**Záródolgozat feladatkiírás**

Tanuló(k) neve: Gere Kovács Dávid, Horváth Gergő, Szalai Árpád Bátor Képzés: nappali munkarend

Szak: 5 0613 12 03 Szoftverfejlesztő és -tesztelő technikus

**A záródolgozat címe: TrashBusters**

Konzulens: Nits László

Beadási határidő: 2024. 04.15.

Győr, 2023. 10. 01

#### Módos Gábor

igazgató

**Konzultációs lap**

|  |  |  |  |
| --- | --- | --- | --- |
|  | A konzultáció | | Konzulens aláírása |
| ideje | témája |
| 1. | 2024.02.15 | Témaválasztás és specifikáció |  |
| 2. | 2024.03.14 | Záródolgozat készültségi fokának  értékelése |  |
| 3. | 2024.04.15. | Dokumentáció véglegesítése |  |

**Tulajdonosi nyilatkozat**

Ez a dolgozat a saját munkánk eredménye. Dolgozatunk azon részeit, melyeket más szerzők munkájából vettünk át, egyértelműen megjelöltük.

Ha kiderülne, hogy ez a nyilatkozat valótlan, tudomásul vesszük, hogy a szakmai vizsgabizottság a szakmai vizsgáról kizár minket és szakmai vizsgát csak új záródolgozat készítése után tehetünk.

Győr, 2023. április 15

Gere Kovács Dávid

Horváth Gergő

Szalai Árpád Bátor

**TrashBusters dokumentáció**

Készítették: Gere Kovács Dávid

Horváth Gergő Szalai Árpád Bátor

**Tartalomjegyzék**

1. [Bevezetés 6](#_bookmark0)
   1. [Témaválasztás 6](#_bookmark1)
   2. [Elérhetőség 6](#_bookmark2)
2. [Fejlesztés 7](#_bookmark3)
   1. [Fejlesztési eszközök 7](#_bookmark4)
      1. [Visual Studio Code 7](#_bookmark5)
      2. [GitHub 7](#_bookmark6)
      3. [draw.io 7](#_bookmark7)
      4. [dbdiagram.io 8](#_bookmark8)
   2. [Szerepkörök 8](#_bookmark9)
   3. [Munkafolyamat 8](#_bookmark10)
3. [Tervezés 9](#_bookmark11)
   1. [Adatbázis 9](#_bookmark12)
4. [Backend 11](#_bookmark13)
   1. [Használt technológiák 11](#_bookmark14)
      1. [PHP Laravel 11](#_bookmark15)
      2. [MySQL 11](#_bookmark16)
      3. [SendGrid 11](#_bookmark17)
   2. [Felhő szolgáltatások 12](#_bookmark18)
      1. [AWS 12](#_bookmark19)
      2. [RDS 13](#_bookmark20)

[4.2.3 S3 13](#_bookmark21)

[4.2.4 EC2 14](#_bookmark22)

* 1. [Adatbázis 14](#_bookmark23)
     1. [Users 14](#_bookmark24)
     2. [Events 15](#_bookmark25)
     3. [Participants 15](#_bookmark26)
     4. [Dumps 16](#_bookmark27)
     5. [Assigneds 16](#_bookmark28)
  2. [API hívások 16](#_bookmark29)
     1. [AuthController 17](#_bookmark30)
     2. [DumpController 18](#_bookmark31)
     3. [EventController 19](#_bookmark32)
     4. [ParticipantController 22](#_bookmark33)
     5. [UserController 24](#_bookmark34)

1. [Frontend 27](#_bookmark35)
   1. [Használt technológiák 27](#_bookmark36)
      1. [Typescript 27](#_bookmark37)
      2. [Vue.js 27](#_bookmark38)
      3. [Ionic framework 28](#_bookmark39)
      4. [Axios 29](#_bookmark40)
2. [Az oldal működése 31](#_bookmark41)
   1. [Hitelesítés 31](#_bookmark42)
      1. [Regisztráció 31](#_bookmark43)
      2. [Bejelentkezés 32](#_bookmark44)
   2. [Események 33](#_bookmark45)
      1. [Esemény létrehozása 33](#_bookmark46)
      2. [Csatlakozás eseményhez 34](#_bookmark47)
      3. [Események szerkesztése és törlése 35](#_bookmark48)
   3. [Profil 35](#_bookmark49)
      1. [Jelszó-helyreállítás 35](#_bookmark50)
      2. [Jelszó cseréje 36](#_bookmark51)
      3. [Profilkép változtatás 36](#_bookmark52)
      4. [Fiók törlése 37](#_bookmark53)

# Bevezetés

## Témaválasztás

A TrashBusters egy online weboldal, ahol a felhasználók közösségi szemétszedéseket tudnak szervezni és ezeken részt venni.

Manapság sajnos nagyok sok problémát okoz a szemetelés, rengetegszer láthatunk kiborult kukákat, teleszemetelt parkokat és eldobott hulladékokat szerte a környezetünkben.

Erre próbál megoldást nyújtani az oldalunk, ahol, miután valaki hasonló gondokat észlel, létre tud hozni egy eseményt, ahol megadja a fontos

információkat és ezután a többi felhasználónak lehetősége lesz csatlakozni ehhez.

## Elérhetőség

A programhoz tartozó forráskódokat az alábbi linkeken keresztül lehet elérni:

* + - Backend: <https://github.com/Gerihh/TrashBusters-Backend>
    - Frontend: [https://github.com/Gere-Kovacs-David/TrashBusters\_Ionic-](https://github.com/Gere-Kovacs-David/TrashBusters_Ionic-Vue_Frontend) [Vue\_Frontend](https://github.com/Gere-Kovacs-David/TrashBusters_Ionic-Vue_Frontend)

# Fejlesztés

## Fejlesztési eszközök

* + 1. Visual Studio Code1

A Visual Studio Code egy ingyenes és nyílt forráskódú kódszerkesztő szoftver. Rengeteg hasznos beépített funkcióval rendelkezik pl. automatikus hibakereső és Git támogatás. Ezek mellett számos egyéb kiegészítőt is tölthetünk le a szoftverhez, hogy esek segítségével is könnyítsük munkánkat. Az egyik legismertebb és legelterjedtebb kódszerkesztő napjainkban.A Backend és a Frontend fejlesztésére is ezt a programot használtuk.

* + 1. GitHub2

A GitHub a legnépszerűbb verziókövető és kódmegosztó platform. Könnyen átlátható kezelői felületének köszönhetően egyszerűen tudunk navigálni a nekünk szükséges részlegek között. Egyszerűen lehetett munkaköröket és feladatokat kiosztani a fejlesztés során és nagyban megkönnyítette a közös munkát a kódolás terén.

* + 1. draw.io3

A draw.io egy webes diagramkészítő szoftver, elsősorban az ER-modell elkészítése miatt volt szükség a használatára. Az elkészült ábrákat el lehet menteni PDF és képi formátumokban is, ezért a későbbiekben egyszerű volt ezeket használni.

1 <https://code.visualstudio.com/>

2 <https://github.com/>

3 <https://app.diagrams.net/>

* + 1. dbdiagram.io4

Hasonlóan a draw.io-hoz a dbdiagram.io is az adatbázis modellezésében nyújtott segítséget, itt viszont az adatbázis-modell került elkészítésre.

## Szerepkörök

Gere Kovács Dávid: frontend Horváth Gergő: backend, frontend Szalai Árpád Bátor: backend

## Munkafolyamat

A fejlesztés során folyamatosan célokat és feladatokat tűztünk ki magunk elé, majd ezeket igyekeztünk megoldani és ezután újakat létrehozni. A kommunikáció nagyrészt személyesen, az iskolai légtérben történt, ezen kívül pedig Messenger-en és Discord-on egyeztettünk a különböző feladatokkal kapcsolatban, ha a szükség úgy kívánta.

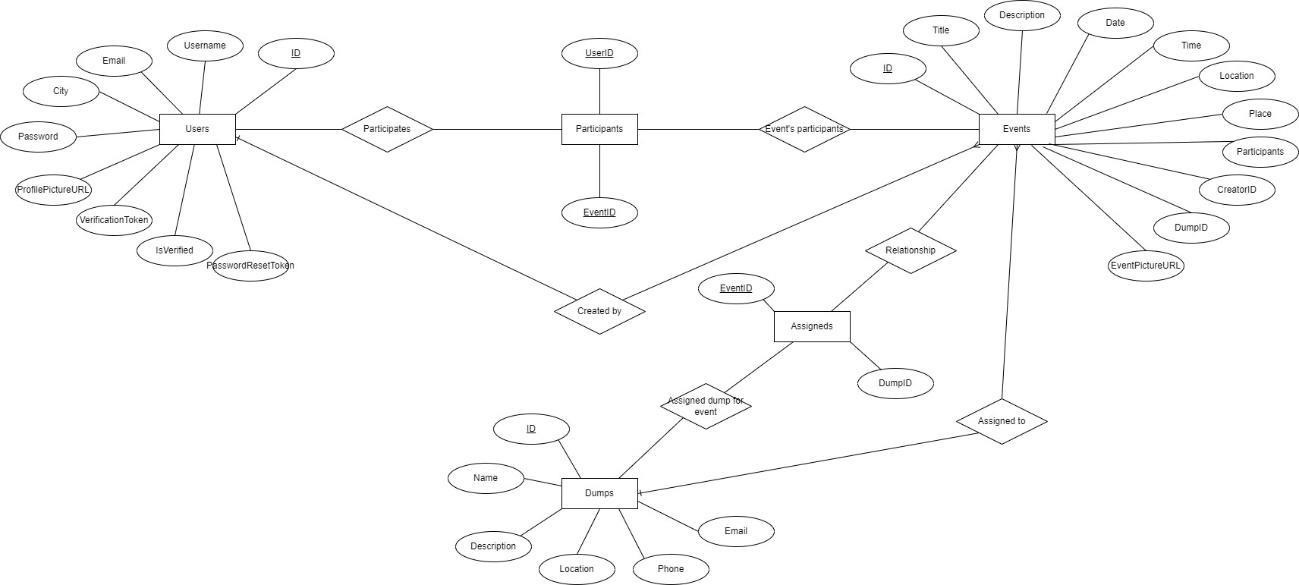
4 <https://dbdiagram.io/>

# Tervezés

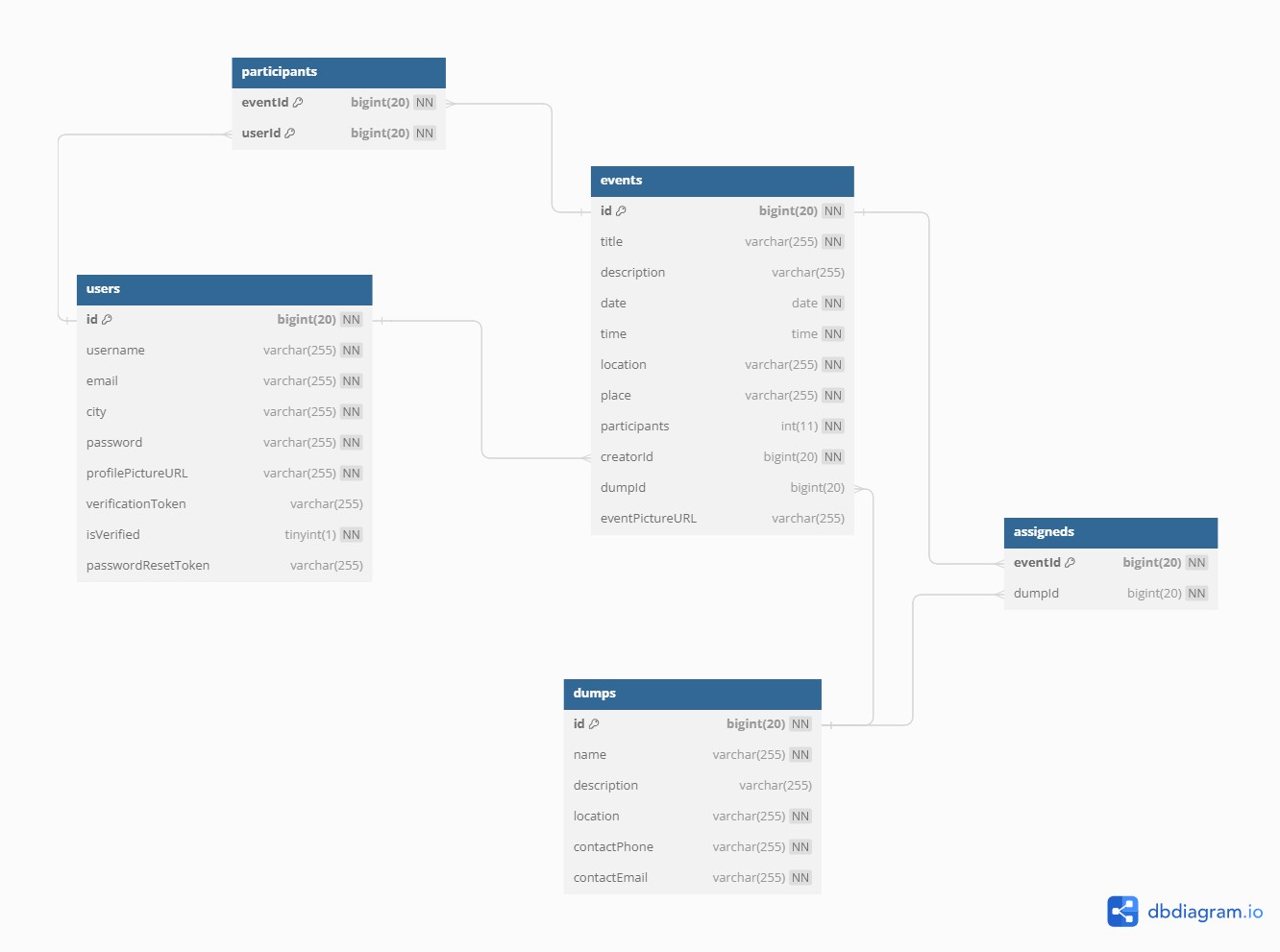
## Adatbázis

Az adatbázis modellje a fejlesztés korai szakaszától kezdve többször is változott, annak függvényében, hogy milyen plusz funkciókat szerettünk volna még létrehozni a projektben. A végleges modell pedig így néz ki:

ER-Modell:



Adatbázis-modell:



# Backend

## Használt technológiák

* + 1. PHP Laravel5

A Laravel egy ingyenes és nyílt forráskódú webes PHP keretrendszer. A modell-nézet-kontroller (MVC) architektúrát követi, emellett támogatja az Eloquent ORM-t is, ami megkönnyíti a relációs adatbázisokkal való kommunikációt. A PHP egy script-nyelv ami elsősorban webfejlesztésre irányul, a PHP kódot a webszerver dolgozza fel, majd fordítja le és ezt küldi vissza HTTP válaszként.

* + 1. MySQL6

A MySQL egy SQL (Structured Query Language, strukturált leíró-nyelv) alapú relációs adatbázis-kezelő szolgáltatás. Az egyik legelterjedtebb adatbázis- kezelő mivel egyszerű megoldást nyújt dinamikus webhelyek szolgáltatására, továbbá számos lehetőséget kínál az adatbázis optimalizálására pl. skálázhatóság és kompatibilitás.

* + 1. SendGrid7

Programunk néhány funkciójához (regisztráció, jelszó-helyreállítás) szükség volt egy felhő-alapú e-mail küldő szolgáltatásra, ehhez választottuk a SendGrid-

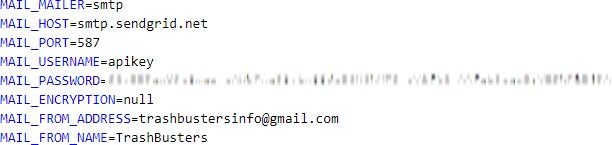
5 <https://laravel.com/>

6 <https://www.mysql.com/>

7 <https://sendgrid.com/>

et. Miután regisztrálunk az oldalon, készítenünk kell egy API-kulcsot, majd később ezt használva tudjuk végezni az e-mail küldést a kódban.

Példa Laravelben való használatra:



## Felhő szolgáltatások

### AWS

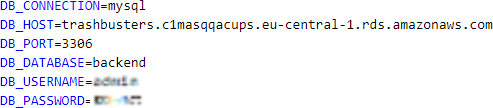
Mivel cél volt, hogy a weboldal ne csak lokális környezetben fusson, és publikusan is elérhető legyen, meg kellett oldani a különböző szerverek hosztolását. A döntés az Amazon Web Services-re esett, mivel így nem kellett több oldalon hosztolni az adatbázis, backend- és frontend-szervereket, hanem mindent el tudtunk érni egy helyen. Hátrányai közé lehetne sorolni azt, hogy ezek a szolgáltatások fizetősek, viszont regisztráció után a felhasználó 1 évig jogosult az Amazon Free Tier-re, így csak kezelési költséget számolnak fel erre az időtartamra , ezért erre az időszakra teljesen megfelelt. Az egyéb ingyenes szolgáltatásokhoz képest pedig többet tud nyújtani pl. biztonság, skálázhatóság és sebesség terén is.

### RDS

Az Amazon RDS (Relational Database Service) lehetővé teszi relációs adatbázisok futtatását a felhőben. Az Amazon Aurora adatbázis motort

használja, ami kifejezetten felhőben való használatra készült és kompatibilis a MySQL-lel és a PostgreSQL-lel is. Miután létrehoztuk az adatbázis-szervert a backend-ünk konfigurációs fájljában kell megadnunk a szükséges adatokat a használatához.

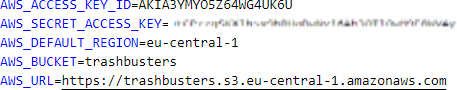
Példa Laravelben való használatra:



### S3

Az Amazon S3 (Simple Storage Service) egy felhő tárhely, ami objektumtárolást tesz lehetővé. Bármilyen típusú objektumot képes tárolni, így lehetővé teszi alkalmazások, biztonsági mentések és egyéb adatforrások tárolását. Mi a felhasználó által feltöltött képek tárolására használtuk. Hasonlóan a többi AWS szolgáltatáshoz, itt is a konfigurációs fájlban kell megadnunk az adatokat a kapcsolódáshoz.

Példa Laravelben való használatra:



### EC2

Az Amazon EC2 (Elastic Compute Cloud) lehetővé teszi a felhasználóknak, hogy virtuális gépeket béreljenek és ezeken futtassák az alkalmazásaikat, továbbá a szolgáltatás támogatja a projektek skálázhatóságát és a földrajzi elhelyezkedés szabályozását.

## Adatbázis

### Users

Felhasználók adatai:

* id, a felhasználó azonosítója (int, unique)
* username, felhasználónév (string)
* email, a felhasználó e-mail címe (string, unique)
* city, a felhasználó városa (string)
* password, hashelt jelszó (string)
* profilePictureURL, a felhasználó profilképének url-je (string)
* verificationToken, regisztráció során kapott token (string)
* isVerified, megmutatja, hogy hitelesített-e a felhasználó (boolean)
* passwordresetToken, elfelejtett jelszó esetén kapott token (string)

### Events

Események adatai:

* + - * id, az esemény azonosítója (int, unique)
      * title, az esemény neve (string)
      * description, az esemény leírása (string)
      * date, az esemény dátuma (date)
      * time, az esemény időpontja (time)
      * location, az esemény városa (string)
      * place, az esemény címe (string)
      * participants, résztvevők száma (int)
      * creatorId, az esemény létrehozójának id-je (int)
      * dumpId, az eseményhez kapcsolódó lerakó azonosítója (int)
      * eventPictureURL, az eseményhez kapcsolódó kép url-je (string)

### Participants

Résztvevők:

* eventId, az esemény azonosítója (int)
* userId, a felhasználó azonosítója (int)

### Dumps

Lerakók adatai:

* id, a lerakó azonosítója (int)
* name, a lerakó neve (string)
* description, a lerakó leírása (string)
* location, a lerakó címe (string)
* contactphone, a lerakó telefonszáma (string)
* contactEmail, a lerakó e-mail címe (string)

### Assigneds

Adott eseményhez társított lerakó:

* eventId, az esemény azonosítója (int)
* dumpId, a lerakó azonosítója (int)

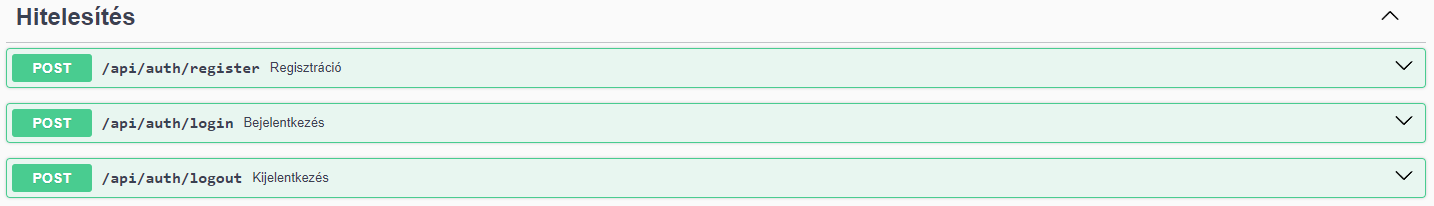
## API hívások

A projektünkhöz a REST API (Representational State Transfer Application Programming Interface) technológiát használtuk. Ez lehetővé teszi a kliens számára kommunikációt a szerverrel. Az API HTTP protokollon keresztül működik és JSON formátumú válaszokat küldenek vissza.

Használt HTTP metódusok:

* + - GET: Elem(ek) lekérése
    - POST: Új elem létrehozása
    - PUT/PATCH: Adott elem módosítása
    - DELETE: Elem törlése

### AuthController



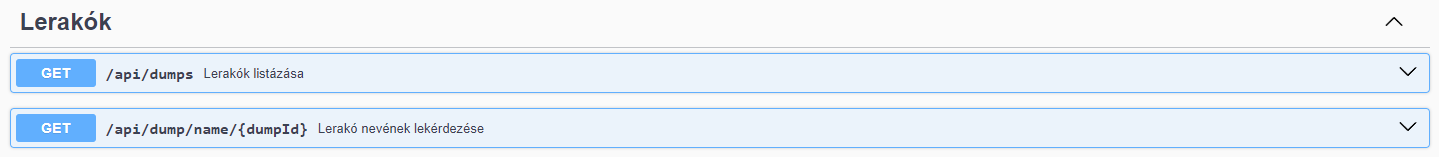
register:

* Metódus: POST
* Útvonal: /api/auth/register
* Feladat: regisztráció
* Paraméter: -
* Válasz: User típusú osztálypéldány, accessToken mező

login:

* Metódus: POST
* Útvonal: /api/auth/login
* Feladat: bejelentkezés
* Paraméter: -
* Válasz: User típusú osztálypéldány, accessToken mező logout:
* Metódus: POST
* Útvonal: /api/auth/logout
* Feladat: kijelentkezés
* Paraméter: -
* Válasz: string

### DumpController



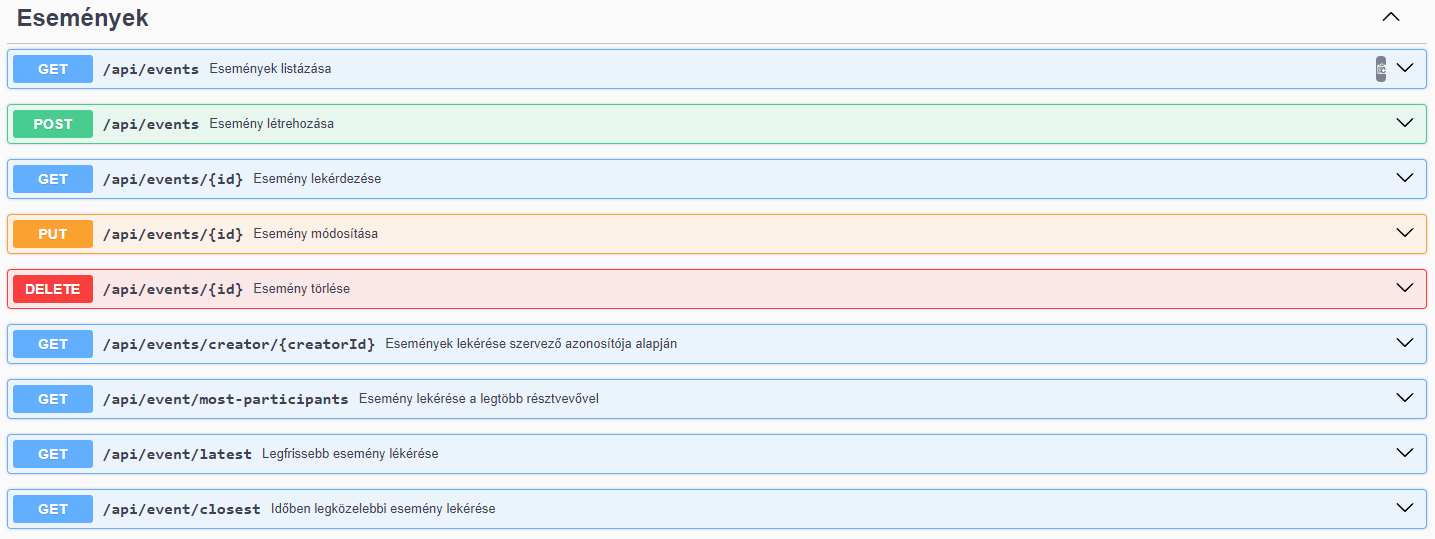
index:

* Metódus: GET
* Útvonal: /api/dumps
* Feladat: az összes lerakó listázása
* Paraméter: -
* Válasz: Dump típusú elemekből álló tömb

getDumpNameById:

* Metódus: GET
* Útvonal: /api/dump/name/{dumpId}
* Feladat: lerakó nevének kiírása id alapján
* Paraméter: dumpId
* Válasz: string

### EventController



index:

* Metódus: GET
* Útvonal: /api/events
* Feladat: az összes esemény listázása
* Paraméter: -
* Válasz: Event típusú elemekből álló tömb

store:

* Metódus: POST
* Útvonal: /api/events
* Feladat: új esemény létrehozása
* Paraméter: -
* Válasz: Event típusú osztálypéldány

show:

* Metódus: GET
* Útvonal: /api/events/{id}
* Feladat: adott esemény listázása
* Paraméter: id
* Válasz: Event típusú osztálypéldány

update:

* Metódus: PUT
* Útvonal: /api/events/{id}
* Feladat: adott esemény módosítása
* Paraméter: id
* Válasz: Event típusú osztálypéldány

destroy:

* Metódus: DELETE
* Útvonal: /api/events/{id}
* Feladat: adott esemény törlése
* Paraméter: id
* Válasz: string

getEventByCreatorId:

* Metódus: GET
* Útvonal: /api/events/creator/{creatorId}
* Feladat: adott felhasználó által létrehozott események listázása
* Paraméter: creatorId
* Válasz: Event típusú osztálypéldány

getEventWithMostParticipants:

* Metódus: GET
* Útvonal: /api/event/most-participant
* Feladat: a legtöbb résztvevővel rendelkező esemény listázása
* Paraméter: -
* Válasz: Event típusú osztálypéldány

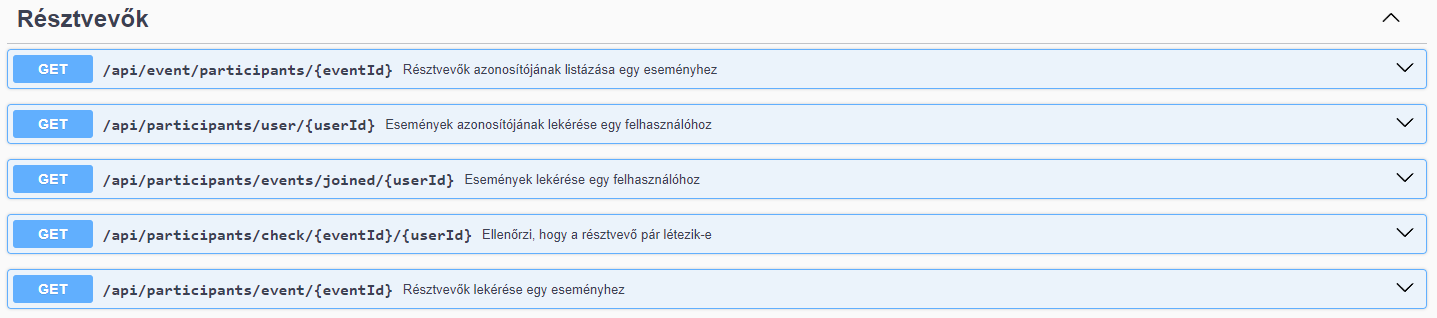
getLatestEvent:

* Metódus: GET
* Útvonal: /api/event/latest
* Feladat: a legfrissebben létrehozott esemény listázása
* Paraméter: -
* Válasz: Event típusú osztálypéldány

getClosestEvent:

* Metódus: GET
* Útvonal: /api/event/closest
* Feladat: a legkorábban megrendezésre kerülő esemény listázása
* Paraméter: -
* Válasz: Event típusú osztálypéldány

### ParticipantController



getByEventId:

* Metódus: GET
* Útvonal: /api/event/participants/{eventId}
* Feladat: résztvevők id-ának listázása az adott eseményhez
* Paraméter: eventId
* Válasz: eventId és userId mezőből álló objektumtömb

getByUserId:

* Metódus: GET
* Útvonal: /api/participants/user/{userId}
* Feladat: események id-ának listázása, amin részt vesz az adott felhasználó
* Paraméter: userId
* Válasz: eventId és userId mezőből álló objektumtömb

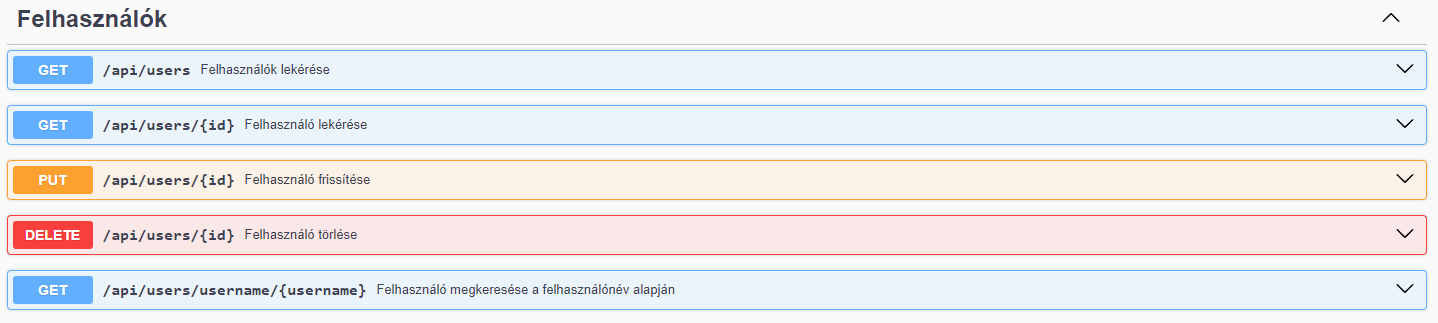
eventsJoinedByUser.

* Metódus: GET
* Útvonal: /api/participants/events/joined/{userId}
* Feladat: események listázása, ahová az adott felhasználó csatlakozott
* Paraméter: userId
* Válasz: Event típusú elemekből álló tömb

pairExists:

* Metódus: GET
* Útvonal: /api/participants/check/{eventId}/{userId}
* Feladat: megnézi, hogy létezik-e adott esemény-felhasznál pár
* Paraméter: eventId, userId
* Válasz: boolean getParticipantsByEventId:
* Metódus: GET
* Útvonal: /api/participants/event/{eventId}
* Feladat: résztvevők listázása az adott eseményhez
* Paraméter: eventId
* Válasz: User típusú elemekből álló users tömb és count mező

### UserController



index:

* Metódus: GET
* Útvonal: /api/users
* Feladat: az összes felhasználó listázása
* Paraméter: -
* Válasz: User típusú elemekből álló tömb

show:

* Metódus: GET
* Útvonal: /api/users/{id}
* Feladat: adott felhasználó listázása
* Paraméter: id
* Válasz: User típusú osztálypéldány

update:

* Metódus: PUT
* Útvonal: /api/users/{id}
* Feladat: adott felhasználó módosítása
* Paraméter: id
* Válasz: User típusú osztálypéldány

destroy:

* Metódus: DELETE
* Útvonal: /api/users/{id}
* Feladat: adott felhasználó törlése
* Paraméter: id
* Válasz: string

getUserByUsername:

* Metódus: GET
* Útvonal: /api/users/username/{username}
* Feladat: felhasználó keresése felhasználónév alapján
* Paraméter: username
* Válasz: User típusú osztálypéldány

# Frontend

## Használt technológiák

### Typescript

A TypeScript egy szabványosított nyelv a JavaScript-hez, amely lehetővé teszi a statikus típusellenőrzést, a fejlettebb IDE támogatást, valamint a kódolási hibák csökkentését. A TypeScript egy olyan nyelv, amely lehetővé teszi a JavaScript kód fejlesztését, amely könnyen skálázható és karbantartható, és képes integrálódni más modern fejlesztési eszközökkel, például az Vue.js keretrendszerrel. A TypeScript nagyobb biztonságot és rugalmasságot biztosít a kódolás során, és a JavaScript kód fejlesztésének egyre népszerűbb alternatívája.

### Vue.js

A Vue.js egy progresszív JavaScript keretrendszer, amely lehetővé teszi a fejlesztők számára, hogy dinamikus és interaktív webalkalmazásokat hozzanak létre. A Vue.js könnyen tanulható és használható, és egyre népszerűbb a webfejlesztők körében.

A Vue.js használata számos előnnyel jár, például:

* + - * Adatokhoz kötött elemek: A Vue.js lehetővé teszi az alkalmazás állapotának könnyű kezelését, így könnyedén frissíthetők és reagálhatnak az adatváltozásokra.
      * Rugalmas és moduláris struktúra: A Vue.js moduláris felépítése és a szervezett kód struktúrája lehetővé teszi a könnyű karbantarthatóságot és kiterjeszthetőséget.
      * Komponensek újra felhasználhatósága: A Vue.js lehetővé teszi a komponensek újraf elhasználását, így egyszerűbbé teszi az alkalmazások fejlesztését és karbantartását.
      * Egyszerű fejlesztői környezet: A Vue.js szabványos fejlesztői környezetet biztosít, és rendelkezik egy könnyen használható CLI (Command Line Interface) segédprogrammal, amely lehetővé teszi a gyors és hatékony fejlesztést.
      * Integráció más technológiákkal: A Vue.js lehetővé teszi az egyszerű integrációt más JavaScript könyvtárakkal és keretrendszerekkel, így könnyen kombinálható más technológiákkal is.

### Ionic framework

Az Ionic egy nyílt forráskódú keretrendszer, amely lehetővé teszi a fejlesztők számára, hogy keresztplatformos mobilalkalmazásokat hozzanak létre HTML, CSS és JavaScript felhasználásával. Az Ionic keretrendszer elsősorban arra összpontosít, hogy könnyen és hatékonyan hozzon létre olyan mobilalkalmazásokat, amelyek hasonlóan néznek ki és működnek, mint a natív mobilalkalmazások.

Az Ionic keretrendszer használata számos előnnyel jár, például:

* + - * Keresztplatformos kompatibilitás: Az Ionic lehetővé teszi a fejlesztők számára, hogy egyetlen kódbázisból keresztplatformos mobilalkalmazásokat készítsenek, amelyek futtathatók iOS, Android és más platformokon is.
      * Könnyen testre szabható felhasználói felület: Az Ionic számos előre elkészített UI komponenst kínál, amelyek segítségével gyorsan és egyszerűen létrehozhatók modern és vonzó felhasználói felületek.
      * Teljesítmény és sebesség: Az Ionic keretrendszer optimalizált és gyors futást biztosít, ami lehetővé teszi az alkalmazások sima és megbízható működését.
      * Integráció más keretrendszerekkel: Az Ionic lehetővé teszi az egyszerű integrációt más JavaScript keretrendszerekkel, például Angular, React vagy Vue.js, így a fejlesztők könnyen kihasználhatják a különböző technológiák előnyeit.
      * Egyszerű fejlesztői környezet: Az Ionic rendelkezik egy könnyen használható CLI (Command Line Interface) segédprogrammal, amely lehetővé teszi a gyors és hatékony fejlesztést, valamint a tesztelést és az alkalmazások kiadását.

### Axios

Axios egy könnyen használható, modern és nyílt forráskódú HTTP kliens JavaScript könyvtár, amely lehetővé teszi a fejlesztők számára, hogy könnyedén kommunikáljanak aszinkron kérésekkel a szerverrel. Az Axios nagyon népszerű a frontend fejlesztők körében, mivel számos előnnyel jár:

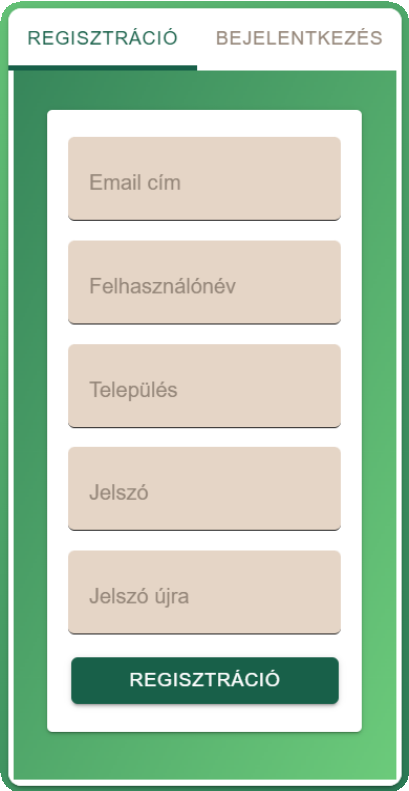
* + - * Egyszerű használat: Az Axios egy egyszerű és intuitív API-t kínál, amely lehetővé teszi a fejlesztők számára, hogy könnyen hajtsanak végre HTTP kéréseket, például GET, POST, PUT és DELETE kéréseket.
      * Promises támogatás: Az Axios aszinkron működik és teljes mértékben támogatja a Promises-t, így könnyen kezelhetők az aszinkron műveletek és a kérések során kapott adatok.
      * Interceptorok: Az Axios lehetővé teszi a kérések és válaszok közbeiktatását, így könnyen kezelhetők az autentikáció, az autorizáció, a hibakezelés és más globális feladatok.
      * Kliensoldali és szerveroldali használat: Az Axios könnyen használható mind a kliensoldalon (például böngészőben vagy mobilalkalmazásokban), mind a szerveroldalon (például Node.js környezetben), ami nagy rugalmasságot biztosít a fejlesztés során.
      * Integráció más technológiákkal: Az Axios egyszerűen integrálható más JavaScript keretrendszerekkel és könyvtárakkal, például React, Vue.js vagy Angular, valamint más szolgáltatásokkal és middleware-ekkel.
      * Könnyű hibakezelés: Az Axios lehetővé teszi a könnyű hibakezelést és hibajelzések kezelését, így az alkalmazások megbízhatóan és stabilan működhetnek még hálózati hibák esetén is.

# Az oldal működése

## Hitelesítés

### Regisztráció

Regisztrációnál a felhasználónak meg kell adnia egy felhasználónevet, az email-jét, a városát és a jelszavát kétszer, ezután pedig, ha sikeres volt a regisztráció kapni fog egy email-t, ahol hitelesítenie kell a fiókját, majd ezután, ha visszatér az oldalra már be tud jelentkezni.



### Bejelentkezés

Ha már van hitelesített felhasználónk, akkor a ”Bejelentkezés” fülön tudunk belépni a fiókunkba, az email és a jelszó segítségével.

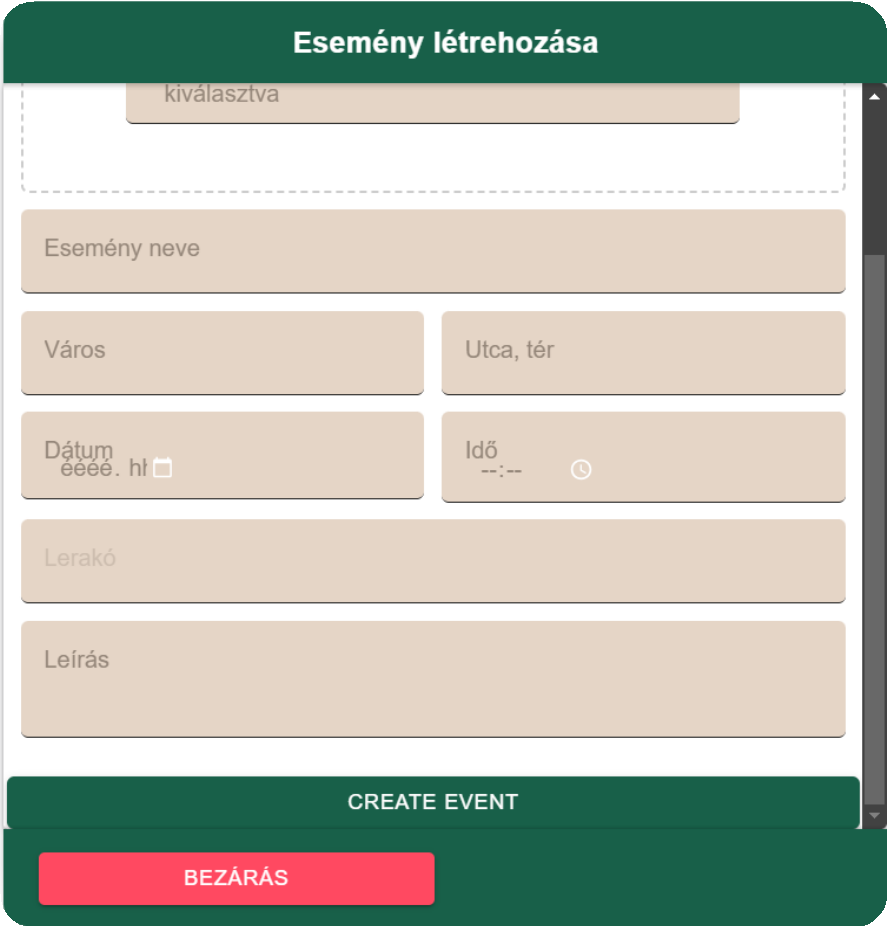
A tesztelés során használt próba jellegű felhasználó adatai:

* + - * email: [testuser@mail.com](mailto:testuser@mail.com)
      * jelszó: Password1

## Események

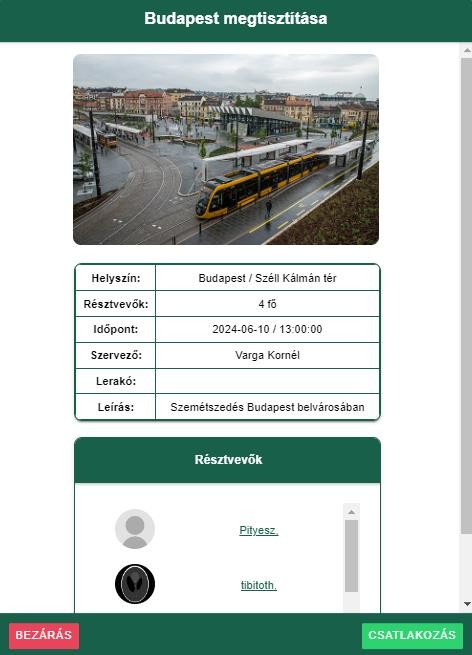
### Esemény létrehozása

Az ”Esemény készítése” fülön tudunk létrehozni különböző eseményeket, itt meg kell adni kötelezően a nevét, a címét, az idejét, valamint igény szerint lehet leírást és képet is hozzáadni. Miután sikeren létrehoztuk az eseményt, az láthatóvá válik az összes többi felhasználó számára az ”Események” fülön.



### Csatlakozás eseményhez

Az ”Események” fülön elérhető az összes létező esemény, itt tudunk bizonyos kategóriák szerint szűrni is pl. város, dátum. Amint meglátunk egy számunkra megfelelő találatot és rákattintunk megjelenik egy kártya amin elérhető lesz a ”Csatlakozás” gomb, ha ezt megnyomjuk és sikeresen csatlakozunk bekerülünk a résztvevők közé és az esemény megjelenik a saját profilunkon is.



### Események szerkesztése és törlése

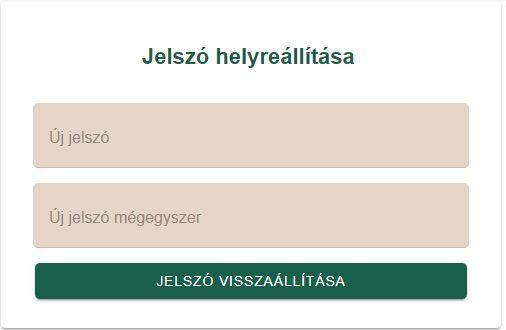
Amennyiben mi vagyunk egy esemény szervezői (mi hoztuk létre), lehetőségünk vannak szerkesztésére vagy törlésére. A saját profilunkon

megkeressük az eseményt és miután rákattintnk megjelennek a ”Szerkesztés” és ”Törlés” gombok.

## Profil

### Jelszó-helyreállítás

Elfelejtett jelszó esetén a ”Bejelentkezés” fülön lehetőségünk van e-mailt küldeni magunknak, az ebben szereplő link megnyitásával pedig új jelszót tudunk megadni a fiókunkhoz.



### Jelszó cseréje

Ha meg akarjuk változtatni a jelenlegi jelszónkat azt a ”Beállítások” közt tudjuk megtenni, itt meg kell adni a régi jelszónkat és az új jelszót is és ezután tudjuk megváltoztatni azt.



### Profilkép változtatás

Sznitén a ”Beállítások” fülön tudjuk a profilképünket megváltoztatni, ehhez meg kell adnunk valamilyen képi-formátumú fájlt és megnyomni a ”Profilkép cseréje” gombot.



### Fiók törlése

Ha meg szeretnénk szüntetni a profilunkat a ”Beállítások” menüben lévő ”Fiók törlése” gombot kell megnyomni, ezután fogunk kapni egy e-mailt és az ebben szereplő 6 jegyű kódot kell megadni az oldalon ahhoz, hogy véglegesítsük a törlést. Ezt abból a célból oldottuk meg így, hogy biztosan csak a felhasználó tudja törölni a fiókját.