Тестове завдання

Завдання: Розробити full-stack застосунок з авторизацією, відображенням списку елементів після успішного логіну користувача, а також інтеграцією WebSocket для взаємодії між фронтендом і бекендом.

Технології обов'язкові до використання:

- ***Front-end:*** *React.js, Redux, React Query (для http запитів), React Router або аналоги.*
- ***Back-end:*** *Node.js, Express, MongoDb (mongoose).*
- ***WebSocket:*** *Використати для реального часу (наприклад, для оновлення ігор, онлайн-статусів користувачів, сповіщень про нові коментарі і т.д.).*

1. Авторизація користувача

1. *Користувач потрапляє на сторінку авторизації, реєструється або входить в особистий кабінет (авторизацію розробляєте на свій погляд, бажано з шифруванням паролю користувача при записі в БД).*
2. *При авторизації користувач має бачити помилки у разі:*
    - *Неправильного пароля чи пошти.*
    - *Якщо такий користувач вже існує або пошта зайнята.*

2. Сторінка з іграми

1. *Після успішної авторизації користувач потрапляє на сторінку з іграми, де є фільтрація за жанрами або датою.*
2. *На сторінці відображається список ігор з такими жанрами:*
    - *"ALL", "FREE", "MOBA", "SHOOTERS", "LAUNCHERS", "MMORPG", "STRATEGY", "FIGHTING", "RACING", "SURVIVAL", "ONLINE"*
3. *За замовчуванням показано 9 елементів, при натисканні кнопки «Показати ще» довантажуються ще 9 і так далі.*
4. *Коли всі ігри завантажено, кнопка «Показати ще» зникає.*

---

3. Реалізація WebSocket для реального часу

1. ***Реальний час:*** *При додаванні нової гри на сервері, список ігор має оновлюватися на всіх підключених клієнтах без оновлення сторінки.*

2. ***Інші варіанти перевірки роботи WebSocket:***

    - ***Онлайн користувачі:*** *Вивести список користувачів, які зараз онлайн. Якщо користувач підключається або відключається, список має оновлюватися у реальному часі.*

    - ***Сповіщення про нові коментарі до ігор:*** *Коли користувач додає новий коментар до гри, він автоматично з'являється у всіх відкритих вкладках.*

    - ***Популярні ігри:*** *Якщо гра потрапляє в топ (наприклад, поле* `inTop: true` *змінюється в БД), користувачі мають отримувати сповіщення у реальному часі.*

Як перевірити:

- *Відкрити сторінку зі списком ігор у двох вкладках.*
- *Додати нову гру через API або адмінку.*
- *Перевірити, що нова гра з'являється в обох вкладках без оновлення сторінки.*

---

4. API для отримання ігор

Отримання списку ігор за допомогою API:

- *URL:* `https://api-test.miraplay.cloud/games`

5. Дизайн і шрифт

Проект має частково або повністю відповідати дизайну сайту https://miraplay.cloud/.
Обов'язковий шрифт: ***Neue Machina****.*

---

6. Деплой

1. *Деплоїти Front-end і Back-end частини на будь-який сервіс для деплою на ваш вибір.*

2. *Вказати посилання на живу сторінку Front-end і репозиторії для Front-end і Back-end.*

Час на виконання:

4 днів (вихідні дні не враховуються).

Успіхів у виконанні тестового завдання!

Додаткові зауваження:

1. *Якщо хочеш додати ще додаткові функції або перевірки, можна використовувати WebSocket для більш глибоких інтерфейсних оновлень (наприклад, для матчмейкінгу, лічильників активних користувачів і т.д.).*

2. *Важливо продемонструвати правильну інтеграцію WebSocket з реальним часом у контексті роботи з іграми та взаємодії між користувачами.*