```
import os
from datetime import datetime
from typing import Any, Dict, List

from plaid import Client
from plaid.api import plaid_api
from plaid.model.error import PlaidError
from plaid.model.transactions_get_request import (
    TransactionsGetRequest,
)
from plaid.model.transactions_get_response import (
    TransactionsGetResponse,
)

from swarms import Agent
from swarm_models import OpenAIChat
from swarms.prompts.finance_agent_sys_prompt import (
    FINANCIAL_AGENT_SYS_PROMPT,
)



def fetch_transactions(
    start_date: str, end_date: str
) -> List[Dict[str, Any]]:
    """
    Fetches a list of transactions from Plaid for a given time period.
```

```
Args:
    access_token (str): The access token associated with the Plaid item.
    start_date (str): The start date for the transaction query in 'YYYY-MM-DD' format.
    end_date (str): The end date for the transaction query in 'YYYY-MM-DD' format.

Returns:
    List[Dict[str, Any]]: A list of transactions as dictionaries.

Raises:
    PlaidError: If there is an error with the request to the Plaid API.
    ValueError: If the date format is incorrect.
"""
try:
    access_token = os.getenv("PLAID_ACCESS_TOKEN")
    # Validate date format
    datetime.strptime(start_date, "%Y-%m-%d")
    datetime.strptime(end_date, "%Y-%m-%d")

    # Initialize the Plaid client with your credentials
    plaid_client = plaid_api.PlaidApi(
        Client(
            client_id=os.getenv("PLAID_CLIENT_ID"),
            secret=os.getenv("PLAID_SECRET"),
            environment=os.getenv("PLAID_ENV", "sandbox"),
        )
```

```
        )

        # Create a request object for transactions
        request = TransactionsGetRequest(
            access_token=access_token,
            start_date=start_date,
            end_date=end_date,
        )

        # Fetch transactions from the Plaid API
        response: TransactionsGetResponse = (
            plaid_client.transactions_get(request)
        )

        # Return the transactions list
        return response.transactions

    except PlaidError as e:
        print(f"Plaid API Error: {e}")
        raise
    except ValueError as e:
        print(f"Date Format Error: {e}")
        raise


# Get the OpenAI API key from the environment variable
```

```
api_key = os.getenv("OPENAI_API_KEY")


# Create an instance of the OpenAIChat class
model = OpenAIChat(
    api_key=api_key, model_name="gpt-4o-mini", temperature=0.1
)


# Initialize the agent
agent = Agent(
    agent_name="Financial-Analysis-Agent_sas_chicken_eej",
    system_prompt=FINANCIAL_AGENT_SYS_PROMPT,
    llm=model,
    max_loops=1,
    autosave=True,
    # dynamic_temperature_enabled=True,
    dashboard=False,
    verbose=True,
    # interactive=True, # Set to False to disable interactive mode
    dynamic_temperature_enabled=True,
    saved_state_path="finance_agent.json",
    user_name="swarms_corp",
    # # docs=
    # # docs_folder="docs",
    retry_attempts=1,
    # context_length=1000,
    # tool_schema = dict
```

```
    context_length=200000,
    return_step_meta=False,
    tools=[fetch_transactions],
)



out = agent.run(
    "How can I establish a ROTH IRA to buy stocks and get a tax break? What are the criteria"
)
print(out)
```