

BGSzC Pestszentlőrinci Közgazdasági és Informatikai Szakgimnáziuma

1184 Budapest Hengersor 34.

Záró dolgozat

Videoderify

Konzulens tanár:

Bencze István

Készítette:

Nyirfa Balázs

Tartalom

1	Bevezetés.....	2
1.1	Feladatleírás	2
1.2	A felhasznált ismeretek	2
1.3	A felhasznált szoftverek.....	2
2	Felhasználói dokumentáció.....	4
2.1	A program általános specifikációja.....	4
2.2	Rendszerkövetelmények	4
2.2.1	Hardver követelmények	4
2.2.2	Szoftver követelmények.....	4
2.3	A program telepítése (GNU+Linux)	5
-	2.3.2 A program telepítése (Windows)	18
2.4	A program használatának a részletes leírása	20
3	Fejlesztői dokumentáció.....	25
3.1	Az alkalmazott fejlesztői eszközök	25
3.2	Adatmodell Adatmodell	27
3.3	Részletes feladatspecifikáció, algoritmusok.....	28
3.4	Tesztelési dokumentáció	34
4	Összefoglaló.....	36
-	4.1 Továbbfejlesztési lehetőségek	36
-	4.2 Önértékelés	36
5	Felhasznált irodalom	38
6	Ábrajegyzék:	41

1 Bevezetés

1.1 Feladatleírás

Egy olyan applikációra volt szüksége a baráti körömnek, mellyel összegyűjtött média tartalmainkat egy biztonságos és gyors szerverről elérhetővé tehetjük magunk közt, vagy bárkinek. Fő szempont volt, hogy egyrészt az adatok rendezettek legyenek, másrészt, hogy azokat ne kelljen minden alkalommal letöltve elmenteni a felhasználó saját eszközére, hanem akár magán a tároló felületen egyből azt kiválasztva meg is lehessen tekinteni. Tehát lényegében, egy privát, saját szerveren futó (self-hosted) videó megosztó applikációra volt szükségünk.

1.2 A felhasznált ismeretek

- GitHub: A Git segítségével szoftverfejlesztési verziókövetés-szolgáltatást nyújt, egy közösségi hálózat formájában, ahol bárki meg tudja osztani a szoftverének a forráskódját publikusan, melyeket license-től függően fel is lehet használni saját munkánkba. Könnyű olyan applikációkat találni rajta, melyekből számunkra új dolgokat lehet tanulni.
- Fejlesztői dokumentációk: Egy felhasználható fejlesztő eszköznek, könyvtárnak, API-nak stb... manapság már jól dokumentált részletes leírásai vannak, melyekből könnyen neki láthatunk az adott eszközzel dolgozni, illetve gyakori, hogy egy gyakori problémára is választ ad. Ilyen például a ReactJS, vagy a Bootstrap saját dokumentációja.
- Oktató jellegű videók: Legfőképpen a YouTube-on az utóbbi pár évben elterjedtek a szoftverfejlesztés oktatásával foglalkozó tartalmak, melyek ingyenesen megtekinthetőek és friss információkat tartalmaznak, az elmagyarázott témában pedig sok esetben csatolnak letölthető példákat is.

1.3 A felhasznált szoftverek

- Figma: Egy web alapú grafikus szerkesztő program, melyet főleg felhasználó nézetek tervezésére lehet használni, ingyenesen elérhető a kezdő verzió.
- Chrome: Modern böngésző.
- Google Docs: Ingyenes felhő alapú szövegszerkesztő program.

- One Commander: Modern, ingyenes fájlkezelő applikáció rengeteg hasznos funkcióval, amiket egy OS-en található alapértelmezett fájlkezelő nem tartalmaz.
- Snipping Tool: Windows 10-ben alapértelmezetten megtalálható képernyőkép készítésére alkalmas applikáció.

2 Felhasználói dokumentáció

2.1 A program általános specifikációja

A videoderify egy olyan web applikáció, ahová admin jogosultsággal rendelkező felhasználók média tartalmakat tölthetnek fel, melyek sorozat formájában jelennek meg a felhasználó felületen. Ez azt jelenti, hogy egy ilyen feltöltött objektumnak van neve, leírása, egy “thumbnail” képe és a hozzá tartozó média fájlok, melyek a kezelőfelületen számozott kiválasztható epizódokként jelennek meg. Egy adott sorozathoz tartozhatnak kommentek is, melyeket sima felhasználó és admin is létrehozhat. Kommentet törölni sima felhasználó nem tud a sajátján kívül, viszont egy admin felhasználó bárkiét ki törölheti. Egy kommentnek van egy üzenet mezője és egy “timestamp”-je. Sima felhasználót bárki létrehozhat, amit vagy kommentelésre lehet használni, vagy az adatbázisban admin jogosultságot lehet neki adni, amivel már tartalmat is tud feltölteni. Csak a tartalmak megtekintéséhez egyáltalán nem szükséges bejelentkezni semmilyen felhasználóval. Az egész applikációt két külön szoftverből áll össze, egy REST API-ból és egy kliens-ből. A cél az volt, hogy az API független legyen a klienstől, így a jövőben akár több egymástól független kliens érheti el ugyanazt a szerveret ugyanazzal az adatbázissal, ugyanazzal a tárhellyel. A web applikáción keresztül feltöltött fájlok azon a webszerveren lesznek tárolva, ahol az API fut.

2.2 Rendszerkövetelmények

2.2.1 Hardver követelmények

- CPU: Intel Pentium 4, vagy újabb
- RAM: 2GB
- Internet kapcsolat
- Böngésző (minimum): Chrome 42, Edge 14, Firefox 40, Safari 10.1, Opera 29

2.2.2 Szoftver követelmények

OS (64 bit)

- Windows 7
- Windows 8
- Windows 8.1
- Windows 10
- Windows 11

- GNU/Linux
- Mac OS El Capitan, vagy újabb
- Android
- IOS

Szükséges szoftverek a futtatáshoz:

- NodeJS és npm (legfrissebb verzió)
- Terminál emulátor
- MongoDB Atlas (felhő alapú)

2.3 A program telepítése (GNU+Linux)

Példában használt Linux distribution: Manjaro 21.2.4

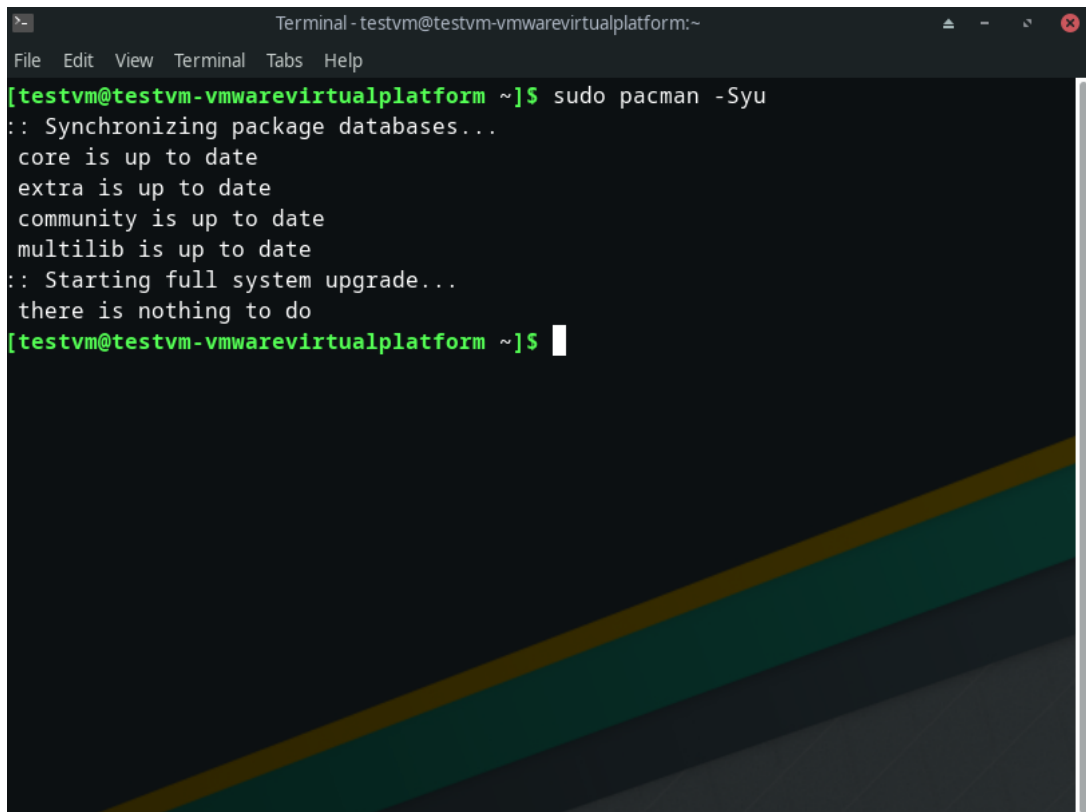
Linux kernel verzió: 5.15.28

Az ábrán látható terminál parancsok egy frissen telepített környezetben (OS, NodeJS, npm, stb...) lettek le futtatva és a sikeres futtatás esetén létrejött kimeneteket mutatja meg

A parancsokat bármilyen tetszőleges, például az OS-en alapértelmezett terminál emulátorban futtassuk

1. OS frissítése:

`sudo pacman -Syu`

A terminal window titled "Terminal - testvm@testvm-vmwarevirtualplatform:~" with a menu bar (File, Edit, View, Terminal, Tabs, Help). The terminal shows the command `sudo pacman -Syu` being executed. The output is: `:: Synchronizing package databases...
core is up to date
extra is up to date
community is up to date
multilib is up to date
:: Starting full system upgrade...
there is nothing to do`. The prompt `[testvm@testvm-vmwarevirtualplatform ~]$` is visible at the bottom.

```
Terminal - testvm@testvm-vmwarevirtualplatform:~
File Edit View Terminal Tabs Help
[testvm@testvm-vmwarevirtualplatform ~]$ sudo pacman -Syu
:: Synchronizing package databases...
core is up to date
extra is up to date
community is up to date
multilib is up to date
:: Starting full system upgrade...
there is nothing to do
[testvm@testvm-vmwarevirtualplatform ~]$
```

OS frissítés

2. NodeJS telepítése:

`sudo pacman -S nodejs npm`

- Ez a parancs telepíti a NodeJS és Npm programok legfrissebb verzióját egyszerre
- Részletes leírás a NodeJS/Npm telepítéséről a NodeJS saját dokumentációjában is található: <https://nodejs.org/en/download/package-manager/>

```
[testvm@testvm-vmwarevirtualplatform ~]$ sudo pacman -S nodejs npm
[sudo] password for testvm:
warning: nodejs-17.7.1-1 is up to date -- reinstalling
resolving dependencies...
looking for conflicting packages...

Packages (5) node-gyp-9.0.0-1  nodejs-nopt-5.0.0-2  semver-7.3.5-2  nodejs-17.7.1-1  npm-8.5.4-1

Total Download Size:    2,37 MiB
Total Installed Size:  44,39 MiB
Net Upgrade Size:       10,56 MiB

:: Proceed with installation? [Y/n] Y
:: Retrieving packages...
  npm-8.5.4-1-any
  node-gyp-9.0.0-1-any
  semver-7.3.5-2-any
  nodejs-nopt-5.0.0-2-any
Total (4/4)
```

NodeJS telepítése

3. Források letöltése

A project legfrissebb verzióját a következő kettő GitHub repository-ból lehet elérni. Ezek tartalmazzák a futtatáshoz szükséges forráskódokat és konfigurációkat.

- API: <https://github.com/anymus0/videoderify-API>

- Kliens: <https://github.com/anymus0/videoderify-react-client>

Ezeknek az ajánlott letöltése a következő parancsok futtatásával lehetséges:

“*git clone* <https://github.com/anymus0/videoderify-react-client.git>” és

“*git clone* <https://github.com/anymus0/videoderify-API.git>”

Ezután az **ls** paranccsal ellenőrizhetjük, hogy valóban sikerült letölteni a kettő darab könyvtárat.


```
→ ~ git clone https://github.com/anymus0/videoderify-react-client.git
Cloning into 'videoderify-react-client'...
remote: Enumerating objects: 310, done.
remote: Counting objects: 100% (310/310), done.
remote: Compressing objects: 100% (207/207), done.
remote: Total 310 (delta 177), reused 222 (delta 93), pack-reused 0
Receiving objects: 100% (310/310), 365.45 KiB | 573.00 KiB/s, done.
Resolving deltas: 100% (177/177), done.
→ ~ git clone https://github.com/anymus0/videoderify-API.git
Cloning into 'videoderify-API'...
remote: Enumerating objects: 616, done.
remote: Counting objects: 100% (616/616), done.
remote: Compressing objects: 100% (384/384), done.
remote: Total 616 (delta 289), reused 448 (delta 152), pack-reused 0
Receiving objects: 100% (616/616), 193.91 KiB | 501.00 KiB/s, done.
Resolving deltas: 100% (289/289), done.
→ ~ ls
gitRepos rebase.js seriesUml.png videoderify-API videoderify-react-client
→ ~
```

Letöltött források ellenőrzése

4. NodeJS/Npm telepítésének ellenőrzése a verziószám megtekintésével:

- node -v

- npm -v

```
>-
File Edit View Terminal Tabs Help
[testvm@testvm-vmwarevirtualplatform ~]$ node -v
v17.7.1
[testvm@testvm-vmwarevirtualplatform ~]$ npm -v
8.5.4
[testvm@testvm-vmwarevirtualplatform ~]$
```

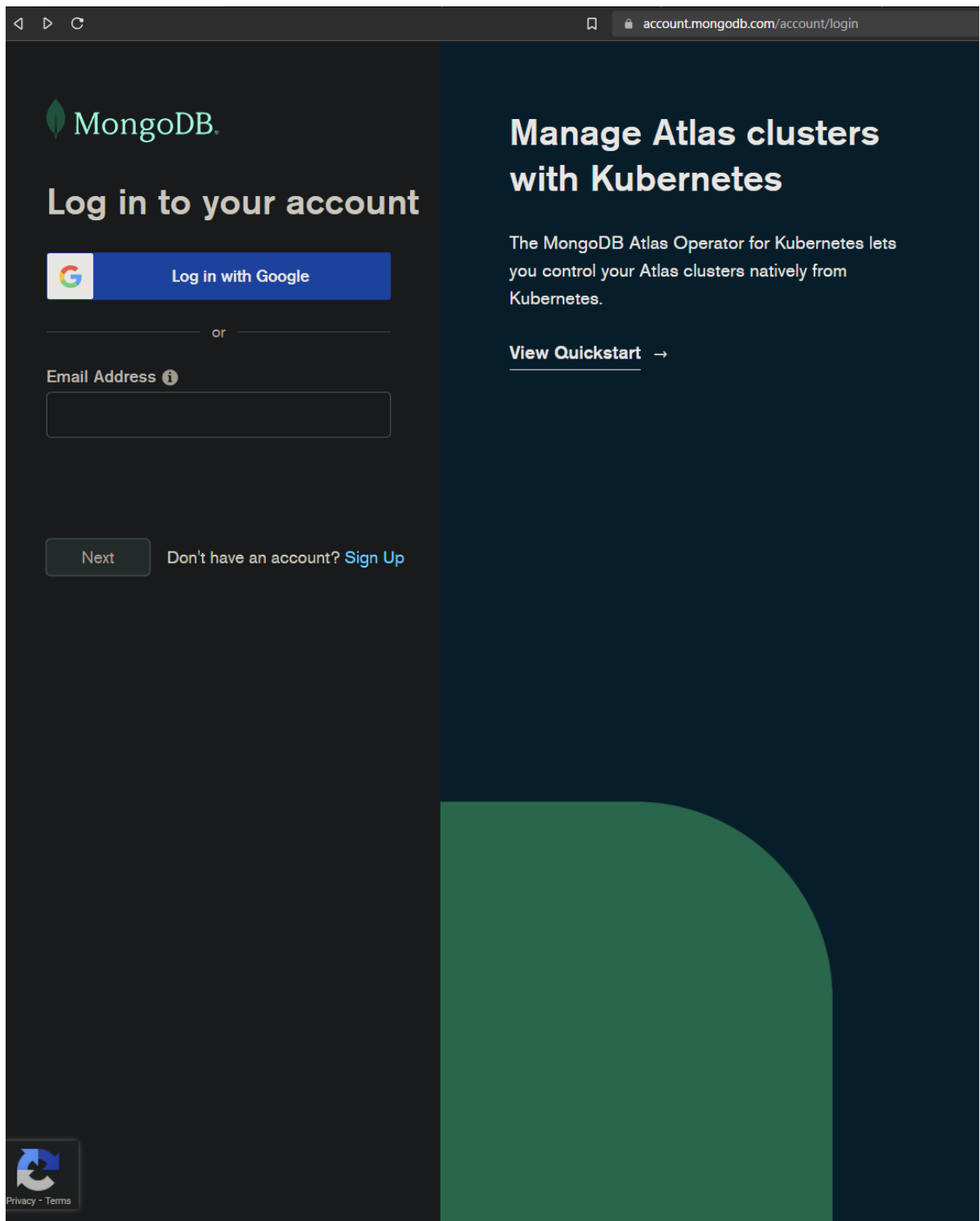
NodeJS verzió ellenőrzése

5. Adatbázis létrehozása

- Mivel a **MongoDB** elérhető felhő alapú szolgáltatásként is, ezért azt nem kell helyileg telepíteni.

- Az ingyenes verzió bőven elegendő egy teszt környezet futtatásához, viszont ha van rá igény, akkor lehet nagyobb kapacitást venni.

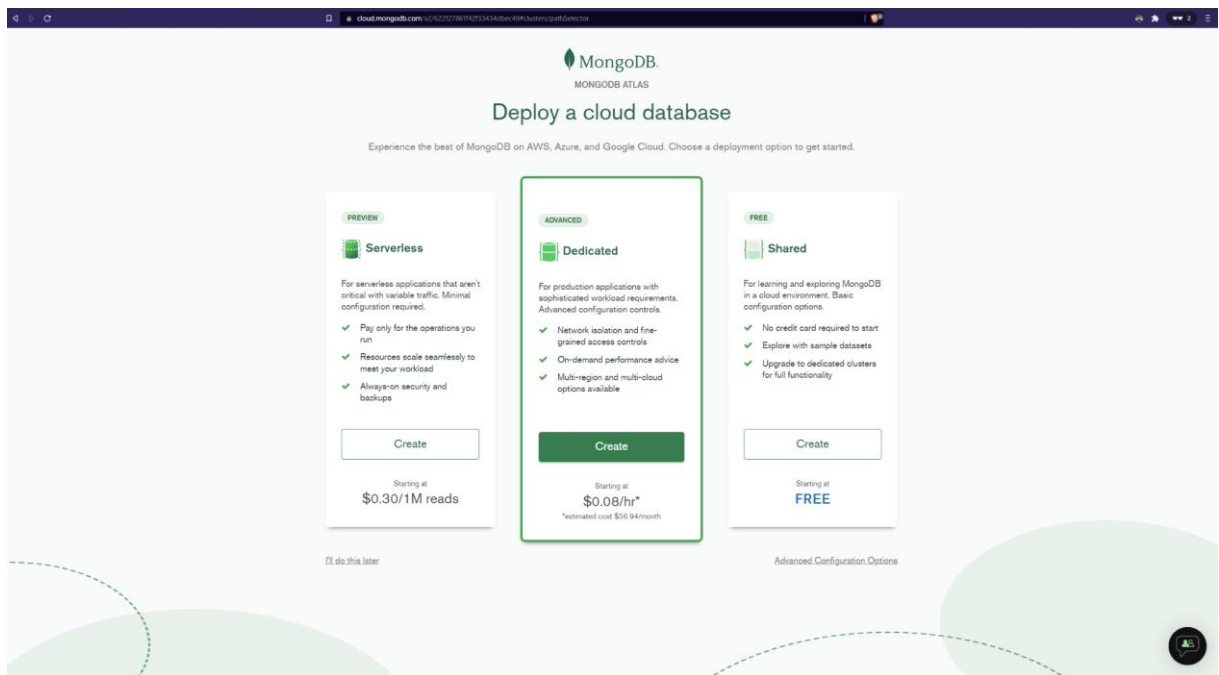
1. Bejelentkezés a MongoDB weboldalan (<https://www.mongodb.com/>), vagy új fiók regisztrációja ha még nincs fiók.



MongoDB regisztrációs oldal

2. Felhő alapú szerver kiválasztása

- Első bejelentkezés után (email visszaigazolás után) a Mongo Atlas elkezd megmutatni a konfigurációs lépések közül az elsőt.
- teszteléshez válasszuk ki az ingyenes tervet



Ingyenes szerver előfizetés kiválasztása

- utána válasszuk ki a szolgáltatót és a régiót (saját preferencia alapján), majd nyomjunk a zöld "Create Cluster" gombra.

Cloud Provider & Region AWS, Frankfurt (eu-central-1) ▾

aws

Google Cloud

Azure

★ Recommended region ⓘ (9) Paid tier region ⓘ

NORTH AMERICA	EUROPE	AUSTRALIA
Oregon (us-west-2) ★	Ireland (eu-west-1) ★	Sydney (ap-southeast-2) ★
N. Virginia (us-east-1) ★	Stockholm (eu-north-1) ★	ASIA
Ohio (us-east-2) ★ (9)	Paris (eu-west-3) ★	Singapore (ap-southeast-1) ★
N. California (us-west-1) (9)	<div style="border: 2px solid green; padding: 2px;"> Frankfurt (eu-central-1) ★ </div>	Mumbai (ap-south-1)
Montreal (ca-central-1) (9)	London (eu-west-2) ★ (9)	Tokyo (ap-northeast-1) ★
SOUTH AMERICA	Milan (eu-south-1) ★ (9)	Seoul (ap-northeast-2)
Sao Paulo (sa-east-1)	MIDDLE EAST	Hong Kong (ap-east-1) ★
	Bahrain (me-south-1) ★	Osaka (ap-northeast-3) ★ (9)
	AFRICA	
	Cape Town (af-south-1) ★	

MongoDB szerver lokációjának kiválasztása

3. Biztonsági/hozzáférési beállítások

- miután létrejött a “Cluster”, adjunk meg az adatbázisnak egy felhasználónevet és jelszót a “*Username*” és “*Password*”, mezők kitöltésével, amit a “*Create User*” gombbal kell hozzá adni. Ajánlott az egyszerűség kedvéért egy teszt környezetben speciális karakter nélkül létrehozni a jelszót, hogy később ne kelljen “*URL encode*”-olni.

- IP címeket is lehet fehér listázni, ez opcionális, ha minden hálózatról engedélyezni akarjuk az elérést, akkor a **0.0.0.0** IP címet kell csak hozzáadni a listához. Válasszuk ki a “*My Local Environment*” opciót és az alatta lévő mezőbe töltjük ki az “*IP Address*” opciót “0.0.0.0”-ra, a “*Description*”, pedig bármi lehet, majd nyomjunk rá az “*Add entry*” gombra, végül pedig a “*Finish and Close*” gombra. Elő fog jönni egy felugró ablak, ott a “*Go to Database*” gombra kell rá nyomni.

✓ **How would you like to authenticate your connection?**

Your first user will have permission to read and write any data in your project.

Username and Password

Certificate

Create a database user using a username and password. Users will be given the *read and write to any database* privilege by default. You can update these permissions and/or create additional users later. Ensure these credentials are different to your MongoDB Cloud username and password. You can manage existing users via the [Database Access Page](#).

Username

Password

Username	Authentication Type	
admin	Password	<input type="button" value="EDIT"/>

✓ **Where would you like to connect from?**

Enable access for any network(s) that need to read and write data to your cluster.

My Local Environment

Use this to add network IP addresses to the IP Access List. This can be modified at any time.

ADVANCED

Cloud Environment

Use this to configure network access between Atlas and your cloud or on-premise environment. Specifically, set up IP Access Lists, Network Peering, and Private Endpoints.

Add entries to your IP Access List

Only an IP address you add to your Access List will be able to connect to your project's clusters. You can manage existing IP entries via the [Network Access Page](#).

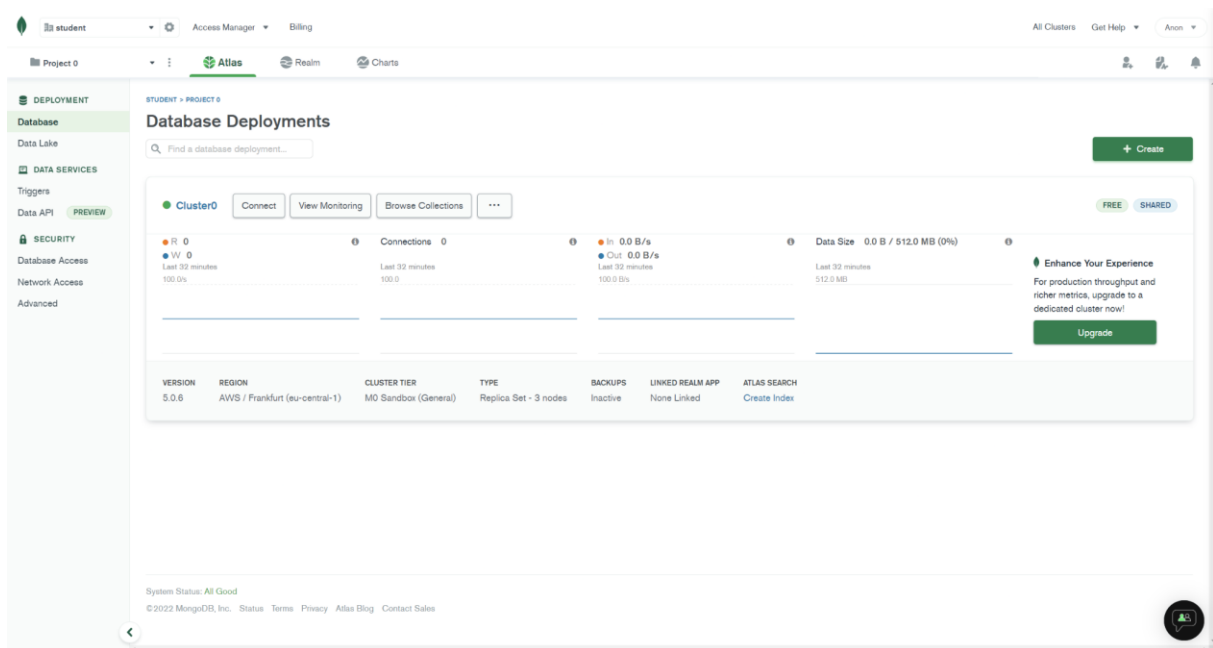
IP Address	Description	
<input type="text" value="Enter IP Address"/>	<input type="text" value="Enter description"/>	<input type="button" value="Add Entry"/> <input type="button" value="Add My Current IP Address"/>

IP Access List	Description	
0.0.0.0/0	*	<input type="button" value="REMOVE"/>

Felhasználó és elérés biztonsági beállítások

4. MongoDB adatbázis menü (*Database Deployments*)

- az adatbázis menüben az adatbázishoz kapcsolódó statisztikák és almenük láthatóak, pl a “Browse Collection”-ben lehet majd elérni a feltöltött adatokat, illetve operációkat végezni rajtuk.



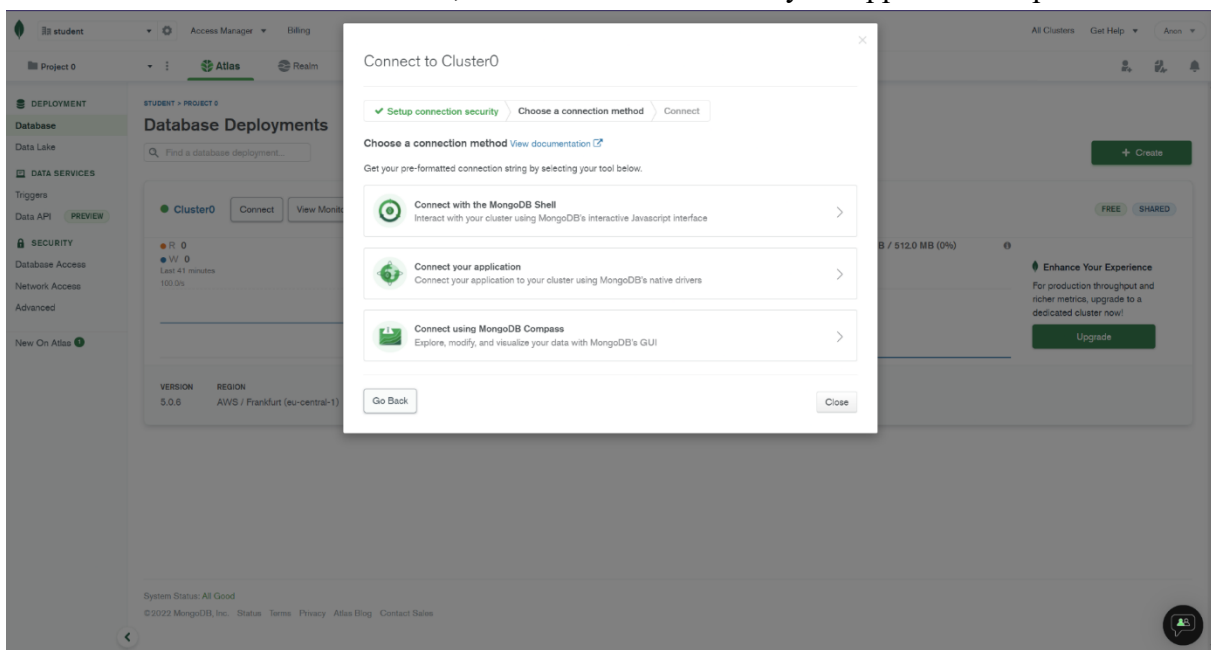
MongoDB irányítópult

5. Connection string

- ahhoz, hogy az API csatlakozzon az adatbázishoz, kell egy “connection string”, erről részletesebben a MongoDB saját dokumentációjában lehet olvasni:

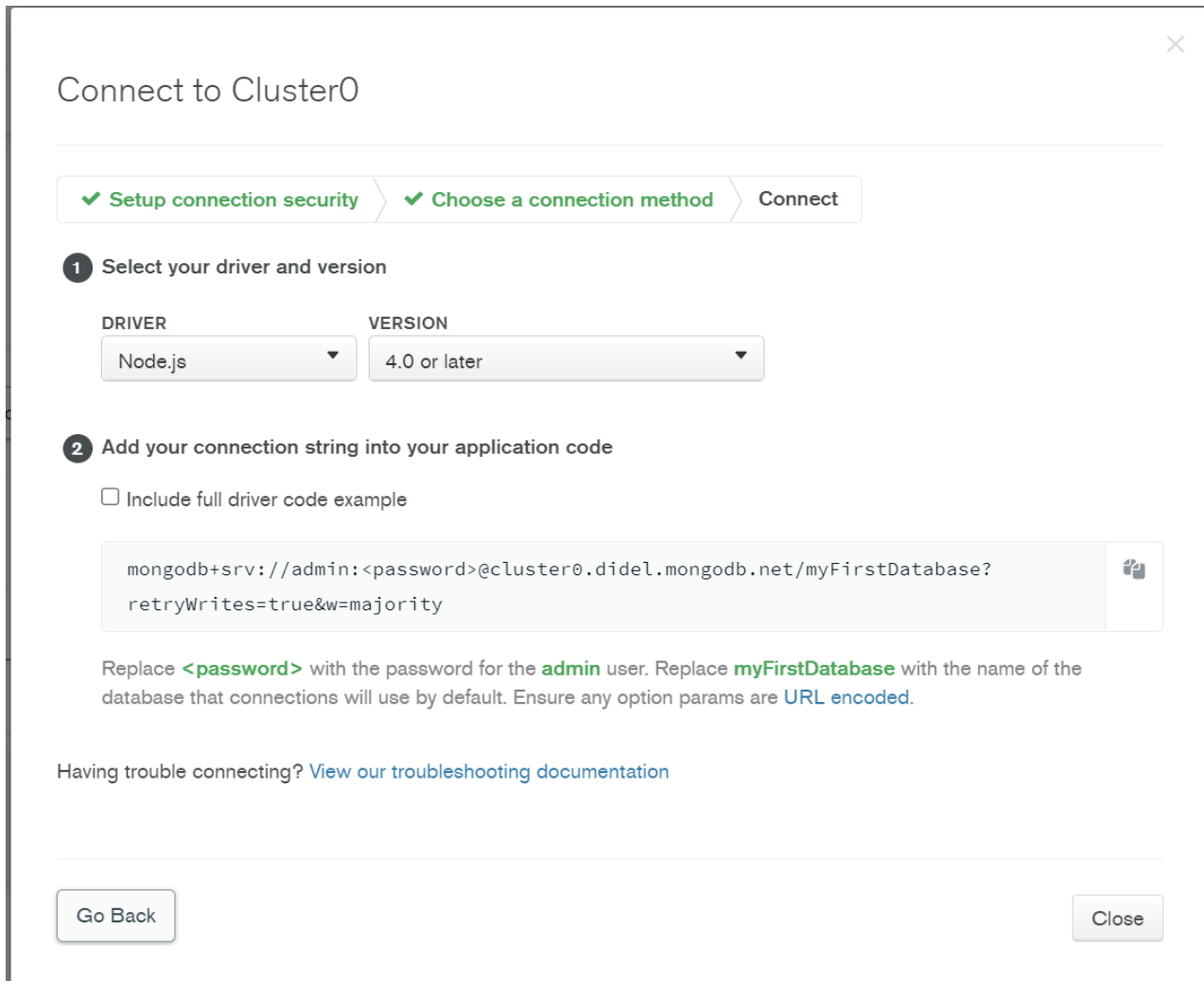
<https://docs.mongodb.com/manual/reference/connection-string/>

- Az adatbázis menüben a “Connect” gombra kattintva kapunk választásokat, hogy mivel akarunk az adatbázisra csatlakozni, válasszuk ki a “Connect your application” opciót



MongoDB csatlakozási mód kiválasztása

- az 1. szekcióban válasszuk ki “*DRIVER*”-nek a Node.js opciót, a verzióhoz pedig a legfrissebbet. A 2. szekcióban a “Include full driver code example” opciót nem kell bekapcsolni, így megkapjuk azt az “*URI*”-t, amivel az API csatlakozni fog az adatbázishoz.



Connect to Cluster0

✓ Setup connection security > ✓ Choose a connection method > Connect

1 Select your driver and version

DRIVER: Node.js | VERSION: 4.0 or later

2 Add your connection string into your application code

☐ Include full driver code example

```
mongodb+srv://admin:<password>@cluster0.didel.mongodb.net/myFirstDatabase?
retryWrites=true&w=majority
```

Replace **<password>** with the password for the **admin** user. Replace **myFirstDatabase** with the name of the database that connections will use by default. Ensure any option params are **URL encoded**.

Having trouble connecting? [View our troubleshooting documentation](#)

Go Back Close

MongoDB “connection string” másolása

- A connection string-ben a jelszót ki kell egészíteni a sajátunkra, illetve az adatbázis nevét is itt kell megadnunk. Ezeknek az értékeknek a connection string-ben “*URL encoded*” formában kell szerepelniük, erről a MongoDB saját dokumentációjában lehet részletesebben olvasni:

<https://docs.atlas.mongodb.com/troubleshoot-connection/#special-characters-in-connection-string-password>

Például:

mongodb+srv://admin:jelszo123%40@cluster0.didel.mongodb.net/tesztAdatbazisNeve?retryWrites=true&w=majority

6. Npm csomagok telepítése

- Az applikáció kettő darab “repository”-ből áll, a frontend kliensből és a backend szerverből.

- navigálni a könyvtárak között a következő paranccsal lehet: **cd <elérési útvonal>**

PL.: **cd videoderify-react-client**, kilépni a könyvtárból pedig a **cd ..** paranccsal lehet.

- mindkettő főkönyvtárban (root directory) futtassuk le a következő parancsot: **npm i**
Ez telepíteni fogja az applikáció futtatásához szükséges **npm** csomagokat, amik a *package.json* fájlokban vannak megadva és nem kell őket egyesével telepíteni.

```
[testvm@testvm-vmwarevirtualplatform ~]$ cd videoderify-react-client/
[testvm@testvm-vmwarevirtualplatform videoderify-react-client]$ npm i
npm WARN deprecated source-map-url@0.4.1: See https://github.com/lydell/source-map-url#deprecated
npm WARN deprecated source-map-resolve@0.6.0: See https://github.com/lydell/source-map-resolve#deprecated
npm WARN deprecated svgo@1.3.2: This SVGO version is no longer supported. Upgrade to v2.x.x.

added 1406 packages, and audited 1407 packages in 20s

170 packages are looking for funding
  run `npm fund` for details

7 moderate severity vulnerabilities

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
[testvm@testvm-vmwarevirtualplatform videoderify-react-client]$ cd ..
[testvm@testvm-vmwarevirtualplatform ~]$ cd videoderify-API/
[testvm@testvm-vmwarevirtualplatform videoderify-API]$ npm i
npm WARN deprecated @types/node-fetch@3.0.3: This is a stub types definition. node-fetch provides its own type definitions, so you do not need this installed.

added 317 packages, and audited 318 packages in 4s

23 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
[testvm@testvm-vmwarevirtualplatform videoderify-API]$
```

npm csomagok telepítése

7. Környezeti változók beállítása

- Az API környezeti változói tartalmazzák a szerver specifikus beállításokat, illetve a connection string-et is. Ezeket a fájlokat bármilyen szövegszerkesztővel meg lehet nyitni, a példában a terminálba beépített “NANO” szövegszerkesztő van használva.

- Példa elérési útvonala: **.env.example**

- Másoljuk át a példát a rendes .env fájlba: **cp .env.example .env**

- Elérési útvonal: **.env**

Megnyitása az API törzs könyvtárból: **nano .env**

Kilépés: **ctrl+x**, majd a mentéshez nyomjunk egy **y** karaktert, végül egy **entert**


- **PORT**: az API szerver hálózati portja, alapértelmezetten: **3001**

- **NODE_ENV**: a szerver kódjában használt változó, amivel fejlesztő környezetre tervezett logikákat lehet futtatni
- **DB_URI**: a MongoDB-ből kiszedett “connection string”
- **MEDIA_DIR**: Az az abszolút elérési útvonal, ahova a feltöltött média fájlokat az API tárolja, illetve ahonnan eléri őket. Ezt ha nem adjuk meg, akkor alapértelmezetten az API törzskönyvtáran belül fogja a fájlokat kezelni a szerver.

Unix környezetben az elérési útvonalat a következő formátumban kell megadni:

Példa: “/home/userName/media”

- **JWT_SECRET**: Egy random generált biztonsági kulcs, amivel az API felhasználókat tud biztonságosan azonosítani. Például lehet egy random GUID, vagy egy emberileg nem megjegyezhető hosszú karakter szekvencia.
- **API_KEY**: Szintén egy random generált biztonsági kulcs, amivel a fejlesztők megkerülhetik a felhasználói azonosítást bizonyos útvonalakon.
- **ORIGIN**: A kliens elérési útvonala, PL.: <http://localhost:3000>



```

GNU nano 6.2
PORT=3001
NODE_ENV="production"
DB_URI="mongodb+srv://"
MEDIA_DIR="/home/yourname/yourpath"
JWT_SECRET="bb8a41c1-6e8e-4a3a-819f-18d49c5d3f48"
API_KEY="f94e8b72-74da-4d7d-be26-3753997c0859"
ORIGIN="http://localhost:3000"
  
```

API környezeti változók beállítása a .env fájlba

- A kliens-nek csak egy darab környezeti változója van, az API-val ellentétben itt nem szabad érzékeny adatokat tárolni.
- Elérési útvonal: **src/.env**

Megnyitása a kliens törzs könyvtárából: **nano src/.env**

Kilépés: *ctrl+x*, majd a mentéshez nyomjunk egy *y* karaktert, végül egy *entert*

- **REACT_APP_API**: az API elérési útvonala, PL.: <http://localhost:3001>



```

GNU nano 6.2
REACT_APP_API http://localhost:3001
  
```

Kliens környezeti változó a .env fájlban

8. Programok futtatása

- API szerver elindításához a következő parancsot kell futtatni az API törzs könyvtárában: **npm run dev**

```
Terminal - testvm@testvm-vmwarevirtualplatform-~/videoderify-API
File Edit View Terminal Tabs Help
[testvm@testvm-vmwarevirtualplatform videoderify-API]$ npm run dev
> videoderify-api@2.0.0 dev
> tsc && node dist/index.js

Full path to your media directory: /home/testvm/videoderify-API/media
Server is listening on port 3001
```

API szerver elindítása

- A kliens futtatásához a következő parancsot kell futtatni a kliens törzs könyvtárában: **npm run start**
- Alapértelmezetten a <http://localhost:3000> címen érhető el a web applikáció.

```
WARNING in src/components/CommentForm.tsx
  Line 1:10: 'Dispatch' is defined but never used    @typescript-eslint/no-unused-vars
  Line 1:20: 'SetStateAction' is defined but never used @typescript-eslint/no-unused-vars

src/pages/EpisodesPage.tsx
  Line 147:6: React Hook useEffect has a missing dependency: 'onMount'. Either include it or remove the dependency array @typescript-eslint/no-unused-vars

src/pages/LibraryPage.tsx
  Line 14:10: 'errorMessage' is assigned a value but never used @typescript-eslint/no-unused-vars

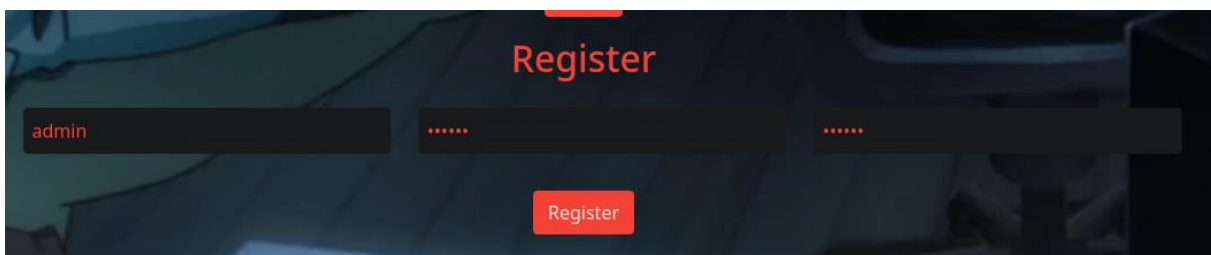
src/pages/UploadPage.tsx
  Line 11:10: 'errorMessage' is assigned a value but never used @typescript-eslint/no-unused-vars

webpack 5.67.0 compiled with 1 warning in 9991 ms
No issues found.
```

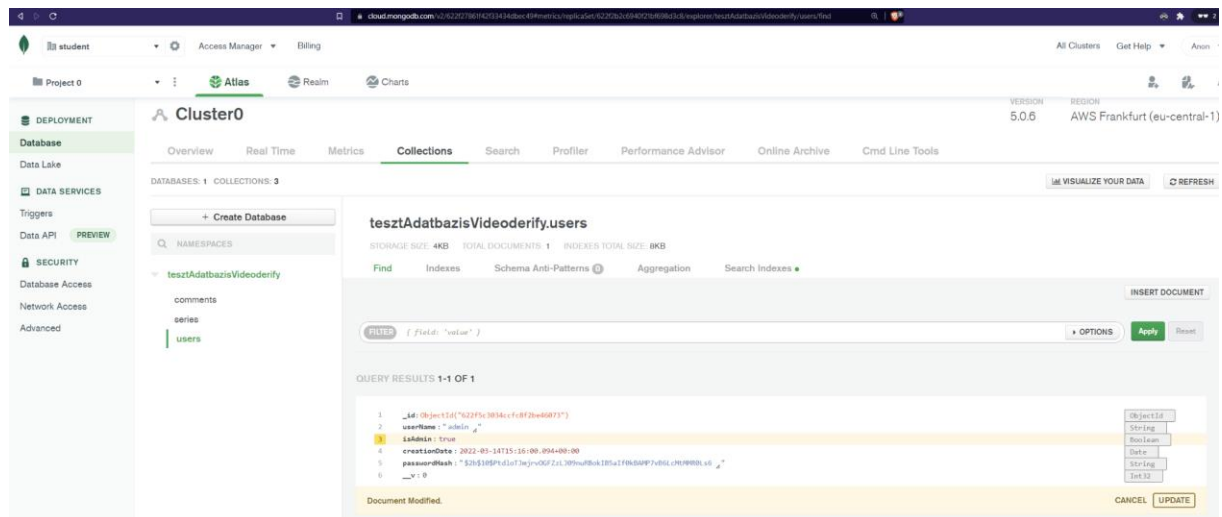
Kliens szerver elindítása

9. Admin felhasználó létrehozása

- Először regisztráljunk egy sima felhasználót a web applikáció főoldalán, aztán a MongoDB “Browse Collection” szekciójában válasszuk ki a **users** kollekciót és írjuk át a kívánt felhasználó **isAdmin** tulajdonságát **false**-ről **true**-ra és mentjük el a változtatást az **Update** gombbal.



Felhasználó regisztrálása



Felhasználó jogosultságának módosítása

```
_id: ObjectId("622f5c3034ccfc8f2be46073")
userName: "admin"
isAdmin: true
creationDate: 2022-03-14T15:16:00.094+00:00
passwordHash: "$2b$10$PtdloTJmjrvOGFZZLJ09nuRBokIB5aIf0kBAMP7vB6LcMtMMR0Ls6"
__v: 0
```

Példa egy admin felhasználóról az adatbázisban

Innentől az applikáció minden részét el lehet kezdeni használni.

- 2.3.2 A program telepítése (Windows)

1. Git windows verzió telepítése:

Töltsük le a számunkra megfelelő Git verziót a hivatalos oldalról és kövessük a telepítő utasításait: <https://git-scm.com/download/win>

2. NodeJS telepítése:

Töltsük le a NodeJS legfrissebb Windows verzióját és kövessük a telepítő utasításait: <https://nodejs.org/en/download/current/>

3. A fentiek telepítése után a "Git Bash" nevű terminál emulátort nyissuk meg, innen lehet majd a továbbiakban a parancsokat kiadni.



Git Bash terminál emulátor windows-on

4. Innentől a lépések megegyeznek a **“A program telepítése (GNU+Linux)”** szekció 3. lépésétől kezdve mindegyik lépésével (3-9.).

- Windows OS-en a **MEDIA_DIR** API környezeti változót a következő formában kell megadni: „C:\Users\felhasznaloNev\Downloads\media”

```

<> Found no color leading to 4.5:1 contrast ratio against #6c757d...
<> node_modules\bootstrap\scss\_functions.scss 168:3    color-contrast()
<> node_modules\bootstrap\scss\mixins\_buttons.scss 19:20  button-variant()
<> node_modules\bootstrap\scss\_buttons.scss 61:5        @import
<> node_modules\bootstrap\scss\bootstrap.scss 24:9       @import
<> src\style\customStyle.scss 2:9                         root stylesheet
<> Found no color leading to 4.5:1 contrast ratio against #6c757d...
<> node_modules\bootstrap\scss\_functions.scss 168:3    color-contrast()
<> node_modules\bootstrap\scss\mixins\_buttons.scss 80:17  button-outline-variant()
<> node_modules\bootstrap\scss\_buttons.scss 67:5        @import
<> node_modules\bootstrap\scss\bootstrap.scss 24:9       @import
<> src\style\customStyle.scss 2:9                         root stylesheet
<> Found no color leading to 4.5:1 contrast ratio against #6c757d...
<> node_modules\bootstrap\scss\_functions.scss 168:3    color-contrast()
<> node_modules\bootstrap\scss\helpers\_colored-links.scss 8:18  @import
<> node_modules\bootstrap\scss\_helpers.scss 2:9         @import
<> node_modules\bootstrap\scss\bootstrap.scss 49:9       @import
<> src\style\customStyle.scss 2:9                         root stylesheet
WARNING in
src\components\CommentForm.tsx
  Line 11:10: 'Dispatch' is defined but never used  @typescript-eslint/no-unused-vars
  Line 11:20: 'SetStateAction' is defined but never used  @typescript-eslint/no-unused-vars
src\pages\EpisodesPage.tsx
  Line 14:7:6: React Hook useEffect has a missing dependency: 'onMount'. Either include it or remove the dependency array
src\pages\LibraryPage.tsx
  Line 14:10: 'errorMessage' is assigned a value but never used  @typescript-eslint/no-unused-vars
src\pages\UploadPage.tsx
  Line 11:10: 'errorMessage' is assigned a value but never used  @typescript-eslint/no-unused-vars
webpack 5.67.0 compiled with 1 warning in 11040 ms
No issues found.

Anymus@Anymus-PC MINGW64 ~/Downloads/video
derify-API (master)
$ npm run dev

> videoderify-api@2.0.0 dev
> tsc && node dist/index.js

Full path to your media directory: C:\User
s\Anymus\Downloads\videoderify-API\media
Server is listening on port 3001

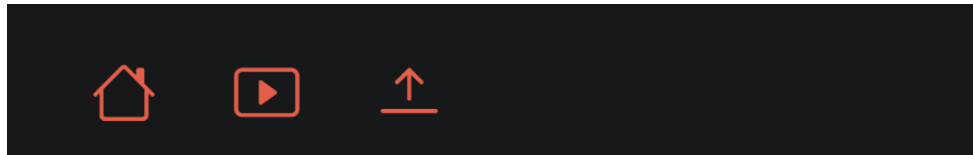
```

Példa az API és kliens szerverek futtatására Windows-on

2.4 A program használatának a részletes leírása

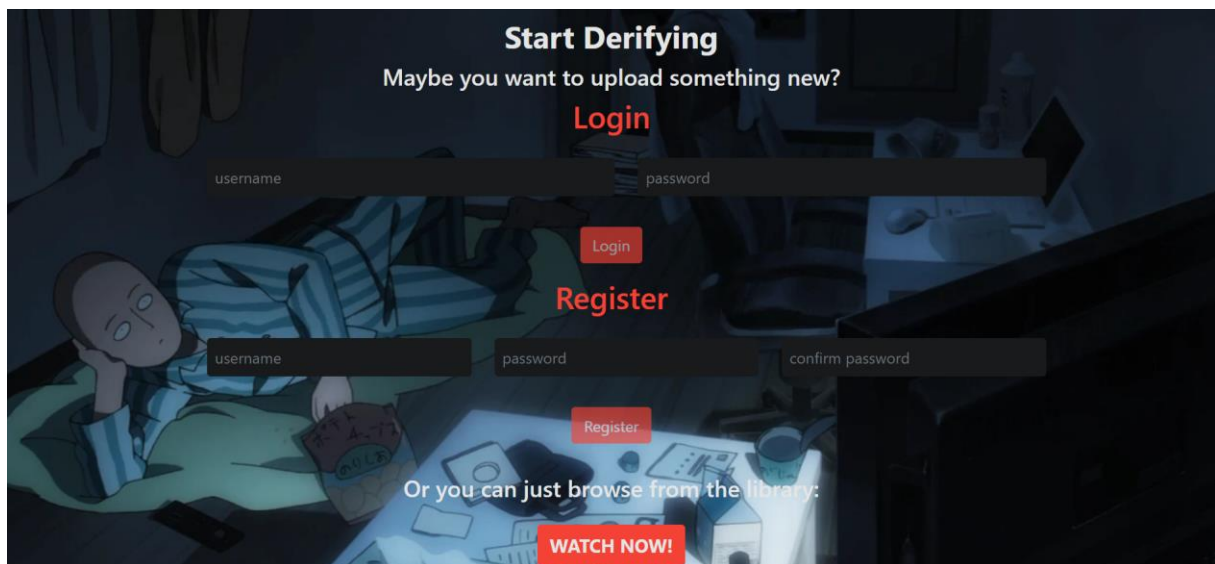
Menü

- Az oldalak között a menüben lévő ikonok megnyomásával lehet navigálni
- Oldalak: *Kezdőoldal, Könyvtár, Feltöltés, Tartalom oldala*



Navigációs menü

Kezdőoldal



Kezdőoldal

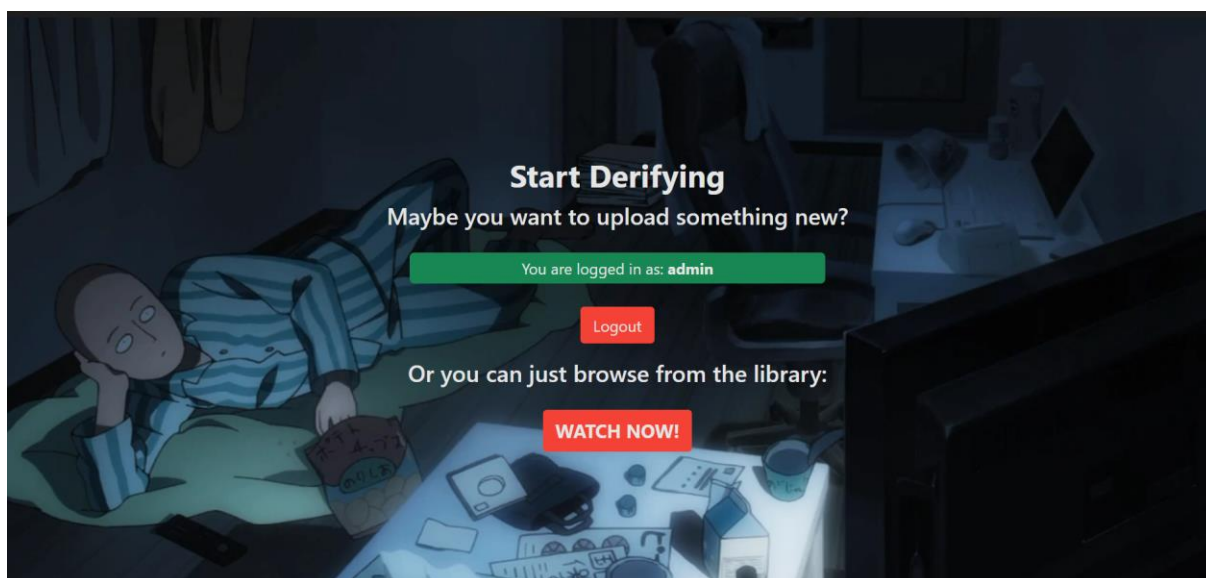
- **Bejelentkezés:** 2 kitöltendő mező van, a felhasználónév és a jelszó, ezek sikeres kitöltése után aktiválódik a “Login” gomb, aminek megnyomásával a szerver ellenőrzi a bevitt adatokat és ha sikeres, akkor a böngésző elmenti a bejelentkezés státuszát egy sütiben, amit a weboldal egy zöld háttérben lévő szöveggel fog jelezni, benne a felhasználó névvel: “You are logged in as: admin”. Hiba esetén egy piros betűkkel írt üzenet jelenik meg, ami részletezi a hibát.

- **Kijelentkezés:** A “Login” gomb “Logout” gombbá változik, ennek a megnyomására a szerver elfelejteti a bejelentkezés státuszát és újra megjelenik a bejelentkezéshez szükséges mezők.

- **Regisztráció:** Itt három darab mezőt kell kitölteni, *felhasználónév, jelszó és jelszó még egyszer*. Sikeres kitöltés esetén aktiválódik a “Register” gomb és ha sikeres, akkor

hibaüzenet nélkül üresek lesznek a kitöltött mezők és már be is lehet jelentkezni. Hiba esetén egy piros betűkkel írt üzenet jelenik meg, ami részletezi a hibát. Ez a regisztráció egy nem admin felhasználót hoz létre, admin jogosultságot az adatbázis kezelő felületén lehet adni egy felhasználónak, ahogy az a telepítés 9. pontjában demonstrálva van.

- **“WATCH NOW!”**: Ez a gomb átirányít a könyvtár oldalra.



Kezdőoldal bejelentkezve

Könyvtár

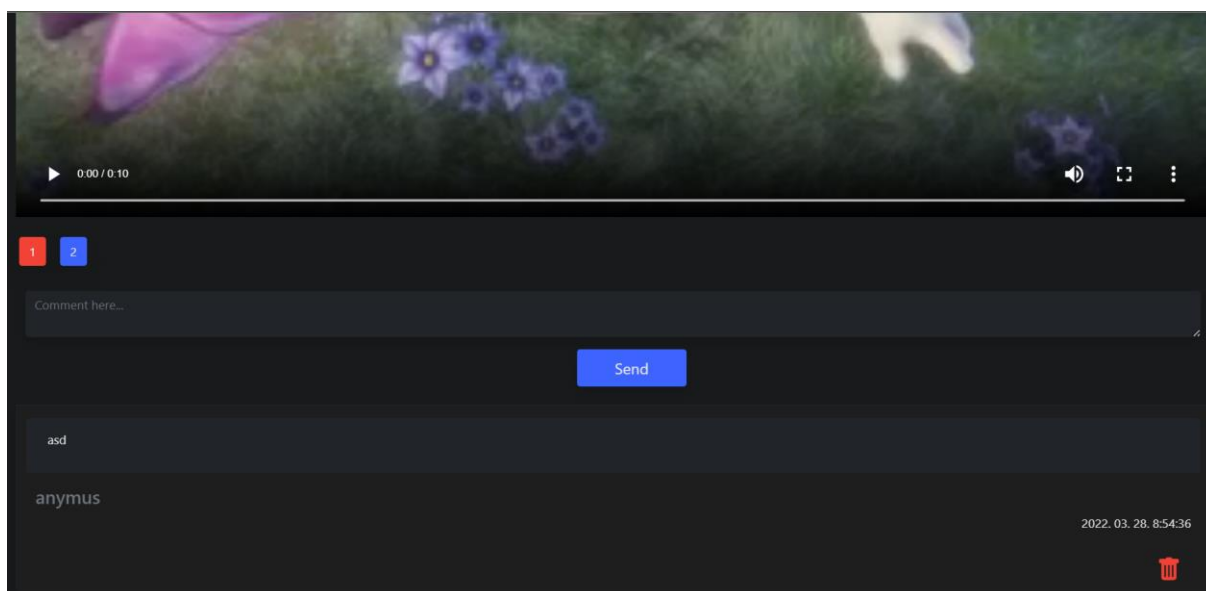


Könyvtár

- **Sorozat kártyák:** Minden feltöltött sorozat/média tartalmak a könyvtár oldalon vannak felsorolva kártyák formájában egymás mellett. Egy kártya tartalmaz egy “thumbnail” képet, a címet, a tartalmat feltöltő felhasználónak a nevét, illetve egy “Watch” feliratú gomb, amely át irányít annak a tartalomnak az oldalára, amelyik kártyához tartozik a gomb. Az egér rávitelével egy kártyára, az egész kártya elem mérete megnő a többihez képest, ezzel jelezve, hogy az a jelenleg fókuszban lévő kártya.

Abban az esetben, ha nincs egyáltalán semmi tartalom feltöltve, akkor nem jelenik meg kártya, helyettük egy sárga betűs üzenet jelenik meg, hogy nincs feltöltve a könyvtárba semmi. Ha a szerver, amivel, a kliens kommunikál nem elérhető, akkor egy piros betűs üzenet jelenik meg a problémáról.

Tartalom



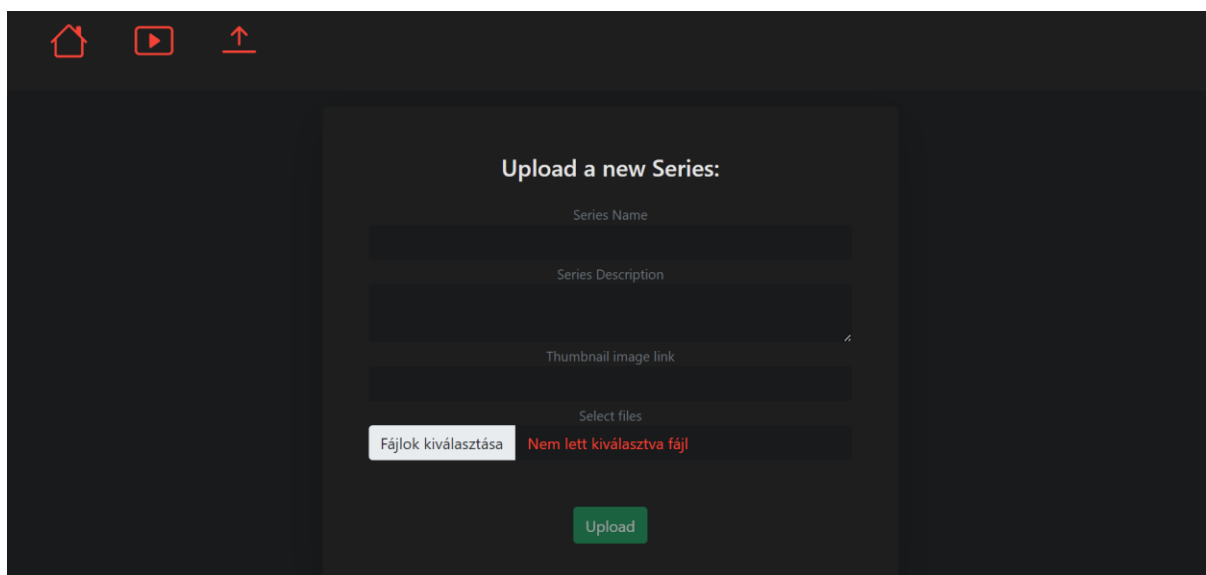
Tartalom oldal

- **Cím:** A legfelső sorban a feltöltéskor adott címe jelenik meg a tartalomnak.
- **Videó lejátszó:** Itt a tartalomnak a jelenleg kiválasztott epizódját lehet megtekinteni, ami alapértelmezetten a legelső. A lejátszó tartalmaz alapvető irányító elemeket, mint például a hang kezelésére használható sáv, vagy a videó jelenlegi pozícióját irányító sáv, illetve a teljes képernyő módot aktiváló gomb.
- **Epizód lista:** A tartalomhoz tartozó fájlok közül lehet választani, amik a feltöltés sorrendjében van számozott sorrendben megjelenítve. A kiválasztott epizódot a videólejátszó be fogja tölteni. Az összes epizód számnak kék színű a háttere, kivéve a kiválasztottnak, aminek piros háttere van.

- **Komment szekció:** Minden tartalomhoz tartozik egy komment szekció, amit bárki publikusan láthat bejelentkezés nélkül is. Egy bejelentkezett felhasználó a “Comment here...” mezőbe beírhatja a kommentjét és a “Send” gombbal azt publikálhatja. Egy kommentben megjelenik maga a szöveg, amit a felhasználó be gépelt, alatta a felhasználó neve aki kommentelt és a dátum amikor azt be küldte. Nem admin felhasználó csak a saját kommentjeit törölheti, másokét nem, admin felhasználó viszont bárki kommentjét bármilyen tartalom alatt ki törölheti.

- Ezt az oldalt csak a könyvtáron keresztül lehet elérni, a kártyákon lévő “Watch” gomb megnyomásával. A menüben ez az oldal nem szerepel.

Feltöltés



Feltöltés oldal

- Ez az oldal csak admin felhasználóval tud funkcionálni. Sima felhasználóval, illetve nem bejelentkezett vendégként nem lehet tartalmat feltölteni.
- **Series name:** A feltöltendő tartalom címe
- **Series description:** A tartalomhoz tartozó rövid ismertető/leírás.
- **Thumbnail image link:** Egy webes link, ami a tartalomhoz tartozó képre mutat, bármilyen publikusan elérhető link lehet. Ez a kép jelenik meg a könyvtár oldalon lévő kártyákon.
- **Select files:** Itt több fájlt lehet kiválasztani, ezek a tartalomhoz tartozó epizódok, amik a tartalom oldal epizódlistáján jelenik meg kiválasztható számozott sorrendben. Fontos, hogy a fájlokat a fájlkezelőben olyan sorrendben legyenek,

ahogy azt a epizódlistán meg szeretnénk jeleníteni. Erre a számozott sorrend ajánlott.

- **Feltöltés:** Az “Upload” gombra kattintva meg kezdődik a feltöltés, ami időigényes folyamat lehet a fájlok mennyiségétől, méretétől és a kliens/szerver internet kapcsolatától függően. A feltöltés folyamatát egy forgó kör fogja jelezni, sikeres feltöltés esetén a mezők tartalma üres lesz a töltő jel el már nem lesz látható. Hiba esetén a töltő jel helyén egy piros betűs hiba üzenet jelenik meg.

3 Fejlesztői dokumentáció

3.1 Az alkalmazott fejlesztői eszközök

A következő fejlesztő eszközök és npm modulok mind olyan liszenszel rendelkeznek, amelyek engedélyezik a szabad felhasználást.

- Visual Studio Code: Egy ingyenes kód szerkesztő, ami a legtöbb platformon fut és bármilyen program nyelven egyszerű vele elkezdni dolgozni, Egyik legnagyobb előnye a közösség által készített rengeteg kiegészítő, mellyel a szerkesztő magas szinten testre szabható bárkinek.

© 2022 Microsoft

MICROSOFT SOFTWARE LICENSE

- Git: A világ egyik legelterjedtebb verziókövető szoftvere

Linus Torvalds

GNU General Public License version 2.0

- GitHub
- NodeJS: Egy JavaScript runtime, ami lehetővé teszi, hogy JavaScript szerver oldalon is futhasson.
- MongoDB: Egy dokumentum orientált noSQL adatbázis, JSON dokumentumokat használ opcionális sémákkal.

© 2021 MongoDB, Inc.

Server Side Public License

- MongoDB Atlas: A MongoDB saját felhő alapú adatbázis és adat szolgáltatások platformja.
- Insomnia (<https://insomnia.rest/>): REST API építésére és tesztelésére használt applikáció, amellyel kéréseket lehet küldeni egy API szervernek http protokollon ke
- Archiver (<https://www.npmjs.com/package/archiver>)
- Bcrypt (<https://www.npmjs.com/package/bcrypt>)
- Compression (<https://www.npmjs.com/package/compression>)
- cookie-parser (<https://www.npmjs.com/package/cookie-parser>)
- cors (<https://www.npmjs.com/package/cors>)

- dotenv (<https://www.npmjs.com/package/dotenv>)
- ExpressJS: Egy gyors keretrendszer NodeJS-hez, amivel API-at lehet fejleszteni.

Copyright © 2017 StrongLoop, IBM, and other expressjs.com contributors.

The MIT License

- express-mongo-sanitize (<https://www.npmjs.com/package/express-mongo-sanitize>)
- fs-extra (<https://www.npmjs.com/package/fs-extra>)
- helmet (<https://www.npmjs.com/package/helmet>)
- jsonwebtoken (<https://www.npmjs.com/package/jsonwebtoken>)
- mongoose (<https://www.npmjs.com/package/mongoose>)
- multer (<https://npmjs.com/package/multer>)
- node-fetch (<https://www.npmjs.com/package/node-fetch>)
- tslib (<https://www.npmjs.com/package/tslib>)
- uuid (<https://www.npmjs.com/package/uuid>)
- ReactJS (<https://reactjs.org/>)
- jest-dom (<https://www.npmjs.com/package/@testing-library/jest-dom>)
- user-event (<https://www.npmjs.com/package/@testing-library/user-event>)
- Bootstrap 5: A világ egyik legelterjedtebb keretrendszere a reszponzív és mobilbarát weboldalak, web applikációk fejlesztésére rengeteg felhasználható példával, tippel és bő dokumentációval ellátva.

Copyright (c) 2011-2022 Twitter, Inc.

The MIT License

- bootstrap-icons (<https://www.npmjs.com/package/bootstrap-icons>)
- React: Egy JavaScript könyvtár felhasználói felületek készítésére.

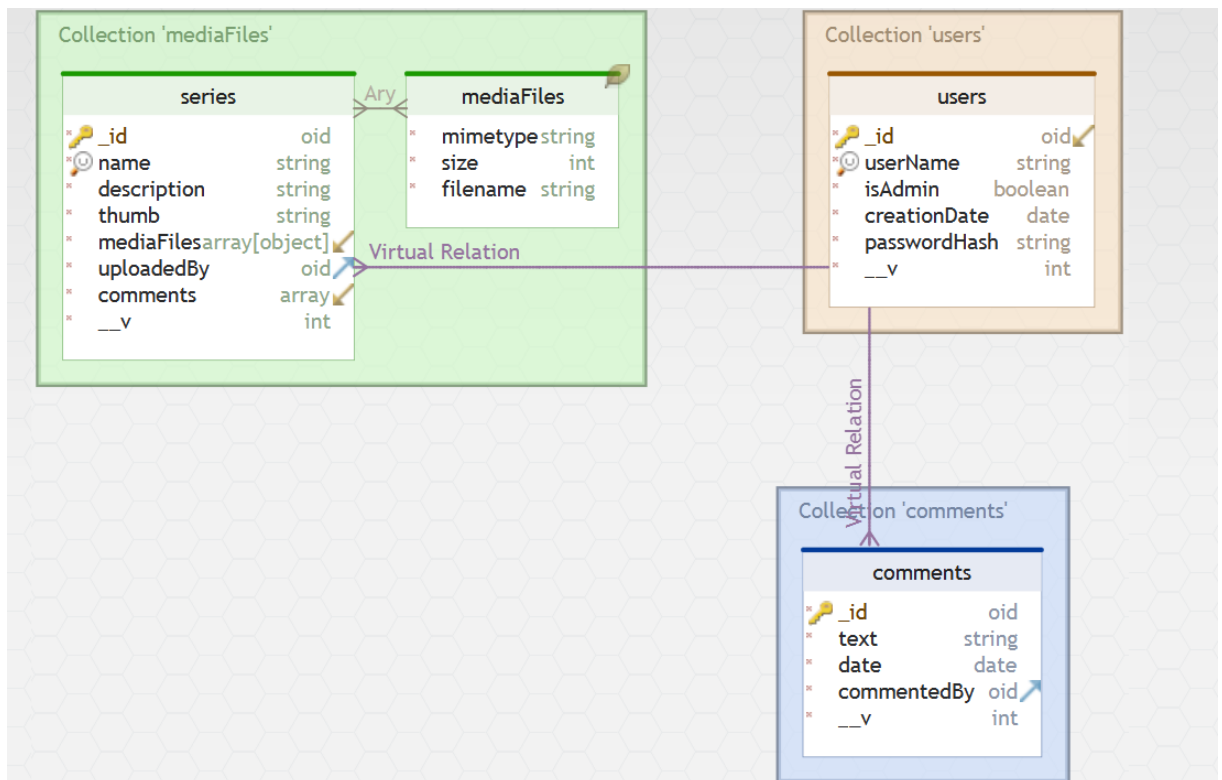
Copyright © 2022 Meta Platforms, Inc.

The MIT License

- react-dom (<https://www.npmjs.com/package/react-dom>)
- react-loader-spinner (<https://www.npmjs.com/package/react-loader-spinner>)
- react-router-dom (<https://www.npmjs.com/package/react-router-dom>)
- react-scripts (<https://www.npmjs.com/package/react-scripts>)
- sass (<https://www.npmjs.com/package/sass>)
- TypeScript: Egy JavaScript kiegészítő, ami lehetővé teszi, hogy a fejlesztés során adattípusokkal dolgozzunk fejlesztés során.

- web-vitals (<https://www.npmjs.com/package/web-vitals>)
- eslint (<https://www.npmjs.com/package/eslint>)
- React testing library (<https://reactjs.org/docs/testing.html>)
- Create React App (<https://create-react-app.dev/>)
- webpack (<https://webpack.js.org/>)

3.2 Adatmodell



Adatmodell

A következő dokumentumok jelennek meg az adatbázisban:

- **series:** Ez egy média tartalomnak a dokumentuma az adatbázisban
- **mediaFiles:** ez a dokumentum a médiának a hozzátartozó fájlját írja le
- **users:** a felhasználó tulajdonságait tároló dokumentum
- **comments:** a felhasználó által közzétett kommentnek a dokumentuma

Kapcsolatok:

- A *series* dokumentum kapcsolatban van a *users* dokumentummal az *uploadedBy* tulajdonságon keresztül. Ez egy olyan tulajdonság, amely mindig egy *user*

objektum *objectId*-t tartalmaz, mellyel referencálni lehet arra a *user*-ra, aki létrehozott éppen egy *series* objektumot.

- A *comments* dokumentum kapcsolatban van a *users* dokumentummal a *commentedBy* tulajdonságon keresztül, ami szintén egy referencia típus és annak a *user* objektumnak az *objectId*-ét tartalmazza, aki éppen létrehozta azt a *comment* objektumot.
- A *series* dokumentum kapcsolatban van a *comments* dokumentummal a *comments* tömbön keresztül. Ez egy olyan referencia típusú tömb, ami eltárolja az összes olyan *comment* objektumnak az *objectId*-ét, ami arra a *series* objektumra lett létrehozva.

3.3 Részletes feladatspecifikáció, algoritmusok

- **Felépítés:** A kliens és az API két külön szerveren futnak párhuzamosan, a kliens a környezeti változóban megadott érték alapján kapja meg az API elérési útvonalát, az API pedig szintén a saját környezeti változóban megadott értékekkel tudja felvenni a kapcsolatot az adatbázissal, illetve egyéb környezettől függő változót is itt lehet megadni.

Tehát például: fut egy API szerver a <http://localhost:3001> címen, fut egy másik szerver, ami a kliens a <http://localhost:3000> címen. Az API-nak tudnia kell a kliens címét, a kliensnek pedig az API címét, amiket a környezeti változókból lehet meg adni.

REST API Modulok:

Kiindulópont: package.json

Ebben a fájlban található a projekthez tartozó modulok, npm konfigurációk és futtatási parancsok. A szervert a “npm run dev” paranccsal lehet futtatni.

Környezeti változók: a *.env* fájlba definiálni kell a *.env.example*-ben látható változókat és kiegészíteni őket a futtatási környezetnek megfelelően.

A REST API controller-ek response-a mindig ugyanazt a sémát követi, ami a következő:
status: egy objektum, ami a következő tulajdonságokat tartalmazza

- success: egy bool, ami a response sikerét jelzi
- message: rövid leírás arról, hogy mi nem történt
- details: egy részletes leírás, ami hibák esetén szokott használni lenni.

result: a kimeneti adat, amit a response-ba el akarunk küldeni a kliensnek

series

Controller-ek:

A controller-ek JSON formátumban kapják és adják a kérést/választ.

- **DownloadAll:** Bemenete a *request.params.id*, feladata megkeresni a bemeneti id alapján azt a *series* objektumot, amelynek az id megegyezik a bemeneti id-vel és az összes hozzá tartozó *mediaFile*-t egy zip állományba tömörítve visszaküldeni a kliensnek a *response*-ban.
- **DownloadSingle:** Bemenete a *req.params.fileName*, feladata megkeresni a tárolt fájlt a *fileName* paraméter alapján és azt vissza küldeni a kliensnek a *response*-ban.
- **FindAllSerieses:** Nincs bemenete, feladata megkeresni az adatbázisban tárolt összes *series* objektumot és azokat visszaküldeni egy tömb formájában a kliensnek a *response*-ban.
- **FindSeries:** Bemenete a *request.params.id*, feladata megkeresni egy specifikus *series* objektumot, aminek az *objectId*-je megegyezik a paraméterben megadottal, a találatot visszaküldi a *response*-ban.
- **Upload:** Bemenete a kliensben kitöltött *form*, mely a *req.body*-ban elérhető. Feladata feldolgozni a *form* adatait és a hozzá csatolt fájlokat elmenteni a környezeti változóban megadott helyre, majd a bemeneti adatok alapján egy új *series* objektumot létrehozni az adatbázisban. Siker esetén egy "Series was created and saved!" üzenetet küld vissza a *response*-ban.
- **VideoStream:** Bemenete a *req.params.fileName* és a *req.headers.range*. Feladata a *fileName* alapján megkeresni a fájlt és azt szegmensekre osztva elküldeni a kliensnek. Ennek az az előnye, hogy a kliensnek nem egyből az egész fájl méretét kell betölteni, ami fájlról függően sok is lehet, hanem mindig csak azt a szegmenst, ami éppen következik. Ez memóriát és töltési időt tud spórolni bizonyos esetekben.

user

Controller-ek:

- **addUser:** Bemenete a kliensben kitöltött regisztrációs form, ami a *req.body*-ban elérhető. Feladata ellenőrizni a bevitt adatokat és siker esetén létrehozni egy új

user objektumot az adatbázisban. A jelszó paraméter a kientől *SHA256* formában érkezik és *bcrypt* hash-ként van elmentve az adatbázisba.

- **getAllUsers:** Nincs bemenete, feladata megkeresni az összes *user* objektumot és azokat vissza küldeni a response-ban.
- **getAuthenticatedUser:** Bemenete a *req.body.jwtUserId*, ami a jelenleg bejelentkezett felhasználóhoz tartozó *objectId*. Feladata a bemenet alapján megkeresni a vele egyező *user* objektumot az adatbázisban és azt vissza küldeni response-ban.
- **getUserById:** Bemenete a *req.params.userId*. Feladata a bemenet alapján megkeresni a hozzátartozó *user* objektumot. A **getAuthenticatedUser**-hez itt az a különbség, hogy nem a bejelentkezett felhasználóhoz tartozó objektumot keresi meg, hanem a paraméterben a szabadon megadott *objectId*-hez tartozót.
- **login:** Bemenete a felhasználó neve és jelszava. Feladata összehasonlítani a bemeneti adatokat az adatbázisban lévővel és siker esetén egy *JWT token*-t visszaküldeni response-ban. Ezt a *JWT token*-t a **getAuthenticatedUser** controller fogja használni, arra, hogy a kliens be bizonyítsa, hogy valóban az kommunikál a szerverrel, aki bejelentkezett. A *JWT token* egy sütiben lesz tárolva.
- **logout:** Nincs bemenete, törli a bejelentkezéskor kapott *JWT token*-t a sütikből.

comment

- **addComment:** Bemenete a kliensen kitöltött form. Feladata a bemenet alapján létrehozni egy új *comment* objektumot az adatbázisba.
- **deleteComment:** Bemenete a *req.params.seriesId* és a *req.params.commentId*. Feladata kitörölni a bemenetek által meghatározott *comment* objektumot. Ez a művelet a REST API-ban korlátozva van, hogy csak egy admin felhasználó tudjon bármit ki törölni, minden más esetben csak akkor elérhető a controller, ha az a felhasználó akar törölni, akihez tartozik a törlendő komment.

Kliens:

- Az API teljesen független a kliens implementációját, akármilyen kliens képes használni az API-t, ami rendelkezik egy http klienssel és egy JSON parser-el.

Kiindulópont:

package.json

Ebben a fájlban található a projekthez tartozó modulok, npm konfigurációk és futtatási parancsok. A szerveret a “npm run start” paranccsal lehet futtatni.

Környezeti változók: a *.env* fájlba definiálni kell a *REACT_APP_API* változót az API szerver elérési útvonalával (URL), PL.: *http://localhost:3001*

Videoderify React kliens implementáció: Ez egy komponens alapú Single Page Application (SPA), ami a REST API-al kommunikál JSON dokumentumokon keresztül. A beviteli mezők és a kiviteli adatok renderelése az API alapján lettek követve. A kliens is ugyan azokat a TypeScript modelleket használja, amik az API-ban már definiálva vannak.

Főbb oldalak a “pages” könyvtárban:

- HomePage: ez a kezdőoldal.
- LibraryPage: Ez a könyvtár oldal, ahol a tartalom kártyák jelennek meg egymás mellett.
- EpisodesPage: Itt jelenik meg a videólejátszó és a médiához tartozó epizódok, illetve a komment szekció is.
- UploadPage: Itt lehet kitölteni a média tartalom feltöltéséhez szükséges form-ot.

Komponensek:

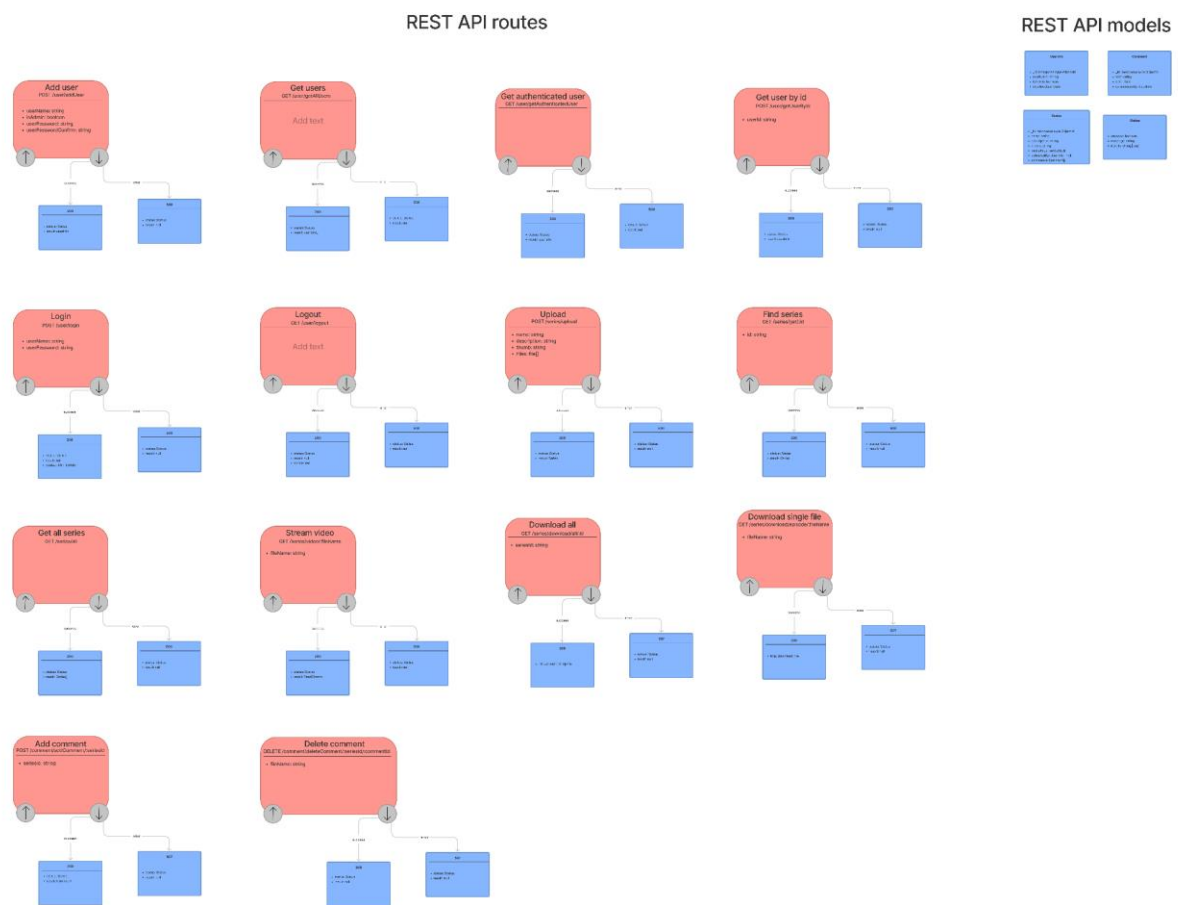
- Comment: egy komment objektumot jelenít meg
- CommentForm: Egy komment létrehozásához szükséges form-ot tartalmaz.
- Login: a bejelentkezéshez szükséges form-ot tartalmazza, illetve sikeres bejelentkezés esetén a bejelentkezett státuszt jeleníti meg.
- NavBar: A weboldal navigációs menüjét tartalmazza.
- Register: a felhasználó regisztrációjához szükséges form-ot tartalmazza.
- SeriesCard: egy series objektumot jelenít meg kártya formájában, illetve tartalmazza a linket, ami arra a média tartalomnak az “EpisodesPage”-ére irányít át.
- Status: több oldalon van használva, http fetch státuszt és hibákat jelenít meg a felhasználónak.

- UploadSeriesForm: Egy média tartalom feltöltéséhez szükséges form-ot tartalmaz.
- VideoPlayer: Ez a html videó lejátszó, ami megkapja a videó stream-et a “/series/video/:fileName” API elérési útvonaltól.

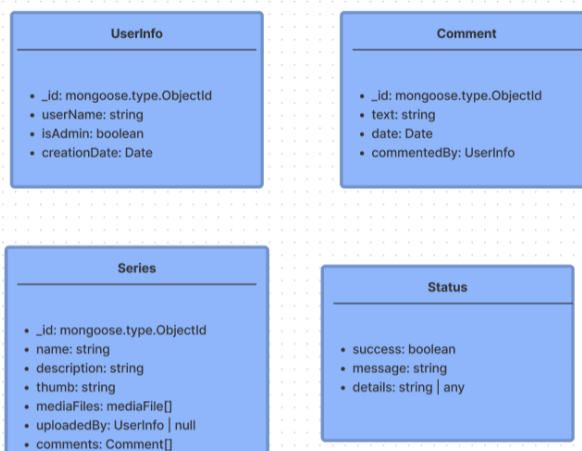
Stílus használat: A legtöbb helyen a bootstrap 5 komponensei, illetve css class-ai vannak alkalmazva, de lehetséges saját stílus formázás a style könyvtárban elhelyezett scss fájlokkal. A *global.scss*-ben elhelyezett tartalom globálisan az egész applikáción elérhető lesz, míg a többi fájl a komponensekre, vagy oldalakra specifikusak, illetve lehetséges a bootstrap alapértelmezett stílusait felülírni a *customStyle.scss* fájlon keresztül.

Az ábra a REST API felépítését, útvonalait és modeljeit ábrázolja:

Nagy felbontású kép: <https://i.imgur.com/v7gPUos.png>



REST API models



REST API ábrázolásai

3.4 Tesztelési dokumentáció

React komponensek tesztelése

React Testing Library: ez a React által készített könyvtár, melynek segítségével előre megírt funkciókat lehet használni arra, hogy a le renderelt DOM elemeket teszteljük, ami azért előnyös, mert így az igazi felhasználáshoz nagyon hasonló teszt környezetet lehet ki alakítani. Ennek a tesztnek a során az egyik legfőbb probléma, amit sikerült kiszűrni, az a form-ok kitöltése során volt, mivel a program nem ellenőrizte, hogy a mezők például ki vannak-e töltve, a felhasználó üres adatokat volt képes küldeni.

Hibaüzenet az API-ból kapott sikertelen kérés volt, mivel ezt a fajta hibát az API le tudja kezelni, ezért semmi kritikus nem történt, viszont a felhasználó élményt rontja.

Ennek a tesztelésnek nem feltétlen van szüksége egy programatikus eszközhöz, itt szimplán a felhasználó szempontjából az applikáció használatának tesztelése a cél, azaz a szoftver viselkedését teszteli.

Jest: Egy JavaScript teszt keretrendszer, ami kompatibilis ReactJS-el, NodeJS-el és TypeScript-el. Egyik nagy előnye, hogy nem igényel konfigurációt. Ez a keretrendszer egyszerűvé teszi, hogy nagy objektumokat követni lehessen. Itt az API tesztelésnél jött elő az a probléma, hogy ha egy felhasználó töltött fel tartalmat, vagy kommentet, de a felhasználó törölve lett utána, akkor azok a tartalmak amik a törölt felhasználótól származnak, nem tudtak rendesen megjeleníteni *undefined* hibák miatt.

A következő volt a hibaüzenet: “Uncaught ReferenceError: series.result.uploadedBy is not defined at <anonymous>:1:1”

A program egy olyan tulajdonságot próbált elérni, ami nem létezett, hiszen nem talált hozzá az adatbázisban hozzá referenciát. Ennek a megoldása a controllerben átírni ezt a tulajdonságot *null* típusra abban az esetben, ha nincs rá találat. Ezt a null típust kliensben már le lehet kezelni. Ehhez a teszteléshez szükséges programozás, hiszen a szoftver belső struktúrája, illetve kódja van tesztelve, azaz a logikát teszteli.

TypeScript type safety: A TypeScript egyik legelterjedtebb funkciója az, hogy még fejlesztés közben ellenőrzi a változók/függvények/objektumok adattípusait, ami alapvetően a JavaScript-ben nincs benne. Ez rendkívül fontos fejlesztés közben, ennek a segítségével még a kód futtatása előtt ki lehet fogni a típus hibákat, ez egy gyorsabb és

fájdalommentesebb fejlesztéshez vezet. Itt a hiba üzenetek magában a VScode fejlesztő környezetben jelennek meg szokásos syntax hibák formájában, PL.: *“Type 'string' is not assignable to type 'number'”*

Ilyenkor a teendő visszakeresni a kódban, hogy hol adunk meg rossz értéket és azt kijavítani.

4 Összefoglaló

- 4.1 Továbbfejlesztési lehetőségek

- Fájlok tárolása egy disztributált object storage fájlrendszeren. Ez azért lenne előnyös, mert a médiához tartozó fájlokat felosztva több csomóponti szerveren lehet tárolni, ami azt jelenti, hogy nem egy helyen egy lemezen léteznek a fájlok, hanem több egymástól független tárolóegység, ami nagyobb biztonságot és redundanciát ad jelentősen kevesebb költségekkel a szokásos felhő alapú tároláshoz képest. Egy object storage meta adatokra fókuszál, ezért jobbak a lehetőségek az analitikákra és tovább skálázható, mint egy szokványos fájl tárolási megoldás, illetve a fájl hierarchia hiányának köszönhetően sokkal gyorsabbak is az olvasási műveletek, a metaadatok pedig teljesen testre szabhatóak, ami jobb integrációt nyújt egy modern applikációnak.
- Egy smart contract alapú blockchain hálózaton tárolt autentikációs rendszert érdemes lehet a jövőben kifejleszteni. Ennek az az előnye az, hogy a felhasználók nem egy centralizált adatbázison keresztül férnek hozzá az applikációhoz, hanem a saját privát kulcsokkal egy aszimmetrikus titkosító algoritmus segítségével, ami rengeteg “privacy” problémára lenne megoldás, illetve az applikáció egy olyan ökoszisztéma része lenne, ahol a felhasználók ugyanazt az identitást tudják használni (privát kulcs) több applikáción keresztül, ez könnyebb és magán adat tisztelőbb szociális hálózathoz vezethet.
- Az applikáció szociális részének további elemekkel való bővítése is érdemes, a nagyobb felhasználó bázis bevonásához, illetve felhasználói elkötelezettségek növeléséhez. Ilyen elem lehet például egy fórum szerű poszt rendszer és egy privát chat funkció.
- Egy olyan funkció, ami elmenti a böngészőbe (PL.: localStorage) a videólejátszó jelenlegi státuszát egy kiválasztott epizódnak, hogy ha kilép a felhasználó az applikációból, de visszalép, akkor ott tudja folytatni a lejátszást, ahol abbahagyta.

- 4.2 Önértékelés

Tudásom java részét eddigi projektek fejlesztéséből szereztem, amiket főleg saját érdeklődésből űztem, ezekhez, ahogyan a Videoderify-hoz is az interneten

lévő szabadon megtekinthető források használatával voltam képes a szükséges eszközöket elsajátítani. A legtöbb tapasztalat számomra abból jön, mikor egy problémába akadok és annak utána járok és mikor sikerül a megoldásra rátalálni, vagy rájönni, akkor egyrészt megértem a probléma okát, másrészt az a probléma legközelebb már ismerős lesz, vagyis ezekből a problémákból tanulni lehet. A projectemben meg próbáltam követni más, nagyobb cégek által használt standardokat is, melynek segítségével számomra és mások számára is átláthatóbb a fejlesztési folyamat.

5 Felhasznált irodalom

ReactJS dokumentáció: <https://reactjs.org/docs/getting-started.html>

Utoljára megtekintve: 2022.03.22

ExpressJS dokumentáció: <https://expressjs.com/en/4x/api.html>

Utoljára megtekintve: 2022.01.15

Mongoose dokumentáció: <https://mongoosejs.com/docs/>

Utoljára megtekintve: 2022.03.23

Multer dokumentáció: <https://github.com/expressjs/multer#readme>

Utoljára megtekintve: 2022.04.02

fs-extra dokumentáció:

Utoljára megtekintve: 2021.11.21

<https://github.com/jprichardson/node-fs-extra/blob/master/README.md>

eslint dokumentáció: <https://www.npmjs.com/package/eslint>

Utoljára megtekintve: 2022.03.25

jsonwebtoken dokumentáció: <https://www.npmjs.com/package/jsonwebtoken>

Utoljára megtekintve: 2022.03.25

bcrypt dokumentáció: <https://www.npmjs.com/package/bcrypt>

Utoljára megtekintve: 2022.03.26

Traversy Media - “Learn The MERN Stack - Express & MongoDB Rest API” videó:

<https://www.youtube.com/watch?v=-0exw-9YJBo>

Utoljára megtekintve: 2022.02.12

Traversy Media - “Learn The MERN Stack - JWT Authentication” videó:

<https://www.youtube.com/watch?v=enopDSs3DRw>

Utoljára megtekintve: 2022.02.16

React testing library dokumentáció: <https://reactjs.org/docs/testing.html>

Utoljára megtekintve: 2022.01.18

TypeScript dokumentáció: <https://www.typescriptlang.org/docs/>

Utoljára megtekintve: 2021.09.19

sass dokumentáció: <https://sass-lang.com/documentation>

Utoljára megtekintve: 2021.10.13

w3schools: <https://www.w3schools.com/>

Utoljára megtekintve: 2022.04.02

A videó streamelés algoritmus forráskódjának idézete:¹

```
const stat = fs.statSync(path)

const fileSize = stat.size

const range = req.headers.range

if (range) {
```

¹ Idézet forrása: (NodeJS video stream példa: <https://nodeblogger.com/video-streaming-in-node-js/>)


```

const parts = range.replace(/bytes=/, "").split("-")

const start = parseInt(parts[0], 10)

const end = parts[1]
  ? parseInt(parts[1], 10)
  : fileSize-1

if(start >= fileSize) {
  res.status(416).send('Requested range not satisfiable\n'+start+'
>= '+fileSize);

  return
}

const chunksize = (end-start)+1
const file = fs.createReadStream(path, {start, end})
const head = {
  'Content-Range': `bytes ${start}-${end}/${fileSize}`,
  'Accept-Ranges': 'bytes',
  'Content-Length': chunksize,
  'Content-Type': 'video/mp4',
}

res.writeHead(206, head)

file.pipe(res)
} else {
  const head = {
    'Content-Length': fileSize,
    'Content-Type': 'video/mp4',
  }

  res.writeHead(200, head)

  fs.createReadStream(path).pipe(res)
}

```

6 **Ábrajegyzék:**

[OS frissítés](#)

[Letöltött források ellenőrzése](#)

[NodeJS telepítése](#)

[NodeJS verzió ellenőrzése](#)

[MongoDB regisztrációs oldal](#)

[Ingyenes szerver előfizetés kiválasztása](#)

[MongoDB szerver lokációjának kiválasztása](#)

[Felhasználó és elérés biztonsági beállítások](#)

[MongoDB irányítópult](#)

[MongoDB csatlakozási mód kiválasztása](#)

[MongoDB “connection string” másolása](#)

[npm csomagok telepítése](#)

[API környezeti változók beállítása a .env fájlba](#)

[Kliens környezeti változó a .env fájlban](#)

[API szerver elindítása](#)

[Kliens szerver elindítása](#)

[felhasználó regisztrálása](#)

[felhasználó jogosultságának módosítása](#)

[példa egy admin felhasználóról az adatbázisban](#)

[Git Bash terminál emulátor windows-on](#)

[Példa az API és kliens szerverek futtatására Windows-on](#)

[Navigációs menü](#)

[Kezdőoldal](#)

[Kezdőoldal bejelentkezve](#)

[Könyvtár](#)

[Tartalom oldal](#)

[Feltöltés oldal](#)

[Adatmodell](#)

[REST API ábrázolása](#)