CSIT 6000Q - Blockchain and Smart Contracts
Assignment 3

Due: 11:59 PM on Sunday, Nov 30, 2024
Released: November 9, 2024

Problem 1 — 40 points

Write test cases for all smart contracts you designed in the last assignment (Assignment 2). And verify them for vulnerabilities using any verification tool. Also, verify with etherscan after successful deployment in your testnet.

Problem 2 — 15+15=30 points

Design your Token (ERC20) and NFT (ERC721) with OpenZeppelin. Add some additional specifications to make your Token and NFT unique. Verify them for vulnerabilities using any verification tool before deploying. Also, verify with etherscan after successful deployment in your testnet.

- **Note:** Grading will be on additional specifications.

Problem 3 — 30 points

Implement a basic trading strategy using a 3-day and 7-day simple moving average (SMA) on Ethereum's daily price data for the past 150 days. Plot the Ethereum price along with the 3-day and 7-day moving averages and mark the buy and sell signals.

Disclaimer: This is an extremely basic trading strategy, please do not use it for real world trading.

Details and Steps

1. **Fetch Ethereum Price Data:**
   Using the provided Python script, retrieve the Ethereum price data for the last 150 days.

```python
import requests
import pandas as pd

def fetch_market_data(token, days):
    # Fetch daily market data for Ethereum (ETH) over the last 150 days
    url = f"https://api.coingecko.com/api/v3/coins/{token}/market_chart"
    params = {"vs_currency": "usd", "days": days, "interval": "daily"}

    response = requests.get(url, params=params)
    data = response.json()

    # Process data into DataFrame
    timestamps = [item[0] for item in data['prices']]
    prices = [item[1] for item in data['prices']]
    df = pd.DataFrame({"timestamp": pd.to_datetime(timestamps, unit='ms'), "price": prices})

    # Save to CSV
    df.to_csv("ethereum_data.csv", index=False)
    print("Data fetched and saved as ethereum_data.csv")

# Run this function with the appropriate function parameters to get the data.
```

Listing 1: Python Code for Fetching Ethereum Price Data

2. **Calculate Moving Averages:**

   - Calculate the 3-day SMA and the 7-day SMA on the price data.
   - Generate a **"buy"** signal when the 3-day SMA crosses above the 7-day SMA (indicating an uptrend).
   - Generate a **"sell"** signal when the 3-day SMA crosses below the 7-day SMA (indicating a downtrend).

   You can implement this part either in Python (simpler, using the Pandas library) or in SQL if you so choose.

   If you choose to implement this in SQL, here is a Python function where you can pass your SQL query as a parameter to execute it:

```python
import sqlite3
import pandas as pd

def execute_sql_query(query):
    # Load CSV data
    df = pd.read_csv("ethereum_data.csv")

    # Create an in-memory SQLite database
    conn = sqlite3.connect(":memory:")
    df.to_sql("eth_prices", conn, index=False, if_exists
        ="replace")

    # Execute the query
    result = pd.read_sql_query(query, conn)

    # Close the connection
    conn.close()
    return result

# Example structure for the SQL query (modify as needed)
query = """
SELECT * FROM ...
...
...
"""
# Run the query
result = execute_sql_query(query)
result
```

Listing 2: Python SQL Execution Function

3. **Visualize the Strategy:**

   - Plot the daily Ethereum price line.
   - Plot the 3-day and 7-day SMA lines.
   - Mark the buy signals with a green dot and sell signals with a red dot on the plot.

   You can use the `matplotlib` library in Python to create the visualizations. Here are some hints:

   - To plot a line chart, use the `plot()` function from `matplotlib.pyplot`.
   - To set the x-axis tick interval (e.g., every 30 days), you can use `mdates.DayLocator(interval=30)` along with `mdates.DateFormatter()` for formatting the date labels.
   - To add labels to the x-axis and y-axis, use `xlabel()` and `ylabel()`.
   - To add a legend to describe each line or marker, use `legend()`.

Your output should look something like this:

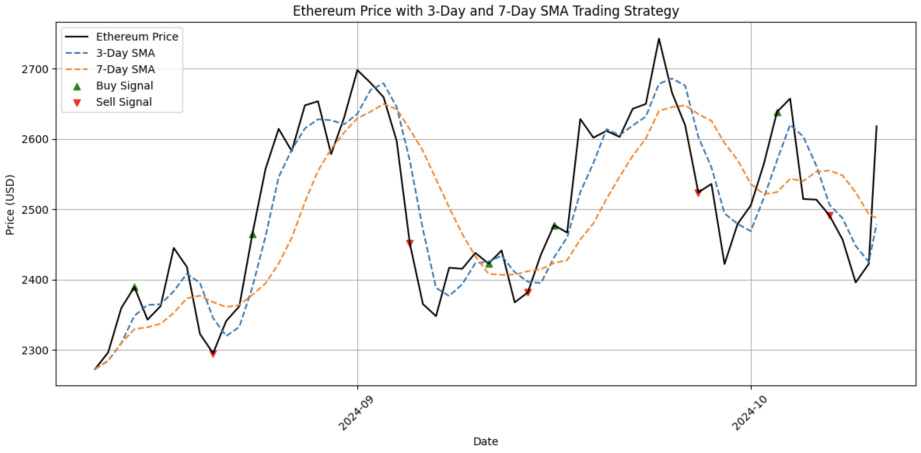

Figure 1: Example output for the SMA Trading Strategy visualization.

4. **Describe Observations:**
    - Briefly summarize what you observe from the visualization: Do the buy/sell signals align with the price trends?

Submission

You must submit your answer as a Jupyter Notebook file (with `.ipynb` extension). Below are instructions on how to set up Python and Jupyter Notebook locally or run your notebook in Google Colab.

Option 1: Installing Python and Jupyter Notebook Locally

1. **Install Python:**
    - Download the latest version of Python from the official Python website.
    - Run the installer and follow the instructions. Make sure to check the box to "Add Python to PATH" during installation.
2. **Install Jupyter Notebook:**
    - Open a command prompt or terminal.
    - Run the following command to install Jupyter Notebook via `pip`:

      ```
      pip install jupyter
      ```

3. **Start Jupyter Notebook:**
    - Navigate to the folder where you want to save your notebook.
    - In the command prompt or terminal, run:

      ```
      jupyter notebook
      ```

    - A new browser tab should open, displaying the Jupyter Notebook interface. You can create a new notebook and start working on your assignment.
4. **Save and Submit:**
    - After completing the assignment, save the notebook as a `.ipynb` file.
    - Upload the `.ipynb` file as your submission.

Option 2: Using Google Colab

1. **Open Google Colab:**
    - Go to Google Colab in your web browser.
    - Sign in with your Google account.
2. **Create a New Notebook:**
    - In the Google Colab interface, click on `File` → `New notebook` to create a new notebook.
3. **Write Your Code:**
    - Google Colab provides an environment similar to Jupyter Notebook, where you can write and execute Python code directly in the browser.
    - Install any necessary libraries (e.g., `pandas`, `matplotlib`) using `!pip install` commands within Colab if they aren't already available.
4. **Download the Notebook:**
    - Once you have completed the assignment, download your notebook as a `.ipynb` file.
    - Click on `File` → `Download` → `Download .ipynb` to save the file to your computer.
5. **Submit the Notebook:**
    - Upload the downloaded `.ipynb` file as your submission.