**Beadandó Feladat Fejlesztési dokumentáció**

WEB Programozás II. GINFBAL-WEBPROG2-1

Lovas Bálint (RYGD2D)

Fekete Zoltán (3PCTLA)

2021/22/1 5. félév

Neumann János Egyetem GAMF Műszaki és Informatikai Kar

Mérnökinformatika - Levelező Bsc

Gyakorlat vezető: Dr. Subecz Zoltán

1. A program célja, és feladata

A program célja Webprogramozás-II. első beadandó feladatainak követelményeinek megfeleljen. Témaválasztásunk Hatoslottóra esett, melynek hivatalos weboldala volt számunkra mintának követendő. A feladat megoldásához kapcsolódó igényeket kívánja kielégíteni, amiket pontokba szedve tárgyal ez a dokumentáció, hogy hasznos segédje legyen a fejlesztő vizsgálatának

* 1. Gépi és nyelvi igények:

A weblaphoz tartozó fájlok értelmezéséhez PHP, HTML5, Javascript, CSS, Bootstrap és SQL ismeretek szükségeltetnek. PHP-Strom használata javasolt és a futtatásához szükséges gépigény az irányadó a minimális és az ajánlott követelmények alapján.

[https://www.jetbrains.com/help/phpstorm/installation-guide.html#requirements](https://www.jetbrains.com/help/phpstorm/installation-guide.html%23requirements)

* 1. Teszt környezet gépleírása:

PC 1:

OS: Microsoft Windows 10 Pro

Processzor: Intel(R) Core(TM)2 CPU 6420 @2.13GHz 2.13 GHz

Memória: 8GB RAM, 250 GB SSD + 500 GB HDD ROM

Használt böngészők: Google Chrome

PC 2:

* 1. A weblap fejlesztéshez kapcsolódó elérhetőségek, hozzáférési adatok

URL[: http://hatoslotto.nhely.hu/](file:///C:\Users\fekzo\OneDrive\Asztali%20gép\%20http\hatoslotto.nhely.hu\%0d)

**Nethely**

account: [fekzoletetan@gmail.com](mailto:fekzoletetan@gmail.com)

password: JC!Pia!GRsfX4Y8

**FTP**

account: fekez

password: JC!Pia!GRsfX4Y8

**Adatbázis**

account: fekez

password: JC!Pia!GRsfX4Y8

GIT: <https://github.com/trays93/webprog2>

* 1. Feladatok eloszlása

**Lovas Bálint**

* Project architektúra, routing elkészítése
* Adatbázis kialakítása, adatfeltöltési script
* Oldal layout kialakítása
* Felhasználó regisztráció
* SOAP szerver implementálása
* MNB árfolyamok megjelenítése
* Számstatisztika

**Fekete Zoltán**

* „Kezdőoldal” tartalom
* „Előző sorsolások” év és hét alapú keresés
* „Hírek” oldal
* Java és css állományok auto betöltése
* Ezen feladatok dokumentációja
* Internetes tárhelyre való feltöltés

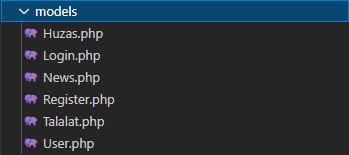
1. Mappa, fájlszerkezet, állományok bemutatása

A mappaszerkezet bemutatása során megkülönböztetünk többféle csoportot is. A mappák rendezettsége logikai struktúrát takar, ami segíti a fejlesztőt az átláthatóság végett. A feladat során az MVC (Modell–View-Controller) tervezési mintát használtuk, mely a mappák és állományok szerkezetét és kapcsolatait is meghatározta. Kapcsolati téren OOP jelenik meg.

* 1. index.php

Az alkalmazás belépési pontja. A használt osztályok autoloadolása és az alkalmazáson belüli útválasztás megkezdése.

* 1. Models

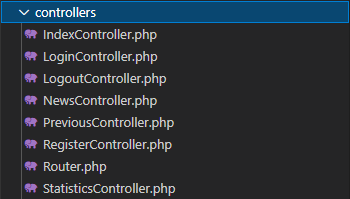


1. ábra

Adatok tárolására szolgáló osztályok, melyek a tárolni kívánt tartalomhoz adaptálódnak (1. ábra).

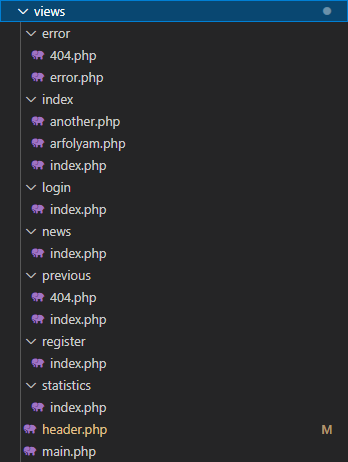
* Login.php: a felhasználó bejelentkeztetéséhez használt osztály.
* Register.php: a felhasználó regisztrációjához használt osztály.
* User.php: a bejelentkezett felhasználó adatait tároló osztály.
* News.php: Hírek/fórum oldal adatszerkezetéért felelős. News osztály a felhasználó nevét, azonosítóját, a bejegyzés dátumát és a bejegyzést képes tárolni. Tartozik hozzá egy konstruktor, mely, inicializálja az elemeket, ha minden paramétert megkap hozzá. Valamint, minden taghoz tartozik egy getter-setter tulajdonság.
* Huzas.php: a lottóhúzások eredményeinek tárolásához használt osztály.
* Talalat.php: egy adott lottóhúzáshoz tartozó találatok eredményeit tároló osztály.
  1. Controllers

1. ábra



Eseményekre, felhasználói műveletekre reagál. A reakció lehet, az aktuális vezérlőhöz tartozó modell egy függvényhívása vagy adat átadása a nézetnek (2. ábra).

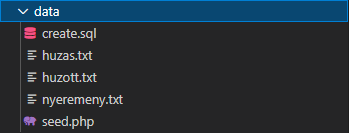
* Router.php: az alkalmazás belső útválasztó (routing) logikáját implementáló osztály. A statikus *route* metódus az URL alapján példányosítja a megfelelő controller osztályt és meghívja annak a kért metódsát. Alapértelmezetten az *IndexController* *indexAction* metódusát hívja meg, hiba esetén egy 404-es oldalt jelenít meg.
* LoginController.php: megjelenít egyűrlapot, aminek a segítségével lehet bejelentkezni az alkalmazásba. Sikeres bejelentkezést követően a kezdőoldalra irányítja a felhasználót, sikertelenség esetén kiírja a hibát a felhasználónak.
* LogoutController.php: ez az osztály végzi el a bejelentkezett felhasználó kijelentkeztetését.
* NewsController.php: Ennek az osztálynak 3 metódusa van. Az indexAction() függvény felelős a főbb adatokért: Átadja a sanitizeData() és validateData() függvényeknek a POST-olt kommentet, és azt feldolgozhatóvá teszik (szűrik a kapott adatokat, megelőzve a script injectálást vagy egyéb nem érvényes adatokat). Ellenőrzi, hogy kapott-e POST metóduson keresztül bejegyzeni való bejegyzést, valamit, hogy be van-e jelentkezve bármilyen felhasználó. Ha mindkettő igaz, és az adat is feldolgozható, akkor adatbázist kérünk le adatokat. Ellenőrizzük, hogy ez a felhasználó elküldte-e már ezt az üzenetet egyszer? Ha nem, akkor elkezdjük rögzíteni az adatbázisban a commentet az aktuális idővel és felhasználóval együtt. Ezután lekérjük a bejegyzéseket, és átadjuk a helyes nézetnek az adatokat.
* PreviousController: a régebbi húzásokat kezeli. Nulladikként azonosító értéke nulla. Következő lépésben ellenőrzi, hogy kapott-e GET metóduson keresztül azonosítót. Ha igen, átadja azt. Ezután ellenőrzi, kapott-e POST metóduson keresztül adatot a kért évről és hétről. Ha igen, lekéri SOAP szerveren keresztül az adott időhöz a húzás azonosítóját. Ha nem volt húzás, azonosító értéke -1 lesz és hibaüzenet és hibaoldal fog megjelenni. Ha az azonosító rendben van, akkor lekérjük a húzásról a kért adatokat a SOAP szerveren keresztül és továbbadjuk a Lotto modell konstruktorának, amely eltárolja az adatokat. Ezt tovább adjuk az osztályhoz tartozó nézetnek.
* RegisterController.php: egy űrlapot jelenít meg, amin a felhasználók be tudnak jelentkezni az alkalmazásba. A sikeres regisztrációt vagy a hibákat is kijelzi a felhasználó számára.
* StatisticsController.php: a felhasználónak megjelenít egy felületet, ahol kiválaszthatja a kedvenc számait és lekérdezheti, hogy azokat összesen hányszor és mikor húzták ki utoljára.
  1. Views



1. ábra

A felhasználónak jeleníti meg a Controllerek által átadott adatokat, valamint egyéb elemeket. Felelős, a felhasználóval való kapcsolattartásért. Html tartalom ténylegesen itt jelenik meg. A nézetek konstruktora a nézetet tartalmazó mappa nevét és a nézet kiterjesztés nélküli nevét várja paraméterül. A mappa neve az oldal nevével kell megegyeznie, a nézet neve pedig a vezérlő adott oldalhoz meghívott metódusnevével kell megegyeznie. (3. ábra)

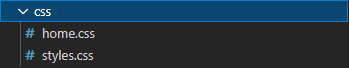
* error/404.php, error.php: hibás, nem létező URL-ra reagál, figyelmezteti róla a felhasználót.
* index/index.php: Ez a főoldal. Itt egy rövid bemutatkozás/megvalósítási cél szerepel. Alatta egy JavaScript-ben megvalósított visszaszámláló, ami az aktuális dátum-időtől számol vissza az adott hét vasárnap délután 4 órájától.
* news/index.php: A bejelentkezett felhasználók számára fenntartott oldal. A lapon található egy űrlap, amelyen a felhasználó új véleményt, hírt adhat hozzá a laphoz. Alatta a korábbi bejegyzések szerepelnek csökkenő idősorrendben.
* previous/index.php: Előző sorsolásokat jeleníti meg. Választani a sorsolások között lehet év és hét alapján. Ha nem megfelelő a dátum, lehet lapozni a heteket az előző és következő húzott hét alapján. Megjeleníti a húzott számokat, a találatokat, azok darabszámait és a hozzájuk tartozó nyereményt. Az évek kiválasztásánál, csak az adatbázisban fellelhető évszámok találhatóak meg. A heteknél az összes hét szerepel, független attól, hogy volt-e húzás az adott héten vagy nem. Ezért, ha olyan hét kerül kiválasztásra, amelyen nem volt húzás, akkor átirányítódig az oldal a error/error.php oldalra, ami jelzi a felhasználónak, hogy nem volt húzás az adott héten.
* header.php: kakukktojás, mert nem teljesen nézet, de ez tartalmazza a fejlécben lévő reszponzív menüt, a ki- és bejelentkezést, valamint a regisztrációt.
  1. Data



1. ábra

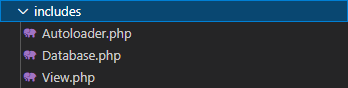
Az adatbázis létrehozásához és kezdeti adatfeltöltésre szolgáló állományok találhatóak itt. (4. ábra)

* create.sql: az adatbázist, táblákat és megszorításokat létrehozó SQL utasításokat tartalmazza.
* huzas.txt, huzott.txt, nyeremeny.txt: az alkalmazásban használt adatok, CSV formátumban.
* seed.php: php script, ami az előzőekben felsorolt fájlokból kinyeri, majd eltárolja az adatokat.
  1. Styles



1. ábra

A mappa kettő darab css állományt tartalmaz, amelyek a megjelenés, tördelés, betűméretek és különböző elemek egyedi azonosítóval ellátott (class, id) megjelenését állítják (6. ábra).



1. ábra
   1. Includes

Az alkalmazáshoz használt segédosztályok. (7. ábra)

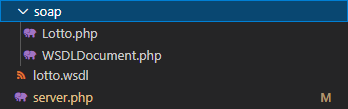
* Autoloader.php: ennek az osztálynak a segítségével töltjük be az osztályokat (spl\_autoloader\_function).
* Database.php: statikus osztály ami egy PDOConnection példányt ad vissza. Az egyke tervezési mintát valósítja meg.
* View.php: a megfelelő nézet megjelenítését végzi el.
  1. Scripts

1. ábra



A script mappában jelenleg a home.js az egyetlen fennlelhető állomány (8. ábra). Feladata: a főoldalon található visszaszámlálás vezérlése. Az oldal betöltődése után meghívja a counter() függvényt. Ez a függvény milliszekundumomként frissíti a számlálók értékét. Működését tekintve elsőnek lekéri az aktuális napot a hétből, órát, percet és másodpercet és másodpercként összegzi azt. Ugyanezt megcsinálja a kitűzött hétvégi időponttal is, amiből végül kivonja az aktuális időt. A különbözetet visszaalakítja nap, óra, perc, másodperc formátumúvá és kiíratja a főoldalra. Az állományok nevének meg kell egyeznie a felhasználandó oldal nevével, az automatikus betöltésért cserébe.

* 1. Soap



1. ábra

SOAP megvalósítás segítségével kérdezhetőek le az adatbázisból a korábbi húzások. (9. ábra)

* Lotto.php: az adatbázis lekérdezéseket megvalósító osztály.
* WSDLDocument.php: a SOAP kommunikációhoz szükséges lotto.wsdl fájl generálja le a Lotto.php-ból.
* server.php: publikus SOAP interfészt nyújt (SOAP server).

1. **Tesztelési terv, és eredményei**

A tesztelési tervezetünkben pár kívülről is látványos interakciót kívánunk bemutatni.  
A tesztelési terve tartalmazza:

„Előző sorsolások” közül létező időpont lekérése (a), valamint olyan időpont, amin nem történt sorsolás (b). „Előző”, „Következő” ikonok megfelelő működése.

„Számstatisztika” oldal lekérése (a), valamint hiányzó adatokkal (b).

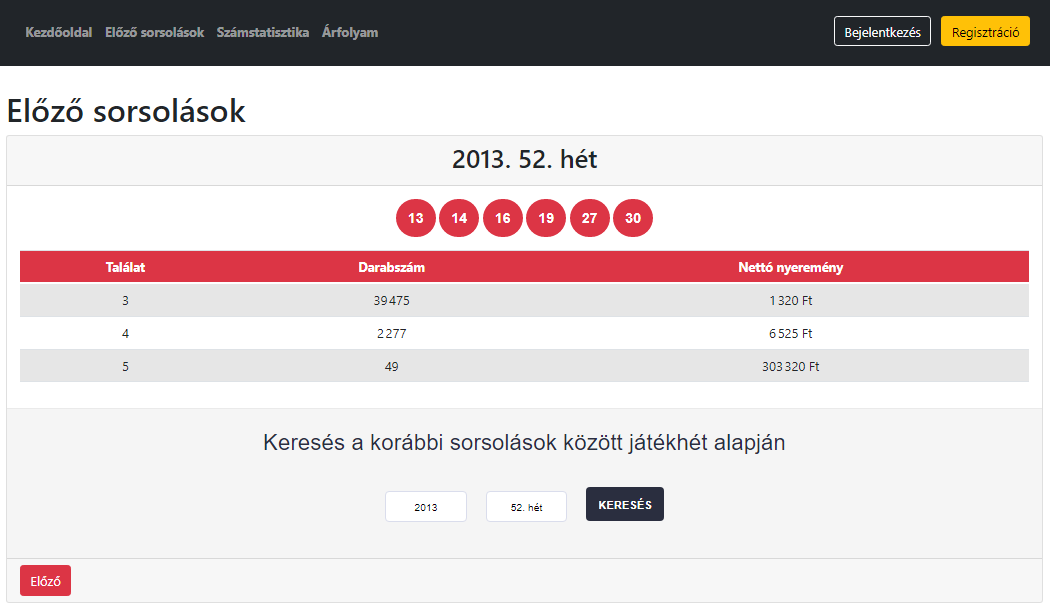
„Árfolyamok" helyes működése

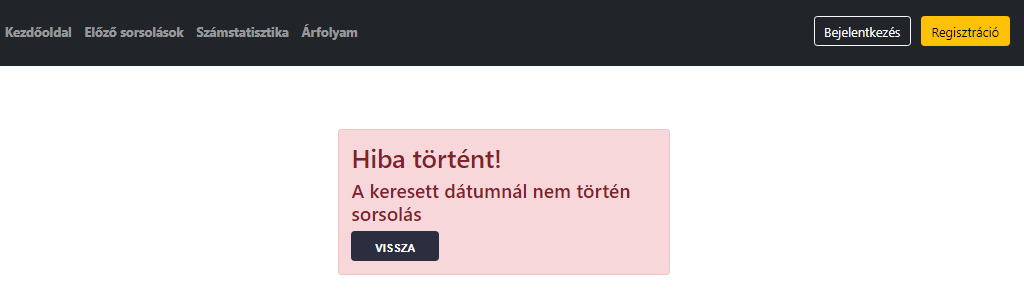
„Regisztráció” hibás adatokkal (a) és helyesekkel (b).

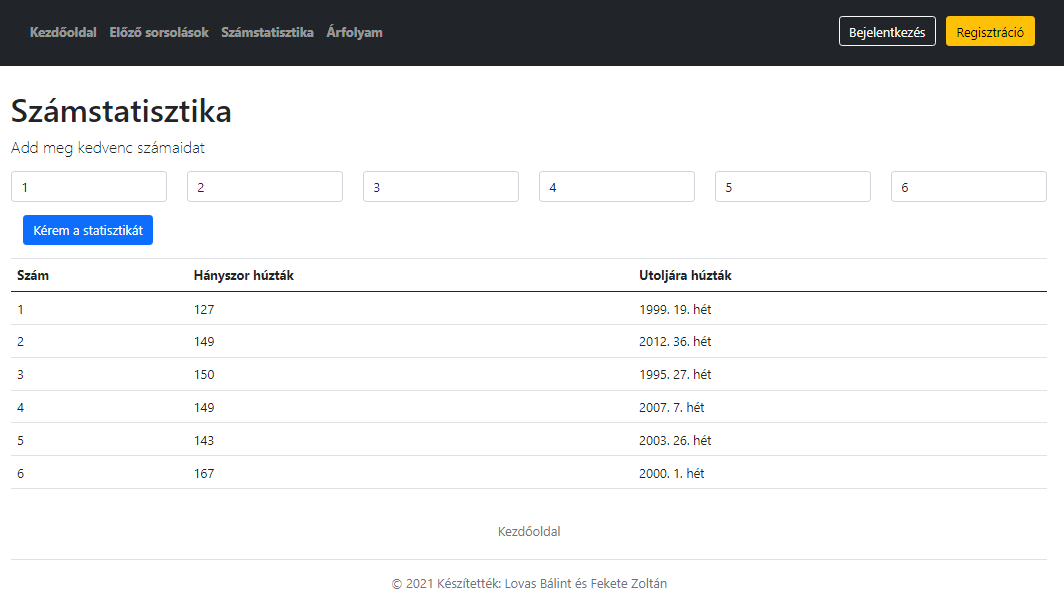
„Bejelentkezés” hibás adatokkal (a) és helyesekkel (b).

„Hírek” oldal bejegyzés hibás adatokkal (a) és helyesekkel (b).

Sikeres-e a főoldal reszponzív működése?

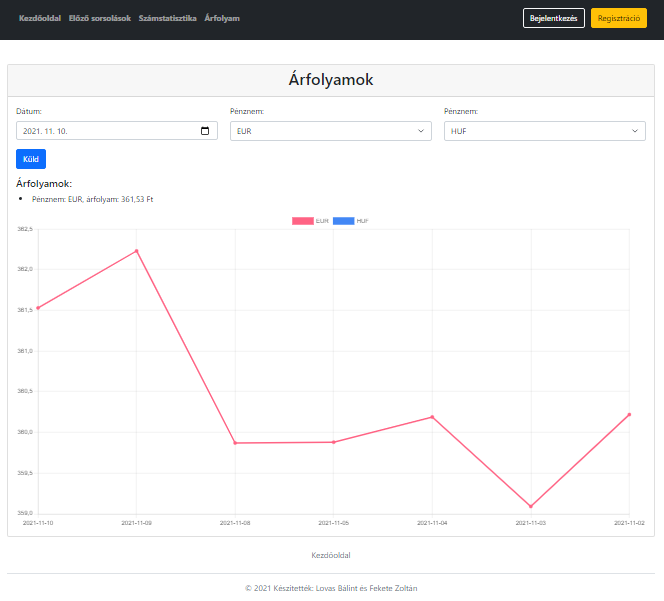
1/a

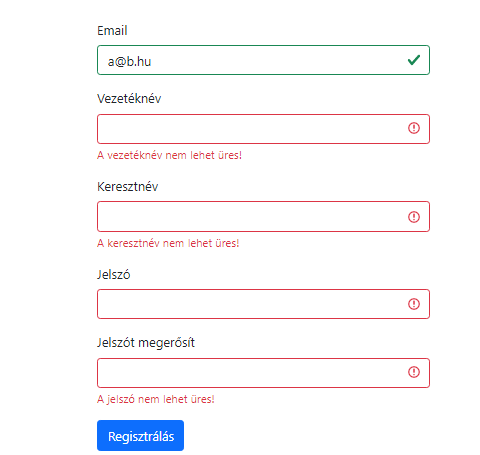
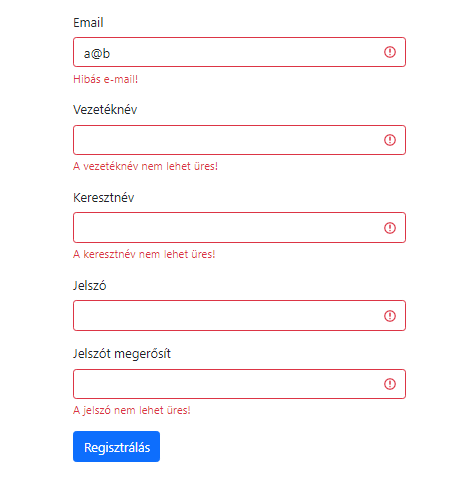
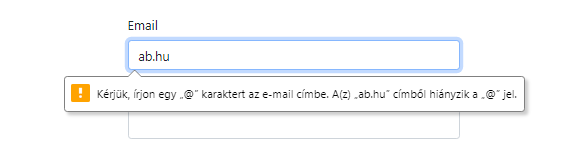
1/b

2/a

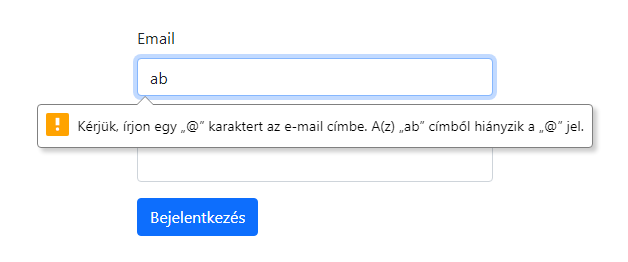
2/bA képen szöveg látható

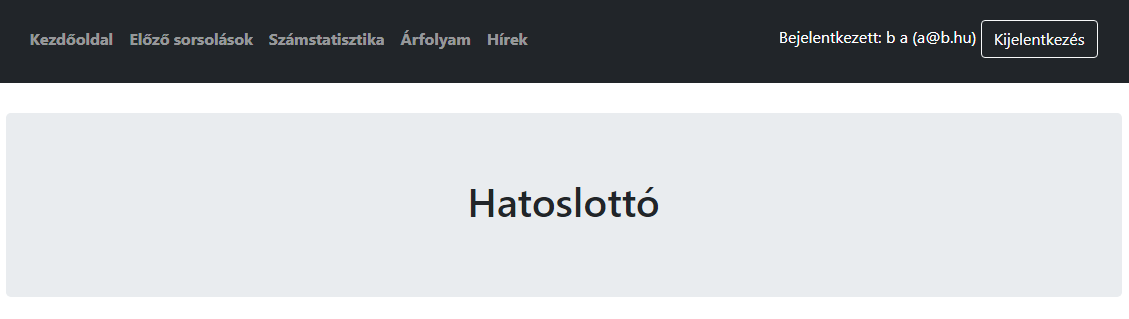
Automatikusan generált leírás

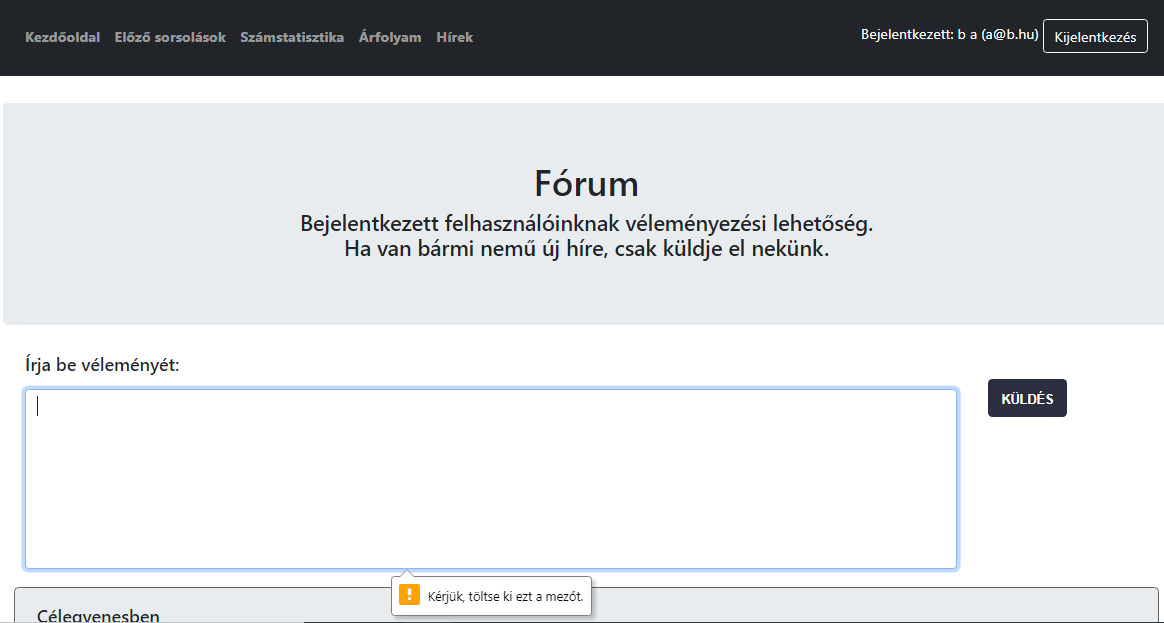
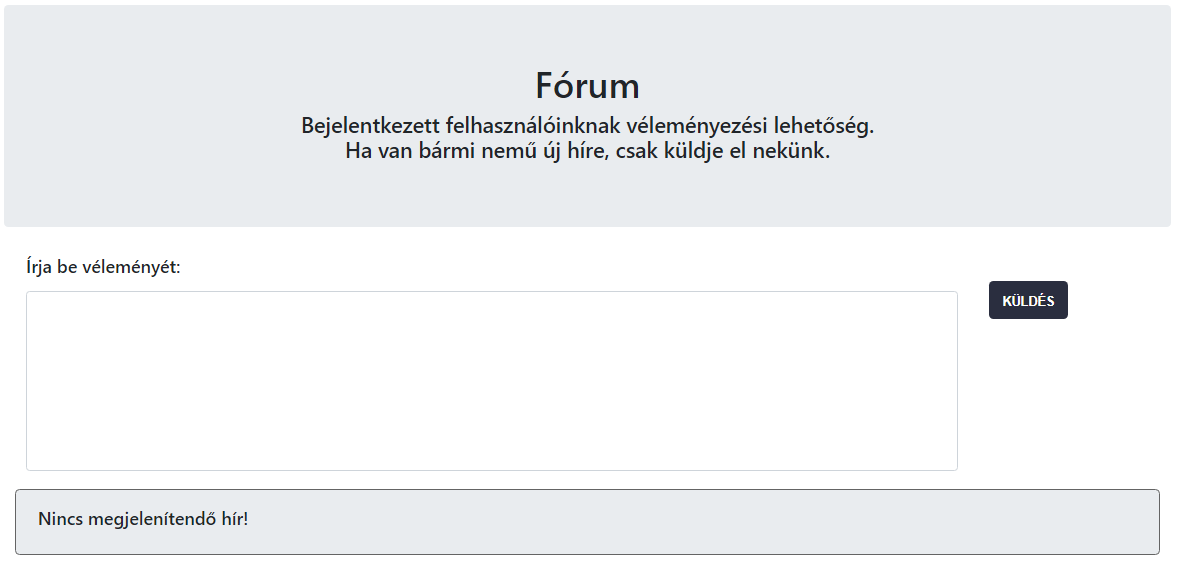
3

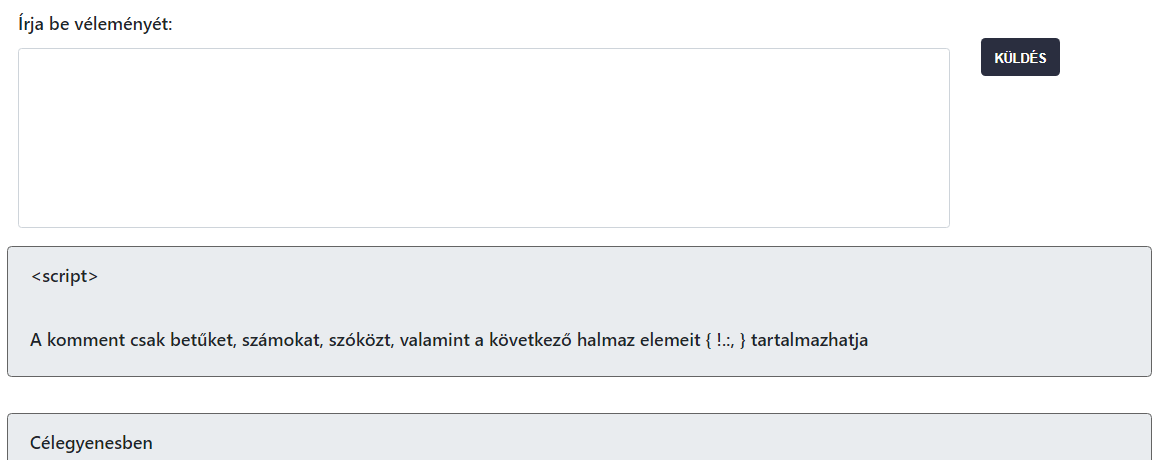
4/a

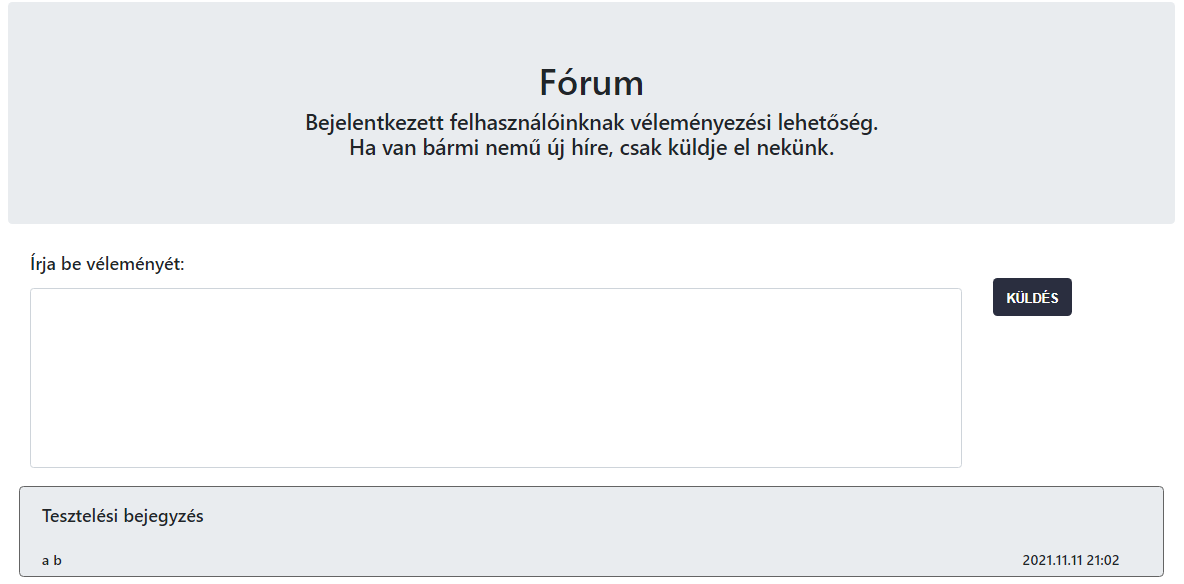
5/a  
A képen szöveg látható

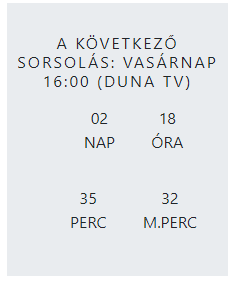
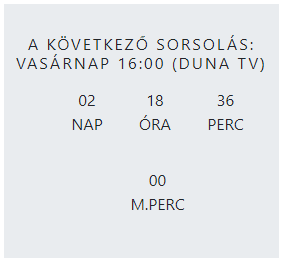
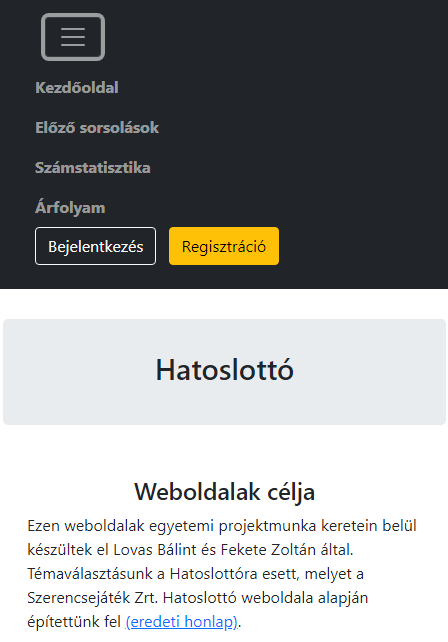
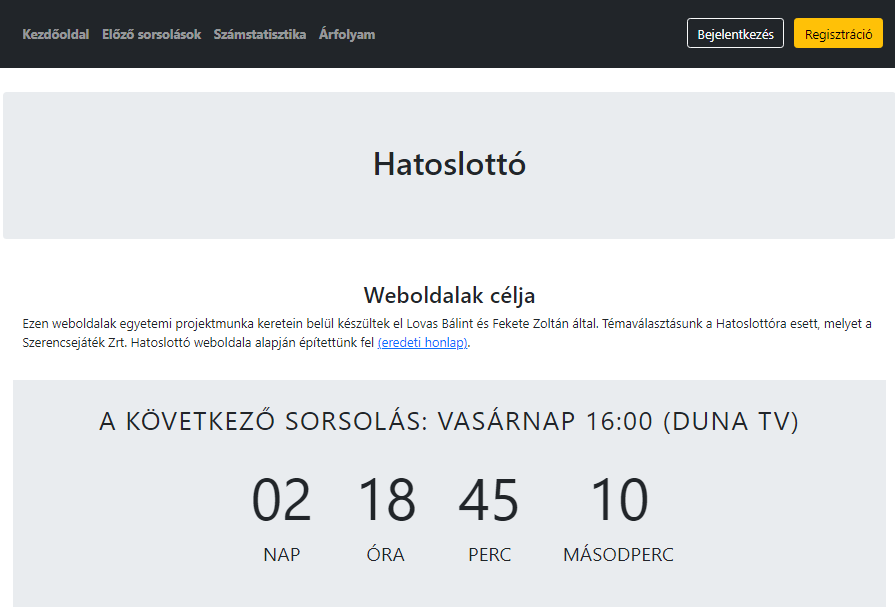
Automatikusan generált leírás



6/a



6/b

7

1. **Program továbbfejlesztési lehetőségei**

A program potenciális továbbfejlesztése során lehetne az oldalakat stílusosabbá tenni, valamint képeket feltölteni. Lehetne akár még egy online fogadási felület is. Kapcsolatok menüpont.