In [16]:
```
# importing necessary libraries

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import norm, binom
```

In [17]:
```
df = pd.read_csv('Walmart.csv')
```

In [18]:
```
df.head()
```

Out[18]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purch
0	1000001	P00069042	F	0-17	10	A	2	0	3	8
1	1000001	P00248942	F	0-17	10	A	2	0	1	15
2	1000001	P00087842	F	0-17	10	A	2	0	12	1
3	1000001	P00085442	F	0-17	10	A	2	0	12	1
4	1000002	P00285442	M	55+	16	C	4+	0	8	7

In [19]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 550068 entries, 0 to 550067
Data columns (total 10 columns):
 #   Column                      Non-Null Count   Dtype 
---  ------                      --------------   ----- 
 0   User_ID                     550068 non-null  int64 
 1   Product_ID                  550068 non-null  object
 2   Gender                      550068 non-null  object
 3   Age                         550068 non-null  object
 4   Occupation                  550068 non-null  int64 
 5   City_Category               550068 non-null  object
 6   Stay_In_Current_City_Years  550068 non-null  object
 7   Marital_Status              550068 non-null  int64 
 8   Product_Category            550068 non-null  int64 
 9   Purchase                    550068 non-null  int64 
dtypes: int64(5), object(5)
memory usage: 42.0+ MB
```

In [20]:
```
# Checking for total rows and columns in the dataset

df.shape
```

Out[20]:
```
(550068, 10)
```

In [21]:
```
# There are total 550068 rows and 10 columns in the entire dataset.
```

In [22]:
```
# checking for null/missing values
df.isna().sum()
```

Out[22]:
```
User_ID                       0
Product_ID                    0
Gender                        0
Age                           0
Occupation                    0
City_Category                 0
Stay_In_Current_City_Years    0
Marital_Status                0
Product_Category              0
Purchase                      0
dtype: int64
```

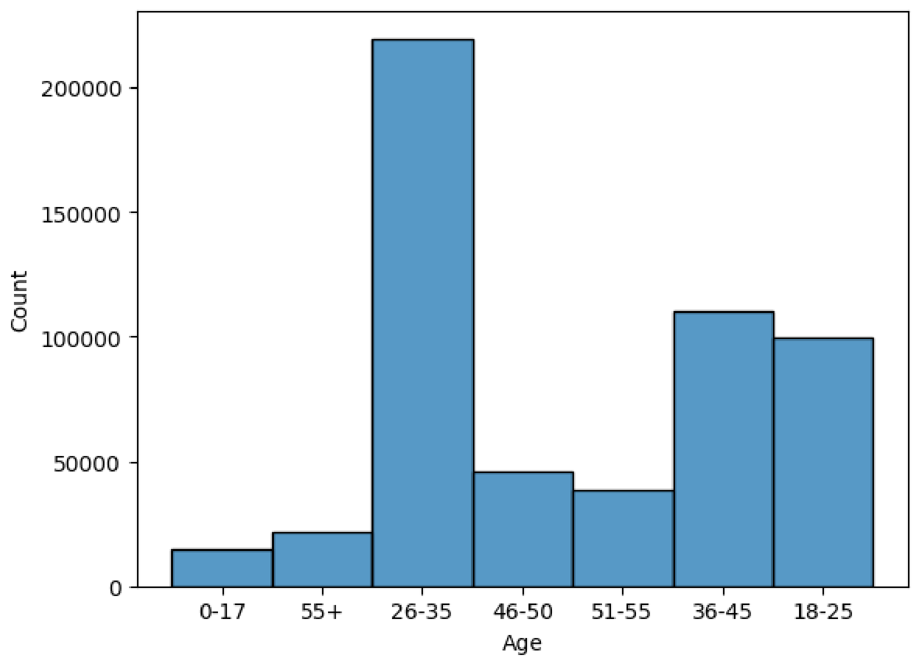
In [23]:
```
df.describe()
```

Out[23]:

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	5.500680e+05	550068.000000	550068.000000	550068.000000	550068.000000
mean	1.003029e+06	8.076707	0.409653	5.404270	9263.968713
std	1.727592e+03	6.522660	0.491770	3.936211	5023.065394
min	1.000001e+06	0.000000	0.000000	1.000000	12.000000
25%	1.001516e+06	2.000000	0.000000	1.000000	5823.000000
50%	1.003077e+06	7.000000	0.000000	5.000000	8047.000000
75%	1.004478e+06	14.000000	1.000000	8.000000	12054.000000
max	1.006040e+06	20.000000	1.000000	20.000000	23961.000000

In [24]:
```
# Average purchase done by a customer is 9263.968713 where the min purchase is 12.00 and max purchase is 239
```

In [25]:
```
sns.histplot(x = 'Age', data = df)
plt.show()
```



In [26]:
```
# total percentage of male and female customers
round(df.Gender.value_counts(normalize = True)*100,2)
```

Out[26]:
```
Gender
M    75.31
F    24.69
Name: proportion, dtype: float64
```

In [27]:
```
# Observation -
# Above analysis show that Male customers purchase more than female customers.
# Around 75.31 % Males customers do purchasing in Walmart compared to Female which
# is 24.69% only.
```

In [28]:
```
# Business Analysis
# Average spending analysis of male and female customers

from scipy.stats import ttest_ind
```

In [29]:
```
male_mean = df[df['Gender'] == 'M']['Purchase'].mean()
female_mean = df[df['Gender'] == 'F']['Purchase'].mean()
```

In [30]:
```
male_mean, female_mean
```

Out[30]:
```
(9437.526040472265, 8734.565765155476)
```

In [31]:
```
# Hypothesis testing to check if the spending is gender biased?

male = df[df['Gender'] == 'M']['Purchase']
female = df[df['Gender'] == 'F']['Purchase']

t_stat, p_value = ttest_ind(male, female, alternative = 'less')
```

In [32]:
```
p_value
```

Out[32]:
```
1.0
```

In [33]:
```
male
```

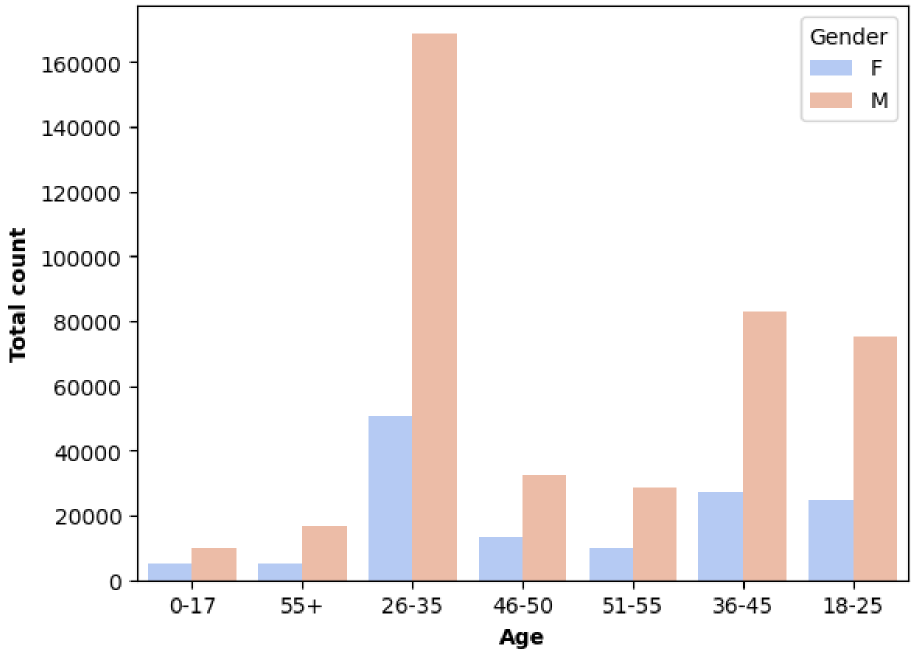
Out[33]:
```
4          7969
5         15227
6         19215
7         15854
8         15686
          ...
550057       61
550058      121
550060      494
550062      473
550063      368
Name: Purchase, Length: 414259, dtype: int64
```

In [34]:
```
df.Product_Category.nunique()
```

Out[34]:
```
20
```

In [35]:
```
# There are 20 unique product categories in this dataset.
```

In [36]:
```
# Gender purchases based on age

sns.countplot(data = df, x = 'Age', hue = 'Gender', palette = 'coolwarm')
plt.xlabel('Age', fontweight = 'bold')
plt.ylabel('Total count', fontweight = 'bold')
plt.show()
```



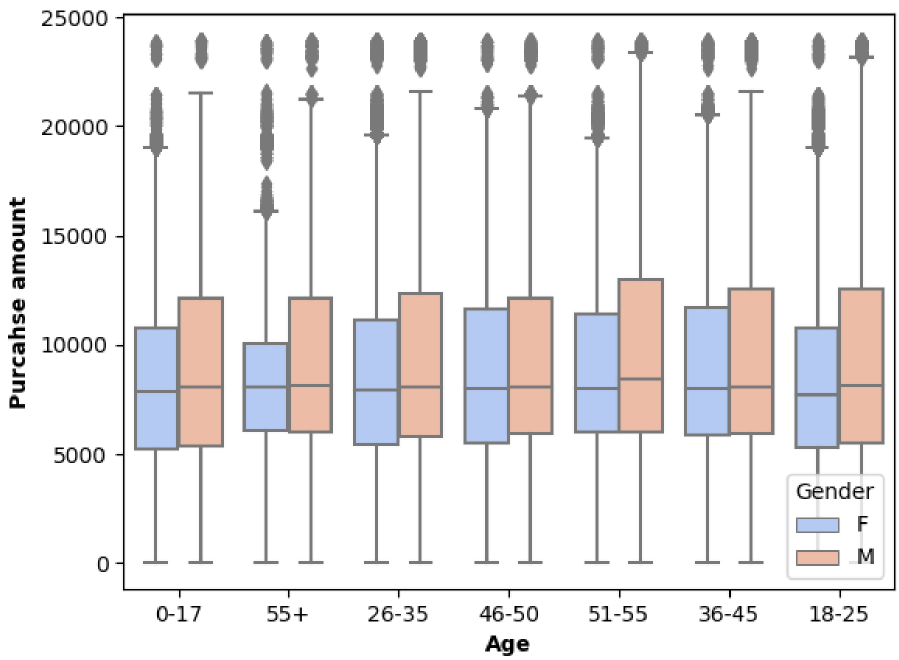
In [37]:
```
# Above analysis show that male customers in the age group of 26-35 do maximum spending compared to other age
# Also, for both the genders, customers in the age-group of 26-35 spend the most.
```

In [38]:
```
# No of customer purchases based on product type
round(df.Product_Category.value_counts(normalize = True)*100,2)
```

Out[38]:
```
Product_Category
5     27.44
1     25.52
8     20.71
11     4.42
2      4.34
6      3.72
3      3.67
4      2.14
16     1.79
15     1.14
13     1.01
10     0.93
12     0.72
7      0.68
18     0.57
20     0.46
19     0.29
14     0.28
17     0.11
9      0.07
Name: proportion, dtype: float64
```

In [39]:
```
# Product Categories 1,5 and 8 are the maximum purchases.
# Thus, least popular product among the customers is Product_Category 9.
```

In [40]:
```
# Outlier detection using boxplot
sns.boxplot(data = df, x = 'Age', y = 'Purchase', hue = 'Gender', palette = 'coolwarm')
plt.xlabel('Age', fontweight = 'bold')
plt.ylabel('Purcahse amount', fontweight = 'bold')
plt.show()
```

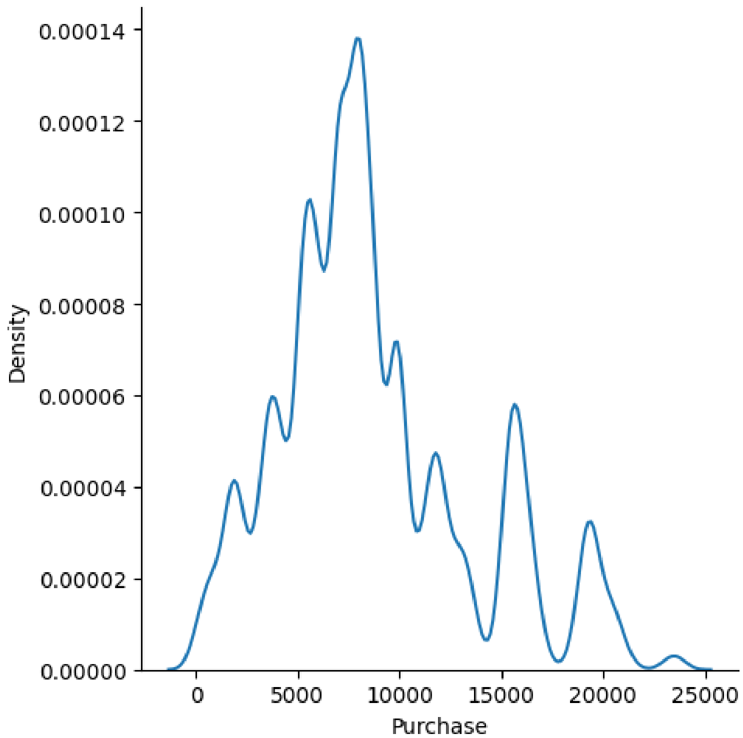


In [41]:
```
# Observations -
# Outliers are seen in both the plots, where maximum outliers are observed for purchases
# of female customers of the age-group 55+ and the age-group 18-45.
```

In [81]:
```
sns.displot(data = female,  kind = 'kde')
```

```
C:\Users\Admin\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:118: UserWarni
ng: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[81]: <seaborn.axisgrid.FacetGrid at 0x29c343d6950>



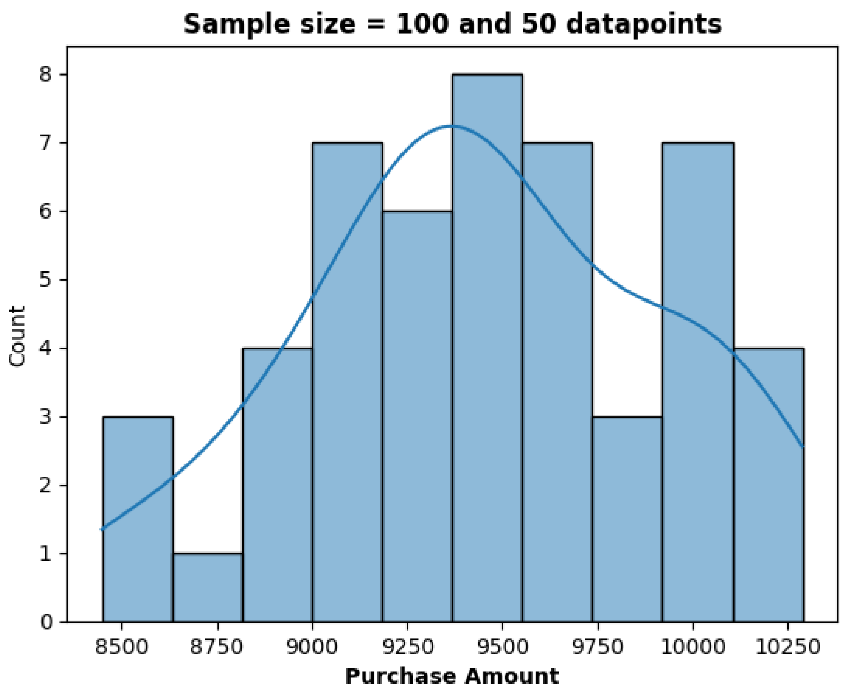
In [82]:
```
male.mean()
```

Out[82]: 9437.526040472265

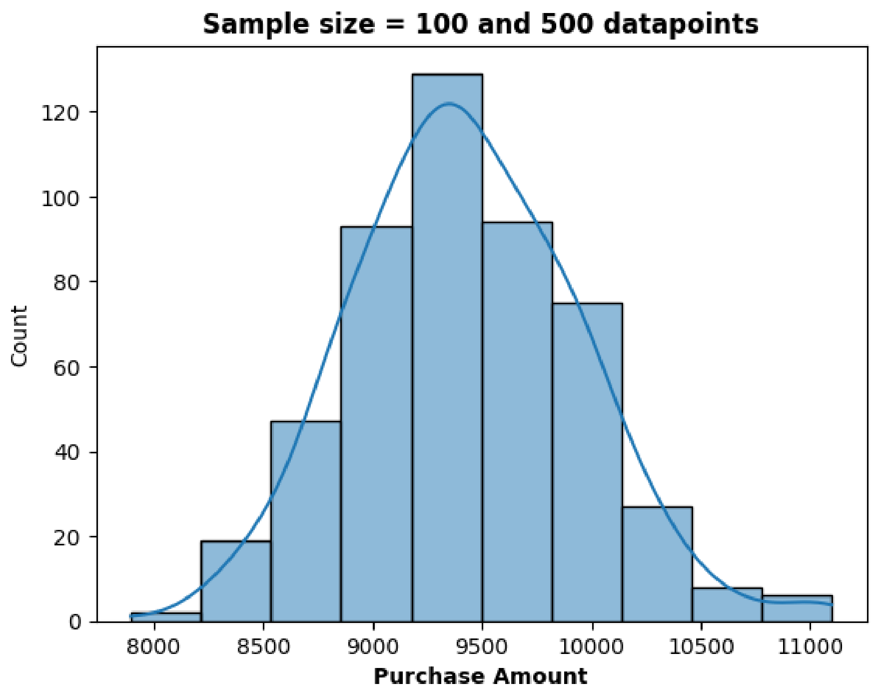
In [83]:
```
np.random.choice(male, size = 10)
```

Out[83]:
```
array([ 7033,  3245,  4200,  1767,  7021,  9983,  3525, 12071,  2811,
       16558], dtype=int64)
```

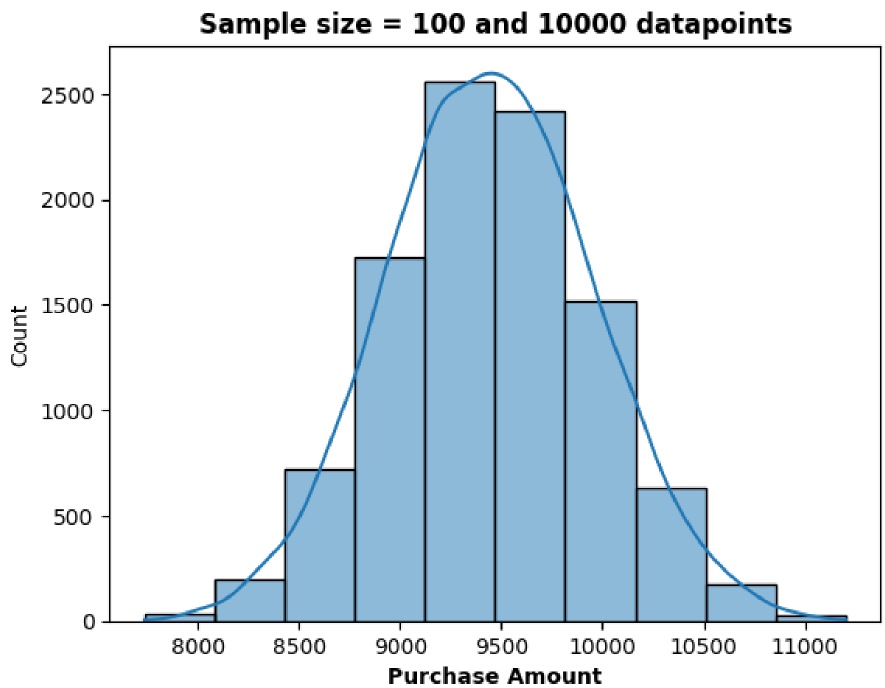
In [84]:
```
# Distribution of the mean spending by MALE customers

# 1. Taking sample size = 100 and checking for 50 datapoints.

bootstrap = []

for i in range(50):
    bootstrap_samples = np.random.choice(male, size = 100)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap.append(bootstrap_mean)

sns.histplot(bootstrap, bins = 10, kde = True)
plt.title('Sample size = 100 and 50 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



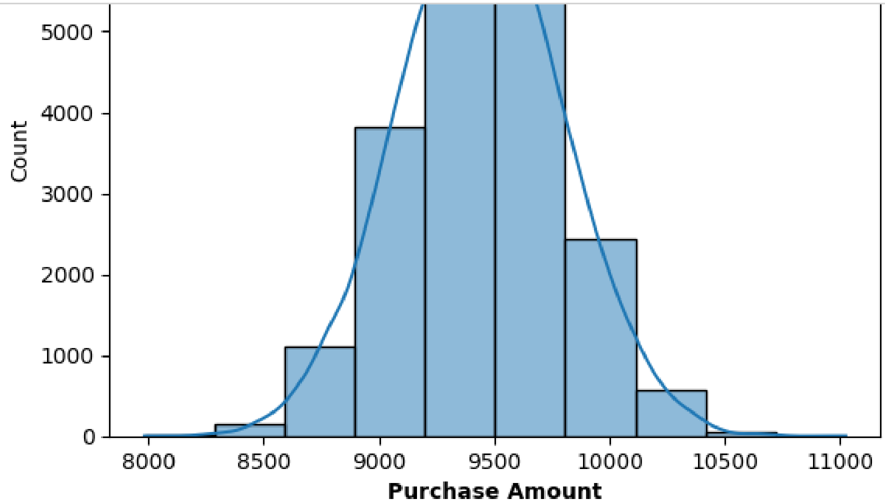
In [85]:
```
# 2. Taking sample size = 100 and 500 datapoints

bootstrap_1 = []

for i in range(500):
    bootstrap_samples = np.random.choice(male, size = 100)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_1.append(bootstrap_mean)

sns.histplot(bootstrap_1, bins = 10, kde = True)
plt.title('Sample size = 100 and 500 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



In [86]:
```
# 3. Taking sample size = 100 and 10,000 datapoints

bootstrap_2 = []

for i in range(10000):
    bootstrap_samples = np.random.choice(male, size = 100)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_2.append(bootstrap_mean)

sns.histplot(bootstrap_2, bins = 10, kde = True)
plt.title('Sample size = 100 and 10000 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



In [52]:
```
# 4. Taking sample size = 200 and 20,000 datapoints

bootstrap_3 = []

for i in range(20000):
    bootstrap_samples = np.random.choice(male, size = 200)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_3.append(bootstrap_mean)

sns.histplot(bootstrap_3, bins = 10, kde = True)
plt.title('Sample size = 200 and 20000 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```

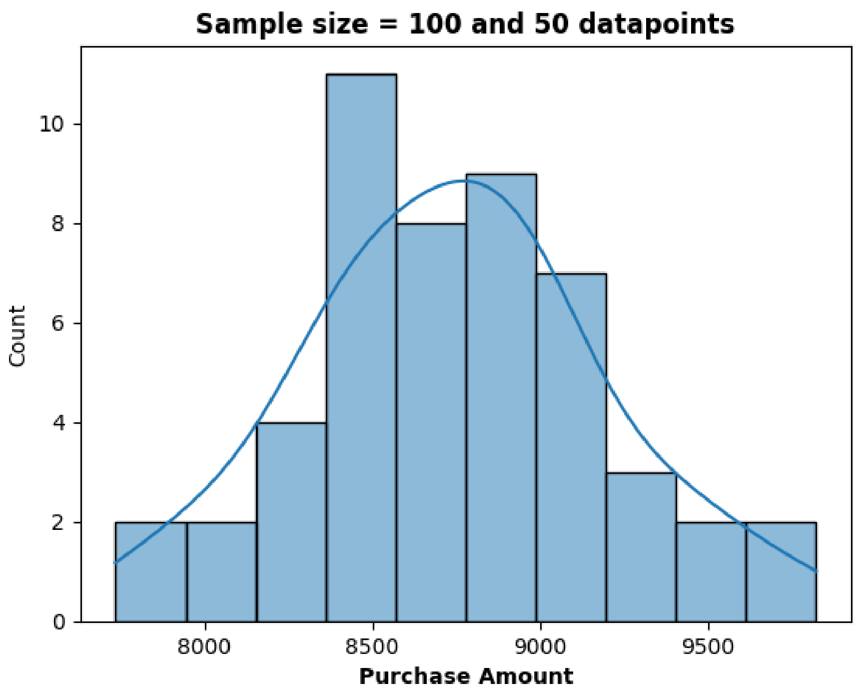


In [88]:
```
# Observations -
#Variance (spread of data) decreases as sample size and datapoints increase.
```

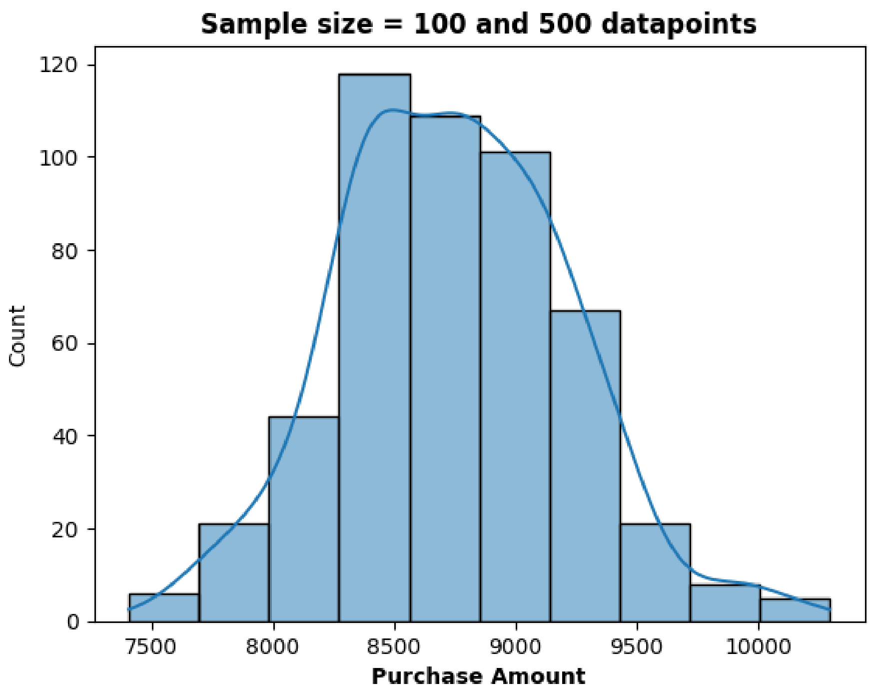
In [89]:
```
male_std = round(male.std(),2)
```

In [90]:
```
female_std = round(female.std(),2)
```

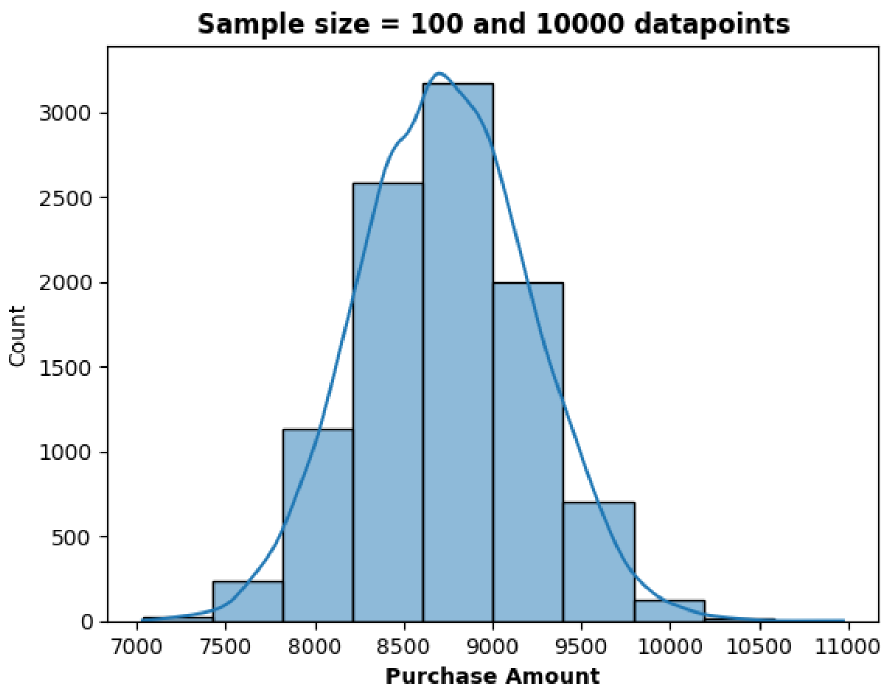
In [92]:
```
# Distribution of the mean spending of FEMALE customers

bootstrap_female = []

for i in range(50):
    bootstrap_samples = np.random.choice(female, size = 100)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_female.append(bootstrap_mean)

sns.histplot(bootstrap_female, bins = 10, kde = True)
plt.title('Sample size = 100 and 50 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



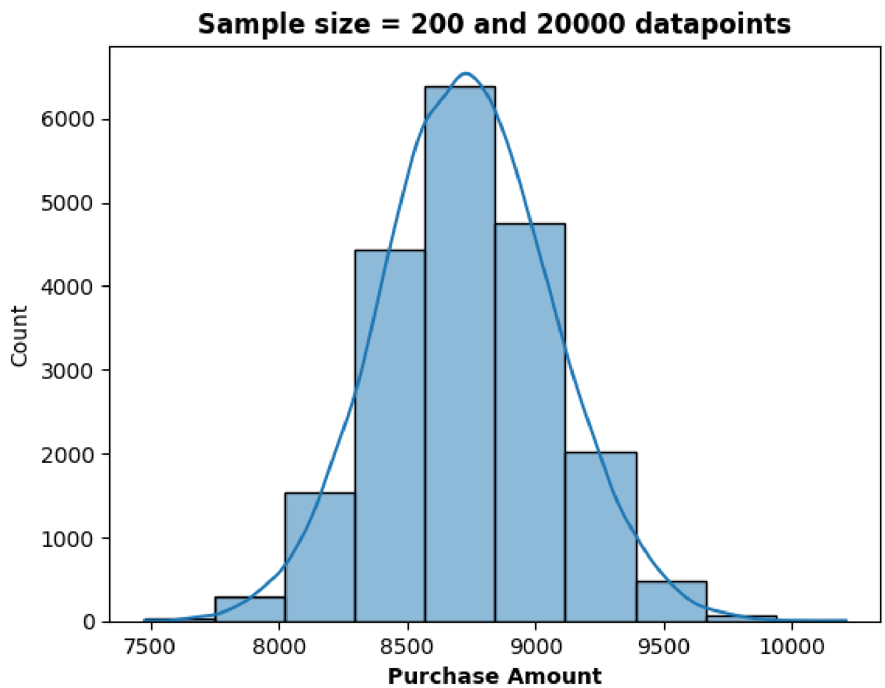
In [93]:
```
bootstrap_1_female = []

for i in range(500):
    bootstrap_samples = np.random.choice(female, size = 100)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_1_female.append(bootstrap_mean)

sns.histplot(bootstrap_1_female, bins = 10, kde = True)
plt.title('Sample size = 100 and 500 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



In [94]:
```
bootstrap_2_female = []

for i in range(10000):
    bootstrap_samples = np.random.choice(female, size = 100)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_2_female.append(bootstrap_mean)

sns.histplot(bootstrap_2_female, bins = 10, kde = True)
plt.title('Sample size = 100 and 10000 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



In [95]:
```
bootstrap_3_female = []

for i in range(20000):
    bootstrap_samples = np.random.choice(female, size = 200)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_3_female.append(bootstrap_mean)

sns.histplot(bootstrap_3_female, bins = 10, kde = True)
plt.title('Sample size = 200 and 20000 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



In [96]:
```
# Observations -
# Comparison between both the distributions infer that the mean spending of Male
# customers is more compared to the female customers.
```

In [97]:
```
# As the variance is less for sample size 200 and 20,000 datapoints, we calculate the
# confidence intervals for 90%, 95% and 99% confidence levels for the specified
# sample size and sample mean.
```

In [98]:
```
# Sample_mean for sample size = 200
sample_mean_male = sum(bootstrap_3)/len(bootstrap_3)
round(sample_mean_male,2)
```

Out[98]: 9439.55

In [99]:
```
sample_mean_female = sum(bootstrap_3_female)/len(bootstrap_3_female)
round(sample_mean_female,2)
```

Out[99]: 8734.31

In [100]:
```
# From the above sample means, we can say that the average amount spent by male
# customers is 9434.45 and the average amount spent by female customers is 8730.19.
```

In [101]:
```
# Validating the difference in the mean spending with different confidence interval-

# For Male customers

# 90% Confidence Level
x1 = np.percentile(bootstrap_3, 5)
x2 = np.percentile(bootstrap_3, 95)
```

In [102]:
```
x1, x2
```

Out[102]:
```
(8849.384, 10033.428)
```

In [103]:
```
# With 90% Confidence Interval, the mean spending of male customers lie in the range (8848.144, 10024.076).
```

In [104]:
```
# 95% Confidence Level
x1 = np.percentile(bootstrap_3, 2.5)
x2 = np.percentile(bootstrap_3, 97.5)
```

In [105]:
```
x1, x2
```

Out[105]:
```
(8742.31725, 10143.500999999998)
```

In [106]:
```
# With 95% Confidence Interval, the mean spending of male customers lie in the range (8741.456, 10133.311).
```

In [107]:
```
# 99% Confidence Level
x1 = np.percentile(bootstrap_3, 0.5)
x2 = np.percentile(bootstrap_3, 99.5)
```

In [108]:
```
x1, x2
```

Out[108]:
```
(8532.731800000001, 10353.893075)
```

In [109]:
```
# With 99% Confidence Interval, the mean spending of male customers lie in the range  (8513.545, 10380.782).
```

In [110]:
```
# For Female customers

# 90% Confidence Level
y1 = np.percentile(bootstrap_3_female, 5)
y2 = np.percentile(bootstrap_3_female, 95)
```

In [111]:
```
y1,y2
```

Out[111]:
```
(8192.4145, 9292.535249999999)
```

In [112]:
```
# With 90% Confidence Interval, the mean spending of female customers lie in the range (8174.314, 9292.295).
```

In [113]:
```
# 95% Confidence Level
y1 = np.percentile(bootstrap_3_female, 2.5)
y2 = np.percentile(bootstrap_3_female, 97.5)
```

In [114]:
```
y1,y2
```

Out[114]:
```
(8084.8575, 9404.265374999999)
```

In [115]:
```
# With 95% Confidence Interval, the mean spending of female customers lie in the range  (7867.459, 9603.082)
```

In [116]:
```
# 99% Confidence Level
y1 = np.percentile(bootstrap_3_female, 0.5)
y2 = np.percentile(bootstrap_3_female, 99.5)
```

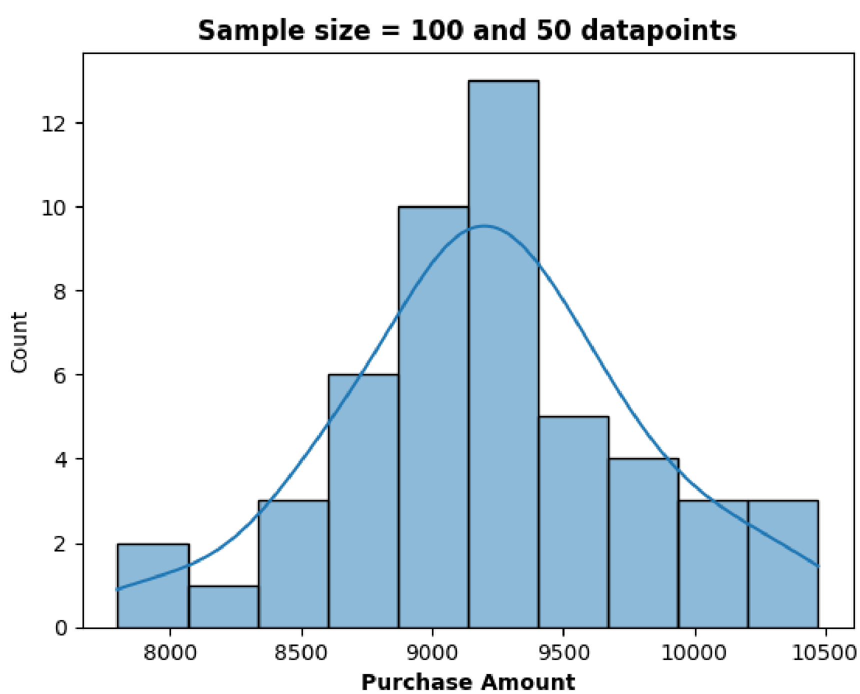
In [117]:
```
y1,y2
```

Out[117]:
```
(7885.33705, 9610.6409)
```

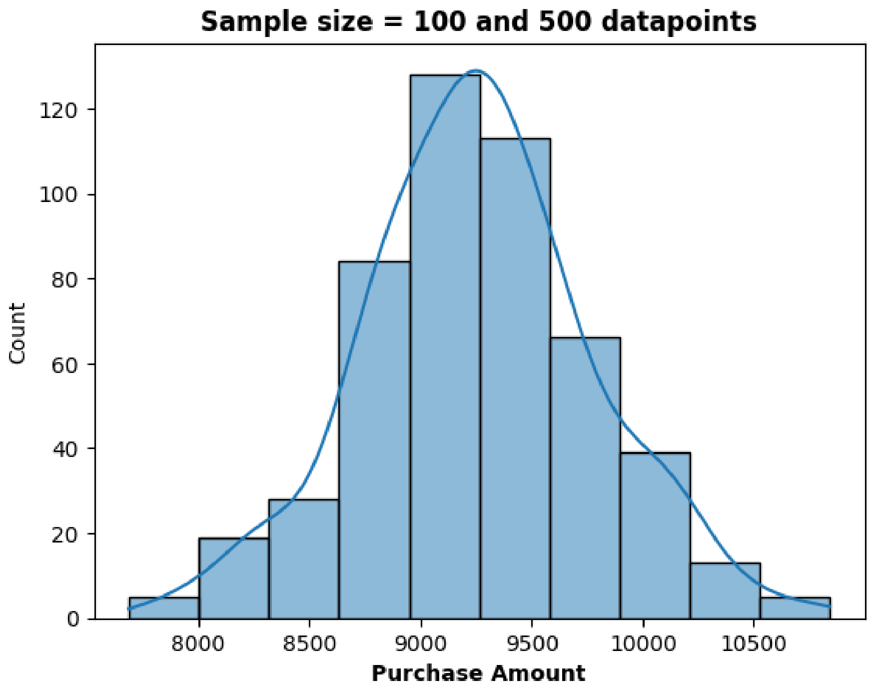
In [118]:
```
# With 99% Confidence Interval, the mean spending of female customers lie in the range (7867.4595, 9603.917)
```

In [119]:
```
# Observations -
# Confidence intervals are found to be overlapping. This concludes that there is no
# significant difference in the mean spending by male and female customers.
```

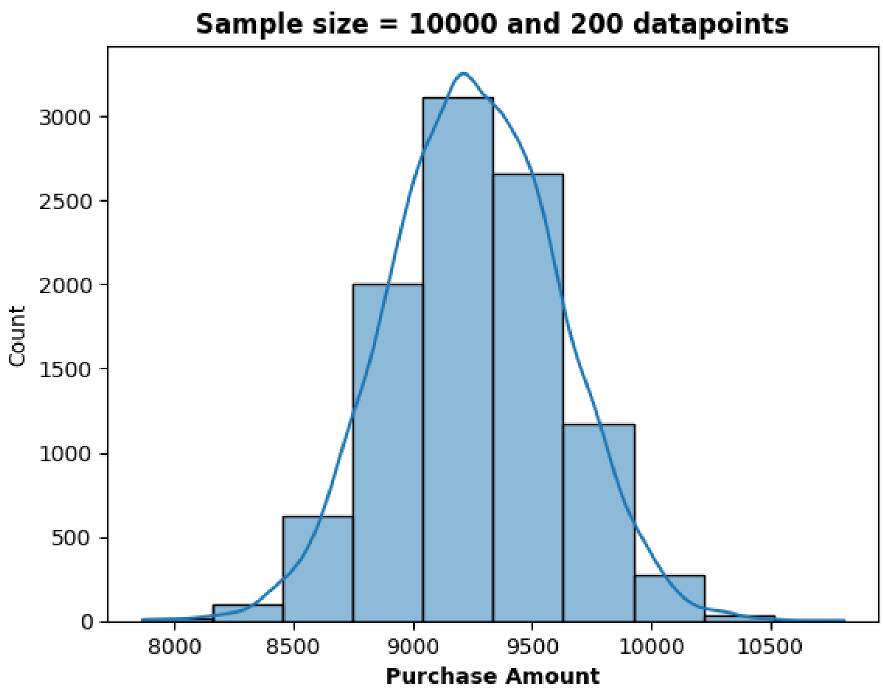
In [121]:
```
# Distribution of mean spending based on Marital Status

# 1 - Married
# 0 - Unmarried


um = df[df["Marital_Status"] == 0]['Purchase']
m = df[df["Marital_Status"] ==1]['Purchase']

# MARRIED CUSTOMERS

bootstrap_1_m = []
for i in range(50):
    bootstrap_samples = np.random.choice(m, size = 100)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_1_m.append(bootstrap_mean)

sns.histplot(bootstrap_1_m, bins = 10, kde = True)
plt.title('Sample size = 100 and 50 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



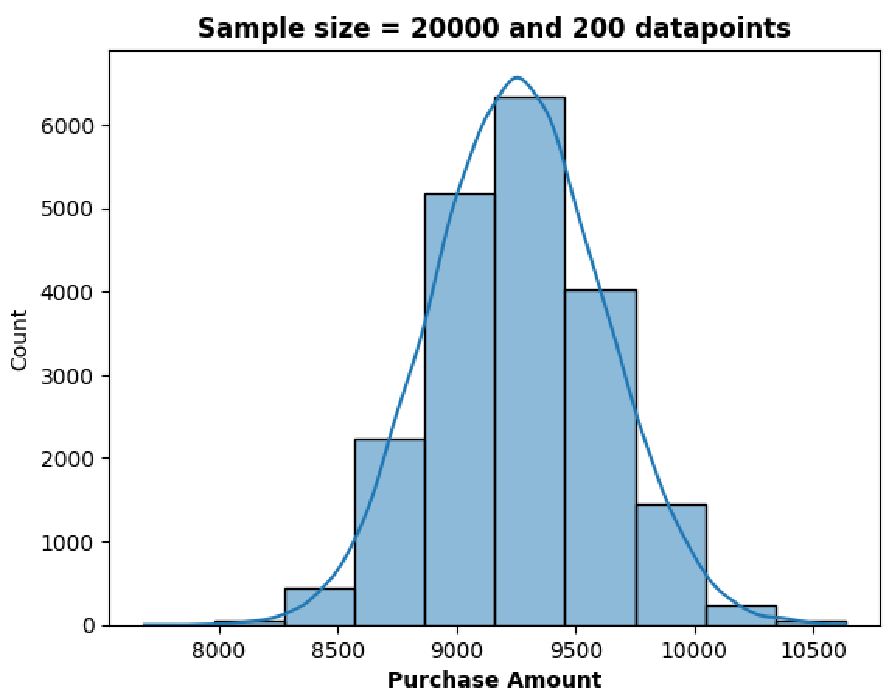
In [122]:
```
bootstrap_2_m = []
for i in range(500):
    bootstrap_samples = np.random.choice(m, size = 100)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_2_m.append(bootstrap_mean)

sns.histplot(bootstrap_2_m, bins = 10, kde = True)
plt.title('Sample size = 100 and 500 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



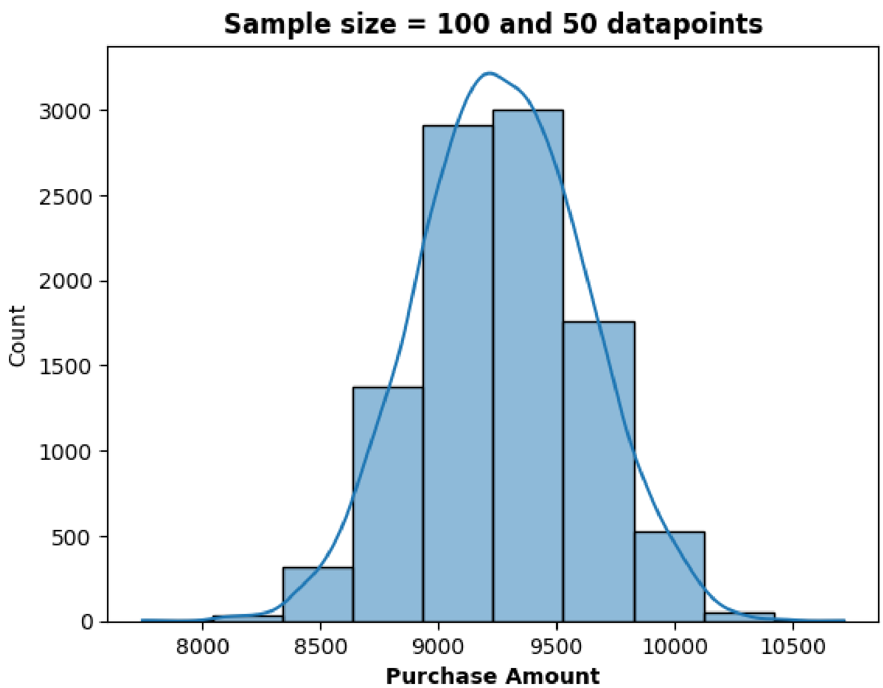
In [123]:
```
bootstrap_3_m = []

for i in range(10000):
    bootstrap_samples = np.random.choice(m, size = 200)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_3_m.append(bootstrap_mean)

sns.histplot(bootstrap_3_m, bins = 10, kde = True)
plt.title('Sample size = 10000 and 200 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



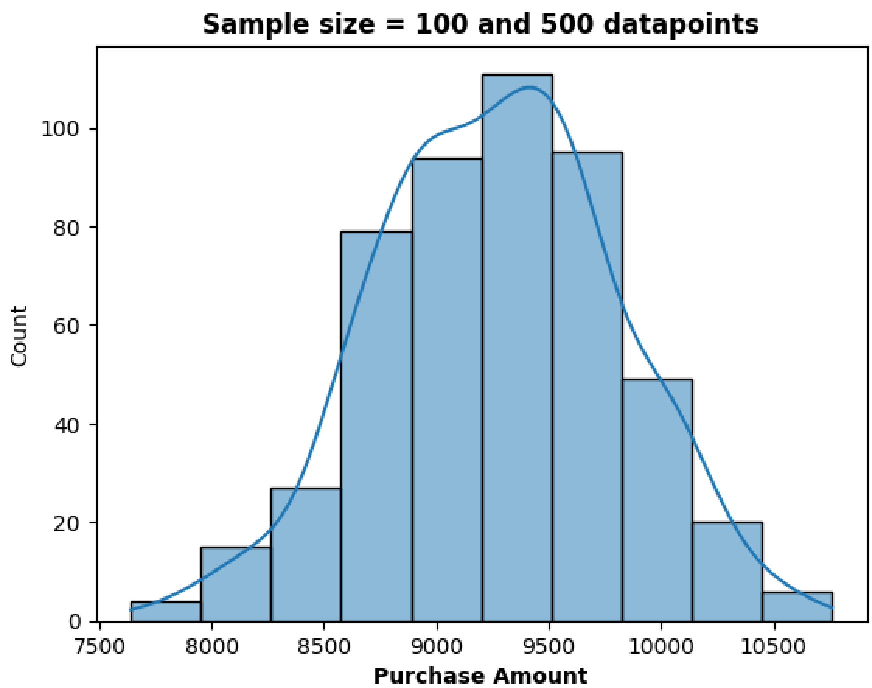
In [124]:
```
bootstrap_4_m = []

for i in range(20000):
    bootstrap_samples = np.random.choice(m, size = 200)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_4_m.append(bootstrap_mean)

sns.histplot(bootstrap_4_m, bins = 10, kde = True)
plt.title('Sample size = 20000 and 200 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



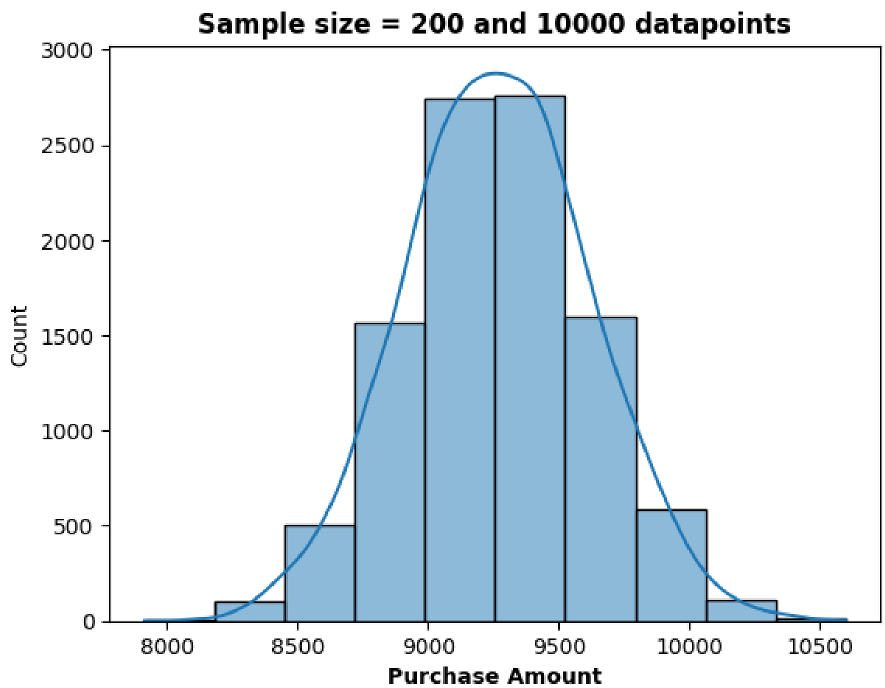
In [125]:
```
# UNMARRIED CUSTOMERS

bootstrap_1_um = []
um = df[df["Marital_Status"] == 0]['Purchase']
m = df[df["Marital_Status"] ==1]['Purchase']

for i in range(10000):
    bootstrap_samples = np.random.choice(um, size = 200)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_1_um.append(bootstrap_mean)

sns.histplot(bootstrap_1_um, bins = 10, kde = True)
plt.title('Sample size = 100 and 50 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



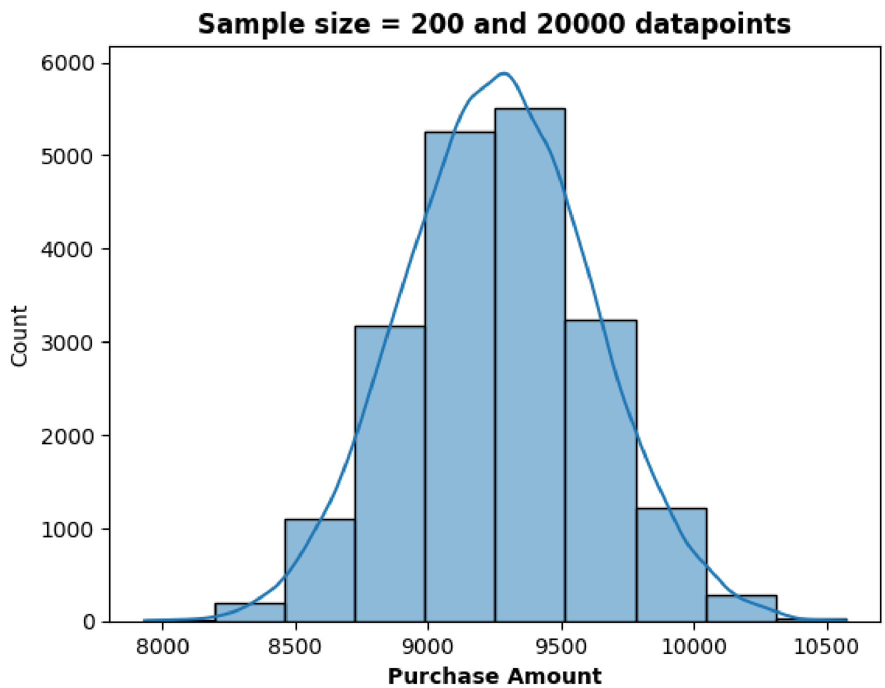
In [126]:
```
bootstrap_2_um = []

for i in range(500):
    bootstrap_samples = np.random.choice(um, size = 100)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_2_um.append(bootstrap_mean)

sns.histplot(bootstrap_2_um, bins = 10, kde = True)
plt.title('Sample size = 100 and 500 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



In [127]:
```
bootstrap_3_um = []

for i in range(10000):
    bootstrap_samples = np.random.choice(um, size = 200)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_3_um.append(bootstrap_mean)

sns.histplot(bootstrap_3_um, bins = 10, kde = True)
plt.title('Sample size = 200 and 10000 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



In [128]:
```
bootstrap_4_um = []

for i in range(20000):
    bootstrap_samples = np.random.choice(um, size = 200)
    bootstrap_mean = np.mean(bootstrap_samples)
    bootstrap_4_um.append(bootstrap_mean)

sns.histplot(bootstrap_4_um, bins = 10, kde = True)
plt.title('Sample size = 200 and 20000 datapoints', fontweight = 'bold')
plt.xlabel('Purchase Amount', fontweight = 'bold')
plt.show()
```



In [129]:
```
def confidence(ms = marital_status(), r, n):

    if ms == m:
        bootstrap_m = []
        for i in range(r):
            bootstrap_samples = np.random.choice(ms, size = n)
            bootstrap_mean = np.mean(bootstrap_samples)
            bootstrap_um.append(bootstrap_mean)

        sns.histplot(bootstrap_m, bins = 10, kde = True)
        plt.title('Sample size = 100 and 50 datapoints', fontweight = 'bold')
        plt.xlabel('Purchase Amount', fontweight = 'bold')
        plt.show()
    else:
        bootstrap_um = []
        for i in range(r):
            bootstrap_samples = np.random.choice(ms, size = n)
            bootstrap_mean = np.mean(bootstrap_samples)
            bootstrap_um.append(bootstrap_mean)

        sns.histplot(bootstrap_um, bins = 10, kde = True)
        plt.title('Sample size = 100 and 50 datapoints', fontweight = 'bold')
        plt.xlabel('Purchase Amount', fontweight = 'bold')
        plt.show()
```

```
  Cell In[129], line 1
    def confidence(ms = marital_status(), r, n):
                                          ^
SyntaxError: non-default argument follows default argument
```

In [ ]:
```
def marital_status(input()):
    if m == m:
        m = df[df["Marital_Status"] ==1]['Purchase']
        return m
    else:
        um = df[df["Marital_Status"] == 0]['Purchase']
        return um
```

In [ ]:
```
confidence(m,500,100)
```

In [ ]:
```
# Validating the difference in the mean spending based on Marital Status

sample_mean_married = sum(bootstrap_4_m)/len(bootstrap_4_m)
round(sample_mean_married,2)
```

In [ ]:
```
sample_mean_unmarried = sum(bootstrap_4_um)/len(bootstrap_4_um)
round(sample_mean_unmarried,2)
```

In [206]:
```
# 90% Confidence Level
m1 = np.percentile(bootstrap_4_m, 5)
m2 = np.percentile(bootstrap_4_m, 95)
```

In [207]:
```
m1,m2
```

Out[207]:
```
(8683.029, 9853.856)
```

In [208]:
```
# 95% Confidence Level
m1 = np.percentile(bootstrap_4_m, 2.5)
m2 = np.percentile(bootstrap_4_m, 97.5)
```

In [209]:
```
m1,m2
```

Out[209]:
```
(8585.49225, 9964.86175)
```

In [211]:
```
# 99% Confidence Level
m1 = np.percentile(bootstrap_4_m, 0.5)
m2 = np.percentile(bootstrap_4_m, 99.5)
```

In [212]:
```
m1,m2
```

Out[212]:
```
(8366.3583, 10184.240425)
```

In [213]:
```
# 90% Confidence Level
um1 = np.percentile(bootstrap_4_um, 5)
um2 = np.percentile(bootstrap_4_um, 95)
```

In [214]:
```
um1,um2
```

Out[214]:
```
(8683.460500000001, 9857.69975)
```

In [215]:
```
# 95% Confidence Level
um1 = np.percentile(bootstrap_4_um, 2.5)
um2 = np.percentile(bootstrap_4_um, 97.5)
```

In [216]:
```
um1,um2
```

Out[216]:
```
(8571.387625, 9973.452625)
```

In [217]:
```
# 99% Confidence Level
um1 = np.percentile(bootstrap_4_um, 0.5)
um2 = np.percentile(bootstrap_4_um, 99.5)
```

In [218]:
```
um1,um2
```

Out[218]:
```
(8373.65, 10174.08725)
```

In [ ]:
```
# Observations -
# Confidence intervals are found to be overlapping. This concludes that there is no
# significant difference in the mean spending by married and unmarried customers.
```

In [ ]:
```
# Recommendations
# 1. Men spent more money than women, so company should focus on retaining the male customers and getting mor
# 2. Product_Category - 1, 5 & 8 have highest purchasing frequency. It means these are the products in these
# focus on selling more of these products or selling more of the products which
# are purchased less.
# 3. Customers in the age 18-45 spend more money than the others, so company should focus on acquisition of c
# 4. Male customers living in City_Category C spend more money than other male customers living in B or C, Se
# help the company increase the revenue
```

In [221]:
```
categorical_cols = ['Gender', 'Age', 'Occupation', 'City_Category', 'Stay_In_Current_City_Years', 'Marital_St
df[categorical_cols].melt().groupby(['variable', 'value'])[['value']].count()/len(df)
```

Out[221]:

variable	value	value
Age	0-17	0.027455
	18-25	0.181178
	26-35	0.399200
	36-45	0.199999
	46-50	0.083082
	51-55	0.069993
	55+	0.039093
City_Category	A	0.268549
	B	0.420263
	C	0.311189
Gender	F	0.246895
	M	0.753105
Marital_Status	0	0.590347
	1	0.409653
Occupation	0	0.126599
	1	0.086218
	2	0.048336
	3	0.032087
	4	0.131453
	5	0.022137
	6	0.037005
	7	0.107501
	8	0.002811
	9	0.011437
	10	0.023506
	11	0.021063
	12	0.056682
	13	0.014049
	14	0.049647
	15	0.022115
	16	0.046123
	17	0.072796
	18	0.012039
	19	0.015382
	20	0.061014

		value
variable	**value**	
Product_Category	**1**	0.255201
	2	0.043384
	3	0.036746
	4	0.021366
	5	0.274390
	6	0.037206
	7	0.006765
	8	0.207111
	9	0.000745
	10	0.009317
	11	0.044153
	12	0.007175
	13	0.010088
	14	0.002769
	15	0.011435
	16	0.017867
	17	0.001051
	18	0.005681
	19	0.002914
	20	0.004636
Stay_In_Current_City_Years	**0**	0.135252
	1	0.352358
	2	0.185137
	3	0.173224
	4+	0.154028

In [222]:
```
amt_df = df.groupby(['User_ID', 'Gender'])[['Purchase']].sum()
amt_df = amt_df.reset_index()
amt_df
```

Out[222]:

	User_ID	Gender	Purchase
0	1000001	F	334093
1	1000002	M	810472
2	1000003	M	341635
3	1000004	M	206468
4	1000005	M	821001
...	...	...	...
5886	1006036	F	4116058
5887	1006037	F	1119538
5888	1006038	F	90034
5889	1006039	F	590319
5890	1006040	M	1653299

5891 rows × 3 columns

In [223]:
```
df.head()
```

Out[223]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purch
0	1000001	P00069042	F	0-17	10	A	2	0	3	8
1	1000001	P00248942	F	0-17	10	A	2	0	1	15
2	1000001	P00087842	F	0-17	10	A	2	0	12	1
3	1000001	P00085442	F	0-17	10	A	2	0	12	1
4	1000002	P00285442	M	55+	16	C	4+	0	8	7

In [224]:
```
df.User_ID.nunique()
```

Out[224]: 5891

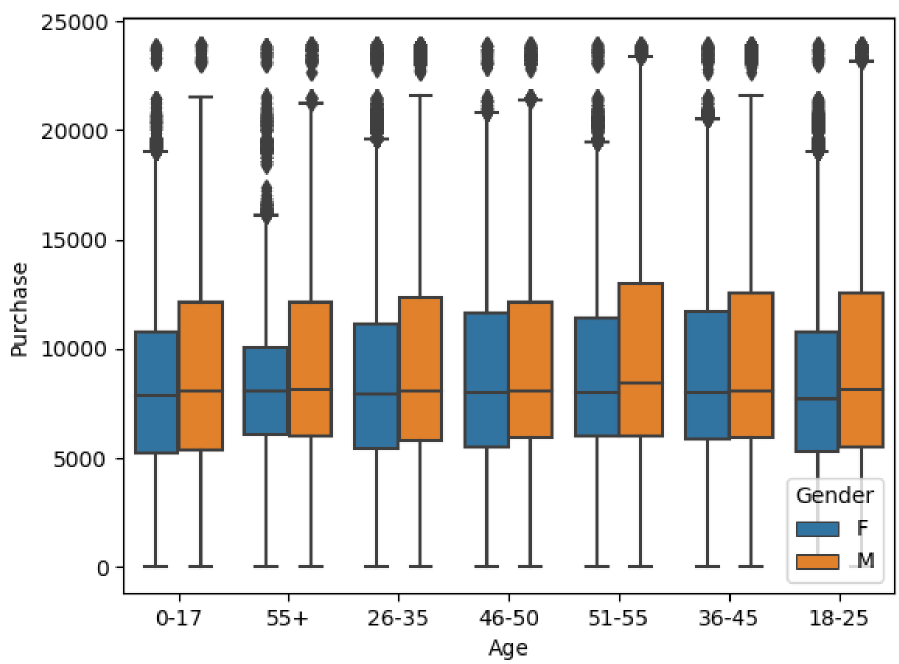
In [225]:
```
categorical_cols = ['Gender', 'Age', 'Occupation', 'City_Category', 'Stay_In_Current_City_Years',
                    'Marital_Status', 'Product_Category']
df[categorical_cols].melt().groupby(['variable', 'value'])[['value']].count()/len(df)
```

Out[225]:

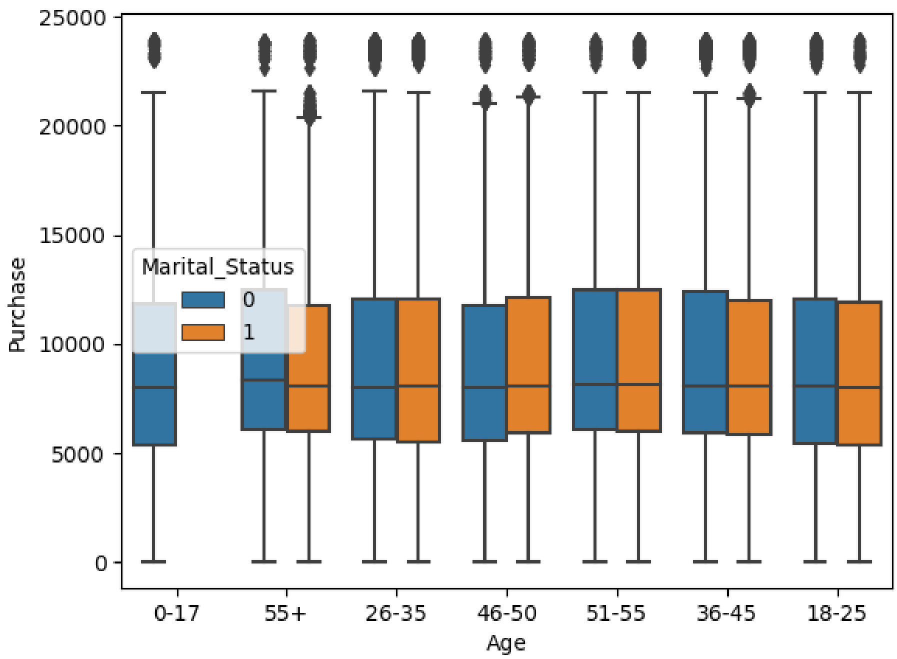
variable	value	value
Age	0-17	0.027455
	18-25	0.181178
	26-35	0.399200
	36-45	0.199999
	46-50	0.083082
	51-55	0.069993
	55+	0.039093
City_Category	A	0.268549
	B	0.420263
	C	0.311189
Gender	F	0.246895
	M	0.753105
Marital_Status	0	0.590347
	1	0.409653
Occupation	0	0.126599
	1	0.086218
	2	0.048336
	3	0.032087
	4	0.131453
	5	0.022137
	6	0.037005
	7	0.107501
	8	0.002811
	9	0.011437
	10	0.023506
	11	0.021063
	12	0.056682
	13	0.014049
	14	0.049647
	15	0.022115
	16	0.046123
	17	0.072796
	18	0.012039
	19	0.015382
	20	0.061014

		value
variable	**value**	
Product_Category	**1**	0.255201
	2	0.043384
	3	0.036746
	4	0.021366
	5	0.274390
	6	0.037206
	7	0.006765
	8	0.207111
	9	0.000745
	10	0.009317
	11	0.044153
	12	0.007175
	13	0.010088
	14	0.002769
	15	0.011435
	16	0.017867
	17	0.001051
	18	0.005681
	19	0.002914
	20	0.004636
Stay_In_Current_City_Years	**0**	0.135252
	1	0.352358
	2	0.185137
	3	0.173224
	4+	0.154028

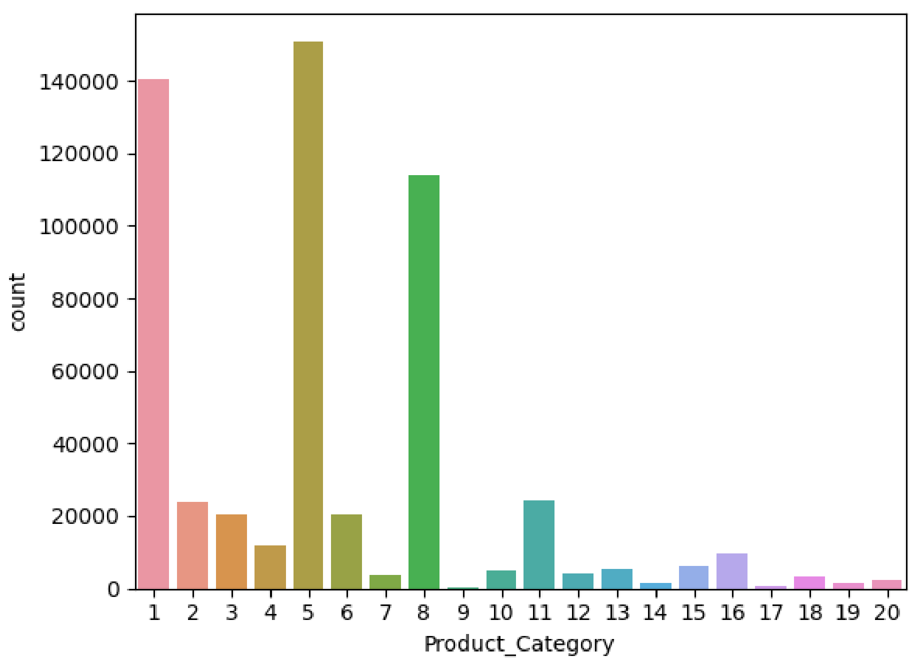
In [230]:
```
#outlier detection
sns.boxplot(data = df, y = 'Purchase', x = 'Age', hue = 'Gender' )
plt.show()
```



In [232]:
```
sns.boxplot(data = df, y = 'Purchase', x = 'Age', hue = 'Marital_Status' )
plt.show()
```



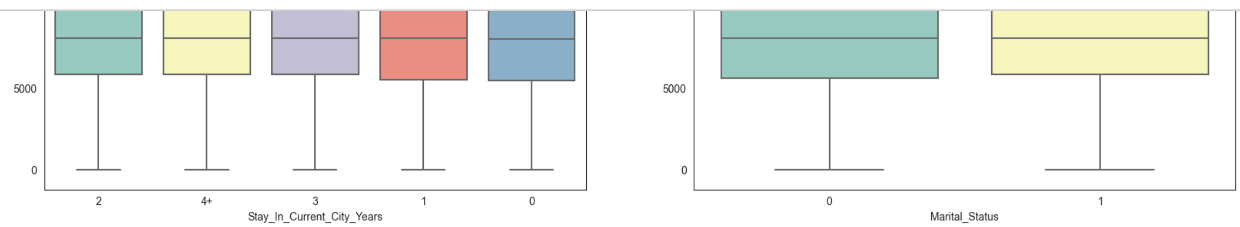
In [236]:
```
sns.countplot(data = df, x = 'Product_Category')
plt.show()
```

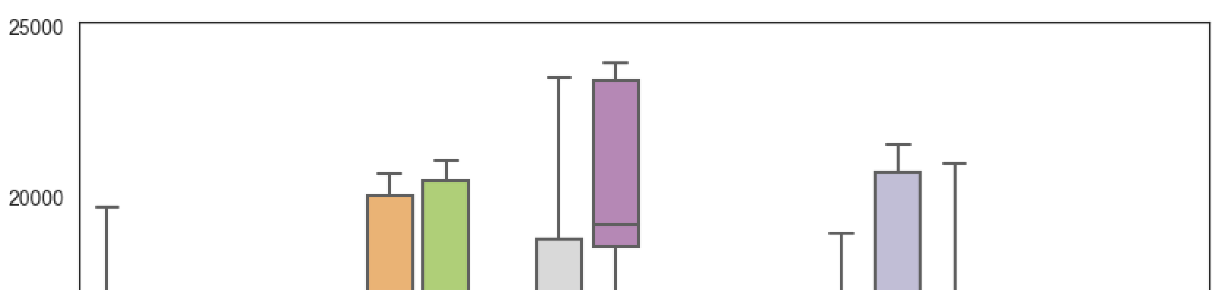


In [246]:
```
round(df.Product_Category.value_counts(normalize = True)*100,2)
```

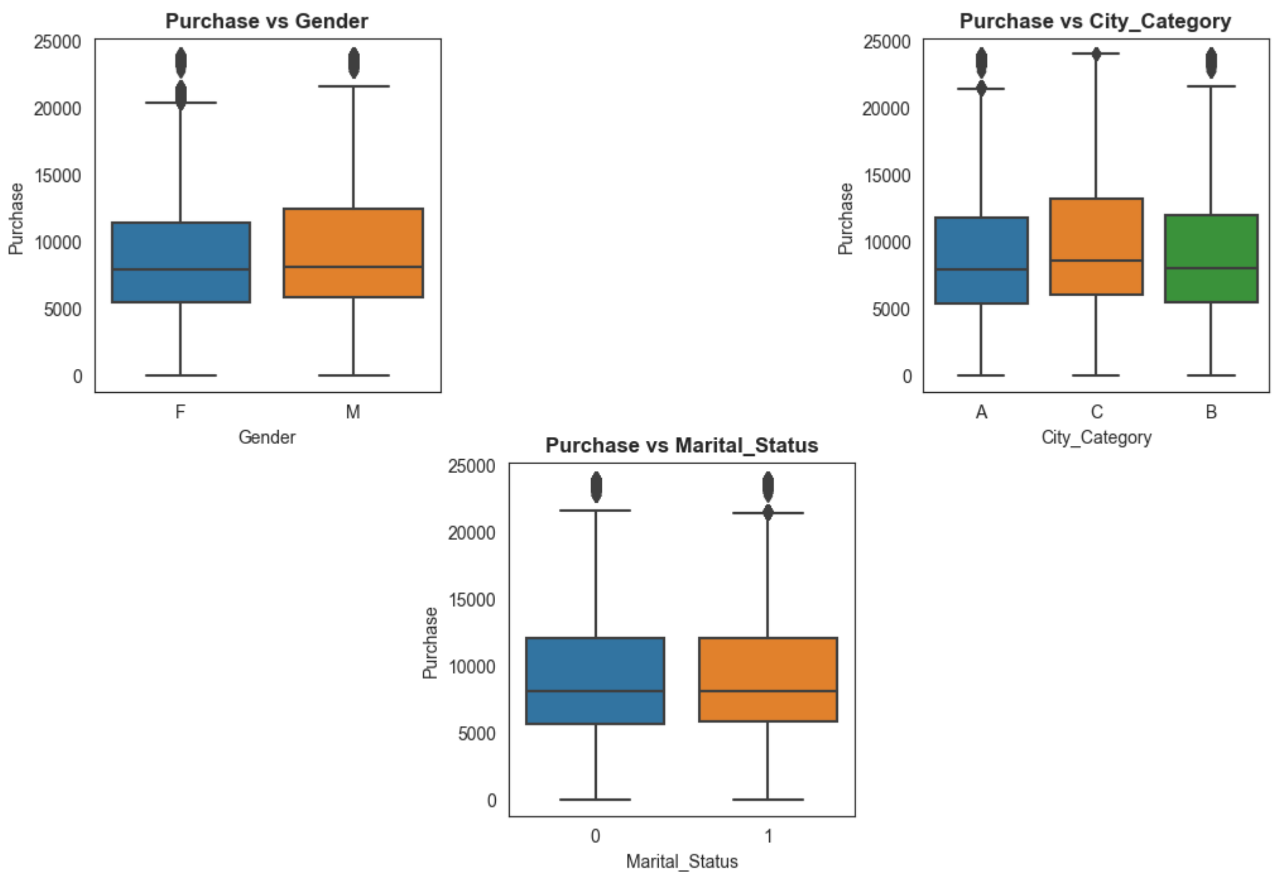
Out[246]:
```
Product_Category
5     27.44
1     25.52
8     20.71
11     4.42
2      4.34
6      3.72
3      3.67
4      2.14
16     1.79
15     1.14
13     1.01
10     0.93
12     0.72
7      0.68
18     0.57
20     0.46
19     0.29
14     0.28
17     0.11
9      0.07
Name: proportion, dtype: float64
```

In [247]:
```
attrs = ['Gen
         der', 'Age', 'Occupation', 'City_Category', 'Stay_In_Current_City_Years', 'Marital_Status', 'Produc
sns.set_style("white")

fig, axs = plt.subplots(nrows=3, ncols=2, figsize=(20, 16))
fig.subplots_adjust(top=1.3)
count = 0
for row in range(3):
    for col in range(2):
        sns.boxplot(data=df, y='Purchase', x=attrs[count], ax=axs[row, col], palette='Set3')
        axs[row,col].set_title(f"Purchase vs {attrs[count]}", pad=12, fontsize=13)
        count += 1
plt.show()

plt.figure(figsize=(10, 8))
sns.boxplot(data=df, y='Purchase', x=attrs[-1], palette='Set3')
plt.show()
```





In [256]:
```
fig = plt.figure(figsize = (12,8))

plt.subplot(2,3,1)
sns.boxplot(data = df, x = 'Gender', y = 'Purchase')
plt.title('Purchase vs Gender', fontweight = 'bold')

plt.subplot(2,3,3)
sns.boxplot(data = df, x = 'City_Category', y = 'Purchase')
plt.title('Purchase vs City_Category', fontweight = 'bold')

plt.subplot(2,3,5)
sns.boxplot(data = df, x = 'Marital_Status', y = 'Purchase')
plt.title('Purchase vs Marital_Status', fontweight = 'bold')

plt.show()
```



In [287]:
```
# Checking the variation in the mean/ average of customers using Hypothesis Testing.
# Using ANOVA, we can find the variation in the mean spending for 5% significance.
# Spending based on Marital_Status

# H0: All means are similar
# Ha: Means are different.
# Alpha = 0.05 (95% Confidence Level)

married = df[df['Marital_Status'] == 0]['Purchase']
unmarried = df[df['Marital_Status'] == 1]['Purchase']
```

In [288]:
```
from scipy.stats import f_oneway
```

In [289]:
```
f_stats, p_value = f_oneway(married, unmarried)
```

In [290]:
```
p_value
```

Out[290]:
```
0.7310947526475329
```

In [297]:
```
# Spending based on Age

age18_25 = df[df['Age'] == '18-25']['Purchase']
age26_35 = df[df['Age'] == '26-35']['Purchase']
age36_45 = df[df['Age'] == '36-45']['Purchase']
```

In [ ]:
```
# Since, p_value > alpha, We fail to reject the Null Hypothesis (H0).
# Hence, we can conclude that with a 95% Confidence Level, there is no difference in the
# average spending of customers based on their Marital Status.
```

In [ ]:
```
# 2. Spending based on Age
# H0: All means are similar
# Ha: Means are different.
# Alpha = 0.05 (95% Confidence Level)
```

In [298]:
```
f_stats, p_value = f_oneway(age18_25, age26_35, age36_45)
```

In [299]:
```
p_value
```

Out[299]:
```
1.6399600244032668e-12
```

In [ ]:
```
# Since, p_value < alpha, We reject the Null Hypothesis (H0).
# Hence, we can conclude that with a 95% Confidence Level, there is a significant
# difference in the average spending of customers based in the age-groups 18-25, 26-35, 36-45.
```

In [ ]:
```
# Recommendations -
# Walmart must focus to retain the customers in these age-groups by providing discount
# offers and special contests to attract more customers belonging to these age-groups to
# increase their revenue
```