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

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, roc_curve, roc_auc_score
```

In [2]:
```
retail = pd.read_csv("Retail_Customer_Insights.csv")
retail.head()
```

Out[2]:

	Customer_ID	Age	Annual_Income	Gender	Purchase_History	Product_Category	Customer_Satisfaction	Loyalty_Points	Marital
0	CID770487	45	72633.53	Non-binary	0	Electronics	9.0	541.11	
1	CID216739	38	61816.55	Non-binary	0	Books	6.0	497.41	
2	CID126225	47	57338.15	Non-binary	0	Grocery	3.0	634.90	
3	CID877572	58	83800.37	Female	0	Furniture	4.0	505.82	
4	CID388389	37	64875.12	Male	0	Furniture	6.0	610.39	

In [3]:
```
retail.isnull().sum()
```

Out[3]:
```
Customer_ID                      0
Age                              0
Annual_Income                 5000
Gender                           0
Purchase_History                 0
Product_Category                 0
Customer_Satisfaction         3000
Loyalty_Points                2000
Marital_Status                   0
Number_of_Children               0
Employment_Status                0
Credit_Score                     0
Owns_House                       0
Monthly_Expenditure           5000
Internet_Usage_Hours_per_Week    0
dtype: int64
```

In [4]:
```
retail.isna().sum()
```

Out[4]:
```
Customer_ID                      0
Age                              0
Annual_Income                 5000
Gender                           0
Purchase_History                 0
Product_Category                 0
Customer_Satisfaction         3000
Loyalty_Points                2000
Marital_Status                   0
Number_of_Children               0
Employment_Status                0
Credit_Score                     0
Owns_House                       0
Monthly_Expenditure           5000
Internet_Usage_Hours_per_Week    0
dtype: int64
```

In [5]:
```
retail.fillna({'Annual_Income':retail['Annual_Income'].median(),
                       'Customer_Satisfaction':retail['Customer_Satisfaction'].median(),
                       'Loyalty_Points':retail['Loyalty_Points'].median(),
                       'Monthly_Expenditure':retail['Monthly_Expenditure'].median()},inplace=True)
```

In [6]:
```
retail.isnull().sum()
```

Out[6]:
```
Customer_ID                     0
Age                             0
Annual_Income                   0
Gender                          0
Purchase_History                0
Product_Category                0
Customer_Satisfaction           0
Loyalty_Points                  0
Marital_Status                  0
Number_of_Children              0
Employment_Status               0
Credit_Score                    0
Owns_House                      0
Monthly_Expenditure             0
Internet_Usage_Hours_per_Week   0
dtype: int64
```

In [7]:
```
retail.isna().sum()
```

Out[7]:
```
Customer_ID                     0
Age                             0
Annual_Income                   0
Gender                          0
Purchase_History                0
Product_Category                0
Customer_Satisfaction           0
Loyalty_Points                  0
Marital_Status                  0
Number_of_Children              0
Employment_Status               0
Credit_Score                    0
Owns_House                      0
Monthly_Expenditure             0
Internet_Usage_Hours_per_Week   0
dtype: int64
```

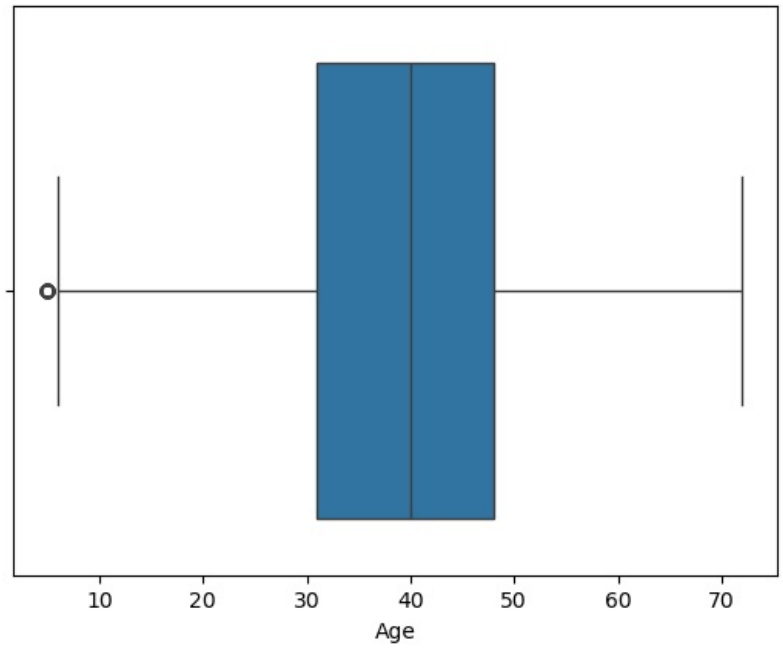
In [8]:
```
# Removing outliers from Age column
dataset=np.array(retail['Age']).tolist()
dataset.sort()
median_= np.median(dataset)
q1=np.percentile(dataset,25)
q3=np.percentile(dataset,75)
iqr=q3-q1
ll=q1-1.5*iqr
ul=q3+1.5*iqr
data=retail.loc[(retail['Age']>ll) & (retail['Age']<ul)]
```

In [9]:
```
print(f'LL:{ll} UL:{ul}')
```

```
LL:4.5 UL:72.5
```

In [10]:
```
sns.boxplot(data['Age'],orient='h')
```

Out[10]:
```
<Axes: xlabel='Age'>
```

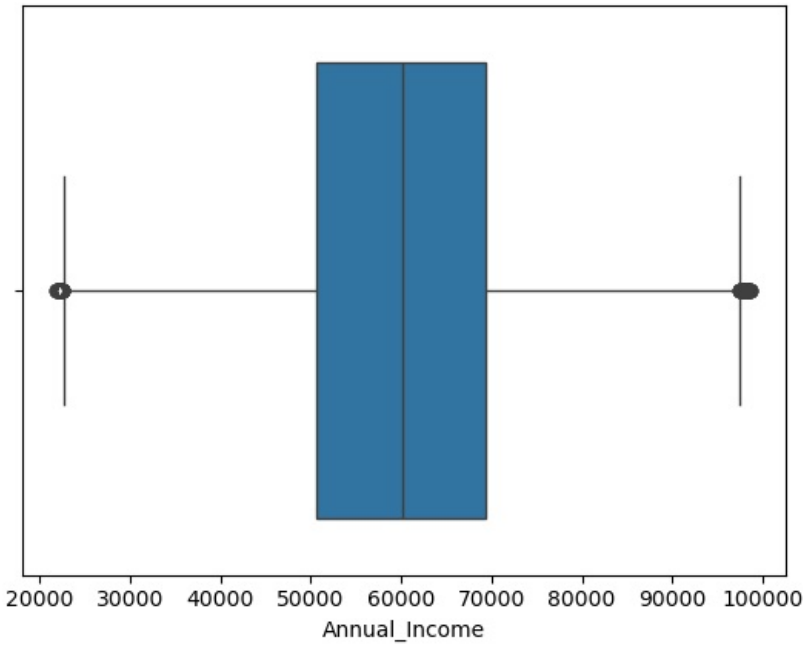


In [11]:
```
# Removing outliers from Annual_Income column
dataset=np.array(data['Annual_Income']).tolist()
```

```
dataset.sort()
median_= np.median(dataset)
q1=np.percentile(dataset,25)
q3=np.percentile(dataset,75)
iqr=q3-q1
ll=q1-1.5*iqr
ul=q3+1.5*iqr
data=data.loc[(data['Annual_Income']>ll) & (data['Annual_Income']<ul)]
```

In [12]:
```
sns.boxplot(data['Annual_Income'],orient='h')
```

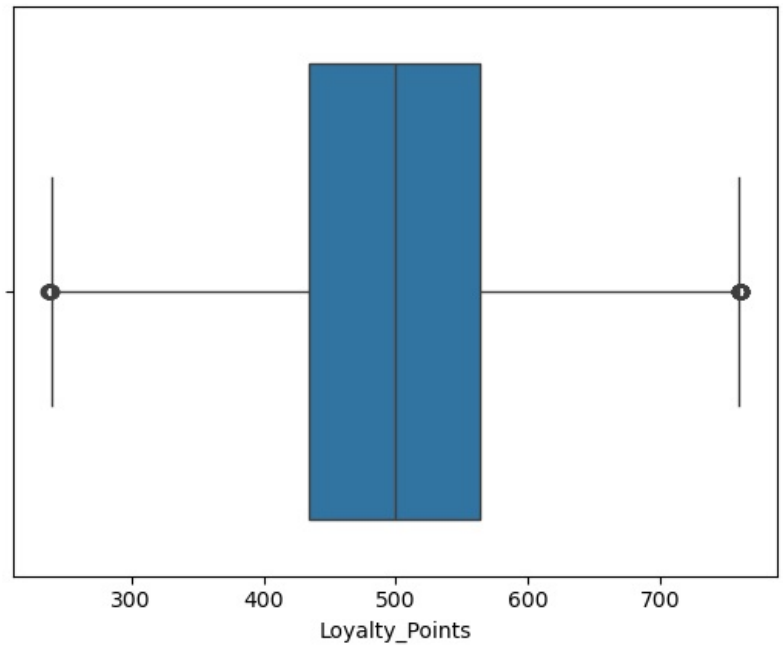
Out[12]: <Axes: xlabel='Annual_Income'>



In [13]:
```
# Removing outliers from Loyalty_Points column
dataset=np.array(data['Loyalty_Points']).tolist()
dataset.sort()
median_= np.median(dataset)
q1=np.percentile(dataset,25)
q3=np.percentile(dataset,75)
iqr=q3-q1
ll=q1-1.5*iqr
ul=q3+1.5*iqr
data=data.loc[(data['Loyalty_Points']>ll) & (data['Loyalty_Points']<ul)]
```

In [14]:
```
sns.boxplot(data['Loyalty_Points'],orient='h')
```

Out[14]: <Axes: xlabel='Loyalty_Points'>

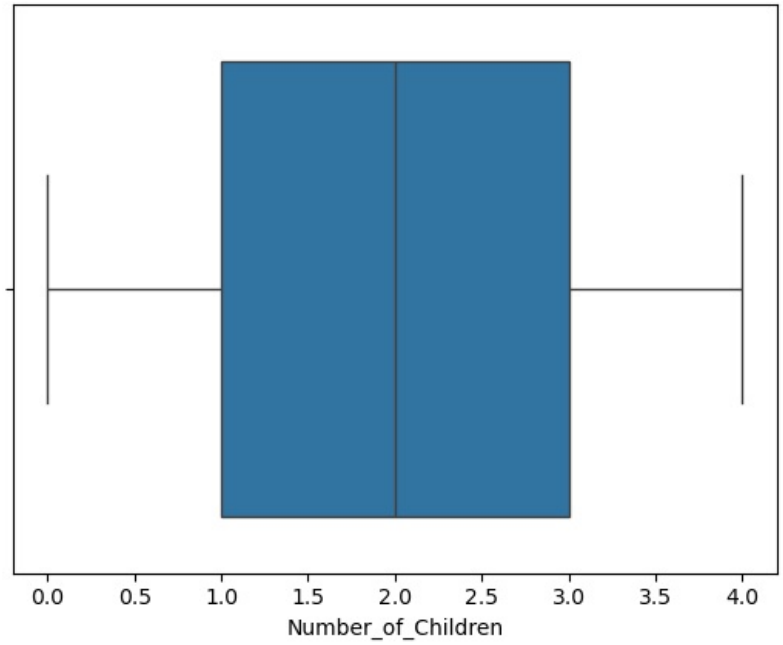


In [15]:
```
# Removing outliers from Number_of_Children column
dataset=np.array(data['Number_of_Children']).tolist()
dataset.sort()
```

```
median_= np.median(dataset)
q1=np.percentile(dataset,25)
q3=np.percentile(dataset,75)
iqr=q3-q1
ll=q1-1.5*iqr
ul=q3+1.5*iqr
data=data.loc[(data['Number_of_Children']>ll) & (data['Number_of_Children']<ul)]
```

In [16]:
```
sns.boxplot(data['Number_of_Children'],orient='h')
```

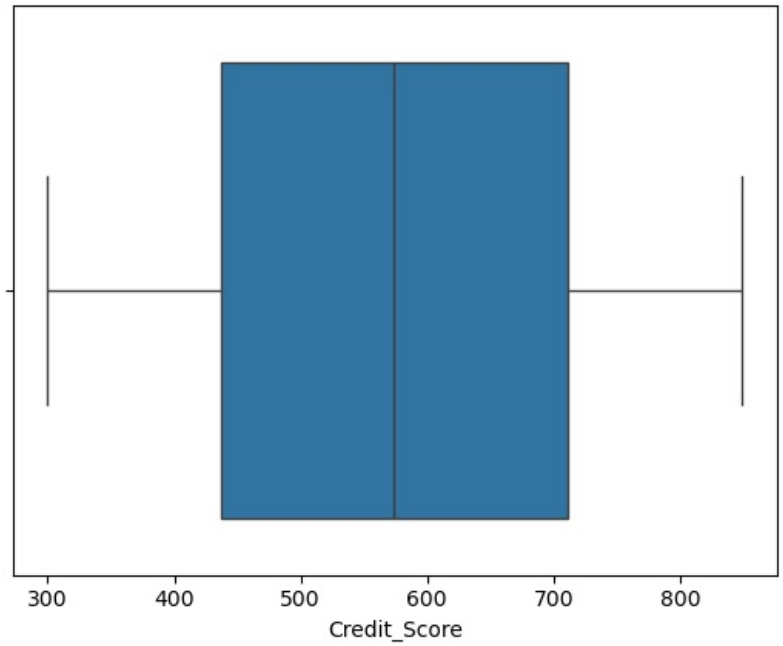
Out[16]: <Axes: xlabel='Number_of_Children'>



In [17]:
```
# Removing outliers from Credit_Score column
dataset=np.array(data['Credit_Score']).tolist()
dataset.sort()
median_= np.median(dataset)
q1=np.percentile(dataset,25)
q3=np.percentile(dataset,75)
iqr=q3-q1
ll=q1-1.5*iqr
ul=q3+1.5*iqr
data=data.loc[(retail['Credit_Score']>ll) & (data['Credit_Score']<ul)]
```

In [18]:
```
sns.boxplot(data['Credit_Score'],orient='h')
```

Out[18]: <Axes: xlabel='Credit_Score'>

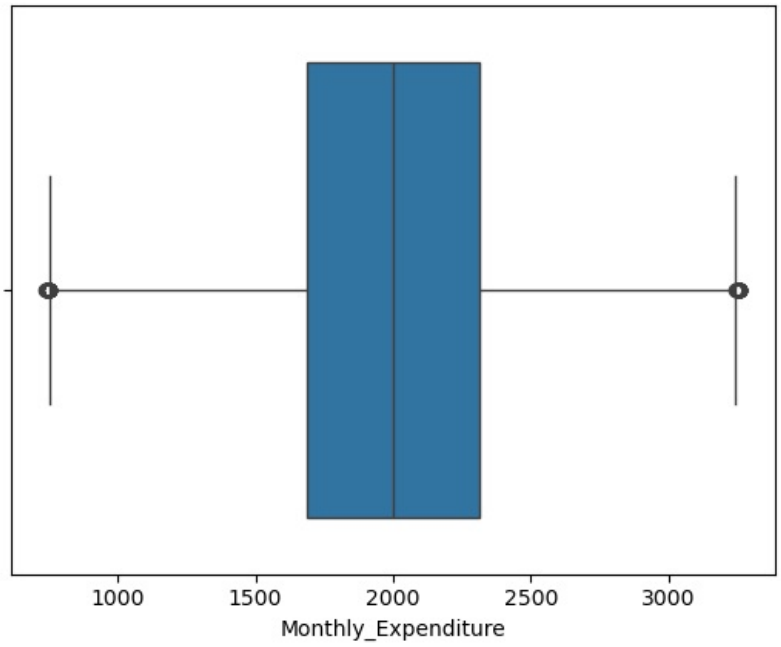


In [19]:
```
# Removing outliers from Number_of_Children column
dataset=np.array(data['Monthly_Expenditure']).tolist()
dataset.sort()
median_= np.median(dataset)
```

```
q1=np.percentile(dataset,25)
q3=np.percentile(dataset,75)
iqr=q3-q1
ll=q1-1.5*iqr
ul=q3+1.5*iqr
data=data.loc[(data['Monthly_Expenditure']>ll) & (data['Monthly_Expenditure']<ul)]
```

In [20]:
```
sns.boxplot(data['Monthly_Expenditure'],orient='h')
```

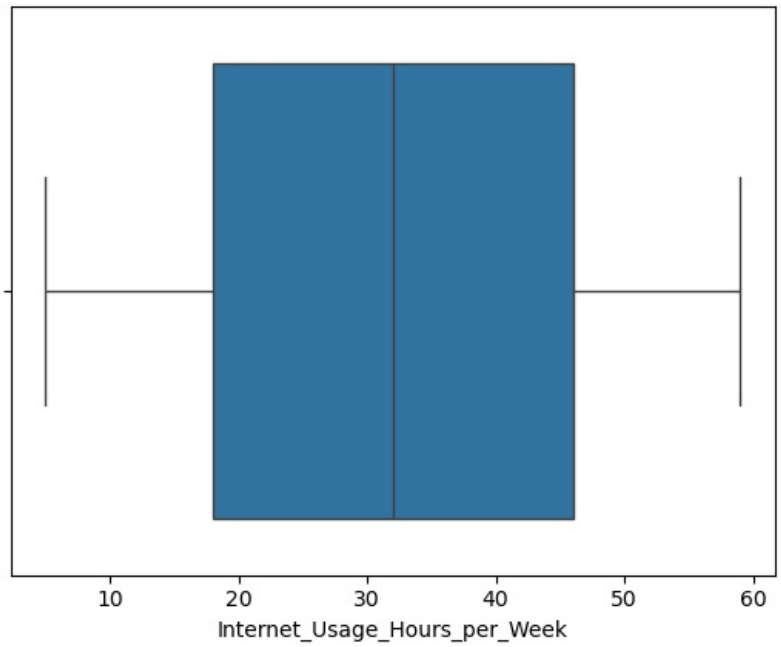
Out[20]: <Axes: xlabel='Monthly_Expenditure'>



In [21]:
```
# Removing outliers from Internet_Usage_Hours_per_Week column
dataset=np.array(data['Internet_Usage_Hours_per_Week']).tolist()
dataset.sort()
median_= np.median(dataset)
q1=np.percentile(dataset,25)
q3=np.percentile(dataset,75)
iqr=q3-q1
ll=q1-1.5*iqr
ul=q3+1.5*iqr
data=data.loc[(data['Internet_Usage_Hours_per_Week']>ll) & (data['Internet_Usage_Hours_per_Week']<ul)]
```

In [22]:
```
sns.boxplot(data['Internet_Usage_Hours_per_Week'],orient='h')
```

Out[22]: <Axes: xlabel='Internet_Usage_Hours_per_Week'>



In [23]:
```
# setting customer_id as index
data.set_index('Customer_ID',inplace=True)
data.head()
```

Out[23]:

Customer_ID	Age	Annual_Income	Gender	Purchase_History	Product_Category	Customer_Satisfaction	Loyalty_Points	Marital_St
CID770487	45	72633.53	Non-binary	0	Electronics	9.0	541.11	Divc
CID216739	38	61816.55	Non-binary	0	Books	6.0	497.41	Ma
CID126225	47	57338.15	Non-binary	0	Grocery	3.0	634.90	S
CID877572	58	83800.37	Female	0	Furniture	4.0	505.82	Divc
CID356787	37	57270.25	Prefer not to say	1	Grocery	3.0	458.98	Divc

In [24]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 81315 entries, CID770487 to CID966793
Data columns (total 14 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Age                           81315 non-null  int64  
 1   Annual_Income                 81315 non-null  float64
 2   Gender                        81315 non-null  object 
 3   Purchase_History              81315 non-null  int64  
 4   Product_Category              81315 non-null  object 
 5   Customer_Satisfaction         81315 non-null  float64
 6   Loyalty_Points                81315 non-null  float64
 7   Marital_Status                81315 non-null  object 
 8   Number_of_Children            81315 non-null  int64  
 9   Employment_Status             81315 non-null  object 
 10  Credit_Score                  81315 non-null  int64  
 11  Owns_House                    81315 non-null  bool   
 12  Monthly_Expenditure           81315 non-null  float64
 13  Internet_Usage_Hours_per_Week  81315 non-null  int64  
dtypes: bool(1), float64(4), int64(5), object(4)
memory usage: 8.8+ MB
```

In [25]:
```
#Data Preprocessing
data = pd.get_dummies(data,columns=['Gender','Product_Category','Marital_Status','Employment_Status'],drop_firs
data.head()
```

Out[25]:

Customer_ID	Age	Annual_Income	Purchase_History	Customer_Satisfaction	Loyalty_Points	Number_of_Children	Credit_Score	O
CID770487	45	72633.53	0	9.0	541.11	1	664	
CID216739	38	61816.55	0	6.0	497.41	3	623	
CID126225	47	57338.15	0	3.0	634.90	1	342	
CID877572	58	83800.37	0	4.0	505.82	2	810	
CID356787	37	57270.25	1	3.0	458.98	4	844	

5 rows × 23 columns

In [26]:
```
data = pd.get_dummies(data,columns=['Owns_House'],drop_first=True,dtype=int)
data.head()
```

Out[26]:

Customer_ID	Age	Annual_Income	Purchase_History	Customer_Satisfaction	Loyalty_Points	Number_of_Children	Credit_Score	M
CID770487	45	72633.53	0	9.0	541.11	1	664	
CID216739	38	61816.55	0	6.0	497.41	3	623	
CID126225	47	57338.15	0	3.0	634.90	1	342	
CID877572	58	83800.37	0	4.0	505.82	2	810	
CID356787	37	57270.25	1	3.0	458.98	4	844	

5 rows × 23 columns

In [27]:
```
data.columns
```

Out[27]:
```
Index(['Age', 'Annual_Income', 'Purchase_History', 'Customer_Satisfaction',
       'Loyalty_Points', 'Number_of_Children', 'Credit_Score',
       'Monthly_Expenditure', 'Internet_Usage_Hours_per_Week', 'Gender_Male',
       'Gender_Non-binary', 'Gender_Prefer not to say',
       'Product_Category_Clothing', 'Product_Category_Electronics',
       'Product_Category_Furniture', 'Product_Category_Grocery',
       'Marital_Status_Married', 'Marital_Status_Single',
       'Marital_Status_Widowed', 'Employment_Status_Retired',
       'Employment_Status_Student', 'Employment_Status_Unemployed',
       'Owns_House_True'],
      dtype='object')
```

In [28]:
```
data.duplicated().sum()
```

Out[28]:
```
0
```

In [29]:
```
#splitting data into independent/dependent varibles
x=data.loc[:,data.columns!='Purchase_History']
y=data.loc[:,data.columns=='Purchase_History']
```

In [30]:
```
#splitting data into train and test
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=75)
```

In [31]:
```
#Feature Scaling
scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

In [32]:
```
# Model Training
model = LogisticRegression()
model.fit(x_train_scaled,y_train.values.flatten(order='A'))
```

Out[32]:
▾ LogisticRegression

LogisticRegression()

In [33]:
```
#predict Training
y_train_pred = model.predict(x_train_scaled)
```

In [34]:
```
#predict Test
y_test_pred = model.predict(x_test_scaled)
```

In [35]:
```
# Train accuracy
train_accuracy = accuracy_score(y_train,y_train_pred)
train_accuracy
```

Out[35]:
```
0.5973221422861711
```

In [36]:
```
# Test accuracy
test_accuracy = accuracy_score(y_test,y_test_pred)
test_accuracy
```

Out[36]:
```
0.6046854823833241
```

In [37]:
```
# confusion matrix
cm = confusion_matrix(y_test,y_test_pred)
cm
```

Out[37]:
```
array([[9834,    0],
       [6429,    0]], dtype=int64)
```

In [38]:
```
data.shape
```

Out[38]:
```
(81315, 23)
```

In [ ]: