perplexity

instead of redis.opsForSet(), use rep api

To replace the raw `redis.opsForSet()` usage with a reactive repository (rep) API, you should extend your repository interface or create a custom repository to handle the secondary index operations.

Repository-based approach for secondary index

1. **Define a repository for managing the secondary index set** e.g., `PlacesByQuoteRepository`

```
package com.example.pricing.repository;

import org.springframework.data.repository.reactive.ReactiveCrudRepository;
import org.springframework.stereotype.Repository;
import reactor.core.publisher.Flux;
import reactor.core.publisher.Mono;

@Repository
public interface PlacesByQuoteRepository {

    Mono<Long> addPlaceKeyToQuoteSet(String quoteId, String placeKey);

    Flux<String> getPlaceKeysByQuoteId(String quoteId);

    Mono<Long> removePlaceKeyFromQuoteSet(String quoteId, String placeKey);
}
```

2. **Implement this interface with Redis reactive commands under the hood**

```
import org.springframework.data.redis.core.ReactiveStringRedisTemplate;
import org.springframework.stereotype.Repository;
import reactor.core.publisher.Flux;
import reactor.core.publisher.Mono;

@Repository
public class PlacesByQuoteRepositoryImpl implements PlacesByQuoteRepository {

    private final ReactiveStringRedisTemplate redisTemplate;

    public PlacesByQuoteRepositoryImpl(ReactiveStringRedisTemplate redisTemplate) {
        this.redisTemplate = redisTemplate;
    }

    @Override
    public Mono<Long> addPlaceKeyToQuoteSet(String quoteId, String placeKey) {
        return redisTemplate.opsForSet().add("places-by-quote:" + quoteId, placeKey);
    }

    @Override
    public Flux<String> getPlaceKeysByQuoteId(String quoteId) {
        return redisTemplate.opsForSet().members("places-by-quote:" + quoteId);
    }

    @Override
    public Mono<Long> removePlaceKeyFromQuoteSet(String quoteId, String placeKey) {
        return redisTemplate.opsForSet().remove("places-by-quote:" + quoteId, placeKey);
    }
}
```

3. **Usage in your service**

```
// To add place key
placesByQuoteRepository.addPlaceKeyToQuoteSet(quoteId, placeKey).subscribe();

// To get place keys
placesByQuoteRepository.getPlaceKeysByQuoteId(quoteId)
    .flatMap(placeRepository::findById)
    .subscribe(placeStatus -> { ... });
```

Summary

This lets you keep a clean repository API over your Redis sets rather than calling `opsForSet()` directly in your service logic and fits better with Spring Reactive idioms while managing your secondary quote-to-place index.