# **ZÁRÓDOLGOZAT**

# **Eke Nikolett**

**Nyakó Tamás  
Szabó Richárd**

# **Budapest**

2022

# **BUDAPESTI KOMPLEX SZC WEISS MANFRÉD SZAKGIMNÁZIUMA, SZAKKÖZÉPISKOLÁJA ÉS KOLLÉGIUMA**

**Halloweeni Weboldal**

Készítette:

Témavezető:  
Kovács László  
  
Eke Nikolett  
Nyakó Tamás  
Szabó Richárd

# **Budapest**

2022

Tartalomjegyzék:

[Bevezetés 4](#_Toc102500612)

[Záródolgozatom témája: 4](#_Toc102500613)

[A weboldal témája 4](#_Toc102500614)

[Funkció: 5](#_Toc102500615)

[Célközönségünk: 5](#_Toc102500616)

[Milyen programot tart a felhasználó a kezében?: 6](#_Toc102500617)

[Fejlesztői Dokumentáció 7](#_Toc102500618)

[JAVA, SpringBoot alkalmazás 7](#_Toc102500619)

[User controller 7](#_Toc102500620)

[User model 7](#_Toc102500621)

[DAO 8](#_Toc102500622)

[UserDaoJDBC 8](#_Toc102500623)

[EmailSender 8](#_Toc102500624)

[DemoApplication 9](#_Toc102500625)

[POM 9](#_Toc102500626)

[Application properties 9](#_Toc102500627)

[Legfőbb JavaScript funkciók 10](#_Toc102500628)

[navMenu 10](#_Toc102500629)

[Swiper 10](#_Toc102500630)

[Scrolling 11](#_Toc102500631)

[SignUpButton 11](#_Toc102500632)

[Felhasználói dokumentáció: 12](#_Toc102500633)

[Szükséges hardver-szoftver: 12](#_Toc102500634)

[A program használata és felépítése : 12](#_Toc102500635)

[Navigáció: 12](#_Toc102500636)

[Oldal felépítése: 12](#_Toc102500637)

[A csapatról 13](#_Toc102500638)

[Szövegbeviteli mezők: 14](#_Toc102500639)

[A program hasznosulása: 14](#_Toc102500640)

[Az elkészült munka értékelése: 14](#_Toc102500641)

[Források: 15](#_Toc102500642)

Bevezetés

Záródolgozatom témája:

A záródolgozat témája egy reszponzív felépítésű weboldal, ami tartalmaz:  
 -HTML

-CSS

-SCSS

-JavaScript

-Java

-springboot

-postgreSQL

Kiválóan működő képes telefonon, Tableten és pc-n egyaránt.

A weboldal témája

Azért választottuk ezt a témát, mert úgy gondoltuk, hogy itthon Magyarországon, nem kap kellő figyelmet, a Halloween, mint ünnep. Amerikában ezen az ünnepségen, beöltöznek a gyermekek, ijesztő maskarákba bújnak ,és házról házra járnak. Ijesztgetik egymást, nevetgélnek és nem utolsó sorban boldogok.

Nagyon sok ember, szereti rémisztővé varázsolni otthonát, hogy elkápráztassa az odalátogatókat.

Úgy érezzük, hogy itthon sajnálatos módon nincs meg, a kellő odafigyelés, és a kellően rémisztő hangulat.

Ebben szeretnénk segíteni, és szív ügyünknek érezzük azt, hogy vérfagyasztóvá tegyük, néhány ember otthonát, noha még csak egy éjszakára is.

Ennek megvalósítása érdekében jött létra a weboldalunk!

Ami rengeteg, kiegészítőt és még egy kis nyalánkságot is tartalmaz.

Kezdve a: -szellemekkel kísértetekkel

-a rémísztő tökökkel töklámpásokkal

-boszorkány sapkákkal

- boszorkány seprűkkel

-csontokkal

-madárijesztőkkel

-kísértetházakkal

-kristálygömbökkel

-Halloween témájú gyertyákkal

-és nem utolsó sorban nyalánkságokkal

Funkció:

Egy rettentően egyszerű, és felhasználó barát weboldalt szerettünk volna elkészíteni, amit mindenki pehely könnyedséggel tud kezelni.

Ami mindenkinek egyből leköti a figyelmét, és megragadja a neten szörfölő Halloweeni közösséget, akik le sem tudják majd venni a szemüket a weboldalról.

Célközönségünk:

A célközönségünk a Halloween ünnep rajongói lennének legfőképp, de természetesen szeretnénk, ha minél több ember figyelmét felkeltené, és fokozatosan bővülne az érdeklődés a téma iránt.

Milyen programot tart a felhasználó a kezében?:

A felhasználó egy olyan webes felületet tart a kezében, ahol Hallowen témájú kiegészítőket, díszítő, rémísztő elemeket tud megvásárolni, olyanokat, amilyeneket csak szeretne.

Fejlesztői Dokumentáció

JAVA, SpringBoot alkalmazás

User controller

A kód a @RestController annotációval kezdődik. Ez egy Spring Boot annotáció, ami arra kéri a Springet, hogy készítsen http kontroller osztályt ehhez az osztályhoz. Ezt követően importálásra kerül a UserDaoJDBC és EmailSender osztályok, amelyek részei a com.myown.demo package-nek valamint az org.springframework.beans.factory.annotation-nek. A következő sor instance-t készít a UserDaoJDBC-ből Autowired használatával és tovább adja a controller osztály konstruktorának. Ezt követően két @Autowired annotációnk van az adatbázisból származó adatok elérésehez használt módszerekhez: az egyik az email küldéshez (emailSender) a másik pedig az új felhasználók hozzáadásához (user). Ezek a módszerek JDBC annotációkat használnak az adatbázisunkhoz való csatlakozáshoz és lekérdezések vagy frissítések révén lekérik onnan az adatokat: @Autowired private EmailSender emailSender; @PostMapping("/add") public String addUser(@RequestBody User user){ ... }

A kód egy példa egy vezérlőosztályra, amely a @RestController segítségével lett megjelölve. Ez a megjegyzés arra utasítja a Springet, hogy hozzon létre egy RESTful webszolgáltatást az adott osztályhoz. A fenti kód megjegyzéseket is tartalmaz, amelyek a RESTful webszolgáltatás konfigurálására szolgálnak. Ezek a megjegyzések a következőket tartalmazzák: @CrossOrigin, amely megadja, hogy ez a vezérlő bármely forrásból elérhető, és nem igényel hitelesítést, valamint a @RequestMapping, amely meghatározza, hogy a vezérlőosztály mely metódusaihoz milyen HTTP-metódusok legyenek leképezve.

User model

A kódrészlet egy osztály, amelynek két metódusa van, amelyek közül az egyik egy getter, a másik pedig egy setter. A getter metódus a felhasználói objektum e-mail mezőjét adja vissza. A setter metódus az e-mail mezőt valami másra állítja be. A kód egy import utasítással kezdődik a lombok számára, amely lehetővé teszi a Java néhány hasznos funkciójának használatát. Ezután van egy @Getter annotáció a 3. sorban, majd egy @Setter annotáció a 4. sorban; ezek az annotációk megmondják Lomboknak, hogy mit kell tenniük, amikor végrehajtásra kerülnek (getter értéket ad vissza, a setter pedig megváltoztatja azokat). Ezek után két privát mező következik: email az 5. sorban és név a 6. sorban; mindkettő véglegesnek van deklarálva, így senki sem módosíthatja őket a deklaráció után anélkül, hogy ne módosítaná a típusukat String-ről UserName-re (ami megszakítaná a beágyazást).

DAO

A kód egy Dao interfész, amely módszereket tartalmaz az adatok lekérésére, hozzáadására, törlésére és frissítésére. A get metódus bemenetként egy azonosítót vesz fel, és visszaadja a megfelelő elemet az adatbázisból. Az add metódus új elemeket ad hozzá az adatbázishoz. A kód a következőket hivatott bemutatni: Egy adattár interfész, amely egyetlen módszert biztosít az összes CRUD művelethez.  
A getAll() metódus a lerakatban lévő összes elem listáját adja vissza.  
A deleteAll() metódus törli az összes elemet a tárolóból.  
Az addAll() metódus több elemet ad a lista végére, és így tovább.

UserDaoJDBC

A kód egy osztály, amely implementálja a Dao interfészt. A get metódus meghívása egy objektum adatbázisból való lekérésére és visszaadására szolgál. A runQuery hozza létre a lokális adatbáziskapcsolatot, amely ha sikeres volt egy „200 ok” stringgel tér vissza, míg ha hibára fut hibaüzenetet kapunk. A kódba két funkció került megírásra, az add, amely eltárolja a feliratkozó email címét, a deleteAll funkcióval pedig üríteni lehet az adatbázist.

EmailSender

A kódrészlet egy osztály, amely a @Component megjegyzéssel van ellátva, tehát egy alkalmazás összetevőjeként fogjuk használni. Az EmailSender osztály két privát mezővel rendelkezik: az egyik a naplózásra, a másik pedig az e-mail adatforrás tárolására szolgál. Tartalmaz egy konstruktort is, amely három paramétert vesz fel: a naplózót, az e-mailek küldéséhez használt modellobjektumot és egy EmailException kezelőt. A konstruktor létrehozza a HtmlEmail új példányát az argumentumok átadásával (a naplózó, a modellobjektum az e-mailek küldéséhez és a kivételkezelő). Ezután beállít néhány tulajdonságot ezen az e-mailen, például a tárgysort és a törzsszöveget.

Létrehoztunk egy új HtmlEmail objektumot, és a hostot az „smtp.gmail.com” értékre állítottuk be 587-es porttal, amely a Gmail alapértelmezett SMTP-portja. Az SSL-t is igazra állítottuk, így a Gmail szervereihez való csatlakozáskor HTTP helyett HTTPS-t használ. Az emailBody változó az e-mail törzsét HTML formátumban tárolja, míg a tárgy az e-mail tárgysorát egyszerű szöveges formátumban.

A sendEmail() függvény úgy küld e-mailt, hogy létrehoz egy új HtmlEmail objektumot, majd meghívja a setHostName(), setSmtpPort(), setSSLOnConnect() metódusait a megfelelő értékekkel: HtmlEmail email = new HtmlEmail(); email.setHostName("smtp.gmail.com"); email .setSmtpPort(587); email .setSSLOnConnect(true). Ezután a hitelesítést "halloweenproject01" és "HWproject1" értékre állítja, majd létrehoz egy e-mail törzset egy tárgy és html üzenettel. Végül elküldi az e-mailt a send() metódussal az email osztály újonnan létrehozott példányán.

A createAndSendEmail függvény ezután beállítja azt az e-mail címet, ahonnan ez az e-mail küldésre kerüljön, mint "halloweenproject01@gmail.com". Ezután hozzáadja a felhasználó e-mail címét, hogy a jövőben e-maileket küldhessen neki. Végül ez a módszer egy e-mailt küld az adott felhasználónak.

DemoApplication

A kód egy Spring Boot alkalmazás, amely a parancssorból fut. Van egy DemoApplication nevű osztálya, amely megvalósítja a CommandLineRunner-t. A fő metódus a DemoApplication osztály kódját futtatja, amely ezután végrehajtja az összes metódusát.

Az afterInit() metódus automatikusan meghívásra kerül, amikor az alkalmazás egy példánya első alkalommal indul el. A kód main() metódusa elindítja a Spring Boot alkalmazást.

POM

A POM a Maven alapvető munkaegysége. Ez egy XML-fájl, amely információkat tartalmaz a projektről és a Maven által a projekt felépítéséhez használt konfigurációs részletekről. Feladat vagy cél végrehajtásakor a Maven az aktuális könyvtárban keresi a POM-ot. Beolvassa a POM-ot, megkapja a szükséges konfigurációs információkat, majd végrehajtja a célt.

Application properties

A spring.datasource tulajdonság a jdbc:postgresql://localhost:5432/postgres értékre van állítva, amely az 5432-es porttal rendelkező localhost PostgreSQL-kiszolgáló adatbázis-kapcsolatának részleteit adja meg. A következő kódsor beállít egy felhasználónevet és jelszót az adatbázishoz.

Legfőbb JavaScript funkciók

navMenu

Elsőként a navMenu nevű változót hoztuk létre. Ez az az elem, amely a menü megjelenítésére és elrejtésére szolgál. A kód ezután létrehoz egy eseményfigyelőt, amikor a felhasználó a "nav-toggle" gombra kattint, amely hozzáad egy linkAction() nevű eseményfigyelő függvényt, amely elrejti vagy megjeleníti a menüt attól függően, hogy éppen megjelenik-e vagy sem.

A következő kódrészlet: const navLink = document.querySelectorAll('.nav\_\_link') az összes (class="nav\_\_link") hivatkozást megtalálja. Ezek mind a navigációs sávunkban található hivatkozások, amelyeket a .nav\_\_link--blue és .nav\_\_link--red osztályú címkék használatával hoztunk létre. Ezután mindegyik hivatkozáson végigfut, és eseményfigyelőt ad hozzá, hogy amikor rájuk kattintanak, a színüktől függően mást tehetünk (ebben az esetben módosíthatjuk a háttérszínüket).

A következő függvény (navClose) egy kattintásra hallgató eseményfigyelőt hív meg, amely eltávolítja a show-menu osztályt a navMenu-ból, amikor a felhasználó bármelyik hivatkozásra kattint.

Swiper

Létrehoztunk egy új Swiper objektumot "home-swiper" osztálynévvel, amely átad egy elemválasztót a tárolóként használt DOM csomóponthoz. A következő sor létrehozza a newSwiper nevű változót, amely a Swiper új példányához van hozzárendelve.

Ezt a példányt a következő beállításokkal hoztuk létre: a centeredSlides értéke igaz, a slidesPerView beállítása "auto", a hurok beállítása "true", a spaceBetween 16 képpontra van állítva, a lapozásEl pedig ".swiper-pagination"-ként van definiálva.

A kódba ezután egy eseményfigyelőt csatoltunk az ablakgörgetési eseményhez, így amikor felfelé vagy lefelé görgetünk, meghívja a scrollHeader() függvényt. Ebben a funkcióban a classList fejléc hozzáadódik vagy eltávolításra kerül attól függően, hogy a felhasználó aktuális pozíciója meghaladja-e a nézetablak magasságának 50%-át vagy sem.

A kódban létrehoznktu egy új Swiper-t, és hozzárendeltük az „új-swiper” azonosítóját. Ezután a slidesPerView tulajdonságot "auto"-ra állítjuk, ami azt jelenti, hogy nem lesz beállított számú dia nézetenként.

Scrolling

A következő részlet a dokumentum összes szakaszának lekérésével kezdődik. Ezután végigfut az egyes szakaszokon, és kiszámítja a magasságot, a felső eltolást és az id attribútumot. Ezután, ha a scrollY nagyobb vagy egyenlő, mint egy szakasz felső eltolása plusz a magassága, akkor az a szakasz aktív.

A kód a window.pageYOffset segítségével számítja ki, hogy éppen hol görgetünk a képernyőn, hogy összehasonlíthassa ezt az értéket egy szakasz felső eltolása + magassága (sectionTop) értékével.

Ha egyeznek, akkor ez azt jelenti, hogy a dokumentum egy aktív részébe görgetünk, és addig folytatjuk a hurkot, amíg el nem érjük azt, amelyre nem igazak a feltételek. A kód végigfut a dokumentum összes szakaszán, és ellenőrzi, hogy a scrollY nagyobb-e vagy egyenlő-e a sectionTop értékkel. Ha igen, akkor a sectionId a szakasz azonosítója lesz.

A kód azt is ellenőrzi, hogy a scrollY kisebb-e vagy egyenlő-e a sectionTop + sectionHeight értékkel. Amennyiben egyenlő, akkor a sectionId értéket nullára állítja.

A kódban egy ScrollReveal objektumot hoztunk létre. Az origó a tetejére van állítva, a távolság 60 képpont, az időtartam pedig 2500 ezredmásodperc. A késleltetés 400 ezredmásodperc. A következő kódsor eseményfigyelőt hoz létre az ablak görgetési eseményeihez, amely meghívja a scrollUp nevű függvényt, amikor felfelé görgetés történik. A funkció 2500 ezredmásodpercenként 60 képponttal feljebb görgeti az oldalt.

SignUpButton

Készítettünk egy gombot a megfelelő signup ID-val. A kódban ezután létrehoztunk egy eseményfigyelőt, amikor a felhasználó rákattint erre a gombra, meghív egy függvényt, amely fogad egy e-mail címet, és elküldi a http://localhost:8080/user/add címre.

A kódba ezután létrehoztuk az xhr nevű változót, amely egy XMLHttpRequest objektum létrehozására szolgál. Ennek az objektumnak két tulajdonsága van: open és setRequestHeader.

Az open tulajdonság arra szolgál, hogy adatokat küldjön egy helyről (ebben az esetben a localhost-ról) egy másik helyre (jelen esetben a 8080-as portra).

A setRequestHeader tulajdonság annak meghatározására szolgál, hogy milyen típusú tartalmat kell elküldeni a kérés fejlécen keresztül; ebben az esetben a JSON-adatokat a kérelem fejlécén keresztül kell elküldeni alkalmazás/json néven.

Felhasználói dokumentáció:

Szükséges hardver-szoftver:

A weboldal használatához nincs szükség másra csak egy számítógépre vagy telefonra illetve tabletre.

Ez egy interneten fellelhető weboldal amit bármelyik eszközön meg lehet találni amelyiken van internet lehetőség.

Én személy szerint a Google Chrome-ot használtam a készítés közben, de Microsoft Edgen is hiba nélkül működik.

A felhasználó csak elindítja az általa kedvelt böngészőjét, beírja a keresőbe az oldal nevét, vagy a rá vonatkozó tulajdonságokat és kiadja az oldalunkat.

(Jelen esetben még csak mappából a html dokumentummal van lehetőség az indítására.)

A program használata és felépítése :

Navigáció:

A program használata és felépítése roppant egyszerű. Az oldal tetején található a navigációs sáv. Rendre a főoldal, rólunk, cukorkák, újdonságok.

Bármelyikre kattintva, az adott jelzőhöz kapcsolódó oldalrészhez fog ugrani az oldal.

A navigációs sáv tartalmaz még egy bejelentkezés gombot, amit aktiválva megnyílik egy felugró oldal.

Itt van lehetőség a már regisztrált használóknak bejelentkezniük, valamint az újdonsült vásárlóknak akár regisztrálni is lesz lehetőségük.

Oldal felépítése:

Az első "poszt" az oldalon egy bemutató sáv a 3 legijesztőbb szellemről amit lehet mozgatni akárcsak úgy ha rányom a felhasználó a szövegre vagy a képre és elhúzza valamelyik irányba.

Lejjebb görgetve 3 kategóriát láthatunk, amiknél ha rávisszük az egeret a képekre, azok kicsit megemelkedve jelzik hogy ők vannak most a középpontban.

Ezután a halloweenról olvashatunk pár szót, hogy mit is takar ez az ünnep. Ide a "rólunk" navigációs gombbal is eljuthatunk.

Alább a "Csokit vagy csalok" elhíresült kéréssel egyetemben találhatunk pár cukorkát és kiegészítőket, a fajtájukkal és az árukkal együtt.

A kategóriáknál alkalmazott felugrás itt is használva lett, kiegészítve egy kis kosár gombbal.

Még ennél is lejjebb 2 denevér hívja fel a figyelmet a kedvezményes új termékekre, a gombra kattintva pedig oda is navigál minket a tőle nem messze lévő tartozékokhoz.

Egyedi megjelenésük abban rejlik, hogy a felugró kiemelésük és a kiskosár megjelenése mellett megkapták a bemutató sáv egérrel mozgatható sajátosságát.

Ide akár az "újdonságok" menü ponttal is eljuthatunk könnyedén.

Az oldal utolsó bejegyzése egy hírlevél feliratkozás. Ehhez a felhasználónak nincs szüksége regisztrálni/bejelentkezni elég csak megadnia az e-mail címét.

Az e-mail cím beírása és az "Iratkozz fel" gomb megnyomása után a felhasználó kapni fog egy visszaigazoló e-mailt.

A csapatról

Mint a legtöbb oldalnál itt is fellelhetőek legalul az elérhetőségeink, megtalálható a csapatunk tagjai a facebookon, instagrammon és twitteren,

ehhez csak valamelyik ikonra rá kell kattintani és be fogja adni az egyikünk elérhetőségét. Mindegyikhez csak 1 tag van megjelölve.

Továbbá a szolgáltatásainkról, kis "cégünkről" és elsősorban rólunk a készítőkről is lehet majd tájékoztatást kapni.

Nem mehetünk el amellett hogy itt két kísértet tartja szemmel mit csinál a felhasználó és hova nyúlkál, de ha őket akarjuk megvizsgálni már kapják is fel magukat,

a már ismert felugró kiemeléssel.

Nem utolsó sorban pedig van egy gyöngyszem gombunk, aminek hála nem kell feltekerni a legtetejére. Elég ha csak rá nyomunk és újra a bemutatott szellemeinkkel találjuk szembe magunkat.

Szövegbeviteli mezők:

A bejelentkezésnél az e-mailnél egy példát láthat az ide látogató hogy hogyan kell kinéznie az elvárt szövegnek.

Az előtagja a felhasználó képzeletére van szabva, de csak ékezet nélkül. Azt követi egy @ jel, majd a hivatalos szolgáltatók egyike: hotmail, citromail, freemail, outlook, gmail.

A jelszónál annyiban komplikáltabb a helyzet, hogy a beviteli mező fölött kell tartani az egeret, sajnos ez telefonokon nem működik.

A jelszónak tartalmaznia kell egy nagy- illetve egy kis betűt, számot és legalább 8 karakterből kell hogy álljon.

A feliratkozásnál ugyanaz van érvényben mint a bejelentkezési e-mailnél.

A program hasznosulása:

A programot csiszolgatva még igen sok haszna válhat, hiszen nagyon sok ember szereti a halloweeni hangulatot, dacára annak hogy itthon nem illik tartani.

Azonban minél több ember veszi a bátorságot az ünnepnek a hangsúlyozásához, annál többen és többen behetik példájukat.

Az elkészült munka értékelése:

A csapat munka rendkívül zökkenőmentesen zajlott, minden csapattag a lehető legjobb módon próbált igazodni a tárasaihoz. Minden nagyobb volumenű lépést megbeszéltünk, rengeteg segítséget kértünk egymástól, illetve nagyon sok segítséghez folyamodtunk az internet világában. Különböző website-ok fórumok olvasása, illetve videók formájában.  
Végeredményképpp, egy nagyon esztétikus, és működő képes munkát tudtunk kiadni a kezeink közül.

Források:

A home1-2-3 mon található szöveg némi kreativitással és ennek az oldalnak a segítségével jött össze:

<http://afantasyfoldje.qwqw.hu/?modul=oldal&tartalom=1098703>

animációk css: <https://github.com/atisawd/boxicons/blob/master/css/animations.css>

boxicons css: https://github.com/atisawd/boxicons/blob/master/css/boxicons.css

boxicons: <https://boxicons.com/?query=x>

hsl szín konverter: <https://cssgradient.io/?fbclid=IwAR1BSRetyP9y9_oSZnPRJ96E0NgTOTLPmPgNl_ap85zjDGVliRajKyvR_7g>

jobbra nyíl: <https://boxicons.com/?query=arrow>

swiper js : <https://swiperjs.com/get-started>

<https://unpkg.com/browse/swiper@8.1.4/>

<https://unpkg.com/browse/swiper@8.1.4/swiper-bundle.min.js>

sqiper min.css: <https://unpkg.com/browse/swiper@8.1.4/swiper-bundle.min.css>

slide home: <https://codesandbox.io/s/xlzq5j?file=/index.html>

kosár: <https://boxicons.com/?query=cart>

facebook kicsi ikon: <https://boxicons.com/?query=face>

instagramm kicsi ikon: <https://boxicons.com/?query=instagram>

twtiier kicsi ikon: <https://boxicons.com/?query=twitter>

felfelé mutató nyíl az oldal alján : <https://boxicons.com/?query=arrow>

scroll reveal: <https://github.com/jlmakes/scrollreveal/blob/master/dist/scrollreveal.min.js>

Képek (elnézést ha nem tudunk minden képnek az URL-jét megadni)

Tökök: <https://pngtree.com/free-png-vectors/jack-o-lantern>

boszorkány sapka (1) <https://pngtree.com/freepng/halloween-terror-witch-hat_3838926.html>

boszorkány sapka (2) <https://pngtree.com/freepng/happy-halloween-witch-hat-broom_5321425.html>

nyalókák: <https://pngtree.com/freepng/lollipop-halloween-candy_7648931.html>

üst: <https://pngtree.com/freepng/halloween-cauldron-on-a-transparent-background-3d-illustration_5386423.html>

szellemek: <https://pngtree.com/so/ghost>

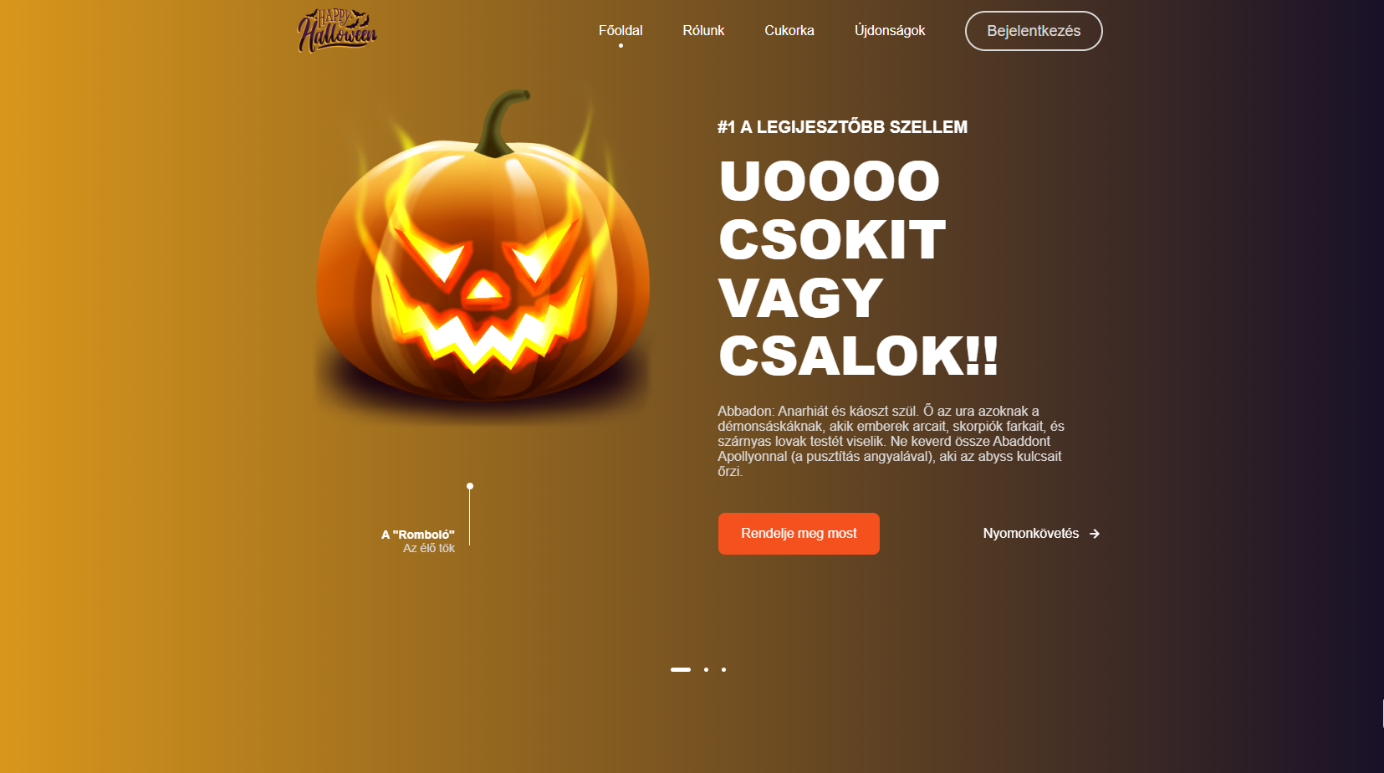
halloween hold: <https://www.pikpng.com/freepng/halloween-moon/>

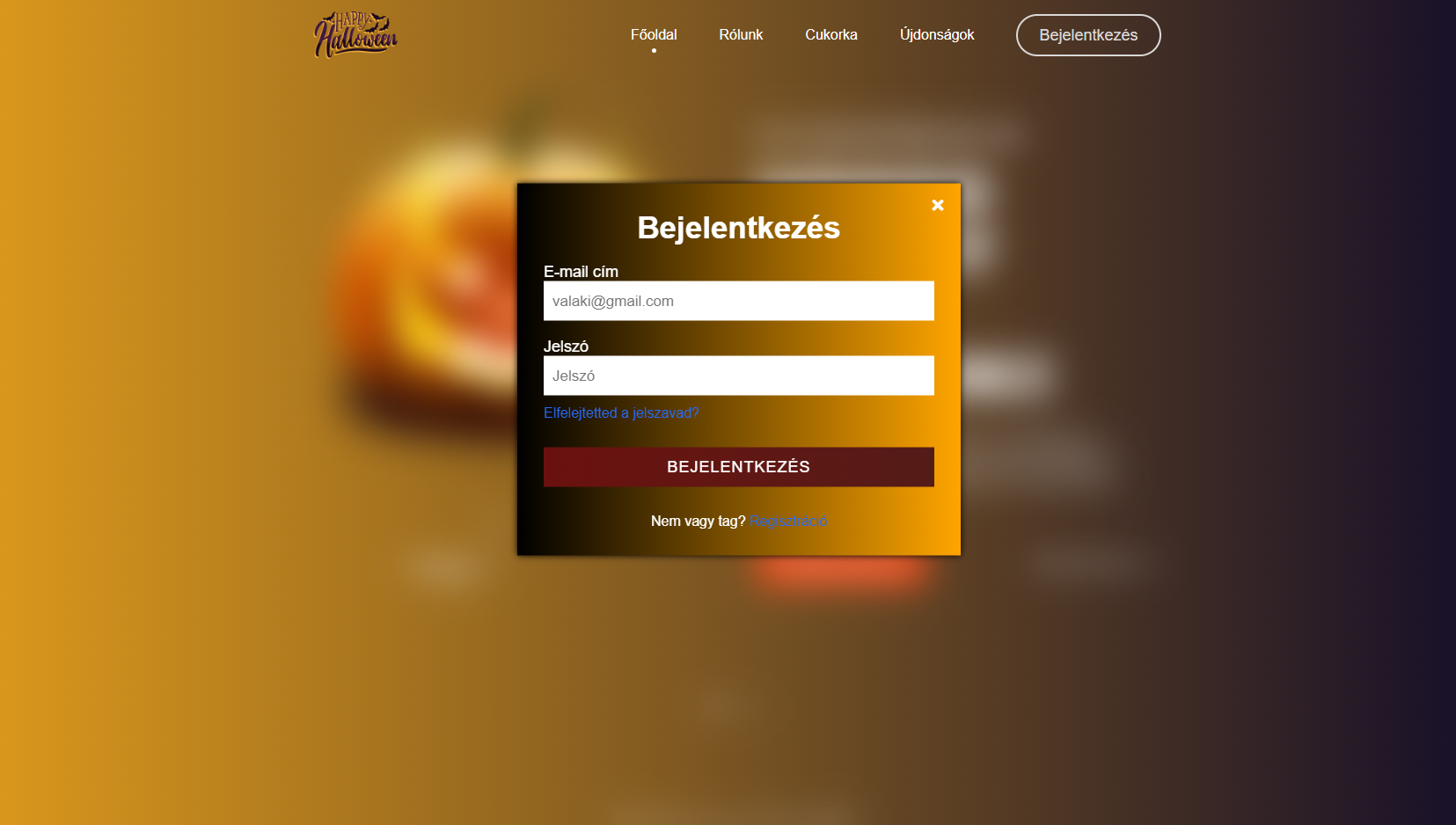
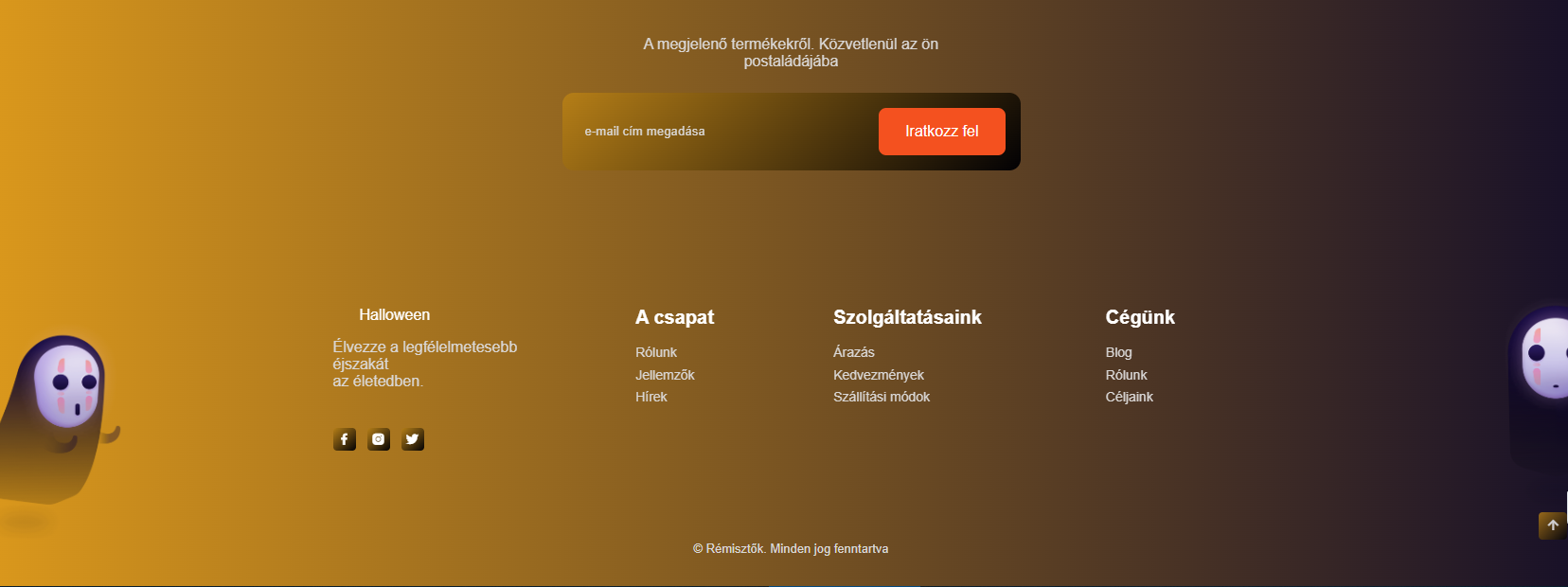
karamell: <https://www.pikpng.com/pngvi/iiRoohR_cartoon-candy-sweet-clipart-images-of-toffees-png/>

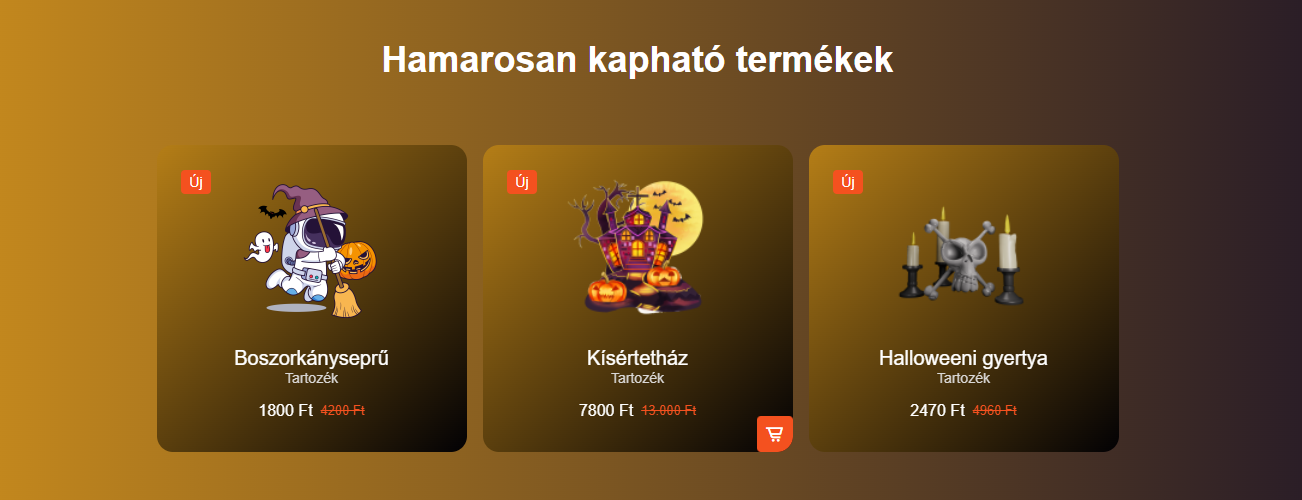
madárijesztő: <https://www.pikpng.com/search.html?k=+scarecrow>

kísértetház: <https://pngtree.com/freepng/halloween-haunted-house-pumpkin-bat-dead-tree_5767751.html>

boszorkány seprű: <https://pngtree.com/freepng/little-ghost-little-witch-astronaut-halloween_6513258.html>

Függelék  






Köszönetnyilvánítás

Első sorban szeretnénk megköszönni szaktanárainknak név szerint Bak Zsuzsanna tanárnőnek, Monok Judit tanárnőnek, Vonyigás István tanárúrnak, Kreisz Róbert tanárúrnak, Fodor Péter tanárúrnak, Kovács László tanárúrnak a rengeteg segítséget és legfőképp a türelmüket. Nélkülük nem jöhetett volna létre ez a vizsgaremek.

Nem utolsó sorban szeretnénk megköszönni családtagjainknak és barátainknak a támogatásukat, türelmüket, kitartásukat, hogy elviselték bezártságunkat, idegeskedésünket.  
A csapat minden tagja hálás köszönettel tartozik egymás segítsége és odafigyelése iránt.