Vas Megyei Szakképzési Centrum

Nádasdy Tamás Technikum és Kollégium

PROJEKTFELADAT

Irodai naptár

Horváth Márk, Michtits Patrik

Konzulens:
Varga Gábor

2024

Nyilatkozat

Alulírott, Horváth Márk, Michtits Patrik kijelentem, hogy a Irodai naptár című projektfeladat kidolgozása a saját munkám, abban csak a megjelölt forrásokat, és a megjelölt mértékben használtam fel, az idézés szabályainak megfelelően, a hivatkozások pontos megjelölésével.

Eredményeim saját munkán, számításokon, kutatáson, valós méréseken alapulnak, és a legjobb tudásom szerint hitelesek.

Csepreg, 2024.04.22 _____

Hallgató

_____

Hallgató

_____

Kivonat

Irodai Naptár

Célunk, hogy segítsünk a vállalati vezetőknek és alkalmazottaiknak hatékonyabban szervezni és kezelni mindennapi feladataikat és találkozóikat. Legyen szó nagy vagy kis vállalatról, fontos, hogy a munkavállalók hatékonyan kommunikáljanak és szervezzék meg tevékenységeiket, hogy a cég sikeres legyen.

Az Irodai Naptár segítségével könnyen és gyorsan beütemezhetik feladataikat és találkozóikat egy intuitív és felhasználóbarát felületen keresztül. Nemcsak előre betervezhetik tevékenységeiket, hanem visszanézhetik korábbi teendőiket is, így mindig átláthatják és nyomon követhetik a saját és kollégáik feladatait.

Főbb jellemzőink:

Egyszerű feladat- és találkozóütemezés: A naptár segítségével könnyedén beütemezhetik feladataikat és találkozóikat. Egyszerűen csak válasszák ki a dátumot és időpontot, majd írják le a teendőt vagy találkozót. Lehetőség van helyszín és résztvevők hozzáadására is.

Visszatekintés korábbi teendőkre: Fontos, hogy a múltbeli teendőkre is visszatekinthessenek. A naptár lehetővé teszi a korábbi feladatok és találkozók megtekintését, így könnyen nyomon követhetik a már elvégzett munkát és találkozókat.

Együttműködés és megosztás: Lehetőség van a naptárak megosztására és együttműködésre is. Ezáltal a csapat könnyen egyeztetheti tevékenységeit és találkozóit, és mindig mindenki naprakész lehet.

Egyedi beállítások és emlékeztetők: Személyre szabhatják a naptár beállításait, és emlékeztetőket állíthatnak be fontos találkozókra és feladatokra, így biztosítva, hogy semmi ne maradjon figyelmen kívül.

Könnyen elérhető, bárhonnan: A naptár webes felületen és mobilon keresztül is elérhető, így bárhol és bármikor könnyen hozzáférhetnek a legfrissebb információkhoz és teendőkhöz.

Ne hagyja, hogy a napi teendők és találkozók összezavarják Önt vagy csapatát. Használja az Irodai Naptárt, hogy könnyedén szervezhesse meg mindennapi tevékenységeit, és mindig hatékonyan dolgozhasson!

Abstract

Office Calendar

Our goal is to help corporate leaders and their employees organize and manage their daily tasks and meetings more effectively. Whether it's a large or small company, it's important for employees to communicate and organize their activities efficiently for the company to be successful.

With the Office Calendar, they can easily and quickly schedule their tasks and meetings through an intuitive and user-friendly interface. They can not only schedule their activities in advance but also review their previous tasks, ensuring they can always keep track of their own and their colleagues' tasks.

Key features:

Simple task and meeting scheduling: With the calendar, they can easily schedule their tasks and meetings. Simply select the date and time, then describe the task or meeting. There's also an option to add location and participants.

Review of previous tasks: It's important to be able to look back on past tasks. The calendar allows them to view previous tasks and meetings, making it easy to track completed work and meetings.

Collaboration and sharing: They can share calendars and collaborate easily. This allows the team to coordinate their activities and meetings, ensuring everyone is always up to date.

Custom settings and reminders: They can customize the calendar settings and set reminders for important meetings and tasks, ensuring nothing is overlooked.

Easily accessible from anywhere: The calendar is accessible via web and mobile, so they can easily access the latest information and tasks anytime, anywhere.

Don't let daily tasks and meetings overwhelm you or your team. Use the Office Calendar to easily organize your daily activities and work efficiently at all times!

Tartalomjegyzék

1. Bevezetés

A 2024-es záróvizsgánk beadandójához témának egy o irodai naptár weblapot választottam, amely túlmutat az egyszerű időbeosztáson és az események megjelenítésén. Fő célja az, hogy segítse a felhasználókat az időhatékony munkavégzésben, a hatékony csapatmunkában és a feladatok hatékony kezelésében, egy egyszerű felületen. Ez a weblap a nagyvállalatok, kisebb cégek hatékony időbeosztását és feladat kezelését segítené, a nagyobb vezetők és a beosztottak életét, munka végzését is meg könnyítené. A hatékonyabb munkavégzés céljával hoztuk létre . Észrevételünk szerint nem minden kisseb, nagyobb vállalatnál hatékony, professzionális a munkavégzés mert hiszen sokan manapság még az elavult levelező rendszereket használnak az országon belül, és kívül is, Azért alkudtuk meg ezt a weblapot hogy könnyű, letisztult, felhasználóbarát felületet teremtsünk a munkavállalók, dolgozok életébe hogy minél egyszerűbben tudják kezelni mindennapi feladataikat, teendőiket a munkavilágában. Mellékesen saját egyéni célra is lehet használni az irodai weblapunkat bár a weblap fő célja hogy a nagyobb, kisseb vállalatok, cégek életét könnyítse meg. Régebben programozás órán a tanárunk készített egy szabadságolós naptár programot, ami nagyon tetszett. Ez is motivált arra, hogy ezt válasszam a záró dolgozatom témájának.

2. Hasonló weblapoldalak

Az irodai naptárhoz hasonló weboldalak széles körben elterjedtek az interneten, és számos vállalat és szolgáltató kínál hasonló szolgáltatásokat. Ezek az oldalak általában számos funkciót kínálnak az időmenedzsmentre, az események ütemezésére és a feladatok kezelésére.

3. Felhasználó dokumentáció

3.1. Regisztrációs Űrlap:

A Regisztrációs Űrlap egy felhasználói felület, amely lehetővé teszi a felhasználók számára, hogy új fiókot hozzanak létre az weblap.
A felhasználók megadhatják az email címüket, jelszavukat és a jelszó megerősítését.
Az űrlapon található mezők validálásra kerülnek, és hibaüzenetek jelennek meg, ha valamelyik mező nincs kitöltve vagy ha a jelszavak nem egyeznek meg.
Az űrlap alján található egy "Regisztráció" gomb, amelynek megnyomására a regisztrációs adatok ellenőrzése és küldése történik.

3.2. Bejelentkezési Űrlap:

A Bejelentkezési Űrlap lehetővé teszi a felhasználók számára, hogy belépjenek a weboldal az email címük és jelszavuk megadásával.
Az űrlapon található mezők validálásra kerülnek, és hibaüzenetek jelennek meg, ha a mezők nincsenek megfelelően kitöltve.
Az űrlap alján található egy "Bejelentkezés" gomb, amelynek megnyomására a bejelentkezési adatok ellenőrzése és küldése történik.

3.3. Fontos Megjegyzések:

Mindkét űrlap a Material-UI által biztosított TextField komponensét használja az input mezők megjelenítésére.
A jelszavak mezőihez hozzáadtam a jelszó láthatóság funkciót, amely lehetővé teszi a felhasználók számára, hogy megtekinthessék vagy elrejtsék a beírt jelszavakat.
A felhasználói interakciókat a weboldal useState hookjaival kezelem, amelyek segítségével nyomon követhetem és frissíthetem a komponensek állapotát.

3.4. Felhasználói Profil:

A Felhasználói Profil komponens lehetővé teszi a felhasználók számára, hogy megtekintsék és szerkesszék a saját profiladataikat az weblapbon.
A felhasználók képesek megtekinteni a felhasználói nevüket, email címüket és egyéb profilinformációkat.
Az oldalon lehetőség van a profilinformációk szerkesztésére, például a felhasználónév vagy email cím módosítására.
A felhasználók jelszavukat is frissíthetik ezen az oldalon keresztül.
A Profil oldalon található gombok segítségével a felhasználók könnyen elmenthetik a módosításokat vagy visszavonhatják azokat.
Főbb Jellemzők:

4. . Fejlesztői dokumentáció

4.1. weboldal főbb jellemzői

Egyszerű kezelőfelület: A weboldal felhasználói számára intuitív és könnyen kezelhető felületet biztosít az események és feladatok kezelésére.
Rugalmas ütemezés: Lehetőség van a felhasználók számára saját időbeosztásuk alapján eseményeket és találkozókat ütemezni, módosítani vagy törölni.
Csapatmunka támogatása: A weboldal lehetővé teszi a felhasználók számára, hogy megosszák naptárjukat más csapattagokkal, így könnyen egyeztethetik a tevékenységeket és találkozókat.
Testreszabható emlékeztetők: A felhasználók személyre szabhatják az emlékeztetők beállításait, így soha nem felejthetnek el fontos eseményeket vagy feladatokat.
Összességében az weblap egy modern és hatékony eszköz a vállalati időmenedzsmenthez és csapatmunkához, amely segíti a felhasználókat a hatékonyabb munkavégzésben és a jobb együttműködésben.

4.2. Tesztelés és minőségellenőrzés:

A weboldal minőségének biztosítása érdekében alapos tesztelési folyamatokat alkalmaztam. Az egységtesztek segítségével ellenőriztem a komponensek helyes működését és funkcionalitását. Emellett integrációs teszteket is futtattam, hogy megbizonyosodjak arról, hogy az egyes részek jól működnek együtt. A felhasználói felületet alaposan ellenőriztem különböző eszközökön és böngészőkben, hogy biztos legyek a felhasználói élmény egyenletes és megfelelő.

4.3. Teljesítményoptimalizálás:

A weboldal teljesítményének optimalizálása érdekében törekedtem a kódbázis hatékonyságára és a gyors betöltési időre. Csoportosítottam és minimalizáltam a JavaScript és CSS fájlokat, valamint optimalizáltam a képek és egyéb erőforrások méretét. Emellett

kihasználtam a böngésző gyorsítótár funkcióit és az adatok gyorsítótárazását, hogy a weboldal még reaktívabb legyen a felhasználók számára.

4.4. Biztonság és adatvédelem:

Az adatbiztonság és a felhasználók személyes adatainak védelme kiemelt fontosságú volt a weboldal fejlesztése során. Szigorú biztonsági intézkedéseket alkalmaztam az adatbázisban és a hálózati kommunikáció során is. Továbbá gondoskodtam arról, hogy a weboldal megfeleljen a GDPR és egyéb adatvédelmi előírásoknak, és hogy a felhasználók mindig teljes körű ellenőrzést és átláthatóságot kapjanak adataik kezelése tekintetében.

4.5. Kiterjesztések és jövőbeli tervek:

A weboldal folyamatos fejlesztése és bővítése érdekében számos kiterjesztést és továbbfejlesztést tervezek a jövőben. Ezek közé tartozik új funkciók hozzáadása, mint például videókonferencia integráció, csoportos feladatkezelés vagy okostelefonos alkalmazások kifejlesztése. Emellett tervezem a weboldal nemzetközi piacra való kiterjesztését is, hogy minél több felhasználó számára elérhető legyen az irodai hatékonyságot növelő eszköz.

4.6. Termékfejlesztés és Kódolás:

4.7. Frontend Architektúra:

A weboldal frontend architektúrájának tervezése során nagy hangsúlyt fektettem a modularitásra és a skálázhatóságra. Példaként említhetjük a komponensalapú struktúrát, ahol minden funkcióhoz külön komponens tartozik. Ez lehetővé teszi a könnyű újrafelhasználást és a fejlesztési folyamatok egyszerűsítését.

```
// Példa komponens: Események listája
import React from 'react';

const EventList = ({ events }) => {
```

```
  return (
    <ul>
      {events.map(event => (
        <li key={event.id}>{event.title}</li>
      ))}
    </ul>
  );
};

export default EventList;
```

4.8. Backend Adatbázis-kezelés:

A weboldal háttéradatbázis-kezelése során fontos volt a hatékonyság és a biztonság. A MongoDB adatbázist választottuk, és a Mongoose ORM segítségével könnyedén kezeljük az adatokat. Például az alábbi kód egy egyszerű esemény adatmodellt definiál a MongoDB-hez.

```
// Példa adatmodell: Esemény
const mongoose = require('mongoose');

const eventSchema = new mongoose.Schema({
  title: { type: String, required: true },
  date: { type: Date, required: true },
  description: String,
  location: String,
  // ...
});

const Event = mongoose.model('Event', eventSchema);

module.exports = Event;
```

4.9. Felhasználókezelés és Hitelesítés:

A weboldal felhasználókezeléséhez és hitelesítéséhez JSON Web Token (JWT) alapú megoldást alkalmaztunk. Az alábbi kód például egy middleware, amely ellenőrzi a kérések hitelességét a JWT token alapján.

```
// Példa JWT middleware
const jwt = require('jsonwebtoken');
const config = require('../config');

function verifyToken(req, res, next) {
  const token = req.headers['authorization'];
  if (!token) return res.status(403).send({ auth: false, message: 'Nincs token megadva.' });

  jwt.verify(token, config.secret, function(err, decoded) {
    if (err) return res.status(500).send({ auth: false, message: 'Hitelesítés sikertelen.' });

    req.userId = decoded.id;
    next();
  });
}

module.exports = verifyToken;
```

4.10. API Fejlesztés és Dokumentáció:

A weboldal backend részéhez RESTful API-t fejlesztettünk, és Swagger segítségével dokumentáltuk az API végpontokat. Példaként az alábbi kód egy egyszerű API végpontot definiál, amely visszaadja az összes eseményt a rendszerből.

```
// Példa API végpont: Összes esemény lekérése
const express = require('express');
const router = express.Router();
const Event = require('../models/event');
```

```
router.get('/events', async (req, res) => {
  try {
    const events = await Event.find();
    res.json(events);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});

module.exports = router;
```

4.11. Tesztelés és Minőségbiztosítás:

A kód minőségét és a hibák elkerülését számos tesztelési módszerrel biztosítottuk. Példaként az alábbi kód egy egységtesztet mutat be, amely ellenőrzi, hogy a regisztrációs űrlap helyesen validálja-e az email címeket.

```
// Példa egységteszt: Regisztrációs űrlap validációja
import { validateEmail } from './validation';

test('Email validáció', () => {
  expect(validateEmail('example@example.com')).toBe(true);
  expect(validateEmail('invalidemail')).toBe(false);
});
```

4.12. Teljesítményoptimalizálás:

A weboldal teljesítményének optimalizálása érdekében számos módszert alkalmaztunk, például gyorsítótárazást és kódminifikálást. Az alábbi kód egy példát mutat a gyorsítótárazásra a Node.js szerveroldalon.

```
// Példa gyorsítótárazás Node.js-ben
```

```
const express = require('express');
const app = express();

app.use(express.static('public', { maxAge: 31557600 }));
```

4.13. Felhasználói Interfész és CSS:

Az weboldal felhasználói felületének kialakításához és stílusozásához Sass-t használtunk. Az alábbi kód egy példát mutat egy egyszerű stíluslapra Sass-ban.

```
// Példa Sass stíluslap
$primary-color: #007bff;

.button {
  background-color: $primary-color;
  color: white;
  padding: 10px 20px;
  border: none;
  border-radius: 5px;
}
```

4.14. Frontend Állapotkezelés:

A weboldal frontend részén a React Context API-t használtuk az állapotkezelésre. Az alábbi kód példaként szolgál egy egyszerű állapotkezelő kontextusra React-ban.

```
// Példa állapotkezelő kontextus React-ban
import React, { createContext, useState } from 'react';

export const UserContext = createContext();

export const UserProvider = ({ children }) => {
```

```
  const [user, setUser] = useState(null);

  return (
    <UserContext.Provider value={{ user, setUser }}>
      {children}
    </UserContext.Provider>
  );
};
```

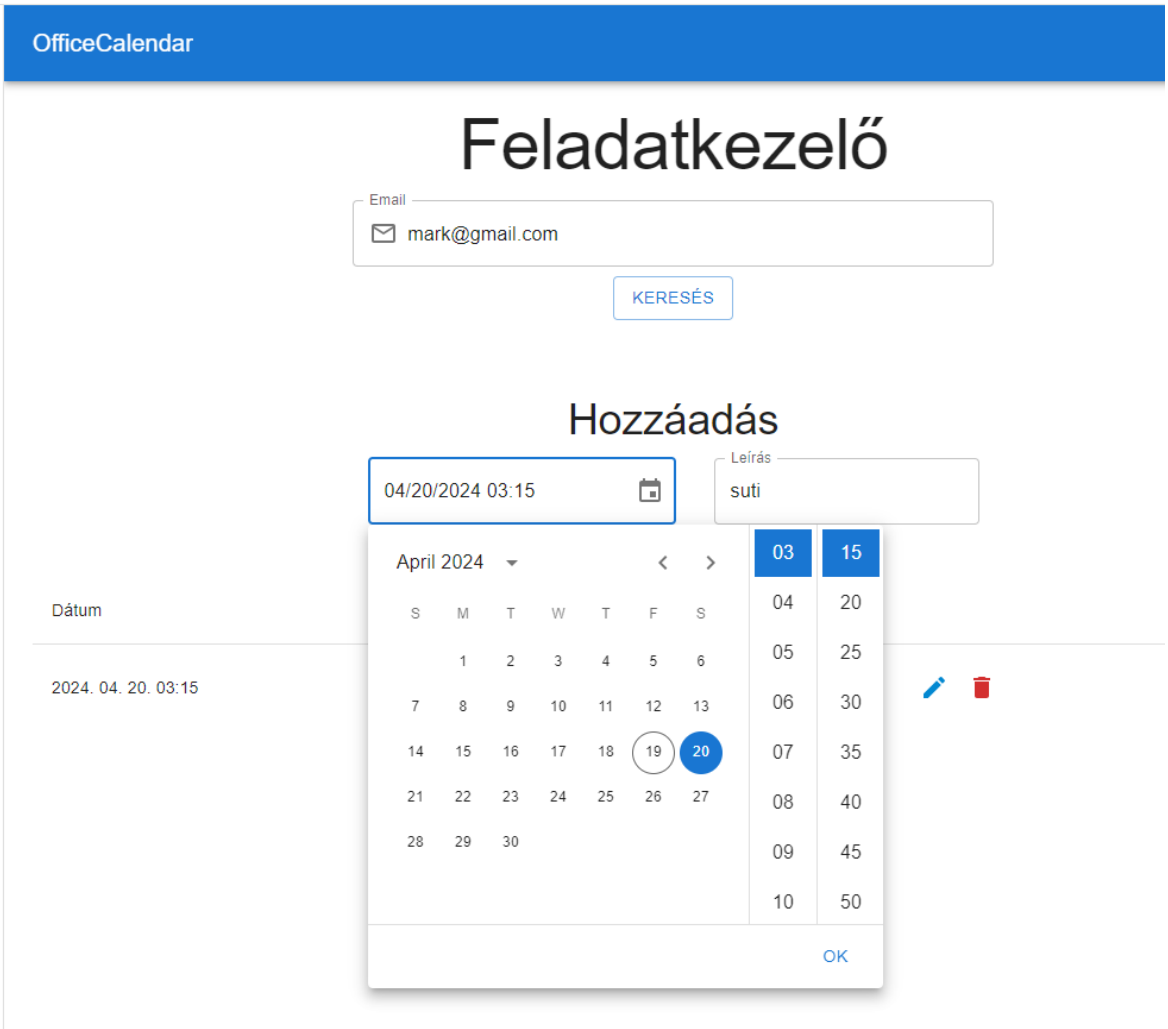
Continuous Integration és Deployment (CI/CD): A weboldal fejlesztése és frissítése során Continuous Integration és Deployment (CI/CD) folyamatot alkalmaztunk. Az alábbi példa egy egyszerű GitHub Actions konfigurációt mutat be a CI/CD folyamat automatizálására.

```
# Példa GitHub Actions konfiguráció
name: CI/CD

on:
 push:
  branches:
   - main

jobs:
 build:
  runs-on: ubuntu-latest
  steps:
   - name: Checkout repository
     uses: actions/checkout@v2
   - name: Install dependencies
     run: npm install
   - name: Build
     run: npm run build
   - name: Deploy
```

```
uses: JamesIves/github-pages-deploy-action@4.1.5
with:
  ACCESS_TOKEN: ${{ secrets.ACCESS_TOKEN }}
  BRANCH: gh-pages
  FOLDER: build
```

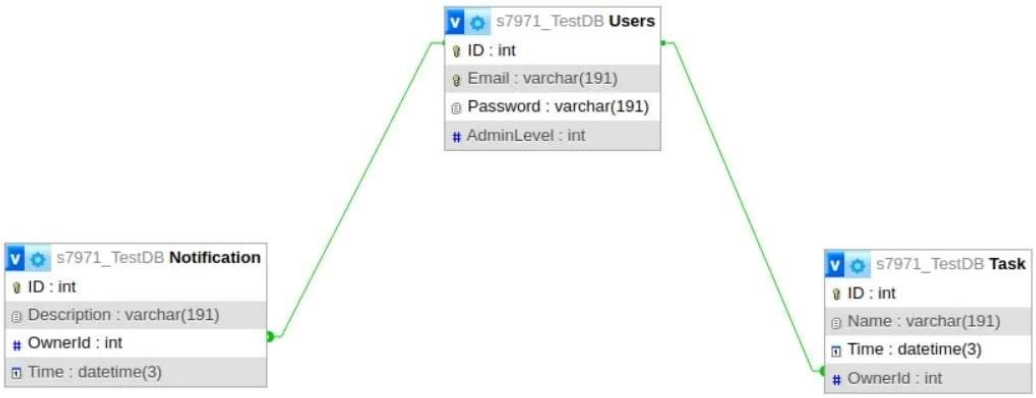
5. Az weblap designja

A weblap megtervezésénél az egyszerű minimalista desingra törekedtünk a weboldal két főszínből áll az első és a fehér a weblap egész háttere fehér. A weboldal felső sávjának kék színt állítottunk be ahol mag a weboldal neve jelenik meg. Weboldalunkban vannak másodlagos színek amik szinte el vesznek az emberi szemek mert nem feltűnően és szemet szúróan használtunk ezek a színek pedig nem más mint a szürke és a piros a szürke szint a weboldal menü sávjának raktuk be ahol a felhasználó a különböző módokba azaz funkciókba tud belépni a piros szint mint a törlés funkcióra használtuk .



1.ábra A weboldal kinézetéről és szín világáról

6. Az adatbázis



2. ábra Az adatbázis tábla és összekötések(a mellékelt képen láthatók)

Az adatbázisban három darab adatbázis táblát hoztam létre az adatbázis táblák nevei:

Az adatbázis első táblájának a Notification nevet adtunk ami négy columot az az oszlopot tartalmaz az oszlopok nevei a következők ID, Description, OwnerId, Time az első oszlop az ID amihez egy int típusú adat változót adtunk hozzá. A Notification tábla második oszlopát Descriptionak neveztük el amihez a varchar változót állítottunk be amihez azt állítottuk be hogy 191 karaktert lehessen bevinni.

A Notification tábla harmadik oszlopának Ownerlid nevet adtunk és be állítottunk neki egy int típusú változót. A Notification tábla negyedik oszlopának a Time nevet adtuk amihez hozzá rendeltünk egy datetime nevű változót.

Az adatbázis második táblájának a Users (felhasználók) nevet adtuk amiben létrehoztunk három oszlopot aminek a következő neveket adtuk ID, Email, Password, AdminLevel. AZ első oszlop ID oszlop amihez egy int típusú változót változót adtunk hozzá . A Users tábla második oszlopának az Email nevet adtunk amihez hozzá rendeltünk egy varchar típusú változót. A Users tábla harmadik oszlopának a Password nevet adtuk amihez hozzá rendeltünk egy varchar típusú változót . A Users tábla negyedik oszlopának az AdminLevel nevet adtunk amihez hozzá rendeltünk egy int típusu változót.

Az adatbázis harmadik táblájának a Task nevet adtam és négy oszlopot hoztam létre ID, Name, Time, OwnerId néven.

A Users tábla az alábbi oszlopokkal rendelkezik:

- ID: Egyedi azonosító, egész szám, automatikusan növekvő érték.
- Email: Szöveges típusú oszlop, egyedi értékkel rendelkezik.
- Password: Szöveges típusú oszlop.
- AdminLevel: Egész szám, alapértelmezett értéke 0.
- Tasks: Kapcsolódik a Task táblához.
- Notifications: Kapcsolódik a Notification táblához.

A Task tábla az alábbi oszlopokkal rendelkezik:

- ID: Egyedi azonosító, egész szám, automatikusan növekvő érték.
- Name: Szöveges típusú oszlop.
- Time: Dátum/idő típusú oszlop.
- OwnerId: Egész szám, tulajdonos azonosítója, a Users tábla ID oszlopára mutat.
- Owner: Kapcsolat a Users táblához, a tulajdonos azonosítójára hivatkozik.

A Notification tábla az alábbi oszlopokkal rendelkezik:

- ID: Egyedi azonosító, egész szám, automatikusan növekvő érték.
- Description: Szöveges típusú oszlop.
- Time: Dátum/idő típusú oszlop, alapértelmezett értéke a jelenlegi időpont.
- OwnerId: Egész szám, tulajdonos azonosítója, a Users tábla ID oszlopára mutat.
- Owner: Kapcsolat a Users táblához, a tulajdonos azonosítójára hivatkozik.

7. A Route-ok a backendben

Notification

Importok és környezeti változók beállítása:

- Az **@prisma/client** modulból importálja a **PrismaClient** típust, amely az adatbázis műveletek végrehajtására szolgál.
- Az **express**, **Router**, **Request** és **Response** modulokat importálja az Express keretrendszer használatához.
- A **bcrypt** és **jsonwebtoken** modulokat importálja a jelszavak titkosításához és JWT autentikációhoz.
- Az **dotenv** modult használja a környezeti változók beállításához.

Express útvonalkezelő inicializálása és Prisma kliens létrehozása:

- Létrehoz egy Express útvonalkezelőt és egy Prisma klienst.

GET végpont (/get):

- Ezen az útvonalon keresztül lehet lekérdezni egy felhasználó értesítéseit.
- Ellenőrzi, hogy van-e megfelelő JWT token a kérésben, különben **401 Unauthorized** választ küld.
- A JWT tokent ellenőrzi és kinyeri belőle az azonosítót.
- Lekéri a felhasználó adatait a Prisma segítségével és visszaadja az értesítéseit.

DELETE végpont (/delete):

- Ezen az útvonalon keresztül lehet törölni egy értesítést.
- Ellenőrzi, hogy van-e megfelelő JWT token a kérésben, különben **401 Unauthorized** választ küld.
- A JWT tokent ellenőrzi és kinyeri belőle az azonosítót.
- Törli az értesítést a megadott azonosító alapján, figyelembe véve a felhasználóhoz tartozó azonosítót.

Hibakezelés:

- Minden útvonal esetén megfelelő hibakezelést végez, hogy biztosítsa a megfelelő válaszokat a különböző hibás kérésekre.

Felhasználó Értesítéskezelése

- Ez a modul felelős a felhasználókhoz kapcsolódó értesítések kezeléséért és

műveletek végrehajtásáért.

- **/get - Felhasználó értesítéseinek lekérése**
- **Metódus: GET**
- **Útvonal: /get**
- **Leírás:** Ezen az útvonalon keresztül lehet lekérni egy felhasználóhoz tartozó értesítéseket.
- **Funkciók:**
    - Ellenőrzi a felhasználó bejelentkezett-e, ellenkező esetben **401 Unauthorized** választ küld.
    - Lekéri a felhasználóhoz tartozó értesítéseket a Prisma segítségével és visszaadja azokat a kérésre. Az értesítések időrendben vannak rendezve a legfrissebbtől a legrégebbiig.

- **/delete - Értesítés törlése**
- **Metódus: DELETE**
- **Útvonal: /delete**
- **Leírás:** Ezen az útvonalon keresztül lehet törölni egy értesítést.
- **Funkciók:**
    - Ellenőrzi a felhasználó bejelentkezett-e, ellenkező esetben **401 Unauthorized** választ küld.
    - Törli a megadott azonosítóval rendelkező értesítést a Prisma segítségével és visszajelzést ad a sikerességről.
- Ez a dokumentáció részletezi az alkalmazás felhasználókhoz kapcsolódó értesítések kezeléséért felelős útvonalakat és műveleteket. A Prisma segítségével biztosítja az adatbázisműveletek átláthatóságát és hatékonyságát. Az útvonalak csak bejelentkezett felhasználók számára elérhetők.

Task

Felhasználók Feladatkezelése

Ez a modul felelős a felhasználókhoz kapcsolódó feladatok kezeléséért és műveletek végrehajtásáért.

/getUserTask - Felhasználó feladatainak lekérése

- **Metódus: POST**
- **Útvonal: /getUserTask**
- **Leírás:** Ezen az útvonalon keresztül lehet lekérni egy felhasználóhoz tartozó feladatokat.
- **Funkciók:**
    - Ellenőrzi a felhasználó bejelentkezett-e, ellenkező esetben **401 Unauthorized** választ küld.
    - Ellenőrzi a felhasználó adminisztrátori jogosultságait, csak az adminisztrátorok használhatják ezt a funkciót.
    - Lekéri a felhasználóhoz tartozó feladatokat a Prisma segítségével és visszaadja azokat a kérésre.

/delete - Feladat törlése

- **Metódus: DELETE**
- **Útvonal: /delete**
- **Leírás:** Ezen az útvonalon keresztül lehet törölni egy feladatot.
- **Funkciók:**
    - Ellenőrzi a felhasználó bejelentkezett-e, ellenkező esetben **401 Unauthorized** választ küld.
    - Ellenőrzi a felhasználó adminisztrátori jogosultságait, csak az adminisztrátorok használhatják ezt a funkciót.
    - Törli a megadott azonosítóval rendelkező feladatot a Prisma segítségével és visszajelzést ad a sikerességről.

/edit - Feladat módosítása

- **Metódus: PUT**
- **Útvonal: /edit**
- **Leírás:** Ezen az útvonalon keresztül lehet módosítani egy feladatot.
- **Funkciók:**
    - Ellenőrzi a felhasználó bejelentkezett-e, ellenkező esetben **401 Unauthorized** választ küld.
    - Ellenőrzi a felhasználó adminisztrátori jogosultságait, csak az adminisztrátorok használhatják ezt a funkciót.
    - Módosítja a megadott azonosítóval rendelkező feladatot a Prisma segítségével és visszajelzést ad a sikerességről.

/create - Új feladat létrehozása

- **Metódus: POST**
- **Útvonal: /create**
- **Leírás:** Ezen az útvonalon keresztül lehet létrehozni egy új feladatot.
- **Funkciók:**
    - Ellenőrzi a felhasználó bejelentkezett-e, ellenkező esetben **401 Unauthorized** választ küld.
    - Ellenőrzi a felhasználó adminisztrátori jogosultságait, csak az adminisztrátorok használhatják ezt a funkciót.
    - Létrehozza az új feladatot a megadott e-mail címhez tartozó felhasználóhoz a Prisma segítségével és visszajelzést ad a sikerességről.

/getOwnDay - Felhasználó feladatainak lekérése egy adott napon

- **Metódus: POST**
- **Útvonal: /getOwnDay**
- **Leírás:** Ezen az útvonalon keresztül lehet lekérdezni egy felhasználóhoz tartozó feladatokat egy adott napon.
- **Funkciók:**
    - Ellenőrzi a felhasználó bejelentkezett-e, ellenkező esetben **401**

Unauthorized választ küld.

- Lekéri a felhasználóhoz tartozó feladatokat az adott napon a Prisma segítségével és visszaadja azokat a kérésre.

Ez a dokumentáció részletezi az alkalmazás felhasználókhoz kapcsolódó feladatkezeléséért felelős útvonalakat és műveleteket. A Prisma segítségével biztosítja az adatbázisműveletek átláthatóságát és hatékonyságát. Az útvonalak csak bejelentkezett felhasználók számára elérhetők, és az adminisztrátori jogosultságok ellenőrzését is biztosítják.

Users

Felhasználókezelés

Ez a modul felelős a felhasználókkal kapcsolatos műveletek végrehajtásáért és az azokhoz kapcsolódó funkciók biztosításáért.

/create - Felhasználó létrehozása

- **Metódus: POST**
- **Útvonal: /create**
- **Leírás:** Ezen az útvonalon keresztül lehet új felhasználót létrehozni.
- **Funkciók:**
  - Ellenőrzi, hogy az adott e-mail cím foglalt-e, és ha igen, hibaüzenetet küld (**500 Internal Server Error**).
  - Létrehozza az új felhasználót a megadott e-mail címmel és jelszóval a Prisma segítségével.

/login - Bejelentkezés

- **Metódus: POST**
- **Útvonal: /login**
- **Leírás:** Ezen az útvonalon keresztül lehet bejelentkezni egy felhasználói fiókba.
- **Funkciók:**

  - Ellenőrzi, hogy a megadott e-mail cím és jelszó páros megfelel-e valamely felhasználónak az adatbázisban.
  - Ha a bejelentkezés sikeres, JWT tokent generál a felhasználó azonosításához és visszaadja a kliensnek.

/logout - Kijelentkezés

- **Metódus: POST**
- **Útvonal: /logout**
- **Leírás:** Ezen az útvonalon keresztül lehet kijelentkezni egy felhasználói fiókból.
- **Funkciók:**
  - Törli a felhasználó JWT tokenjét a kliens oldalon, így kijelentkezett állapotba kerül.

/userData - Felhasználó adatainak lekérése

- **Metódus: GET**
- **Útvonal: /userData**
- **Leírás:** Ezen az útvonalon keresztül lehet lekérni egy felhasználó adatait.
- **Funkciók:**
  - Lekéri az aktuális felhasználó adminisztrátori jogosultságát és visszaadja azt a kliensnek.

/getUser - Felhasználó adatainak lekérése az e-mail cím alapján

- **Metódus: POST**
- **Útvonal: /getUser**
- **Leírás:** Ezen az útvonalon keresztül lehet lekérni egy felhasználó adatait az e-mail címe alapján.
- **Funkciók:**
  - Ellenőrzi, hogy a kérő felhasználónak van-e jogosultsága a keresett felhasználó adatainak lekérésére.
  - Lekéri a felhasználó adatait az e-mail cím alapján és visszaadja azokat a kliensnek.

/changeAdminLevel - Felhasználó adminisztrátori jogosultságának módosítása

- **Metódus: PUT**
- **Útvonal: /changeAdminLevel**
- **Leírás:** Ezen az útvonalon keresztül lehet módosítani egy felhasználó adminisztrátori jogosultságát.
- **Funkciók:**
    - Ellenőrzi, hogy a kérő felhasználó rendelkezik-e megfelelő jogosultsággal a művelet végrehajtásához.
    - Módosítja a felhasználó adminisztrátori jogosultságát a megadott értékre és visszajelzést ad a sikerességről.

/validate - Felhasználó bejelentkezett állapotának ellenőrzése

- **Metódus: GET**
- **Útvonal: /validate**
- **Leírás:** Ezen az útvonalon keresztül lehet ellenőrizni, hogy a felhasználó bejelentkezett-e.
- **Funkciók:**
    - Ellenőrzi, hogy a kliens oldali token érvényes-e, és ha igen, visszaadja az "toDashboard" üzenetet a kliensnek.

Ez a dokumentáció részletezi az alkalmazás felhasználókkal kapcsolatos műveletekért felelős útvonalakat és az azokhoz tartozó funkcionalitásokat. A Prisma segítségével biztosítja az adatbázisműveletek hatékonyságát és biztonságát. Az útvonalak csak bejelentkezett felhasználók számára elérhetők.

8. Funkciók

Bejelentkezés/regisztráció

Bejelentkezéskor a vagy regisztrációkor be kell írni az adott ügyfél email címét és be ír egy jelszót ami minimum nyolc karakter és számot és nagybetűt is tartalmaz és úgy tud bejelentkezni a tulajdonos tudja beállítani a jogosultságokat három féle jogosultság van felhasználó, adminisztrátor , tulajdonos .

Feladat hozzáadása

Új feladatot úgy lehet hozzá a adni ha először rá keresünk az illető email cimére és utána kezdjük belé írni írni a feladatot és az idejét

Értesítések

Minden új feladat hozzáadásakor értesítés kap az adott felhasználó és törölni is tudja az értesítést.

A naptár menüpont

A naptár menüpontba amikor belép az adott felhasználó akkor látja a naptárban karikázva az adott napot és ha rákattint akkor ki is hozza az idejét hogy mire van kiadva az adott feladat ha esetleg van feladata .

Kijelentkezés

Ha a felhasználó rákatint a kijelentkezés gombra akkor automatikusan vissza dobja a felhasználót a bejelentkező/regisztrációs felületre felületre a honnan a felhasználó ujra be tud jelentkezni vagy létre tud hozni egy új felhasználói fiokot

9. Felhasználói jogosultságok és funkciók

Tulajdonos

A tulajdonos a legmagasabb szintű felhasználó, aki teljes körű jogosultsággal rendelkezik az alkalmazásban. Ő felelős az alkalmazás általános üzemeltetéséért, valamint a felhasználók és feladatok kezeléséért.

- **Felhasználókezelés:** A tulajdonos létrehozhat, módosíthat és törölhet felhasználói fiókokat.
- **Feladatkezelés:** Teljes körű hozzáférést kap az összes feladat létrehozásához, módosításához és törléséhez.
- **Értesítések:** Megkapja az összes értesítést a rendszerben történt eseményekről, például új feladatok létrehozása vagy felhasználói fiókok módosítása esetén.

Adminisztrátor

Az adminisztrátor a középszintű felhasználó, akinek jogosultságai korlátozottak a tulajdonoshoz képest, de még mindig jelentős szerepet tölt be az alkalmazás kezelésében.

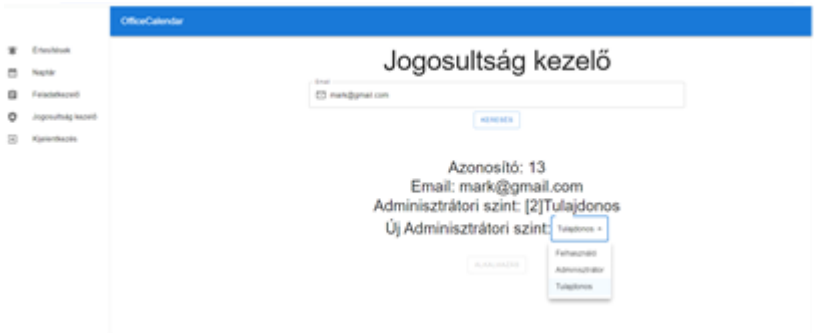
- **Feladatkezelés:** Az adminisztrátor hozzáférést kap az összes feladat létrehozásához, módosításához és törléséhez.
- **Értesítések:** Megkapja az értesítéseket azokról az eseményekről, amelyek a feladatkezelés területén történnek, például feladatok módosítása vagy törlése.

Felhasználó

A felhasználó az alapvető felhasználói szint, aki főként a naptár- és értesítési funkciókat használja az alkalmazásban.

- **Naptár:** A felhasználó képes megnézni a saját naptárát, amely tartalmazza az összes számára kijelölt feladatot és eseményt.
- **Értesítések:** Megkapja az értesítéseket azokról az eseményekről, amelyek érintik a saját feladatait vagy fiókját, például új feladatok létrehozása vagy módosítása.

10. Felhasználói jogosultságok és funkciók

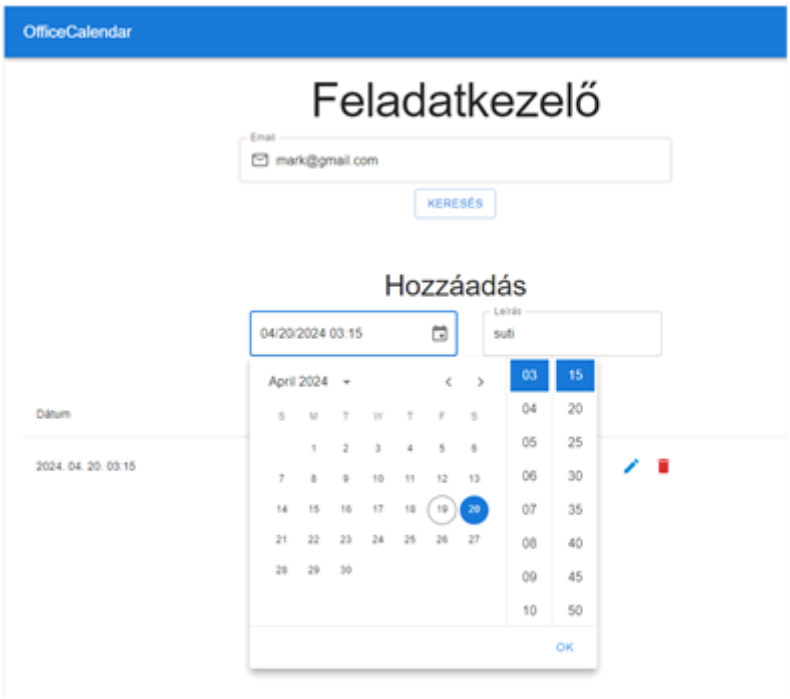


3. árba a jogosultság kezelő

Tulajdonos

A tulajdonos a legmagasabb szintű felhasználó, aki teljes körű jogosultsággal rendelkezik az alkalmazásban. Ő felelős az alkalmazás általános üzemeltetéséért, valamint a felhasználók és feladatok kezeléséért.

- **Felhasználókezelés:** A tulajdonos létrehozhat, módosíthat és törölhet felhasználói fiókokat.
- **Feladatkezelés:** Teljes körű hozzáférést kap az összes feladat létrehozásához, módosításához és törléséhez.



4. árba a feladatkezelő és funkciói

- **Értesítések:** Megkapja az összes értesítést a rendszerben történt eseményekről, például új feladatok létrehozása vagy felhasználói fiókok módosítása esetén.

Adminisztrátor

Az adminisztrátor a középszintű felhasználó, akinek jogosultságai korlátozottak a tulajdonoshoz képest, de még mindig jelentős szerepet tölt be az alkalmazás kezelésében.

- **Feladatkezelés:** Az adminisztrátor hozzáférést kap az összes feladat létrehozásához, módosításához és törléséhez.
- **Értesítések:** Megkapja az értesítéseket azokról az eseményekről, amelyek a feladatkezelés területén történnek, például feladatok módosítása vagy törlése.

Felhasználó

A felhasználó az alapvető felhasználói szint, aki főként a naptár- és értesítési funkciókat használja az alkalmazásban.

- **Naptár:** A felhasználó képes megnézni a saját naptárát, amely tartalmazza az

összes számára kijelölt feladatot és eseményt.

- **Értesítések:** Megkapja az értesítéseket azokról az eseményekről, amelyek érintik a saját feladatait vagy fiókját, például új feladatok létrehozása vagy módosítása.

11. Tesztelésijegyzőkönyv

1. Felhasználók kezelése (users route):

- Ellenőrzés, hogy sikeresen létrehozható-e új felhasználó a POST /users/create végponton keresztül.
- Ellenőrzés, hogy a jelszavak biztonságosan titkosításra kerülnek-e bcrypt segítségével.
- Ellenőrzés, hogy az autentikáció JWT tokennel működik-e a bejelentkezés és a kijelentkezés során.

2. Feladatok kezelése (task route):

- Ellenőrzés, hogy sikeresen létrehozhatók, módosíthatók és törölhetők-e feladatok a megfelelő HTTP műveletekkel (POST, PUT, DELETE).
- Ellenőrzés, hogy az értesítések automatikusan generálódnak-e a feladatokhoz kapcsolódóan.

3. Értesítések kezelése (notification route):

- Ellenőrzés, hogy az értesítések megfelelően tárolódnak-e és jelennek meg a felhasználók számára.
- Ellenőrzés, hogy az értesítések helyesen informálnak-e a felhasználók tevékenységeiről.

4. Felhasználói műveletek és funkciók (getUserTask, getUser stb.):

- Ellenőrzés, hogy a különböző felhasználói műveletek és funkciók megfelelően működnek-e az adott jogosultságok figyelembevételével.
- Ellenőrzés, hogy a jogosultságok és hozzáférési szintek korlátozása megfelelően működik-e.

5. Általános funkcionalitás és hibakezelés:

- Ellenőrzés, hogy a kódok jól strukturáltak és modulárisak-e, ami lehetővé teszi az új funkciók hozzáadását és a meglévő funkciók módosítását.
- Ellenőrzés, hogy a kivételkezelés megfelelően van-e kezelve, biztosítva az alkalmazás stabilitását és biztonságát.

	Tesztelési Jegyzőkönyv :							
	Teszt	**Bemenet**			**Eredmény**			
	Regisztrációs Űrlap							
	Üres Űrlap Beküldése	-			Hibás mező hibaüzenetek megjelennek			
	Bejelentkezési Űrlap							
	Érvénytelen Email Formátum	v1210@v1234			Érvénytelen email formátum hibaüzenet			
	Készítette: Horváth Márk				Kelt: 2024.04.08			

12. Összefoglalás

Az "Irodai Naptár" egy modern és innovatív webes alkalmazás, amelyet Michtits Patrik és Horváth Márk fejlesztett ki a cégek számára. A szakdolgozat célja az volt, hogy egy olyan szervezőplatformot tervezzenek és valósítsanak meg, amely segíti a vállalatokat hatékonyabban tervezni és koordinálni a munkafolyamatokat. Az alkalmazás felhasználóinak lehetőséget nyújt arra, hogy könnyedén nyomon kövessék a teendőiket, határidőket és eseményeket a mindennapi munkavégzés során. A szakdolgozat a fejlesztés folyamatát dokumentálja, és bemutatja az alkalmazás főbb jellemzőit, technológiai háttérét és jövőbeli fejlesztési lehetőségeit. Főbb Jellemzők: Az "Irodai Naptár" számos hasznos funkciót kínál a felhasználók számára a hatékony időgazdálkodás és munkaszervezés érdekében. Ezek közé tartoznak: Naptár Megjelenítése: Az alkalmazás egy áttekinthető naptárat kínál, amelyen láthatók a heti és napi tevékenységek. Feladatok Kezelése: A felhasználók könnyedén hozzáadhatnak, szerkeszthetnek és törölhetnek feladatokat a naptárban. Események és Megbeszélések: Lehetőség van események és megbeszélések létrehozására a naptárban, valamint azok részletes információinak kezelésére. Csapatmunka és Megosztás: Funkciók állnak rendelkezésre a feladatok és események megosztására a csapattagokkal, valamint közös naptárak létrehozására és kezelésére. Értesítések és Emlékeztetők: Az alkalmazás automatikusan értesítéseket küld a fontos határidőkről és eseményekről, valamint lehetőség van emlékeztetők beállítására. Technológiai Háttér: Az "Irodai Naptár" frontendjének fejlesztéséhez a következő technológiákat használták: React: A React keretrendszer segítségével dinamikus és felhasználóbarát felhasználói felületet terveztek. Material-UI (MUI) komponensek: A Material-UI komponensek gyors és esztétikus felhasználói felületet biztosítottak. Tailwind CSS: A Tailwind CSS segítségével testre szabható stílusokat és elrendezéseket készítettek. A backend részt a Prisma ORM segítségével valósították meg, amely hatékony adatbázis kezelést biztosított számukra. Jövőbeli Fejlesztési Lehetőségek: Az "Irodai Naptár" jelenlegi verziója már számos hasznos funkciót kínál, azonban további fejlesztési lehetőségeket is felismernek a fejlesztők:
Mobilalkalmazás Fejlesztése: Egy különálló mobilalkalmazás létrehozása iOS és Android platformokra, hogy a felhasználók bármikor és bárhol hozzáférhessenek az irodai naptárhoz.

Továbbfejlesztett Csapatmunka Funkciók: Kibővített csapatmunka funkciók, például közös feladatokhoz való hozzászólási lehetőség, kommentek, és valós idejű közösségi üzenetküldés.

Integráció Más Platformokkal: Integráció más népszerű platformokkal, mint például Google Naptár vagy Microsoft Outlook, hogy a felhasználók könnyebben szinkronizálhassák az eseményeket.

Analitikai Eszközök: Adatok gyűjtése és elemzése az alkalmazás használatáról, hogy segítsen a felhasználóknak még hatékonyabban tervezni és időt takarítani.

Az "Irodai Naptár" egy átfogó és hatékony megoldást kínál a vállalati időgazdálkodás és munkaszervezés területén. Michtits Patrik és Horváth Márk által fejlesztett alkalmazás modern technológiákat alkalmaz, és további fejlesztésekkel még inkább optimalizálható a felhasználók igényeinek megfelelően.

13. Irodalomjegyzék

Abishek Vats: https://stackoverflow.com/questions/69479595/property-id-does-not-exist-on-type-string-jwtpayload-property-id-does

Umar Abdullah: https://stackoverflow.com/questions/77076798/error-while-creating-prismaclient-instance-in-express-js

Kevin Jafet Morán Oroczó: https://stackoverflow.com/questions/70428026/how-can-i-receive-a-json-in-express-server

Jay Kim : https://stackoverflow.com/questions/64566405/react-router-dom-v6-usenavigate-passing-value-to-another-component

Christos Iraklis Tsatsoulis: https://stackoverflow.com/questions/69708504/module-not-found-cant-resolve-mui-icons-material-filedownload