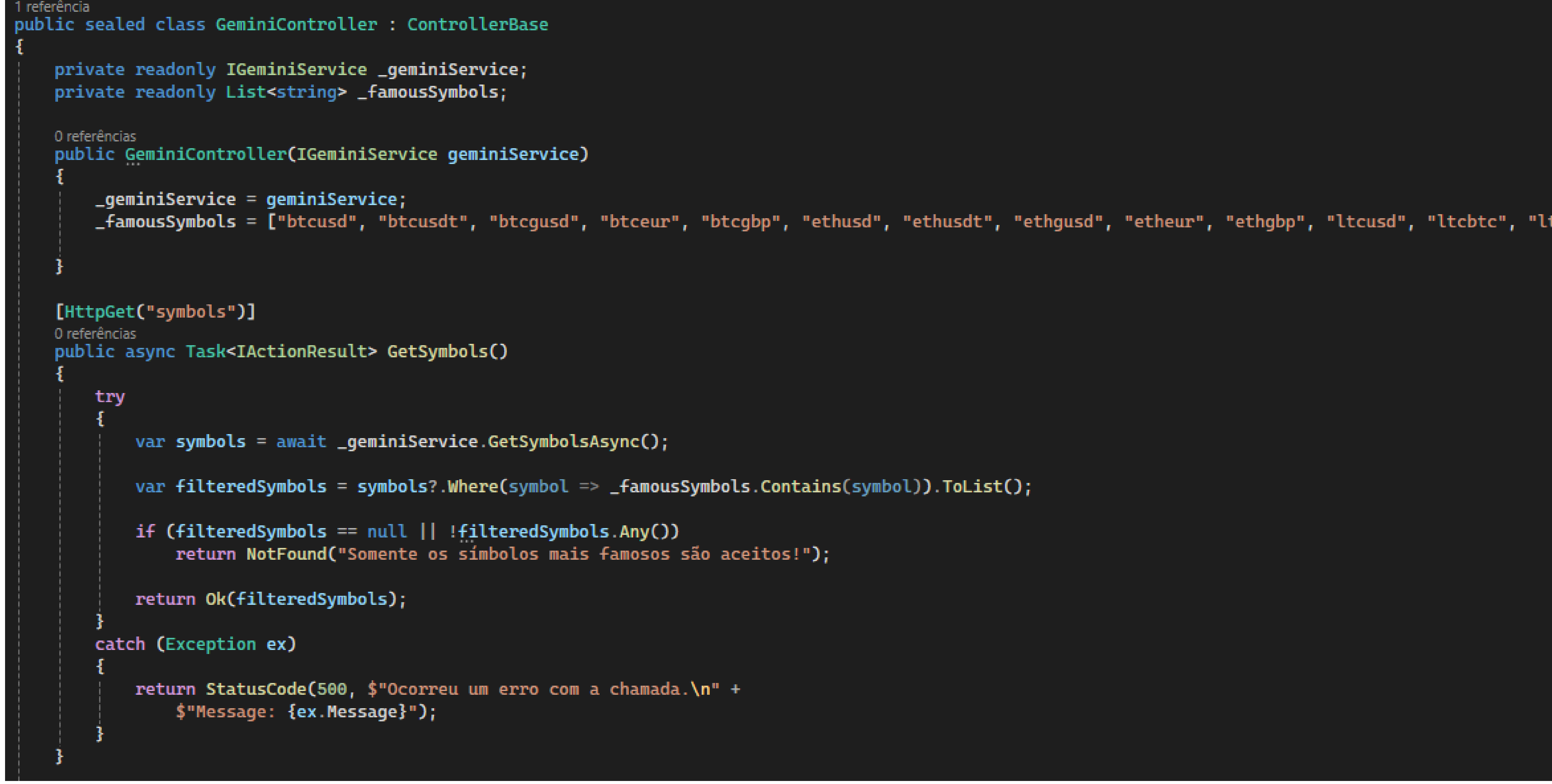
DADOS DE CRIPTOMOEDAS USANDO GEMINI

API

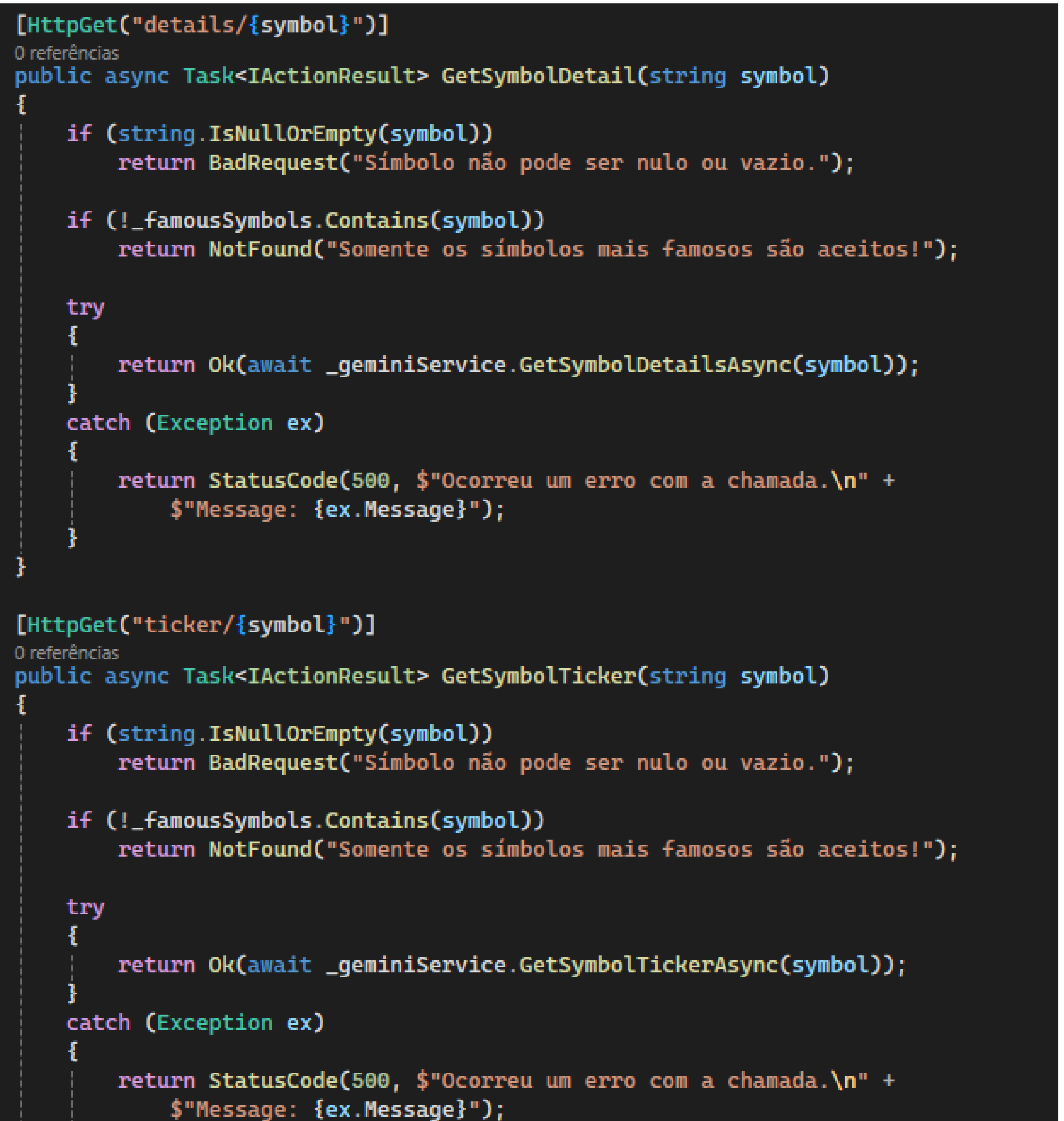
DESENVOLVIDO EM C#

1.



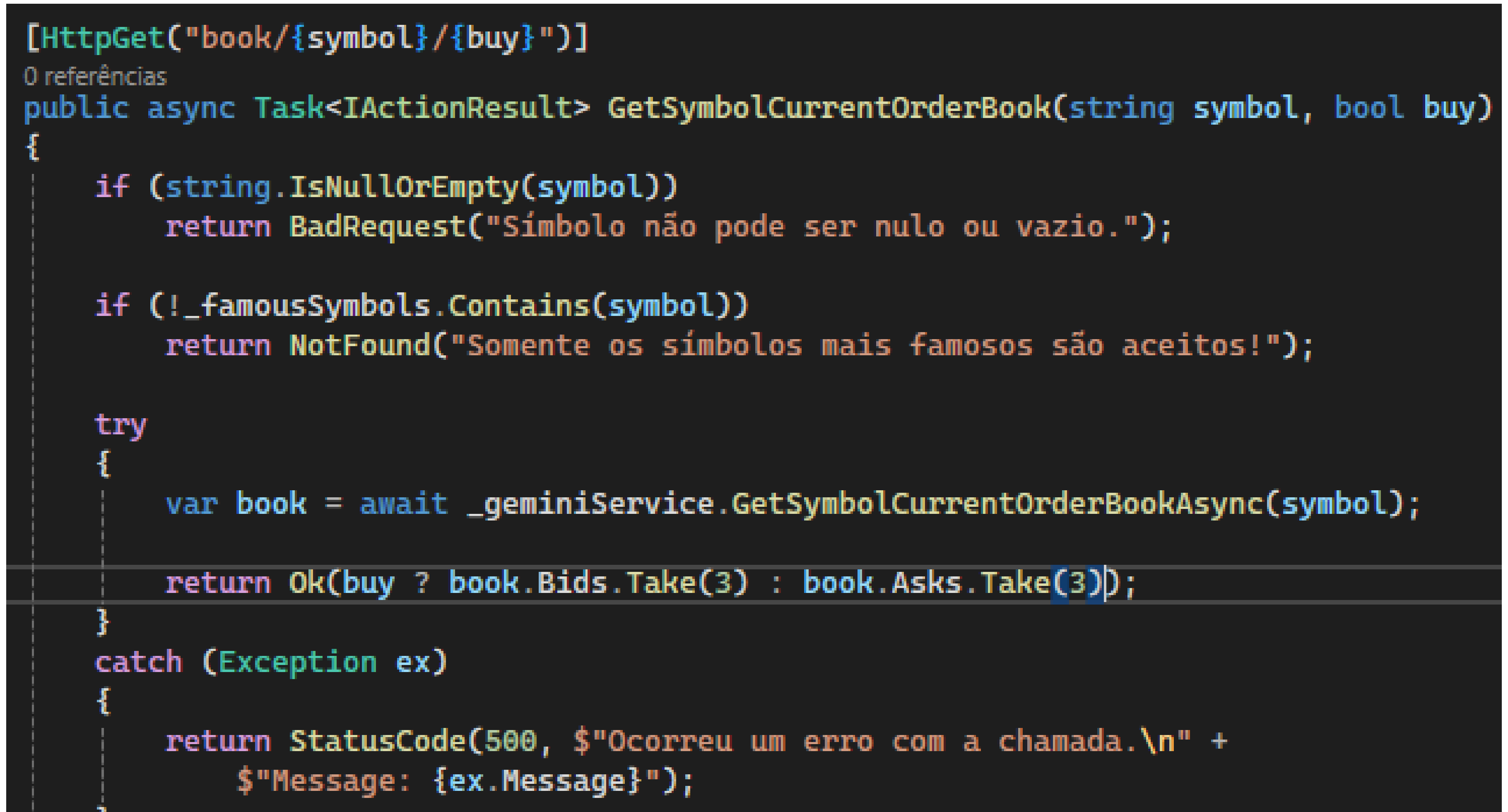
```
1 referência
public sealed class GeminiController : ControllerBase
{
    private readonly IGeminiService _geminiService;
    private readonly List<string> _famousSymbols;

    0 referências
    public GeminiController(IGeminiService geminiService)
    {
        _geminiService = geminiService;
        _famousSymbols = ["btcusd", "btcusdt", "btcgusd", "btceur", "btcgbp", "ethusd", "ethusdt", "ethgusd", "etheur", "ethgbp", "ltcusd", "ltcbtc", "lt

    }

    [HttpGet("symbols")]
    0 referências
    public async Task<IActionResult> GetSymbols()
    {
        try
        {
            var symbols = await _geminiService.GetSymbolsAsync();

            var filteredSymbols = symbols?.Where(symbol => _famousSymbols.Contains(symbol)).ToList();

            if (filteredSymbols == null || !filteredSymbols.Any())
                return NotFound("Somente os símbolos mais famosos são aceitos!");

            return Ok(filteredSymbols);
        }
        catch (Exception ex)
        {
            return StatusCode(500, $"Ocorreu um erro com a chamada.\n" +
                $"Message: {ex.Message}");
        }
    }
```

2.



```
[HttpGet("details/{symbol}")]
0 referências
public async Task<IActionResult> GetSymbolDetail(string symbol)
{
    if (string.IsNullOrEmpty(symbol))
        return BadRequest("Símbolo não pode ser nulo ou vazio.");

    if (!_famousSymbols.Contains(symbol))
        return NotFound("Somente os símbolos mais famosos são aceitos!");

    try
    {
        return Ok(await _geminiService.GetSymbolDetailsAsync(symbol));
    }
    catch (Exception ex)
    {
        return StatusCode(500, $"Ocorreu um erro com a chamada.\n" +
            $"Message: {ex.Message}");
    }
}

[HttpGet("ticker/{symbol}")]
0 referências
public async Task<IActionResult> GetSymbolTicker(string symbol)
{
    if (string.IsNullOrEmpty(symbol))
        return BadRequest("Símbolo não pode ser nulo ou vazio.");

    if (!_famousSymbols.Contains(symbol))
        return NotFound("Somente os símbolos mais famosos são aceitos!");

    try
    {
        return Ok(await _geminiService.GetSymbolTickerAsync(symbol));
    }
    catch (Exception ex)
    {
        return StatusCode(500, $"Ocorreu um erro com a chamada.\n" +
            $"Message: {ex.Message}");
```

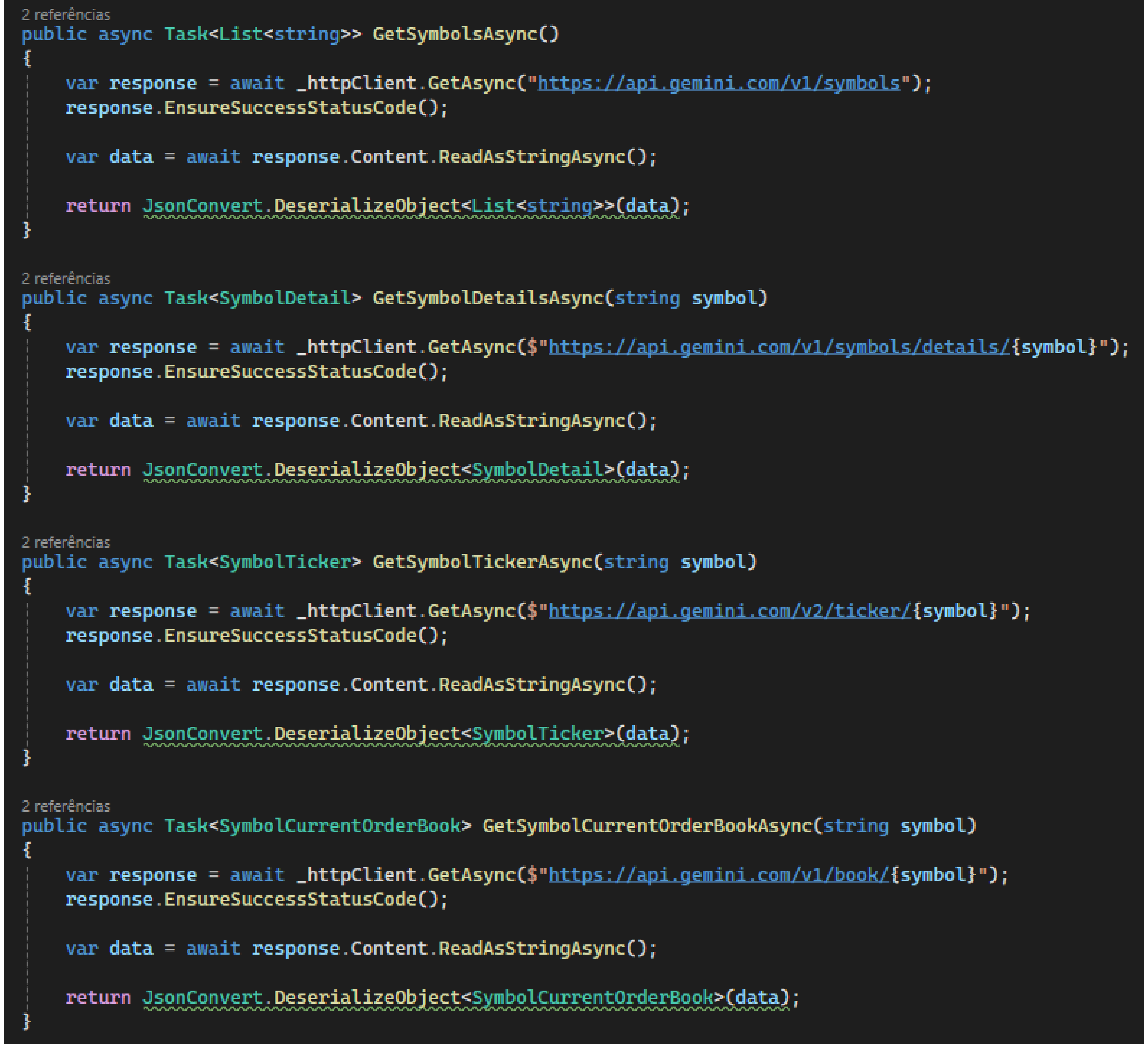
3.



```
[HttpGet("book/{symbol}/{buy}")]
0 referências
public async Task<IActionResult> GetSymbolCurrentOrderBook(string symbol, bool buy)
{
    if (string.IsNullOrEmpty(symbol))
        return BadRequest("Símbolo não pode ser nulo ou vazio.");

    if (!_famousSymbols.Contains(symbol))
        return NotFound("Somente os símbolos mais famosos são aceitos!");

    try
    {
        var book = await _geminiService.GetSymbolCurrentOrderBookAsync(symbol);

        return Ok(buy ? book.Bids.Take(3) : book.Asks.Take(3));
    }
    catch (Exception ex)
    {
        return StatusCode(500, $"Ocorreu um erro com a chamada.\n" +
            $"Message: {ex.Message}");
```

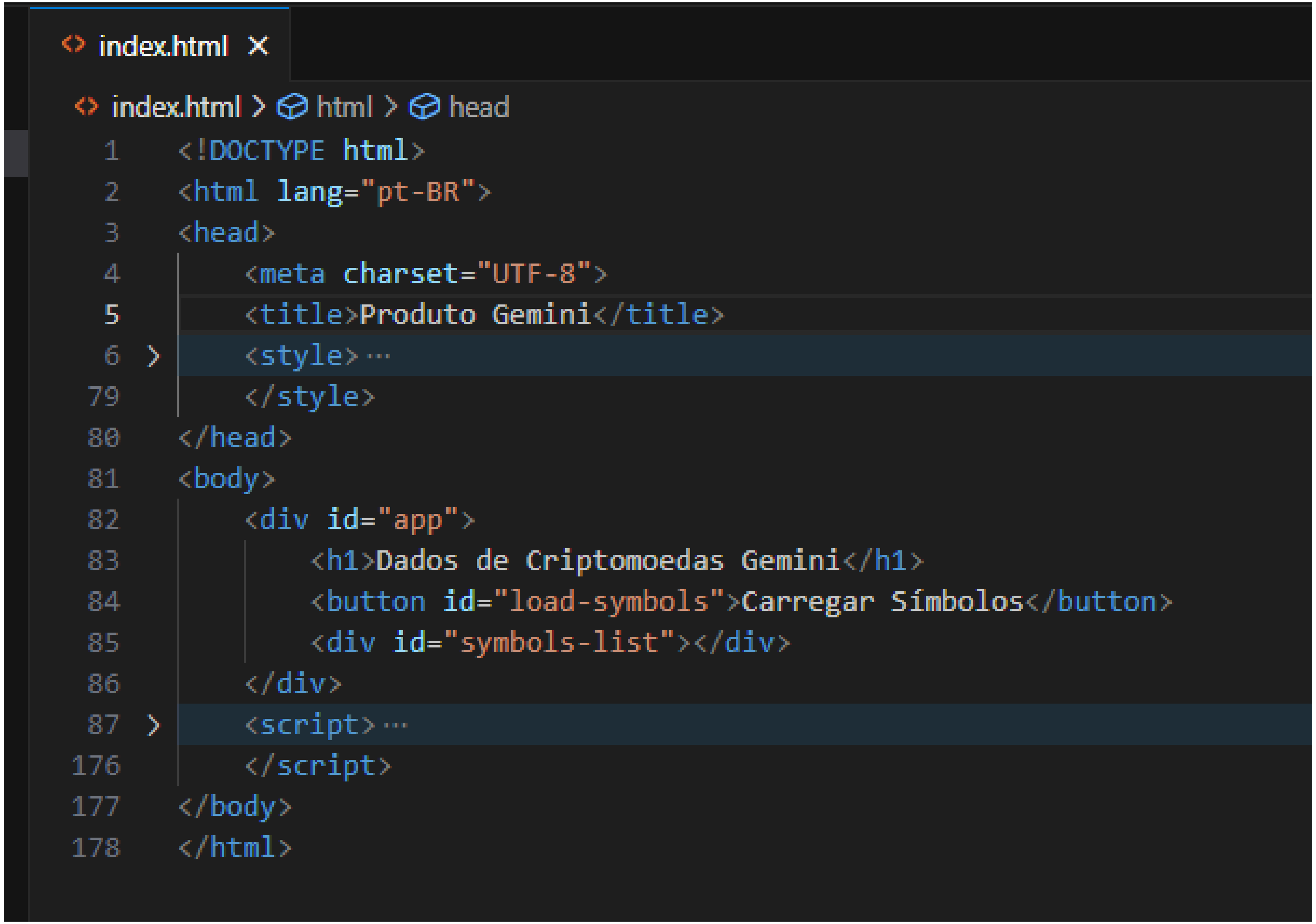
GEMINI

CONSULTAS
AO
GEMINI
USADAS
PARA
CRIAR
O
SITE



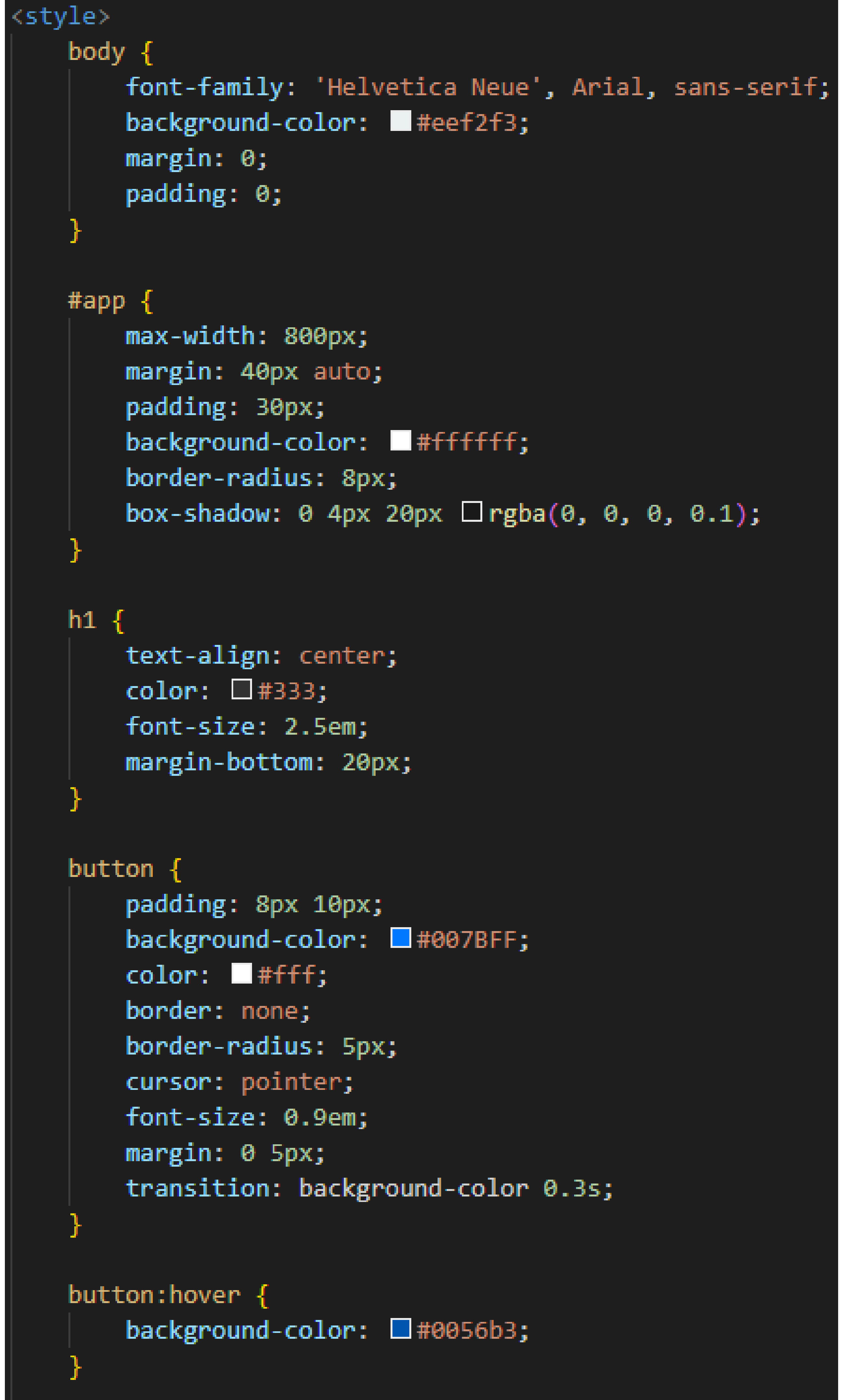
```
2 referências
public async Task<List<string>> GetSymbolsAsync()
{
    var response = await _httpClient.GetAsync("https://api.gemini.com/v1/symbols");
    response.EnsureSuccessStatusCode();

    var data = await response.Content.ReadAsStringAsync();

    return JsonConvert.DeserializeObject<List<string>>(data);
}

2 referências
public async Task<SymbolDetail> GetSymbolDetailsAsync(string symbol)
{
    var response = await _httpClient.GetAsync($"https://api.gemini.com/v1/symbols/details/{symbol}");
    response.EnsureSuccessStatusCode();

    var data = await response.Content.ReadAsStringAsync();

    return JsonConvert.DeserializeObject<SymbolDetail>(data);
}

2 referências
public async Task<SymbolTicker> GetSymbolTickerAsync(string symbol)
{
    var response = await _httpClient.GetAsync($"https://api.gemini.com/v2/ticker/{symbol}");
    response.EnsureSuccessStatusCode();

    var data = await response.Content.ReadAsStringAsync();

    return JsonConvert.DeserializeObject<SymbolTicker>(data);
}

2 referências
public async Task<SymbolCurrentOrderBook> GetSymbolCurrentOrderBookAsync(string symbol)
{
    var response = await _httpClient.GetAsync($"https://api.gemini.com/v1/book/{symbol}");
    response.EnsureSuccessStatusCode();

    var data = await response.Content.ReadAsStringAsync();

    return JsonConvert.DeserializeObject<SymbolCurrentOrderBook>(data);
}
```

HTML

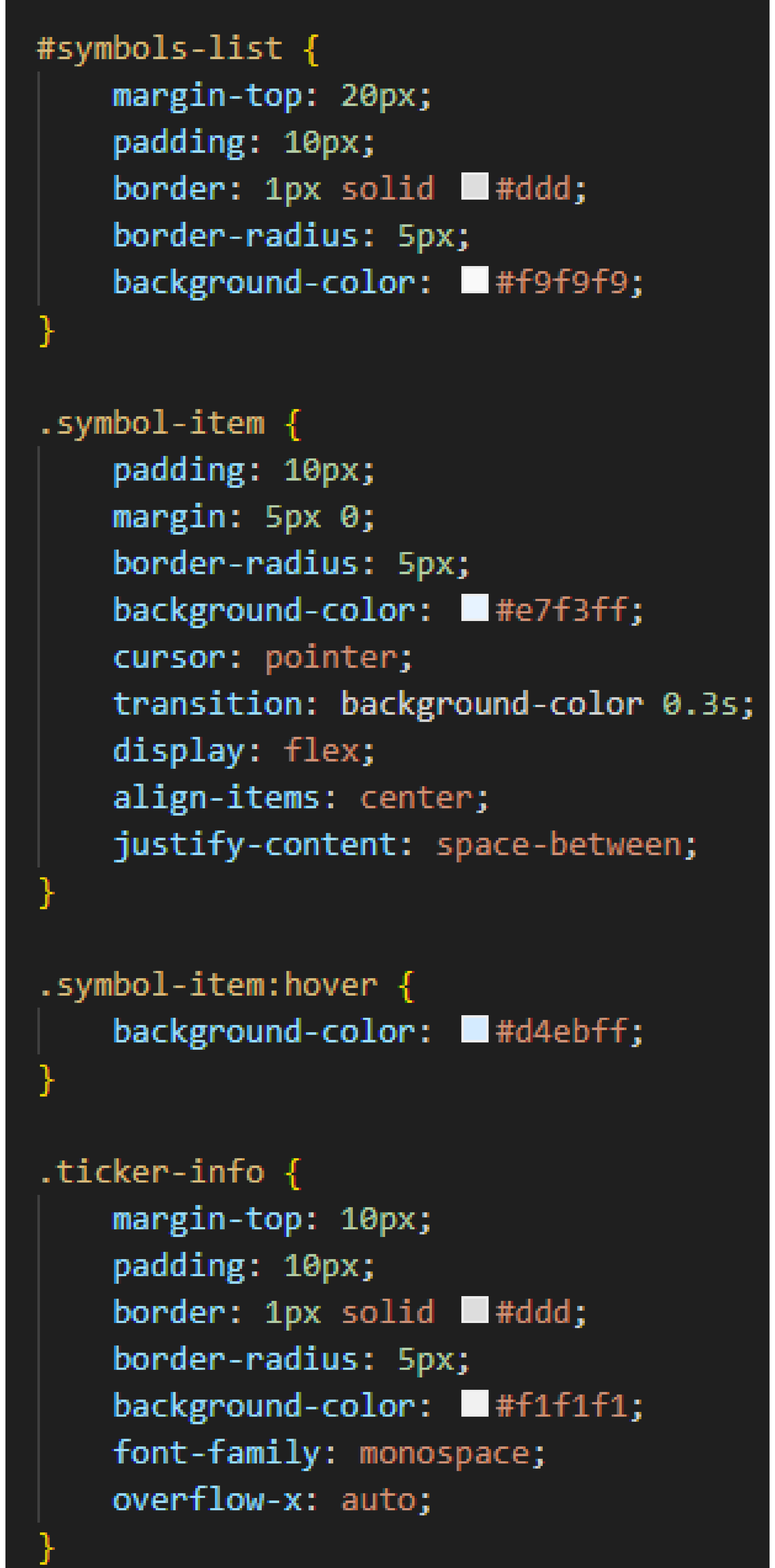


```html
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <title>Produto Gemini</title>
    <style>⋯
    </style>
</head>
<body>
    <div id="app">
        <h1>Dados de Criptomoedas Gemini</h1>
        <button id="load-symbols">Carregar Símbolos</button>
        <div id="symbols-list"></div>
    </div>
    <script>⋯
    </script>
</body>
</html>
```

CSS

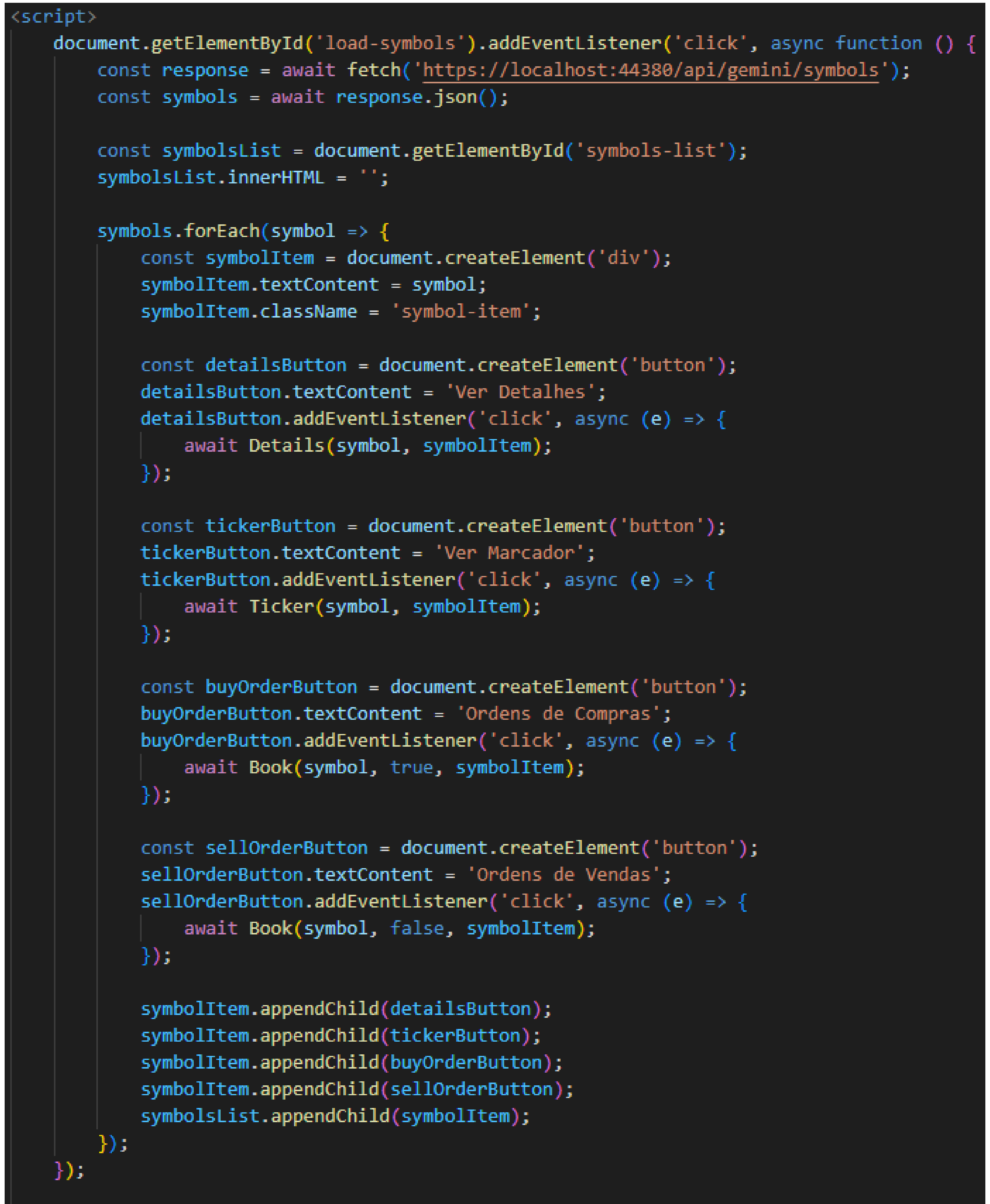


```css
<style>
    body {
        font-family: 'Helvetica Neue', Arial, sans-serif;
        background-color: #eef2f3;
        margin: 0;
        padding: 0;
    }

    #app {
        max-width: 800px;
        margin: 40px auto;
        padding: 30px;
        background-color: #ffffff;
        border-radius: 8px;
        box-shadow: 0 4px 20px rgba(0, 0, 0, 0.1);
    }

    h1 {
        text-align: center;
        color: #333;
        font-size: 2.5em;
        margin-bottom: 20px;
    }

    button {
        padding: 8px 10px;
        background-color: #007BFF;
        color: #fff;
        border: none;
        border-radius: 5px;
        cursor: pointer;
        font-size: 0.9em;
        margin: 0 5px;
        transition: background-color 0.3s;
    }

    button:hover {
        background-color: #0056b3;
    }
```

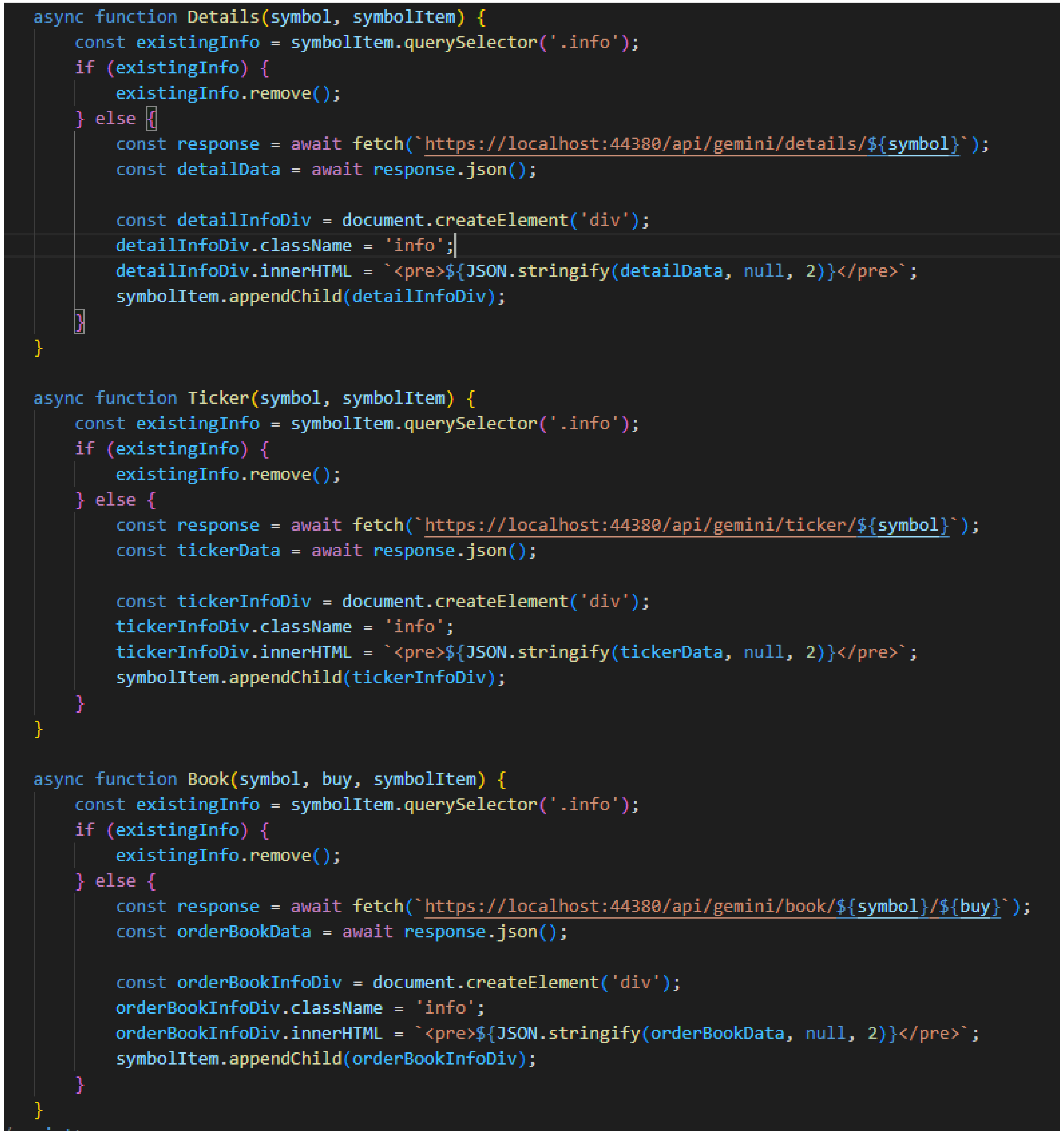


```css
#symbols-list {
    margin-top: 20px;
    padding: 10px;
    border: 1px solid #ddd;
    border-radius: 5px;
    background-color: #f9f9f9;
}

.symbol-item {
    padding: 10px;
    margin: 5px 0;
    border-radius: 5px;
    background-color: #e7f3ff;
    cursor: pointer;
    transition: background-color 0.3s;
    display: flex;
    align-items: center;
    justify-content: space-between;
}

.symbol-item:hover {
    background-color: #d4ebff;
}

.ticker-info {
    margin-top: 10px;
    padding: 10px;
    border: 1px solid #ddd;
    border-radius: 5px;
    background-color: #f1f1f1;
    font-family: monospace;
    overflow-x: auto;
}
```

JAVASCRIPT

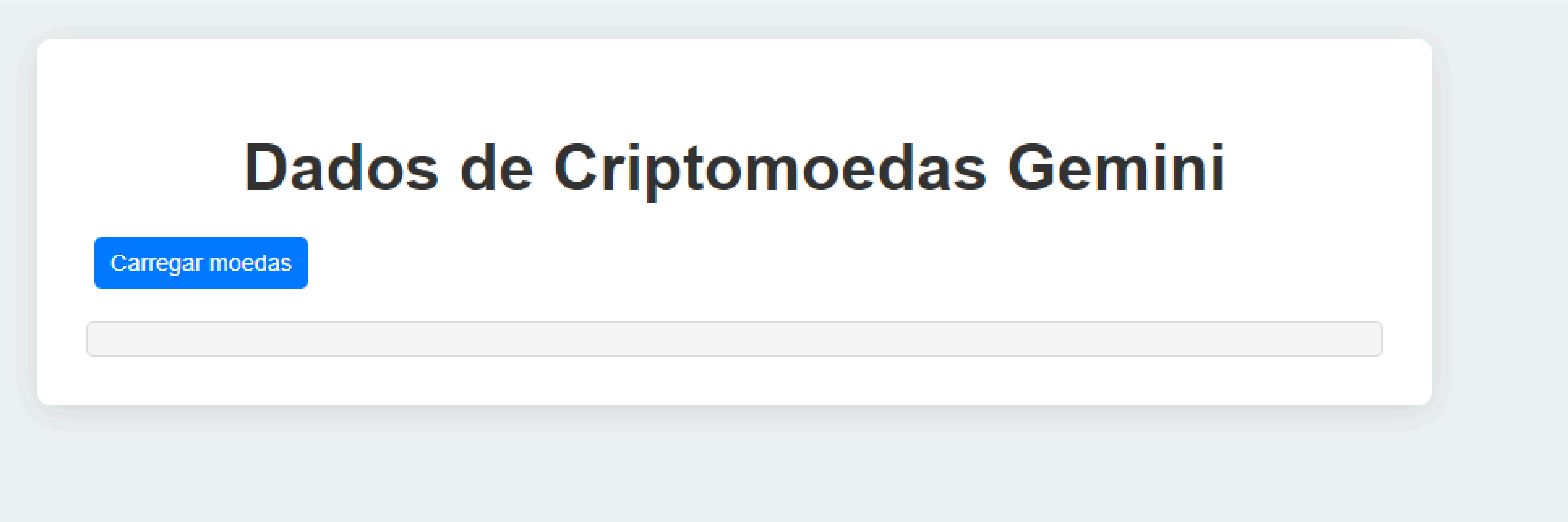


```
<script>
    document.getElementById('load-symbols').addEventListener('click', async function () {
        const response = await fetch('https://localhost:44380/api/gemini/symbols');
        const symbols = await response.json();

        const symbolsList = document.getElementById('symbols-list');
        symbolsList.innerHTML = '';

        symbols.forEach(symbol => {
            const symbolItem = document.createElement('div');
            symbolItem.textContent = symbol;
            symbolItem.className = 'symbol-item';

            const detailsButton = document.createElement('button');
            detailsButton.textContent = 'Ver Detalhes';
            detailsButton.addEventListener('click', async (e) => {
                await Details(symbol, symbolItem);
            });

            const tickerButton = document.createElement('button');
            tickerButton.textContent = 'Ver Marcador';
            tickerButton.addEventListener('click', async (e) => {
                await Ticker(symbol, symbolItem);
            });

            const buyOrderButton = document.createElement('button');
            buyOrderButton.textContent = 'Ordens de Compras';
            buyOrderButton.addEventListener('click', async (e) => {
                await Book(symbol, true, symbolItem);
            });

            const sellOrderButton = document.createElement('button');
            sellOrderButton.textContent = 'Ordens de Vendas';
            sellOrderButton.addEventListener('click', async (e) => {
                await Book(symbol, false, symbolItem);
            });

            symbolItem.appendChild(detailsButton);
            symbolItem.appendChild(tickerButton);
            symbolItem.appendChild(buyOrderButton);
            symbolItem.appendChild(sellOrderButton);
            symbolsList.appendChild(symbolItem);
        });
    });
```



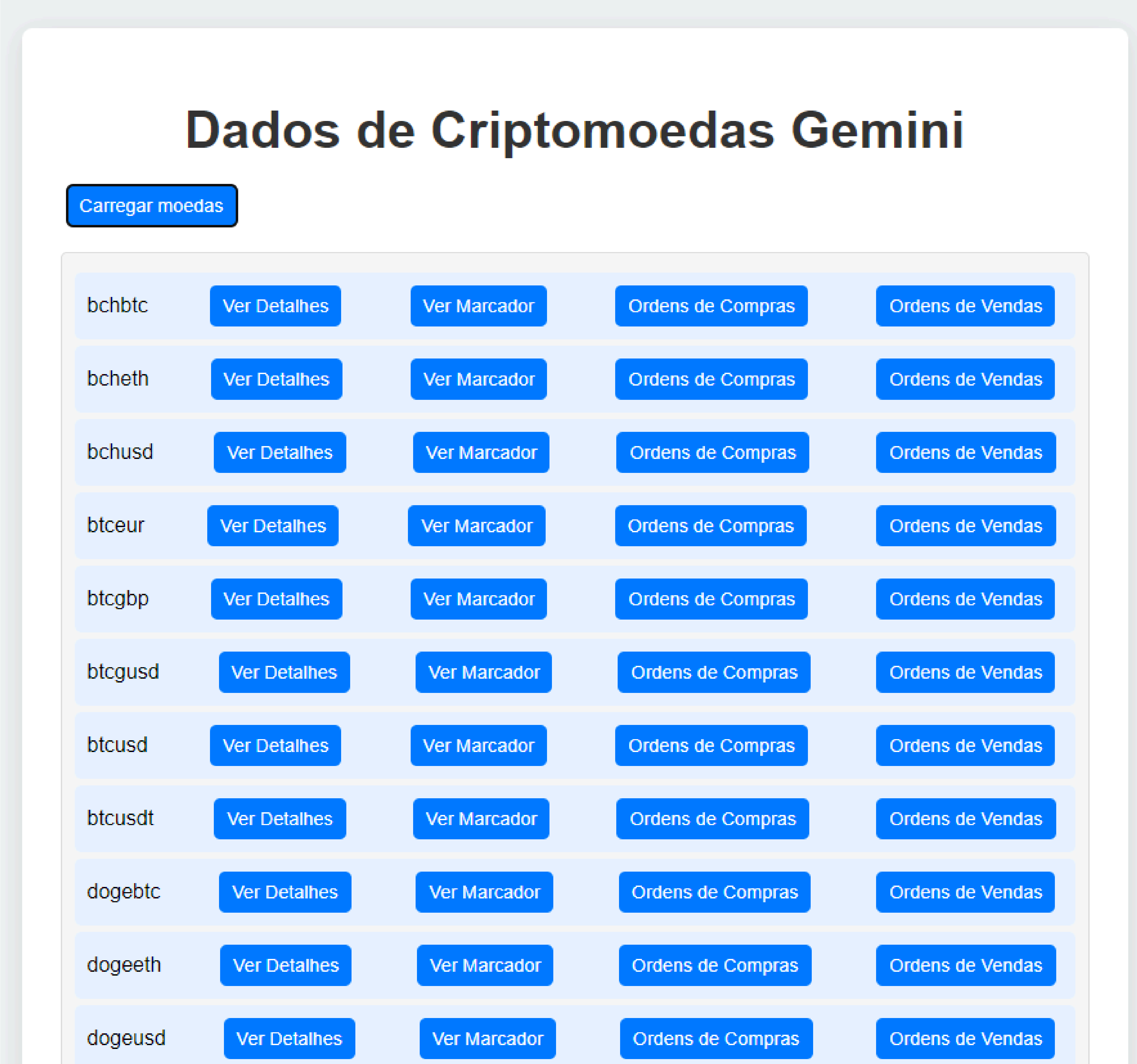
```
async function Details(symbol, symbolItem) {
    const existingInfo = symbolItem.querySelector('.info');
    if (existingInfo) {
        existingInfo.remove();
    } else {
        const response = await fetch(`https://localhost:44380/api/gemini/details/${symbol}`);
        const detailData = await response.json();

        const detailInfoDiv = document.createElement('div');
        detailInfoDiv.className = 'info';
        detailInfoDiv.innerHTML = `<pre>${JSON.stringify(detailData, null, 2)}</pre>`;
        symbolItem.appendChild(detailInfoDiv);
    }
}

async function Ticker(symbol, symbolItem) {
    const existingInfo = symbolItem.querySelector('.info');
    if (existingInfo) {
        existingInfo.remove();
    } else {
        const response = await fetch(`https://localhost:44380/api/gemini/ticker/${symbol}`);
        const tickerData = await response.json();

        const tickerInfoDiv = document.createElement('div');
        tickerInfoDiv.className = 'info';
        tickerInfoDiv.innerHTML = `<pre>${JSON.stringify(tickerData, null, 2)}</pre>`;
        symbolItem.appendChild(tickerInfoDiv);
    }
}

async function Book(symbol, buy, symbolItem) {
    const existingInfo = symbolItem.querySelector('.info');
    if (existingInfo) {
        existingInfo.remove();
    } else {
        const response = await fetch(`https://localhost:44380/api/gemini/book/${symbol}/${buy}`);
        const orderBookData = await response.json();

        const orderBookInfoDiv = document.createElement('div');
        orderBookInfoDiv.className = 'info';
        orderBookInfoDiv.innerHTML = `<pre>${JSON.stringify(orderBookData, null, 2)}</pre>`;
        symbolItem.appendChild(orderBookInfoDiv);
    }
}
```

TELA INICIAL



Dados de Criptomoedas Gemini

TELA COM MOEDAS

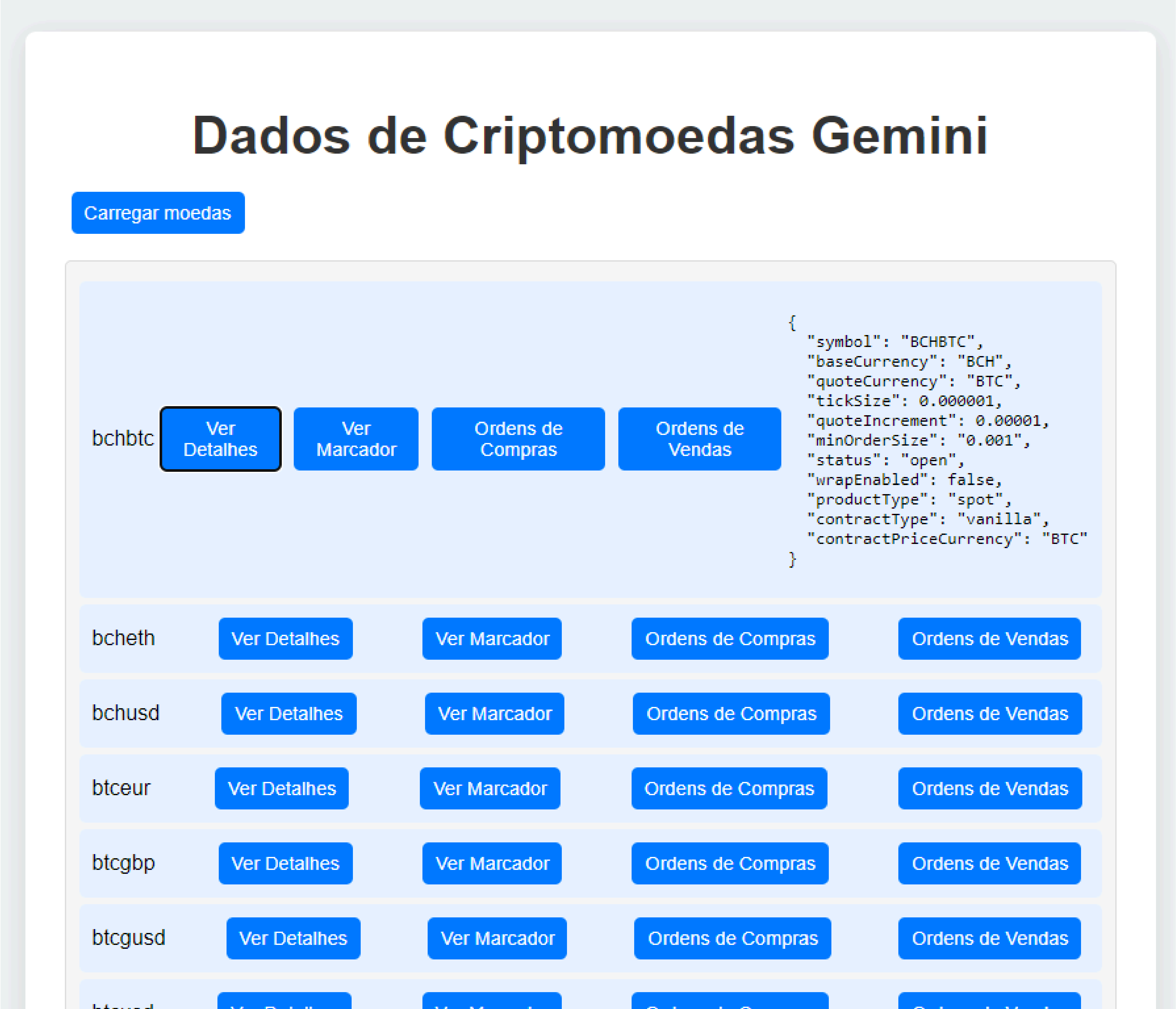


Dados de Criptomoedas Gemini

Carregar moedas

bchbtc	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
bcheth	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
bchusd	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
btceur	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
btcgbp	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
btcgusd	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
btcusd	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
btcusdt	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
dogebtc	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
dogeeth	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
dogeusd	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas

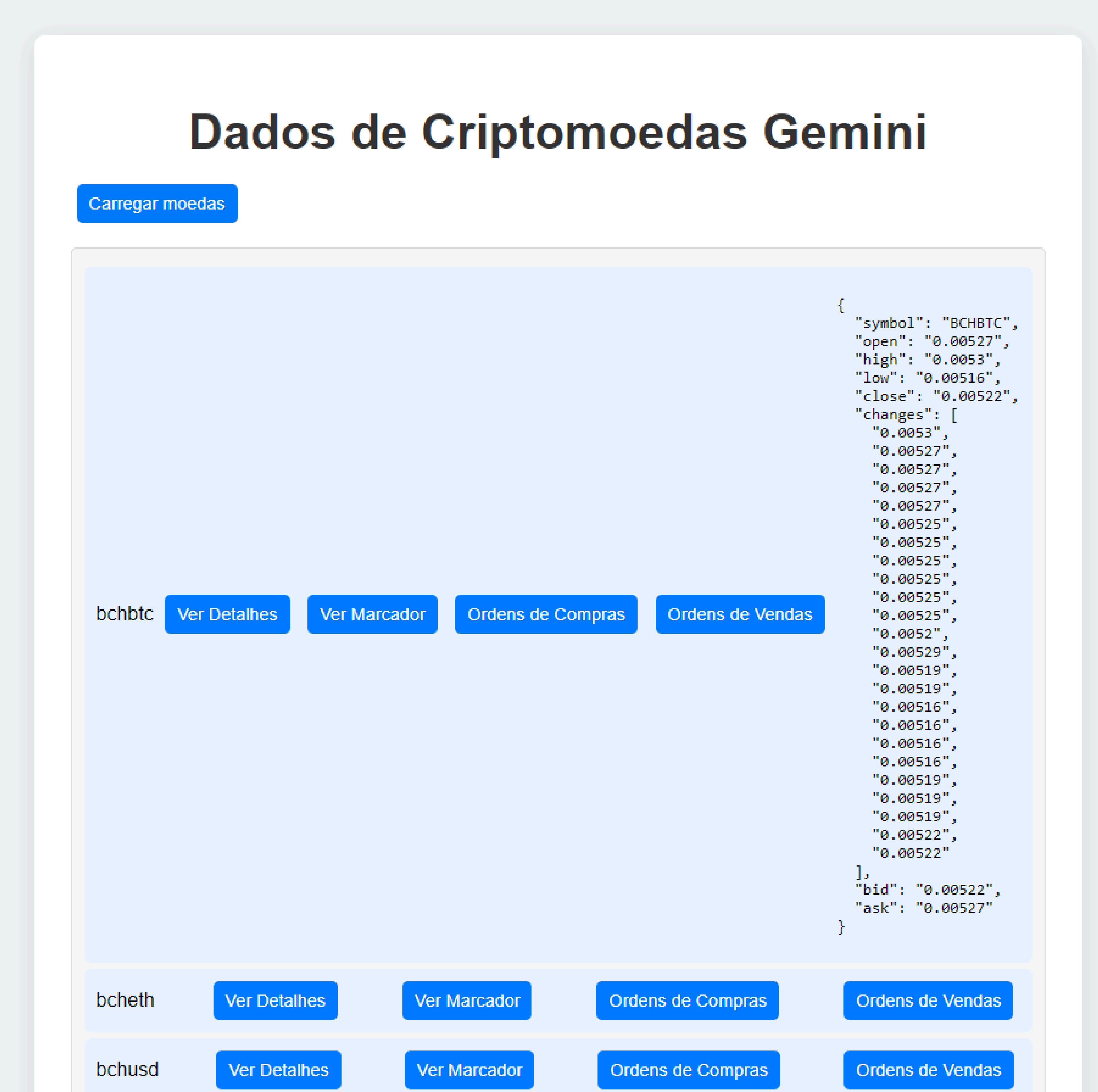
DETALHES DA MOEDA



Dados de Criptomoedas Gemini

```
{
  "symbol": "BCHBTC",
  "baseCurrency": "BCH",
  "quoteCurrency": "BTC",
  "tickSize": 0.000001,
  "quoteIncrement": 0.00001,
  "minOrderSize": "0.001",
  "status": "open",
  "wrapEnabled": false,
  "productType": "spot",
  "contractType": "vanilla",
  "contractPriceCurrency": "BTC"
}
```

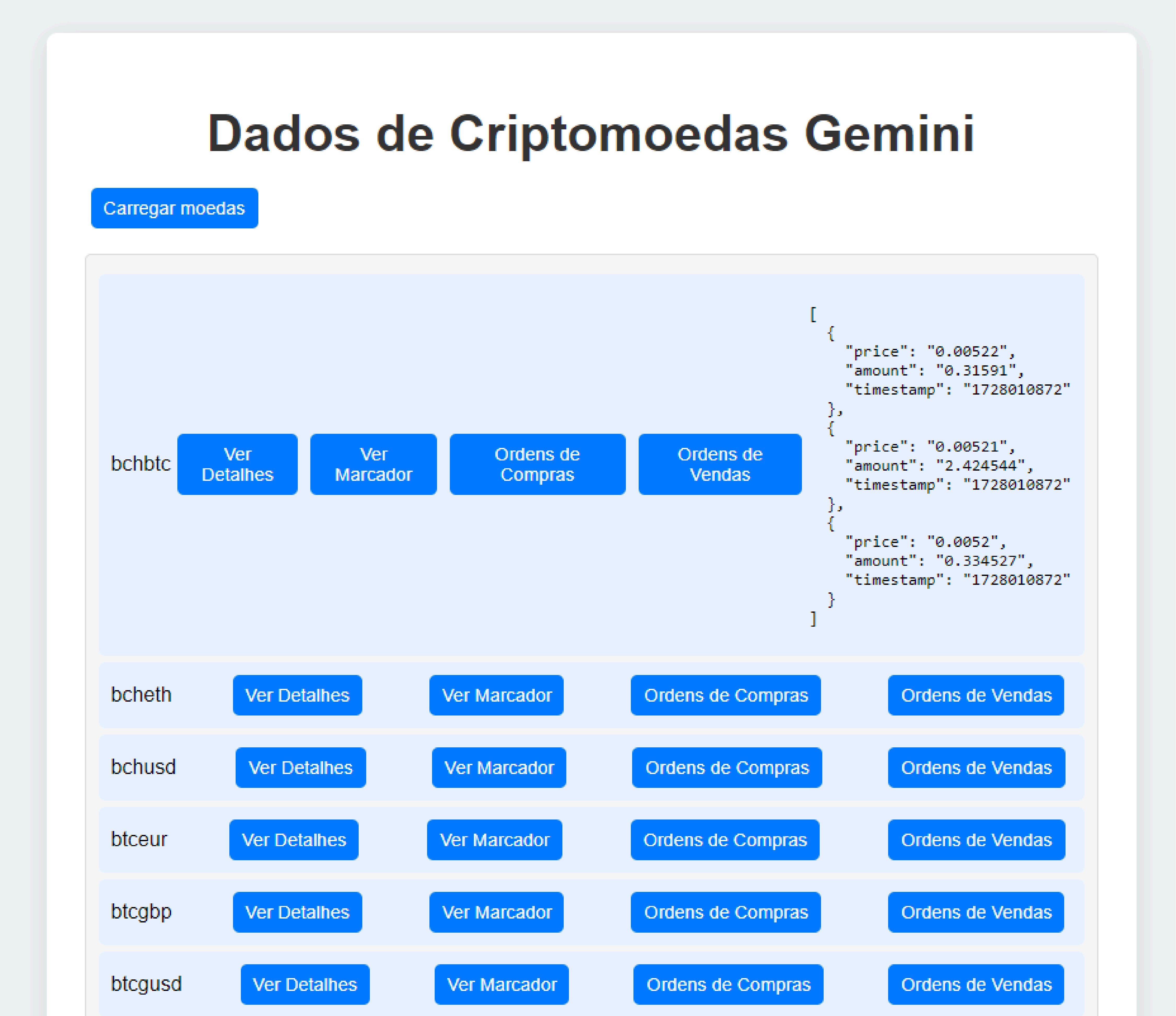
MARCADOR DA MOEDA



Dados de Criptomoedas Gemini

```
{
  "symbol": "BCHBTC",
  "open": "0.00527",
  "high": "0.0053",
  "low": "0.00516",
  "close": "0.00522",
  "changes": [
    "0.0053",
    "0.00527",
    "0.00527",
    "0.00527",
    "0.00527",
    "0.00525",
    "0.00525",
    "0.00525",
    "0.00525",
    "0.00525",
    "0.00525",
    "0.0052",
    "0.00529",
    "0.00519",
    "0.00519",
    "0.00516",
    "0.00516",
    "0.00516",
    "0.00516",
    "0.00519",
    "0.00519",
    "0.00519",
    "0.00522",
    "0.00522"
  ],
  "bid": "0.00522",
  "ask": "0.00527"
}
```

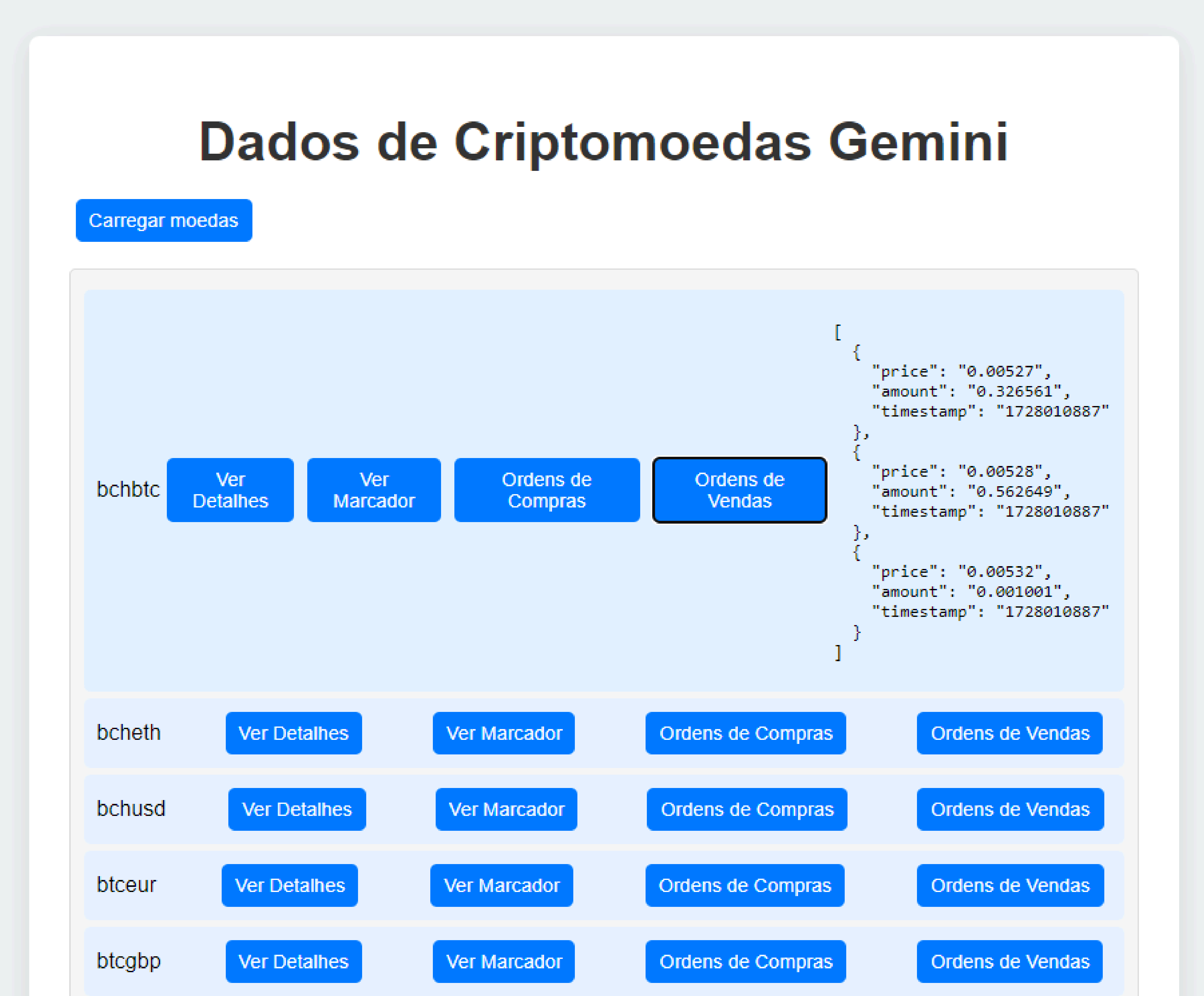
ORDENS DE COMPRA DA MOEDA



Dados de Criptomoedas Gemini

```
[
  {
    "price": "0.00522",
    "amount": "0.31591",
    "timestamp": "1728010872"
  },
  {
    "price": "0.00521",
    "amount": "2.424544",
    "timestamp": "1728010872"
  },
  {
    "price": "0.0052",
    "amount": "0.334527",
    "timestamp": "1728010872"
  }
]
```

ORDENS DE VENDA DA MOEDA



Dados de Criptomoedas Gemini

Carregar moedas

bchbtc	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
bcheth	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
bchusd	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
btceur	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas
btcgbp	Ver Detalhes	Ver Marcador	Ordens de Compras	Ordens de Vendas

```
[
  {
    "price": "0.00527",
    "amount": "0.326561",
    "timestamp": "1728010887"
  },
  {
    "price": "0.00528",
    "amount": "0.562649",
    "timestamp": "1728010887"
  },
  {
    "price": "0.00532",
    "amount": "0.001001",
    "timestamp": "1728010887"
  }
]
```

OBRIGADO!

ALUNO: VITOR FACCIO
RA: 04721-016