

Webtechnológiák 2

Beadandó - Dokumentáció

Berecz Antónia
LV8GJH

Dátum: 2024. 06. 04.

Képzés: BSc nappali

A beadandó bemutatása

Feladat bemutatása

A feladat alapvetően egy nyilvántartórendszer elkészítése volt, amely lehetőséget biztosít a felhasználó számára, hogy regisztráljon, bejelentkezzen. Bejelentkezés után biztosítani kell a már eltárolt elemek listázásának lehetőségét, valamint hozzáadás és a törlés funkciókat is. A feladat megoldásához az Angular (+Material), a Node.js, és a MongoDB eszközök voltak használhatóak.

A rendszer témája:

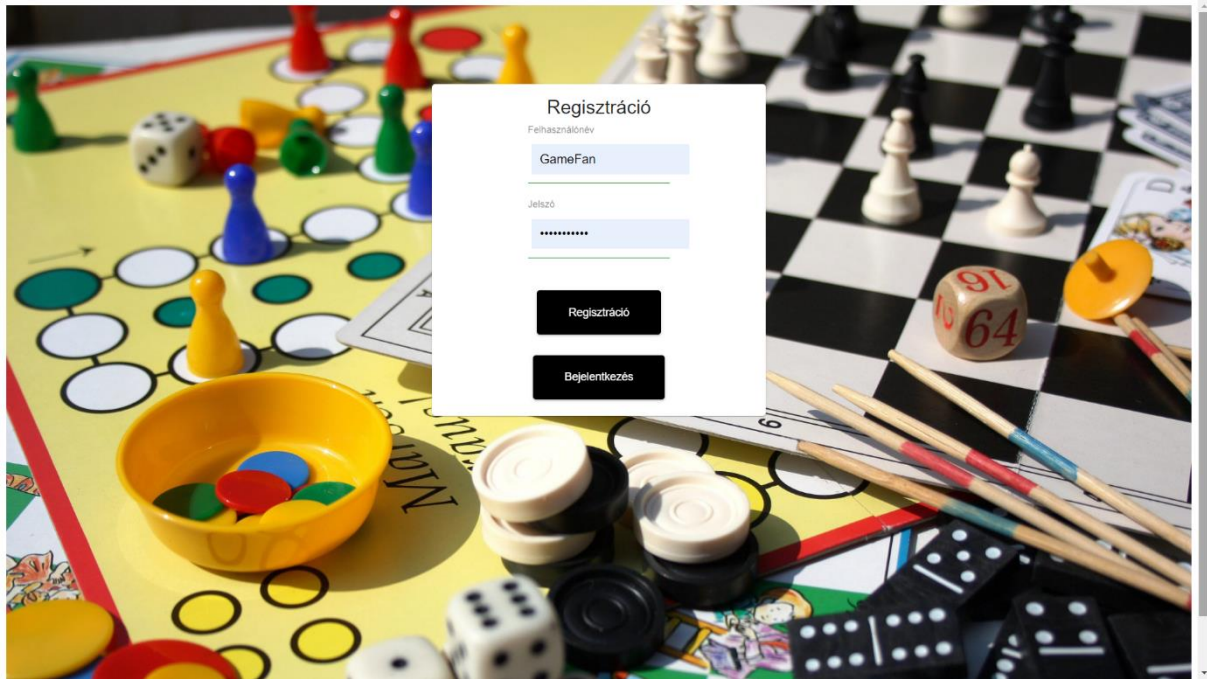
A nyilvántartórendszer témájaként a társasjátékok témakörét választottam, mivel alapvetően szívesen foglalkozom ilyen típusú játékokkal, ráadásul rengetegféle adat létezik róluk, amiket érdemes eltárolni. Hasonló rendszert képzeltem el, mint amilyen az online webáruházak rendelkeznek és azt a célt tartottam szem előtt, hogy amikor egy vásárló szűrőket szeretne alkalmazni, akkor milyen információkat érdemes külön tárolni az egyes játékokról.

Fájlok bemutatása

Az applikáció része alapvetően négy fő egységet tartalmaz. Ezek együttesen felelősek a gördülékeny regisztrációért, a bejelentkezésért, a termékek listázásáért és szerkesztéséért. Mindezt felhasználóbarát módon igyekeztem kivitelezni a program elkészítése során.

Regisztrációs felület

Amikor egy felhasználó először látogat az oldalra, elkészítheti a regisztrációját, amely egy felhasználónevet, valamint egy jelszót tartalmaz. A további alkalmakkor ugyanezekkel az azonosítókkal lesz képes bejelentkezni és hozzáférni a program további funkcióihoz, mint például a társasjáték hozzáadása, listázása, illetve módosítása.



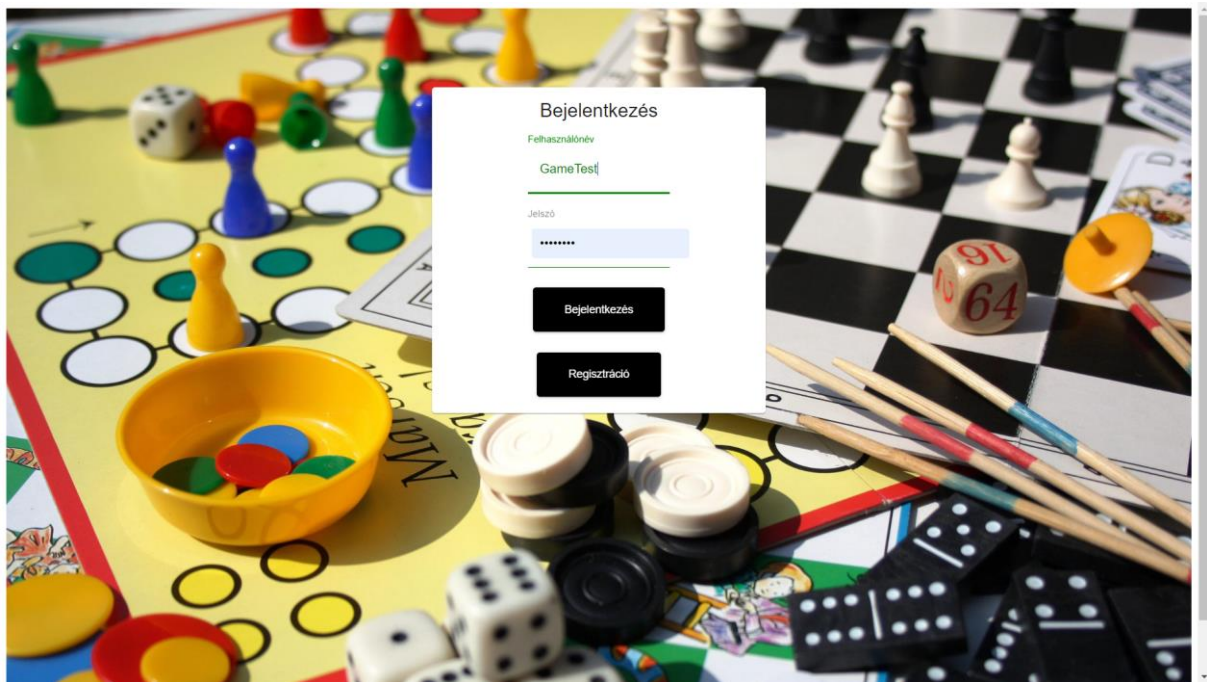
A regisztrációs felület felépítéséért és kinézetéért a `register.component.html`, valamint a `register.component.css` fájlok felelnek, a regisztráció megvalósítását a `register.component.ts` fájl teszi lehetővé, amely eltárolja a megadott adatokat, valamint ellenőrzi a kritériumokat, azaz, hogy a felhasználónév legalább három karakterből, míg a jelszó legalább hat karakterből álljon.

```
mainForm() {
  this.createForm = this.formBuilder.group({
    uname: ['', [Validators.required, Validators.minLength(3)]],
    password: ['', [Validators.required, Validators.minLength(6)]],
  });
}

onSubmit() {
  this.submitted = true;
  if (!this.createForm.valid) {
    alert('Hiba történt a regisztrációkor. Ügyeljen a felhasználónév és a jelszó hosszúságára!');
    return false;
  } else {
    this.appService.createUser(this.createForm.value).subscribe(
      (res) => {
        alert('A regisztráció sikeres.');
```

Belépés

A következő egység, amivel a felhasználó találkozik, az a beléptető rendszer, amely ellenőrzi, hogy a belépéshez megadott adatok szerepelnek-e a rendszerben, vagyis hogy a felhasználó jogosult-e a belépésre. Amennyiben igen, úgy a belépés megtörténik és a felhasználó számára elérhetővé válik minden további funkció.

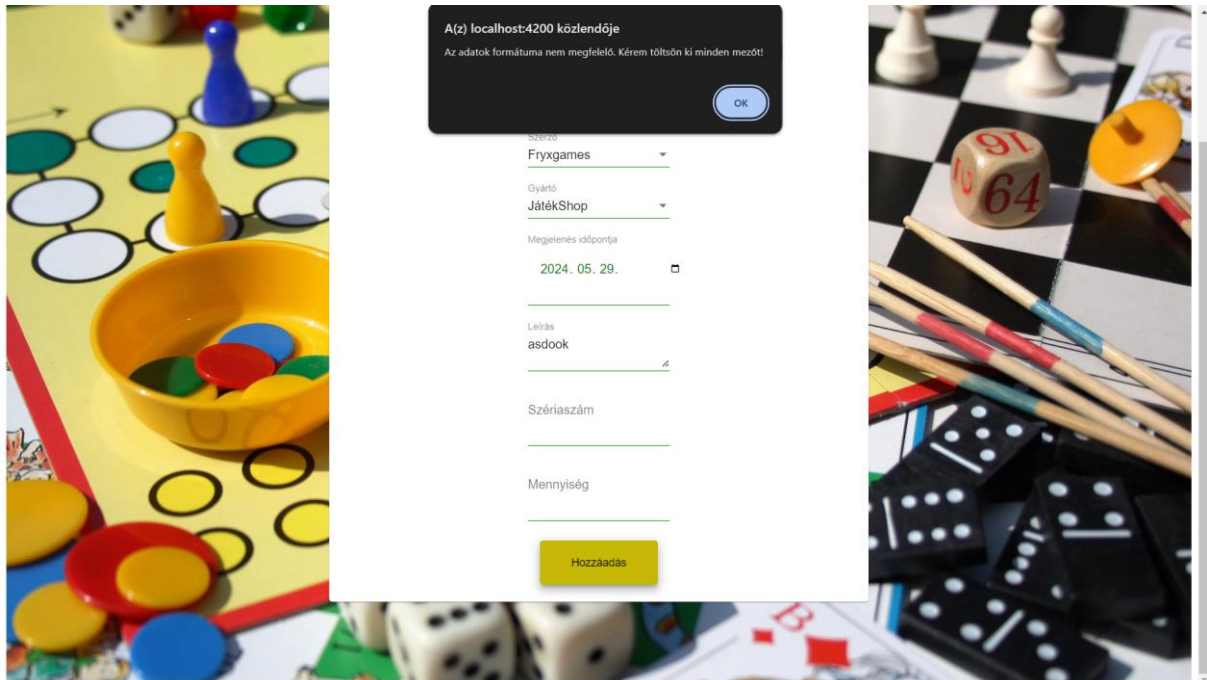


A belépési felület felépítéséért és kinézetéért a login.component.html, valamint a login.component.css fájlok felelnek, a belépés megvalósítását a login.component.ts fájl teszi lehetővé, amely ellenőrzi, hogy az adott felhasználó rendelkezik-e regisztrációval.

```
onSubmit() {  
  const loggedInUser = (JSON.stringify(this.loginForm.value.username));  
  const loggedInPassword = (JSON.stringify(this.loginForm.value.password));  
  
  let match = false;  
  this.authService.getUsers().subscribe((data) => {  
    this.users = data;  
  
    for (const user of this.users) {  
      if (JSON.stringify(user.username) === loggedInUser && loggedInPassword === JSON.stringify(user.password)) {  
        match = true;  
        this.authService.setLoggedInUser(user);  
      }  
    }  
  
    if (match) {  
      this.router.navigate(['/add']);  
    }  
    else {  
      alert('Sikertelen bejelentkezés! Hibás a felhasználónév vagy a jelszó.');    }  
  });  
}
```

Társasjáték bejegyzése

A következő nagy egység a társasjátékok adatbázisba történő felvezetéséért felel. Itt megadhatjuk a társasjáték nevét, azonosítóját (amely ellenőrzöten egy egyedi érték), szerzőjét, gyártóját, a megjelenés dátumát, a játék szériaszámát, illetve a mennyiséget is, amely elérhető belőle a raktáron. Ezt az értéket a későbbiekben természetesen módosíthatjuk szükség esetén.



The screenshot shows a web form for adding a board game. The form is overlaid on a background image of various board games, including a yellow board with blue and red pawns, a chessboard with white pawns, and a domino set. The form has a dark header with a message: "A(z) localhost:4200 közlendője Az adatok formátuma nem megfelelő. Kérem töltsön ki minden mezőt!" and an "OK" button. The form fields are: "Szérió" (Fryxgames), "Gyártó" (JátékShop), "Megjelenés időpontja" (2024. 05. 29.), "Leírás" (asdoók), "Szériaszám", and "Mennyiség". A yellow "Hozzáadás" button is at the bottom.

A bejegyző felület felépítéséért és kinézetéért az `add.component.html`, valamint az `add.component.css` fájlok felelnek, a bejegyzés megvalósítását az `add.component.ts` fájl teszi lehetővé, amely ellenőrzi, hogy a megadott információk megfelelnek-e a várt adattípusnak, illetve szükség esetén azt is megvizsgálja, hogy egyedi-e az érték.

```
onSubmit() {
  this.submitted = true;
  const gameId = this.createForm.get('gameID').value;


  if (!this.createForm.valid) {
    alert('Az adatok formátuma nem megfelelő. Kérem töltsön ki minden mezőt!');
    return false;
  }

  this.appService.checkID(gameID).subscribe(
    (isUnique) => {
      if (!isUnique) {
        alert('Kérem, egyedi azonosítót adjon meg!');
        return false;
      } else {
        this.appService.createGame(this.createForm.value).subscribe(
          (res) => {
            alert('Játék hozzáadva');
            this.ngZone.run(() => this.router.navigateByUrl('/list'));
          },
          (error) => {
            alert('Kérem, egyedi azonosítót adjon meg!');
            return false;
          }
        );
      }
    },
    (error) => {
      alert('Hiba: ' + error);
      return false;
    }
  );
}
```

Áttekintés

Az utolsó lényeges felület, amely a felhasználó számára elérhető, a raktáron lévő társasjátékok adatai, valamint a darabszámok táblázatba szedve. A táblázat interaktív, szükség esetén a felhasználó módosíthatja az egyes játékok mennyiségét.

Játékok

Játék neve	Azonosító	Szerző	Gyártó	Megjelenés dátuma	Bemutató	Szériaszám	Darabszám	Törlés
Carcassonne	846531	ZmanGames	JátékShop	2009-06-20	Szuper	15000	— 32 +	
Mars	641565	Fryxgames	ReflexShop	2020-01-09	Király	17000	— 15 +	
TerraMystica	6541365	Feuerland	RegioJatek	2016-02-13	Varázslatos	20000	— 6 +	
RuneStones	9648513	QueenGames	JátékShop	2018-07-21	Mágikus	14000	— 15 +	
Builders	8645133	ZmanGames	JátékShop	2022-09-20	Játékos	12000	— 19 +	

A táblázat felépítéséért és kinézetéért a `list.component.html`, valamint a `list.component.css` fájlok felelnek, a lista elkészítéséért és megvalósítását az `list.component.ts` fájl teszi lehetővé, amely a megadott adatokat kilistázza a felhasználó számára.

```
<table class="mat-elevation-z8">
  <thead>
    <tr>
      <th>Játék neve</th>
      <th>Játék azonosítója</th>
      <th>Szerző</th>
      <th>Gyártó</th>
      <th>Megjelenés dátuma</th>
      <th>Leírás</th>
      <th>Szériaszám</th>
      <th>Darabszám</th>
      <th>Törlés</th>
    </tr>
  </thead>
  <tr *ngFor="let game of Game; let i = index">
    <td>{{game.name}}</td>
    <td>{{game.gameID}}</td>
    <td>{{game.author}}</td>
    <td>{{game.company}}</td>
    <td>{{game.releaseDate | date:'yyyy-MM-dd'}}</td>
    <td>{{game.introduction}}</td>
    <td>{{game.seriesNumber}}</td>
    <td>
```

db.js

A `db.js` fájl az adatbázis kezeléséért felelős, és egy szerver segítségével kommunikál az adatbázissal. Ebben a fájlban definiáljuk a `User` és a `Game` modelleket. Először a szerver csatlakozik az adatbázishoz. HTTP kérések érkezésekor végrehajtja a megfelelő műveleteket.

Alkalmazás futtatása:

1. Nyissunk meg egy CMD-t!
2. A cd parancs segítségével navigáljunk el a projektet tartalmazó mappába (assignment)!
3. Futtassuk a következő parancsot: npm install
4. Futtassuk a következő parancsot is: npm run start:new
5. Nyissunk meg egy újabb CMD-t!
6. Ismét navigáljunk el a projekt mappájába (assignment)!
7. Futtassuk a következő parancsot: node db.js
8. Nyissunk meg egy böngészőt!
9. Írjuk be a következő URL-t: <http://localhost:4200/login>