public Rewards calculateRewards(List<Transaction> transactions, int customerId) {

Rewards rewards = new Rewards();

rewards.setCustomerId(customerId);

Map<String, Integer> monthlyRewards = new HashMap<>();

int totalRewards = 0;

for (Transaction transaction : transactions) {

// Only consider transactions made by the specified customer

if (transaction.getCustomerId() != customerId) {

continue;

}

int amount = transaction.getAmount();

int rewardPoints = 0;

if (amount > 100) {

rewardPoints += 2 \* (amount - 100);

}

if (amount > 50) {

rewardPoints += 1 \* Math.min(amount - 50, 100);

}

// Add the reward points to the total and to the monthly count

totalRewards += rewardPoints;

String month = transaction.getDate().format(DateTimeFormatter.ofPattern("yyyy-MM"));

monthlyRewards.put(month, monthlyRewards.getOrDefault(month, 0) + rewardPoints);

}

rewards.setMonthlyRewards(monthlyRewards);

rewards.setTotalRewards(totalRewards);

return rewards;

}

---------------------------------------------------------------------------------

@RestController

@RequestMapping("/rewards")

public class RewardsController {

private final RewardsService rewardsService;

public RewardsController(RewardsService rewardsService) {

this.rewardsService = rewardsService;

}

@PostMapping

public Rewards calculateRewards(@RequestBody CalculateRewardsRequest request) {

return rewardsService.calculateRewards(request.getTransactions(), request.getCustomerId());

}

}

[

{

"id": 1,

"customerId": 1,

"amount": 120,