Ethereum Price Prediction (Next 2 Days) Code

```python
# -------------------------------------------------------------
# ETHEREUM PRICE PREDICTION (Next 2 Days) USING ML MODELS
# -------------------------------------------------------------

import pandas as pd
import numpy as np
import yfinance as yf
import matplotlib.pyplot as plt
from datetime import datetime, timedelta
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error

# -------------------------------------------------------------
# STEP 1: Fetch Ethereum Price Data
# -------------------------------------------------------------
end_date = datetime.today()
start_date = end_date - timedelta(days=365)

# Download ETH-USD historical data
eth_df = yf.download('ETH-USD', start=start_date, end=end_date, interval='1d')
eth_df = eth_df[['Close']].rename(columns={'Close': 'price'})
eth_df['price'].fillna(method='ffill', inplace=True)  # Handle missing data
eth_df.reset_index(inplace=True)

# -------------------------------------------------------------
# STEP 2: Create Lag Features
# -------------------------------------------------------------
# Generate previous day prices as features
eth_df['price_t-1'] = eth_df['price'].shift(1)
eth_df['price_t-2'] = eth_df['price'].shift(2)
eth_df['price_t-3'] = eth_df['price'].shift(3)
eth_df.dropna(inplace=True)

# Feature matrix (X) and target (y)
X = eth_df[['price_t-1', 'price_t-2', 'price_t-3']]
y = eth_df['price']

# Scale the features for better performance
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)

# -------------------------------------------------------------
# STEP 3: Train-Test Split
# -------------------------------------------------------------
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.80, shuffle=False)
```

```
# ------------------------------------------------------------
# STEP 4: Train & Evaluate ML Models
# ------------------------------------------------------------
models = {
    'Linear Regression': LinearRegression(),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(),
    'Support Vector Machine': SVR(),
    'K-Nearest Neighbors': KNeighborsRegressor()
}

print(" Model Performance (MSE):")
for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    print(f"{name}: MSE = {mse:.2f}")

# ------------------------------------------------------------
# STEP 5: Predict Next 2 Days (Using Random Forest)
# ------------------------------------------------------------
best_model = models['Random Forest']

# Get last 3 known prices as starting features
latest_features = eth_df[['price_t-1', 'price_t-2', 'price_t-3']].iloc[-1:].values
latest_scaled = scaler.transform(latest_features)

# Predict Day 1 (July 20)
day1_price = best_model.predict(latest_scaled)[0]

# Prepare features for Day 2 using predicted Day 1
new_features = np.array([[day1_price, latest_features[0][0], latest_features[0][1]]])
new_scaled = scaler.transform(new_features)
day2_price = best_model.predict(new_scaled)[0]

print(f"\n Predicted Ethereum Price for July 20: ${day1_price:.2f}")
print(f" Predicted Ethereum Price for July 21: ${day2_price:.2f}")

# ------------------------------------------------------------
# STEP 6: Visualization
# ------------------------------------------------------------
rf_pred = best_model.predict(X_test)
plt.figure(figsize=(10, 5))
plt.plot(y_test.values, label='Actual Price')
plt.plot(rf_pred, label='Predicted Price (RF)', linestyle='--')
plt.axhline(day1_price, color='green', linestyle='--', label='Predicted July 20')
plt.axhline(day2_price, color='orange', linestyle='--', label='Predicted July 21')
plt.title("Ethereum Price Prediction (Next 2 Days)")
plt.xlabel("Sample Index")
plt.ylabel("ETH Price (USD)")
plt.legend()
plt.tight_layout()
plt.show()
```