继续用 Flask、Redis 等扩展它，变成一个真实网络服务的 2PC 示例

ChatGPT said:

Updated saved memory

下面是一个基于 **Flask + Redis** 的简单真实网络服务版 **Two-Phase Commit（2PC）** 示例，包括：

- **协调者服务**（Coordinator）
- **参与者服务**（两个 Flask 服务模拟库存和账户）
- **Redis**：模拟持久化状态（如事务准备状态）

📦 项目结构

```
bash
CopyEdit
2pc_demo/
├── coordinator.py          # 协调者服务
├── participant_a.py        # 参与者 A（如库存服务）
├── participant_b.py        # 参与者 B（如账户服务）
└── requirements.txt
```

✅ 0. 安装依赖

```
bash
CopyEdit
pip install flask redis requests
```

确保你本地有一个 Redis 服务运行：

```
bash
CopyEdit
redis-server
```

✅ 1. 参与者服务（如库存服务 / participant_a.py）

```
python
```

CopyEdit

```
from flask import Flask, request, jsonify
import redis

app = Flask(__name__)
r = redis.Redis()

SERVICE_NAME = "participant_a"

@app.route('/prepare', methods=['POST'])
def prepare():
    data = request.json
    txn_id = data['txn_id']
    can_commit = data.get('can_commit', True)

    if can_commit:
        r.set(f"{SERVICE_NAME}:{txn_id}:prepared", 1)
        return jsonify({"status": "prepared"})
    else:
        return jsonify({"status": "abort"}), 400

@app.route('/commit', methods=['POST'])
def commit():
    txn_id = request.json['txn_id']
    if r.get(f"{SERVICE_NAME}:{txn_id}:prepared"):
        r.delete(f"{SERVICE_NAME}:{txn_id}:prepared")
        return jsonify({"status": "committed"})
    return jsonify({"status": "not_prepared"}), 400

@app.route('/rollback', methods=['POST'])
def rollback():
    txn_id = request.json['txn_id']
    r.delete(f"{SERVICE_NAME}:{txn_id}:prepared")
    return jsonify({"status": "rolled_back"})

if __name__ == '__main__':
    app.run(port=5001)
```

✅ 2. 另一个参与者服务（账户服务 / participant_b.py）

只需要改 SERVICE_NAME 和端口号即可：

python
CopyEdit
```
# 只修改这些
SERVICE_NAME = "participant_b"
...
app.run(port=5002)
```

✅ 3. 协调者服务（coordinator.py）

python
CopyEdit
```
import requests
import uuid

participants = {
    "participant_a": "http://localhost:5001",
    "participant_b": "http://localhost:5002"
}

def coordinator_transaction():
    txn_id = str(uuid.uuid4())
    print(f"Start 2PC Transaction: {txn_id}")

    # Phase 1: Prepare
    print("Phase 1: Sending prepare...")
    prepared = True
    for name, url in participants.items():
        try:
            res = requests.post(f"{url}/prepare",
json={"txn_id": txn_id})
            if res.status_code != 200:
                print(f"{name} voted ABORT.")
                prepared = False
                break
            print(f"{name} voted YES.")
        except Exception as e:
            print(f"{name} failed: {e}")
            prepared = False
```

```
            break

    # Phase 2: Commit or Rollback
    if prepared:
        print("All participants are ready. Sending
COMMIT.")
        for name, url in participants.items():
            requests.post(f"{url}/commit", json={"txn_id":
txn_id})
        print("✅ Transaction committed.")
    else:
        print("Something went wrong. Sending ROLLBACK.")
        for name, url in participants.items():
            requests.post(f"{url}/rollback",
json={"txn_id": txn_id})
        print("❌ Transaction rolled back.")

if __name__ == '__main__':
    coordinator_transaction()
```

✅ 4. 启动并测试

```
bash
CopyEdit
# 启动参与者服务
python participant_a.py
python participant_b.py

# 在新终端运行协调者
python coordinator.py
```

✅ 示例输出

```
bash
CopyEdit
Start 2PC Transaction: 9d83-...
Phase 1: Sending prepare...
participant_a voted YES.
```

```
participant_b voted YES.
All participants are ready. Sending COMMIT.
✅ Transaction committed.
```

若你在某个参与者 `prepare` 中模拟失败（如返回 400），协调者将发出 rollback。

📌 总结：你学到了什么？

技术点	用途
Flask	模拟微服务/参与者 API
Redis	持久化 prepared 状态
requests	协调者调用各参与者的 REST 接口
UUID	唯一事务 ID
2PC 协议流程	Prepare → Commit / Rollback