In [1]:

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

In [2]:

```
walmart=pd.read_csv('walmart.csv')
```

In [3]:

```
walmart.describe()
```

Out[3]:

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	3.252600e+04	32525.000000	32525.000000	32525.000000	32525.000000
mean	1.002509e+06	8.261091	0.401107	5.303397	9270.080461
std	1.436629e+03	6.552088	0.490130	3.735383	4944.819000
min	1.000001e+06	0.000000	0.000000	1.000000	185.000000
25%	1.001262e+06	3.000000	0.000000	1.000000	5846.000000
50%	1.002496e+06	7.000000	0.000000	5.000000	8048.000000
75%	1.003773e+06	14.000000	1.000000	8.000000	12027.000000
max	1.004989e+06	20.000000	1.000000	18.000000	23958.000000

In [4]:

```
walmart.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32526 entries, 0 to 32525
Data columns (total 10 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   User_ID                     32526 non-null  int64  
 1   Product_ID                  32526 non-null  object 
 2   Gender                      32525 non-null  object 
 3   Age                         32525 non-null  object 
 4   Occupation                  32525 non-null  float64
 5   City_Category               32525 non-null  object 
 6   Stay_In_Current_City_Years  32525 non-null  object 
 7   Marital_Status              32525 non-null  float64
 8   Product_Category            32525 non-null  float64
 9   Purchase                    32525 non-null  float64
dtypes: float64(4), int64(1), object(5)
memory usage: 2.5+ MB
```

In [5]:

```
pd.isnull(walmart).sum()
```

Out[5]:

```
User_ID                       0
Product_ID                    0
Gender                        1
Age                           1
Occupation                    1
City_Category                 1
Stay_In_Current_City_Years    1
```

```
Stay_In_Current_City_Years    1
Marital_Status                1
Product_Category              1
Purchase                      1
dtype: int64
```

In [6]:

```
walmart = walmart.drop(walmart.index[32525])
```

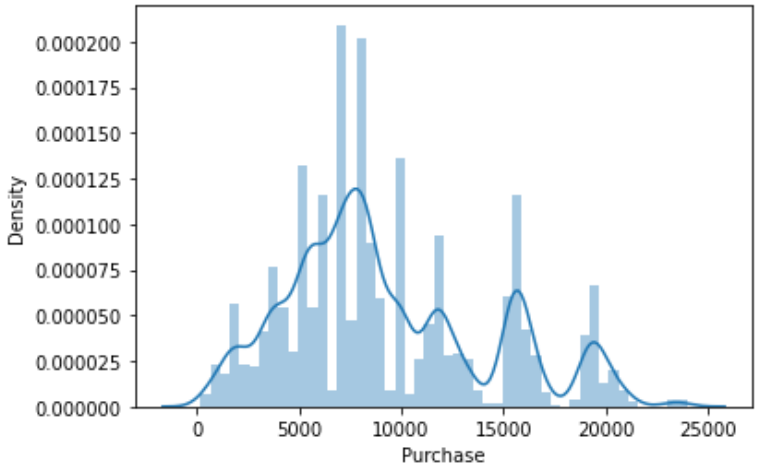
In [7]:

```
sns.distplot(walmart['Purchase'])
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[7]:

```
<AxesSubplot:xlabel='Purchase', ylabel='Density'>
```



In [8]:

```
f=walmart[walmart['Gender']=='F']
m=walmart[walmart['Gender']=='M']
```

In [9]:

```
f.describe()
```

Out[9]:

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	7.556000e+03	7556.000000	7556.000000	7556.000000	7556.000000
mean	1.002498e+06	6.874934	0.408682	5.591583	8734.241265
std	1.442465e+03	6.257933	0.491623	3.426367	4657.438584
min	1.000001e+06	0.000000	0.000000	1.000000	347.000000
25%	1.001203e+06	1.000000	0.000000	3.000000	5447.000000
50%	1.002441e+06	4.000000	0.000000	5.000000	7913.500000
75%	1.003713e+06	12.000000	1.000000	8.000000	11042.500000
max	1.004987e+06	20.000000	1.000000	18.000000	23910.000000

In [10]:

```
m.describe()
```

Out[10]:

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	2.496900e+04	24969.000000	24969.000000	24969.000000	24969.000000
mean	1.002512e+06	8.680564	0.398815	5.216188	9432.233570
std	1.434813e+03	6.581362	0.489664	3.819759	5017.371624
min	1.000002e+06	0.000000	0.000000	1.000000	185.000000
25%	1.001272e+06	3.000000	0.000000	1.000000	5889.000000
50%	1.002504e+06	7.000000	0.000000	5.000000	8100.000000
75%	1.003778e+06	15.000000	1.000000	8.000000	12278.000000
max	1.004989e+06	20.000000	1.000000	18.000000	23958.000000

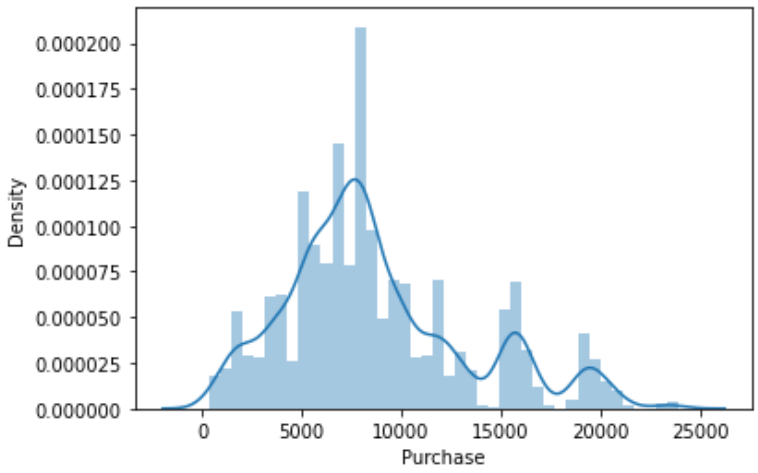
In [11]:

```
sns.distplot(f['Purchase'])
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[11]:

```
<AxesSubplot:xlabel='Purchase', ylabel='Density'>
```



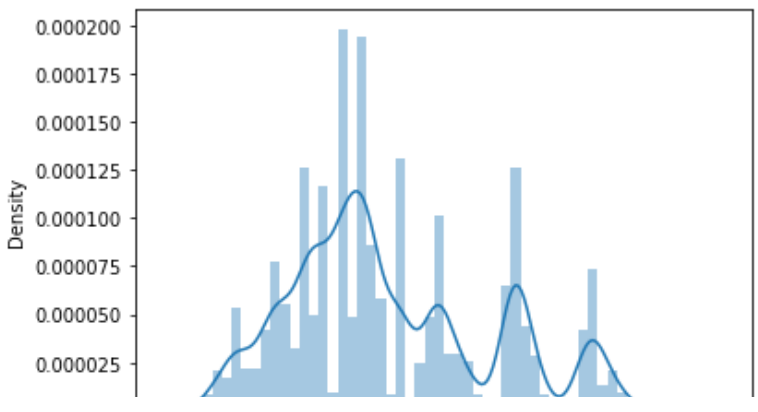
In [12]:

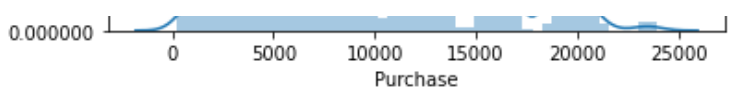
```
sns.distplot(m['Purchase'])
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[12]:

```
<AxesSubplot:xlabel='Purchase', ylabel='Density'>
```





In [13]:

```
f_mean=f['Purchase'].mean()
m_mean=m['Purchase'].mean()
```

In [14]:

```
mea=walmart['Purchase'].mean()
```

In [15]:

```
[mea,f_mean,m_mean]
```

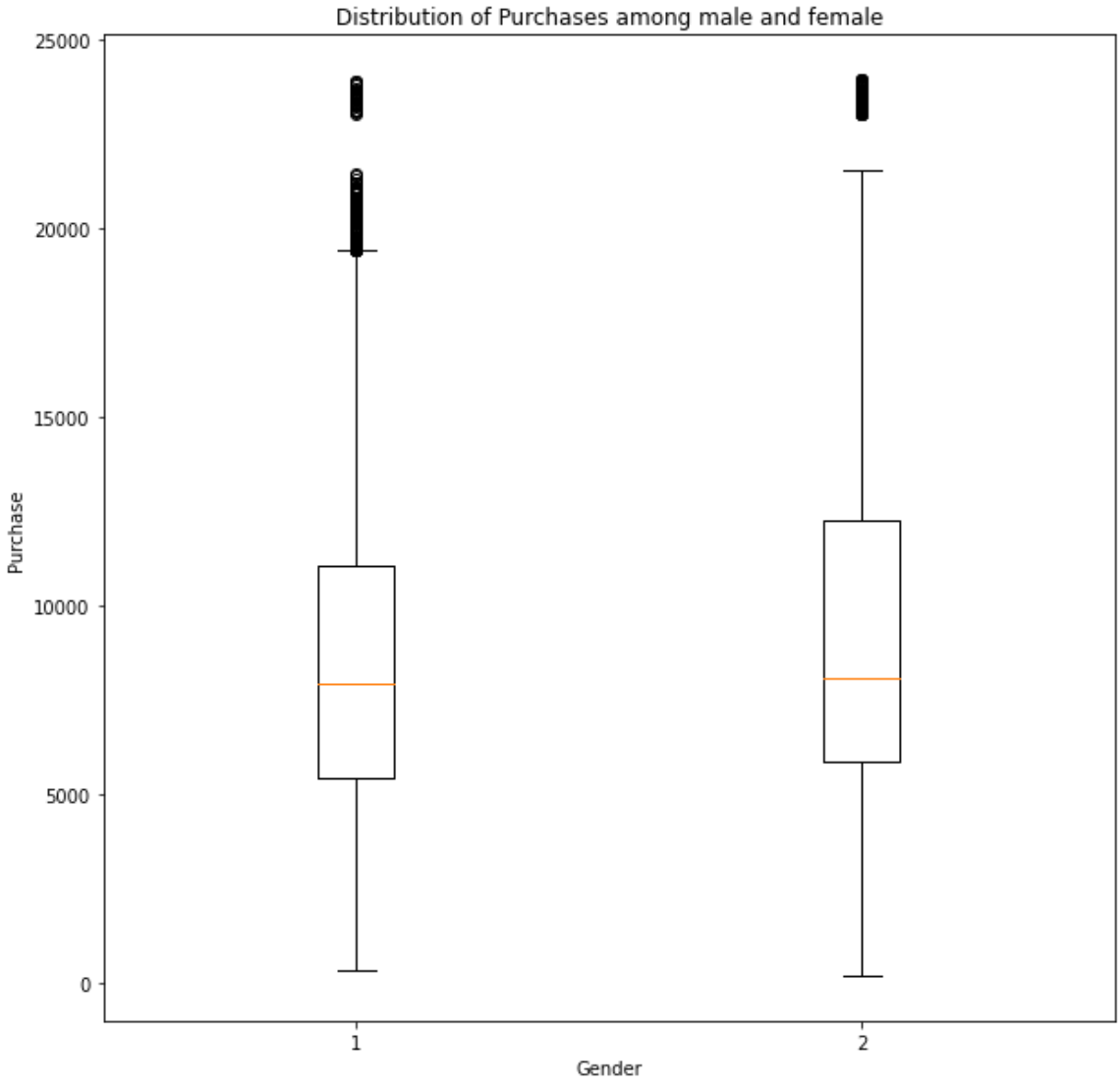
Out[15]:

```
[9270.080461183705, 8734.241265219693, 9432.233569626336]
```

In [16]:

```
tm=[f['Purchase'],m['Purchase']]
plt.figure(figsize=(10,10))
# Creating plot
plt.boxplot(tm)
plt.xlabel('Gender')
plt.ylabel('Purchase')
plt.title('Distribution of Purchases among male and female')

# show plot
plt.show()
```



Interval caluclation of Whole population,Females and males per transaction

Confidence Interval for whole population per transaction

In [17]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(mea-((t_val[i])*(np.std(walmart['Purchase']))/math.sqrt(len(walmar
t))),3)
    higher_limit=round(mea+((t_val[i])*(np.std(walmart['Purchase']))/math.sqrt(len(walma
rt))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [9224.978, 9315.183]
95 Confidence interval : [9216.341, 9323.82]
98 Confidence interval : [9206.306, 9333.855]
99 Confidence interval : [9199.452, 9340.709]
99.9 Confidence interval : [9179.848, 9360.313]
```

Confidence Interval for Female population per transaction

In [18]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(f_mean-((t_val[i])*(np.std(f['Purchase']))/math.sqrt(len(f))),3)
    higher_limit=round(f_mean+((t_val[i])*(np.std(f['Purchase']))/math.sqrt(len(f))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [8646.108, 8822.374]
95 Confidence interval : [8629.232, 8839.251]
98 Confidence interval : [8609.623, 8858.86]
99 Confidence interval : [8596.229, 8872.254]
99.9 Confidence interval : [8557.922, 8910.561]
```

Confidence Interval for Male population per transaction

In [19]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(m_mean-((t_val[i])*(np.std(m['Purchase']))/math.sqrt(len(m))),3)
    higher_limit=round(m_mean+((t_val[i])*(np.std(m['Purchase']))/math.sqrt(len(m))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [9380.002, 9484.465]
95 Confidence interval : [9370.0, 9494.467]
98 Confidence interval : [9358.379, 9506.088]
99 Confidence interval : [9350.441, 9514.026]
99.9 Confidence interval : [9327.739, 9536.728]
```

Caluclating Interval of Whole population,Females and males per transaction using CLT

In [20]:

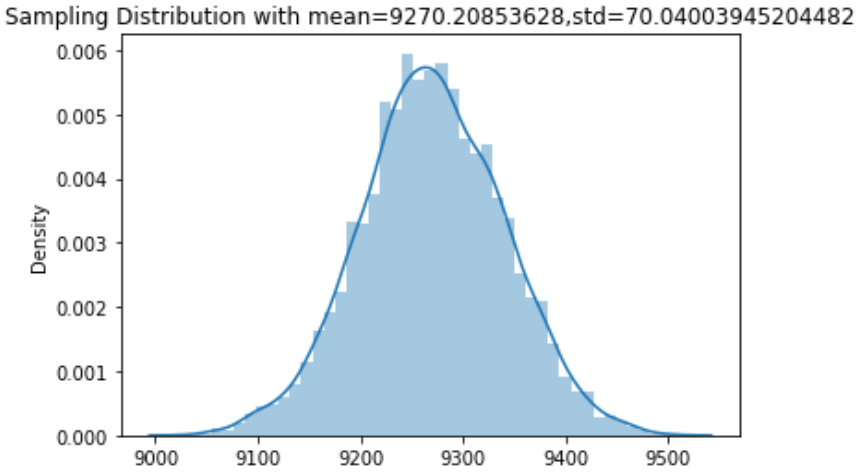
```
w_l=[]
for i in range(5000):
    sample=np.random.choice(walmart['Purchase'],5000)
    w_l.append(sample.mean())
sns.distplot(w_l)
```

```
plt.title(f"Sampling Distribution with mean={np.mean(w_l)},std={np.std(w_l)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[20]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=9270.20853628,std=70.04003945204482')
```



Confidence Interval for whole population per transaction using CLT

In [21]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):

    print(str(CI[i])+" Confidence interval : " +str(np.percentile(w_l,[100-CI[i],CI[i]])
))
```

```
90 Confidence interval : [9182.70924 9360.69458]
95 Confidence interval : [9156.07856 9385.22325]
98 Confidence interval : [9124.169704 9414.379376]
99 Confidence interval : [9100.677156 9432.341224]
99.9 Confidence interval : [9058.4073812 9473.3623482]
```

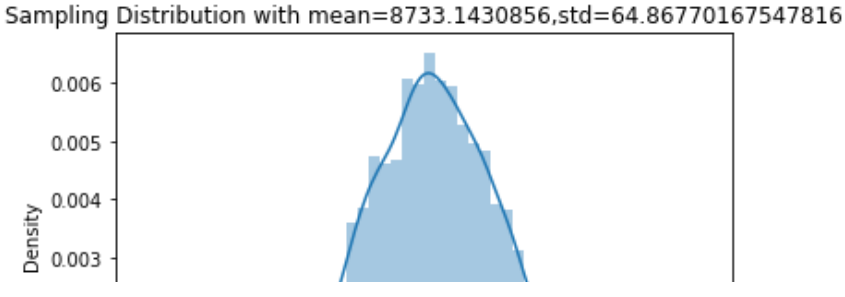
In [22]:

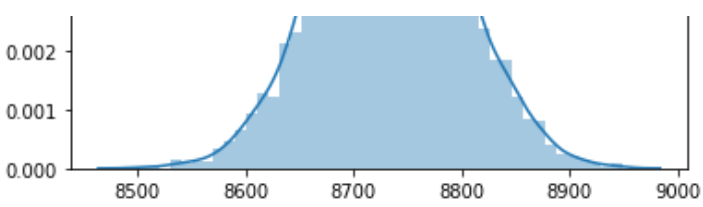
```
f_l=[]
for i in range(5000):
    sample=np.random.choice(f['Purchase'],5000)
    f_l.append(sample.mean())
sns.distplot(f_l)
plt.title(f"Sampling Distribution with mean={np.mean(f_l)},std={np.std(f_l)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[22]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=8733.1430856,std=64.86770167547816')
```





Confidence Interval for female population per transaction using CLT

In [23]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):

    print(str(CI[i])+" Confidence interval : " +str(np.percentile(f_l,[100-CI[i],CI[i]])
))
```

```
90 Confidence interval : [8651.89892 8816.57184]
95 Confidence interval : [8628.03053 8841.559  ]
98 Confidence interval : [8601.250756 8867.088876]
99 Confidence interval : [8585.868204 8880.390442]
99.9 Confidence interval : [8531.2520182 8927.6883756]
```

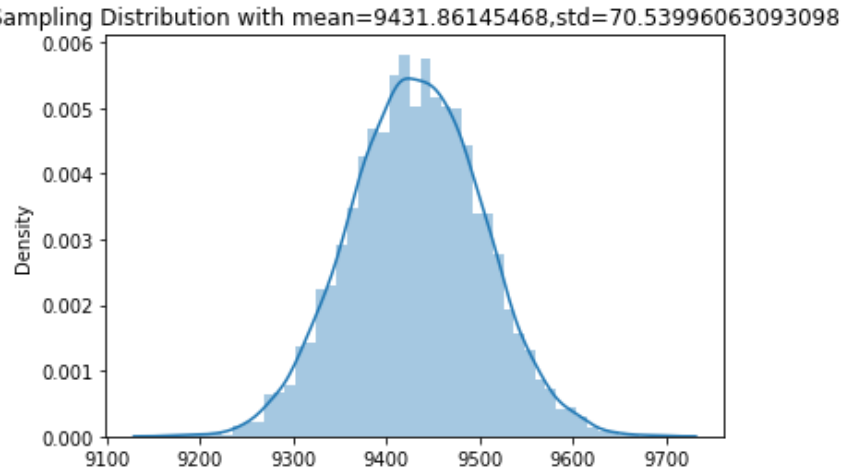
In [24]:

```
m_l=[]
for i in range(5000):
    sample=np.random.choice(m['Purchase'],5000)
    m_l.append(sample.mean())
sns.distplot(m_l)
plt.title(f"Sampling Distribution with mean={np.mean(m_l)},std={np.std(m_l)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[24]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=9431.86145468,std=70.53996063093098')
```



Confidence Interval for male population per transaction using CLT

In [25]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):

    print(str(CI[i])+" Confidence interval : " +str(np.percentile(m_l,[100-CI[i],CI[i]])
))
```

```
90 Confidence interval : [9341.09574 9521.06894]
95 Confidence interval : [9315.17273 9547.40206]
```

```
98 Confidence interval : [9286.740212 9575.57454 ]
99 Confidence interval : [9270.762398 9597.911654]
99.9 Confidence interval : [9215.343878  9646.1874708]
```

Interval caluclation of Whole population,Females and males per User_ID

In [26]:

```
User_subset=walmart.groupby('User_ID').sum().reset_index()

female_subset=f.groupby('User_ID').sum().reset_index()

male_subset=m.groupby('User_ID').sum().reset_index()
```

In [27]:

```
avg=round(walmart['Purchase'].sum()/len(walmart['User_ID'].unique()),3)
avg
```

Out[27]:

```
68587.208
```

In [28]:

```
female_avg=round(f['Purchase'].sum()/len(f['User_ID'].unique()),3)
female_avg
```

Out[28]:

```
56795.118
```

In [29]:

```
male_avg=round(m['Purchase'].sum()/len(m['User_ID'].unique()),3)
male_avg
```

Out[29]:

```
72824.193
```

In [30]:

```
len(walmart['User_ID'].unique())
```

Out[30]:

```
4396
```

Confidence Interval for Whole population per User_ID

In [31]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(avg-((t_val[i])*(np.std(User_subset['Purchase']))/math.sqrt(len(Us
er_subset))),3)
    higher_limit=round(avg+((t_val[i])*(np.std(User_subset['Purchase']))/math.sqrt(len(U
ser_subset))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [66714.405, 70460.011]
95 Confidence interval : [66355.783, 70818.633]
98 Confidence interval : [65939.099, 71235.317]
99 Confidence interval : [65654.479, 71519.937]
99.9 Confidence interval : [64840.464, 72333.952]
```

Confidence Interval for Female population per User_ID

In [32]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(female_avg-((t_val[i])*(np.std(female_subset['Purchase']))/math.sq
rt(len(female_subset))),3)
    higher_limit=round(female_avg+((t_val[i])*(np.std(female_subset['Purchase']))/math.s
qrt(len(female_subset))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [53706.039, 59884.197]
95 Confidence interval : [53114.514, 60475.722]
98 Confidence interval : [52427.217, 61163.019]
99 Confidence interval : [51957.752, 61632.484]
99.9 Confidence interval : [50615.083, 62975.153]
```

Confidence Interval for Male population per User_ID

In [33]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(male_avg-((t_val[i])*(np.std(male_subset['Purchase']))/math.sqrt(l
en(male_subset))),3)
    higher_limit=round(male_avg+((t_val[i])*(np.std(male_subset['Purchase']))/math.sqrt(
len(male_subset))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [70545.62, 75102.766]
95 Confidence interval : [70109.298, 75539.088]
98 Confidence interval : [69602.333, 76046.053]
99 Confidence interval : [69256.045, 76392.341]
99.9 Confidence interval : [68265.662, 77382.724]
```

Caluclating Interval of Whole population,Females and males per User_ID using CLT

In [34]:

```
l=[]
for i in range(1000):
    sample=np.random.choice(User_subset['Purchase'],3000)
    l.append(sample.mean())
```

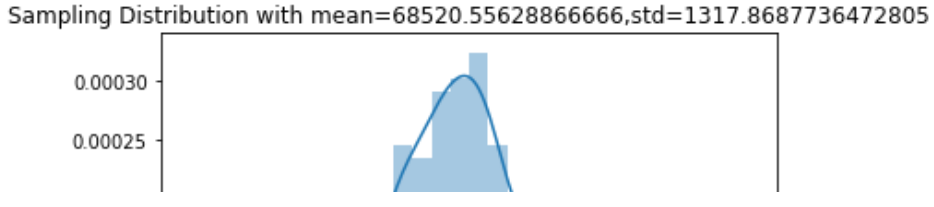
In [35]:

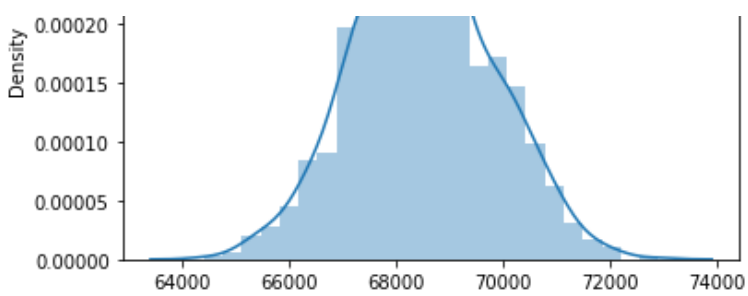
```
sns.distplot(l)
plt.title(f"Sampling Distribution with mean={np.mean(l)},std={np.std(l)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[35]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=68520.55628866666,std=1317.8687736472805'
)
```





Confidence Interval for Whole population per User_ID using CLT

In [40]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):

    print(str(CI[i])+" Confidence interval : " +str(np.percentile(l,[100-CI[i],CI[i]])))
```

```
90 Confidence interval : [66895.77266667 70275.22223333]
95 Confidence interval : [66372.8948     70737.58083333]
98 Confidence interval : [65824.98555333 71159.62777333]
99 Confidence interval : [65481.07330667 71615.84783667]
99.9 Confidence interval : [64803.98206167 72217.54763   ]
```

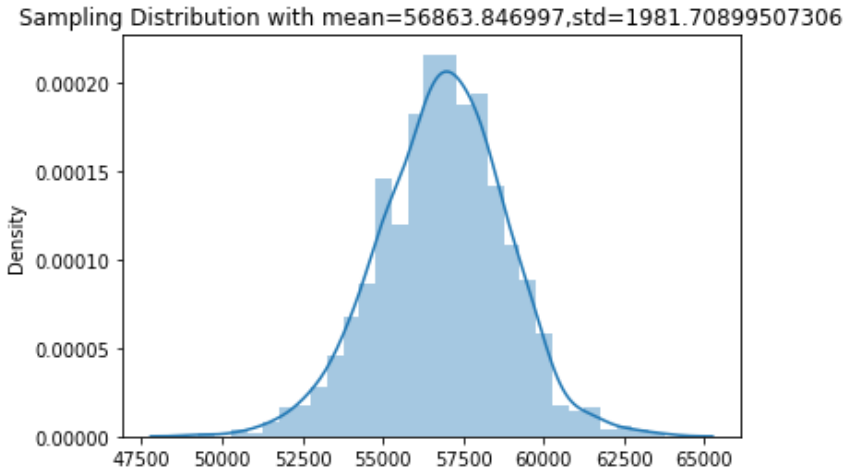
In [41]:

```
fl=[]
for i in range(1000):
    sample=np.random.choice(female_subset['Purchase'],1000)
    fl.append(sample.mean())
sns.distplot(fl)
plt.title(f"Sampling Distribution with mean={np.mean(fl)},std={np.std(fl)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[41]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=56863.846997,std=1981.70899507306')
```



Confidence Interval for Female population per User_ID using CLT

In [42]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):

    print(str(CI[i])+" Confidence interval : " +str(np.percentile(fl,[100-CI[i],CI[i]]))
)
```

```
90 Confidence interval : [54348.2128 59312.9495]
```

```
90 Confidence interval : [54348.2128  59512.9495]
95 Confidence interval : [53586.6625  59910.72295]
98 Confidence interval : [52664.8989  60881.12174]
99 Confidence interval : [51912.03964 61616.12472]
99.9 Confidence interval : [50330.690864 63220.155851]
```

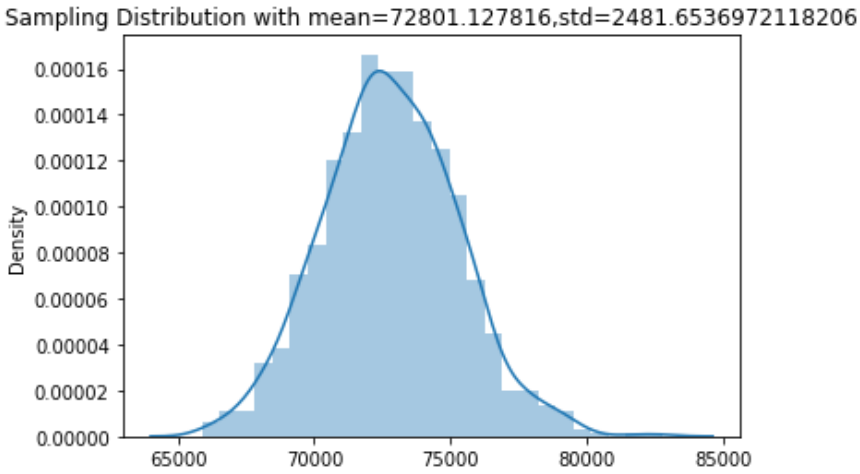
In [43]:

```
ml=[]
for i in range(1000):
    sample=np.random.choice(male_subset['Purchase'],1000)
    ml.append(sample.mean())
sns.distplot(ml)
plt.title(f"Sampling Distribution with mean={np.mean(ml)},std={np.std(ml)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[43]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=72801.127816,std=2481.6536972118206')
```



Confidence Interval for Male population per User_ID using CLT

In [44]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):

    print(str(CI[i])+" Confidence interval : " +str(np.percentile(ml,[100-CI[i],CI[i]]))
)
```

```
90 Confidence interval : [69599.7392 75892.1268]
95 Confidence interval : [68755.0186  76755.68795]
98 Confidence interval : [67872.36458 78238.37788]
99 Confidence interval : [67163.21441 78941.69185]
99.9 Confidence interval : [65996.228171 81788.520471]
```

Interval caluclation for single and married

In [45]:

```
avg_1=walmart['Purchase'].mean()
avg_1
```

Out[45]:

```
9270.080461183705
```

In [46]:

```
single=walmart[walmart['Marital_Status']==0.0]
married=walmart[walmart['Marital_Status']==1.0]
```

In [47]:

```
single_avg=round(single['Purchase'].mean(),3)
single_avg
```

Out[47]:

9234.775

In [48]:

```
married_avg=round(married['Purchase'].mean(),3)
married_avg
```

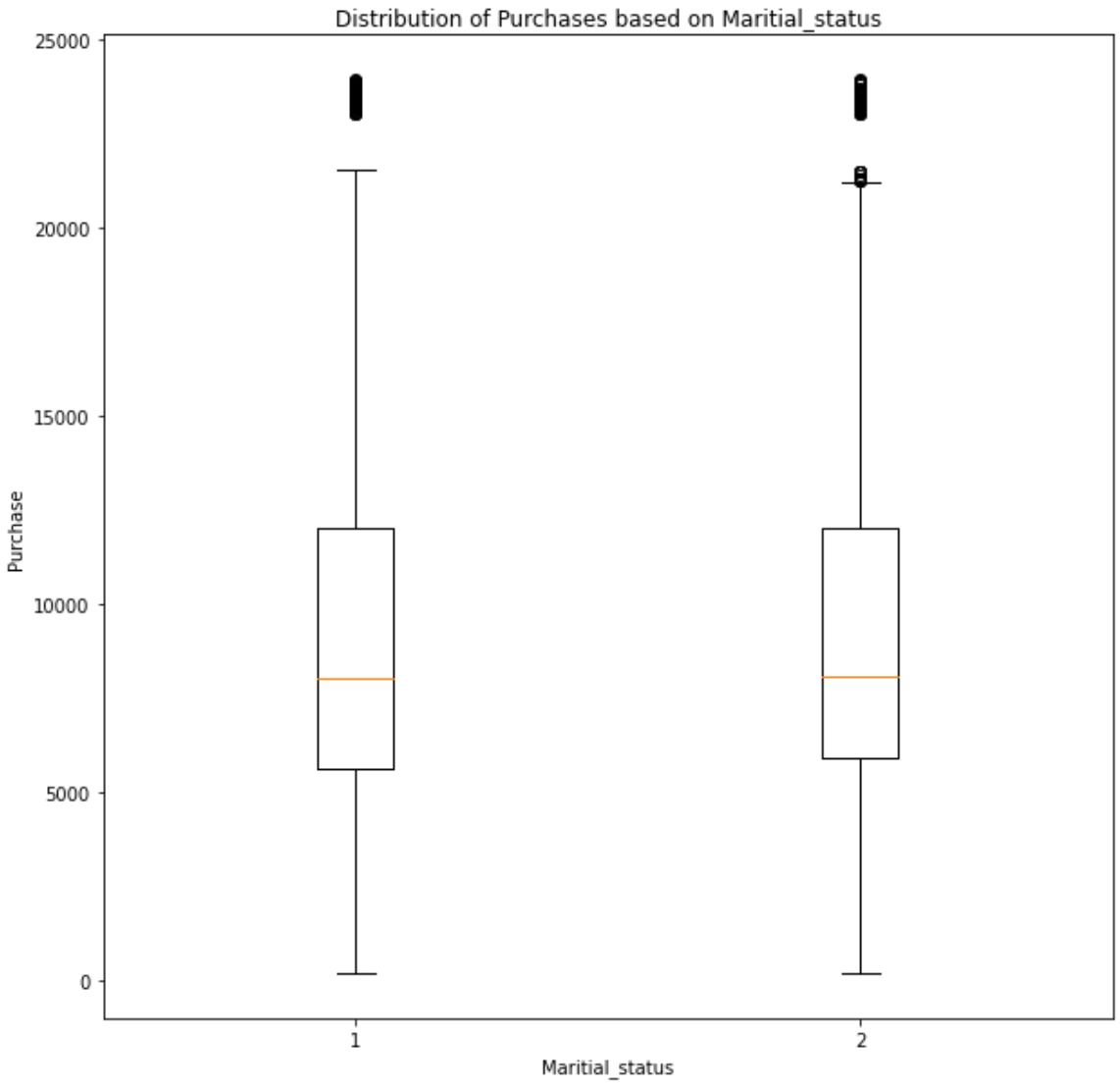
Out[48]:

9322.796

In [49]:

```
tm1=[single['Purchase'],married['Purchase']]
plt.figure(figsize=(10,10))
# Creating plot
plt.boxplot(tm1)
plt.xlabel('Maritial_status')
plt.ylabel('Purchase')
plt.title('Distribution of Purchases based on Maritial_status')

# show plot
plt.show()
```



Confidence Interval for all the population

In [50]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(avg_1-((t_val[i])*(np.std(walmart['Purchase']))/math.sqrt(len(walm
art))),3)
    higher_limit=round(avg_1+((t_val[i])*(np.std(walmart['Purchase']))/math.sqrt(len(wal
mart))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [9224.978, 9315.183]
95 Confidence interval : [9216.341, 9323.82]
98 Confidence interval : [9206.306, 9333.855]
99 Confidence interval : [9199.452, 9340.709]
99.9 Confidence interval : [9179.848, 9360.313]
```

Confidence Interval for single population

In [51]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(single_avg-((t_val[i])*(np.std(single['Purchase']))/math.sqrt(len(
single))),3)
    higher_limit=round(single_avg+((t_val[i])*(np.std(single['Purchase']))/math.sqrt(len
(single))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [9176.392, 9293.158]
95 Confidence interval : [9165.213, 9304.337]
98 Confidence interval : [9152.223, 9317.327]
99 Confidence interval : [9143.35, 9326.2]
99.9 Confidence interval : [9117.974, 9351.576]
```

Confidence Interval for Married population

In [52]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(married_avg-((t_val[i])*(np.std(married['Purchase']))/math.sqrt(le
n(married))),3)
    higher_limit=round(married_avg+((t_val[i])*(np.std(married['Purchase']))/math.sqrt(l
en(married))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [9251.774, 9393.818]
95 Confidence interval : [9238.174, 9407.418]
98 Confidence interval : [9222.372, 9423.22]
99 Confidence interval : [9211.578, 9434.014]
99.9 Confidence interval : [9180.709, 9464.883]
```

Interval caluclation for single and married using CLT

In [53]:

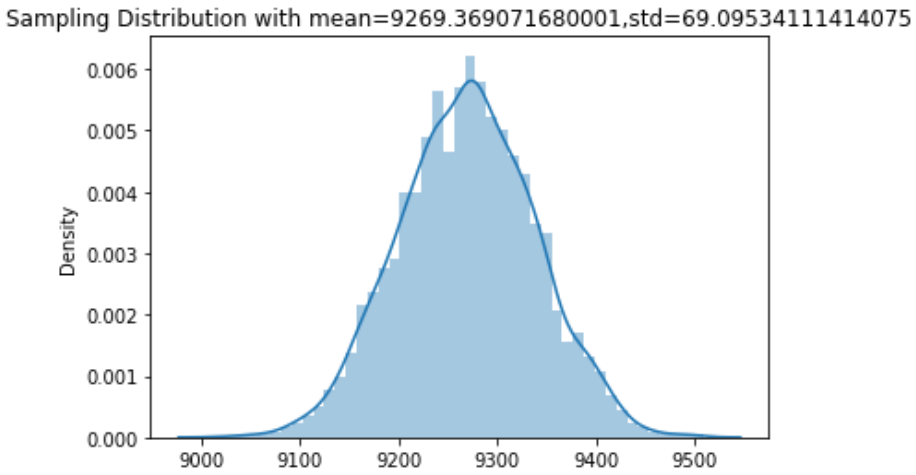
```
wal=[]
for i in range(5000):
    sample=np.random.choice(walmart['Purchase'],5000)
    wal.append(sample.mean())
sns.distplot(wal)
plt.title(f"Sampling Distribution with mean={np.mean(wal)},std={np.std(wal)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
```

```
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[53]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=9269.369071680001,std=69.09534111414075')
```



Confidence Interval for all the population using CLT

In [54]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):
    print(str(CI[i])+" Confidence interval : " +str(np.percentile(wal,[100-CI[i],CI[i]])
))
```

```
90 Confidence interval : [9179.07122 9356.49186]
95 Confidence interval : [9156.30898 9385.47481]
98 Confidence interval : [9129.28692 9409.36334]
99 Confidence interval : [9108.344438 9423.438714]
99.9 Confidence interval : [9052.1169024 9486.2037436]
```

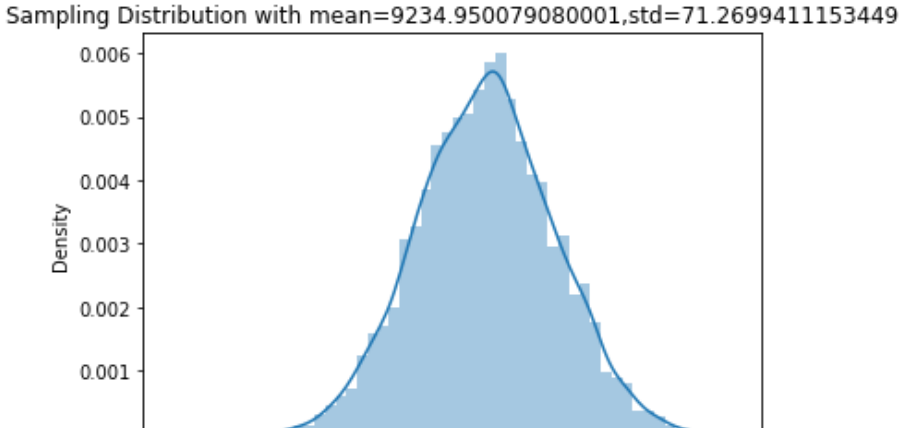
In [55]:

```
sin=[]
for i in range(5000):
    sample=np.random.choice(single['Purchase'],5000)
    sin.append(sample.mean())
sns.distplot(sin)
plt.title(f"Sampling Distribution with mean={np.mean(sin)},std={np.std(sin)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[55]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=9234.950079080001,std=71.2699411153449')
```



Confidence Interval for all the single population using CLT

In [56]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):
    print(str(CI[i])+" Confidence interval : " +str(np.percentile(sin,[100-CI[i],CI[i]])
))
```

```
90 Confidence interval : [9144.69188 9327.977  ]
95 Confidence interval : [9117.59859 9350.91005]
98 Confidence interval : [9089.321044 9380.612304]
99 Confidence interval : [9071.247142 9399.775434]
99.9 Confidence interval : [9004.3332426 9442.1726456]
```

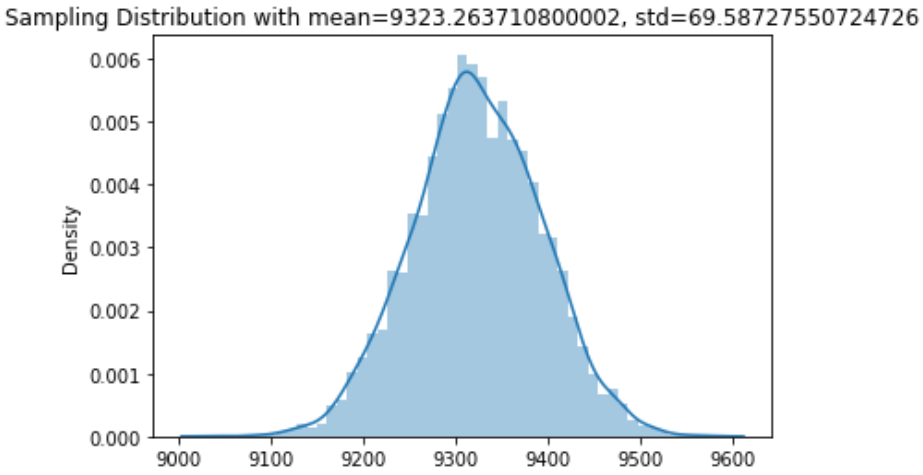
In [57]:

```
mar=[]
for i in range(5000):
    sample=np.random.choice(married['Purchase'],5000)
    mar.append(sample.mean())
sns.distplot(mar)
plt.title(f"Sampling Distribution with mean={np.mean(mar)}, std={np.std(mar)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[57]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=9323.263710800002, std=69.58727550724726'
)
```



Confidence Interval for all the married population using CLT

In [58]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):
    print(str(CI[i])+" Confidence interval : " +str(np.percentile(mar,[100-CI[i],CI[i]])
))
```

```
90 Confidence interval : [9232.99882 9412.85848]
95 Confidence interval : [9207.06373 9436.08036]
98 Confidence interval : [9181.525124 9467.248956]
99 Confidence interval : [9163.950618 9482.57704 ]
99.9 Confidence interval : [9110.8413242 9523.2081138]
```

Interval caluclation for different age groups

Interval calculation for different age groups

In [59]:

```
walmart['Age'].value_counts()
```

Out[59]:

```
26-35    12801
36-45     6627
18-25     6092
46-50     2620
51-55     2313
55+       1268
0-17       804
Name: Age, dtype: int64
```

In [60]:

```
#classification based on age groups
_0to17=walmart[walmart['Age']=='0-17']
_17to25=walmart[walmart['Age']=='18-25']
_26to35=walmart[walmart['Age']=='26-35']
_36to50=pd.concat([walmart[walmart['Age']=='36-45'],walmart[walmart['Age']=='46-50']])
_51to=pd.concat([walmart[walmart['Age']=='51-55'],walmart[walmart['Age']=='55+']])
```

In [61]:

```
_0to17['Purchase'].mean()
```

Out[61]:

```
8958.781094527363
```

In [62]:

```
_17to25['Purchase'].mean()
```

Out[62]:

```
9125.437951411688
```

In [63]:

```
_26to35['Purchase'].mean()
```

Out[63]:

```
9247.407546285447
```

In [64]:

```
_36to50['Purchase'].mean()
```

Out[64]:

```
9325.94430626149
```

In [65]:

```
_51to['Purchase'].mean()
```

Out[65]:

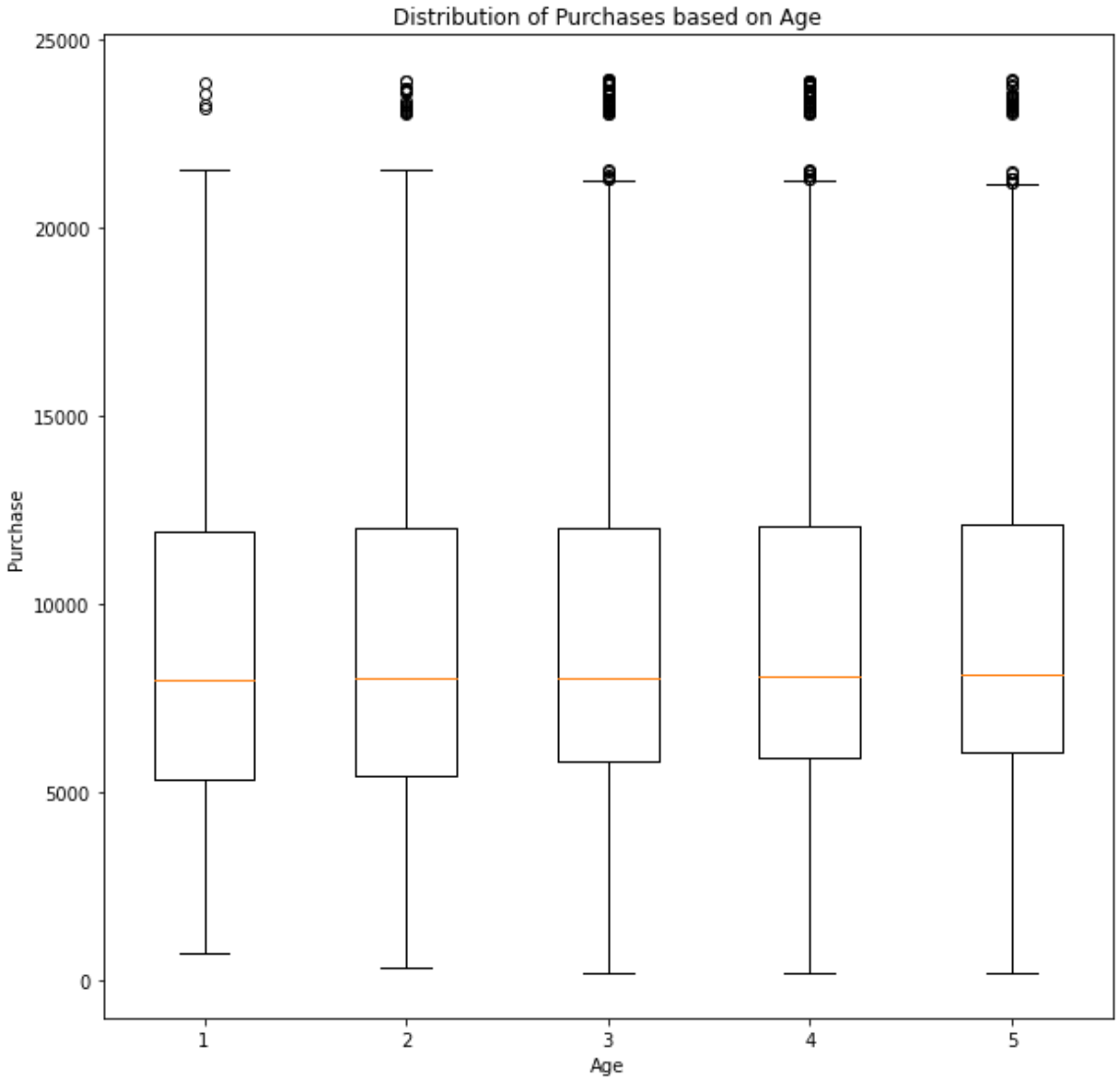
```
9522.83384529461
```

In [66]:

```
tm2=[_0to17['Purchase'],_17to25['Purchase'],_26to35['Purchase'],_36to50['Purchase'],_51t
o['Purchase']]
plt.figure(figsize=(10,10))
# Creating plot
plt.boxplot(tm2)
```

```
plt.xlabel('Age')
plt.ylabel('Purchase')
plt.title('Distribution of Purchases based on Age')

# show plot
plt.show()
```



Confidence Interval for 0 to 17 age group

In [67]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(_0to17['Purchase'].mean()-((t_val[i])*(np.std(_0to17['Purchase']))
/math.sqrt(len(_0to17))),3)
    higher_limit=round(_0to17['Purchase'].mean()+((t_val[i])*(np.std(_0to17['Purchase'])
)/math.sqrt(len(_0to17))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [8666.329, 9251.234]
95 Confidence interval : [8610.327, 9307.235]
98 Confidence interval : [8545.259, 9372.304]
99 Confidence interval : [8500.813, 9416.749]
99.9 Confidence interval : [8373.698, 9543.864]
```

Confidence Interval for 17to25 age group

In [68]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
```

```
    lower_limit=round(_17to25['Purchase'].mean()-((t_val[i])*(np.std(_17to25['Purchase']
))/math.sqrt(len(_17to25))),3)
    higher_limit=round(_17to25['Purchase'].mean()+((t_val[i])*(np.std(_17to25['Purchase'
]))/math.sqrt(len(_17to25))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [9020.925, 9229.951]
95 Confidence interval : [9000.912, 9249.964]
98 Confidence interval : [8977.659, 9273.217]
99 Confidence interval : [8961.775, 9289.101]
99.9 Confidence interval : [8916.348, 9334.527]
```

In [69]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(_26to35['Purchase'].mean()-((t_val[i])*(np.std(_26to35['Purchase']
))/math.sqrt(len(_26to35))),3)
    higher_limit=round(_26to35['Purchase'].mean()+((t_val[i])*(np.std(_26to35['Purchase'
]))/math.sqrt(len(_26to35))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [9175.552, 9319.263]
95 Confidence interval : [9161.792, 9333.023]
98 Confidence interval : [9145.805, 9349.01]
99 Confidence interval : [9134.885, 9359.93]
99.9 Confidence interval : [9103.653, 9391.162]
```

Confidence Interval for 36 to 50 age group

In [70]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(_36to50['Purchase'].mean()-((t_val[i])*(np.std(_36to50['Purchase']
))/math.sqrt(len(_36to50))),3)
    higher_limit=round(_36to50['Purchase'].mean()+((t_val[i])*(np.std(_36to50['Purchase'
]))/math.sqrt(len(_36to50))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [9241.941, 9409.948]
95 Confidence interval : [9225.855, 9426.034]
98 Confidence interval : [9207.165, 9444.724]
99 Confidence interval : [9194.398, 9457.491]
99.9 Confidence interval : [9157.886, 9494.003]
```

Confidence Interval for 50+ age group

In [71]:

```
CI=[90,95,98,99,99.9]
t_val=[1.645,1.96,2.326,2.576,3.291]
for i in range(len(CI)):
    lower_limit=round(_51to['Purchase'].mean()-((t_val[i])*(np.std(_51to['Purchase']))/m
ath.sqrt(len(_51to))),3)
    higher_limit=round(_51to['Purchase'].mean()+((t_val[i])*(np.std(_51to['Purchase']))/
math.sqrt(len(_51to))),3)
    print(str(CI[i])+" Confidence interval : " +str([lower_limit,higher_limit]))
```

```
90 Confidence interval : [9385.867, 9659.801]
95 Confidence interval : [9359.639, 9686.029]
98 Confidence interval : [9329.165, 9716.503]
99 Confidence interval : [9308.349, 9737.319]
99.9 Confidence interval : [9248.816, 9796.852]
```

Interval caluclation for different age groups using CLT

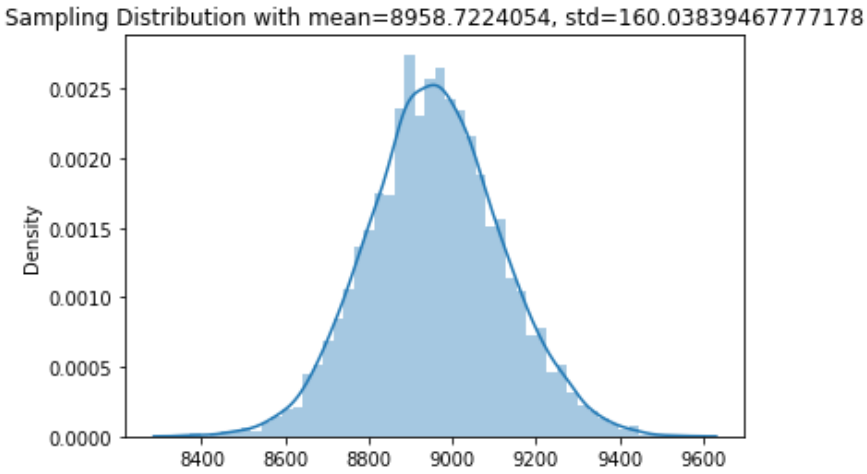
In [72]:

```
_17=[]
for i in range(5000):
    sample=np.random.choice(_0to17['Purchase'],1000)
    _17.append(sample.mean())
sns.distplot(_17)
plt.title(f"Sampling Distribution with mean={np.mean(_17)}, std={np.std(_17)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[72]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=8958.7224054, std=160.03839467777178')
```



Confidence Interval for 0 to 17 age group using CLT

In [73]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):
    print(str(CI[i])+" Confidence interval : " +str(np.percentile(_17,[100-CI[i],CI[i]])
))
```

```
90 Confidence interval : [8756.216  9164.6345]
95 Confidence interval : [8699.4886 9226.9222]
98 Confidence interval : [8636.40832 9293.80186]
99 Confidence interval : [8584.08678 9341.91561]
99.9 Confidence interval : [8455.304222 9443.818492]
```

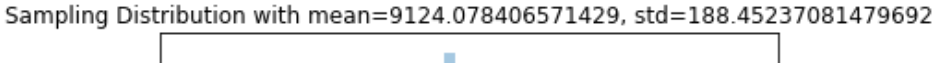
In [74]:

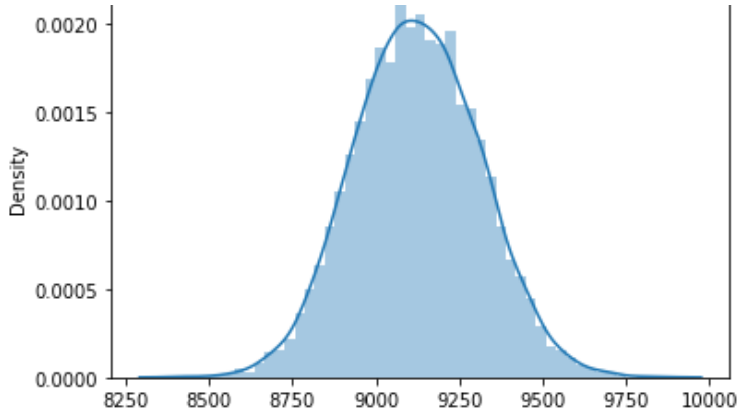
```
_25=[]
for i in range(5000):
    sample=np.random.choice(_17to25['Purchase'],700)
    _25.append(sample.mean())
sns.distplot(_25)
plt.title(f"Sampling Distribution with mean={np.mean(_25)}, std={np.std(_25)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[74]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=9124.078406571429, std=188.45237081479692
')
```





Confidence Interval for 18 to 25 age group using CLT

In [75]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):
    print(str(CI[i])+" Confidence interval : " +str(np.percentile(_25,[100-CI[i],CI[i]])
))
```

```
90 Confidence interval : [8885.25185714 9364.44371429]
95 Confidence interval : [8822.29328571 9435.39021429]
98 Confidence interval : [8746.91445714 9509.76108571]
99 Confidence interval : [8692.08078571 9565.74727143]
99.9 Confidence interval : [8549.65754429 9717.79919429]
```

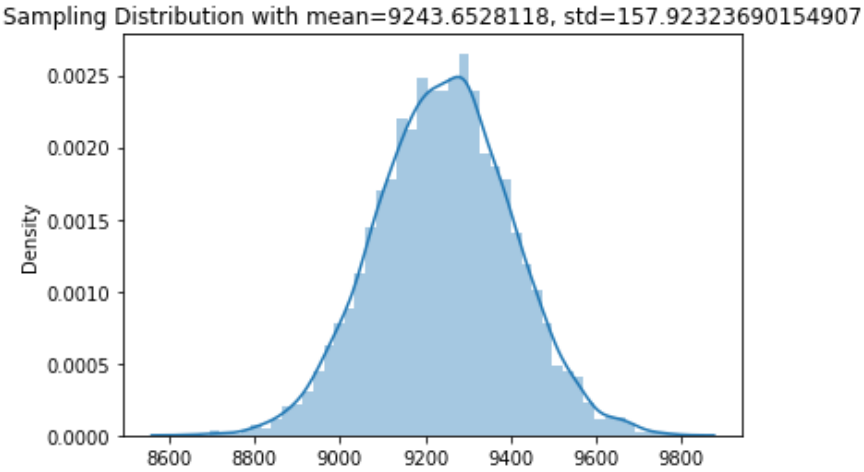
In [76]:

```
_35=[]
for i in range(5000):
    sample=np.random.choice(_26to35['Purchase'],1000)
    _35.append(sample.mean())
sns.distplot(_35)
plt.title(f"Sampling Distribution with mean={np.mean(_35)}, std={np.std(_35)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[76]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=9243.6528118, std=157.92323690154907')
```



Confidence Interval for 26 to 35 age group using CLT

In [77]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):
```

```
    print(str(CI[i])+" Confidence interval : " +str(np.percentile(_35,[100-CI[i],CI[i]])
))
```

```
90 Confidence interval : [9041.4299 9446.1094]
95 Confidence interval : [8983.82555 9499.4254 ]
98 Confidence interval : [8921.60322 9567.01938]
99 Confidence interval : [8878.58796 9624.08139]
99.9 Confidence interval : [8742.128446 9722.954621]
```

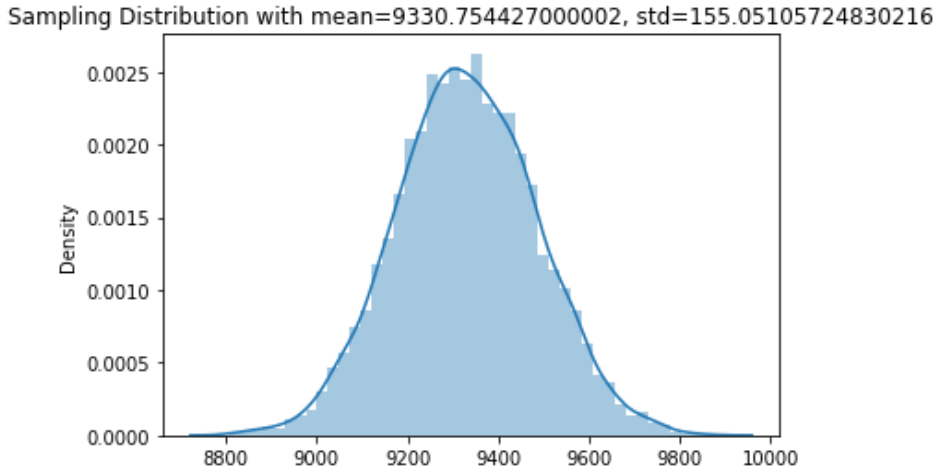
In [78]:

```
_50=[]
for i in range(5000):
    sample=np.random.choice(_36to50['Purchase'],1000)
    _50.append(sample.mean())
sns.distplot(_50)
plt.title(f"Sampling Distribution with mean={np.mean(_50)}, std={np.std(_50)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[78]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=9330.754427000002, std=155.05105724830216
')
```



Confidence Interval for 36 to 50 age group using CLT

In [79]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):
    print(str(CI[i])+" Confidence interval : " +str(np.percentile(_50,[100-CI[i],CI[i]])
))
```

```
90 Confidence interval : [9134.315 9532.949]
95 Confidence interval : [9076.14695 9586.0388 ]
98 Confidence interval : [9018.55936 9650.22578]
99 Confidence interval : [8974.66094 9697.15761]
99.9 Confidence interval : [8851.504522 9797.847863]
```

In [80]:

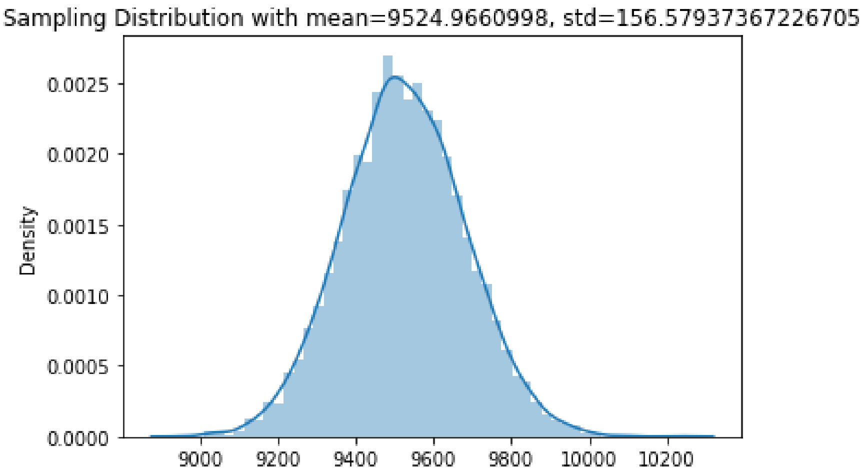
```
_51=[]
for i in range(5000):
    sample=np.random.choice(_51to['Purchase'],1000)
    _51.append(sample.mean())
sns.distplot(_51)
plt.title(f"Sampling Distribution with mean={np.mean(_51)}, std={np.std(_51)}")
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
```

```
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[80]:

```
Text(0.5, 1.0, 'Sampling Distribution with mean=9524.9660998, std=156.57937367226705')
```



Confidence Interval for 51+ age group using CLT

In [81]:

```
CI=[90,95,98,99,99.9]
for i in range(len(CI)):
    print(str(CI[i])+" Confidence interval : " +str(np.percentile(_51,[100-CI[i],CI[i]])
))
```

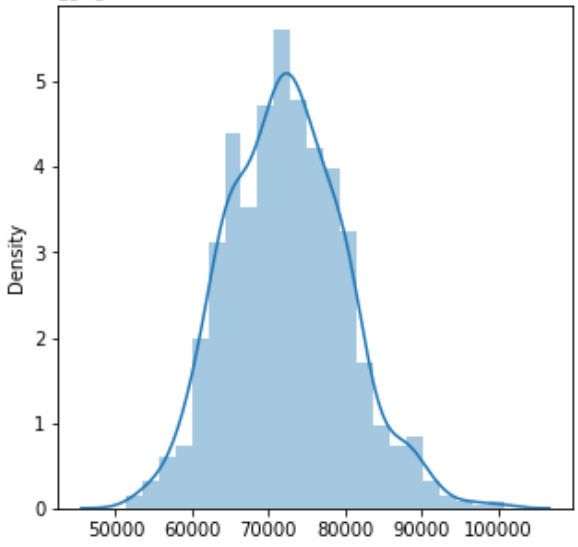
```
90 Confidence interval : [9326.2143 9727.1888]
95 Confidence interval : [9270.1199  9783.02385]
98 Confidence interval : [9207.70086 9849.0967 ]
99 Confidence interval : [9168.02709 9897.08604]
99.9 Confidence interval : [ 9038.495918 10008.192996]
```

In [88]:

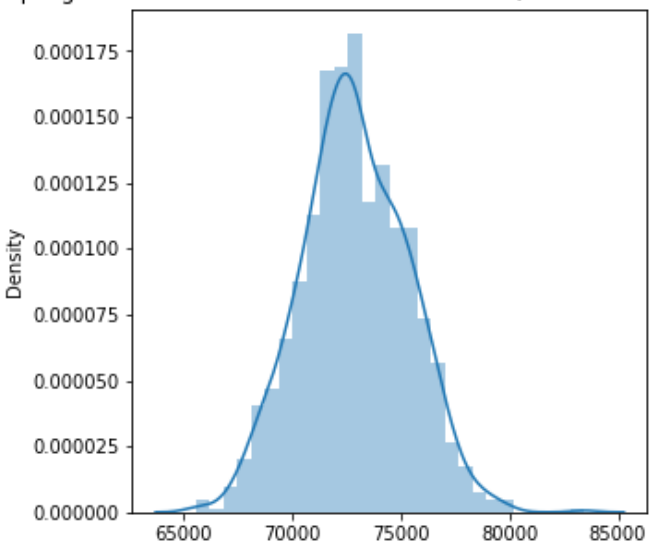
```
s=[100,1000,1500,2000]
for i in s:
    d=[]
    for j in range(1000):
        sample=np.random.choice(male_subset['Purchase'],i)
        d.append(sample.mean())
    fig, ax = plt.subplots(figsize=(5, 5))
    plt.title(f"Sampling Distribution with mean={np.mean(d)},std={np.std(d)}")
    sns.distplot(d)
```

```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

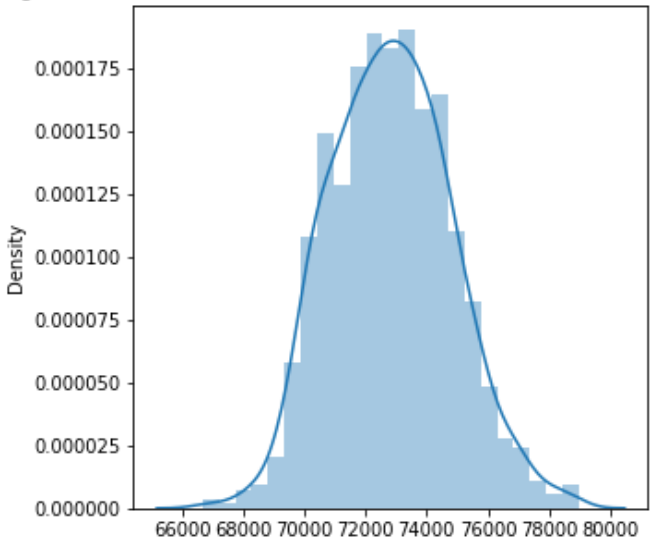
Sampling Distribution with mean=72336.52144999999,std=7706.125551006147



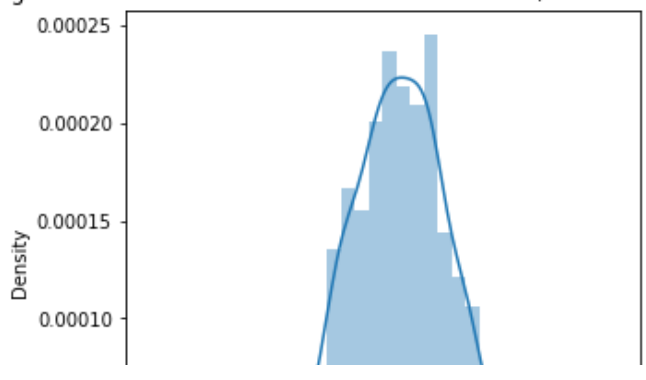
Sampling Distribution with mean=72866.470743,std=2443.699073965217

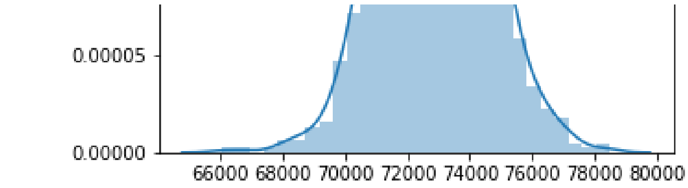


Sampling Distribution with mean=72835.41684133332,std=1976.4044162134878



Sampling Distribution with mean=72852.92831999999,std=1694.9470289788287



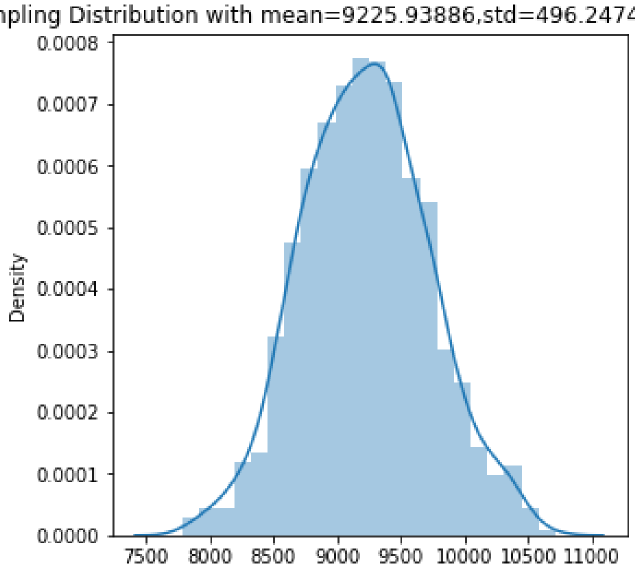


In [89]:

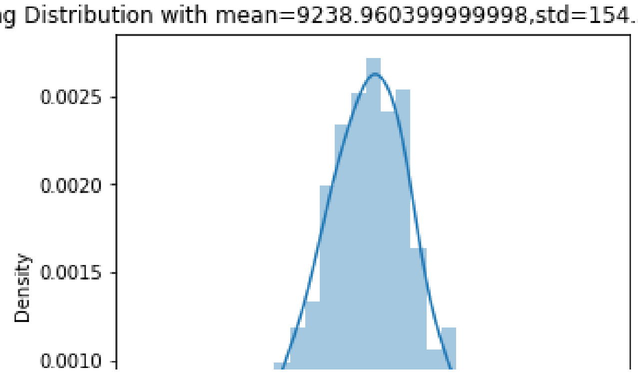
```
s=[100,1000,1500,2000]
for i in s:
    e=[]
    for j in range(1000):
        sample=np.random.choice(single['Purchase'],i)
        e.append(sample.mean())
    fig, ax = plt.subplots(figsize=(5, 5))
    plt.title(f"Sampling Distribution with mean={np.mean(e)},std={np.std(e)}")
    sns.distplot(e)
```

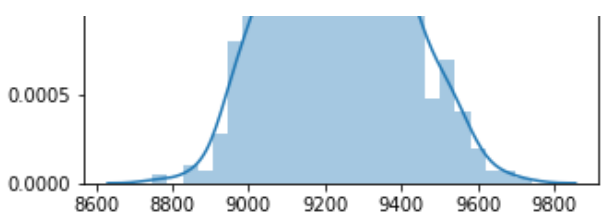
```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Sampling Distribution with mean=9225.93886,std=496.24744214544864

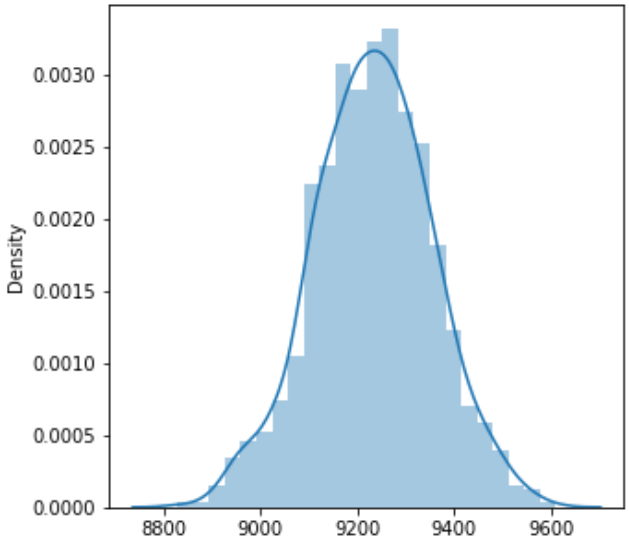


Sampling Distribution with mean=9238.960399999998,std=154.55335701137648

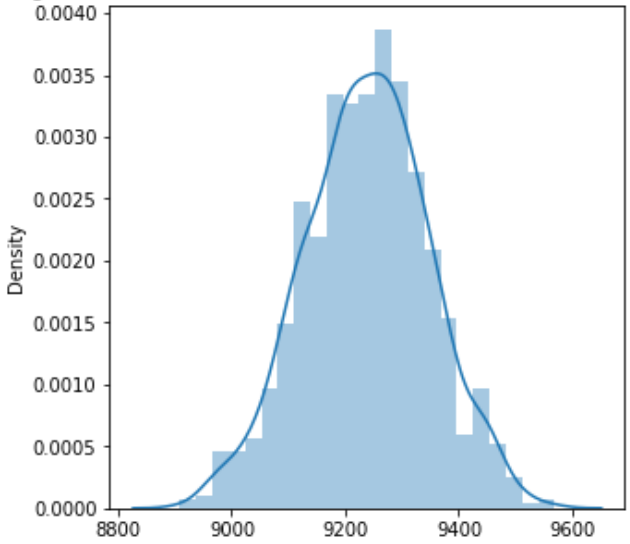




Sampling Distribution with mean=9231.765188000001,std=122.69870765489922



Sampling Distribution with mean=9238.3431895,std=109.29103722526081



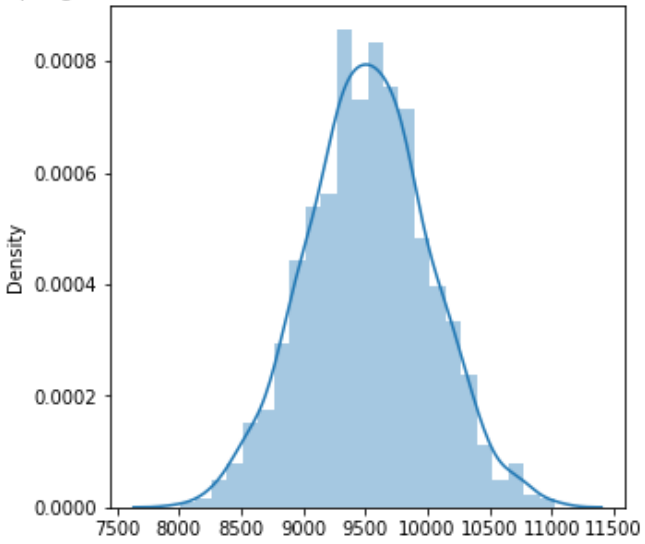
In [90]:

```
s=[100,1000,1500,2000]
for i in s:
    x=[]
    for j in range(1000):
        sample=np.random.choice(_51to['Purchase'],i)
        x.append(sample.mean())
    fig, ax = plt.subplots(figsize=(5, 5))
    plt.title(f"Sampling Distribution with mean={np.mean(x)},std={np.std(x)}")
    sns.distplot(x)
```

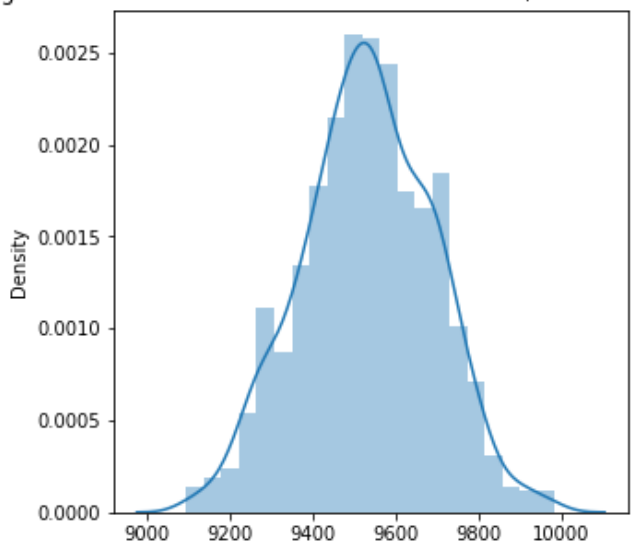
```
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
```

```
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\doolla.vennela\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn
\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be re
moved in a future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

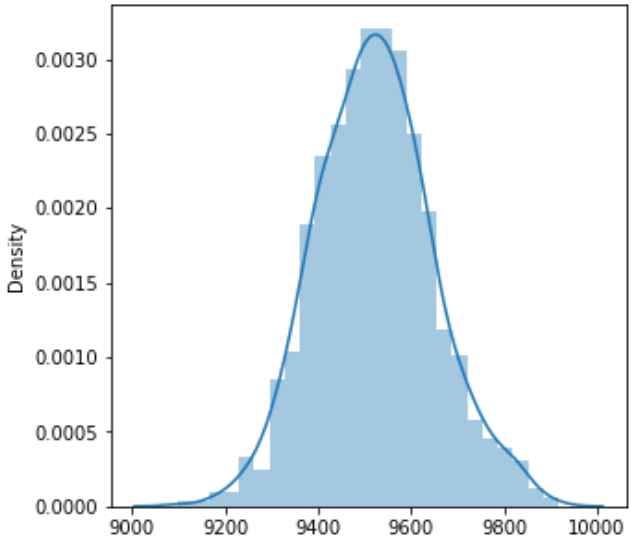
Sampling Distribution with mean=9522.32022,std=491.124643171926



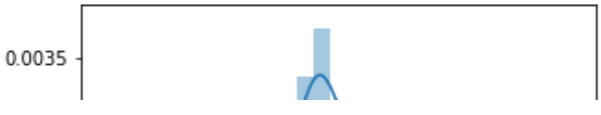
Sampling Distribution with mean=9531.424438000002,std=157.68509158075835

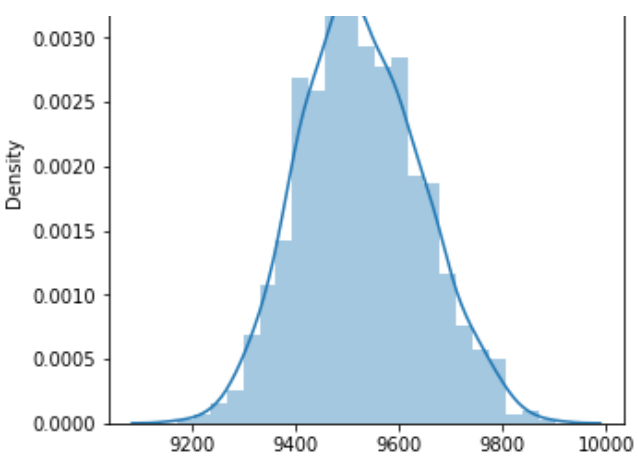


Sampling Distribution with mean=9522.384946,std=125.9019295892366



Sampling Distribution with mean=9526.9600055,std=116.10503503124326




Density
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
9200
9400
9600
9800
10000

In [ ]:

In [ ]: