# Kizöldítjük a Földet!

## projekt megvalósításának lépésEi Laravel/JS/React

## Tartalomjegyzék

[Kizöldítjük a Földet! 1](#_Toc162902520)

[Tesztelés Thunder Client-tel 9](#_Toc162902521)

[Frontend JS-sel 10](#_Toc162902522)

[Frontend React-tal 16](#_Toc162902523)

### Alapfeltételek:

PHP 8 + Adatbázis (XAMPP csomag tartalmazza, DB: MySQL)

Készíts egy új laravel projektet!

composer create-project laravel/laravel kizolditjukafoldet\_backend

**//composer install**

**xamp indítása**

**localhost/phpmyadmin: adatbázis létrehozása**

**.env** módosítása, hogy a kapcsolódási paraméterek legyenek élesek (DB kapcsolódási paraméterek beállítása)

php artisan serve

**A projekt rendszerint a localhost:8000 proton érhető el.**

## feladat

Kizöldítjük a Földet! címmel weblapot készítünk, amelyben rögzítheti minden osztály minden diákja, hogy milyen tevékenységeket végzett a fenntarthatóság kapcsán.

A programnak most csak a publikus felületét készítjük el, melyen a diákok megadhatják a tevékenységeket.

Az asztali és a mobil verzió mintaképe:

A képen szöveg, képernyőkép, szám, Betűtípus látható

Automatikusan generált leírás A képen szöveg, képernyőkép, szám, Betűtípus látható

Automatikusan generált leírás

A felhasználók és a lehetséges tevékenységek az adatbázisban előre rögzítve vannak, tehát nem feladat ezek rögzítésére felület létrehozása.

Az osztályokat pedig a JS kódban rögzítjük.

**Lehetséges tevékenységek:  (ezek fixen szerepelnek az adatbázisban, elegendő csak néhányat létrehozni ezek közül)**

* kerékpárral jöttem iskolába
* rollerrel jöttem iskolába
* 10 km-t gyalogoltam buszozás helyett
* ültettem fát
* ültettem virágot
* ültettem egyéb növényt
* kevesebb vizet használtam a fürdéshez
* összeszedtem a szemetet egy közterületen, erdőben, stb
* tartós szatyorba vásároltam, nem nylonba
* nem használtam egyszer használatos műanyagot
* tartós csomagolású terméket vásároltam – pl üvegbe vettem a tejet, nem használtam pet palackot,
* környezetbarát csomagolású terméket vásároltam
* kevesebb húst ettem a héten
* ökológiai gazdaságból származó élelmiszert vettem
* kirándultam, szabadban töltöttem időt a héten
* kevesebb ruhát/terméket vásároltam a héten, hogy fenntarthatóbb legyen a környeztünk!
* önkénteskedtem a Greenpeace-nél, vagy más zöld szervezetnél

Minden tevékenységhez tartozik egy pontszám is.

Ez alapján versenyt hirdethetünk az osztályok között a  **Legzöldebb Osztály** címért.

Amikor a felhasználó egy tevékenységet rögzít a rendszerben, akkor növekszik az osztály összpontszáma a kiválasztott tevékenységnek megfelelően.

A táblázat alatt egy listában jelennek meg a rögzített tevékenységek.

**++++Az oldalhoz tartozik egy admin felület, ahol az osztályhoz tartozó tanár jóvá tudja hagyni a tevékenységeket.**

Minden ***új bejegyzéshez***  tartozik:

* + - osztály
    - bejegyzés neve
    - állapot (jóváhagyásra vár/jóváhagyott)

### Táblák

|  |  |  |  |
| --- | --- | --- | --- |
| BEJEGYZES | TEVEKENYSEG | USER |  |
| id | tevekenyseg\_id | id |  |
| tevekenyseg\_id | tevekenyseg\_nev | name |  |
| osztaly\_id | pontszam | password |  |
| allapot |  | email |  |
|  |  | osztaly\_id | Az egyszerűség kedvéért most nem kell osztály tábla. |
|  |  |  |  |

#### API végpontok és feladatok

GET all /bejegyzesek Visszaadja az összes bejegyzést a tevékenység nevével és pontszámával együtt

GET osztaly\_id /bejegyzesek/osztaly\_id Visszaadja egy adott osztály összes bejegyzését a tevékenység nevével és pontszámával együtt

POST /bejegyzes Új bejegyzést rögzíthetünk a táblában

## Backend - laravel

1. Indítsuk el a Xamp-ot.
2. Hozzunk létre egy **-–** adatbázist
3. Ellenőrizzük, hogy a .env fájlban ez az adatbázisnév van megadva!

A képen szöveg, monitor, képernyő, képernyőkép látható

Automatikusan generált leírás

### Adatbázis táblák létrehozása

1. Létre kell hozni egy-egy modelt a táblákhoz

php artisan make:model Tevekenyseg -m

php artisan make:model Bejegyzes -m

1. Létre kell hozni egy-egy controllert is a táblához

php artisan make:controller TevekenysegController -m

php artisan make:controller BejegyzesController -m

**Fájl elérési helyek:**

migrációs fájlok: database/migrations

model fájlok: app/Models

controller fájlok: app/Http/Controllers

view fájlok: resources/views

1. **database/migration** útvonalon létrejött egy új migrációs fájl egy CreateValamilyennevTable osztállyal.   
   Az **up()** metódusában megadjuk a táblákban használni kívánt mezőneveket a típusukkal és a méretükkel együtt. A mezőtípusokhoz itt találunk útmutatót: <https://laravel.com/docs/7.x/migrations>

### Migrációs fájlok

USER

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

TEVEKENYSEG

A képen szöveg, képernyőkép, dokumentum, Betűtípus látható

Automatikusan generált leírás

BEJEGYZESEK

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

1. A tábla migrálása:   
   php artisan migrate
2. Ha módosul a tábla mezője, akkor vissza kell vonni:   
   php artisan migrate:rollback  
   Majd módosítani a tábla mezőit és újra futtatni (3. 4. pont)

### Modellek létrehozása

Egészítsük aki a modell fájlokat az alábbiek szerint:

USER - app/Models/User.php

A képen szöveg, képernyőkép, Betűtípus, tervezés látható

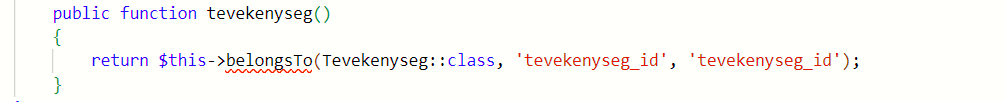
Automatikusan generált leírás

BEJEGYZESEK - app/Models/Bejegyzesek.php

A képen szöveg, Betűtípus, képernyőkép, fehér látható

Automatikusan generált leírás

Mivel a bejegyzséek táblához kapcsolódik a tevékenységek tábla, ezért összekapcsoljuk az alábbi függvénnyel.



TEVEKENYSEGEK - app/Models/Tevekenyseg.php

A képen szöveg, Betűtípus, képernyőkép, fehér látható

Automatikusan generált leírás

### Api útvonalak előállítása

1. routes/api.php fájlban a Route-olás elkészítése
   1. Csatoljuk a BejegyzésekControllert és a TevekenysegekControllert!



* 1. Készítsük el a végpontokat!

A képen szöveg, képernyőkép, Betűtípus, sor látható

Automatikusan generált leírás

### A TevekenysegController

A képen szöveg, képernyőkép, Betűtípus, sor látható

Automatikusan generált leírás

### BejegyzesController

Az index metódusban figyeljünk arra, hogy a bejegyzéseknél a tevekenyseg\_id-hez tartozó tevékenység neveket is kérjük le.

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

# Tesztelés Thunder Client-tel

Példa a post kérés tesztelésére:

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

# Frontend JS-sel

Hozzuk létre a MODEL, CONTROLLER, VIEW mappaszerkezetet!

Az index. html-be importáljuk a JQuery-t, axios és bootstrap CDN-eket.

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

Az index.html-ben létrehozzuk a navigációt és a tárolókat.

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

A main.js-ben pedig a contorllereket példányosítjuka megfelelő helyre. Ez a programunk belépési pontja.

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

## Modell - DataService.js fájl

Ez felel az adatok lekérésért a szerverről, itt vannak az axios hívások: get, post kérések.

A képen szöveg, képernyőkép, szám, szoftver látható

Automatikusan generált leírás

## PublikusController.js

Feladata a modell és a view példányosítása

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

## PublikusView.js

Megjeleníti a táblázatot.

A táblázat megjelenítéséhez a SorView osztályt példányosítja, mely a táblázat egyetlen sorát jelenti.

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

## SorView.js fájl

Feladata a táblázat egyetlen sorának megjelenítése.

A képen szöveg, képernyőkép, szám, Betűtípus látható

Automatikusan generált leírás

## UrlapView osztály

A képen szöveg, képernyőkép, dokumentum, Betűtípus látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

# Frontend React-tal

**Hozzuk létre a react alkalamazást:**

npx create-react-app frontend\_react

**Telepítsük a bootstrap és axios csomagokat.**

npm install react-bootstrap bootstrap

npm install axios

**importáljuk a bootstrappet az index.js fájlba.**

import "bootstrap/dist/css/bootstrap.min.css";

**indítsuk el az applikációnkat**

npm start

**Hozzuk létre a mappaszerkezetet**

A képen szöveg, képernyőkép, Betűtípus, tervezés látható

Automatikusan generált leírás

## Api/Axios.js

Állítsuk be az axios paramétereit

A képen szöveg, képernyőkép, Betűtípus, sor látható

Automatikusan generált leírás

## context/AdatContext.js

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

## Provider használata

Ne felejtsük el az index.js-ben az App komponenst körülölelni a providerünkkkel.

A képen szöveg, Betűtípus, képernyőkép, sor látható

Automatikusan generált leírás

## APP komponens

Alakítsuk ki az oldalunk alapszerkezetetét

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

## Publikus komponens

Megjeleníti a táblázatot. A táblázat megjelenítéséhez a SorView komponenst használja, mely a táblázat egyetlen sorát jelenti.

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

## SorView.js fájl

Feladata a táblázat egyetlen sorának megjelenítése.

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

## Urlap komponens

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás