Jegyzet React használatához

Tartalom

[Előkészítés (Suliban) 2](#_Toc153744185)

[Lépések: 2](#_Toc153744186)

[Új React projekt Létrehozása 2](#_Toc153744187)

[Ha Github-ról szedted le 3](#_Toc153744188)

[Képernyőképek 4](#_Toc153744189)

[Előkészítés (Node.js-sel) 5](#_Toc153744190)

[Lépések 5](#_Toc153744191)

[Új React projekt Létrehozása 5](#_Toc153744192)

[Ha Github-ról szedted le 6](#_Toc153744193)

# Előkészítés (Suliban)

## Lépések:

1. Először megnyitjuk a parancssort (beírjuk a keresőbe ez a leggyorsabb)
   1. Ide beírjuk: **wsl**
   2. **Ezt az alakot nyitva hagyjuk**
2. A böngészőbe beírjuk a következőt:
   1. <https://localhost:9443/>
   2. Itt feljöhet egy ablak, ahol a speciális részen engedélyezni kell.
   3. Tovább
   4. A belépési adatok ezek lesznek
      1. admin
      2. #Aa123456789@
3. A képernyőképeken látható helyen mindent ki kell jelölni, majd leállítani.
   1. Ezek után Stack szerint rendezünk
      1. A következőket kell kijelölni: docker\_react\_laravel (ez lehet, hogy később változni fog docker\_react -ra)
   2. A kiválasztottakat el kell indítani
4. Megnyitjuk a File/Fájl kezelőt
   1. Itt megnyitjuk a Hálózat mappát, és beírjuk a következőt:
      1. [**\\wsl**](file:///\\wsl)
      2. A képen szöveg, képernyőkép, sor, Betűtípus látható

         Automatikusan generált leírás
   2. A következő elérési útvonalat fogjuk használni mindig
   3. \\wsl.localhost\Debian\root\docker\_react\_laravel\.webhtml

### Új React projekt Létrehozása

1. Megnyitjuk a Visual Studio Code-ot
2. Megnyitunk egy új terminált
   1. A képen szöveg, képernyőkép, Betűtípus látható

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

      Automatikusan generált leírás
   3. Ide beírjuk a következőket, sorrend Fontos
      1. **wsl**
      2. **docker ps**
      3. **docker exec -it szam /bin/bash**
      4. **npx create-react-app appneve**
      5. **cd appneve**
      6. **npm run start&**
      7. **exit** kilpéshez, Csak ha ki akarsz lépni!
3. Jó React-zást!

### Ha Github-ról szedted le

A terminálba a következőket beírni:

1. **wsl**
2. **docker ps**
3. **docker exec -it szam /bin/bash**
4. **cd appneve**
5. **npm install**
6. **npm run start&**

# Képernyőképek

A képen szöveg, képernyőkép, szoftver, Számítógépes ikon látható

Automatikusan generált leírásA képen szöveg, képernyőkép, szoftver, Számítógépes ikon látható

Automatikusan generált leírásA képen szöveg, képernyőkép, szoftver, Számítógépes ikon látható

Automatikusan generált leírásA képen szöveg, képernyőkép, szoftver, képernyő látható

Automatikusan generált leírás

Előkészítés (Docker-rel) régi ver

# Előkészítés (Node.js-sel)

## Lépések

1. Ha még nincs Node.js letöltve a számítógépre:
   1. Ezen a linken le lehet tölteni a neked szükségeset
   2. <https://nodejs.org/en/download/>

### Új React projekt Létrehozása

1. Megnyitjuk a Visual Studio Code-ot
2. Megnyitunk egy új terminált
   1. A képen szöveg, képernyőkép, Betűtípus látható

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

      Automatikusan generált leírás
   3. Ide beírjuk a következőket, sorrend Fontos
      1. **npx create-react-app appneve**
      2. **cd appneve**
      3. **npm start**
      4. **itt lesz megtalálható**
      5. <http://localhost:3000/>
      6. Ha készen állsz a deploy-mentre
         1. npm run build

### Ha Github-ról szedted le

A terminálba a következőket beírni:

1. **cd appneve**
2. **cd elérési-útvonal**
3. **npm install**
4. **npm start**

# Jegyzet React-hoz

## Mi a React, rövid ismertető

React egy JavaScript könyvtár, amelyet a felhasználói felületek (UI) készítésére használnak.

A Facebook fejlesztette ki, és segít a könnyen karbantartható, hatékony és gyors felhasználói élmények létrehozásában.

## Alapfogalmak

### Komponensek

React alkalmazások fő építőelemei a komponensek. Ezek önálló, újra felhasználható építőkockák, amelyekből az alkalmazás felépül. Egy komponens lehet egy egyszerű gomb, egy űrlap, vagy akár egy teljes oldal.

#### Szülő komponens

import React, { useState } from 'react';

import ChildComponent from './ChildComponent';

const ParentComponent = () => {

// Állapot a szülő komponensben

const [parentState, setParentState] = useState('Kezdeti érték');

// Eseménykezelő függvény a szülő komponensben

const handleParentClick = () => {

setParentState('Új érték a szülőben');

};

return (

<div>

<h2>Szülő komponens</h2>

<p>Állapot a szülőben: {parentState}</p>

<button onClick={handleParentClick}>Szülőben kattints ide</button>

{/\* Gyerek komponens hozzáadása \*/}

<ChildComponent parentState={parentState} />

</div>

);

};

export default ParentComponent;

#### Gyerek komponens

import React, { useEffect } from 'react';

const ChildComponent = (props) => {

// Egy useEffect példa a gyerek komponensben

useEffect(() => {

console.log('Gyerek komponens frissült');

}, [props.parentState]);

return (

<div>

<h3>Gyerek komponens</h3>

<p>Állapot a szülőből a gyerekben: {props.parentState}</p>

</div>

);

};

export default ChildComponent;

## Rootolás (Otthon JAV)

mappaszerk ki alakítása

npm i -D react-router-dom

### Layout.js kialakítása

(ki tudod szedni előzőekből)

gépel rendez válaszol ű

### BrowserRouter

Importok

import { BrowserRouter, Route, Routes } from 'react-router-dom';

import Layout from './Layout.js';

import Jatekok from './pages/Jatekok.js';

import NoPage from './pages/NoPage.js';

Bootstrap

<https://react-bootstrap.netlify.app/docs/getting-started/introduction/>

<https://create-react-app.dev/docs/adding-bootstrap/>

npm install react-bootstrap bootstrap

index.js

*import* 'bootstrap/dist/css/bootstrap.css';

Navbar

<https://react-bootstrap.netlify.app/docs/components/navbar>

# DOGA

props state componensek egymásba illeszkedik

route nem lesz a dogában még

jövőhát

2 komp

adatok meghívása, esemény kezelés