Web-technológiák beadandó feladat dokumentáció

Kórik Edina

2025

GDVK5H

Tartalom

[BűnTárs 3](#_Toc197872970)

[Futtatás 3](#_Toc197872971)

[Kezdőoldal 3](#_Toc197872972)

[Regisztráció 4](#_Toc197872973)

[Jelöltek 5](#_Toc197872974)

[Szabályzat 7](#_Toc197872975)

[Partnereink 7](#_Toc197872976)

[Routing 8](#_Toc197872977)

# BűnTárs

Az oldal egy bűnözőknek készült társkereső. Egyaránt használhatják elítélt és (egyelőre még) nem elítélt bűnözők. Több börtönintézet is részt vesz a kezdeményezésben, hogy társat találjanak a magányos rabok, mert ez javítja a magaviseletüket és a hozzáállásukat rabtársaikhoz.

Sokan a társadalom kitaszított tagjainak vélik a volt vagy jelenlegi rabokat/bűnözőket, és nem éreznek semmilyen empátiát irántuk. Azonban nekik is szükségük van társra, valakire aki jóban-rosszban mellettük áll. És ki lenne erre alkalmasabb, mint egy másik bűnöző?

# Futtatás

Az oldalt a backend szerver futtatása után az ng serve --proxy-config proxy.conf.json paranccsal futtathatjuk az Angular projektet. A proxy.conf.json fájl tartalma mindössze ennyi:

{

    "/api": {

      "target": "http://localhost:3000",

      "secure": false,

      "pathRewrite": {

        "^/api": ""

      }

    }

}

Erre a CORS (Cross-Origin Resource Sharing) megkerülése végett volt szükség.

# Kezdőoldal

A kezdőoldalon egy figyelemfelkeltő üdvözlő szöveg található, illetve szívecske alakú bilincsek, kihangsúlyozva ezzel az oldal fő célját. A „Regisztrálj most!” hivatkozásra kattintva regisztrálhatunk a társkeresőbe.A képen lánc látható

Automatikusan generált leírás

# Regisztráció

A regisztráció 3 lépésből áll, a léptetést az Angular Material Stepper-je segítségével oldottam meg. 

Az első lépésben alapvető adatokat kell megadni, mint név, nem, illetve az elkövetett bűncselekmények. A vezetéknév és a keresztnév egyaránt legalább 1, maximum 20 karakter. Ezek hosszára Validatort állítottam be.

        firstName: ['', [Validators.required, Validators.minLength(1), Validators.maxLength(20)]],

        lastName: ['', [Validators.required, Validators.minLength(1), Validators.maxLength(20)]],

A második lépésben dátumválasztóval adható meg a börtönben töltött idő, de természetesen ez nem kötelező, mert az oldal nem csak elítélt bűnözőknek lett címezve.

Végül a profil testreszabása következik, ahol írhatunk egy mottót vagy rövid bemutatkozást, illetve profilképet tölthetünk fel. Majd a „Regisztrálok!” gombra kattintva fejezhetjük be a regisztrációt. Ez egyúttal átirányít minket a jelöltekhez. A regisztrációkor az adatok bejegyzésre kerülnek a felhasználók adatait tartalmazó MongoDB adatbázisba.

users.service.ts:

  registerUser(formData: FormData) {

    return *this*.http.post<any>('/api/users', formData);

  }

server.js (backend szerver):

app.post('/users', upload.single('photo'), async (req, res) => {

      try {

        const userData = {

          name: req.body.name,

          motto: req.body.motto,

          photo: req.file ? req.file.filename : null,

          crimes: req.body.crimes

        };

        const result = await usersCollection.insertOne(userData);

        res.status(201).json({

          message: 'User created successfully',

          userId: result.insertedId

        });

      } catch (error) {

        console.error('Error creating user:', error);

        res.status(500).send('Internal Server Error');

      }

    });

# Jelöltek

A jelöltek mindegyikét egy Angular Material Card-ban helyeztem el, így egyszerűen lehetett kivitelezni egy szép elrendezést. Mindegyik ilyen kártya tartalmazza a profilképet (ha van), a nevet, a mottót/bemutatkozást, a bűncselekményeket, és egy „Tetszik” gombot. A gombra kattintva a benne található szív piros színű lesz, jelezve ezzel a felhasználónak, hogy a kedvelést az illető megkapta. A képen szöveg, Emberi arc, ember, képernyőkép látható

Automatikusan generált leírásA képen szöveg, Emberi arc, ember, képernyőkép látható

Automatikusan generált leírás

A kedvelések a böngésző localStorage-ében kerülnek tárolásra.

users.service.ts:

  addLike(id: string) {

    const likedUserIds: string[] = *this*.getLikedUsers();

    if (!likedUserIds.includes(id)) {

      likedUserIds.push(id);

      localStorage.setItem('likedUsers', JSON.stringify(likedUserIds));

    }

    return of({ success: true });

  }

A kedvelt személyek megtalálhatók a Kedvencek fül alatt. Itt ugyanúgy jelennek meg a jelöltek, mint az előző oldalon, a kedvelést így el is lehet távolítani.

A jelöltek adatai egy MongoDB adatbázisban vannak tárolva, a következőképpen:

|  |  |  |
| --- | --- | --- |
| Adat | Típus | Leírás |
| \_id | ObjectId | A MongoDB által automatikusan hozzáadott objektumazonosító |
| name | szöveg | A jelölt neve |
| photo | szöveg | A jelölt profilképének fájlneve |
| photodisplacement | szöveg | A profilkép középre igazításában segít |
| crimes | szöveg | A jelölt által elkövetett bűncselekmények vesszővel elválasztva |
| motto | szöveg | A jelölt rövid bemutatkozása |

A profilképek a my-backend/uploads mappába kerülnek mentésre. A backend szerver következőképpen kéri le a felhasználók adatait:

app.get('/users', async (req, res) => {

      try {

        const users = await usersCollection.find({}).toArray();

        const mappedUsers = users.map(user => {

          const { \_id, ...rest } = user;

          return { ...rest, id: \_id.toString() };

        });

        res.json(mappedUsers);

      } catch (error) {

        console.error('Error fetching users', error);

        res.status(500).send('Internal Server Error');

      }

    });

Ezután a candidates.component.ts-ben így kerülnek kilistázásra:

  getCandidates(){

*this*.usersService.listCandidates().subscribe(data => {

*this*.candidateList=data

    }, (err) => {

      console.error('Error fetching candidates:', err);

    });

  }

Ezután a html-ben egy \*ngFor parancs segítségével kilistáztam őket, mindegyiket egy-egy kártyába:

<mat-card \*ngFor="let candidate of candidateList" class="candidate-card">

            <mat-card-header>

              <div

                mat-card-avatar

                class="avatar"

                [style.backgroundImage]="'url(http://localhost:3000/uploads/' + (encodeURI(candidate.photo)) + ')'"

                [style.backgroundPositionY.px]="candidate.photodisplacement">

              </div>

              <div class="content">

                <mat-card-title>{{ candidate.name }}</mat-card-title>

                <mat-card-subtitle>"{{ candidate.motto }}"</mat-card-subtitle>

                <p class="additional-info">{{ candidate.crimes }}</p>

                <button mat-fab extended (click)="like(candidate.id)" id="like-{{candidate.id}}">

                    <mat-icon  [class.favicon]="likedIds.includes(candidate.id)">favorite</mat-icon>

                    Tetszik

                  </button>

              </div>

            </mat-card-header>

          </mat-card>

# Szabályzat

Az oldalsó menüben (amit a bal felső sarokban levő hamburger ikonra kattintva nyithatunk ki vagy zárhatunk be) található Szabályzat menüpontra kattintva tekinthető meg a nagyon egyszerű és bizonyára mindenki számára érthető szabályzatunk. Ezeknek a szabályoknak a megszegése bizonyos időre, vagy akár örökre szóló kitiltással jár az oldalról, esetenként még a profil törlésével is.

# Partnereink

Partnereinket szintén az oldalsó menüből érhetjük el, a Partnereink menüpontra kattintva. Itt egy táblázatban látjuk a börtönök nevét, címét, és weboldalát. A partnerek szintén a MongoDB adatbázisban vannak tárolva, a következő adatokkal:

|  |  |  |
| --- | --- | --- |
| Adat | Típus | Leírás |
| \_id | ObjectId | A MongoDB által automatikusan adott azonosító |
| id | szám | egyedi azonosító |
| name | szöveg | Az intézmény neve |
| address | szöveg | Az intézmény címe |
| website | szöveg | Az intézmény weboldala |

A partnereket a backend szerverről a következőképpen lehet elérni:

    app.get('/organizations', async (req, res) => {

      try {

        const organizations = await partnersCollection.find({}).toArray();

        res.json(organizations);

      } catch (error) {

        console.error('Error fetching organizations', error);

        res.status(500).send('Internal Server Error');

      }

    });

users.service.ts:

  listPartners(): Observable<Partner[]> {

    return *this*.http.get<Partner[]>('/api/organizations');

  }

# Routing

A hivatkozásokat és az egyes komponensek elérését az Angular routing eszközei (RouterModule) segítségével oldottam meg. A *routes.ts* file-ban vannak konfigurálva az egyes útvonalak, melyek mindegyike a megfelelő komponenshez vezet.

const routeConfig: Routes = [

    {

      path: '',

      component: HomeComponent,

      title: 'BűnTárs'

    },

    {

      path: 'register',

      component: RegisterComponent,

      title: 'Regisztráció - bűnTárs'

    },

A path mező tartalma kerül a böngésző címsorába. Például a regisztrációs oldal megtekintésekor ezt látjuk: [*http://localhost:4200/register*](http://localhost:4200/register).

Ezekre az útvonalakra később könnyen tudtam hivatkozni a megfelelő html fájlokban. Például a menüt így alakítottam ki:

      <ul>

        <li><a [routerLink]="['/candidates']">Jelöltek</a></li>

        <li><a [routerLink]="['/liked']">Kedvencek</a></li>

        <li><a [routerLink]="['/rules']">Szabályzat</a></li>

        <li><a [routerLink]="['/partners']">Partnereink</a></li>

      </ul>