Churn Prediction Model

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.preprocessing import LabelEncoder
import joblib
```

In [3]:
```
# Read the data from the specified sheet into a pandas DataFrame
data = pd.read_excel(r"C:\Users\hp\Downloads\Churn Prediction Project\Prediction_Data.xlsx", sheet_name='vw_Chu
```

In [4]:
```
data.head(2)
```

Out[4]:

	Customer_ID	Gender	Age	Married	State	Number_of_Referrals	Tenure_in_Months	Value_Deal	Phone_Service	Multiple_Lin
0	11098-MAD	Female	30	Yes	Madhya Pradesh	0	31	Deal 1	Yes	I
1	11114-PUN	Male	51	No	Punjab	5	9	Deal 5	Yes	I

2 rows × 32 columns

In [5]:
```
# Drop columns that won't be used for prediction
data = data.drop(['Customer_ID', 'Churn_Category', 'Churn_Reason'], axis=1)
```

In [6]:
```
# List of columns to be label encoded
columns_to_encode = [
    'Gender', 'Married', 'State', 'Value_Deal', 'Phone_Service', 'Multiple_Lines',
    'Internet_Service', 'Internet_Type', 'Online_Security', 'Online_Backup',
    'Device_Protection_Plan', 'Premium_Support', 'Streaming_TV', 'Streaming_Movies',
    'Streaming_Music', 'Unlimited_Data', 'Contract', 'Paperless_Billing',
    'Payment_Method'
]
```

In [7]:
```
# Encode categorical variables except the target variable
label_encoders = {}
for column in columns_to_encode:
    label_encoders[column] = LabelEncoder()
    data[column] = label_encoders[column].fit_transform(data[column])
```

In [8]:
```
# Manually encode the target variable 'Customer_Status'
data['Customer_Status']= data['Customer_Status'].replace({'Stayed': 0, 'Churned': 1})
```

In [9]:
```
data['Customer_Status']
```

Out[9]:
```
0       0
1       1
2       0
3       0
4       0
       ..
6002    0
6003    0
6004    1
6005    0
6006    0
Name: Customer_Status, Length: 6007, dtype: int64
```

In [10]:
```
X = data.drop('Customer_Status', axis=1)
y = data['Customer_Status']
```

In [11]:
```
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [28]:
```
X_test.shape
```

Out[28]:
```
(1202, 28)
```

In [29]:
```
X_train.shape
```

Out[29]:
```
(4805, 28)
```

In [30]:
```
y_train.shape
```

Out[30]:
```
(4805,)
```

In [31]:
```
y_test.shape
```

Out[31]:
```
(1202,)
```

In [12]:
```
# Train Random Forest Model
# Initialize the Random Forest Classifier
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the model
rf_model.fit(X_train, y_train)
```

Out[12]:
```
▾        RandomForestClassifier
RandomForestClassifier(random_state=42)
```

In [13]:
```
# Make predictions
y_pred = rf_model.predict(X_test)
y_pred
```

Out[13]:
```
array([0, 0, 0, ..., 0, 0, 1], dtype=int64)
```

In [16]:
```
# Checking Model Score
print(f"rf_model Score is: {rf_model.score(X_test,y_test)}")
```

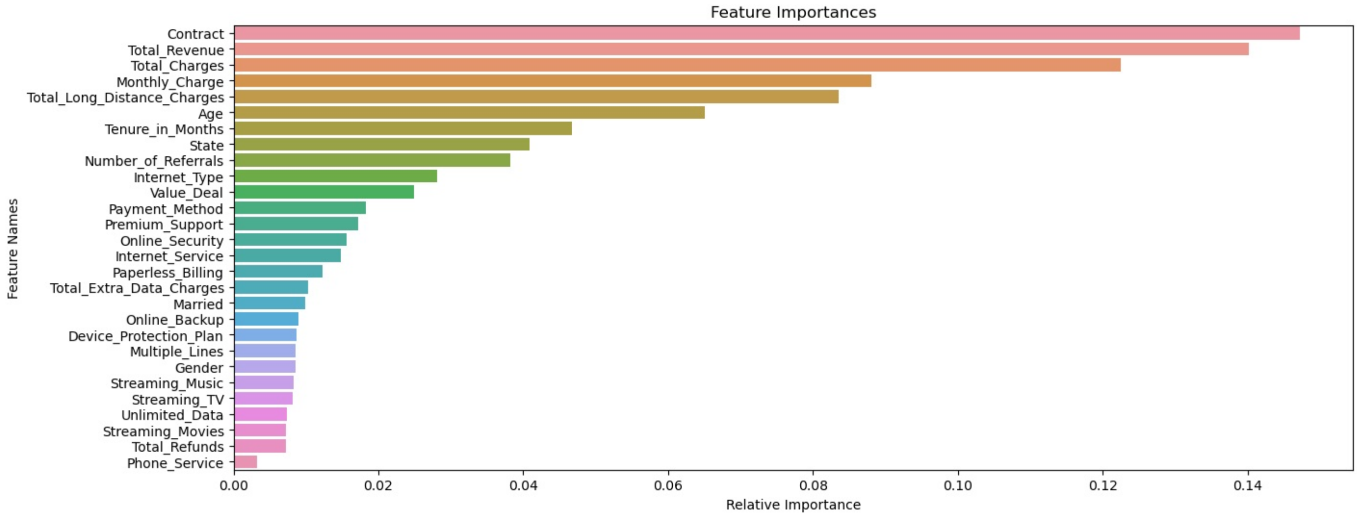
```
rf_model Score is: 0.8419301164725458
```

In [38]:
```
# Evaluate the model
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
Confusion Matrix:
[[783  64]
 [126 229]]

Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.92      0.89       847
           1       0.78      0.65      0.71       355

    accuracy                           0.84      1202
   macro avg       0.82      0.78      0.80      1202
weighted avg       0.84      0.84      0.84      1202
```

In [39]:
```
# Feature Selection using Feature Importance
importances = rf_model.feature_importances_
indices = np.argsort(importances)[::-1]
```

In [40]:
```
# Plot the feature importances
plt.figure(figsize=(15, 6))
sns.barplot(x=importances[indices], y=X.columns[indices])
plt.title('Feature Importances')
plt.xlabel('Relative Importance')
plt.ylabel('Feature Names')
plt.show()
```



In [41]:

```
# Read the data from the specified sheet into a pandas DataFrame
new_data = pd.read_excel(r"C:\Users\hp\Downloads\Churn Prediction Project\Prediction_Join_Data.xlsx", sheet_nam
```

In [42]:

```
# Display the first few rows of the fetched data
new_data.head()
```

Out[42]:

	Customer_ID	Gender	Age	Married	State	Number_of_Referrals	Tenure_in_Months	Value_Deal	Phone_Service	Multiple_Li
0	11751-TAM	Female	18	No	Tamil Nadu	5	7	Deal 5	No	
1	12056-WES	Male	27	No	West Bengal	2	20	NaN	Yes	
2	12136-RAJ	Female	25	Yes	Rajasthan	2	35	NaN	Yes	
3	12257-ASS	Female	39	No	Assam	9	1	NaN	Yes	
4	12340-DEL	Female	51	Yes	Delhi	0	10	NaN	Yes	

5 rows × 32 columns

In [43]:

```
# Retain the original DataFrame to preserve unencoded columns
original_data = new_data.copy()
```

In [44]:

```
# Retain the Customer_ID column
customer_ids = new_data['Customer_ID']
```

In [46]:

```
# Drop columns that won't be used for prediction in the encoded DataFrame
new_data = new_data.drop(['Customer_ID', 'Customer_Status', 'Churn_Category', 'Churn_Reason'], axis=1)
```

In [48]:

```
new_data.head(1)
```

Out[48]:

	Gender	Age	Married	State	Number_of_Referrals	Tenure_in_Months	Value_Deal	Phone_Service	Multiple_Lines	Internet_Serv
0	Female	18	No	Tamil Nadu	5	7	Deal 5	No	No	Y

1 rows × 28 columns

In [49]:

```
# Encode categorical variables using the saved label encoders
for column in new_data.select_dtypes(include=['object']).columns:
    new_data[column] = label_encoders[column].transform(new_data[column])
```

In [50]:

```
# Make predictions
new_predictions = rf_model.predict(new_data)
```

In [51]:

```
new_predictions
```

```
Out[51]: array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1,
                1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
                0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
                0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0,
                1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1], dtype=int64)
```

```
In [52]: # Add predictions to the original DataFrame
         original_data['Customer_Status_Predicted'] = new_predictions
```

```
In [53]: # Filter the DataFrame to include only records predicted as "Churned"
         original_data = original_data[original_data['Customer_Status_Predicted'] == 1]
```

```
In [54]: original_data
```

Out[54]:

	Customer_ID	Gender	Age	Married	State	Number_of_Referrals	Tenure_in_Months	Value_Deal	Phone_Service	Multip
0	11751-TAM	Female	18	No	Tamil Nadu	5	7	Deal 5	No	
1	12056-WES	Male	27	No	West Bengal	2	20	NaN	Yes	
2	12136-RAJ	Female	25	Yes	Rajasthan	2	35	NaN	Yes	
3	12257-ASS	Female	39	No	Assam	9	1	NaN	Yes	
4	12340-DEL	Female	51	Yes	Delhi	0	10	NaN	Yes	
...	...	...	...	...	...	...	...	...	...	
405	99475-KAR	Male	22	Yes	Karnataka	11	11	NaN	Yes	
406	99488-KAR	Male	50	Yes	Karnataka	10	31	NaN	No	
408	99855-MAH	Female	50	Yes	Maharashtra	5	29	NaN	Yes	
409	99862-BIH	Female	30	No	Bihar	12	9	Deal 5	Yes	
410	99996-HAR	Male	22	Yes	Haryana	2	8	Deal 5	Yes	

378 rows × 33 columns

```
In [59]: # Save the results
         original_data.to_csv("C:\\Users\\hp\\Downloads\\Churn Prediction Project\\output.csv", index=False)
```

```
In [ ]:
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js