1. **Configuração do Redis**:

* Confirmar que o Redis está instalado e rodando localmente no Ubuntu (você mencionou que já está configurado).
* Instalar a biblioteca Python redis-py no ambiente virtual (pip install redis).
* Configurar a conexão com o Redis no app.py usando redis.Redis(host='localhost', port=6379, db=0).

2. **Modificação dos Módulos**:

* Ajustar portfolio\_services.py (ou equivalente em Findash.services.portfolio\_services) para cachear dados brutos (ex.: preços de tickers via yfinance) no Redis.
* Criar chaves únicas no Redis baseadas em tickers, quantidades e períodos (ex.: ticker:PETR4.SA:2023-01-01:2025-04-27) para evitar desalinhamentos.
* Cachear métricas calculadas (ex.: retornos, valores do portfólio) em calcular\_metricas ou funções similares, usando JSON para serializar DataFrames do pandas.
* Definir um tempo de expiração para os dados no Redis (ex.: 24 horas) para garantir atualizações.

3. **Integração com Flask**:

* Adicionar um cliente Redis global no app.py para acesso em todas as rotas e serviços.
* Modificar a rota /dashboard para verificar o cache antes de calcular o portfólio, usando redis.get() e redis.set() para leitura/escrita.
* Garantir que os dados cacheados sejam desserializados corretamente (ex.: json.loads() para converter JSON em dicionários Python).

4. **Sincronização com dcc.Store**:

* Não sincronizar diretamente Redis com dcc.Store, pois o Redis armazena dados no backend e o dcc.Store gerencia o estado no frontend.
* Continuar usando o dcc.Store para armazenar temporariamente o portfólio no cliente (como já faz na rota /dash\_entry).

5. **Testes**:

* Testar localmente com um portfólio pequeno (ex.: PETR4.SA, VALE3.SA) para verificar se os dados são cacheados e recuperados corretamente.
* Simular cenários de desalinhamento (ex.: mudar datas ou tickers) para garantir que o cache seja invalidado quando necessário.
* Verificar a performance (tempo de resposta) com e sem cache, usando logs para comparar.

6. **Deploy no Oracle Cloud**:

* Configurar o Redis na VM do Oracle Cloud Free Tier (instalar via sudo apt install redis-server).
* Ajustar o app.py para conectar ao Redis na nuvem (host='IP\_DA\_VM', port=6379).
* Adicionar um script keep-alive (ex.: ping periódico) para evitar que a VM pare após 7 dias de inatividade.
* Testar o deploy com um portfólio completo, verificando a integração com o Flask e o Dash.