Тестовое задание для разработчика: Full-stack приложение с Event Sourcing и историей версий

Общие требования

Напишите full-stack приложение на TypeScript, состоящее из frontend и backend-частей. Приложение должно демонстрировать работу архитектурного подхода Event Sourcing и предоставлять функциональность истории версий на его основе.

Этапы и компоненты

1. Frontend

Интерфейс:

- Используйте canvas для отрисовки графических объектов.

- Минимальная графика — квадраты, которые можно перетаскивать мышкой.

- Справа от canvas расположить панель с историей версий:
  • Каждая версия содержит метку времени.
  • Версии отображаются в виде кликабельных элементов списка.
  • При выборе версии отображение на canvas должно соответствовать состоянию на тот момент.

Функциональность:

- При завершении действия пользователя (например, отпускании мышки после перемещения квадрата), отправляйте событие на сервер через WebSocket.

- После получения подтверждения от сервера обновляйте список версий.

Ограничения:

- Использование React — необязательно.

- Использование Canvas API (или обёрток) — обязательно.

2. Backend

Архитектура:

- Реализовать на NestJS.
- Обеспечить работу через WebSocket.
- Использовать PostgreSQL для хранения данных.

Функциональность:

- Получение полного состояния (на основе всех ивентов).
- Получение состояния до определённого события (например, по ID или дате).
- Получение списка всех версий (событий).
- При получении события от фронта сохранять событие в базе данных

Типы событий (примерные):

- SquareCreated
- SquareMoved
- SquareDeleted

Требования к хранилищу:

- События должны храниться отдельно (event store).
- Состояние должно вычисляться on-the-fly, при запросе, а не кэшироваться в базе.
- Отдельно хранить снимки, например каждый 10-ый ивент. Использовать их, для более быстрого восстановления состояния (опционально)

Дополнительные требования

- Реализация кода на TypeScript на всех уровнях.

- Хорошая архитектурная организация кода (модули, слои).

- Приветствуется покрытие основного бизнес-алгоритма (event reducer) тестами.

- Программа должна обрабатывать ошибки: например, невалидные события или попытка запросить несуществующую версию.

Ожидаемый результат

- Исходный код на GitHub с README и инструкциями по запуску.

- Демонстрация работы (видеозапись или развернутое приложение на Vercel/Render/Heroku и пр.).

Выполненные работы принимаются до 15го июня включительно по ссылке https://forms.yandex.ru/u/683d558ceb6146ace98c1b49/

Контакт для связи @imsergeykrasnov (Telegram)