Assignment 2: Build Your Own Oracle

For this assignment, you must:

1. Build your own Oracle Node

Some incomplete code has been provided in ***oracle-node.py*** to act as your off-chain script. You can use this Python code or design your script (e.g., JavaScript) if Python is not preferred. Below is the information on getting started with ***oracle-node.py***.

Whether you use oracle-node.py or not, your Oracle Node script should:

- Deploy an Oracle contract called ***MyOracle*** on the Sepolia test net – you will need to write this smart contract.
- Retrieve the price of Ether (in USD) (recommended to use CoinmarketCap free API account to get this info)
- Monitor your Oracle contract ***MyOracle*** for requests and update the price of Ether (in USD) to the blockchain whenever a request is made.

2. Write your own Oracle contract

Your Oracle contract ***MyOracle*** needs to have a state variable to store a value representing the price of ETH in USD. It should have functions to get and set the value representing the price of ETH in USD. It should also have a function to request an update for the newest ETH price in USD. When running, the Oracle Node script monitors the event emitted from your Oracle contract. Whenever an update request is detected, the Oracle Node script will fetch the newest ETH price in USD from CoinmarketCap and send a transaction to the blockchain to set the updated ETH price in USD.

Assignment Helper

1. Using ***oracle-node.py***

Make sure you have **Python 3.10** installed. To check the Python version, use the command:

python --version

Install the necessary libraries with pip:

pip install web3 py-solc-x

Run the script:

python ./oracle-node.py

The script will return errors until it has been edited to be completed.

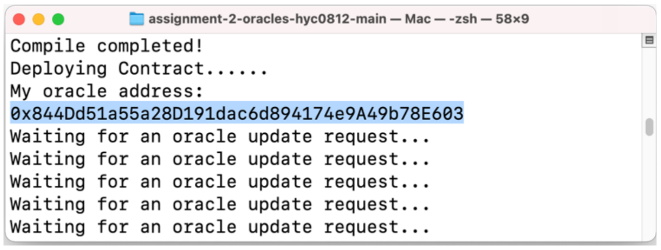
To update the script, you will need to make use of the web3.py documents: https://web3py.readthedocs.io/en/stable/index.html.

Once the script runs, it should compile and deploy the **MyOracle** contract and then wait for an event that will tell it to send a transaction to update the price in the **MyOracle** contract. You can trigger this request by emitting an event using the **MyOracle** contract.

Once you have used the script to compile, deploy your contract and update the price on the blockchain, use CTRL + c to kill your script. Otherwise, it will not exit as it waits for requests to update the price.

2. Testing your framework

Once your script runs, it will print the MyOracle contract address deployed on Sepolia. We will use this contract address in Remix IDE.



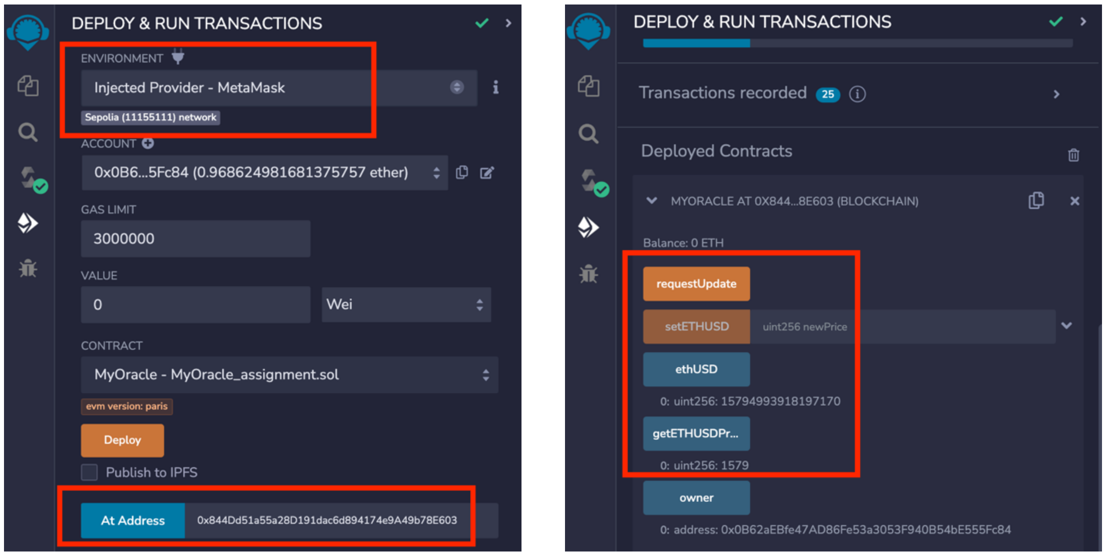
```
Compile completed!
Deploying Contract......
My oracle address:
0x844Dd51a55a28D191dac6d894174e9A49b78E603
Waiting for an oracle update request...
Waiting for an oracle update request...
Waiting for an oracle update request...
Waiting for an oracle update request...
Waiting for an oracle update request...
```

Figure 1

Use Remix IDE to interact with your MyOracle contract, as shown in Figure 2:

- Connect your MetaMask to the Sepolia network

- Select the environment as Injected Provider – MetaMask
- Copy and paste your MyOracle contract into Remix IDE
- Copy and paste the MyOracle contract address into 'At Address' and click the blue button
- Call the function requesting the ETH (in USD) price update.
- Your Oracle Node can detect the request from the blockchain, fetch the newest ETH (in USD) price from CoinmarketCap and write the value to the blockchain.
- After updating the price to the blockchain, you can check the newest price using the State variable or the get function of your smart contract, as shown in Figure 2 on the right.



DEPLOY & RUN TRANSACTIONS

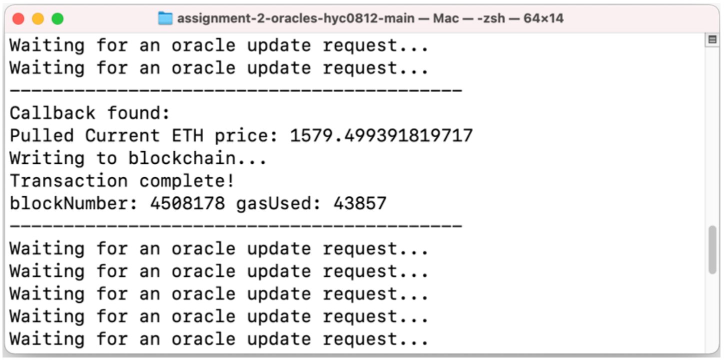
ENVIRONMENT
Injected Provider - MetaMask
Sepolia (11155111) network
ACCOUNT
0x0B6...5Fc84 (0.968624981681375757 ether)
GAS LIMIT
3000000
VALUE
0 Wei
CONTRACT
MyOracle - MyOracle_assignment.sol
evm version: paris
Deploy
Publish to IPFS
At Address 0x844Dd51a55a28D191dac6d894174e9A49b78E603

DEPLOY & RUN TRANSACTIONS

Transactions recorded 25
Deployed Contracts
MYORACLE AT 0X844...8E603 (BLOCKCHAIN)
Balance: 0 ETH
requestUpdate
setETHUSD uint256 newPrice
ethUSD
0: uint256: 15794993918197170
getETHUSDPr...
0: uint256: 1579
owner
0: address: 0x0B62aEBfe47AD86Fe53a3053F940B54bE555Fc84

Figure 2

The result should be something like this in the terminal window:



assignment-2-oracles-hyc0812-main — Mac — -zsh — 64×14

```
Waiting for an oracle update request...
Waiting for an oracle update request...
------------------------------------------
Callback found:
Pulled Current ETH price: 1579.499391819717
Writing to blockchain...
Transaction complete!
blockNumber: 4508178 gasUsed: 43857
------------------------------------------
Waiting for an oracle update request...
Waiting for an oracle update request...
Waiting for an oracle update request...
Waiting for an oracle update request...
Waiting for an oracle update request...
```

Figure 3

Rubrics For the submission:

Please upload three files in the Bright Space:

1. MyOracle.sol (4 points). It should:

- Have no compiling errors. (2 points)
- Have the get function to get the ETH price in USD. (0.5 points)
- Have the set function to set the ETH price in USD. (0.5 points)
- Have a function to request an update for the newest ETH price in USD. (1 point)

2. oracle-node.py (6 points). It should:

- Have no compiling errors. (2 points)
- Have completed the function **get_eth_price.** (1 point)
- Have completed the function **deploy_oracle.** (1 point)
- Have completed the function **update_oracle.** (1 point)
- Have completed the main function. (1 point)

3. A PDF file called **Assignment 2 Report** containing your explanations and screenshots. (5 points). It should:

- Have screenshot(s) and necessary explanations showing your smart contract has been compiled and deployed to an address on the Sepolia test network, as shown in Figure 1. (2 points)
- Have screenshot(s) and necessary explanations showing your Oracle has pulled the new ETH price in USD and written it to the blockchain, as shown in Figure 3. (1.5 points)
- Have screenshot(s) and necessary explanations showing you can check the updated ETH price in USD using the state variable or the get function, as shown in Figure 2 on the right. (1.5 points)