Voici comment vous pourriez adapter votre code existant pour implémenter un serveur MCP qui se connecterait à Salesforce et SAP :

1. **Mise à jour de votre structure actuelle**

Votre setup actuel est un bon point de départ, mais pour implémenter MCP correctement, vous devrez :

- Ajouter les bibliothèques nécessaires pour MCP dans requirements.txt
- Modifier main.py pour implémenter le protocole MCP
- Créer des connecteurs pour Salesforce et SAP

2. **Mise à jour du requirements.txt**

```
fastapi
uvicorn[standard]
python-dotenv
requests
pydantic
simple-salesforce  # Pour l'API Salesforce
pyrfc           # Pour l'interface SAP RFC (si applicable)
websockets       # Pour les communications bidirectionnelles MCP
mcp-server       # Bibliothèque MCP d'Anthropic (si disponible)
```

3. **Modification de votre main.py pour supporter MCP**

Voici comment vous pourriez adapter votre code :

```
from fastapi import FastAPI, WebSocket, HTTPException
from pydantic import BaseModel
import os
import requests
import json
from dotenv import load_dotenv
from simple_salesforce import Salesforce
# Import pour SAP, selon la méthode de connexion choisie
```

```
load_dotenv()

ANTHROPIC_API_KEY = os.getenv("ANTHROPIC_API_KEY")
SALESFORCE_USERNAME = os.getenv("SALESFORCE_USERNAME")
SALESFORCE_PASSWORD = os.getenv("SALESFORCE_PASSWORD")
SALESFORCE_SECURITY_TOKEN = os.getenv("SALESFORCE_SECURITY_TOKEN")
SALESFORCE_DOMAIN = os.getenv("SALESFORCE_DOMAIN", "login")
# Ajoutez les variables SAP ici

app = FastAPI()

# Initialisation des connexions
sf = Salesforce(username=SALESFORCE_USERNAME,
password=SALESFORCE_PASSWORD,
        security_token=SALESFORCE_SECURITY_TOKEN,
domain=SALESFORCE_DOMAIN)
# Initialisation SAP ici

class MCPRequest(BaseModel):
  action: str
  parameters: dict

# Point d'entrée pour la connexion WebSocket MCP
@app.websocket("/mcp")
async def mcp_endpoint(websocket: WebSocket):
  await websocket.accept()

  # Envoyer les capacités du serveur
  capabilities = {
```

```
    "server_info": {
        "name": "Custom MCP Server",
        "version": "1.0.0"
    },
    "tools": {
        "salesforce": {
            "description": "Accès aux données Salesforce",
            "operations": ["query", "create", "update", "delete"]
        },
        "sap": {
            "description": "Accès aux données SAP",
            "operations": ["read", "write", "execute"]
        }
    }
}

await websocket.send_json({"type": "capabilities", "data": capabilities})

try:
    while True:
        data = await websocket.receive_text()
        request = json.loads(data)

        if request["type"] == "request":
            response = await handle_mcp_request(request["data"])
            await websocket.send_json({"type": "response", "data": response})
except Exception as e:
    await websocket.close(code=1000)
```

```
async def handle_mcp_request(request):
    action = request.get("action")
    params = request.get("parameters", {})

    if action == "salesforce.query":
        return {"result": execute_salesforce_query(params.get("query", ""))}
    elif action == "sap.read":
        return {"result": execute_sap_read(params.get("table"), params.get("criteria", {}))}
    # Autres actions...
    else:
        return {"error": "Action non supportée"}

def execute_salesforce_query(query):
    try:
        result = sf.query(query)
        return result
    except Exception as e:
        return {"error": str(e)}

def execute_sap_read(table, criteria):
    # Implémentation de l'accès SAP
    pass

# Gardez votre API existante comme point de test/diagnostic
@app.get("/")
def hello():
    return {"message": "Serveur MCP opérationnel"}
```

```
@app.post("/claude")
def ask_claude(request: MessageRequest):
  # Votre code existant...
```

4. **Dockerfile mis à jour**

Votre Dockerfile actuel fonctionne bien, mais vous pourriez vouloir exposer le port WebSocket pour MCP :

```
FROM python:3.10-slim

WORKDIR /app

COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

COPY . .

# Exposer le port pour FastAPI et pour MCP WebSocket
EXPOSE 8000

CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "8000"]
```

5. **Fichier docker-compose.yml**

```
version: '3'
services:
 mcp-server:
  build: .
  ports:
   - "8000:8000"
  volumes:
```

```
    - .:/app
    environment:
      - ANTHROPIC_API_KEY=${ANTHROPIC_API_KEY}
      - SALESFORCE_USERNAME=${SALESFORCE_USERNAME}
      - SALESFORCE_PASSWORD=${SALESFORCE_PASSWORD}
      - SALESFORCE_SECURITY_TOKEN=${SALESFORCE_SECURITY_TOKEN}
      - SALESFORCE_DOMAIN=${SALESFORCE_DOMAIN}
      # Variables SAP ici
```

Pour tester ce système, vous devrez :

1. Créer un fichier .env avec toutes les informations d'authentification
2. Construire et lancer votre conteneur Docker
3. Configurer un client MCP (comme Claude Desktop) pour se connecter à votre serveur

Notez que cette implémentation est simplifiée. Les détails exacts de l'implémentation MCP peuvent nécessiter des ajustements en fonction de la spécification officielle la plus récente.

Avez-vous des questions sur des parties spécifiques de cette implémentation ou avez-vous besoin d'aide pour configurer les connecteurs Salesforce ou SAP en particulier ?