Тестовое задание Node.js Dev

Цель: разработать приложение для игры в крестики-нолики с регистрацией пользователей, поддержкой многопользовательского режима, и ведением статистики побед.

Основные требования:

1. **Регистрация пользователя:**

• Пользователю предлагается зарегистрироваться, указав имя на латинице.
• Имя пользователя должно быть уникальным.
• После регистрации пользователь может подключиться к игре.

2. **Подключение к игре:**

• Игроки подключаются к игре через WebSocket.
• В одной игре участвуют два игрока. Один играет «крестиками», другой – «ноликами».
• Начинает игру игрок, ставящий «крестики».
• Игровое поле имеет размер NxN, где N задается через переменную окружения DESK_SIZE. По умолчанию размер поля 3x3.
• Партия продолжается до тех пор, пока один из игроков не выстроит в ряд свои знаки по вертикали, горизонтали или диагонали.

3. **Победа и завершение игры:**

• Для победы необходимо выиграть три партии.
• В случае ничьи партия не засчитывается ни одному из игроков.
• После завершения всех партий, игра считается оконченной, и результат записывается в базу данных (опционально).

4. **Статистика и лидеры:**

• Должен быть реализован HTTP эндпоинт, который возвращает список игроков, отсортированных по количеству побед в порядке убывания (опционально).

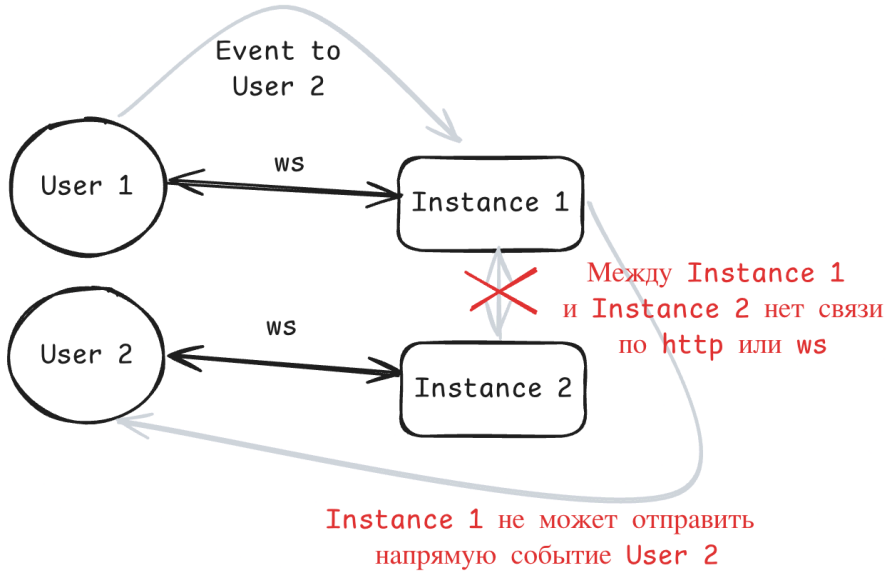
Дополнительные требования:

1. **Поддержка нескольких инстансов:**

• Приложение должно поддерживать запуск нескольких инстансов за балансировщиком, при этом WebSocket и HTTP запросы могут поступать в разные инстансы.

• Требуется обеспечить корректную работу игры в таких условиях.

В идеале надо решить такую проблему:



2. **База данных:**

• Использовать любую реляционную базу данных для сохранения результатов игр и статистики игроков.
• Структура базы данных на усмотрение разработчика.

Рекомендации:

• Использовать **TypeScript** для разработки.
• Использовать **Express.js** или **Nest.js** для разработки.
• Для реализации WebSocket взаимодействия использовать **Socket.IO** или аналогичные библиотеки.
• Разграничивать ответственность в коде
• Для работы с базой данных можно использовать **ORM** (например, **Knex**, **Prisma**, **TypeORM**).
• При написании кода использовать тесты (можно использовать **Jest**).

Критерии оценки:

• Корректность и полнота выполнения задания.
• Структура и качество кода.
• Способность работать с WebSocket и базами данных.
• Умение проектировать системы, которые могут масштабироваться.

Срок выполнения: 7 дней

Пожалуйста, добавьте к решению инструкции по запуску и описание архитектуры приложения в README.