Blockchain & Cryptocurrency

Eng: Islam Ahmed

Lab1

Name: Ahmed Ismail Farouk

Section. 1

Code: 220100279

❖ **Code and Output :-**

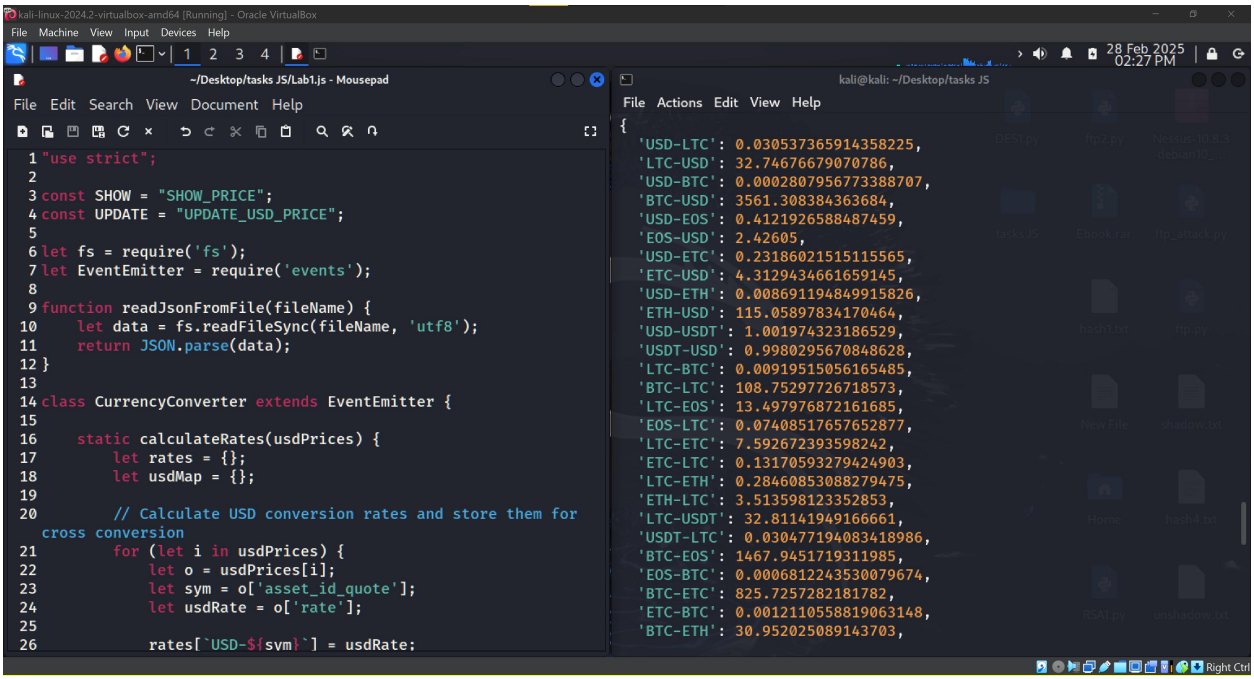
```
"use strict";


const SHOW = "SHOW_PRICE";

const UPDATE = "UPDATE_USD_PRICE";


let fs = require('fs');

let EventEmitter = require('events');


function readJsonFromFile(fileName) {

    let data = fs.readFileSync(fileName, 'utf8');

    return JSON.parse(data);

}


class CurrencyConverter extends EventEmitter {


    static calculateRates(usdPrices) {

        let rates = {};

        let usdMap = {};


        // Calculate USD conversion rates and store them for cross conversion

        for (let i in usdPrices) {

            let o = usdPrices[i];
```

```
        let sym = o['asset_id_quote'];
        let usdRate = o['rate'];

        rates[`USD-${sym}`] = usdRate;
        rates[`${sym}-USD`] = 1 / usdRate;
        usdMap[sym] = usdRate;
    }

    // Calculate direct crypto-to-crypto conversion rates
    let symbols = Object.keys(usdMap);
    for (let from of symbols) {
        for (let to of symbols) {
            if (from !== to) {
                let tag = `${from}-${to}`;
                rates[tag] = usdMap[to] / usdMap[from];
                rates[`${to}-${from}`] = usdMap[from] / usdMap[to];
            }
        }
    }

    return rates;
}

constructor(coin2USD) {
    super();
    this.rates = this.constructor.calculateRates(coin2USD.rates);

    this.on(SHOW, (o) => {
        console.log("SHOW event received.");
        console.log(o);
        const { from, to } = o;
```

```
    try {
        let rate = this.convert(1, from, to);
        console.log(`1 ${from} is worth ${rate} ${to}`);
    } catch (e) {
        console.error(e.message);
    }
});

this.on(UPDATE, (o) => {
    const { sym, usdPrice } = o;
    if (!sym || !usdPrice || usdPrice <= 0) {
        console.error("Invalid update parameters.");
        return;
    }
    console.log(`Updating ${sym} price to ${usdPrice} USD.`);

    // Update USD rates
    // complete the equality
    this.rates[`USD-${sym}`] = usdPrice;
    this.rates[`${sym}-USD`] = 1 / usdPrice;

    // Recalculate all crypto-to-crypto rates
    const symbols = Object.keys(this.rates)
        .filter(key => key.startsWith('USD-'))
        .map(key => key.split('-')[1]);

    console.log("symbols", symbols);

    for (let from of symbols) {
        for (let to of symbols) {
            if (from !== to) {
```

```
                        this.rates[`${from}-${to}`] = this.rates[`USD-${to}`] / this.rates[`USD-
${from}`];
                    }
                }
            }

            console.log("Rates updated successfully.");
        });
    }

    convert(amount, fromUnits, toUnits) {
        let tag = `${fromUnits}-${toUnits}`;
        let rate = this.rates[tag];
        if (rate === undefined) {
            throw new Error(`Rate for ${tag} not found`);
        }
        return rate * amount;
    }
}

// All prices listed are in USD
// write here your JSON File Path (rates.json)
const PATH = './rates.json';
let cnv = new CurrencyConverter(readJsonFromFile(PATH));

console.log(cnv.rates);

console.log("========================================================================");

function test(amt, from, to) {
    console.log(`${amt} ${from} is worth ${cnv.convert(amt, from, to)} ${to}.`);
}
```

```
test(4000, 'ETH', 'BTC');

test(200, 'BTC', 'EOS');


console.log("========================================================================");


// Test event handling

cnv.emit(SHOW, { from: "EOS", to: "BTC" });

console.log("========================================================================");


cnv.emit(SHOW, { from: "EOS", to: "ETH" });

console.log("========================================================================");


cnv.emit(SHOW, { from: "ETC", to: "ETH" });

console.log("========================================================================");


cnv.emit(SHOW, { from: "LTC", to: "BTC" });

console.log("========================================================================");


cnv.emit(UPDATE, { sym: "BTC", usdPrice: 50000 });

console.log("========================================================================");


cnv.emit(SHOW, { from: "LTC", to: "BTC" });
```



❖ First part : -

- Function: **`readJsonFromFile`**

```
 9 function readJsonFromFile(fileName) {
10     let data = fs.readFileSync(fileName, 'utf8');
11     return JSON.parse(data);
12 }
```

- **What does this function do?**

This function reads a JSON file and converts its content into a JavaScript object.

➢ **Step-by-step explanation:**

1. **`fs.readFileSync(fileName, 'utf8');`**
   - Reads the file synchronously using the **`fs`** (File System) module.
   - **`'utf8'`** ensures the file is read as a UTF-8 encoded string.
   - The file content is stored in the **data** variable as a string.
2. **`JSON.parse(data);`**
   - Converts the JSON string into a JavaScript object.
   - return JSON.parse(data);

- Returns the parsed object, allowing us to use the data in JavaScript.

❖ Second part :-

- **Calculating Crypto-to-Crypto Conversion Rates:**

```
rates[tag] = usdMap[to] / usdMap[from];
rates[`${to}-${from}`] = usdMap[from] / usdMap[to];
```

- **What does this code do?**

This code calculates conversion rates between different cryptocurrencies based on their USD values.

➢ Step-by-step explanation:

1. `usdMap[to] / usdMap[from]`

   - `usdMap` is an object containing the **USD price** of each cryptocurrency.
   - `usdMap[to]` represents the **USD value** of the target currency.
   - `usdMap[from]` represents the **USD value** of the source currency.
   - The division calculates the **conversion rate** between the two cryptocurrencies.

2. `rates[tag] = usdMap[to] / usdMap[from];`

   - Saves the conversion rate from `from → to` in the `rates` object.
   - Example: If `tag = "BTC-ETH"`, it stores how many `ETH` are equal to **1 BTC**.

3. `rates[${to}-${from}] = usdMap[from] / usdMap[to];`

   - Saves the reverse conversion rate from to → from.
   - Example: If `tag = "ETH-BTC"`, it stores how many BTC are equal to **1 ETH.**

❖ Third part : -

- **Updating Cryptocurrency Prices:**

```
// Update USD rates
// complete the equality
this.rates[`USD-${sym}`] = usdPrice;
this.rates[`${sym}-USD`] = 1 / usdPrice;
```

- **What does this code do?**

This code updates the exchange rate of a cryptocurrency against USD and calculates its reverse rate.

➢ **Step-by-step explanation:**

1. this.rates[USD-${sym}] = usdPrice;

- Updates the USD price of the cryptocurrency in the rates object.

this.rates[${sym}-USD] = 1 / usdPrice;

- Calculates the reverse exchange rate, determining how much of the cryptocurrency is equal to 1 USD.
