```
# bitget_api_client,py

import requests

import time

import hmac

import hashlib

import base64

import json

class BitgetClient:

    def __init__(self, api_key, api_secret,
passphrase):

        self.api_key = api_key
```

```
        self.api_secret = api_secret

        self.passphrase = passphrase

    def _get_timestamp(self):

        return str(int(time.time() * 1000))

    def _sign(self, timestamp, method, path,
body_str):

        message = f"{timestamp}{method}
{path}{body_str}"

        signature =
hmac.new(self.api_secret.encode(),
message.encode(),
hashlib.sha256).digest()

        return
base64.b64encode(signature).decode()
```

```
    def _request(self, method, path,
body=None):

        timestamp = self._get_timestamp()

        body_str = json.dumps(body) if body
else ""

        signature = self._sign(timestamp,
method, path, body_str)

        headers = {

            "ACCESS-KEY": self.api_key,

            "ACCESS-SIGN": signature,

            "ACCESS-TIMESTAMP": timestamp,

            "ACCESS-PASSPHRASE":
```

```
self.passphrase,

        "Content-Type": "application/json"

{

    url = f"https://api.bitget.com{path}"

    try:

        if method == "GET":

            response = requests.get(url,
headers=headers, timeout=10)

        elif method == "POST":

            response = requests.post(url,
headers=headers, data=body_str,
timeout=10)
```

```
        else:
            raise ValueError(f"Unsupported HTTP method: {method}")
        if response.status_code == 200:
            data = response.json()
            if data.get("code") == "00000":
                return data['data']
            else:
                print(f"[Bitget] API رد بخطأ: {data}")
        else:
            print(f"[Bitget] خطأ في الاستجابة:
```

```
{response.status_code} - {response.text}")

        except Exception as e:

            print(f"[Bitget] استثناء في الطلب: {e}")

        return None

    def get_ticker(self, symbol, market_type='spot'):

"""

الحصول على سعر آخر لـ symbol

        market_type: 'spot' أو 'onchain' (حسب النوع - حالياً endpoint spot فقط مدعوم هنا)

"""

        if market_type == 'spot':
```

```
            path = f"/api/spot/v1/market/ticker?symbol={symbol}"

        elif market_type == 'onchain':

# لو في endpoint خاص لـ onchain يمكن تعديله هنا

# لكن حسب ملفاتك الحالية يتم استخدام spot endpoint للسعر

            path = f"/api/spot/v1/market/ticker?symbol={symbol}"

        else:

            print(f"[BitgetClient] نوع السوق غير مدعوم: {market_type}")

            return None
```

```
        return self._request("GET", path)

    def place_order(self, symbol, price,
quantity, side, order_type,
market_type='spot'):

"""

تنفيذ أمر شراء أو بيع

        side: 'buy' أو 'sell'

        order_type: 'market' أو 'limit'

        market_type: spot أو onchain ( للتوسع
مستقبلاً)

"""

        path = "/api/spot/v1/trade/orders" #
```

spot order endpoint (مدعوم spot حالياً فقط)

جسم الطلب #

```
        body = {
            "symbol": symbol,
            "side": side,
            "orderType": order_type,
            "force": "gtc", # good till cancel
            "price": str(price) if price else "",
            "size": str(quantity)
{
```

يمكنك هنا إضافة دعم لـ onchain أو future #

لو كان هناك API خاص

```
        return self._request("POST", path, body)
```