In [1]:
```
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, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix, classification_report, roc_auc_score,
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
import warnings
warnings.filterwarnings("ignore")
```

In [3]:
```
# Load data
df = pd.read_excel(r"C:\Users\mukki\OneDrive\Desktop\Online Retail.xlsx", sheet_nam
df.dropna(subset=["CustomerID"], inplace=True)
df["InvoiceDate"] = pd.to_datetime(df["InvoiceDate"])
df["Sales"] = df["Quantity"] * df["UnitPrice"]
df
```

Out[3]:

	InvoiceNo	Quantity	InvoiceDate	UnitPrice	CustomerID	Sales
0	536365	6	2010-12-01 08:26:00	2.55	17850.0	15.30
1	536365	6	2010-12-01 08:26:00	3.39	17850.0	20.34
2	536365	8	2010-12-01 08:26:00	2.75	17850.0	22.00
3	536365	6	2010-12-01 08:26:00	3.39	17850.0	20.34
4	536365	6	2010-12-01 08:26:00	3.39	17850.0	20.34
...	...	...	...	...	...	...
4995	536836	2	2010-12-02 18:08:00	10.95	18168.0	21.90
4996	536836	2	2010-12-02 18:08:00	2.55	18168.0	5.10
4997	536836	3	2010-12-02 18:08:00	4.95	18168.0	14.85
4998	536836	2	2010-12-02 18:08:00	1.65	18168.0	3.30
4999	536836	2	2010-12-02 18:08:00	0.85	18168.0	1.70

3795 rows × 6 columns

In [2]:
```
df["Sales"]
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[2], line 1
----> 1 df["Sales"]

NameError: name 'df' is not defined
```

In [ ]:
```
# Compute Recency, Frequency, Monetary (RFM)
max_date = df["InvoiceDate"].max()
rfm = df.groupby("CustomerID").agg({
    "InvoiceDate": lambda x: (max_date - x.max()).days,
    "InvoiceNo": "nunique",
    "Sales": "sum"
}).reset_index()
rfm.columns = ["CustomerID", "Recency", "Frequency", "Monetary"]

# Simulate churn: Recency > 90 days
rfm["Churned"] = (rfm["Recency"] > 90).astype(int)
rfm["Churned"]
```

In [ ]:
```
# Features and target
X = rfm[["Recency", "Frequency", "Monetary"]]
y = rfm["Churned"]

# Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_scaled
# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
print(y_train.value_counts())
```

In [5]:
```
# Evaluation function
def evaluate_model(name, y_test, y_pred, y_prob):
    print(f"--- {name} ---")
    print(confusion_matrix(y_test, y_pred))
    print(classification_report(y_test, y_pred))
    roc_auc = roc_auc_score(y_test, y_prob)
    print(f"AUC-ROC: {roc_auc:.2f}")
    fpr, tpr, _ = roc_curve(y_test, y_prob)
    plt.plot(fpr, tpr, label=f"{name} (AUC={roc_auc:.2f})")

# Compare all models
plt.figure(figsize=(10, 6))
evaluate_model("Logistic Regression", y_test, y_pred_lr, y_prob_lr)
evaluate_model("Decision Tree", y_test, y_pred_dt, y_prob_dt)
evaluate_model("Random Forest", y_test, y_pred_rf, y_prob_rf)
evaluate_model("XGBoost", y_test, y_pred_xgb, y_prob_xgb)
plt.plot([0, 1], [0, 1], "k--")
plt.title("ROC Curves")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[5], line 13
     11 # Compare all models
     12 plt.figure(figsize=(10, 6))
---> 13 evaluate_model("Logistic Regression", y_test, y_pred_lr, y_prob_lr)
     14 evaluate_model("Decision Tree", y_test, y_pred_dt, y_prob_dt)
     15 evaluate_model("Random Forest", y_test, y_pred_rf, y_prob_rf)

NameError: name 'y_test' is not defined
<Figure size 1000x600 with 0 Axes>
```

In [ ]:

In [ ]: