In [1]:

```
import numpy as np
import pandas as pd
import matplotlib as plt
%matplotlib inline
import seaborn as sns
```

In [2]:

```
df = pd.read_csv(r'D:\projects ds\suv_data_analysis\suv_data.csv')
df.head()
```

Out[2]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

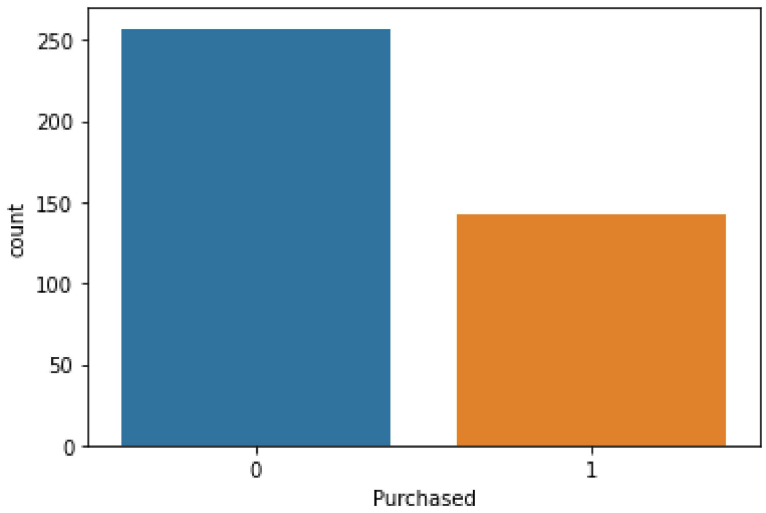
DATA ANALYSIS

In [3]:

```
sns.countplot(x='Purchased',data = df)
```

Out[3]:

```
<AxesSubplot:xlabel='Purchased', ylabel='count'>
```

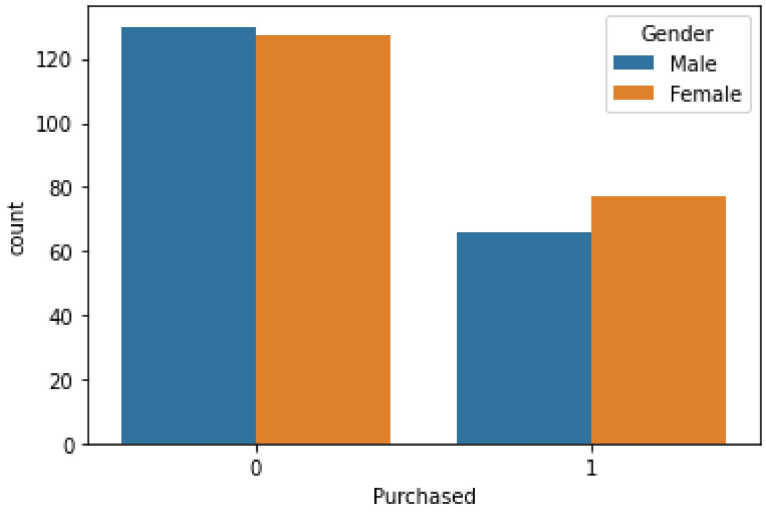


In [4]:

```
sns.countplot(x ='Purchased', hue='Gender',data = df)
```

Out[4]:

```
<AxesSubplot:xlabel='Purchased', ylabel='count'>
```

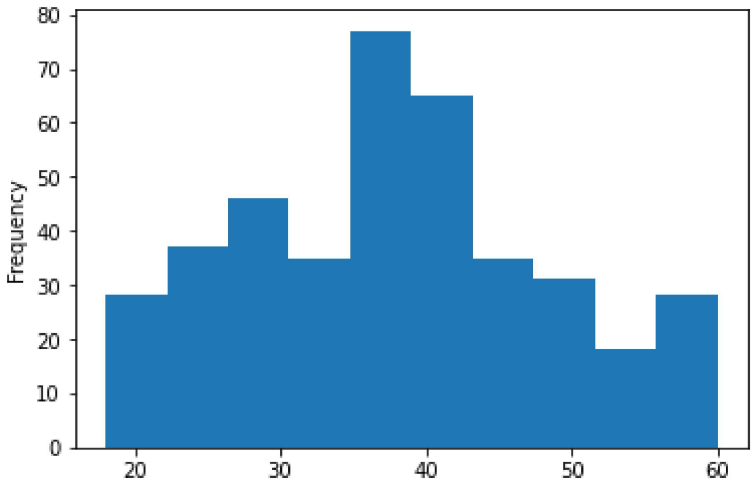


In [5]:

```
df['Age'].plot.hist()
```

Out[5]:

```
<AxesSubplot:ylabel='Frequency'>
```



DATA CLEANING

In [6]:

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

Out[6]:

```
User ID            0
Gender             0
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64
```

In [7]:

```
# NO NEED OF DATA CLEANING AS THERE ARE NO NULL VALUES
```

DATA MODELING

In [9]:

```
X = df.iloc[:,[2,3]].values
y = df.iloc[:,4].values
```

In [12]:

```
from sklearn.model_selection import train_test_split
```

In [13]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
```

In [14]:

```
from sklearn.preprocessing import StandardScaler
```

In [15]:

```
sc = StandardScaler()
```

In [16]:

```
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

'''
This fit_transform() method is basically the combination of fit method and transform method
it is equivalent to fit(). transform().
This method performs fit and transform on the ===input=== data at a single time and convert
```

Out[16]:

```
'\nThis fit_transform() method is basically the combination of fit method an
d transform method, \nit is equivalent to fit(). transform(). \nThis method
performs fit and transform on the ===input=== data at a single time and conv
erts the data points'
```

In [17]:

```
from sklearn.linear_model import LogisticRegression
```

In [18]:

```
logmodel = LogisticRegression()
```

In [19]:

```
logmodel.fit(X_train,y_train)
```

Out[19]:

```
LogisticRegression()
```

In [20]:

```
y_pred = logmodel.predict(X_test)
```

In [21]:

```
#we can also check accuracy from confusion metrics but we have an inbuilt function in pytho
from sklearn.metrics import accuracy_score
```

In [23]:

```
accuracy_score(y_test,y_pred)*100
```

Out[23]:

```
89.0
```

In [24]:

```
# HERE MODEL GIVES US 89%ACCURACY
```