Beléptető rendszer

tervezet

2022.04.26

**Készítette:**

# Tartalomjegyzék

[**Tartalomjegyzék 2**](#_heading=h.2jxsxqh)

[**Feladat megadása 2**](#_heading=h.z337ya)

[**Szereplők és igényeik 2**](#_heading=h.3j2qqm3)

[**Látogató 2**](#_heading=h.1y810tw)

[**Adminisztrátor 2**](#_heading=h.4i7ojhp)

[**Use-Case funkció lista 3**](#_heading=h.2xcytpi)

[**Látogató 3**](#_heading=h.1ci93xb)

[**Adminisztrátor 3**](#_heading=h.3whwml4)

[**Magas szintű rendszerterv 3**](#_heading=h.2bn6wsx)

[**Screenek 3**](#_heading=h.qsh70q)

[**Modellek 7**](#_heading=h.3as4poj)

[**Alkalmazások kiválasztása 7**](#_heading=h.1pxezwc)

[**Routing 7**](#_heading=h.49x2ik5)

[**Middleware-ek (algoritmusok) 9**](#_heading=h.2p2csry)

[**Implementáció 9**](#_heading=h.147n2zr)

[**Tesztelés 9**](#_heading=h.3o7alnk)

[**Bevezetés, éles üzem 9**](#_heading=h.23ckvvd)

# Feladat megadása

Készítsünk egy beléptetőrendszert az *intaccess Bt.* interakcióinak ellenőrzéséhez, amelyben a felhasználók belépnek (vagy nem) a munkába a nap elején és kilépnek (vagy nem) a nap végén. Az egyszeri látogatók legyenek képesek az egyes felhasználók belépési adatait megtekinteni. Az adminisztrátor az előbb felsorolt lehetőségeken felül, legyen képes egy meglévő felhasználót inaktívvá tenni, továbbá joga van az adatbázisban tárolt belépési adatok módosítására, törlésére.

# Szereplők és igényeik

## Látogató

* Use-Case funkció listában már megadva

## Adminisztrátor

* Use-Case funkció listában már megadva

# Use-Case funkció lista

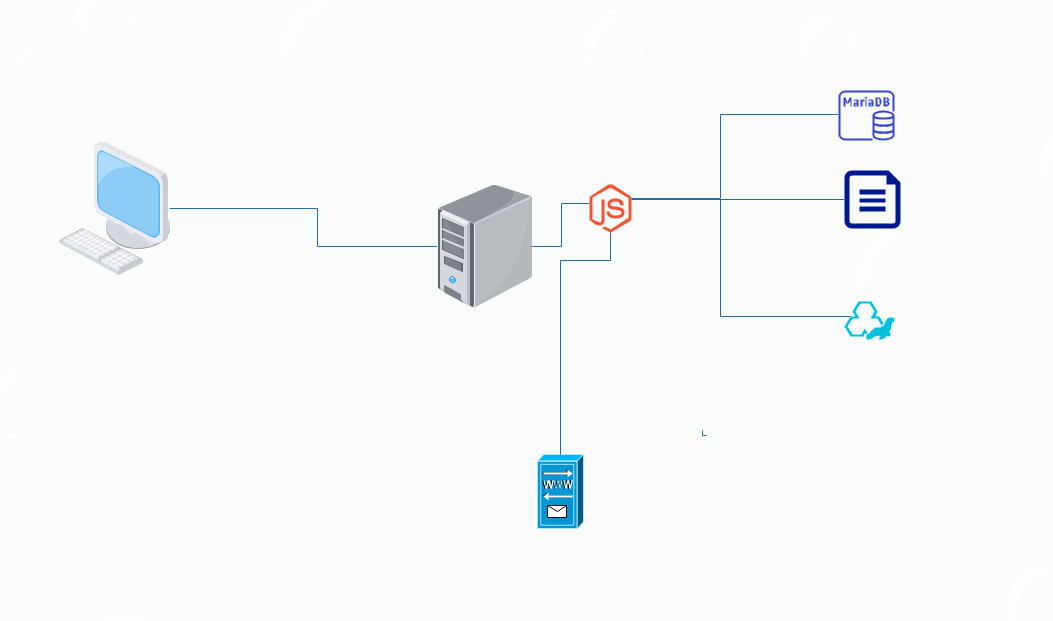
## Látogató

* a látogató a kezdő-, belépő-, a regisztrációs-, valamint a statikus oldalakat éri el
* a látogató regisztrálni tud az oldalra
* a látogatónak lehetősége van belépni az oldalra
* a látogató az adatbázisban tárolt adatokat megtekintheti

## Adminisztrátor

* az adminisztrátornak joga van mindenhez, amihez a látogatónak továbbá*:*
* az adminisztrátornak joga van a belépési adatokat módosítani,
* az adminisztrátornak joga van a belépési adatokat törölni,

# Magas szintű rendszerterv



**?**

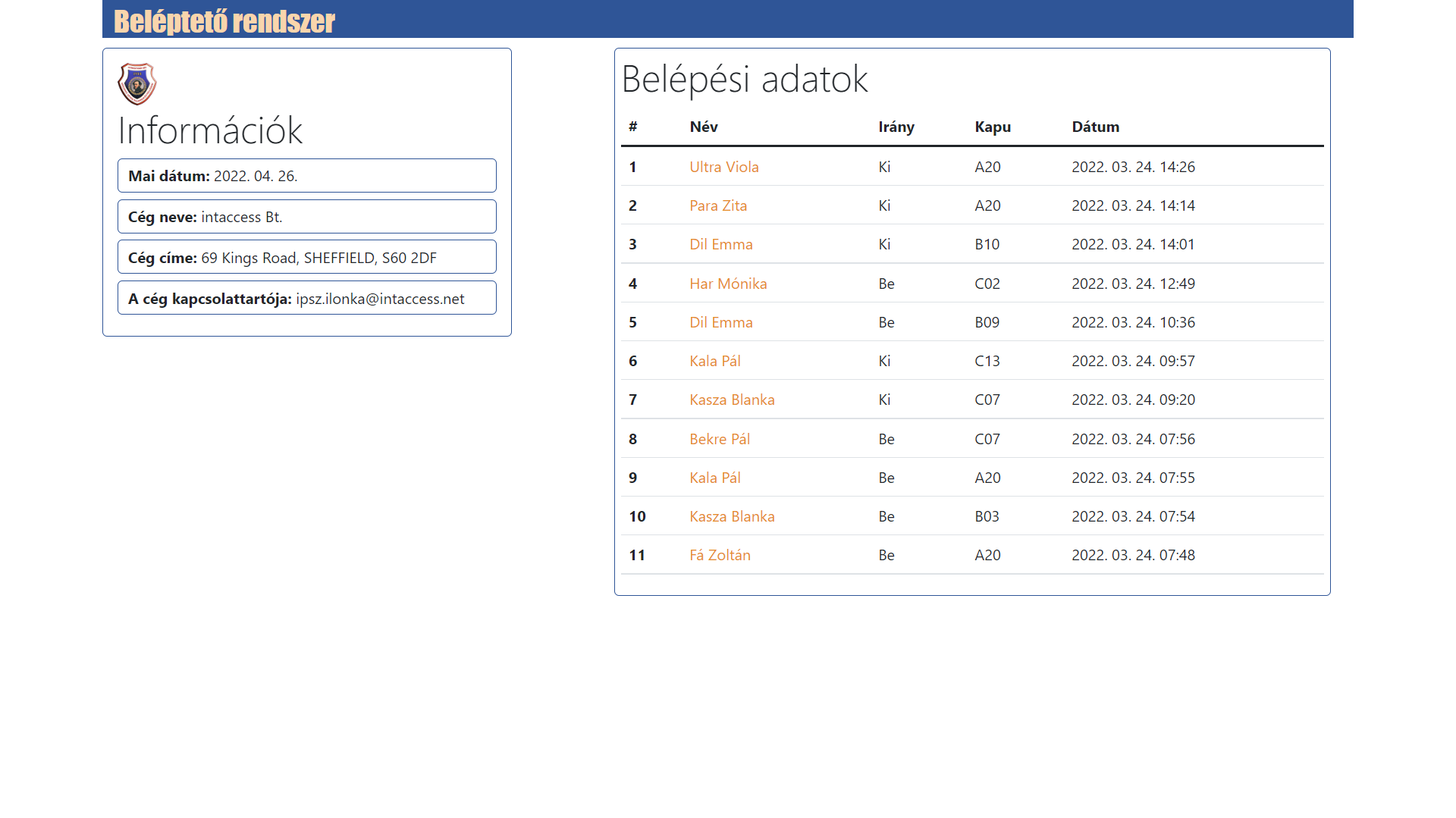
**?**

A jelenlegi megvalósításnál a Loki ORM-et kívánjuk használni, de lehetőséget akarunk biztosítani a később fejlesztések során az adatok tárolásához a MySQL és a MongoDB használatára. Ezért ezeket is jelöltük a fenti ábrán.

# Screenek

A frontend megvalósításhoz rendelkezésre áll már néhány statikus megoldás, ezeket használja fel a szerver oldali rendering megoldáshoz! (EJS)

A mai dátumnál az aktuális dátumnak kell megjelennie, helyes magyar formátumban!

*Nyitó képernyő (static/index.html)*

# Modellek

**user**{

* **uuid** (string - a felhasználó egyedi azonosítója)
* **name** (string - a felhasználó teljes neve)
* **email** (string - a felhasználó e-mail címe, az e-mail és a jelszó együttes használatával tudja hitelesíteni magát a felületen)
* **admin** (boolean – igaz esetén admin jogú)
* **actions** [ ](lista, az adott felhasználó be- és kilépési adatait tartalmazza)

}

**action{**

**Ezt meg kell csinálni**

**}**

# Alkalmazások kiválasztása

* Nodejs
  + express
  + LokiJS
  + EJS
  + bcryptjs
  + uuid
  + moment, ha a dátumok formázásához kívánja használni, nem kötelező
  + egyéb beépített modulok, amelyek a megvalósítás során nem elsődleges fontosságúak

# Routing

A statikus tartalmak a /public mappából kerülnek kiszolgálásra.

Valósítsd meg a következő route-okat:

GET /

**funkció**: nyitóoldal, az adatbázisban tárolt adatok táblázatos megjelenítése

**view**: index

**válasz**:

* **200 OK**: ha minden rendben, akkor rendereljük ki a megadott oldalt

GET /user/:uuid

**funkció**: a uuid azonosítójú felhasználó be- és kilépési adatai

**view**: user

**válasz**:

* **404 Not found**: ha nem létezik a uuid
* **200 OK**: ha létezik az adott uuid, akkor rendereljük ki a megadott oldalt

GET /actions

**funkció**: az adatbázisban tárolt interakciók listázása

**válasz**:

* **200 OK**: ha minden rendben, akkor adjuk vissza az adatbázisban tárolt interakciókat

POST /action

**funkció**: új interakció felvétele

**kérés**: egy interakció JSON formában felküldve:

{

"user\_id": "123e4567-e89b-12d3-a456-426614174000",

"type": 0,

"gate": "A20",

"date": "2022-04-26 16:22"

}

**válasz**:

* **400 Bad request:** hibás adat küldése esetén
* **201 Created:** siker esetén, és adjuk vissza az új interakció adatait

PATCH /action/:uuid

**funkció**: egy meglévő interakció adatainak módosítása

**kérés**: az interakció egy részének adatai JSON formában felküldve:

{

"gate": "C13",

}

**válasz**:

* **400 Bad request:** hibás adat küldése esetén
* **404 Not found:** ha nem létezik a uuid
* **200 OK:** siker esetén, és adjuk vissza a módosított interakció adatait

DELETE /action/:uuid

**funkció**: egy meglévő interakció törlése

**válasz**:

* **404 Not found:** ha nem létezik a uuid
* **200 OK:** siker esetén, és adjuk vissza a törölt interakció adatait

Az erőforrás útvonalakat szükség esetén átírhatja, ez csak egy terv. A módokat is átalakíthatja igény szerint, ha szükségét látja. Az egyes útvonalakat összevonhatja, ha teljesülnek az összevonás feltételei és szeretné összevonni őket. Ha külön maradnak, nem jár érte kevesebb pont.

# Middleware-ek (algoritmusok)

**Ezt kell megcsinálni. Elég a kód**

# Implementáció

**Ezt kell megcsinálni. Elég a kód**

# Tesztelés

A routok tesztelését Postman-ban végezzük el! A Postman-ban Workspace-be hozzuk létre az egyes útvonalak tesztjeit és az adott Workspace-t JSON formátumban exportáljuk ki és a kész projekt mellé helyezzük el!

# Bevezetés, éles üzem

Hamarosan. Nincs vele teendő jelenleg.