Тестовое задание T4. C#

Оценочное время на выполнение задания: 3-4 часа. Общее время на выполнение задания: 7 календарных дней с момента выдачи. **Сделать требуется только одно любое задание из приложенных в архив на выбор.**

Результат выполнения прислать в чат Skype, в котором было выдано задание. Исходный код без бинарных файлов архивом или ссылкой на публичный репозиторий GitHub/GitLab/т.д.

Дан `interface`, описывающий финансовую транзакцию:

```
public interface ITransaction
{
    Guid Id { get; }

    Guid ClientId { get; }

    DateTime DateTime { get; }

    decimal Amount { get; }
}
```

Создайте реализации этого интерфейса CreditTransaction (зачисление средств клиенту), DebitTransaction (списание средств у клиента). Напишите Web API, позволяющее зачислять и списывать средства у указанного клиента, отменять указанную транзакцию и получать актуальный баланс клиента. В сервисе должно быть 4 запроса:

```
POST /credit
{
    "id":"8f0452b2-867b-4ef8-9a9d-3c9c03d9afdf",
    "clientId": "cfaa0d3f-7fea-4423-9f69-ebff826e2f89",
    "dateTime":"2019-04-02T13:10:20.0263632+03:00",
    "amount":23.05
}
Response 200 OK:
{
    "insertDateTime": "2024-10-25T12:03:34+05:00",
    "clientBalance": 23.05
}
```

```
POST /debit
{
    "id":"05eb235c-4955-4c16-bcdd-34e8178228de",
    "clientId": "cfaa0d3f-7fea-4423-9f69-ebff826e2f89",
    "dateTime":"2019-04-02T13:10:25.0263632+03:00",
```

```
    "amount":23.05
}
Response 200 OK:
{
    "insertDateTime": "2024-10-25T12:03:34+05:00",
    "clientBalance": 0
}
```

```
POST /revert?id=05eb235c-4955-4c16-bcdd-34e8178228de
Response 200 OK:
{
    "revertDateTime": "2024-10-25T12:03:34+05:00",
    "clientBalance": 23.05
}
```

```
GET /balance?id=cfaa0d3f-7fea-4423-9f69-ebff826e2f89
Response 200 OK:
{
    "balanceDateTime": "2024-10-25T12:03:34+05:00",
    "clientBalance": 23.05
}
```

- Методы POST должны быть идемпотентными: при повторной отправке запроса он должен возвращать результат отправки предыдущего запроса с таким же Id.
- Необходимо реализовать валидацию: сумма в транзакциях должна быть всегда положительной, дата не может быть в будущем, у клиента нельзя списать больше средств, чем есть на балансе.
- Обработку ошибок необходимо реализовать в соответствии с RFC 9457.
- Способ хранения данных - в реляционной СУБД на усмотрение соискателя, предпочтительно - PostgreSQL или SQL Server.
- Крайне не рекомендуется ограничивать работоспособность сервиса одним инстансом (ничто не должно препятствовать запуску двух и более инстансов сервиса).
- Способ логирования - на усмотрения соискателя.
- Дополнительная функциональность - на усмотрение соискателя.
- Приветствуется использование docker compose для быстрого запуска тестового задания.