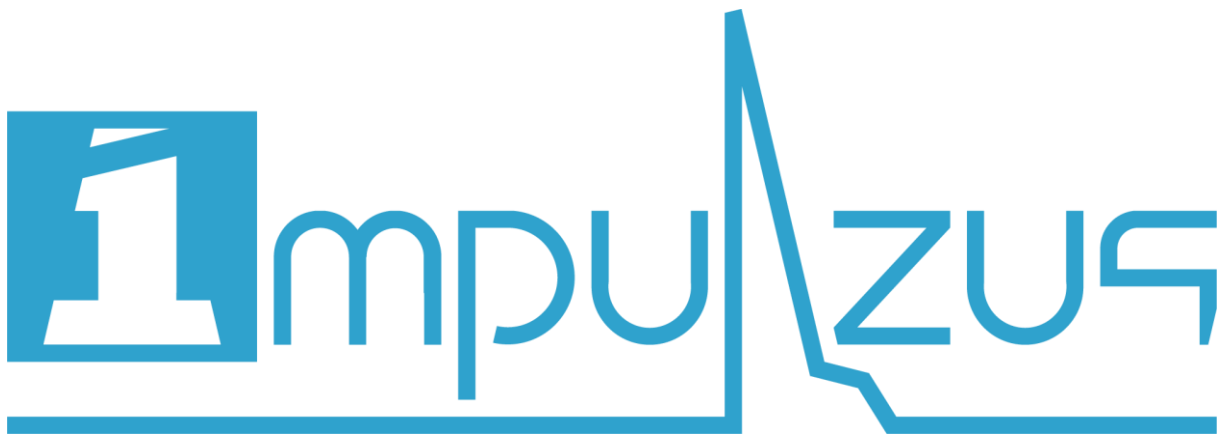


Önálló laboratórium

Száraz Dániel

GT5X34

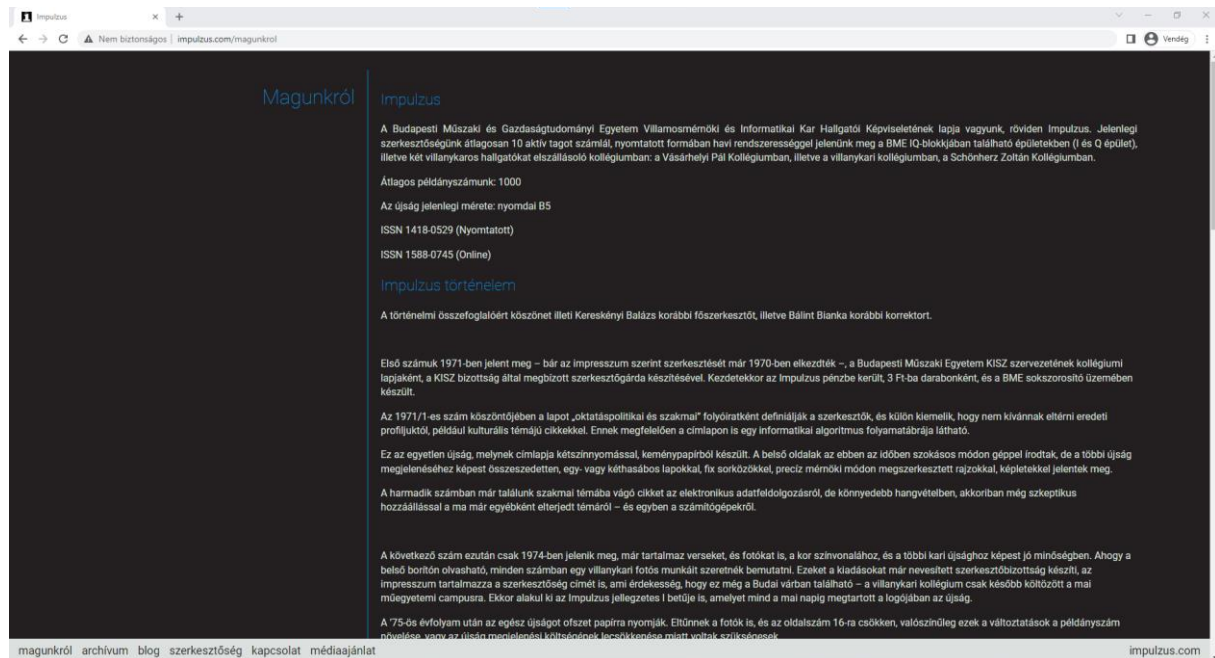


avagy a NextJS felfedezése

2023 tavaszi félév

1. Feladat kiválasztása

Az Impulzus a BME VIK Hallgatói Képviselőtétel lapja. A weboldaluk már kicsit elavult volt, így az ő felkérésükre készítettem nekik egy új weboldalt.



Az Impulzus régi weboldala

2. Specifikáció

A specifikáció alapjait természetesen az Impulzus kör készítette el, a felkérésben már szerepelt a megvalósítandó funkciók nagy része. Ezen kívül tartottunk egy közös megbeszélést, ahol átbeszéltük a specifikációban leírtakat, majd pár dologgal ki is egészítettük azt.

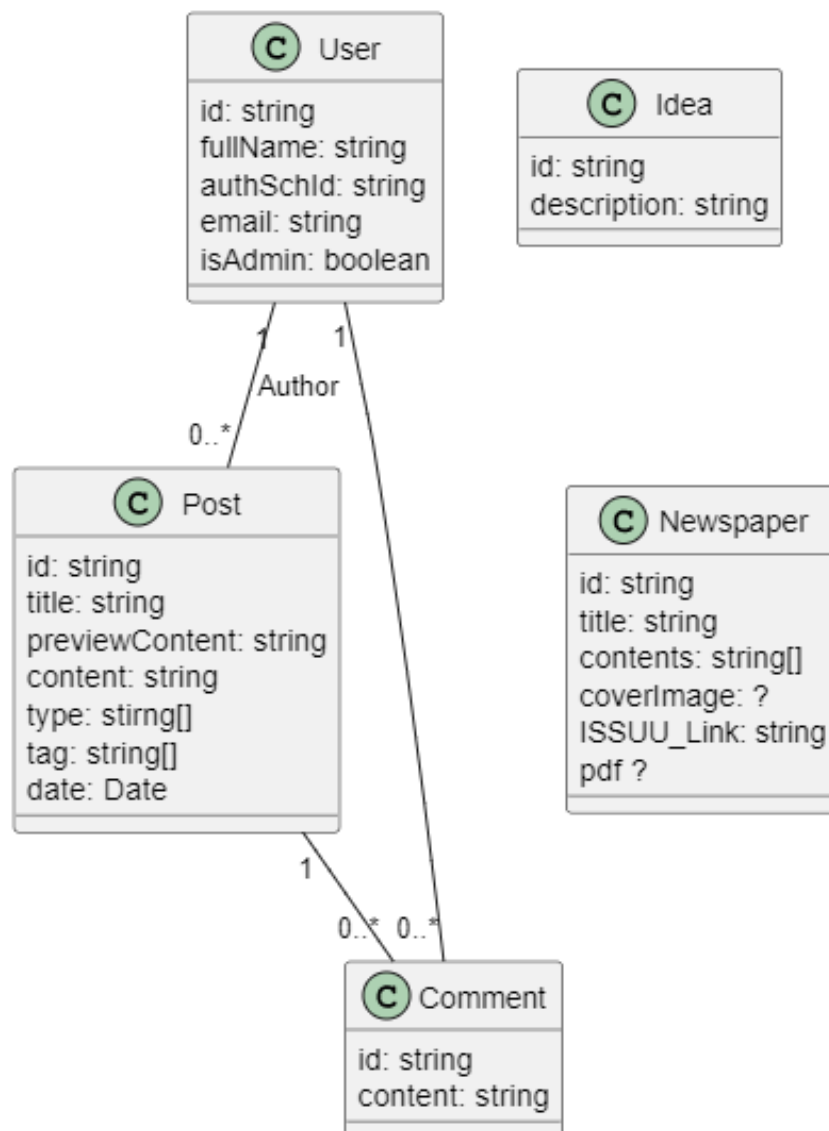
Így a főbb követelmények:

- „Rólunk” oldal, amin látható az Impulzus kör története
- „Archívum” oldal, ahol láthatóak a korábbi Impulzus cikkek
- „Ötletdoboz” oldal, ahol bárki írhat ötleteket a körnek
- „Szerkesztőség” oldal, ahol láthatók a mostani tagok és róluk némi adat
- Blog, ahova lehet posztokat kirakni, és a posztokra lehet kommentet írni
- AuthSch bejelentkezés

3. Tervezés

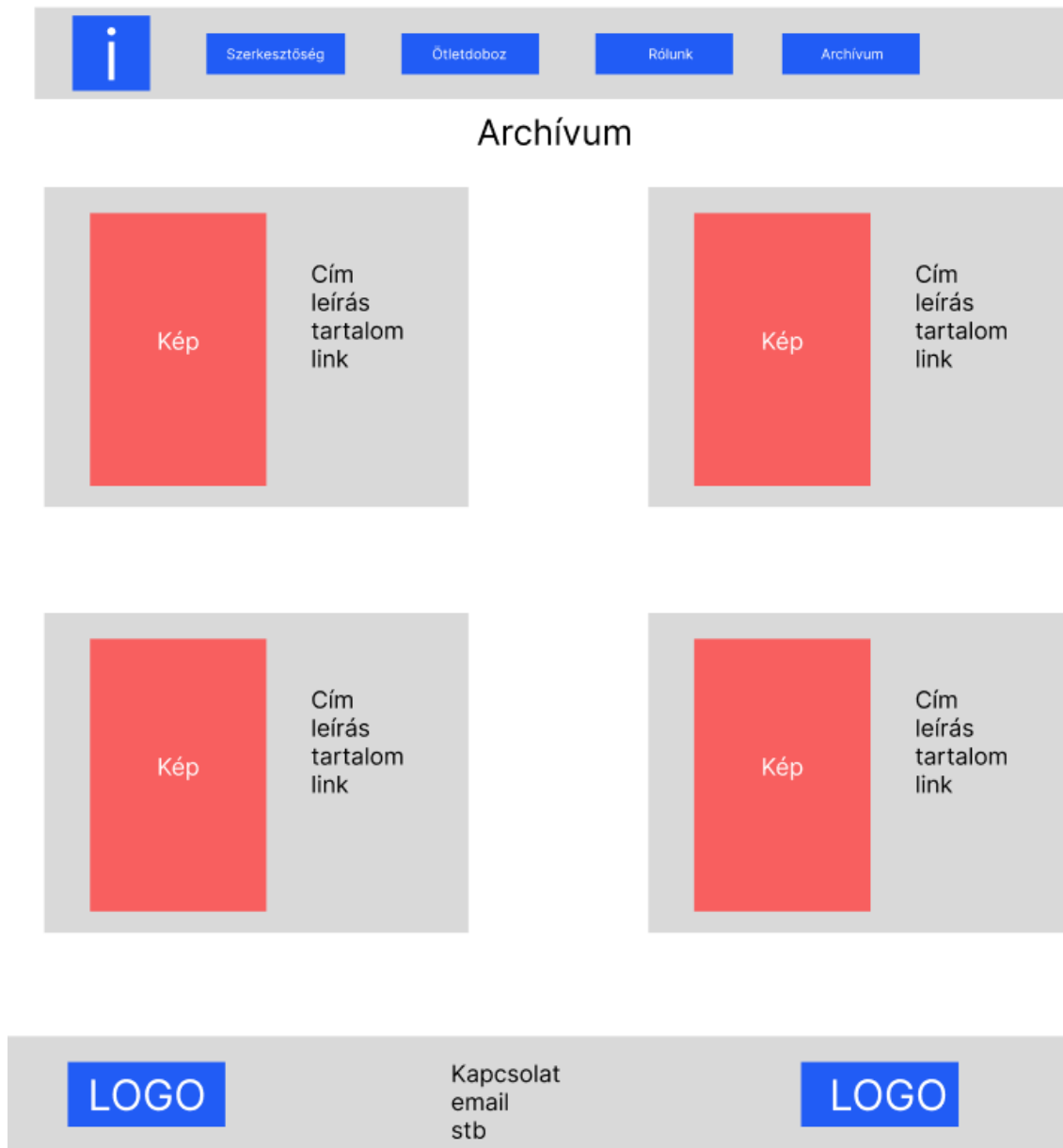
A tervezés első lépése az adatbázis séma megírása volt. Szükség volt felhasználókra, blog posztokra, és a posztokhoz kommentekre. Ezen kívül kellett csinálnom egy ötletdobozt, ahova ötleteket lehet írni, és újságcikkeket is kezelni kell az oldalon. A séma alapjai jól sikerültek, a fejlesztés során csak pár kisebb

módosítás, kiegészítés kellett. Ezen felül volt még egy nagyobb módosítás, az AuthSch bejelentkezés megoldásánál, de ez szinte csak a bejelentkezést érintette.



A kezdetleges adatbázis séma

A tervezés egy másik fontos része a frontend megtervezése volt. Elég nagy szabad kezet kaptam a dizájnt illetően, úgyhogy igyekeztem egy letisztult, könnyen átlátható felületet létrehozni. Mivel nem volt nagyon bonyolultan kinéző oldal, és mert nem dizájnolással szerettem volna eltölteni az időm nagy részét, ezért sok mindent a frontend fejlesztése alatt találtam ki, hogy hogy legyen. Kezdetlegesen csináltam egy Figma tervet is, ami az archívumok oldalt ábrázolja.



Az archívum oldal kezdetleges Figma terve

4. A NextJS

Nem először fejlesztettem node-os alkalmazást, de eddig főleg NestJS-ben volt tapasztalatom. Ennél a projektnél viszont szerettem volna valami újat kipróbálni. Ezen felül mindenképp React-ot szerettem volna frontenden használni, így esett a választás az egyik modern és feltörekvő keretrendszerre, a NextJS-re.



A Next.js egy reaktív JavaScript keretrendszer, amelyet kifejezetten webalkalmazások és webhelyek építésére terveztek. Célja, hogy megkönnyítse a fejlesztők számára alkalmazások létrehozását a weben, és kiterjessze a React keretrendszer alapfunkcióit.

A Next.js, a React-ot a szerveroldali és a kliensoldali rendszerek egyesítésére használja. Ez lehetővé teszi a szerveroldali generálást (Server-Side Rendering - SSR), az előre generált statikus weboldalakat (Static Site Generation - SSG) és a kliensoldali generálást (Client-Side Rendering - CSR) is. Én a legtöbb helyen szerveroldali generálást használtam.

A Next.js fő jellemzői közé tartozik a beépített útválasztás támogatása, a CSS stílusok kezelése, a dinamikus adatbetöltés, a tesztelési és teljesítményoptimalizálási lehetőségek, valamint a TypeScript támogatása. A keretrendszer rugalmasságot és skálázhatóságot kínál, ami lehetővé teszi a fejlesztők számára, hogy nagyobb, összetett webalkalmazásokat építsenek.

A Next egyik szembetűnő tulajdonsága, hogy a route-ok és endpoint-ok a mappaszerkezettől függenek. Egy adott mappában található index fájl hivatkozik az adott mappára, és más névvel elnevezett fájlok pedig a mappa elérési útvonala + a fájl neve.

Vannak dinamikus útvonalak, ami azt jelenti, hogy egy adott útvonalat tudunk több urlhez is rendelni.

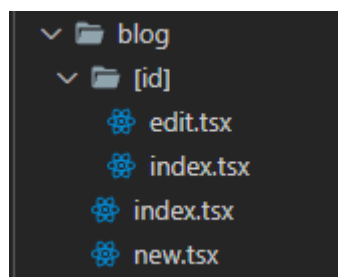
Route	Example URL	params
<code>pages/blog/[slug].js</code>	<code>/blog/a</code>	<code>{ slug: 'a' }</code>
<code>pages/blog/[slug].js</code>	<code>/blog/b</code>	<code>{ slug: 'b' }</code>

Ennél egy fokkal erősebb a Catch-all Segments, amit „...név”-el jelölünk. Ez annyival tér el az előzőtől, hogy minden urlt, aminek egy bizonyos kezdete van, azt az adott útvonalhoz fogja rendelni.

Route	Example URL	params
<code>pages/shop/[...slug].js</code>	<code>/shop/a</code>	<code>{ slug: ['a'] }</code>
<code>pages/shop/[...slug].js</code>	<code>/shop/a/b</code>	<code>{ slug: ['a', 'b'] }</code>

Saját példa mappaszerkezetről, elérhető útvonalak:

- `/blog`
- `/blog/new`
- `/blog/[id]`
- `/blog/[id]/edit`



5. Backend

Adatbázis:

Adatbázisnak a PostgreSQL-t választottam, mivel már dolgoztam vele, és mert mindenképp relációs adatbázist szerettem volna. Ez azért fontos, mert sok kapcsolat van az entitások között (post, user, comment stb.), és mert a PostgreSQL egy népszerű adatbázis, amit a NextJS is támogat.



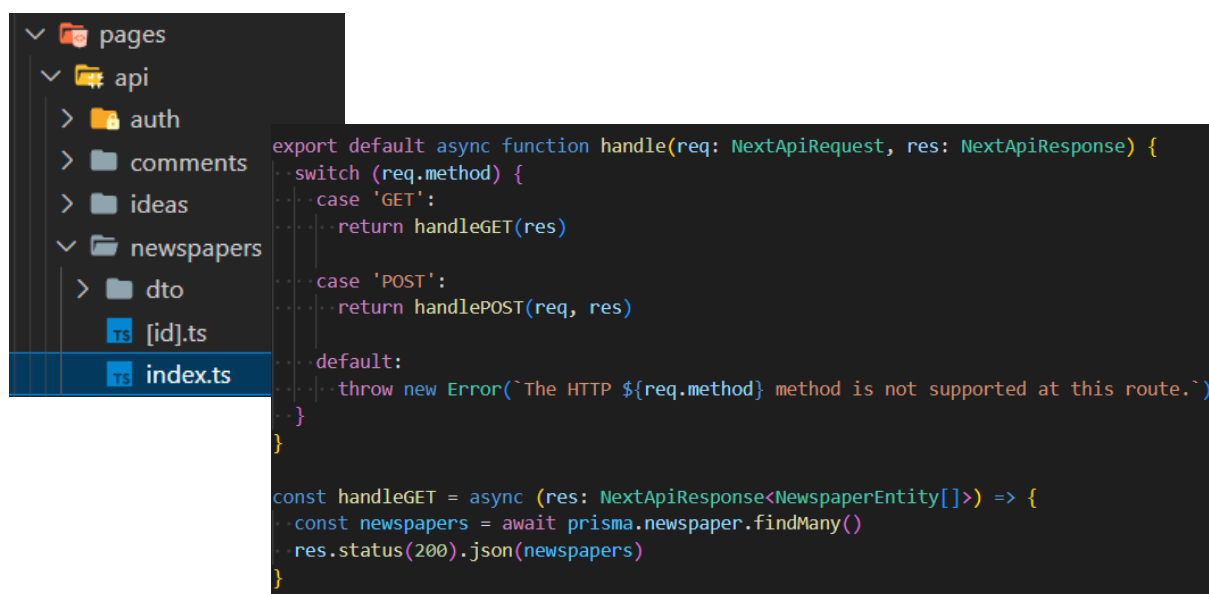
Prisma:

A Prisma egy Node.js és TypeScript ORM (Object-Relational Mapping). Ez nagyban megkönnyíti és biztonságossá is teszi az adatbázissal való kommunikációt. Természetesen támogatja a PostgreSQL-t is. Egy schema.prisma fájlba kell írunk a típusos adatbázis sémát és az adatbázis elérést, és egy parancs futtatása után már fel is kerül a séma az adatbázisba. Az adatbázis elérése ugyanennyire egyszerű, példányosítani kell egy PrismaClient-et, amin keresztül elérhető az adatbázis, és szinte minden művelet előre van definiálva. A PrismaClient-et én globálisan definiáltam, így bárholnan egyből el tudom érni az adatbázist.



Api:

Mivel a Next.js a szerveroldalt és a kliensoldalt is egyben megvalósítja, ezért csupán mappák választják el egymástól az alkalmazás különböző részeit. Next-en belül a pages mappába bekerülő fájlok és almappák elérési útvjai képezik az elérhető útvonalakat. Az API részt tehát a pages mappán belül egy api mappába helyeztem el. Egy példa: ha szeretném az összes újságcikket lekérni, akkor egy GET kérést küldök a /api/newspapers végpontra. Az ehhez tartozó kód és mappa szerkezet a következő:



Autentikáció:

Az oldalra AuthSch-val lehet bejelentkezni. Ennek a megvalósítását a NextAuth.js-el csináltam. Ez egy viszonylag új könyvtár, ami Next.js-hez készült, autentikációhoz. Vannak benne beépített provider-ek, mint például Google vagy GitHub, de támogatja a saját provider elkészítését is. Sajnos a dokumentációja még nem tökéletes, így pár dologra nehéz volt rájönni, de sikerült működésre bírni, így már be lehet jelentkezni AuthSch-n keresztül. Ehhez szükséges volt kibővíteni az adatbázis sémát előre definiált Account és Session-el, és a User-t is hozzájuk kellett kötnöm.



A NextAuth biztosít egy `useSession()` hook-ot, mellyel az lekérhető az éppen bejelentkezett felhasználó. Ez hatalmas segítség, például jogosultság ellenőrzésnél. Az egyetlen probléma az volt vele, hogy a `useSession` egy olyan felhasználót ad vissza, amin csak név, email és kép van. Nekem ennél több mindere volt szükségem, így ki kellett valahogy terjesztenem ezt a felhasználót az éppen bejelentkezett tényleges felhasználó minden adatával. Ezt úgy sikerült megoldanom, hogy a bejelentkezési folyamat közben, amikor már létrejött egy session és egy user, lekérem az adatbázisból azonosító alapján a teljes felhasználót, majd ezt belerakom a session felhasználójába.

A NextAuth támogat különböző adaptereket is, így a Prisma-t is. Bejelentkezéskor, ha megvan adva egy PrismaAdapter, akkor a prisma automatikusan kommunikál az adatbázissal.

6. Frontend

React:



React egy nyílt forráskódú JavaScript könyvtár, amelyet a felhasználói felületek fejlesztésére használnak. Komponensalapú megközelítést alkalmaz, amely lehetővé teszi, hogy az alkalmazás elemeire önálló komponensekként gondoljunk. Ezeket a komponenseket egymásba ágyazhatjuk, és újra fel is használhatjuk, így egyszerűen és hatékonyan kezelhetjük a felhasználói felületeket.

Nem először dolgozok React-ban, nagyon kényelmes és szeretem használni, így nem volt kérdés, hogy ezt választom-e.

ChakraUI:



A Chakra UI alapvetően egy UI komponenskönyvtár, amely előre elkészített, újrafelhasználható építőelemeket kínál, például gombokat, űrlapelemeket, navigációs sávokat, kártyákat és sok mást. Ezeket a komponenseket könnyen testre lehet szabni.

A Chakra UI a stílusok kezelésére a CSS-in-JS (CSS in JavaScript) megközelítést használja. Ez azt jelenti, hogy a stílusokat JavaScript objektumok formájában definiálhatja a komponensek mellett, és a Chakra UI automatikusan alkalmazza őket. Emellett támogatja a tematizálást és a komponensek globális stílusainak testreszabását is.

```
UserGrid.tsx X
src > components > editorship > UserGrid.tsx > ...
You, 2 weeks ago | 1 author (You)
1 import { UserEntity } from '@pages/api/users/dto/UserEntity.dto'
2 import { GridItem, SimpleGrid } from '@chakra-ui/react'
3 import { UserCard } from './UserCard'
4
5 type Props = {
6   users: UserEntity[]
7 }
8
9 export const UserGrid = ({ users }: Props) => {
10   return (
11     <SimpleGrid spacing={10} columns={{ base: 1, xl: 2 }}>
12       {users.map((u) => (
13         <GridItem borderWidth={1} borderRadius={5} p={2} key={u.id}>
14           <UserCard key={u.id} user={u} />
15         </GridItem>
16       ))}
17     </SimpleGrid>
18   )
19 }
20
```

Egy egyszerű saját React komponens, ChakraUI elemek felhasználásával

A fejlesztés során sok más React könyvtárat is felhasználtam, amik nagyban megkönnyítették a munkámat. Néhány fontosabb: react-hook-form, react-markdown és a react-pdf.

A react-hook-form egy nagyon jó formkezelő könyvtár, mellyel nagyon egyszerű form-on felhasználói adatokat ellenőrizni és validálni.

A react-markdown-t egy Markdown szerkesztőhöz használtam, amiben meg lehet adni egy markdown leírást, majd meg lehet nézni, hogy hogy is fog kinézni.

A react-pdf-et pedig az Impulzus cikkek megjelenítésére használtam fel.

Új tag



Név *

Teszt Elek

A név nem lehet üres!

Email *

elek@gmail.com

Az email nem lehet üres!

Posztok *

Író, grafikus

Legalább 1 poszt kell!

Profilkép

https://image

☐ Vezetőségi tag

Mégse

Mentés

Egy react-hook-form-al létrehozott form

Leírás *

Szerkesztés

Előnézet

Markdown útmutató itt.

Typographic replacements

Enable typographer option to see result.

"Smartyants, double quotes" and 'single quotes'

Emphasis

****This is bold text****

__This is bold text__

This is italic text

__This is italic text__

~~Strikethrough~~

Leírás *

Szerkesztés

Előnézet

Typographic replacements

Enable typographer option to see result.

"Smartyants, double quotes" and 'single quotes'

Emphasis

This is bold text

This is bold text

This is italic text

This is italic text

~~Strikethrough~~

Vissza

Markdown szerkesztő



Pdf olvasó

7. Fejlesztő környezet

Fejlesztéshez VS Code-ot használtam. Nagyon kényelmes és szinte mindenhez van extension. A kód formázásához és egységességéhez ESLint-et és Prettier-t használtam. A Projektben belül a .vscode mappában egy settings.json fájlban lehet megadni formázási beállításokat, és mentésre formázást is. Ez nagyon sok időt megspórolt a munkában.

Egyéb eszközök, amiket használtam:

- GitKraken – verziókezelés
- Postman – api tesztelés
- Prisma Studio – a Prisma saját adatbázis kezelő felülete

8. Nehézségek


A félév során több nehézséggel is szembefutottam. Az első nehezebb feladat az AuthSch bejelentkezés volt, ezt nagyjából sikerült megoldani, de még nem mondanám tökéletesnek. A másik nehézséget az újságcikk olvasó megalkotása jelentette. Kezdetben sok másik könyvtárat is kipróbáltam, de mindegyikkel volt valami baj. Végül a legegyszerűbb használata mellett döntöttem, ami működött is.

Ezekon kívül volt még egy probléma, ami végigkísért a félév során: a kommunikáció megrendelővel. Sajnos sok mindent nem kaptam meg, amik meg voltak beszélve, és inkább rábíztak sok mindent. Szerencsére ez nem akadályozott meg a munkámban, hiszen nem én vagyok a felelős a tartalomért.

9. Végeredmény

Az oldalon meg lehet tekinteni a szerkesztőség tagjait, a kör történetét, olvasni lehet a korábbi impulzus cikkeket és korábbi blogposztokat. AuthSch bejelentkezés után lehet blogposztot írni, a posztokhoz lehet kommentelni, és lehet ötleteket feladni. Aki admin jogosultsággal rendelkezik, ő mindent tud csinálni az oldalon, bármilyen tartalmat módjában áll létrehozni, szerkeszteni és törölni is.

[A projekt github oldala elérhető itt.](#)



RólunkBlogArchívumSzerkesztőségÖtletdobozProfil

Magunkról

Impulzus

A Budapesti Műszaki és Gazdaságtudományi Egyetem Villamosmérnöki és Informatikai Kar Hallgatói Képviselőtársaság lapja vagyunk, röviden Impulzus. Jelenlegi szerkesztőségünk átlagosan 10 aktív tagot számlál, nyomtatott formában havi rendszerességgel jelenünk meg a BME IQ-blokkjában található épületekben (I és Q épület), illetve két villanykaros hallgatókat elszállásoló kollégiumban: a Vásárhelyi Pál Kollégiumban, illetve a villanykari kollégiumban, a Schönherz Zoltán Kollégiumban.

Átlagos példányszámunk: 1000

Az újság jelenlegi mérete: nyomdai B5

ISSN 1418-0529 (Nyomtatott)

ISSN 1588-0745 (Online)


Impulzus történelem

A történelmi összefoglalóért köszönet illeti Kereskényi Balázs korábbi főszerkesztőt, illetve Bálint Biana korábbi korrektort.

Első számuk 1971-ben jelent meg - bár az impresszum szerint szerkesztését már 1970-ben elkezdték -, a Budapesti Műszaki Egyetem KISZ szervezetének kollégiumi lapjaként, a KISZ bizottság által megbízott szerkesztőgárda készítésével. Kezdetekkor az Impulzus pénzbe került, 3 Ft-ba darabonként, és a BME sokszorosító üzemében készült.

Az 1971/1-es szám közlőjében a lapot „oktatáspolitikai és szakmai” folyóiratként definiálják a szerkesztők, és külön kiemelik, hogy nem kívánnak eltérni eredeti profiljuktól, például kulturális témákkal. Ennek megfelelően a címlapon is egy informatikai algoritmus folyamatábrája látható.

Ez az egyetlen újság, melynek címlapja kétszínnyomással, keménypapírból készült. A belső oldalak az ebben az időben szokásos módon géppel íródtak, de a többi újság megjelenéséhez képest összeszedetten, egy- vagy kéthasábos lapokkal, fix sorközökkel, precíz mérnöki módon megszerkesztett rajzokkal, képekkel jelentek



RólunkBlogArchívumSzerkesztőségÖtletdobozProfil

Blog

Új poszt

Tudományos Diákköri Konferencia

Gulyás Gergely Zoltán - 2023. 05. 22.

“Ma, november 16-án, kedden oktatási szünet van. Ezt mindenki tudja, és talán még azt is, hogy a TDK miatt, de mi is az a TDK?”

EgyetemSzakmaKultúraKözeletTechRetróEgyébGaming

Pénzbírságra számíthatnak a hallgatók a vészhelyzet előtt kivett könyvekért

Gulyás Gergely Zoltán - 2023. 05. 22.

“Pénzbírságot kapnak a hallgatók a vészhelyzet előtt kivett könyvekre.”

EgyetemKözelet

Csőtörés

Gulyás Gergely Zoltán - 2023. 05. 22.

“Szünetel a hidegvíz-szolgáltatás a Schönherz Kollégiumban, mivel egy karbantartási munkálat közben vízvívárgás történt a harmadik emeleten.”

KözeletEgyéb

Teszt

Szárz Dániel - 2023. 05. 24.

“Teszt”

Egyéb

11

Ötletdoboz

[Új ötlet](#)


Jó lenne egy cikk a Schönherzben élő denevérekről.



Örölnék neki ha írnátok a Q épületet megépítéséről, nagyon izgalmasnak találok a témát, és szeretnék róla egy Impulzus cikkben is olvasni ^^



Vezetőség

Gulyás Gergely Zoltán módosítása

Név *
Gulyás Gergely Zoltán

Email *
gulyas.gergely@impulzus.bme.hu

Posztok *
Főszerkesztő, tördelő, író

Profilkép
https://warp.sch.bme.hu/images/yf9oq5v4rw

☒ Vezetőségi tag

Mégse **Mentés**

Archívum

Évfolyam: [←](#) 49 [→](#)

[Új cikk](#)


XLIX. évfolyam, 3. szám

Tartalomjegyzék:

- A végtelenbe és tovább!
- Simonyi Konferencia az úrkutatásról - Kvantumtechnológia a vodafone-nál
- Határ a csillagos ég
- Mesterséges intelligencia az életminőség javításának szolgálatában
- Bejutsz és talán kijutsz
- Pleisztocén Hagyaték: Az Evolúciós Kontinuum



L. évfolyam, 2. szám

Tartalomjegyzék:

- Már a huszadik?
- A MEMS
- a világmindenség
- Ugródeszka a Schönherzből a saját céged vezetéséhez
- A Konferencia támogatói
- A MESZ-I titkai Világ történései 2022-ben
- Információs Ellenforradalom





10. Publikálás



A publikálás Vercel-en keresztül történt. A Vercel egy felhőalapú szolgáltatás, amely lehetővé teszi a webalkalmazások és webhelyek egyszerű fejlesztését, telepítését és üzemeltetését. A Vercel elsősorban a statikus webhelyek és a frontend alkalmazások számára kínál hosting és fejlesztési infrastruktúrát. Támogatva van a Next.js is és jelenleg ugyan még csak béta verzióban, de lehet létrehozni saját PostgreSQL adatbázist Vercel-en belül. Szerencsére nagyon egyszerű volt felállítani, a Vercel PostgreSQL generál környezeti változókat is a használatához.

A Vercel össze van kötve GitHub-bal. Jelenleg az van beállítva, hogy a master branch-et figyeli a Vercel, és amint pusholok valamit már teszi is ki élőbe.

[A weboldal elérhető itt.](#)

11. További fejlesztés

Ugyan az oldal nagy része elkészült és használható, vannak dolgok, amikkel ki szeretném egészíteni, és van pár dolog, amin szépíteni és javítani tervezek. Ilyenek például: fájlfeltöltés, látványosabb hibakezelés, stb.. Ezen felül remélem a megrendelővel is sikerül még egy beszélgetést folytatnom, ahol meg tudjuk beszélni, hogy megfelel-e neki az oldal, és miket szeretnének még.