Тестовое задание. Back-End

Задание: Разработка упрощенной CRM-системы

Описание задачи:

Вы должны разработать CRM-систему, которая будет управлять информацией о продавцах и их транзакциях. Система должна включать возможности для создания, чтения, обновления и удаления данных о продавцах и транзакциях. Система также должна включать функции аналитики для обработки и анализа данных.

Данные по задаче:

Сущности:

1. **Продавец (Seller):**
    - **ID (id):** уникальный идентификатор продавца (целое число, автоинкремент).
    - **Имя (name):** имя продавца (строка).
    - **Контактные данные (contactInfo):** контактная информация продавца (строка).
    - **Дата регистрации (registrationDate):** дата и время регистрации продавца в системе (тип LocalDateTime).
2. **Транзакция (Transaction):**
    - **ID (id):** уникальный идентификатор транзакции (целое число, автоинкремент).
    - **Продавец (seller):** ссылка на продавца, к которому относится транзакция (внешний ключ на сущность "Продавец").
    - **Сумма (amount):** сумма транзакции (десятичное число).
    - **Тип оплаты (paymentType):** тип оплаты (CASH, CARD, TRANSFER) (строка).
    - **Дата транзакции (transactionDate):** дата и время совершения транзакции (тип LocalDateTime).

Аналитика:

- Вывести самого продуктивного продавца в рамках дня, месяцы, квартала, года (самый продуктивный, тот у которого сумма всех транзакции больше всех других продавцов)
- Вывести список продавцов, у которых сумма всех транзакции за выбранный период меньше переданного параметра суммы
- *Сложная задача (не обязательная):** Реализуйте алгоритм для определения наилучшего периода времени (например, диапазона дат) для конкретного продавца, в течение которого он совершил наибольшее количество транзакций. Этот период может быть любого размера (например, день, неделя, месяц) и должен быть найден с учётом всех данных транзакций.

Реализация API должна содержать методы (REST):

Список всех продавцов Инфо о конкретном продавце Создать нового продавца Обновить инфо о продавце Удалить продавца Получить список всех транзакций Получить информацию о конкретной транзакции Создать новую транзакцию Получить все транзакции продавца Получить самого продуктивного продавца Получить список продавцов с суммой меньше указанной
*Получить самое продуктивное время продавца

Требования:

- **Язык программирования:**
  - Вы можете использовать **Java** или **Kotlin** для реализации задачи.
- **Фреймворк:**
  - Используйте **Spring Boot** для создания RESTful API.
- **База данных:**
  - Используйте **PostgreSQL** как основную базу данных. Допустимо использовать H2.
  - Для тестирования допускается использование **H2** базы данных в режиме inmemory.
- **Сущности и связи:**
  - Придерживайтесь принципов **сохранения историчности данных**.
- **Код и архитектура:**
  - Придерживайтесь принципов **ООП** и используйте **паттерны проектирования** там, где это целесообразно.
  - Обеспечьте **читаемость** и **поддерживаемость** кода:
    - Используйте **читаемые имена переменных и методов**.
    - Структурируйте проект в соответствии с принципами **разделения ответственности**.
- **Тестирование:**
  - Напишите **юнит-тесты** для критически важной логики приложения, покрыв минимум **50% кода**.
  - Обеспечьте тестирование всех основных сценариев использования API.
- **Обработка ошибок:**
  - Реализуйте централизованную обработку ошибок и возвращайте осмысленные коды и сообщения при возникновении ошибок.
- **Документация проекта:**
  - Создайте документ (README) с описанием проекта, включающий функциональность, инструкции по сборке и запуску, примеры использования API и описание необходимых зависимостей.
  - Используйте **Markdown** для оформления документации.
- **Управление зависимостями и сборщик:**
  - Используйте gradle как инструмент для управления зависимостями и сборка проекта