





# 2023

## Forma 1 hírportál


Főoldal Menetrend ▾ Tabelek ▾ Rólunk Összes cikkQ




Verstappen nyerte a kaotikus véget ért Ausztrál Nagydíjat.  
Magnussen balesete után jött a káosz Melbourneben ahol mindkét Ferrari és Alpine is nullázott.




AUSTRALIA  
RACE HIGHLIGHTS




AUSTRALIA  
QUALIFYING HIGHLIGHTS




SAUDI ARABIA  
RACE HIGHLIGHTS




MAX VERSTAPPEN NYERTE AZ AUSZTRÁL NAGYDÍJ ESEMÉNYDUSRA SIKERÜLT ELSŐ SZABADEZSEST




A MERCEDES BAHREIN ELŐTT TISZTÁBAN VOLT A W14-ES KONSTRUKCIÓJUK KORLÁTAIVAL




PÉREZ NYERT SZAÚD-ARÁBIÁBAN, VERSTAPPEN A MÁSODIK HELYIG ZÁRKÓZOTT




SZAÚD-ARÁBIAI NAGYDÍJ: VERSTAPPEN FUTOTTA A LEGJOBB IDŐT A MÁSODIK SZABADEZSESEN KÉT TIZEDDEL MEGLŐZVE ALONÓS



HIVATALOS: LECLERC-E AZ ELSŐ RAJTBUKTATÁS AZ IDEI SZÉZONBAN



KIDERÜLT, MIÉRT ESETT KI CHARLES LECLERC BAHREINBEN



AZ ASTON MARTIN CSAK

27

03

46

55

Nap/Days Ora/Hours Perc/Mins Mp/Secs

És elrajtol a(z)  
Azeri Nagydíj

Szép Norbert, Szekleván Richárd

Nógrád Megyei Szakképzési Centrum

Szent-Györgyi Albert Technikum

Szakma megnevezése:

Szoftverfejlesztő és -tesztelő technikus

Szakma azonosító: 5-0613-12-03

Dátum: 2023.04.18.

## Tartalom

Bevezetés.....	3
1. Felhasználói dokumentáció .....	4
1.1. Főoldal.....	4
1.2. Regisztráció.....	6
1.3. Belépés .....	7
1.4. Új jelszó igénylése .....	7
1.5. Menetrend .....	8
1.6. Tabellák.....	9
1.7. Összes cikk oldala .....	12
1.8. Adminisztrációs felület .....	12
1.9. Cikkek felvitele .....	13
1.10. Cikkek kezelése.....	15
1.11. Felhasználók kezelése .....	16
1.12. Adminisztrátor felvétele.....	17
1.13. Adminisztrátor, felhasználó törlése.....	17
1.14. Adminisztrátori chat.....	17
2. Fejlesztői dokumentáció .....	19
2.1. Fejlesztői környezet .....	19
2.1.1. Visual Studio Code .....	19
2.1.2. CSS .....	19
2.1.3. Bootstrap.....	20
2.1.4. PHP.....	20
2.1.5. JavaScript.....	21
2.1.6. WinSCP .....	21
2.1.7. Nethely.....	21
2.2. Tervezés folyamata .....	21

2.3.	Lapvédelem .....	22
2.4.	Adatbázis .....	23
2.4.1.	Egyedek meghatározása .....	23
2.4.2.	Kapcsolatok: .....	23
2.4.3.	Táblák .....	24
2.4.4.	Információk megszerzése .....	29
2.5.	Főoldal .....	29
2.5.1.	Visszaszámláló megjelenítése a kezdő oldalon .....	32
2.5.2.	Navigáció és lábléc megvalósítása .....	34
2.5.3.	Sötét és világos mód közötti váltás .....	35
2.5.4.	Cikk felvitele .....	36
2.6.	Chat megvalósítása .....	37
2.7.	Hozzászólások lehetőségének megvalósítása .....	38
3.	Felmerült problémák .....	41
3.1.	A főoldalon a kártyák megjelenése .....	41
3.2.	Felhasználók és adminisztrátorok törlése .....	41
3.3.	Összes felvitt cikk megjelenítése egy oldalon .....	43
4.	Tesztelés .....	44
	Összefoglalás .....	46
	Források: .....	48
	Ábrajegyzék .....	49

## Bevezetés

A projektünk témájának meghatározása nem okozott nehézséget, mivel mindkettőnket érdekel az újságírás ezért rögtön valamilyen online hírportálon kezdtünk el gondolkodni. Mivel manapság már a hagyományos újságok helyett az emberek online olvasnak híreket. Ezen kívül úgy gondoljuk, hogy ha jól sikerül, akkor ezt szívesen fejlesztenénk tovább a jövőben is persze, akkor már valóban működő weboldalként.

A hírportálunk témaköréhez több ötletünk is volt végül úgy döntöttünk, hogy sporthírekkel szeretnénk foglalkozni azon belül is a Formula-1-el. Azért döntöttünk emellett, mivel mind a ketten régóta követjük a Forma-1 világot és ez a témakör áll hozzánk a legközelebb.

A projektünk célja az, hogy a sport világában napvilágot látott hírekről időben tudjuk értesíteni az érdeklődőket egy helyen kategorizálva. Egy hírportált működtetni és folyamatosan friss hírekkel feltölteni hatalmas feladat lenne két főnek. Ezért mi eleve úgy terveztük meg a rendszerünket, hogy egyfajta tartalom kezelőként működjön. Vagyis, ha jelentkezik hozzánk hobbi újságíró, és nincsenek komolyabb informatikai ismeretei, akkor is könnyen fel tudjon vinni új cikkeket a weboldalra. Tehát, ha az e-mail címével és jelszavával belép a kezelő felületre, akkor ott úgy tudja szerkeszteni, az általa megírt cikkeket úgy, mintha egy egyszerű Word dokumentumot formázná.

A cikkek előtt ugyanis szabály szerint pár soros bevezetőt kell írni, miről szól a cikk. Ezt általában dőlt betűvel szokás megjeleníteni a komolyabb hírportálokon. Előfordulhat, hogy valamilyen linket, vagy fontosabb mondatot szeretne kiemelni ezt is könnyedén megteheti, anélkül, hogy ismerné a HTML kódokat. Továbbá könnyedén tud képeket is feltölteni a rendszerbe.

A vizsgamunkák fő részei a felhasználói oldalak, amelyeket a látogató megtekinthet. Ezek a hírportál fő oldala, amellyel a felhasználó először találkozik. Innen tud navigálni, a menüpontok segítségével a további oldalakra, mint például a futamok menetrendje, tabellák és az összes cikk oldalra.

A munkánk másik nagy területe az adminisztrációs tartalomkezelő felület, ahol a leendő újságíróink be tudnak regisztrálni az oldalra. Ehhez létrehoztunk beléptető felületet, ahol a már meglévő regisztrációjukkal be tudnak lépni és új cikkeket tudnak létrehozni.

Az olvasók is regisztrálhatnak, de ők csak hozzászólásokat tudnak írni a különböző cikkekhez.

## **1. Felhasználói dokumentáció**

Dolgozatunk e fejezetében bemutatjuk a hírportálunk összes weboldalát, bele értve az adminisztrációs oldalakat is, mi mindent tekinthetnek meg a látogatók regisztráció nélkül is, és hogyan navigálhatnak az oldalak között.

Továbbá, hogyan regisztrálhatnak az oldalra, hol léphetnek be a felületre, és milyen lehetőségei vannak regisztrált felhasználóknak vendégként, valamint miként hozhatnak létre új cikkeket az oldal újságírói jogokkal rendelkező szerkesztői.

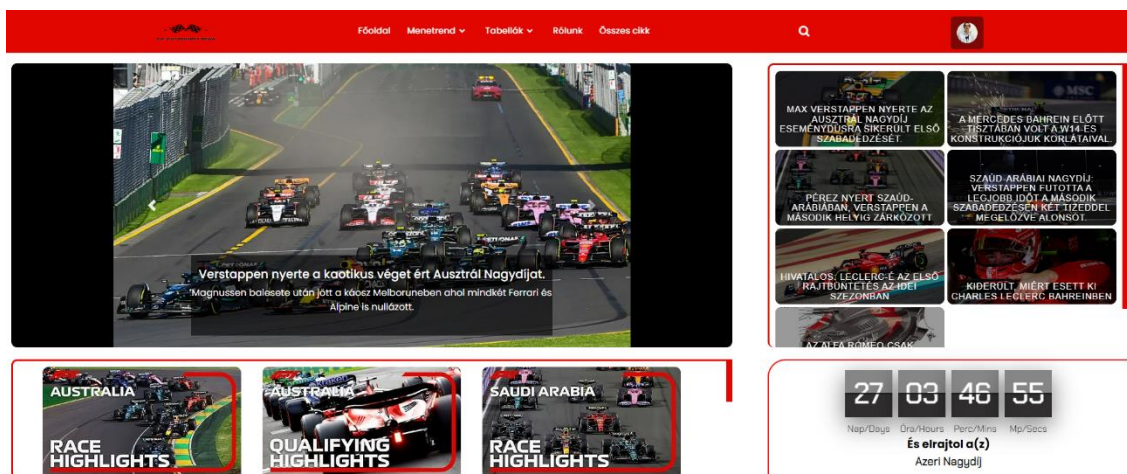
### **1.1.Főoldal**

A főoldal felső részén található a navigációs menü sáv, ahonnan további oldalak nyithatóak meg. Mobil nézetben hamburger ikonná alakul át, ennek köszönhetően teljesen reszponzív, akárcsak az egész weboldalunk elrendezése.

Az olvasó a menü sáv jobb oldalán egy alaphelyzetben beállított képet lát, ami egy V betűt ábrázol, mint „Vendég”, amelyre ha rá kattint, egy felugró ablak jelenik meg, ahol beállíthatja a nyelvet, - angol és magyar nyelv közül lehet választani -, valamint, hogy sötét vagy világos hátteret szeretne-e látni a weboldalon.

A fő oldalunk több különböző látványos blokkból áll. Igyekeztünk úgy kialakítani, hogy a vezető hírek azonnal megragadják a látogató figyelmét, ahol 4 fő kép váltakozik egymás után, és ha bármely képre kattint, akkor az az adott képhez tartozó cikk oldalára vezeti az olvasót.

A menü alatti jobb oldali sávban táblázatos elrendezésben 12 fotó látható, amelyek további régebbi cikkek oldalaira vezetnek. A 12 fotóból 3-3 jelenik meg teljesen, a többi egy gördítő sáv segítségével válik láthatóvá.



1) ábra Főoldal

A következő szakasz szintén 2 részre került felosztásra. A baloldalon névjegykártya méretű elrendezésben, soronként 3 fotó jelenik meg. Ezek száma nincsen fixálva, ez attól függ, mennyi videó linket csatolunk, és hány képet töltünk fel éppen. Ezt is görgethető megoldással oldottuk meg, mivel nem szerettük volna, ha az oldalunk túlságosan hosszú legyen. Ugyanis utána olvastunk az interneten a weboldalak „hő térkép” vizsgálatának, amelyből az derült ki, hogy a látogatók ritkán görgetnek túl sokat lefelé egy weboldalon, így az oldal legalsó szakaszára került cikkek már nem kerülnek olyan gyakran megtekintésre, mint ami a képernyőn azonnal megjelenik.

A második szakasz jobb oldalára egy visszaszámlálót helyeztünk el, amely a következő futam rajtjáig számol vissza.

A harmadik szakaszra egy látványos JavaScript effekttel széthyitható kártyákat helyeztünk el. A 10 kártyából az első

nyitott állapotban jelenik meg, mellette pedig az összecsukott sávokban a különböző istállók logói jelennek meg. Ha a felhasználó rákattint egy logóra, akkor a sáv kinyílik és a csapat rövid bemutatása jelenik meg a logó helyett, illetve a kártyák alsó részén végig futnak a csapatok autói.

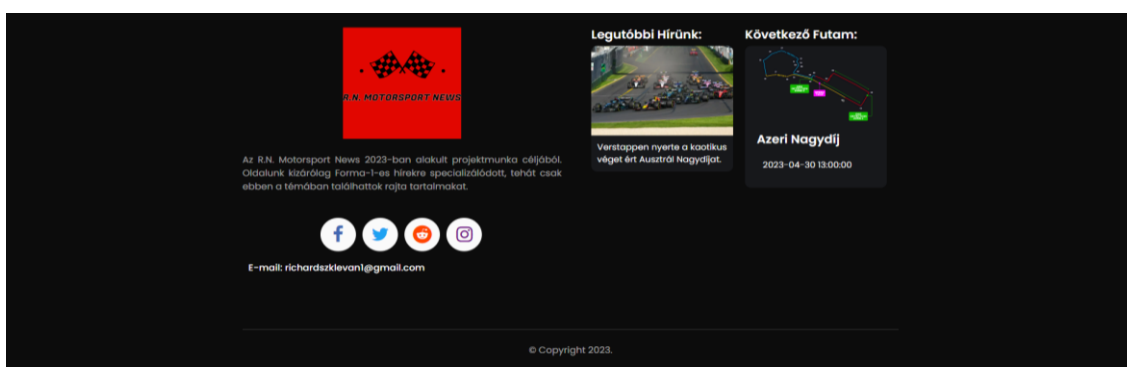


2) ábra Visszaszámláló a főoldalon



3) ábra Szétnyíló kártyák

Az oldal lábléc szakaszában a weboldalunk logója látható, amely alatt a közösségi oldalak – mint Facebook, Twitter, Reddit, és Instagram - ikonjait helyeztük el, arra az esetre, ha majd készítünk a weboldalunkhoz videókat, és média híreket ezekre a közösségi oldalakra is.



4) ábra Lábléc

## 1.2.Regisztráció

A regisztrációs oldal arra szolgál, hogy a felhasználók létre tudják hozni a saját fiókjukat. Ez nem kötelező, de ha valaki szeretne hozzászólni egy cikkhez, akkor azt csak előzetes regisztrációval majd az oldalra való belépéssel tudja megtenni. A regisztrációnál a felhasználónak meg kell adnia a nevét és választania kell egy egyedi felhasználónevet. A felhasználónévnek legalább három karakterből kell állnia. Emellett meg kell adnia az e-mail címét és egy jelszót kell ki találnia. Valamint opcionálisan lehet profilképet feltölteni, de ez nem kötelező.

Az oldal dizájnjának alapjául Rajasthani Coder Codepen oldalon talált mintáját használtuk. Ezt alakítottuk át a saját elképzelésünk szerint, gondolunk itt a képernyőn végig futó autókra, illetve a

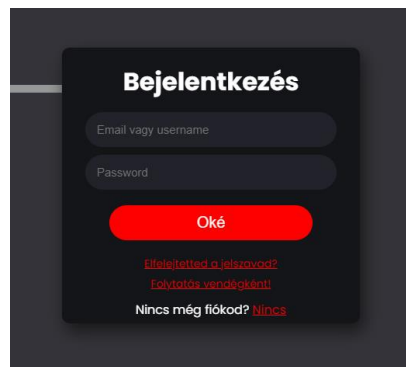
 The image shows a registration form titled "Regisztráció". Below the title is a note: "A \*-al jelölt mezők kitöltése kötelező!". There is a section for "Fotó feltöltése" with a button "Fájl kiválasztása" and the text "Nincs fájl kiválasztva". Below this are input fields for "Név", "Felhasználónév", "E-mail", "Jelszó", and "Jelszó újra". At the bottom is a red button labeled "Regisztráció".

5) ábra Regisztrációs felület

hamburgermenüre rákattintva a többszörösen megjelenő rétegekre. Ezt a dizájnt használtuk fel a belépéshez, regisztrációhoz illetve az új jelszó igénylése oldalunkon is, viszont itt csak az autók láthatóak a menü nincs beépítve.

### 1.3.Belépés

Ha már van fiókja valakinek, akkor abba belépni ezen az oldalon keresztül tud. A belépéshez meg kell adnia azt az e-mail címet vagy felhasználónevét, amivel regisztrált az oldalra és a jelszavát. Ezek nélkül sikertelen lesz a belépés. Ha esetleg a felhasználó elfelejtette a jelszavát, akkor az: „Elfelejtetted a jelszavad?” feliratú linkre kattintva az, e-mail cím megadásával igényelhet újat. Illetve található még egy link a „Folytatás vendégként!”, amivel a felhasználó a nélkül tudja megtekinteni az oldalt, hogy belépett volna.

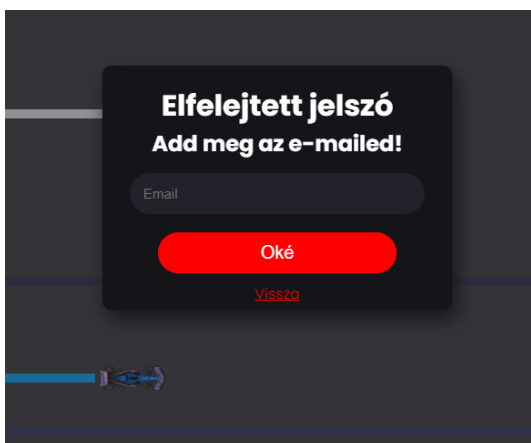


6) ábra Bejelentkezés felülete

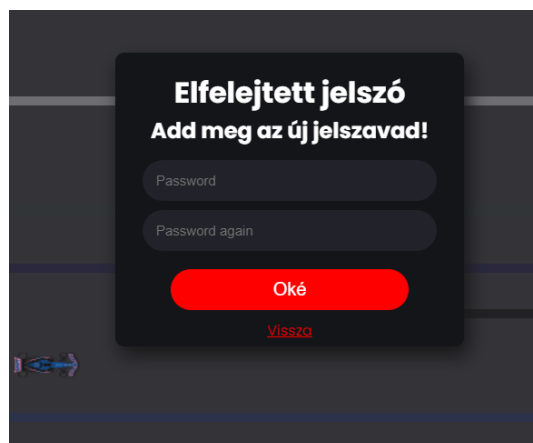
### 1.4.Új jelszó igénylése

Ezen az oldalon tud a felhasználó új jelszót igényelni az e-mail címe megadásával, ha elfelejtette volna azt. Ebben az esetben nem e-mailt kap a felhasználó, hanem a program ellenőrzi, hogy az adott e-mail cím megtalálható-e az adatbázisban, és ha igen akkor meg tudja adni az új jelszavát a felhasználó, valamint ezt meg is kell erősítenie. Ekkor a program frissíti a jelszó mezőt az adott felhasználó email címének megkeresésével arra a jelszóra, amelyet a felhasználó újonnan beírt.





7) ábra Elfelejtett jelszó



8) ábra Új jelszó megadása

## 1.5.Menetrend

A világbajnokság menetrendjét az oldalunkon, két helyen is megjelenítjük. A főoldalon a navigációs sávban, ha a felhasználó ráhúzza a kurzort a menetrendek menüpontra, akkor egy lenyíló menü jelenik meg közvetlenül a menü alatt a következő négy futam kezdési időpontjával. A sorban az első futam el van választva egy szürke kerettel a többitől, hogy ezzel is érzékeltesük, hogy melyik a soron következő futam.



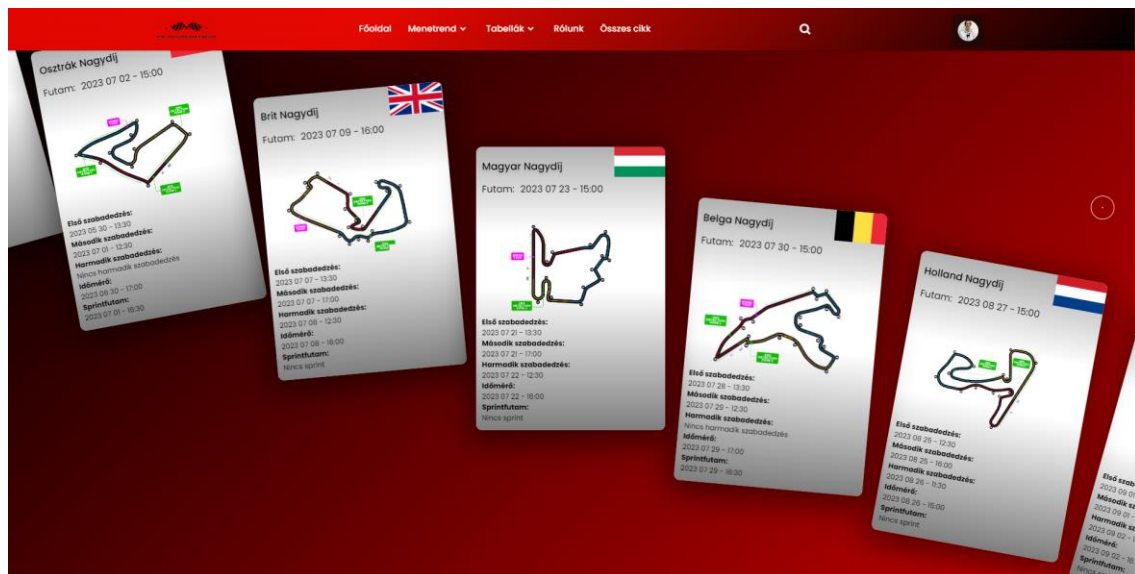
9) ábra Menetrend almenü

Ha részletesebb információkat szeretne megtudni az olvasó az adott versenyhétvégéről, akkor a „Menetrendek” felíratra kattintva az oldal átirányítja egy új oldalra, ahol a teljes szezon menetrendjét láthatja.

Ezen az oldalon kis kártyák formájában jelenítjük meg a szükséges és legfontosabb információkat. Minden kártya felső része tartalmazza a futam nevét illetve közvetlenül alatta magának a futamnak a kezdési időpontját magyar idő szerint. A jobb felső sarokban tartalmazza tovább a rendező ország zászlaját. A kis kártyák közepén található a pálya alaprajza, ami valamennyi pálya esetében ugyanazzal a dizájnnal van rajzolva. Ennek az oka az, hogy a Las Vegas-i Nagydíj idén kerül először megrendezésre és még nincs hivatalos pályarajz róla. A kép alatt felsorolás szerűen pedig a hétvége teljes menetrendje

látható az első szabadedzésztől az időmérőig. Ezeket az adatokat adatbázisból kiíratva jelenítjük meg.

A kártyák a képernyő közepén jelennek meg ívelten. Az egeret oldal irányba húzva tudunk váltani a pályák között. Ezt az ötletet a CodePen oldalán találtuk és alakítottuk át az oldalunk dizájnjának megfelelően.



10) ábra Teljes menetrend

## 1.6.Tabellák

A világbajnokság állását az oldalunkon, két helyen is megjelenítjük. A főoldalon a navigációs sávban, ha a felhasználó ráhúzza a kurzort a tabellák menüpontra, akkor egy lenyíló menü jelenik meg közvetlenül a menü alatt a világbajnokság jelenlegi állásával. Ez a menü tartalmazza az egyéni és a konstruktóri bajnoki tabella állását is. A szükséges adatokat adatbázisból olvassuk be itt is.

Az első három pilóta és csapat megjelenítésére nem a nevüket, hanem a képüket olvastuk be és azt jelenítettük meg olyan elrendezésben mintha egy dobogóra lennének elhelyezve. Ezek mellett látható a képek mögött, hogy a helyezésnek megfelelő színű árnyék van hozzáadva a képekhez. Szerintünk ez a megoldás nagyon látványosra sikerült és ezért is döntöttünk emellett.

<

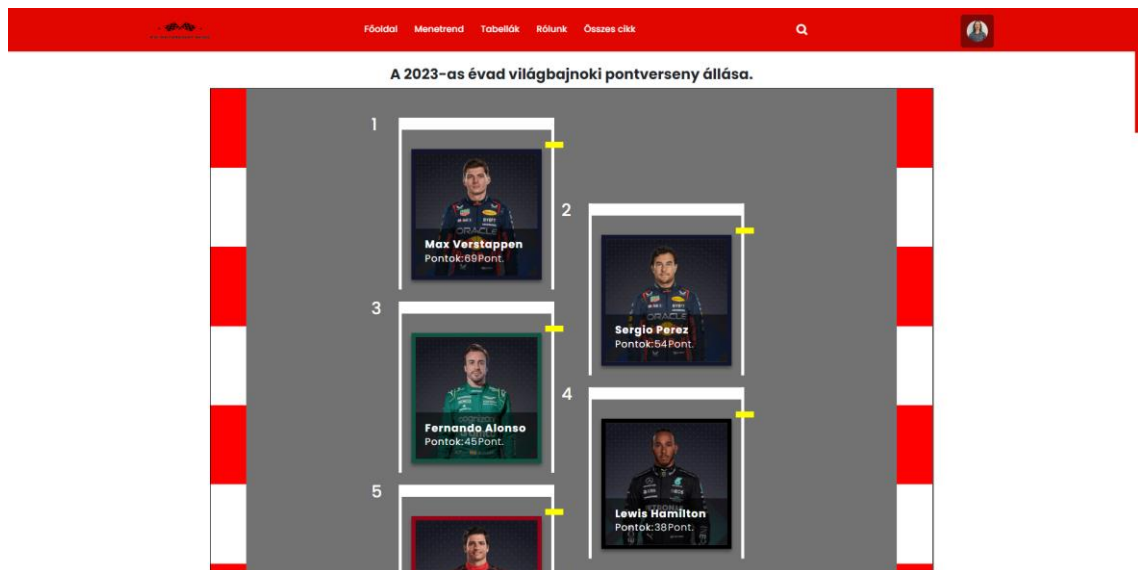
11) ábra Tabella almenü

Ha az olvasó rákattint a lenyíló menüben megtalálható „Tabellák” felíratra akkor az oldal átirányítja, őt egy másik oldalra ahol szintén a tabellákat tekintheti meg. Ez az oldal ugyan azt a funkciót látja el, mint a lenyitható menü.

Ennek az oldalnak a létrehozását azért gondoltuk szükségesnek mivel telefonos nézetben a lenyitható menü nem jelenik meg ezért, hogy ott is meg lehessen tekinteni a világbajnokság állását úgy döntöttünk kiegészítjük egy ilyen oldallal is.

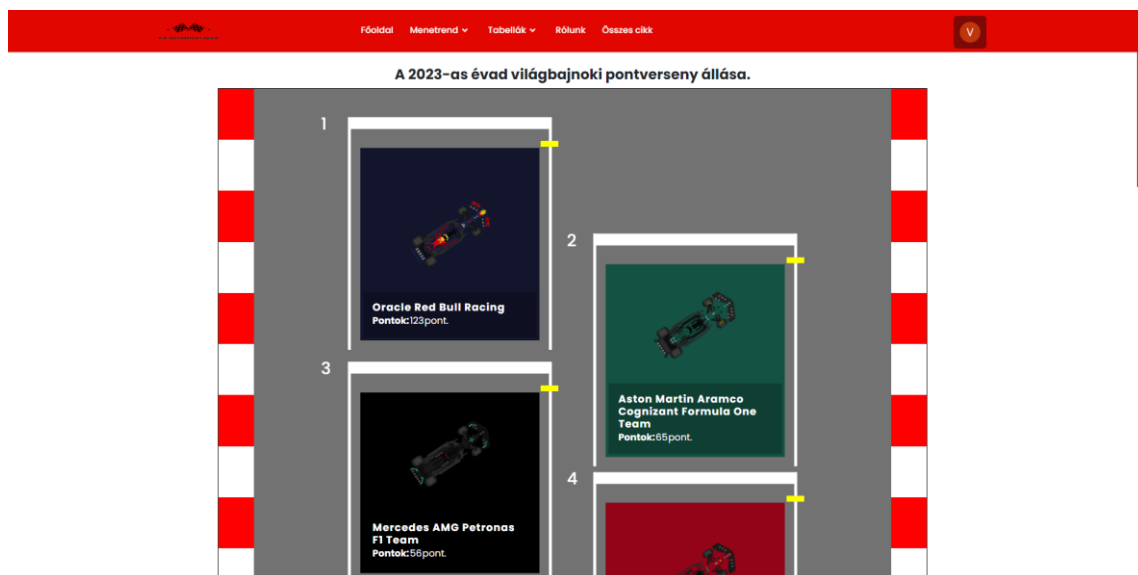
Ezen az oldalon is szintén kártyákkal találkozhatunk, amit flexbox segítségével rendeztünk el. A kártyát teljes egészében kitölti a pilóta fotója. Erre a fotóra írtattuk ki a szezonban elért pontjainak a számát és a teljes nevét is. Minden kártyának van egy szegélye, ami olyan színű, mint a csapatok autóinak a festése.

A megjelenítést úgy valósítottuk meg, mint egy Forma-1-es futam rajtrácsa. A bajnokságot vezető pilóta van az első rajtkockában és a pontjaik szerint csökkenő sorrendben egyre lejjebb helyezkednek el a rajtrácson. Ez a szekció az oldal közepén helyezkedik el, melynek széleit úgy terveztük meg mintha azok, rázókövek lennének.



12) ábra Teljes tabella

Ha a csapatok tabellájára vagyunk kíváncsiak, akkor az oldal alján lévő „Csapatok pontjai” feliratú gombra kattintva azt is meg tudjuk tekinteni. Ez dizájnbán, alapjaiban megegyezik a pilóták tabellájával. Azonban fel akartuk dobni valamivel az oldalt és mivel ide az autóknak a rajzát helyeztük el képként úgy gondoltuk látványos lenne, ha ezeket az autókat forgatnánk folyamatosan. Adatok szempontjából ugyan azokat íratjuk ki itt is, mint a pilóták esetében csak a csapatokra vonatkozóan.

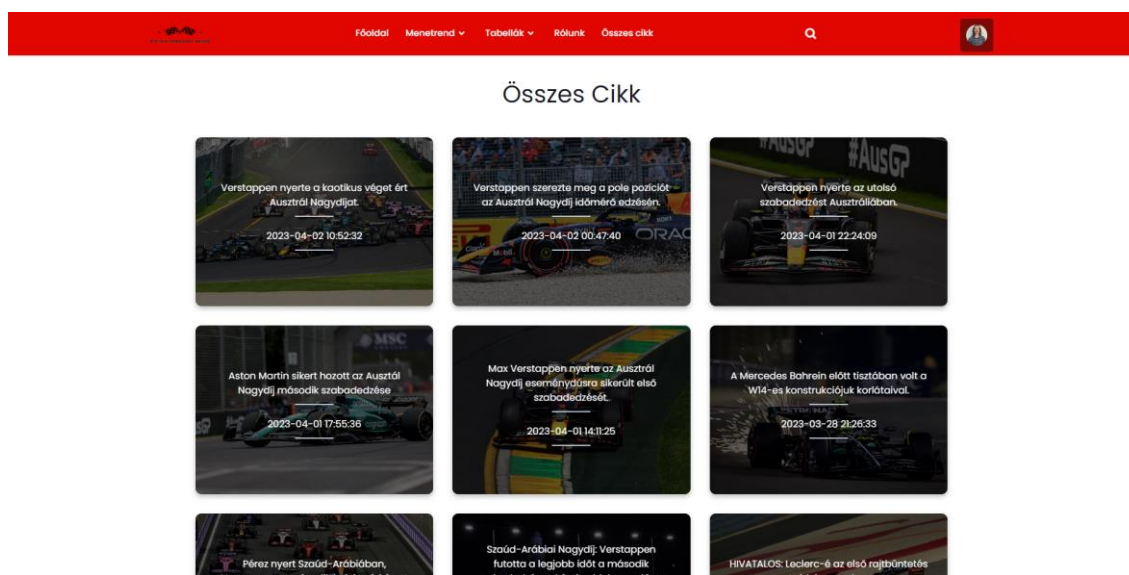


13) ábra Csapat tabella

## 1.7.Összes cikk oldala

Az összes cikk oldalára, a navigációs menüben található utolsó menüponttal lehet elnavigálni.

Ezen az oldalon az eddig felvitt cikkeket lehet megtalálni egy helyen időrendi sorrendbe állítva. Minden sorban 3 cikket jelenítünk meg, kis kártyák formájában. Ezen megtalálható a cikkek borítóképe, amire adtunk egy sötétítést. Erre azért volt szükség, mivel a cikk címét a képre írjuk ki és így könnyebben olvasható. Ha a felhasználó ráhúzza az egeret egy kártyákra, akkor az megfordul és a cikknek a leírása lesz látható. A leírás alatt található egy „Tovább a cikk oldalára” felirat, amire kattintva megnyílik a cikk oldala.



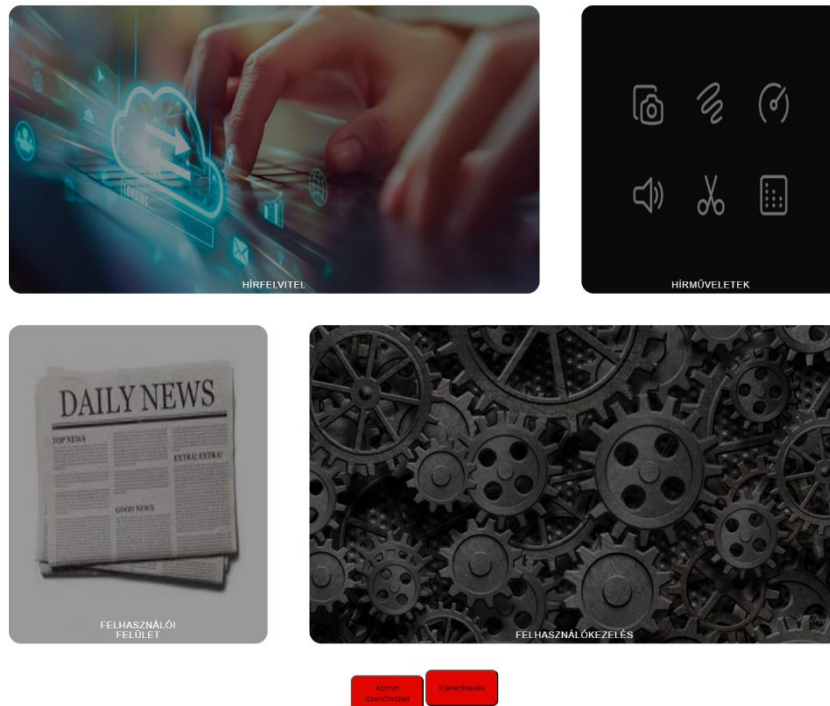
14) ábra Összes cikk oldal

## 1.8.Adminisztrációs felület

Ez az oldal csak az adminisztrátor joggal rendelkező felhasználóknak, azaz az újságíróknak elérhető. Az oldalon a többivel ellentétben nem található navigációs menü, mivel nem éreztük szükségesnek a használatát. Helyette négy darab panellel oldottuk meg a navigációt, amiknek különböző funkcióik vannak és e funkciók a képekre vannak ráírva, hogy az adott panel mire is szolgál.

A felső sor bal oldali paneljére rákattintva tud az újságíró híreket felvinni az oldalra. A mellette található panelen keresztül a tartalom kezelés felületére tudunk elnavigálni. Az alsó sor első panelje visszavisz minket a felhasználói felületre, vagyis a főoldalra, Az utolsó panel pedig egy olyan felületre visz minket, ahol a regisztrált felhasználókat lehet kezelni.

A panelek alatt közvetlenül található két gomb. A jobb oldali gomb egy chatszobába vezet, ahol a belépett újságírók tudnak egymással beszélgetni. A másik gombbal pedig ki lehet jelentkezni a fiókunkból.



15) ábra Adminisztrációs oldal

## 1.9.Cikkek felvitele

Ezen az oldalon keresztül tud az újságíró új cikkeket megírni és közzétenni a közönség számára.

Az oldal tetején látható egy címsor, ami utal arra, hogy melyik oldalon is vagyunk. Közvetlen alatta található egy figyelmeztetés, ami figyelmezteti az újságírót, hogy minden olyan mezőt töltsön ki, ami kötelező különben nem fogja tudni közzétenni a cikket.

Alatta az oldal közepére pozicionálva található egy űrlap, amiben a cikkhez szükséges adatok megadására van lehetőség. Ilyenek a borítókép felöltése, ami a főoldalon jelenik meg és erre kattintva tudja a felhasználó megnyitni a cikket. A következő beviteli mező, az ún. „alias” megadására szolgál.

Az „alias” arra szolgál, hogy a hír linkje a böngészőben egy rövidített elnevezés legyen, ne pedig egy php kód.

Következőnek az újságírónak meg kell adnia a cikk címét, amelyre a cím beviteli mező áll a rendelkezésére. Ennek utalnia kell a cikk tartalmára és olyannak kell lennie, hogy felkeltse az olvasó érdeklődését.

Az oldal fő egysége a tartalom felviteli mező. Itt az újságíró meg tudja fogalmazni a cikk tartalmát, vagy be is tudja másolni a már előre megírt szöveget. Az itt megírt cikket úgy tudja formázni, mintha egy Word dokumentumot formázná persze kevesebb opcióval. Ha esetleg ki akar emelni valamit, akkor félkövérré vagy dőlt betűssé tudja alakítani a szöveget. Emellett ha a cikkhez szeretne képet vagy videót csatolni arra is van lehetősége. Ezt egyszerűen a kép beszúrása menüpontra kattintva tudja megtenni. Videó esetében viszont szükség van egy kis HTML tudásra ugyanis videót csak `<iframe></iframe>` tag-ek között kell beágyazni.

A tartalom felvitele alatt közvetlenül található a leírás beviteli mező, ami arra szolgál, hogy egy rövid bevezetőt, leírást írjunk a cikkhez, ami a főoldalon jelenik meg a cím alatt közvetlenül.

Ha az újságíró úgy gondolta, hogy befejezte a cikk írását akkor tud választani a státusz menüpontban a között, hogy aktív vagy inaktív legyen a hír. Erre azért van szükség, mivel előfordulhat az, hogy egy cikket nem egyhelyben elsőre írnak meg, hanem ha úgy érzi az újságíró, hogy majd később szeretné befejezni, akkor nem szükséges neki publikusra beállítani. Így bekerül a felvitt cikkek közé, de csak az újságírók látják az olvasóknak nem fog megjelenni.

Ezek után szintén egy olyan beviteli mezővel találkozhatunk, ahol két lehetőség közül lehet választani. Itt tudjuk eldönteni, hogy a hír az vezető hír legyen vagy úgynevezett kishír, ami nem közöl olyan fontos dolgot, hogy a vezető hírek közé kerüljön.

Végül az űrlap alján található kettő gomb Egy „Rendben” és egy „Mégsem”. A „Rendben” gombra kattintva a munkánk mentésre kerül, és a cikk kikerül a nyilvánosság elé, illetve ha inaktív státuszt állítottunk be rá akkor csak a rendszerbe kerül felvitelre. Ha az újságíró úgy dönt, hogy nem teszi a cikket akkor ezt a „Mégsem” gombra kattintva megteheti és ekkor, amit eddig megírt nem kerül mentésre.

Az oldal két oldalán található két gomb. A bal oldali arra szolgál, hogy rákattintva az újságíró vissza tud lépni az adminisztrátor felületre, a jobb oldalival pedig ki tud lépni a fiókjából.

**Cikkek módosítása**

A \*al jelölt mezők kitöltése kötelező!

**TARTALOM**

Fotó feltöltése:

Alias\*:

Cím\*:

Tartalom\*: 

Formátum

Stílus

B I U S X
↶ ↷ ↻ ↺ ↻ ↺
↶ ↷ ↻ ↺ ↻ ↺

Stílus
Formátum
Beállítás
Méret

Kiderült, miért esett ki Charles Leclerc Ausztráliában

16) ábra Cikk felvitele

## 1.10. Cikkek kezelése















A cikkek kezelése oldalra az adminisztrátorfelületről tudunk elnavigálni a cikkek kezelése elnevezésű panelre kattintva.

Ezen az oldalon bal oldalt találhatunk három gombot különböző funkciókkal. Az első gombra rákattintva visszajutunk az adminisztrációs felületre, a második gombra kattintva egy új cikket vihetünk fel, az utolsó gombbal pedig ki tudunk jelentkezni a fiókunkból.

Az oldal közepén található egy táblázat, amiben az eddig felvitt cikkek jelennek meg. Ebben a táblázatban a cikkek legfontosabb adatait tároljuk el. Balról haladva az „alias”-át, a cikk címét és tartalmát, valamint a státuszát és azt, hogy vezető hír-e vagy nem. A táblázat utolsó oszlopa a különböző műveleteket tartalmaz. Ezek a cikk törlése és cikk módosítása. A hír törlésére egy piros kuka ikon utal. Erre rákattintva az oldal feldob egy figyelmeztető ablakot egy „Biztos benne, hogy kitörli?” üzenettel. A kuka ikon mellett látható egy kék ceruza ikon, amire ha rákattintunk, akkor a tartalom módosítása oldalon találjuk magunkat.

Ennek az oldalnak a kinézete megegyezik a cikk felvitele oldal kinézetével. Ha belépünk ide, akkor azt láthatjuk, hogy a beviteli mezők kitöltésre kerültek, hogy egyszerűbb legyen az újságíró dolga a módosításkor.



Cikkek szerkesztése						
FŐOLDAL	Alap	Cím	Tartalom	Státusz	Főhír	Művelet
ÚJ OLDAL LETREHÚZÁSA						
KILÉPÉS						
	meghibasodás	Kiderült, miért esett ki Charles Leclerc Bahreinben	Kiderült, miért esett ki Charles Leclerc Ausztráliában.	Nem elérhető	Kis hír	 
	trukk	Az Alfa Romeo csak szórakozott a padlólemezével az autóbemutatón	Az Alfa Romeo bemutatott padlólemeze teljes félrevezetés volt. &nb	Nem elérhető	Kis hír	 
	motorhiba	HIVATALOS: Leclerc-é az első rajtbüntetés az idei szezonban	Charles Leclercre legalább 10 helyes rajtbüntetés	Nem elérhető	Kis hír	 
	saudifpketto	Szaúd-Arábiai Nagydíj: Verstappen futotta a legjobb időt a második szabadedzésen két tizeddel megelőzve Alonsót.	Max Verstappen érte el a leggyorsabb időt a Forma-1 második szabadedzése	Elérhető	Kis hír	 
	szaudiverseny	Pérez nyert Szaúd-Arábiában, Verstappen a második helyig zárkózott	Noha a rajtnál egy kicsit megbotlott, Sergio Pérez magabiztosan nyerte a Szaúd-	Elérhető	Kis hír	 
	hibaskonstrukcio	A Mercedes Bahrain előtt tisztában volt a W14-es konstrukciójuk korlátaival.	A Mercedes elismerte, hogy nem váltak be a fejlesztéseik a 2023-as Forma-1-es autókoc	Elérhető	Kis hír	 
	australialfpegy	Max Verstappen nyerte az Ausztrál Nagydíj eseménydúsra sikerült első szabadedzését.	Két hét után újra Forma-1-es futamot rendeznek, a helyszín ezúszor	Elérhető	Kis hír	 
	australialfketto	Aston Martin sikert hozott az Ausztrál Nagydíj második szabadedzése	Az eseménydús első szabadedzést követően Alonso nyerte az esős folytat	Elérhető	Kis hír	 
	australialfpharm	Verstappen nyerte az utolsó szabadedzést Ausztráliában.	A Red Bullcs végzett az élen az időmérő előtti utolsó gyakorló	Elérhető	Kis hír	 
	australialfquall	Verstappen szerezte meg a pole pozíciót az Ausztrál Nagydíj időmérő edzésén.	Max Verstappen legyőzve a Mercedes párosát szerezte meg az első rajtkockát a v	Elérhető	Kis hír	 
	australialverseny	Verstappen nyerte a kaotikus véget ért Ausztrál Nagydíjat.	Max Verstappen nyerte a 27. alkalommal megrendezett Ausztrál Nagydíjat. Lewis Ham	Elérhető	Főhír	 













17) ábra Cikkek kezelése oldal

## 1.11. Felhasználók kezelése

Az oldalra az adminisztrációs felületen lévő felhasználók kezelése nevű panelről lehet eljutni. Ezen az oldalon valósítjuk meg a regisztrált felhasználók kezelését, illetve adatainak a módosításait. Pontosabban azt, hogy itt tudunk nekik adminisztrátori jogot adni, illetve elvenni tőlük.

Az oldal címe alatt itt is találunk kettő gombot csak úgy, mint az előző oldalakon. Az első gomb az adminisztrátor felületre vezet vissza, a második gombbal pedig ki tudunk jelentkezni a fiókunkból.

Az oldalon található egy táblázat, amiben a felhasználó adatai vannak megjelenítve. Az első oszlopban található a felhasználó profilképe, ha van, a második oszlopban van a regisztrációkor megadott email címe, a harmadik oszlopban a felhasználóneve, a negyedik oszlopban van a felhasználó jelszava, ami titkosított formában van eltárolva a rendszerben. A következő oszlopban a felhasználó neve van eltárolva, amit a regisztrációkor adott meg a mellette lévő oszlopban pedig a jogosultságot tároljuk el, hogy adminisztrátor az oldalon vagy nem. Az utolsó előtti sorban a felhasználó státusza van, ami annyit tesz, hogy itt van eltárolva, hogy a fiókja aktív-e vagy nem. Ha a státusz nem aktív egy adott fióknál, akkor azzal nem lehet többé belépni. Az utolsó oszlop itt is a műveleteknek van fenntartva. Itt tudjuk a felhasználónak módosítani a jogosultságát a kék ceruzára kattintva. A piros kuka ikonnal pedig a fiók státuszát tudjuk változtatni.

Felhasználók kezelése								
<div>FŐOLDAL KILÉPÉS</div>								
Photo	E-mail	Username	Password	Name	Permission	Statusz	Művelet	
	norbi@gmail.com	Norbi	2eb5878a0e31994055624bcaabface2f8bcf3862	Norbi	admin	Aktív felhasználó		
	home@gmail.com	Home	e83249bd3ba79932e16fb1fb5100dafade9954c2	Home	admin	Aktív felhasználó		
	mindegy@gmail.com	SebiFCB1899	c6efaf27673db4f7d2de52c0ab20a0655112cbad	Hives Sebastian	user	Aktív felhasználó		
	teszt@gmail.com	Teszt1	9ca395ace8a7f758044bdcd8642f8fae49687448	Teszt Teszt	user	Inaktív felhasználó		

18) ábra Felhasználók kezelése oldal

## 1.12. Adminisztrátor felvétele

Az előző, felhasználók kezelése oldalon található kék ceruza ikonnal lehet erre az oldalra eljutni. Az oldalon alaphoz beolvasásra kerül annak a fióknak a regisztrációkor megadott email címe, amelyiknek a jogosultságát módosítani akarjuk. Alatta található egy lenyitható beviteli mező, amelyben kétfajta jogosultság közül lehet választani. Ha jóvá szeretnénk hagyni a módosításokat, azt a legalsó „OK” gombbal tehetjük meg. Ekkor a változtatások elmentésre kerülnek és visszakerülünk a felhasználók kezelése oldalra. Az oldal alján itt is megtalálható a lábléc.

## 1.13. Adminisztrátor, felhasználó törlése

Ez a folyamat a felhasználók kezelése oldalon a táblázatban a felhasználók melletti piros kuka ikonra kattintva megy végbe. Nem irányít át külön oldalra, hanem a háttérbe fut le.

## 1.14. Adminisztrátori chat

Az oldalunkon megvalósítottunk egy adminisztrátorok közötti üzenő falat, amin keresztül az újságírók tudnak egymással kommunikálni. Ennek célja az, hogy egy minden adminisztrátor által elérhető felületet hozzunk létre, olyan esetekre, ha valakinek

valamilyen kérdése merül fel bármivel kapcsolatban, akkor egyszerűen meg tudják beszélni a problémáikat.

A felület kinézete az egész oldalon használt színeket használja élénkebb megjelenítésben. Az oldal legnagyobb részét maga az üzenő felület teszi ki. Ez alatt középen található egy beviteli mező, ahol megfogalmazhatjuk az üzenetünket. Az elküldéshez két lehetőség van. Található a beviteli mező mellett jobbra egy „Add” feliratú gomb, amire kattintva elküldhetjük az üzenetünket. Ezen felül egyszerűen az ENTER billentyű megnyomásával is megtehetjük ugyanezt. Nevet nem kell beírunk, hogy milyen néven szeretnénk elküldeni az üzenetünket, mivel a felhasználónevünk egyből beolvasásra kerül.

## **2. Fejlesztői dokumentáció**

A projektünket a Visual Studio Code nevű programban fejlesztettük javarészt, mivel ezt a programot már jól megismertük a tanulmányink során így ezt találtuk a legkézenfekvőbbnek. A weboldalunk felépítéséhez a HTML5-öt (Hypertext Markup Language revision 5) használtuk.

Formázás céljából külső csatolt CSS (Cascading Style Sheet) fájlt használtuk.

Ezek mellett, egy adatbázist is terveztünk mivel ez az egyik legfontosabb része a munkánknak. A weboldal az adatbázissal való összekapcsolódását és a szerver oldali programozást a PHP programnyelv segítségével valósítottuk meg.

### **2.1. Fejlesztői környezet**

#### **2.1.1. Visual Studio Code**

A Visual Studio Code vagy röviden VS Code egy forráskód fejlesztő program, amit a Microsoft fejlesztett az Electron keretrendszerrel. A Microsoft ezt a programot kiadta Windowsra, Linuxra és MACOS-re is. A funkciói közé tartozik a hibakeresés támogatása, a szintaxis kiemelése, az intelligens kódkiegészítés, a kódrészletek, a kód újrafeldolgozása és a beágyazott Git. A felhasználók módosíthatják a témát, a billentyűparancsokat, a beállításokat, és olyan bővítményeket telepíthetnek, amelyek további funkciókat adnak hozzá.

Azért a Visual Studio Code-ot választottuk fejlesztői környezetnek mivel, egyrészt ezt a programot használtuk a tanulmányaink során, másrészt pedig rendkívül egyszerű a használata, mivel számos különböző olyan kiegészítőt adhatunk hozzá, amelyek jelentősen lerövidítik a programozás folyamatát, és felhívják a figyelmünket az esetleges hibákra. Ilyenek például: Emmet, Auto Rename Tag, Indent Rainbow, Open PHP stb.

Ezek mellett, ha látni szerettük volna azt, hogy miközben programozunk, hogyan változik a weboldalunk, akkor csak az előzetesen telepített Live Server modult elindítottuk, és nem kellett folyamatosan újra megnyitni az oldalt, hanem elég volt csak frissíteni, ami szerintünk szintén egy nagyon hasznos funkciója a programnak.

#### **2.1.2. CSS**

Cascading Style Sheet vagy röviden CSS egy külső stíluslap, amit arra használunk, hogy olyan dokumentumoknak a kinézetét, dizájnját formázzuk meg, amik jelölő nyelven íródtak, mint a HTML és XML dokumentumok. A CSS egy fő támasz technológiája a Word Wide Web-nek (WWW) a HTML és a JavaScript mellett. Azért

használtunk CSS-t, mivel az ilyen külső stíluslapokban elég egyszer megírni a kódot és csak csatolni utána a HTML fejrészében így lényegesen megkönnyítve a dolgunkat és lerövidítve a fejlesztési időt.

A CSS legfőbb előnye, hogy el lehet vele szeparálni a HTML kódot a CSS kódtól ezzel megkönnyítve a későbbiekben a karbantartást végző fejlesztő dolgát is. Míg egy ún. inline stílus alkalmazása esetén minden azonos elemet egyesével kellene módosítani, a CSS-ben a HTML-ben megadott osztályokat, és id-eket szelektálhatunk ki, meg adhatunk nekik tulajdonságokat értékekkel. Tehát rengeteg időt spórolhatunk így, hiszen nem kell mindent, amit formázni szeretnénk többször leírni, mint az inline stílus esetében, hanem elég csak egyszer. Csak a CSS-ben tudunk „media query-eket” hozzáadni, amelyek segítségével különböző képernyő méretekre tervezhetjük meg a weboldalunk megjelenését. A külső CSS előnye az is, hogy eltárolásra kerül a böngészőben, így az újabb oldal letöltésnél, már sokkal gyorsabban töltődik be.

### **2.1.3. Bootstrap**

A CSS-el párban használtunk a projektünk elkészítéséhez és tervezéséhez Bootstrap-et is. Azért döntöttünk amellett, hogy használjuk, mert nagyon megkönnyítette a tervezési folyamatot mivel nem kellett mindent a nulláról lekódolnunk maximum csak átalakítani úgy, ahogyan mi szerettük volna, hogy megjelenjen. Például a főoldalon látható „carousel” is egy ilyen elem volt. Nagy előnye a Bootstrapnak, hogy rengeteg komponens meg van előre írva, ami nagymértékben meggyorsítja a tervezési folyamatot, csupán a megfelelő osztály neveket kell hozzá adnunk a HTML elemeinkhez.

### **2.1.4. PHP**

A PHP egy általános szerveroldali szkriptnyelv, amit dinamikus weblapok készítéséhez használunk. Az első szkriptnyelvek egyike, amely külső fájl használata helyett HTML oldalba ágyazható. A projektünkben a PHP-nak az egyik legfontosabb alkalmazása az adatbázis hozzacsatolása a weboldalhoz illetve arra, hogy a weblapunkon az adatbázisból kinyert információkat, adatokat dinamikusan tudjuk megjeleníteni. Mivel a mi projektünk egy online újság ezért ez a nyelv nagyon hasznos volt számunkra, ugyanis ennek a segítségével elég volt mindössze egyszer megírni a PHP-ban a kinézetét a híreknek. Ezek után már csak egy egyszerű kiírtatással jelenítettük meg a további híreket elérve ezzel azt, hogy minden hír egységesen ugyan úgy jelenjen meg a weboldalon.

### **2.1.5. JavaScript**

A JavaScript, amelyet gyakran JS-nek is szoktak rövidíteni, egy programozási nyelv, amely a HTML és a CSS mellett a világháló egyik alapvető technológiája. 2022-től a webhelyek 98%-a JavaScriptet használ az ügyféloldalon a weboldal viselkedéséhez, gyakran harmadik féltől származó könyvtárakat is magában foglalva. Minden fejlett webböngésző rendelkezik egy dedikált JavaScript-motorral, amely végrehajtja a kódot a felhasználók eszközein.

A projektünk során leginkább látványos effektek beépítésére, megvalósítására alkalmaztuk, mint például a főoldalon található visszaszámláló.

### **2.1.6. WinSCP**

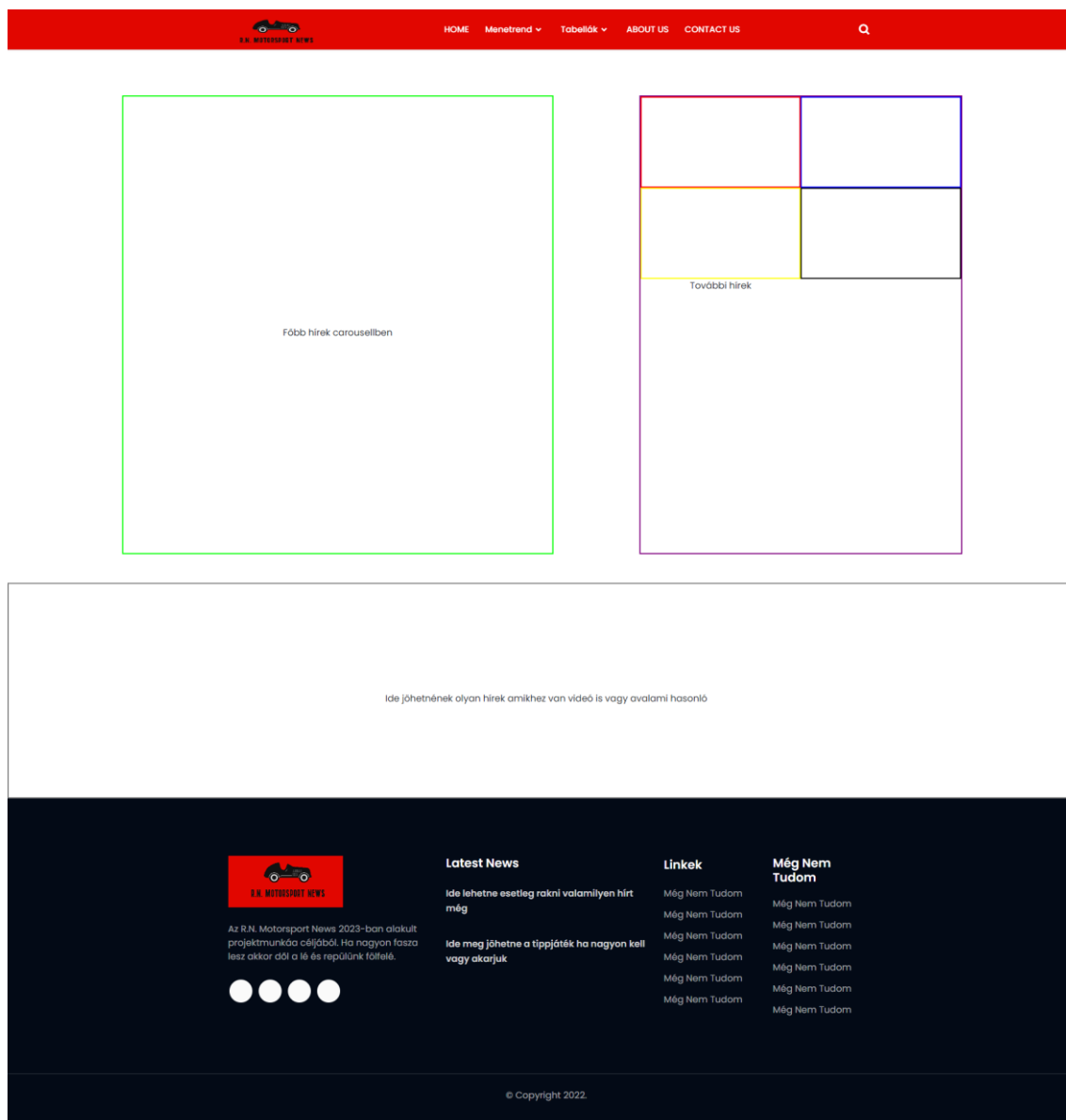
A WinSCP egy olyan program melynek segítségével meg tudtuk azt oldani, hogy mind a ketten hozzá tudjunk férni a projektnek a fájljaihoz. Azért is használtuk ezt, mivel véleményünk szerint egyszerűbb a használata, mint a Githubnak. Mind a kettőnk letöltötte a programot és a fájlokat ott tároljuk, ezáltal ott helyben tudjuk szerkeszteni őket és így mind a kettőnknek megvan egyből a módosított forma. Nem kell egymásnak küldözgetni a fájlokat, vagy nem kell „commit”-olni, és „merge”-lni azaz összefűzni a különböző verziókat.

### **2.1.7. Nethely**

A Nethely egy online weboldal ahol létrehoztunk egy FTP tárhelyet, amit a WinSCP segítségével kezelünk. Leginkább azért használjuk, mert így egyből meg tudjuk nézni online a változtatásokat. A másik nagy előnye számunkra az, hogy a responzivitást is egyszerűbb ellenőrizni, mivel az oldal fent van az interneten, tehát telefonról is be tudunk lépni.

## **2.2. Tervezés folyamata**

Miután megvolt az alapötletünk nekilátunk a tervezésnek. Ehhez először egy vázlat oldalt készítettünk melybe nem írtunk és tettünk semmit csak annyiból állt, hogy a különböző elemeknek a körvonalait rajzoltuk ki az oldalra. Ez a folyamat a WinSCP használatával nagyon egyszerű volt, mivel mind a ketten egyből láttuk. ha a másikunk csinált valamit és egyből meg tudtuk beszélni a dolgokat, amiket változtatni akartunk.



19) ábra Kezdetek

## 2.3. Lapvédelem

Lapvédelmet alkalmaztunk valamennyi oldalon. Ezt annak érdekében tettük, hogy csak adminisztrátorként regisztrált felhasználók tudjanak belépni az adott oldalra. Ilyenek például a hír felvétele és módosítása oldalak. Viszont nem mindegyik oldalon van lapvédelem annak érdekében, hogy az átlag felhasználó is tudja olvasni a híreket például.

### A lapvédelem kódja:

```
if(!isset($_SESSION["belepett"]))
{header("Location:../false.html"); }
```

Ha valaki belépés nélkül próbál az adminisztrátor felületre bejutni, akkor azonnal átirányításra kerül egy másik oldalra, ahol egy figyelmeztető üzenet jelenik meg.

## 2.4. Adatbázis

Az adatbázisunk eredeti célja az volt, hogy az újságírók által felvitt híreket, cikkeket eltároljuk. Mivel lehet regisztrálni az oldalra ezért a felhasználók adatait is eltároljuk és az adatbázis segítségével döntjük el ki újságíró és ki nem. Ahogyan a projektünk fejlődött úgy szükségessé vált, egy második adatbázist is létrehoznunk. Erre azért volt szükség, mert szeretnénk volna a világbajnokságok állását is megjeleníteni az oldalon, ezért további táblák felvételére volt szükség, amit nem tudtunk összekapcsolni a már meglévő táblákkal.

### 2.4.1. Egyedek meghatározása

Kiinduló egyedeink:

- Felhasználók: A beregisztrált felhasználók.
- Hozzászólások: Az elküldött hozzászólások.
- Futamok: Az idényben megrendezésre kerülő futamok.
- Csapatok: A sorozatban résztvevő csapatok.
- Pilóták: Az összes pilóta.

### 2.4.2. Kapcsolatok:

**drivers => teams:** N:M-es kapcsolat áll fent közöttük, mivel egy pilóta egy szezonban több csapatnál is versenyezhet és egy csapatnak több pilótája van. Ezért ezt a kapcsolatot két 1: N-es kapcsolatra kellett bontanunk egy kapcsolótábla segítségével. Ez a kapcsolótábla a 'fullteams' elnevezésű tábla.

**drivers=>fullteams:** 1:N-es kapcsolatot van közöttük. A 'drivers' tábla kapcsolódik a 'fullteams' kapcsolótáblához.

**teams=>fullteams:** 1:N-es kapcsolatot van közöttük. A 'teams' tábla kapcsolódik a 'fullteams' kapcsolótáblához.

**teams=>races:** N:M-es kapcsolat áll fenn közöttük, mivel egy futamon több csapat is részt vesz valamint egy csapat több futamon is részt vesz egy szezonban. Ezt szintén fel kellett bontani két 1: N-es kapcsolatra, amihez ismételtén egy kapcsolótáblát használtunk. Ez a kapcsolótábla a 'participants' elnevezésű tábla.

**teams=>participants:** 1:N-es kapcsolatot van közöttük. A 'teams' tábla kapcsolódik a 'participants' kapcsolótáblához.

**races=>participants:** 1:N-es kapcsolatot van közöttük. A 'races' tábla kapcsolódik a 'participants' kapcsolótáblához.

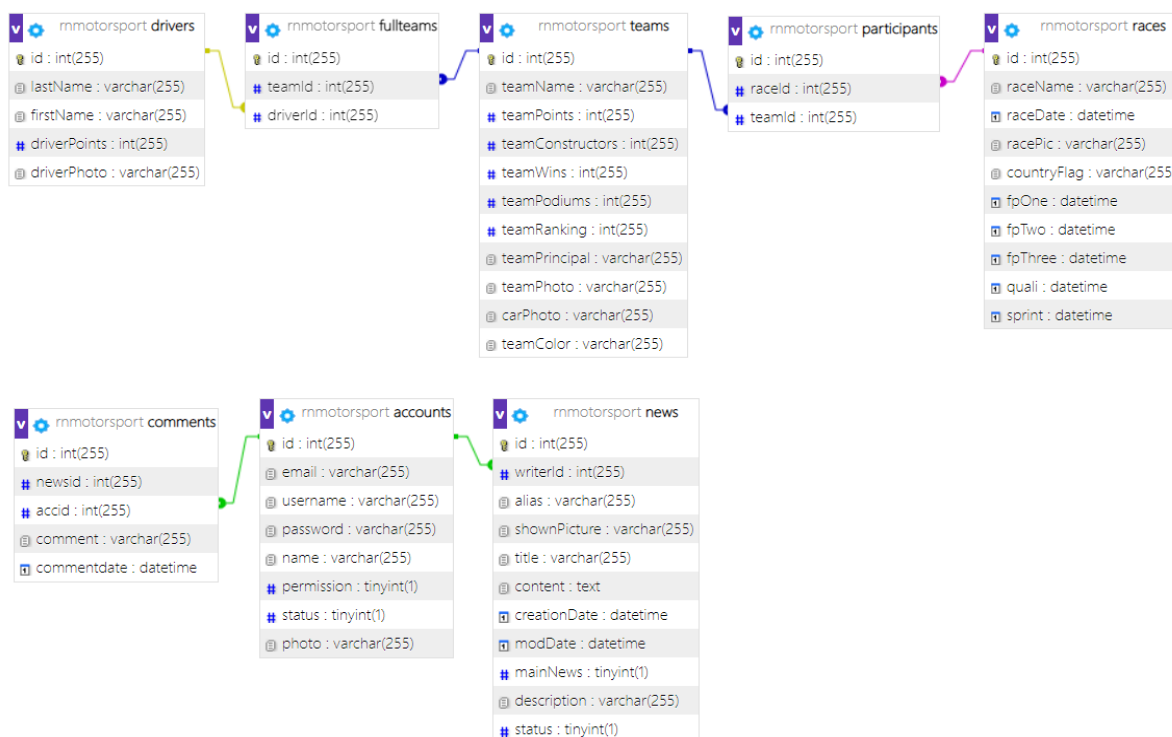


**accounts=>comments:** 1:N-es kapcsolat van közöttük, mivel egy hozzászólást csak egy fiókról lehet elküldeni, viszont egy fiókról lehet több kommentet is írni.

**news=>accounts:** 1:N-es kapcsolat van közöttük, mivel egy cikket csak egy fiókról lehet felvinni az oldalra, viszont egy fiókkal több cikk is felvihető.

### 2.4.3. Táblák

A weboldalunkon minden adat megjelenítése adatbázisból történik. Két adatbázisunk van. Az egyik 5 a másik 3 táblából áll.



20) ábra Teljes adatbázis

#### accounts:

A felhasználók adatainak tárolására szolgál.

Név	Típus	Leírás
Id	int (11)	elsődleges kulcs
email	varchar (255)	email címe a felhasználónak.
username	varchar (255)	a felhasználó felhasználónevei
password	varchar (255)	a felhasználó jelszavai
name	varchar (255)	a felhasználó teljes neve.
permission	tinyint (1)	jogosultság aminek két értéke lehet true vagy false

status	tinyint (1)	a fiók státusza ami két értéket vehet fel true vagy false
photo	varchar (255)	a felhasználó profilképe

id	email	username	password	name	permission	status	photo
22	norbi@gmail.com	Norbi	2eb5878a0e31994055624bcaabface2f8bcf3862	Norbi	1	1	1679577918.jpeg
23	home@gmail.com	Jotato	e83249bd3ba79932e16fb1fb5100dafade9954c2	Home	1	1	1680109756.jpeg
27	mindegy@gmail.com	SebiFCB1899	c6efaf27673db4f7d2de52c0ab20a0655112cbad	Híves Sebastian	0	1	1679578196.jpeg

21) ábra Accounts tábla

### comments:

Az oldalon elküldött hozzászólásokra tárolására szolgál.

Név	Típus	Leírás
Id	int (11)	elsődleges kulcs
newsId	int (255)	idegen kulcs, ami a hozzászólások táblához köti hozzá a hírek táblát
accId	int (255)	idegen kulcs, ami a bejelentkezettek táblát kapcsolja össze a hozzászólások táblával
comment	varchar (255)	a komment szövege
commentDate	datetime	ekkor hozták létre

id	newsid	accid	comment	commentdate
58	30	23	SAINZ MEGVERI 6T	2023-03-07 00:00:00
78	52	22	Jajj ne!	2023-03-21 00:00:00
79	52	23	ez nem lesz jó	2023-03-21 00:00:00

22) ábra Comments tábla

### news:

Az oldalra felvitt hírek, cikkek tárolására szolgál.

Név	Típus	Leírás
Id	int (255)	Elsődleges kulcs
writerId	int (255)	idegen kulcs, ami a hírek táblát kapcsolja össze a bejelentkezettek táblával
alias	varchar (255)	egy szó, ami megjelenik az URL-ben
shownPicture	varchar (255)	cikk borítóképe

title	varchar (255)	cikk címe
content	text	cikk tartalma
creationDate	datetime	a cikk létrehozásának dátuma
modDate	datetime	a cikk utolsó módosításának dátuma
mainNews	tinyint (1)	két értéket vehet fel 1 vagy nulla, ez által döntjük el, hogy hol jelenjen meg a cikk
description	varchar (255)	rövid leírás a cikkről
status	tinyint (1)	két értéket vehet fel 1 vagy 0, ez által kerülnek ki a cikkek

id	writerId	alias	shownPicture	title	content	creationDate	modDate	mainNews	description	status
30	22	meghibasodas	1678137194.png	Kiderült, miért esett ki Charles Leclerc Bahreinbe...	<p>Kiderült, miért esett ki Charles Leclerc Bahrei...	2023-03-06 22:05:21	2023-03-06 22:13:14	0	Így akár az is benne lehet a pakiban, hogy a máso...	1
37	22	trukk	1678536814.png	Az Alfa Romeo csak szórakozott a padlólemezével az...	<p></p><p><b>Az Alfa Romeo bemutatott padlólemeze ...	2023-03-11 12:55:24	2023-03-11 13:13:34	0	Valójában sosem akartak ilyen megoldással versenye...	1
39	22	motorhiba	1678920372.png	HIVATALOS: Leclerc-é az első rajtbüntetés az idei ...	<p><p>Charles Leclerc legalább 10 helyes rajtbün...	2023-03-15 23:46:12	0000-00-00 00:00:00	0	A Ferrari versenyzője Charles Leclerc 10 rajthelye...	1

23) ábra News tábla

### teams:

A csapatok adatainak tárolására szolgál.

Név	Típus	Leírás
Id	int (255)	Elsődleges kulcs
teamName	varchar (255)	a csapat neve
teamPoints	int (255)	a csapat összes pontja ebben a szezonban
teamConsturctors	int (255)	a csapat összes bajnoki címe
teamWins	int (255)	a csapat összes győzelme az eddigi szezonig
teamPodiums	int (255)	a csapat összes dobogós helyezése az eddigi szezonig
teamsRanking	int (255)	a csapat tavalyi bajnoki eredménye
teamPrincipal	varchar (255)	csapatfőnök neve
teamPhoto	varchar (255)	csapat logója
carPhoto	varchar (255)	csapat autójáról egy rajz
teamColor	varchat (255)	a csapat színe

id	teamName	teamPoints	teamConstructors	teamWins	teamPodiums	teamRanking	teamPrincipal	teamPhoto	carPhoto	teamColor
1	Mercedes AMG Petronas F1 Team	38	8	125	281	3	Toto Wolff	mergalogo.jpg	mercedesrajz.png	black
2	Scuderia Ferrari	26	16	241	798	2	Frederic Vasseur	ferrari2.jpg	ferrari2.jpg	#930418
3	Oracle Red Bull Racing	87	5	92	234	1	Christian Horner	redbullfekete.jpg	redbullrajz.png	#13152c

24) ábra Teams tábla

### drivers:

A pilóták adatainak tárolására szolgál.

Név	Típus	Leírás
Id	int (255)	Elsődleges kulcs
firstName	varchar (255)	a pilóta vezetékneve
lastName	varchar (255)	a pilóta keresztnéve
driverPoints	int (255)	a pilóta idei pontjai
driverPhoto	varchar (255)	a pilóta fotója

id	lastName	firstName	driverPoints	driverPhoto
1	Hamilton	Lewis	20	lajos.jpg
2	Russell	George	18	russell.jpg
3	Norris	Lando	0	norris.jpg

25) ábra Drivers tábla

### races:

A versenyek adatait tárolja

név	típus	Leírás
Id	int (255)	elsődleges kulcs
raceName	varchar (255)	a futam neve
raceDate	date time	a futam időpontja
racePic	varchar (255)	a futamról egy kép
countryFlag	varchar (255)	az ország zászlaja ahol a futam van
fpOne	datetime	elsősabadedzés időpontja
fpKetto	datetime	másodikszabadedzés időpontja
fpHarom	datetime	harmadikszabadedzés időpontja
quali	datetime	időmérő időpontja
sprint	date	sprintfutam időpontja

id	raceName	raceDate	racePic	countryFlag	fpOne	fpTwo	fpThree	quali	sprint
1	Bahraini Nagydíj	2023-03-05 16:00:00	../kepek/palyak/bahrain.png	bahrain.png	2023-03-03 12:30:00	2023-03-03 16:00:00	2023-03-04 12:30:00	2023-03-04 16:00:00	0000-00-00 00:00:00
2	Szaudi Nagydíj	2023-03-19 18:00:00	../kepek/palyak/saudi.png	saudi.png	2023-03-17 14:29:00	2023-03-17 18:00:00	2023-03-18 14:30:00	2023-03-18 18:00:00	0000-00-00 00:00:00
3	Ausztrál Nagydíj	2023-04-02 07:00:00	../kepek/palyak/australia.png	australia.png	2023-03-31 03:30:00	2023-03-31 07:00:00	2023-04-01 03:30:00	2023-04-01 07:00:00	0000-00-00 00:00:00

26) ábra Races tábla

### participants:

Kapcsolótábla, ami a Teams és a Races táblákat kapcsolja össze.

név	típus	Leírás
Id	int (255)	elsődleges kulcs
teamId	int (255)	idegen kulcs, ami összeköti a 'participants' táblát a 'teams' táblával.
raceId	int (255)	idegen kulcs, ami összeköti a 'races' táblát a 'participants' táblával

id	racelId	teamId
1	1	1
2	1	2
3	1	3

27) ábra Participants tábla

### fullteams:

Kapcsolótábla, ami a Teams és a Driver táblákat kapcsolja össze.

név	típus	Leírás
Id	int (255)	elsődleges kulcs
teamId	int (255)	idegen kulcs, ami összeköti a 'teams' táblát a 'fullteams' táblával
driverId	int (255)	idegen kulcs, ami összeköti a 'drivers' táblát a 'fullteams' táblával

id	teamId	driverId
1	1	1
2	1	2
9	2	9

28) ábra Fullteams tábla

#### 2.4.4. Információk megszerzése

Miután meghatároztuk az egyedeinket és a táblákat létrehoztuk ezeket adatokkal kellett feltölteni. Ez viszont nem minden esetben szükséges mivel a hozzászólások és a cikkek nem előre vannak feltöltve az adatbázisba, hanem abban a pillanatban kerülnek bele, amikor az oldalon publikálásra kerülnek.

Azokban az esetekben, amikor előre kellett feltöltenünk az adatokat az internetet használtuk segítség gyanánt. Ilyen táblák voltak a Pilóták, Csapatok, Futamok. Az ehhez szükséges információk megszerzésére a Forma-1 hivatalos oldalát, illetve a GP racing stats nevű oldalt használtuk.

Amikor a szükséges információink megvoltak ezeket felvittük az adatbázisba.

#### 2.5. Főoldal

A fő oldalunkon a piros a domináló fő szín, mivel ez a színe a Forma-1 világának is. Megnéztünk számos hasonló weboldalt, és így találtuk a hozzá harmonizáló sötét szürkés-kék árnyalatot, amelynek a színkódját, a böngésző fejlesztői eszköztárának pipettájával másoltuk le.

A menü elkészítésére a Bootstrap sablonját használtuk. Először úgy gondoltuk, hogy jól mutatna egy dupla vagy tripla menüsáv, ahol a menü elemek egymás mellett, és alatt helyezkednek el. El is készítettük, de a végeredménnyel nem voltunk elégedettek, ezért ezt az ötletet elvetettük, és helyette egy sokkal látványosabb megoldásra jutottunk. Ha egy menüpontra a felhasználó ráhúzza az egeret, akkor egy lenyíló oldal, vagy tartalomtól függően fél oldal jelenik meg a főoldalra rácsúsztatva, ahonnan választhat a látogató a további lehetőségek közül.

Ezt úgy oldottuk meg, hogy amikor az egér a menüpontra érkezik, akkor hozzáad a program egy új osztályt, melynek köszönhetően megjelenik a tartalom. Ha lehúzzuk róla az egeret, akkor az osztály eltávolításra kerül, így a megjelent tartalom is eltűnik.

##### Ennek a kódja:

```
let tabellaelem = document.getElementById("tabellaelem");
let almenuu2 = document.getElementById("almenuu2");
let menetelem = document.getElementById("menetelem");
let almenuu = document.getElementById("almenuu");
Id alapján kiválasztjuk az elemeket, és eltároljuk egy-egy változóba.
const x = window.matchMedia("(max-width: 925px)");
```

Az x változóba eltárolunk egy adott szélességet.

```
function myFunction(x) {  
  if (x.matches) {  
    menetelem.innerHTML = `    taebllaelem.innerHTML = `
```

A függvénynek megadjuk paraméterként az előbbiekben deklarált x változót. A függvényen belül egy if-else szerkezetben megvizsgáljuk, hogy a képernyő szélessége megegyezik-e a változóban megadott szélességgel. Ha igen, akkor az előbb kiválasztott elemek csak linkeként funkcionálnak.

```
  } else {  
    tabellaelem.addEventListener("mouseenter", TabellaVan);  
    tabellaelem.addEventListener("mouseleave", TabellaNincs);  
    almenuu2.addEventListener("mouseenter", TabellaVan);  
    almenuu2.addEventListener("mouseleave", TabellaNincs);
```

Ha a változóban megadott értéktől szélesebb a kijelző mérete, akkor a Tabellák menüpontra hozzárendel egy függvényt, amint egy felhasználó ráhúzza az egeret. Amikor a felhasználó lehúzza a kurzort a menüpontról akkor pedig egy másik függvényt.

```
function TabellaVan() {  
  almenuu2.classList.add("almenulat2");};
```

Ezzel a függvénnyel hozzáadásra kerül egy osztály, amikor a kurzort ráhúzzuk a menüpontra. Ennek az osztálynak az a szerepe, hogy a menüsáv alatt megjeleníti az almenüt, mivel annak a display tulajdonságát block típusúra állítjuk, ami eredetileg display: none, vagyis nincs megjelenítve az elem.

```
function TabellaNincs() {  
  almenuu2.classList.remove("almenulat2");};
```

A TabellaNincs() függvény akkor kerül meghívásra, amikor a felhasználó lehúzza a kurzort az almenüről, akkor eltávolításra kerül az előbbiekben hozzáadott osztály.

```
  menetelem.addEventListener("mouseenter", MenetVan);  
  almenuu.addEventListener("mouseenter", MenetVan);  
  menetelem.addEventListener("mouseleave", MenetNincs);  
  almenuu.addEventListener("mouseleave", MenetNincs);
```

Abban az esetben, ha az x változóban eltárolt értéktől szélesebb a kijelző mérete, amikor a Menetrendek menüpontra a felhasználó rátolja az egeret, akkor lefut a „MenetVan” függvény. Amikor a felhasználó lehúzza az egeret a menüpontról az

eseményfigyelő érzékeli, hogy az egér elhagyta a területet és meghívja a „MenetNincs” függvényt.

```
function MenetVan() {  
  almenuu.classList.add("almenulat");};
```

Ezzel a függvénnyel hozzáadunk egy osztály az előbb említett Menetrendek menüponthoz. Az osztály tulajdonságait és értékeit meghatároztuk a CSS-ben, hogy a menüsáv alatt megjeleníti az almenüt vagyis a display tulajdonságát block típusúra állítjuk, amely eredetileg ez display: none.

```
function MenetNincs() {  
  almenuu.classList.remove("almenulat");};
```

Ennek a függvénynek a segítségével, amikor a felhasználó lehúzza az egeret az almenüről, akkor eltávolítjuk az előbbieken hozzáadott osztály.

```
myFunction(x)  
x.addListener(myFunction);
```

Az „x” változóhozott hozzárendelt esemény figyelő, meghívja a „myFunction” függvényt, amikor az oldal betöltésre kerül.

```
const panels = document.querySelectorAll('.panel')
```

Létrehoztunk egy „panels” nevű konstanst, melybe eltároltuk az összes panel osztályú elemet.

```
panels.forEach(panel => {  
  panel.addEventListener('click', () => {  
    removeActiveClasses()  
    panel.classList.add('active'); }) })
```

Egy forEach ciklus segítségével arra a sávra, amire rákattintottunk hozzá adunk egy active nevű osztályt, melynek hatására a sáv kinyílik és a csapat adatai láthatóvá válnak. Eközben a többről pedig eltávolítja az active osztályt.

```
function removeActiveClasses() {  
  panels.forEach(panel => {  
    panel.classList.remove('active'); })}
```

A removeActiveClasses() függvény segítségével az eddigi kinyílt sávról eltávolítjuk az active osztályt abban a pillanatban, amikor egy másik sávhoz hozzá adjuk. Ezzel a módszerrel egyszerre csak egy sáv kapja meg az active osztály.



### 2.5.1. Visszaszámláló megjelenítése a kezdő oldalon

A dátum és idő megjelenítésénél, szeretnénk volna jobban felhívni a figyelmet a másodpercek változására, amit az idő háttér színének folyamatos megváltoztatásával terveztük elérni. Ezt úgy valósítottuk meg a JavaScripttel, hogy a másodperceket maradékos osztással megvizsgáltuk, hogy páros e, vagy páratlan. Egy if elágazás feltételeként azt adtam meg, hogyha a másodperc páros, akkor a html elementhez adja hozzá a háttér nevű osztályt, amelyet a css-ben már meghatároztam előre. Így a szürke-fekete szín átmentes háttér, minden másodperben megváltozik fekete szürkére, de csak a másodpercek mögött.

Ezt az animációt úgy gondoltuk jó lenne a percek esetében is beállítani, de abba a problémába ütköztünk, hogy az animáció folyamatosan váltogatni kezdte a háttér színt. Ezt úgy sikerült végül megoldanunk, hogy megvizsgáljuk, hogy a perc az egyenlő-e 59-el, és ha igen, akkor az elem kap egy „hatter” nevű osztályt, amely a szám felső felét sötétszürkére, az alsó felét világosszürkére váltja át. Ez ugyanígy működik az órák, és napok esetében is.

#### Ennek a kódja:

```
const countdown = document.querySelector(".szamlalogrid");
```

Kiszelektáltuk a „szamologrid” osztállyal ellátott elemet és eltároltuk egy „countdown” nevezetű konstansba.

```
const kovfutamido = document.getElementById("kovfutamido").innerHTML;
```

Következő lépésként eltároltuk a következő futam az időpontját.

```
const interval = setInterval(() => {  
  const deadline = new Date (kovfutamido).getTime();
```

Ezen a ponton megadtuk, hogy meddig számoljon vissza a program az aktuális időponttól.

```
const current = new Date();
```

Eltároltuk az aktuális dátumot.

```
const diff = deadline - current;
```

Ezt követően már ki tudtuk vonni a határidőből a jelenlegi dátum értékét, és ezt a különbséget tároltuk el.

```
const days = Math.floor(diff / (1000 * 60 * 60 * 24)) + "";  
const hours = Math.floor((diff / (1000 * 60 * 60)) % 24) + "";  
const minutes = Math.floor((diff / (1000 * 60)) % 60) + "";
```

```
const seconds = Math.floor((diff / 1000) % 60) + "";
```

Az előző négy sorban kiszámoltuk a hátralévő napokat, órákat, percekét és másodpercekét.

```
countdown.innerHTML = `  
  <div data-content="Nap" class="szin" id="napszam">${days.length === 1 ?  
  `0${days}` : days}</div>  
  <div data-content="Óra" class="szin" id="oraszam">${hours.length === 1 ?  
  `0${hours}` : hours}</div>  
  <div data-content="Perc" class="szin" id="percszam">${  
    minutes.length === 1 ? `0${minutes}` : minutes  
  }</div>  
  <div data-content="Másodperc" class="szin" id="mpszam">${  
    seconds.length === 1 ? `0${seconds}` : seconds  
  }</div>`;
```

Összefűztük a napot, az órát, a percet és a másodpercet és ezekkel az értékekkel módosítottuk a „countdown” változó értékét, mely folyamatosan renderelésre kerül a weboldalon.

```
if (diff < 0) {  
  clearInterval(interval);  
  countdown.innerHTML = "Kedves olvasóink kapcsolják be biztonsági öveiket!";  
  document.getElementById("hh").innerHTML = "";};
```

Megvizsgáltuk, ha a két dátum különbsége eléri a nullát, akkor a fenti kódban olvasható üzenet kerül megjelenítésre, amikor a futam elrajtol.

```
let mpszam = document.getElementById('mpszam');  
if(mpszam.innerHTML % 2 == 0)  
{ mpszam.classList.add("hatter");}
```

Páros számok esetében a dobozok háttérének a színeit felcserétük a másodperc esetében egy új osztály hozzáadása segítségével.

```
let percszam = document.getElementById('percszam');
```

A percszam azonosítóval ellátott elemet eltávolítjuk egy „percszam” nevű változóban.

```
if(mpszam.innerHTML == 59)  
{ percszam.classList.add("hatter");}  
else  
{ percszam.classList.remove("hatter"); }
```

Egy if-else elágazásban feltételnek megadtuk, hogy akkor adjon a percszam azonosítójú elemre egy új osztályt, amikor a másodperc 59-en áll. Ha a másodperc elhagyta az 59-et akkor pedig vegye le róla a hozzáadott osztályt.

```
let oraszam = document.getElementById('oraszam');
```

Az „oraszam” azonosítóval ellátott elemet eltároltuk egy „oraszam” nevű változóban.

```
if(percszam.innerHTML == 59 && mpszam.innerHTML == 59)
{ oraszam.classList.add("hatter");}
else
{ oraszam.classList.remove("hatter");}
```

Egy if-else elágazásban feltételnek megadtuk, hogy ha a perc és a másodperc is 59-en áll, akkor egy új osztályt adunk hozzá, aminek segítségével megváltoztatjuk a háttérét. Ellenkező esetben levesszük róla az előbb hozzáadott osztályt.

```
let napszam = document.getElementById('napszam');
```

A napszam azonosítóval ellátott elemet eltároltuk egy napszam nevű változóban.

```
if(oraszam.innerHTML == 23 && percszam.innerHTML == 59&&
mpszam.innerHTML == 59)
{napszam.classList.add("hatter");}
else
{ napszam.classList.remove("hatter");}
```

Egy if-else elágazásban feltételnek megadtuk, hogy ha a óra 23-on, a perc és a másodperc is 59-en áll, akkor egy új osztályt adunk hozzá, aminek segítségével megváltoztatjuk a háttérét. Ha a megadott feltételek nem teljesülnek, akkor az előzőleg hozzáadott osztály levesszük róla és visszaállítjuk az eredeti állapotába.

```
}, 1000);
```

Ennyi időközönként, azaz 1 másodpercenként hajtja végre ezt a folyamatot a program.

A Code And Create Youtube csatornájának a Countdown with HTML, CSS, and JavaScript / How to create JavaScript Countdown nevű videójából vettük át az ötletet.

### 2.5.2. Navigáció és lábléc megvalósítása

A navigációs menüt és a láblécet annak érdekében, hogy ne kelljen minden oldalon külön leprogramozni, az include() metódussal illesztettük be valamennyi oldalra. Ezt úgy valósítottuk meg, hogy a „projekt” nevű mappában létrehoztunk egy navbar.php és footer.php nevű fájlt, és ezekben a fájlokban írtuk meg a weboldal részeinek a kódját.

Az `include()` utasítás átveszi a megadott fájlban található összes szöveget, kódot és bemásolja az `inculde` utasítást használó fájlba. A fájlok befoglalása nagyon hasznos, ha ugyanazt a PHP-t, HTML-t vagy szöveget egy webhely több oldalára szeretnénk felvenni.

### 2.5.3. Sötét és világos mód közötti váltás

A stílus beállítását a főoldalon található profilképre kattintva érhetük el. Itt a beállítások menüpont alatt található egy „sötét mód” beállítási lehetőség. Ha a felirat alatti csúszkát elcsúsztatjuk, akkor az oldal háttérszíne fehérről feketére változik. Azért gondoltuk, hogy beépítünk az oldalunkba egy ilyen lehetőséget, mivel kényelmesebb a szemnek a sötétben való olvasás sötét háttérrel. Ezt úgy valósítottuk meg, hogy készítettünk még egy stíluslapot, ami teljesen megegyezik az alapból használt stílussal. A kettő között a háttérszín különbözik. Ha a csúszkára kattintunk, akkor JavaScript segítségével megvizsgáljuk azt a `<link>` taget amibe a stíluslapot linkeltük az oldalhoz és kicseréli az útvonalat az adott stíluslaphoz.

**Ezt a következő képpen oldottuk meg:**

```
function ToggleMode() {  
    let stilus = document.getElementById("stilus");
```

Itt kiválogattuk azokat az elemeket, amelyekhez a „stilus” osztályt rendeltük hozzá és eltároltuk őket egy változóba.

```
    if (stilus.getAttribute("href") == "../cssek/stilus.css") {  
        stilus.href = "../cssek/dark.css"; }  
}
```

Ezt követően egy `if-else` elágazásban megvizsgáltuk, ha az oldalra belinkelt stíluslap elérési útvonala megegyezik a feltételben megadottal, akkor a felhasználó, ha rákattint a csúszkára a beállítások menüben a megadott stíluslap útvonalát kicseréli a másikra, ezzel megváltoztatva az oldal színének a megjelenését.

```
    else { stilus.href = "../cssek/stilus.css"; }
```

Az `else` ágban az történik, ha a sötét módot tartalmazó stíluslap van belinkelve, azt állítja vissza az eredetire.

### 2.5.4. Angol nyelv beállítása

Az angol és a magyar nyelv közötti váltást azért építettük bele az oldalunkba, mert úgy gondoltuk, ha esetleg a jövőben ténylegesen használnánk, az oldalt akkor több emberhez tudunk eljutni ezáltal.

Erre az ötletünkre több megoldást is próbáltunk megvalósítani. Az első ilyen az volt, hogy a főoldalon található profilképre kattintva a beállítások menüben található

zászlókra kattintva egy másik oldalra irányít át az oldal minket. Ekkor, ha a magyar oldalon álltunk eredetileg akkor az angol nyelvű oldalra kerültünk és ez ellenkező esetben is működött természetesen. Ez viszont nem nyerte el a tetszésünket, mivel nem akartunk új oldalra lépni, hanem azt szeretttük volna megoldani, hogy ne irányítsuk át a látogatót új oldalra csak a szöveget cseréljük ki az oldalon angolra.

Ezt úgy valósítottuk meg, hogy `<span>` tagek közé angol és magyar nyelven is beírtuk a szöveget, amit le szeretttünk volna fordítani. Mind a kettőnek adtunk egy-egy osztályt „en” és „hu” néven. Erre azért volt szükségünk, mert JavaScripttel nézzük meg, hogy melyik osztály aktív és ha a gombra rákattintunk, akkor a program kicseréli azokat. Hiányossága ennek a megoldásnak az, hogy a cikkek címeit és tartalmát nem sikerült ezzel a módszerrel lefordítani.

#### **Ennek a megvalósítása a következőképpen történt:**

```
let magyar = document.getElementsByClassName("hu");  
let angol = document.getElementsByClassName("en");
```

Az en és hu osztályazonosítóval ellátott elemeket kiszelektáltuk és eltároltuk őket két külön változóba.

```
function Changelang()
```

Létrehoztunk egy függvényt „Changelang” néven.

```
for (let index = 0; index < magyar.length; index++)
```

Egy for ciklussal végig iterálunk

```
{ if(magyar[index].className == "hu")  
  { magyar[index].classList.add("magyarnincs");  
    angol[index].classList.add("angolvan");}  
else  
  { magyar[index].classList.remove("magyarnincs");  
    angol[index].classList.remove("angolvan");} }
```

#### **2.5.4. Cikk felvitele**

A projektünk egy online hírportál ezért a legfontosabb eleme az oldalunknak a cikkek írása és azoknak a felvitele, megjelenítése az oldalon. Erre többfajta megoldást is készítettünk mire megtaláltuk azt, amit végül beépítettünk az oldalunkba.

Elsőnek a Redactor szövegszerkesztőt alkalmaztuk, mivel ezt volt a legegyszerűbb használni és beépíteni az oldalunkba. Viszont hátránya volt neki, hogy viszonylag kevés formázási lehetőséget tartalmazott, illetve ha egy szöveget előre megformáztunk

Wordben és azt szerettük volna beilleszteni, akkor a formázást nem tartotta meg. Ez azért volt probléma, mert a formázás elég nehézkes volt a Redactorral és nem is volt annyi lehetőség rá, mint a Wordben.

Ekkor döntöttünk úgy, hogy egy másik szövegszerkesztőt építünk az oldalunkba. A választásunk a CKEditor-ra esett. Több oka van, hogy erre váltottunk.

A legfontosabb szempont az volt, hogy a formázott szövegnek megtartsa a formázását. Ez a szerkesztő ezt lehetővé teszi így az újságíróknak sokkal egyszerűbb dolga lesz, amikor cikkeket szeretnének fölvenni az oldalra. Másrészt ez a szerkesztő rengeteg formázási lehetőséget tartalmaz. Szinte úgy lehet formázni a szöveget mintha Wordben csinálná az ember.

Amikor már szinte teljesen elkészültünk a projekttel, akkor jutott eszünkbe milyen jó lenne automatizálni a régebbi fő hírek átmozgatását az oldalsávba. A fő híreink mindig 1-es státuszúak az oldalsávban lévő hírek pedig 0. Ezt úgy oldottuk meg, hogy rögtön a felvitelnél lekérdezzük az 1-es státuszú cikkeket. Ezeket létrehozás dátuma szerint sorrendbe rendeztük (order by létrehozás limit 1) és így megkaptuk a legrégebbi cikk azonosítóját, amelyet UPDATE-EL 0-ra módosítottuk. Így elértük, hogy ha az újságíró felvisz egy cikket, aminek a státusza 1-es tehát fő hír, akkor a program a legrégebbi fő híreket átrakja az oldalsávba a státuszának 0-ra módosításával.

## **2.6. Chat megvalósítása**

A chat megvalósításához három darab fájl használtunk. Egy chat.php elnevezésű fájl amiben a kinézetét és a felépítését határoztuk meg, egy chathez.php nevű fájl amiben különféle tiltásokat és korlátozásokat határoztunk meg.

Ilyen például az trágár szavak tiltása. Ezt úgy valósítottuk meg, hogy létrehoztunk egy „spam” elnevezésű tömböt, amibe beletöltöttük manuálisan az ilyesfajta szavakat. Ha valaki megpróbálja ezeket a szavakat beírni a chatbe és el szeretné küldeni, akkor nem történik semmi, csak egyszerűen nem kerül elküldésre az üzenet.

Egy másik ilyen korlátozás az üzenet hosszára vonatkozik. Ezzel 18 sorban maximalizáljuk az elküldhető üzenet hosszát.

A harmadik fájl, amit használtunk egy txt fájl. Ha elküldünk egy, akkor azok itt kerülnek eltárolásra.

## 2.7. Hozzászólások lehetőségének megvalósítása

Az oldalunkon megvalósítottunk egy kommentszekciót, ahol az olvasók ki tudják fejteni a véleményüket az olvasottakkal kapcsolatban. A hozzászólások funkciót a navigációs menüben található chat ikonra kattintva lehet elérni. Ha egy olvasó elküld egy üzenetet, akkor az automatikusan lefrissül mindenhol. Ha olyan üzenet van, amit az olvasó nem olvasott még el akkor az ikon közepén lévő 3 pont hullámozni kezd.

Ez a következőképpen valósítottuk meg:

```
let morecommentgomb = document.querySelector(".tobbbkomment");
```

Létrehozunk egy „morecommentgomb” nevezetű változót, melyben eltároltuk azt a gombot, amelyhez „tobbbkomment” nevű osztályt rendeltük hozzá.

```
const chatBox = document.querySelector(".chatUzenetek");
```

Itt szelektáltuk ki azt a konténert, amibe maguk a kommentek kerülnek.

```
function morecomment() {  
  if (chatBox.className === "comment chatUzenetek") {  
    morecommentgomb.innerHTML = "Kevesebb hozzászólás";  
    chatBox.classList.add("morecommentclass");  
  }  
  else {  
    chatBox.classList.remove("morecommentclass");  
    morecommentgomb.innerHTML = "Többi hozzászólás";  
  }  
}
```

Létrehoztunk egy paraméter nélküli függvényt a komment.js fájlunkban „morecomment” néven. Ebben a függvényben szelekciót végeztünk, amelyben megvizsgáltuk, hogy az chatBox néven eltárolt a konténer milyen osztályt kapott. Alapból megadásra került egy osztály, amelyben a magasságát fixáltuk. Ha csak ezek kerültek hozzárendelésre, akkor adunk hozzá még egy újabbat is, amellyel megnöveljük a magasságát. Ekkor több komment válik láthatóvá és a gomb felirata is kicserélődik. Ellenkező esetben levesszük az előbbieken hozzáadott osztályt.

```
const inputField = document.getElementById("szoveg");
```

Itt változóba tároltuk el a „szoveg” azonosítóval ellátott input mezőt, amibe a felhasználók írják a kommentet.

```
const sendBtn = document.getElementById("new");
```

Itt változóba tároltuk el a „new” azonosítóval ellátott gombot, amivel a kommentet lehet elküldeni.

```
inputField.addEventListener("keypress", function (event) {
```

```

if (event.key === "Enter") {
    event.preventDefault();
    sendBtn.click();
    scrollToBottom();}})

```

Az előző sorokba az valósítottuk meg, hogy egy függvényt hozzáadunk az előbb eltárolt beviteli mezőhöz, ami egy billentyű lenyomására hajtódik végre. ENTER lenyomása esetén elküldi a megírt üzenetet.

```

setInterval(() => {

```

Itt létrehozunk egy metódust, ami adott időn belül hajt végre egy függvényt.

```

let xhr = new XMLHttpRequest();

```

Egy olyan osztály objektumát tároljuk el, ami adatot tud lekérni a szerverről.

```

xhr.open("GET", "kommentbe.php", true);

```

A GET metódussal lefuttatjuk a kommentbe.php-t, aminek az a dolga, hogy átadja az adatbázisba felvitt kommenteket.

```

xhr.onload = () => {
    if (xhr.readyState === XMLHttpRequest.DONE) {

```

Ebben az if elágazásban azt vizsgáljuk, hogy sikeresen megnyitotta-e a kommentbe.php-t.

```

if (xhr.status === 200) {
    let data = xhr.response;
    chatBox.innerHTML = data;
    if (!chatBox.classList.contains("active")) {
        scrollToBottom();}

```

Ha igen, akkor a kapott adatot belerakja abba a konténerbe, amiben meg szeretnénk jeleníteni.

```

const ovanbent = document.querySelector(".felnev").innerHTML;

```

Létrehoztunk egy „ovanbent” változót, aminek értékül adjuk azt a nevet, aki éppen be van jelentkezve.

```

const oirt = document.getElementById("oirt").innerHTML;

```

Létrehoztunk egy „oirt” változót, aminek értékül adjuk azt a nevet, aki a legutóbbi kommentet írta.

```

let popdown1 = document.getElementById("elsole");
let popdown2 = document.getElementById("masodikle");
let popup1 = document.getElementById("elsofel");

```



A fenti három sorban a chat ikonban található fekete pöttyöket tároltuk el azonosítók alapján.

```
if (oirt != ovanbent) {  
    popdown1.classList.add("popdown");  
    popdown2.classList.add("popdown");  
    popup1.classList.add("popup");
```

Ebben az if elágazásban azt vizsgáltuk meg, hogy ha az előbb eltárolt nevek nem egyeznek meg, akkor kezdenek el hullámozni a pöttyök, úgy hogy hozzáadjuk a „popdown”, „popdown2” és „popup” osztályokat.

```
}else {  
    popdown1.classList.remove("popdown");  
    popdown2.classList.remove("popdown");  
    popup1.classList.remove("popup"); }
```

Az else ágban pedig eltávolítjuk róluk az előbb hozzáadott osztályokat.

```
xhr.send();
```

Ezt követően elküldjük a szervernek feldolgozásra.

```
}, 3000)
```

A 3000 azt jelenti, hogy 3 másodperc alatt hajtja végre ezt a procedúrát.

### **3. Felmerült problémák**

A munkánk során előfordultak kisebb, nagyobb nehézségek, amiket a következő bekezdésben mutatunk be a megoldásainkkal egyetemben.

#### **3.1.A főoldalon a kártyák megjelenése**

A tesztidőszakban a kinyitható kártyákba a HTML kódba beégettük az adatokat, amik így statikusan jelentek meg addig tökéletesen úgy működött, ahogy szeretnénk volna. Viszont ezt is dinamikus megoldással akartuk adatbázisból betölteni és ekkor akadt egy kis probléma. Az történt, hogy az eddig megformázott sávok méretei teljesen megváltoztak és szétszúszott az egész sáv.

Először próbáltuk a kezdő és záró konténer jelölőket (<div></div>) elhelyezni mind az aktív mind pedig az inaktív panelen, majd ezt követően a HTML-be helyeztük bele, de egyik megoldás sem segített. Megvizsgáltuk a böngésző fejlesztői eszköztárában a megjelenő elemeket. Itt vettük észre, hogy az inaktív panelekben is benne van a szöveg és ez deformálja a sávokat. Erre azt a megoldást találtuk, hogy a nem aktív kártyák tartalmához hozzáadtunk egy „display: none” tulajdonságot, vagyis nem jelenítjük meg ilyenkor. A tesztelés során így sikerült elérni a kívánt eredményt. A telefonos nézet vizsgálatakor viszont azt tapasztaltuk, hogy ebben az esetben vissza kell állítani a szövegeket. Így a meglévő Media query kódunkat kiegészítettük a display: block kódsorral, azaz a tartalom újra megjelenítésével.

#### **3.2.Felhasználók és adminisztrátorok törlése**

Az oldalra beregisztrált újságíróknak a törlését a felhasználók kezelése oldalon tudjuk megvalósítani.

Eredeti elképzelésünk az volt, hogy az azonosító alapján a felhasználó minden adatát töröltük az adatbázisból. A tesztelési folyamat során viszont abba a hibába ütköztünk, hogy ezzel a megoldással a híreket, amiket megírt és publikált, és a hozzászólásokat is, amelyeket közzétett szintén törlésre kerültek, mert az azonosító alapján töröltünk ki mindent.

Erre először az a megoldásunk született, hogy nem töröltük ki a felhasználó egyetlen adatát sem, hanem helyette a regisztrációkor megadott e-mail címét változtattuk meg egy random generált kódra, sha1-es titkosítással. Amint ez a folyamat lezárult, ha az adott felhasználó be szeretett volna lépni a kezelő felület helyett a kezdő oldalra került.

Ezzel azt sikerült megoldani, hogy a cikkeket (ha adminisztrátori jogosultsággal rendelkezett) és hozzászólásokat, amiket megírt megmaradtak az oldalon.

**Ezt a következő képpen oldottuk meg:**

```
if(isset($_GET['id']))
```

Itt megvizsgáltuk azt, hogy a megadott azonosítóra létezik-e.

```
require("../kapcsolat/kapcsproj.php");
```

Létre hoztuk a kapcsolatot az adatbázissal a require() függvény segítségével.

```
$id= (int)$_GET['id'];
```

```
$replaceemail = sha1(rand(0,10000000000000000000));
```

Eltároltuk az azonosítót, és az ahhoz tartozó email címet sha1-es titkosítással, azaz egy véletlenszerűen generált kódra módosítottuk.

```
$replaceemail .= "@gmail.com";
```

A generált kódhoz hozzá illesztettük a @gmail.com végződést.

```
$sql = "UPDATE projectacc set `email`='{ $replaceemail }' WHERE id = { $id }";
```

Majd frissítettük az email címet az adatbázisban is.

```
mysqli_query($dbconn, $sql);
```

```
}header("Location: admin-kezel.php");
```

Ha folyamat lefutott, akkor az oldal az admin főoldalra navigált vissza minket.

A tesztelés folyamán ezt a megoldást azonban meglehetősen furcsának találtuk ezért tovább gondolkoztunk és végül egy szerintünk sokkal jobb és logikusabb megoldásra jutottunk.

Az adatbázisban a 'projectacc' táblához, amiben a regisztrál felhasználók adatait tároljuk felvettünk egy új oszlopot, aminek segítségével be tudjuk állítani, hogy a fiók aktív vagy nem. A belépéskor azt is megvizsgáljuk, hogy melyik státuszban van a fiók, amibe be akarnak lépni, és ha az adatbázisban inaktívra van állítva az értéke, akkor nem tud belépni a felhasználó az oldalra. Ezzel nem kellett semmilyen adatot megváltoztatni ráadásul ez egy sokkal egyszerűbb és logikusabb megoldás a problémára.

**Ennek a megoldásnak a kódja a következő:**

```
$id= (int)$_GET['id'];
```

```
$sql = "UPDATE bejelentkezettek set `statusz`=\"false\" WHERE id = { $id }";
```

Ha az azonosító megegyezik azzal az azonosítóval, amivel a felhasználó rendelkezik, akkor a fiókja státuszát módosítjuk false értékre, ezzel elérve azt, hogy ne tudjon többet bejelentkezni. Tehát nem töröljük egy az egyben a fiókját.

```
mysqli_query($dbconn, $sql);}
header("Location: admin-kezelő.php");
```

Ha a folyamat véget ért akkor az adminisztrátor kezelő oldalra dob vissza az oldal.

### **3.3.Összes felvitt cikk megjelenítése egy oldalon**

Arra gondoltunk, hogy hasznos lenne, ha egy oldalon az összes eddigi cikket megjelenítenénk és így a későbbiekben egyszerűbb lenne a felhasználónak keresni a cikkek között.

Amikor elkészültünk a dizájnnal, egy újabb kisebb fajta akadályba ütköztünk. Az volta a gondunk, hogy telefonos nézetben nem vette figyelembe a reszponzivitást szolgáló kódokat. Ez annak volt köszönhető, hogy a HTML kód fejlécéből kihagytuk a következő sort:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

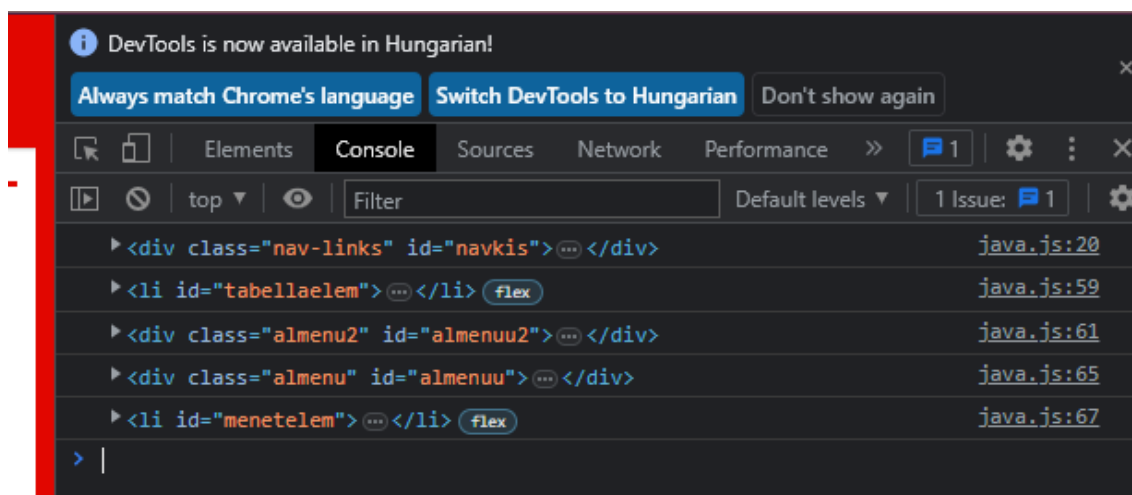
Ez a sor utasítja a böngészőt, hogy mindenféle eszköz, mint pl. mobil, Tablet, stb. különböző képernyő méretének a maximum szélessége legyen a teljes oldal szélessége.

## 4. Tesztelés

A weboldalaink HTML kódrészleteit a <https://validator.w3.org/> oldalon ellenőriztük.

A CSS kódrészleteket a <https://jigsaw.w3.org/css-validator/> oldalon ellenőriztük. A CSS-nél előnyünkre szolgált, hogy a Visual Studio Code észlelte a hibát, így még a programozás folyamán ki tudtuk javítani. Így a validátor nem talált hibát a CSS kódban.

A Javascript kód írása közben `console.log()` segítségével folyamatosan ellenőriztük, hogy sikeresen tároltuk-e el az adatokat a változóknak, és azt az eredményt kapjuk-e, amelyet várunk. A böngésző konzoljában folyamatosan figyelemmel követtük, hogy nem észlel-e hibát a programban. A folyamatos ellenőrzés és JavaScript kód ellenőrzése azért fontos, mert ha csak a legvégén ellenőrizzük és hibát tartalmaz a program, akkor már sokkal nehezebb megtalálni, mint folyamatában. A JavaScript tesztelését több oldalon is benne hagytuk a kódunkban, így látható, ahol használtuk, ha megnyitjuk a fejlesztői eszköztáron belül a konzolt.



29) ábra Javascript tesztelése a főoldalon

A fenti képen látható, hogy folyamatosan ellenőriztük, azaz kiírtattuk a konzolra a változóba eltárolt értékeket, így biztosak lehettünk benne, hogy a megfelelő értékek kerültek a kezünkbe.

A PHP kódok ellenőrzését a <https://www.phptools.online/php-checker> oldalon is ellenőriztük. Habár egyrészt, a Visual Studio Code, ha hibát észlel a PHP kódrészletekben piros aláhúzással jelzi. Továbbá több kiegészítő modult is letöltöttünk a VSC-hez, amelyek segítik a debugolást a php programozás során.

A másik segítségünk az az, hogy weboldalon szintén azonnal megjelenik a hiba üzenet, legyen az akár szintaktikai hiba, vagy akár valamilyen egyéb hiba. `Print_r`-el és

a `var_dump()`-al folyamatosan teszteltük a kódunkat, hogy hozzá jutunk e, azokhoz az adatokhoz, amelyekre épp szükségünk van. Ennek eredményeképpen a PHP kód is hibátlanul működik minden oldalon, a böngészőben nem jelenik meg sehol hiba, bármilyen műveletet is hajtunk végre.

Amikor elkészültünk a munkánkkal teszteltük az oldalak megjelenését különböző mobil eszközökön, és böngészőkkel, mint Edge, Firefox, Chrome, Opera és Safari.

## Összefoglalás

Összegezve a munkánkat elmondhatjuk, hogy nagyon gördülékenyen ment a közös munka. Rendkívül örülünk a végeredménynek, hiszen az elején nem gondolta volna egyikünk sem, hogy ennyi mindent meg tudunk majd valósítani az oldalunkon. Amikor sikerült egy feladatrészt megoldani, óriási siker élményt adott, és egyre bátrabban terveztünk tovább, így folyamatosan bővült a projekt a munka során.

Természetesen lehetne még fejleszteni, bővíteni az oldalt, hiszen még rengeteg új ötletünk született a tervezés során még az utolsó pillanatokban is, de úgy véljük, hogy amit az elején elterveztünk, mindent sikerült megvalósítani maradéktalanul, sőt még egy kicsivel többet is.

Tovább fejlesztési lehetőség az is, hogy jelen állásában, ha valaki elfelejtette a jelszavát és újat szeretne kérni, akkor az oldal csak azt nézi meg, hogy van-e olyan e-mail az adatbázisban, amit beírt, és ha van, akkor átirányítja az oldal a felhasználót egy másik lapra ahol meg tud magának adni új jelszót. A későbbiekben meg lehetne oldani token rendszerrel, hogy a felhasználó az email címére kap egy üzenetet, amiben van egy link a változtatáshoz.

Szeretnénk majd a későbbiekben megvalósítani azt is, hogy a navigációs menüben egy kereső segítségével egy konkrét hírre is rá lehessen keresni az oldalon. Ez azért lenne egy nagyon hasznos funkció, mert így a felhasználónak nem kellene hosszasan keresgélni az oldalon, ha egy konkrét dologról, eseményről szeretne informálódni, hanem egyszerűen rá tudna keresni arra, ami érdekli.

Jelenleg az adminisztrátorok és az újságírók is ugyanazon jogosultsággal rendelkeznek. Egyenlőre, mi töltjük be mindkét szerepet, hiszen nem tudunk bért fizetni addig más újságírónak, amíg az oldalnak nincs bevétele. A későbbiekben viszont szeretnénk megvalósítani azt, hogy egy harmadik fajta jogosultságot vennénk fel az adminisztrátor és az egyszerű felhasználó mellé, ami már az újságíró lenne. Ezzel a jogosultsággal az adminisztrátor felületre kapna lehetőséget belépni, azonban ott csak a cikkek felvétele és módosítása panel lenne számára elérhető és az adminisztrátor műveleteket tartalmazó oldalakra nem lenne lehetősége belépni, hacsak az adminisztrátorok meg nem változtatják a jogosultságait.

Ha rajtunk kívül is lesznek újságírók, akkor a cikkek törlését is át fogjuk alakítani úgy, hogy az újságírók, csak kukába tudják helyezni a cikkeket, ne tudjanak véglegesen törölni. A végleges törléshez csak nekünk lenne majd jogunk.

Az adatbázisunkban jelenleg az idei szezon adatait tároljuk, de vannak viszont olyan adatok is, amik nem, vagy nem csak az idei szezonra vonatkoznak. Ezekre az adatokra azért volt szükségünk, mivel egy rövid csapatbemutató részleget is megvalósítottunk az oldalon. Minden, a Formula 1-hez tartozó adatot manuálisan, a phpMyadmin felületén visszük fel. A kibővítése azért lenne jó az adatbázisnak, mivel akkor, ha például ki szeretnénk írni egy csapatnak a jelenig elért összes futamgyőzelmét, akkor ezt egy egyszerű lekérdezéssel meg lehetne számolni, nem pedig csak manuálisan beírni.

Éles weboldal esetében, már a kereső optimalizálásra is gondot kell fordítanunk, hogy a látogatók megtalálják a weboldalunkat. Az oldal fenntartási költségeinek majdani fedezését, kétféle módon képzeljük el, első körben minimális mennyiségű reklámok bevételeiből, később esetleg egy olyan online webáruházzal egészíthetnénk ki, ahol a Forma1-el kapcsolatos ajándéktárgyakat lehetne megrendelni tőlünk.

Záró dolgozatunk megírásához több szakdolgozatot is tanulmányoztunk, hogyan és milyen formában kell elkészíteni. Nagyon hasznos volt számunkra a projekt munka része is, hiszen mindketten főiskolán tanulunk tovább, és így a majdani szakdolgozatunk elkészítéséhez is szereztünk már gyakorlatot.

Az elkészített projekt munkák elérhetősége a Github-on:

<https://github.com/BaNjeet03/projektmunka2023>

A nethely ingyenes tárhelyen:

<http://rnmotorsport.infora.hu/projekt/user/user.php>



## Források:

- 1) Code and Create :Countdown with HTML, CSS, and JavaScript / How to create JavaScript Countdown Link: <https://www.youtube.com/watch?v=KOZGCBswGBc>
- 2) CSS-TRICKS - A Complete Guide to CSS Grid Link: <https://css-tricks.com/snippets/css/complete-guide-grid/>
- 3) CSS-TRICKS - A Complete Guide to Flexbox Link: <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>
- 4) CSS-TRICKS - Media Queries for Standard Devices Link: <https://css-tricks.com/snippets/css/media-queries-for-standard-devices/>
- 5) Fabi Ottoviani - Menetrendek oldal:- <https://codepen.io/supah/pen/xxJMbbg>
- 6) FIA - F1 adatok: Link: <https://www.formula1.com>
- 7) GP Racing Stats - F1 adatok: <https://gpracingstats.com>
- 8) Összes cikk oldal kártyák: <https://codepen.io/nicolaspavlotsky/pen/wqGgLO>
- 9) Stack Overflow Link: <https://stackoverflow.com/>
- 10) W3School - JavaScript Tutorial Link: <https://www.w3schools.com/js/default.asp>
- 11) W3School - PHP Tutorial Link: <https://www.w3schools.com/php/default.asp>
- 12) Wikipédia – CSS Link: <https://hu.wikipedia.org/wiki/CSS>
- 13) Wikipédia – HTML Link: <https://hu.wikipedia.org/wiki/HTML>
- 14) Wikipédia – JavaScript Link: <https://hu.wikipedia.org/wiki/JavaScript>
- 15) Wikipédia - PHP Link: <https://hu.wikipedia.org/wiki/PHP>

## Ábrajegyzék

1) ábra Főoldal .....	5
2) ábra Visszaszámláló a főoldalon .....	5
3) ábra Szétnyíló kártyák .....	6
4) ábra Lábléc .....	6
5) ábra Regisztrációs felület .....	6
6) ábra Bejelentkezés felülete .....	7
7) ábra Elfelejtett jelszó.....	8
8) ábra Új jelszó megadása.....	8
9) ábra Menetrend almenü .....	8
10) ábra Teljes menetrend .....	9
11) ábra Tabella almenü.....	10
12) ábra Teljes tabella .....	11
13) ábra Csapat tabella .....	11
14) ábra Összes cikk oldal.....	12
15) ábra Adminisztrációs oldal .....	13
16) ábra Cikk felvitele.....	15
17) ábra Cikkek kezelése oldal .....	16
18) ábra Felhasználók kezelése oldal .....	17
19) ábra Kezdetek .....	22
20) ábra Teljes adatbázis .....	24
21) ábra Accounts tábla.....	25
22) ábra Comments tábla .....	25
23) ábra News tábla.....	26
24) ábra Teams tábla .....	27
25) ábra Drivers tábla.....	27
26) ábra Races tábla .....	28
27) ábra Participants tábla .....	28
28) ábra Fullteams tábla .....	28
29) ábra Javascript tesztelése a főoldalon .....	44