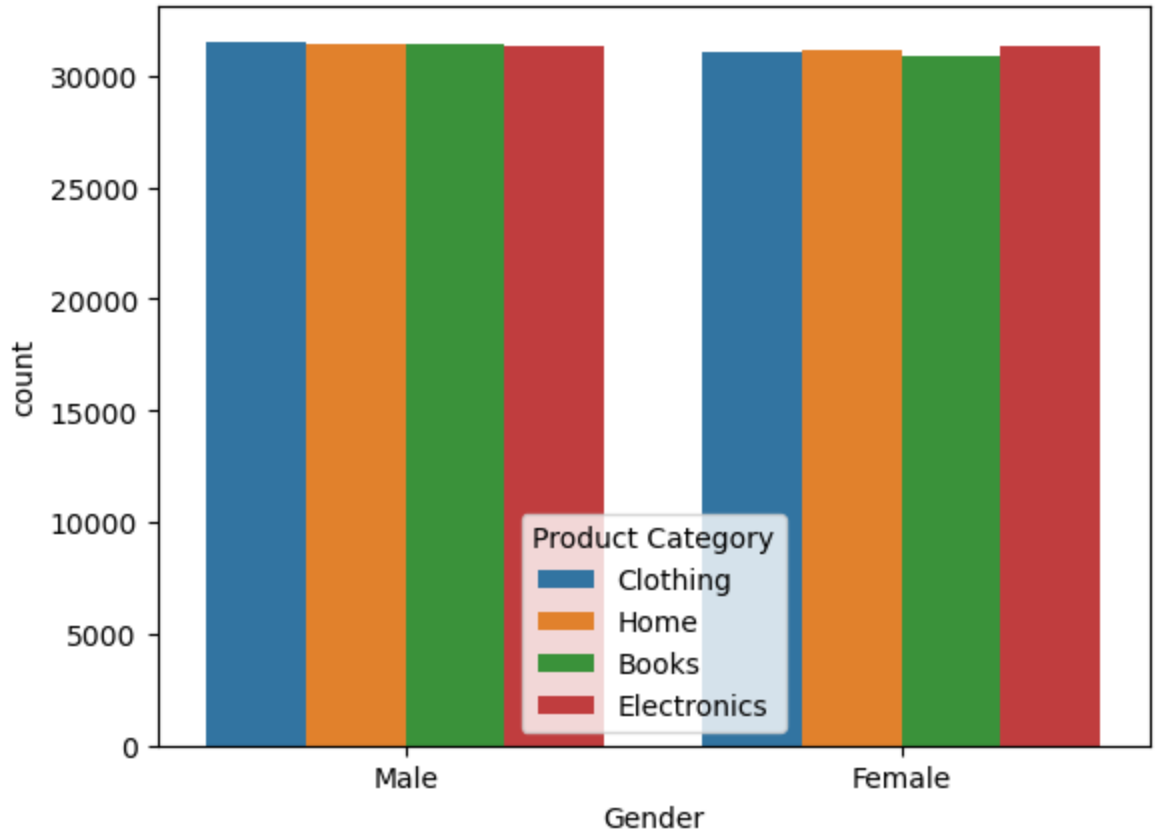
In [42]:
```
sns.distplot(new['Total Amount Purchased'])
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_2364\3639284787.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(new['Total Amount Purchased'])
```

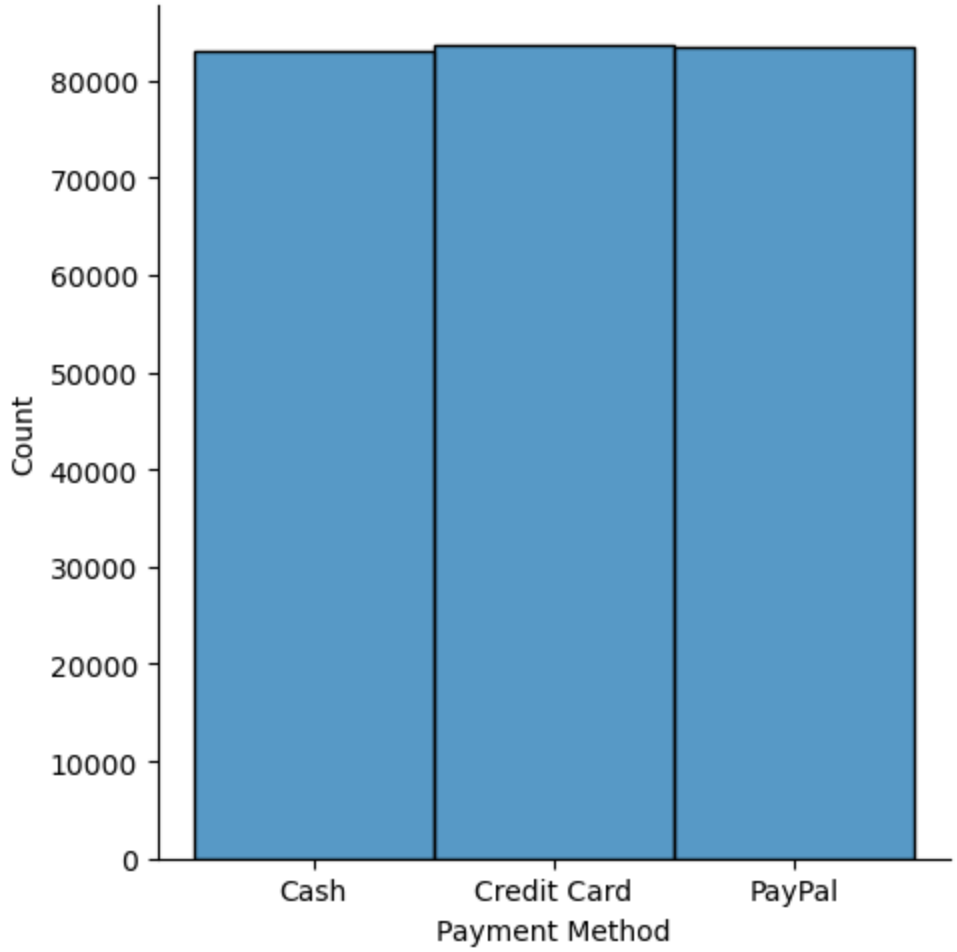
Out[42]:
```
<Axes: xlabel='Total Amount Purchased', ylabel='Density'>
```



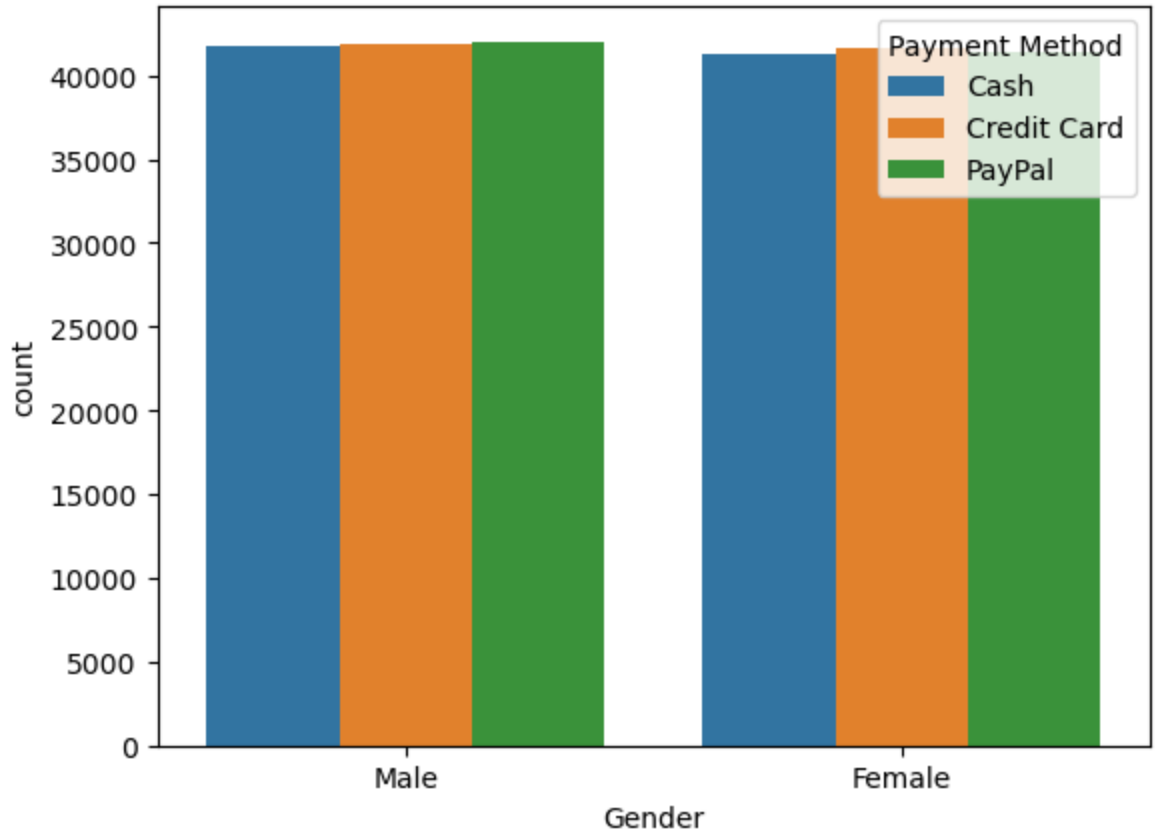
In [43]:
```
sns.countplot(x = 'Gender',hue = 'Product Category', data = new)
plt.show()
```



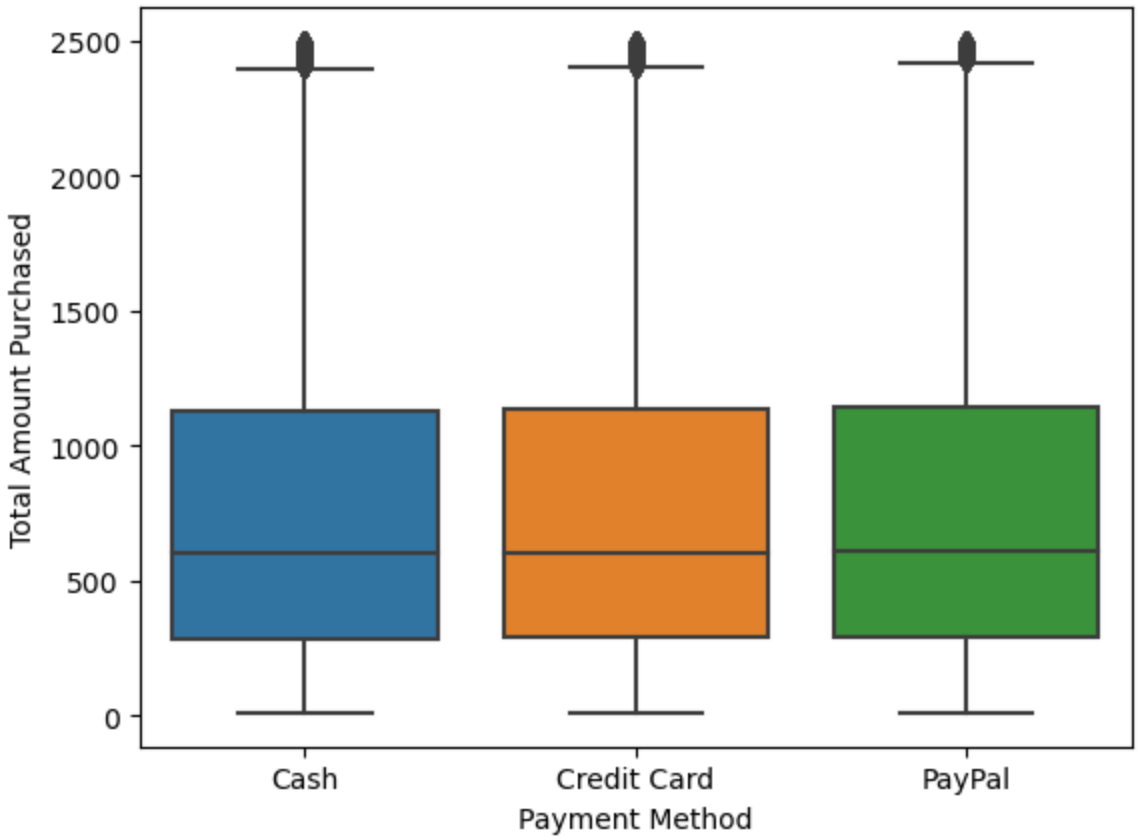
In [48]:
```
sns.displot(new['Payment Method'])
plt.show()
```



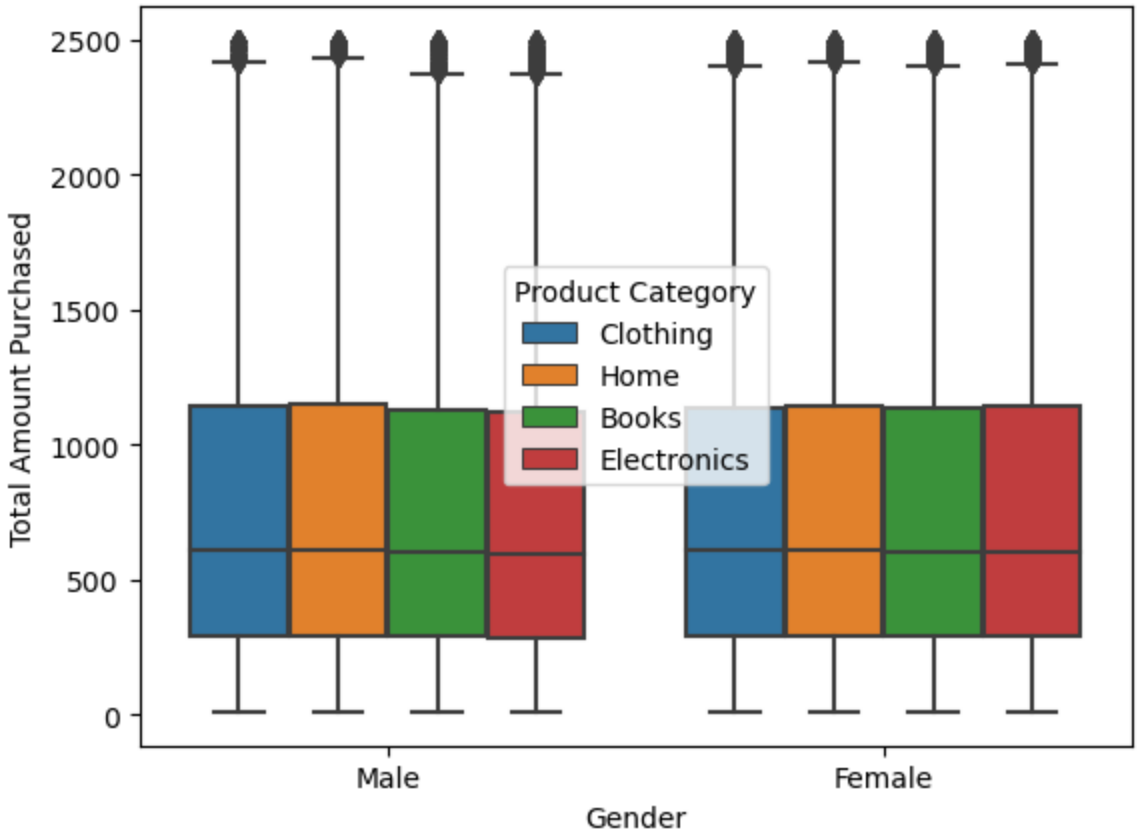
In [49]:
```
sns.countplot(x = 'Gender',hue = 'Payment Method', data = new)
plt.show()
```



In [50]:
```
sns.boxplot(x = 'Payment Method', y = 'Total Amount Purchased', data = new)
plt.show()
```

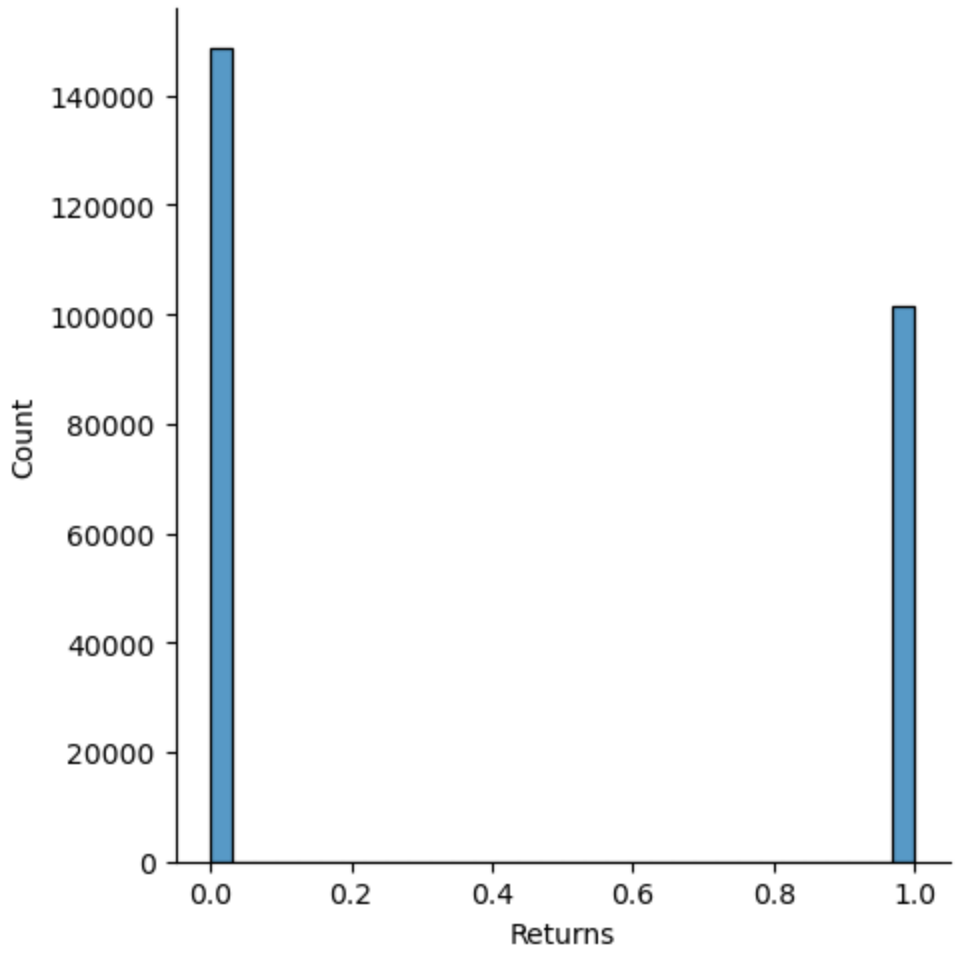


In [51]:
```
sns.boxplot(x = 'Gender', y = 'Total Amount Purchased',hue = 'Product Category', data =
plt.show()
```



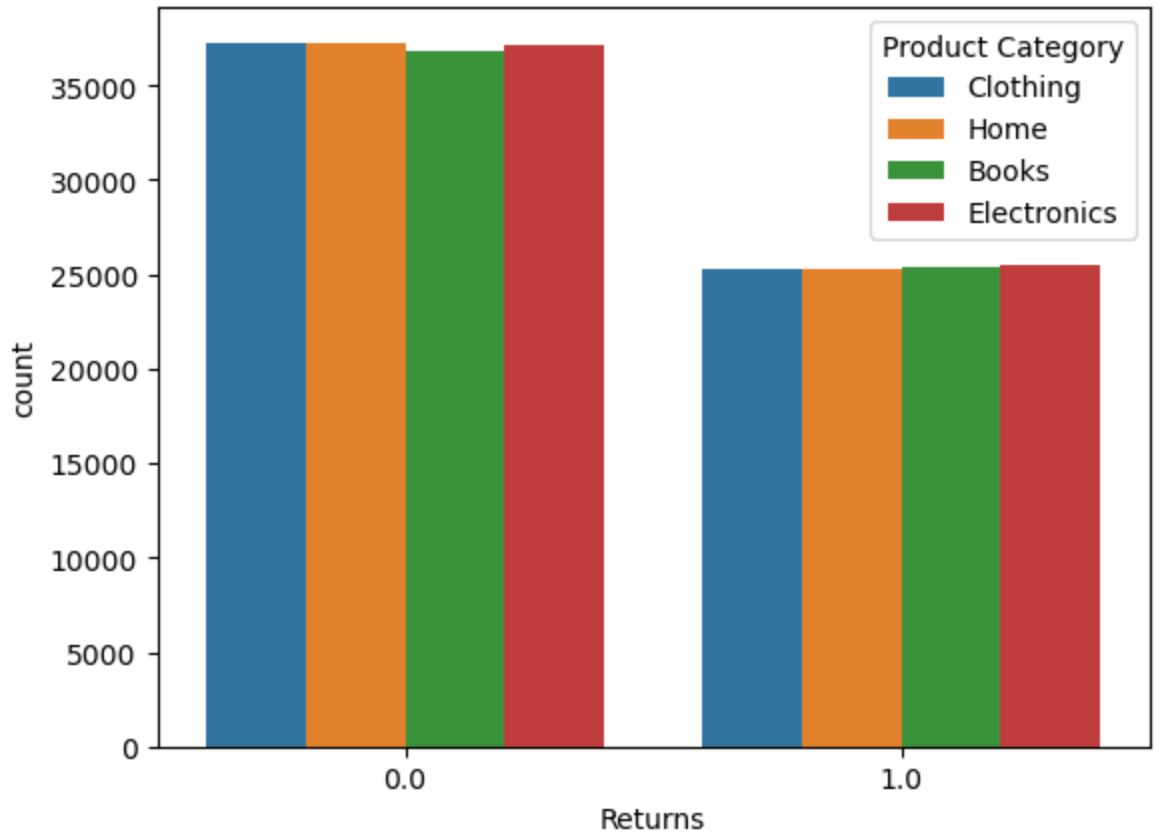
In [52]:
```
sns.displot(new['Returns'])
```

Out[52]:
```
<seaborn.axisgrid.FacetGrid at 0x154a6b01950>
```

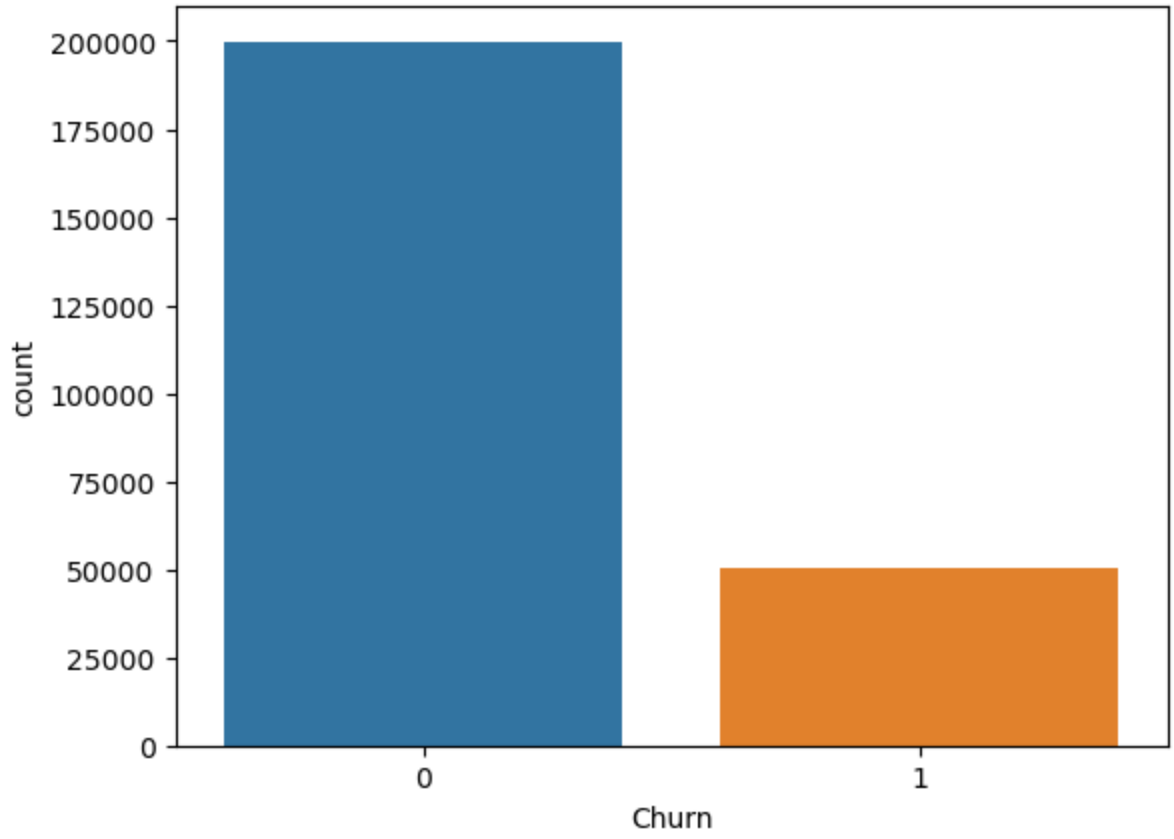


In [53]:
```
sns.countplot(x = 'Returns',hue = 'Product Category', data = new)
```

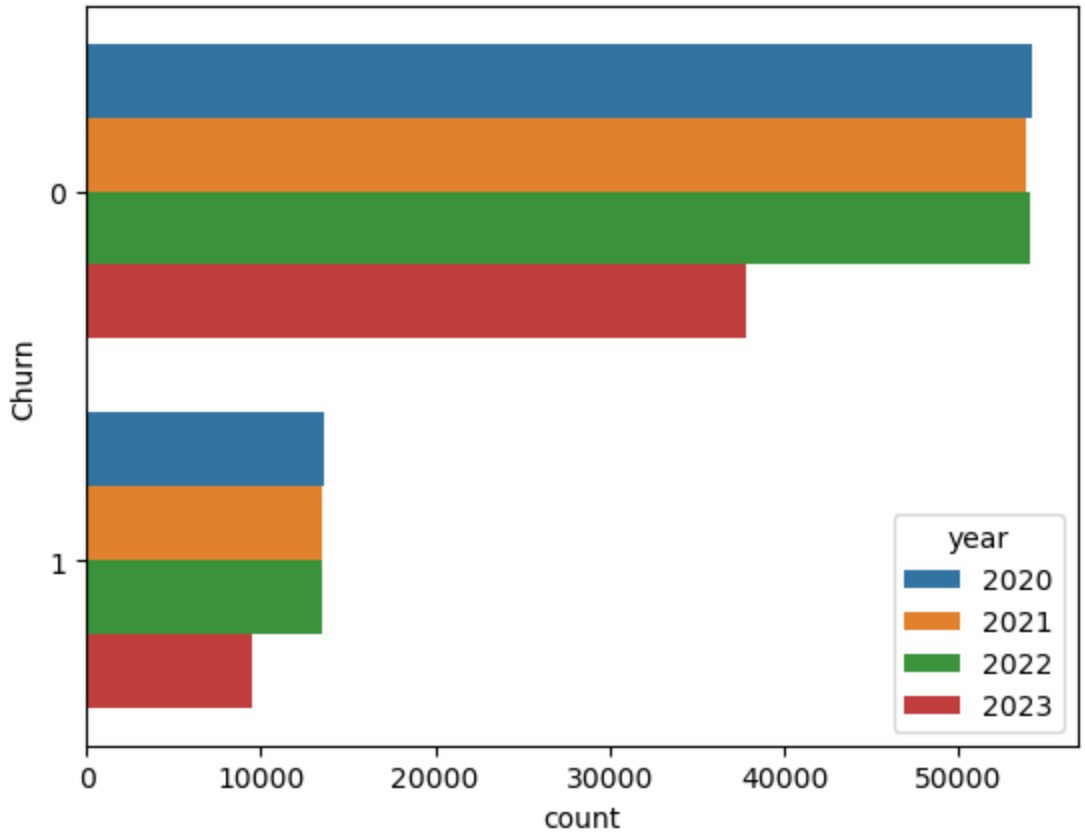
```
plt.show()
```



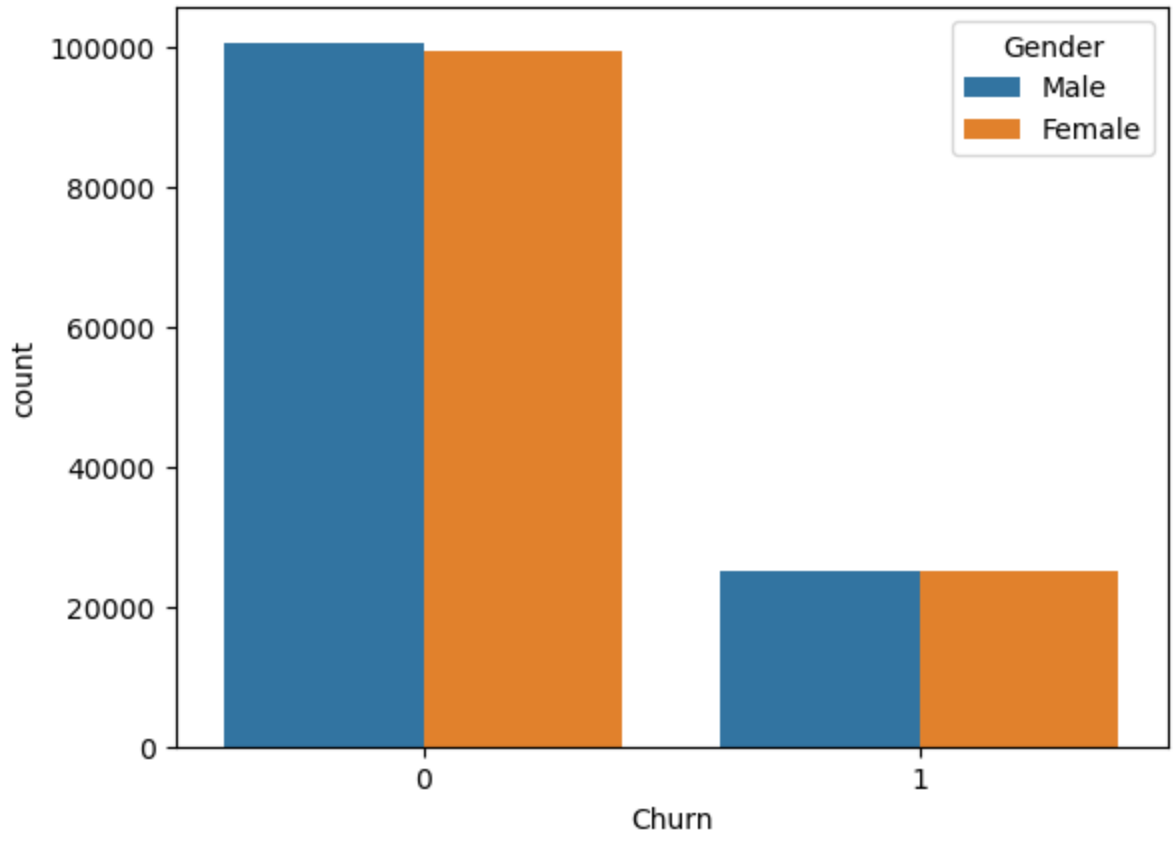
In [54]:
```
sns.countplot(x = 'Churn',data = new)
plt.show()
```



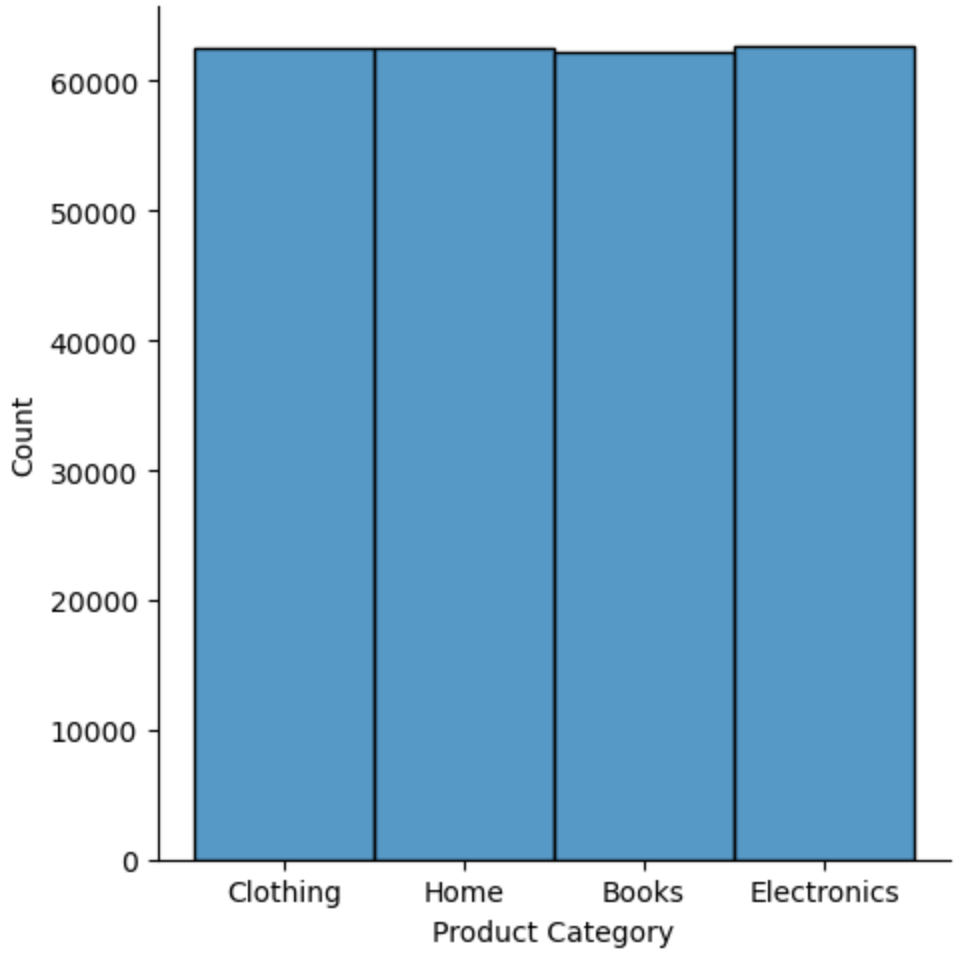
In [55]:
```
sns.countplot(y = 'Churn',hue = 'year',data= new)
plt.show()
```



In [56]:
```
sns.countplot(x = 'Churn',hue = 'Gender',data = new)
plt.show()
```



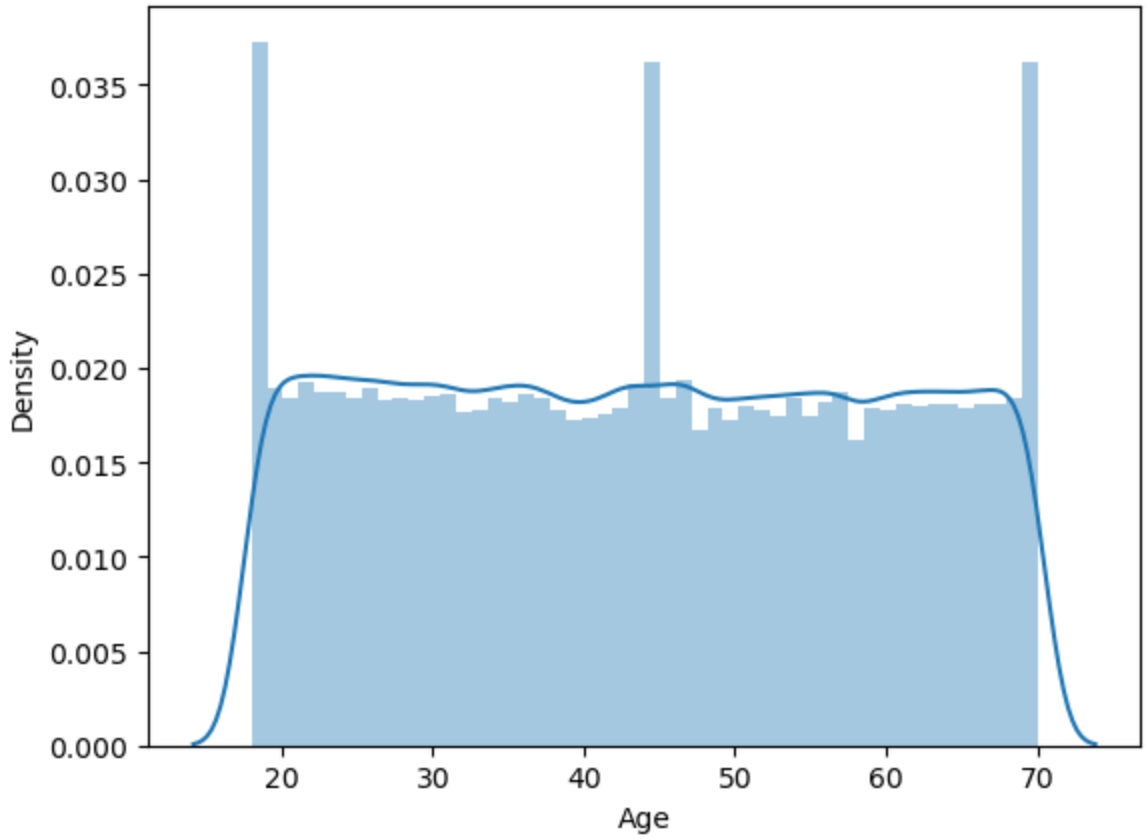
In [57]:
```
sns.displot(new['Product Category'])
plt.show()
```



In [ ]:
```
new
```

In [58]:
```
sns.distplot(new['Age'])
plt.show()
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_2364\964415801.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(new['Age'])
```



In [59]:
```
sns.lineplot(x = 'Age',y = 'Total Amount Purchased',data = new)
plt.show()
```