In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import warnings
warnings.filterwarnings("ignore")
#importing the necessary libraries
```

In [2]:
```
data=pd.read_csv("advertising.csv")
#Data
```

In [3]:
```
data.head()
#displaying the starting five records
```

Out[3]:

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Ad Topic Line	City	Male	Country	Timestamp	Clicked on Ad
0	68.95	35	61833.90	256.09	Cloned 5thgeneration orchestration	Wrightburgh	0	Tunisia	2016-03-27 00:53:11	0
1	80.23	31	68441.85	193.77	Monitored national standardization	West Jodi	1	Nauru	2016-04-04 01:39:02	0
2	69.47	26	59785.94	236.50	Organic bottom-line service-desk	Davidton	0	San Marino	2016-03-13 20:35:42	0
3	74.15	29	54806.18	245.89	Triple-buffered reciprocal time-frame	West Terrifurt	1	Italy	2016-01-10 02:31:19	0
4	68.37	35	73889.99	225.58	Robust logistical utilization	South Manuel	0	Iceland	2016-06-03 03:36:18	0

In [4]:
```
data.tail()

#displaying the last five records
```

Out[4]:

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Ad Topic Line	City	Male	Country	Timestamp	Clicked on Ad
995	72.97	30	71384.57	208.58	Fundamental modular algorithm	Duffystad	1	Lebanon	2016-02-11 21:49:00	1
996	51.30	45	67782.17	134.42	Grass-roots cohesive monitoring	New Darlene	1	Bosnia and Herzegovina	2016-04-22 02:07:01	1
997	51.63	51	42415.72	120.37	Expanded intangible solution	South Jessica	1	Mongolia	2016-02-01 17:24:57	1
998	55.55	19	41920.79	187.95	Proactive bandwidth-monitored policy	West Steven	0	Guatemala	2016-03-24 02:35:54	0
999	45.01	26	29875.80	178.35	Virtual 5thgeneration emulation	Ronniemouth	0	Brazil	2016-06-03 21:43:21	1

In [5]:
```
data.dtypes
#data type of each column index
```

Out[5]:
```
Daily Time Spent on Site    float64
Age                           int64
Area Income                 float64
Daily Internet Usage        float64
Ad Topic Line                object
City                         object
Male                          int64
Country                      object
Timestamp                    object
Clicked on Ad                 int64
dtype: object
```

In [6]:
```
data.info()
#infornatio about the column
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 10 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Daily Time Spent on Site  1000 non-null   float64
 1   Age                       1000 non-null   int64  
 2   Area Income               1000 non-null   float64
 3   Daily Internet Usage      1000 non-null   float64
 4   Ad Topic Line             1000 non-null   object 
 5   City                      1000 non-null   object 
 6   Male                      1000 non-null   int64  
 7   Country                   1000 non-null   object 
 8   Timestamp                 1000 non-null   object 
 9   Clicked on Ad             1000 non-null   int64  
dtypes: float64(3), int64(3), object(4)
memory usage: 78.2+ KB
```

In [7]:
```
data.describe()
#statistical infrmation about each column indx
```

Out[7]:

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Male	Clicked on Ad
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000
mean	65.000200	36.009000	55000.000080	180.000100	0.481000	0.50000
std	15.853615	8.785562	13414.634022	43.902339	0.499889	0.50025
min	32.600000	19.000000	13996.500000	104.780000	0.000000	0.00000
25%	51.360000	29.000000	47031.802500	138.830000	0.000000	0.00000
50%	68.215000	35.000000	57012.300000	183.130000	0.000000	0.50000
75%	78.547500	42.000000	65470.635000	218.792500	1.000000	1.00000
max	91.430000	61.000000	79484.800000	269.960000	1.000000	1.00000

In [8]:
```
data.shape
#no of rows and column in th data
```

Out[8]:
```
(1000, 10)
```

In [9]:
```
data.isnull().sum()
#number of nukkk values in each column index
```

Out[9]:
```
Daily Time Spent on Site    0
Age                         0
Area Income                 0
Daily Internet Usage        0
Ad Topic Line               0
City                        0
Male                        0
Country                     0
Timestamp                   0
Clicked on Ad               0
dtype: int64
```

In [10]:
```
data.duplicated().sum()
#Duplicate data
```

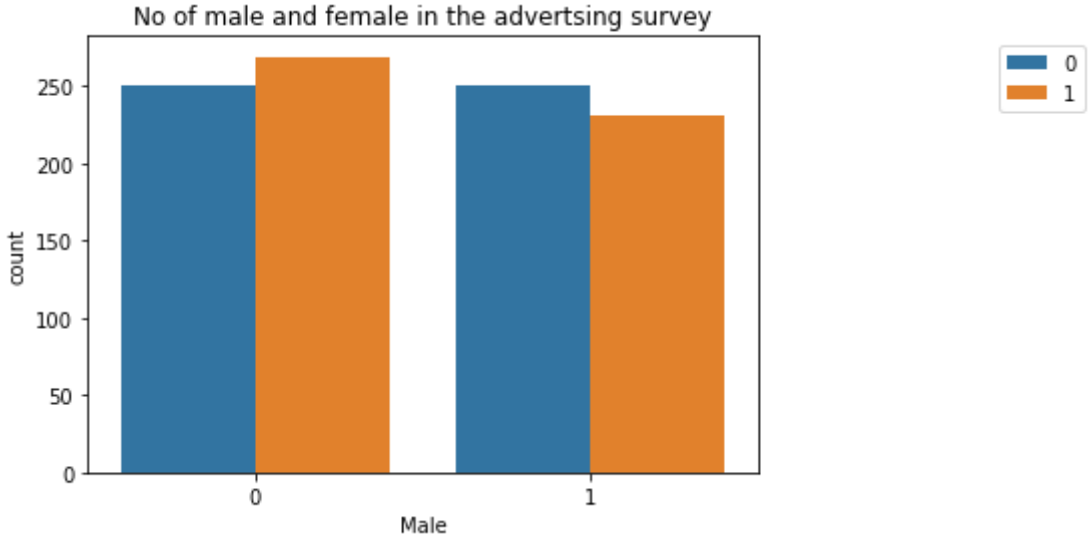
Out[10]:
```
0
```

In [11]:
```
data.columns
#Column index
```

Out[11]:
```
Index(['Daily Time Spent on Site', 'Age', 'Area Income',
       'Daily Internet Usage', 'Ad Topic Line', 'City', 'Male', 'Country',
       'Timestamp', 'Clicked on Ad'],
      dtype='object')
```

In [12]:
```
sns.countplot(x='Male',data=data,hue="Clicked on Ad")
plt.title("No of male and female in the advertsing survey")
plt.legend(bbox_to_anchor=(1.5,1))
#This is used to place the legend outside the plot
#To count number of male and feamale who clicked the add
```

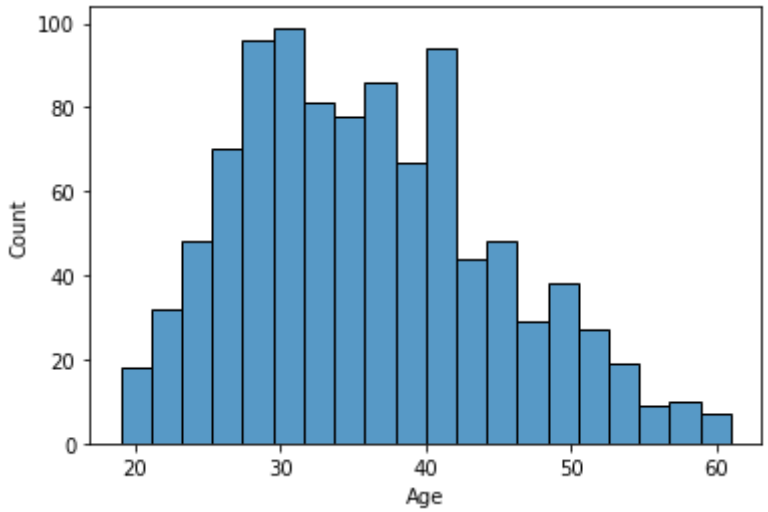
Out[12]: <matplotlib.legend.Legend at 0x2a637d2ff40>



in this set of data 1 represents the number of male and 0 represents the number of female. The counplot is used to visulaise the count of a particular column in the data . There are more numer of females that clicked the advertisement as compared to men

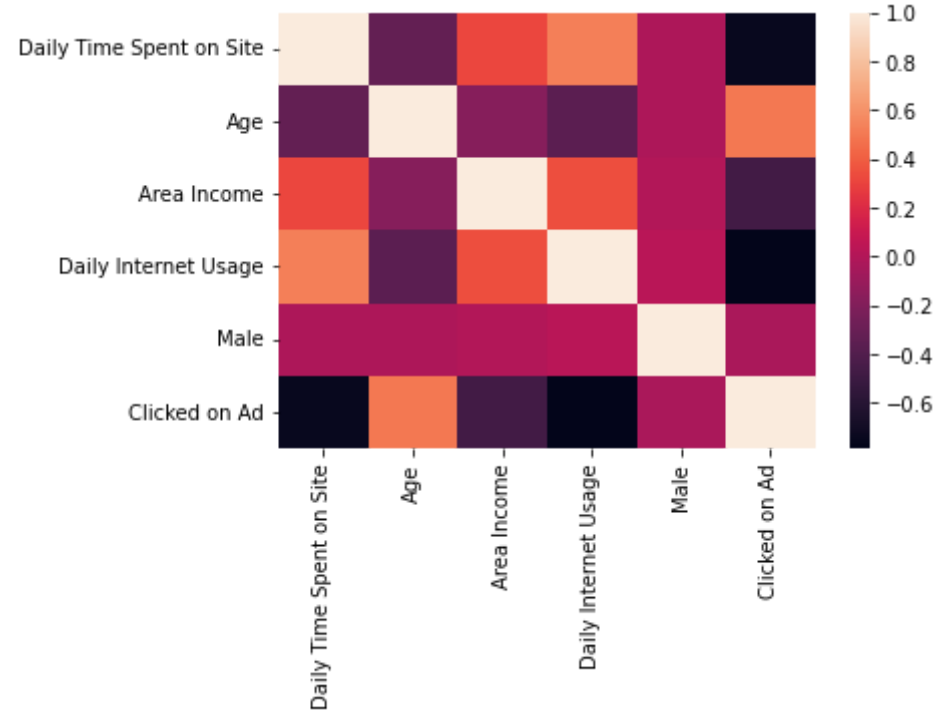
In [13]:
```
sns.histplot(data=data,x="Age",bins=20)
#To understand the distribution of the age
```

Out[13]: <AxesSubplot:xlabel='Age', ylabel='Count'>




```
Histograms are used to display the datain the continous period.
It is used in continous data.With the help of this graph we can undertstand.
With the help of this we can find out the mean ,median and mode
With the help of this visulaisation we can calculate the skewness of the data.
There is the random distribution of data as they have many classes .
the normal distribution is bell shaped and is symetrical
```

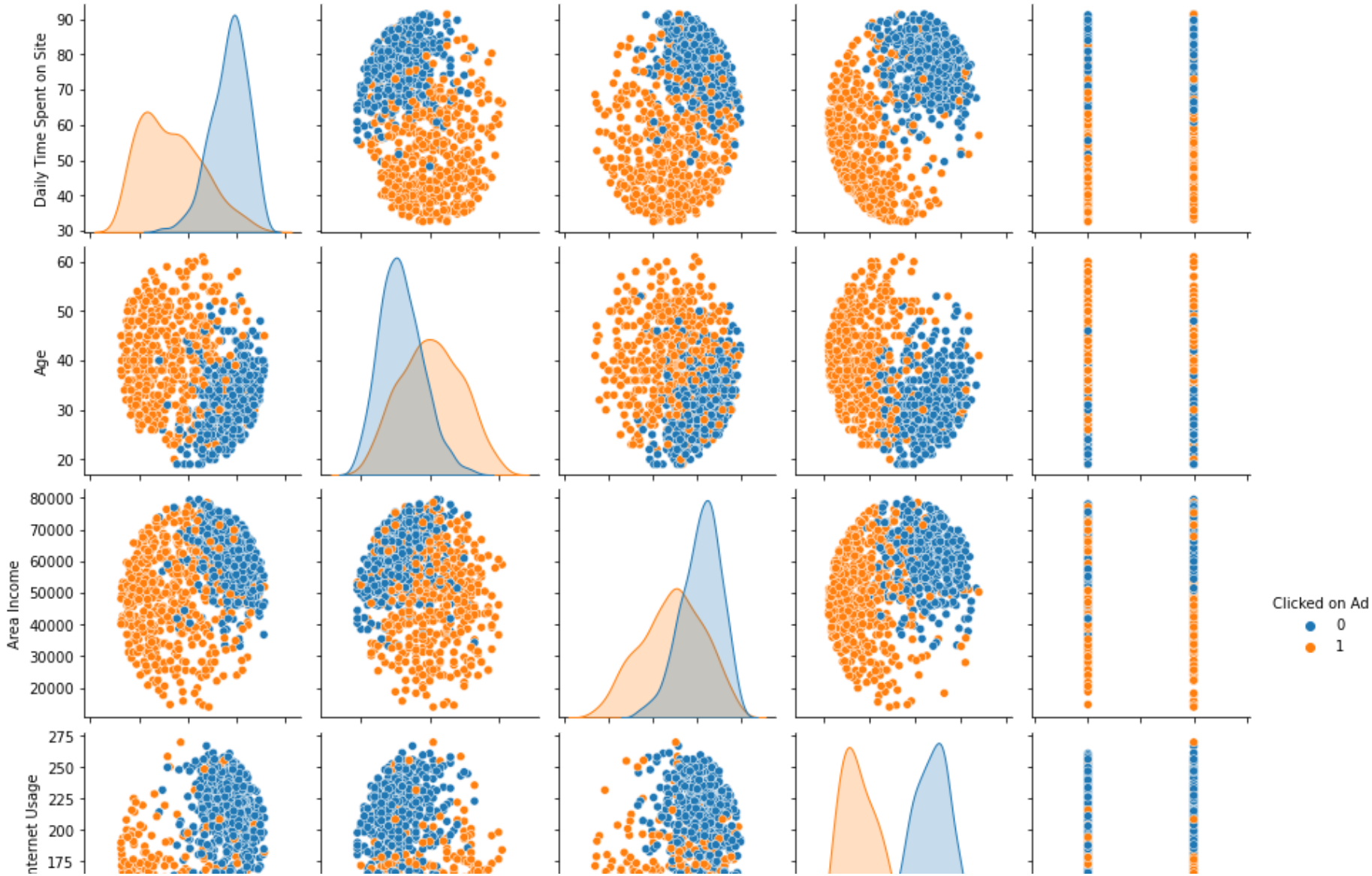
In [14]:
```
sns.heatmap(data.corr());
#To display the correlation of the data
```

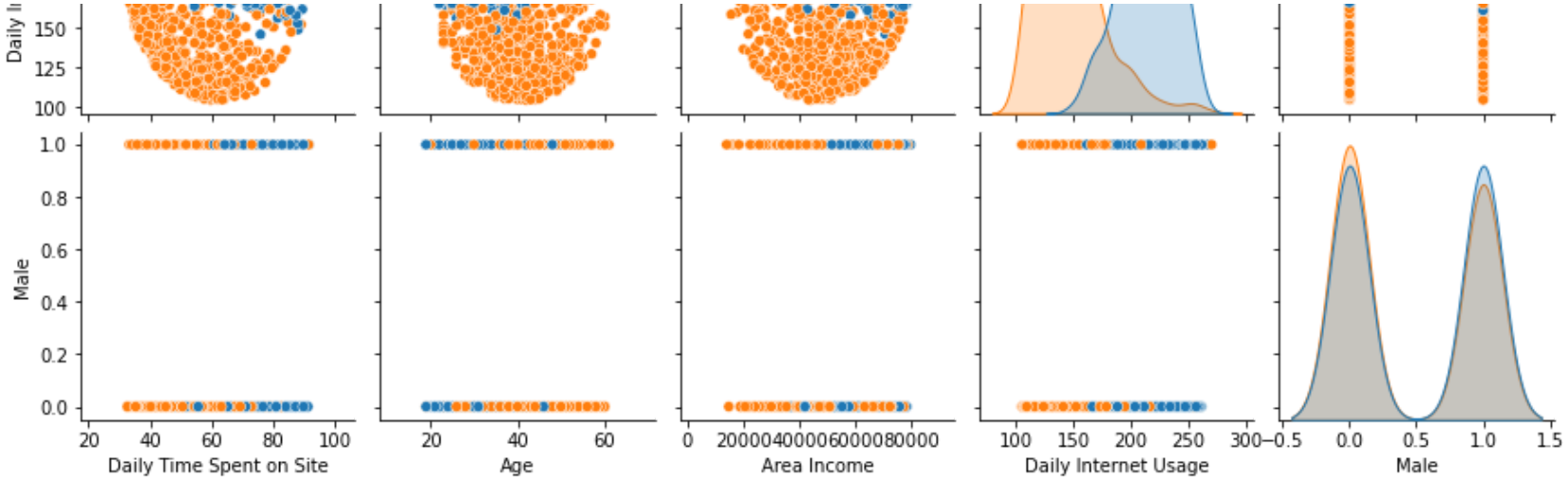


Correlation describes the relation betwween the two variables. There can be three types of coorlation: Positive correlation: if one increases,the other also inxreases negative correlation:if one increases the other decreases no correlation:The both variables have no effect on each other

In [15]:
```
sns.pairplot(data, hue ='Clicked on Ad')
# to show


plt.show()
#To understand the relationship of clicking the ad with each column index
```





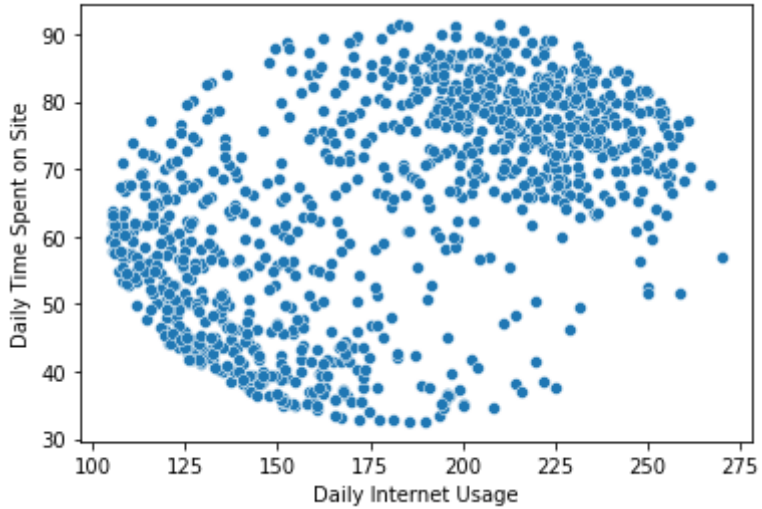
Pairplot is used to understand the relationship between the bivariate variables.

In [16]:
```
data.columns
```

Out[16]:
```
Index(['Daily Time Spent on Site', 'Age', 'Area Income',
       'Daily Internet Usage', 'Ad Topic Line', 'City', 'Male', 'Country',
       'Timestamp', 'Clicked on Ad'],
      dtype='object')
```

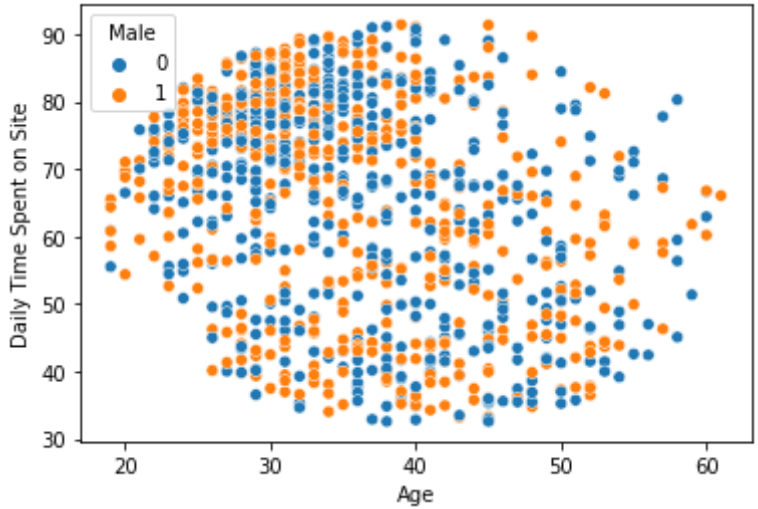
In [17]:
```
sns.scatterplot(x="Daily Internet Usage",y='Daily Time Spent on Site',data=data)
#As the daily internet usage is incresed then the daily time on the internet also increases
```

Out[17]:
```
<AxesSubplot:xlabel='Daily Internet Usage', ylabel='Daily Time Spent on Site'>
```



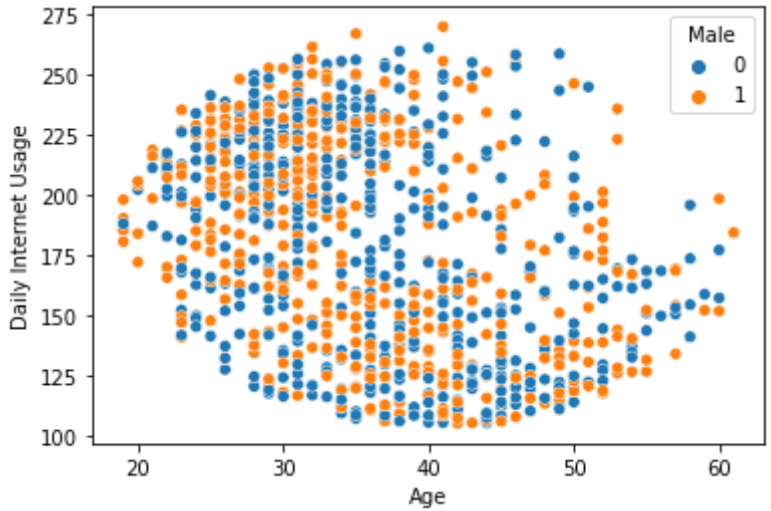
In [18]:
```
sns.scatterplot(x="Age",y='Daily Time Spent on Site',data=data,hue="Male")
#The age group between 30-50 spent much time on the site
```

Out[18]: <AxesSubplot:xlabel='Age', ylabel='Daily Time Spent on Site'>



In [19]:
```
sns.scatterplot(x="Age",y='Daily Internet Usage',data=data,hue="Male")
#To understand the relationship between age and internet usage with the no of male and female
```

Out[19]: <AxesSubplot:xlabel='Age', ylabel='Daily Internet Usage'>



In [20]:
```
data['Date'] = pd.to_datetime(data['Timestamp'])
# We have converted the data type of timestamp that was object earlier to Date
```

In [21]:
```
data.dtypes
```

Out[21]:
```
Daily Time Spent on Site          float64
Age                                 int64
Area Income                       float64
Daily Internet Usage              float64
Ad Topic Line                      object
City                               object
Male                                int64
Country                            object
Timestamp                          object
Clicked on Ad                       int64
Date                       datetime64[ns]
dtype: object
```

In [22]:
```
data['month'] = pd.DatetimeIndex(data['Date']).month
#In this we need to have the month as the year is same throughout the analysis
```

In [23]:
```
data.head()
```

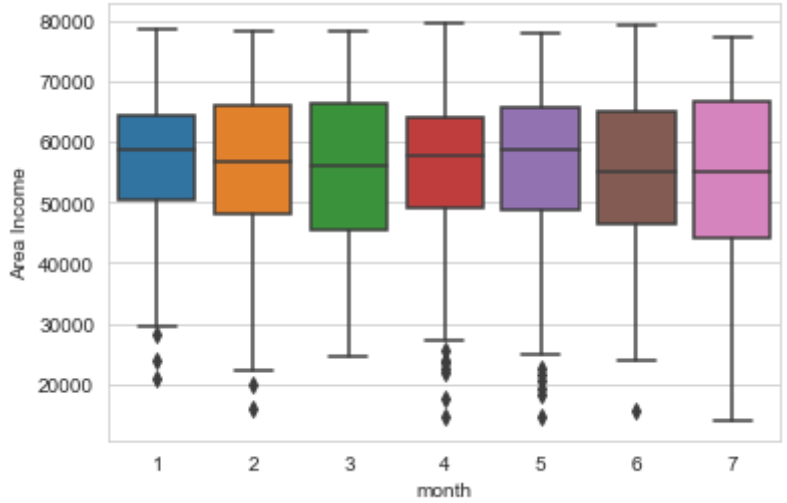
Out[23]:

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Ad Topic Line	City	Male	Country	Timestamp	Clicked on Ad	Date	month
0	68.95	35	61833.90	256.09	Cloned 5thgeneration orchestration	Wrightburgh	0	Tunisia	2016-03-27 00:53:11	0	2016-03-27 00:53:11	3
1	80.23	31	68441.85	193.77	Monitored national standardization	West Jodi	1	Nauru	2016-04-04 01:39:02	0	2016-04-04 01:39:02	4
2	69.47	26	59785.94	236.50	Organic bottom-line service-desk	Davidton	0	San Marino	2016-03-13 20:35:42	0	2016-03-13 20:35:42	3
3	74.15	29	54806.18	245.89	Triple-buffered reciprocal time-frame	West Terrifurt	1	Italy	2016-01-10 02:31:19	0	2016-01-10 02:31:19	1
4	68.37	35	73889.99	225.58	Robust logistical utilization	South Manuel	0	Iceland	2016-06-03 03:36:18	0	2016-06-03 03:36:18	6

In [24]:
```
sns.set_style("whitegrid")

sns.boxplot(x = 'month', y = 'Area Income', data = data)
#To understand the statistical information
```

Out[24]: <AxesSubplot:xlabel='month', ylabel='Area Income'>

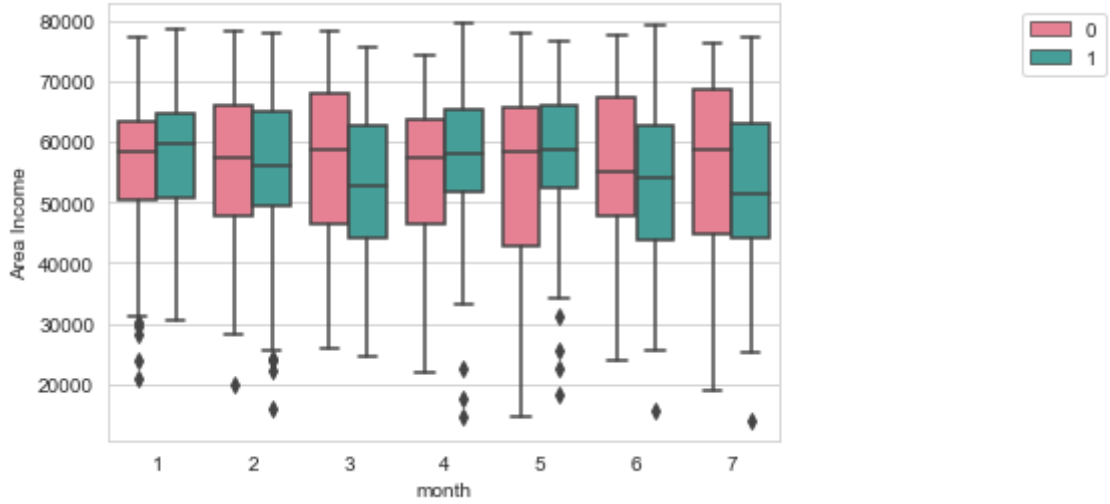



```
Box plot is used for the statistical representation of the data in the form of visualisation.
It is used to reprent the minimuim value,maximuim value, 25% percentile, %0th percentile and 75th percemtile .
The black
```

In [25]:
```
sns.set_style("whitegrid")

sns.boxplot(x = 'month', y = 'Area Income', data = data,hue="Male", palette = 'husl')
plt.legend(bbox_to_anchor=(1.5,1))
#To understand the statistical information with the help of box plot
```

Out[25]: <matplotlib.legend.Legend at 0x2a63a475370>



In [26]:
```
data["Country"].value_counts()
#To know the country wise distribution
```

Out[26]:
```
France                                                9
Czech Republic                                        9
Peru                                                  8
Turkey                                                8
Greece                                                8
                                                     ..
Romania                                               1
British Indian Ocean Territory (Chagos Archipelago)   1
Germany                                               1
Aruba                                                 1
Lesotho                                               1
Name: Country, Length: 237, dtype: int64
```

In [27]:
```
country_wise = data["Country"].value_counts().reset_index()
country_wise.columns = ['country', 'No of people']
country_wise
```

Out[27]:

	country	No of people
0	France	9
1	Czech Republic	9
2	Peru	8
3	Turkey	8
4	Greece	8
...	...	...
232	Romania	1
233	British Indian Ocean Territory (Chagos Archipe...	1
234	Germany	1
235	Aruba	1
236	Lesotho	1

237 rows × 2 columns

In [28]:
```
data.groupby(['Country','City']).count()
# To know country wise and city wise
```

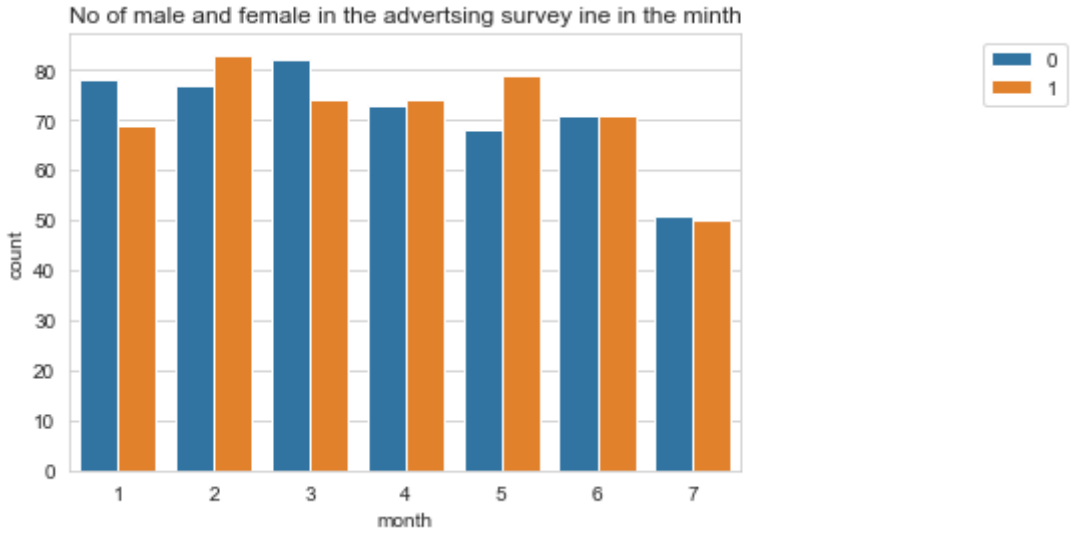
Out[28]:

Country	City	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Ad Topic Line	Male	Timestamp	Clicked on Ad	Date	month
Afghanistan	Christinetown	1	1	1	1	1	1	1	1	1	1
	East Anthony	1	1	1	1	1	1	1	1	1	1
	Kaylashire	1	1	1	1	1	1	1	1	1	1
	Kevinberg	1	1	1	1	1	1	1	1	1	1
	North Debrashire	1	1	1	1	1	1	1	1	1	1
...	...	...	...	...	...	...	...	...	...	...	...
Zimbabwe	Juanport	1	1	1	1	1	1	1	1	1	1
	Lake John	1	1	1	1	1	1	1	1	1	1
	North Samantha	1	1	1	1	1	1	1	1	1	1
	Port Brian	1	1	1	1	1	1	1	1	1	1
	West Gregburgh	1	1	1	1	1	1	1	1	1	1

1000 rows × 10 columns

In [29]:
```
sns.countplot(x='month',data=data,hue="Clicked on Ad")
plt.title("No of male and female in the advertsing survey ine in the minth")
plt.legend(bbox_to_anchor=(1.5,1))
```

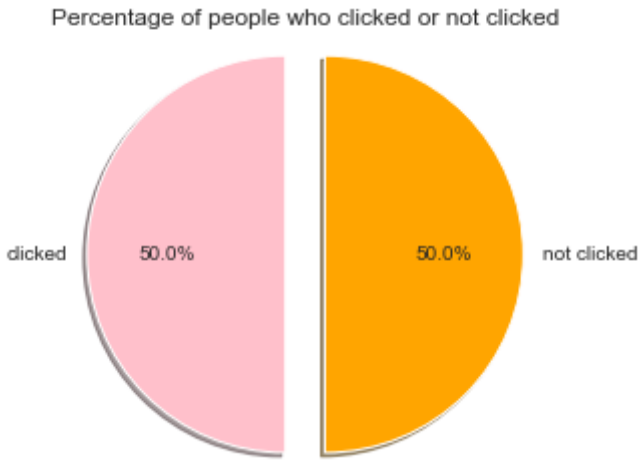
Out[29]:
```
<matplotlib.legend.Legend at 0x2a63a3cf940>
```



In [33]:
```
clicked=data["Clicked on Ad"].value_counts()
clicked
```

Out[33]:
```
0    500
1    500
Name: Clicked on Ad, dtype: int64
```

In [34]:
```
colors = ["pink", "orange"]
explode = (0.1, 0.1)
lables=["clicked","not clicked"]
plt.pie(clicked, explode=explode, colors=colors,
autopct='%1.1f%%', shadow=True, startangle=90,labels=lables)
plt.title("Percentage of people who clicked or not clicked")
plt.axis("equal")
plt.show()
```



In [ ]: