Üzenőfal alkalmazás (React)

Ebben a feladatban egy üzenőfal alkalmazást kell készítened React-ban, mellyel a felhasználók üzeneteket tehetnek közzé, illetve likeolhatnak üzeneteket. Az alkalmazás egy előre elkészített backenddel kommunikál REST API-n keresztül.

Hasznos linkek:

React dokumentáció: https://react.dev/

MUI (Material UI): https://mui.com/

Axios: https://axios-http.com/docs/intro

A frontend projekt már elő van készítve a proxy használatára (Vite `vite.config.js` fájlban), így az API hívásokat a `/api/...` útvonalon keresztül lehet elvégezni. Ha ez valamilyen okból nem működne, a backend CORS modullal is fel van készítve, ezért nyugodtan használhatod a teljes URL-t is, pl. `http://localhost:3333/api/…`.

A frontend projekt támogatja a **tailwindcss** használatát.

A projektkönyvtárban elérhető lesz egy vagy több képernyőkép, amely megmutatja, hogyan nézzen ki az elkészült alkalmazás végleges formája.

A backend minden POST kérés után a teljes, frissült entitás listát visszaadja. Ez azért van így, hogy a változások könnyebben, egyértelműen újrarenderelhetők legyenek a frontend oldalon.

Backend indítása:

1. Navigálj a `backend` mappába.
2. Futtasd az `npm install` parancsot.
3. Indítsd el a szervert: `npm start`

A backend ezután a http://localhost:3333 címen lesz elérhető.

Swagger API dokumentáció: http://localhost:3333/api-docs

Feladatok (összesen 15 pont)

1. Reszponzív kinézet (2p)

Az alkalmazás mobilon és asztali gépen is legyen jól használható. Használj rugalmas elrendezést (pl. Flexbox, Grid), hogy a lista igazodjon a kijelző méretéhez.

2. Kód és projektstruktúra (2p)

Legyen a projekted tagolt és átlátható. A komponenseket, a logikát (pl. API hívások, állapotkezelés) érdemes külön fájlokba szervezni. Ha tudod, használj contextet a globális állapothoz. A kinézet és a működés maradjon külön, ne keveredjenek össze.

3. Modulok telepítése (1p)

Használd az npm-et a szükséges csomagok (pl. axios, MUI, react-icons) telepítésére. Ezeket integráld megfelelően a projektbe.

4. Üzenetek lekérdezése (2p)

A **„/api/messages”** végponton kérd le az üzeneteket a backendtől egy GET kérés segítségével. Az adatokat olyan módon kérdezd le és tárold, hogy azok később megjeleníthetőek legyenek.

5. Üzenetek megjelenítése (3p)

A lekérdezett üzeneteket jelenítsd meg lista elrendezésben. Minden üzenet alatt jelenjen meg a likeok száma is.

6. Új üzenet beküldése (2p)

A felhasználó szöveges mezőbe írt üzenetet kell tudjon beküldeni. Az üzenet formátumát nem kell ellenőrizni, de az üres beküldést kezelni kell (ne kerüljön beküldésre). A sikeres feltöltés után jelenjen meg az új üzenet is.

7. Like küldése (2p)

Minden üzenet alatt legyen lehetőség az adott üzenetet likeolni a **POST „/api/messages/:id/like”** végpont használatával.

8. Animáció használata (1p)

Legalább egy látványos animáció vagy átmenet jelenjen meg az alkalmazásban (pl. gomb hover effektek, lista animációja, üzenet megjelenés). Az animáció történhet CSS-sel vagy animációs könyvtárral is.

Beadás

A projektet tömörített formában (ZIP) kell leadni. A `node_modules` mappákat mind a frontend, mind a backend mappából törölni kell a tömörítés előtt.

A leadandó fájlt kicsomagolás után újra próbáld ki, és indítsd el a projektet. Ezzel biztosíthatod, hogy valóban működő, értékelhető állapotban adod le a feladatod.
Nem futtatható vagy hibával induló projekt nem értékelhető!