# Project tárolása, csoportos dolgozás megvalósítása

# Frontend

1. A frontend felépítése
   1. Mi az a frontend?

A frontend alatt a program megjelenítéséről beszélünk, ami a szerverről lekérdezett (backend) adatokat jeleníti meg a felhasználó számára. Megjelenés mellett különböző műveleteket és számolást is végre kell hajtani a frontend felületnek, annak érdekébe, hogy a felhasználó feltudja dolgozni a kapott adatokat. Ez alatt lehet érteni számok összeadását, kivonását, és ezeknek a számoknak az ábrázolását a megfelelő helyen a képernyőn.

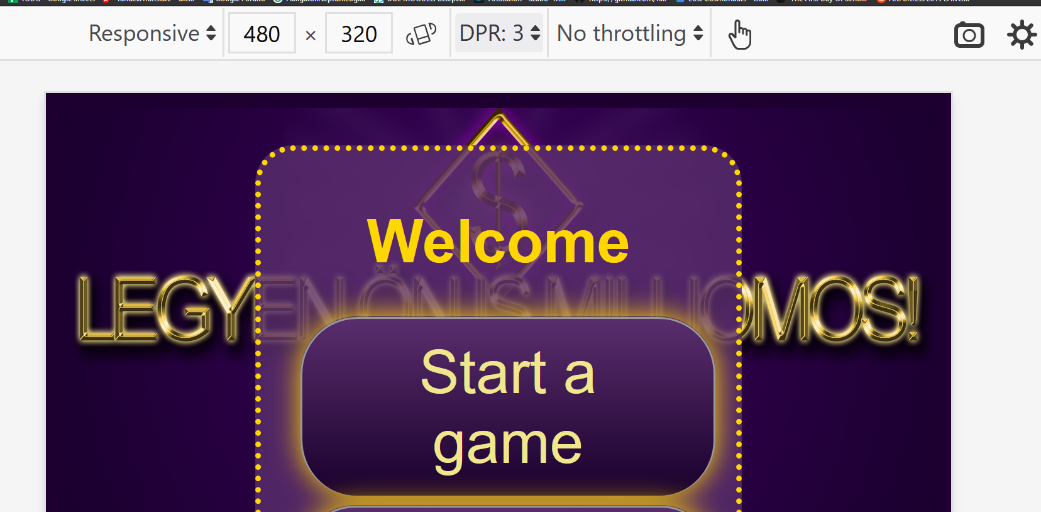
A HTML lehetővé teszi az adatok csoportosítását, megjelenítését megfelelő helyen, és egyéb formai dolgok, például színezés és méretezés. Emellett beépített funkciók és eszközöket is lehet használni vele, például gombok vagy szövegbevitelre mezőt, ahova képes a felhasználó írni. Ezeket az úgynevezett „objektumokat” személyre lehet szabni, és rengeteg tulajdonságot hozzájuk tudunk rendelni, például a felhasználó jelszavát a program csak csillagokkal jeleníti meg, vagy csak bizonyos hosszúságú jelszót írhatunk be. Ilyenkor fontos figyelembe venni, hogy a frontend képes megakadályozni érvénytelen adatoknak a bevitelét,

* 1. Frontend a Legyen ön is Milliomos applikációban



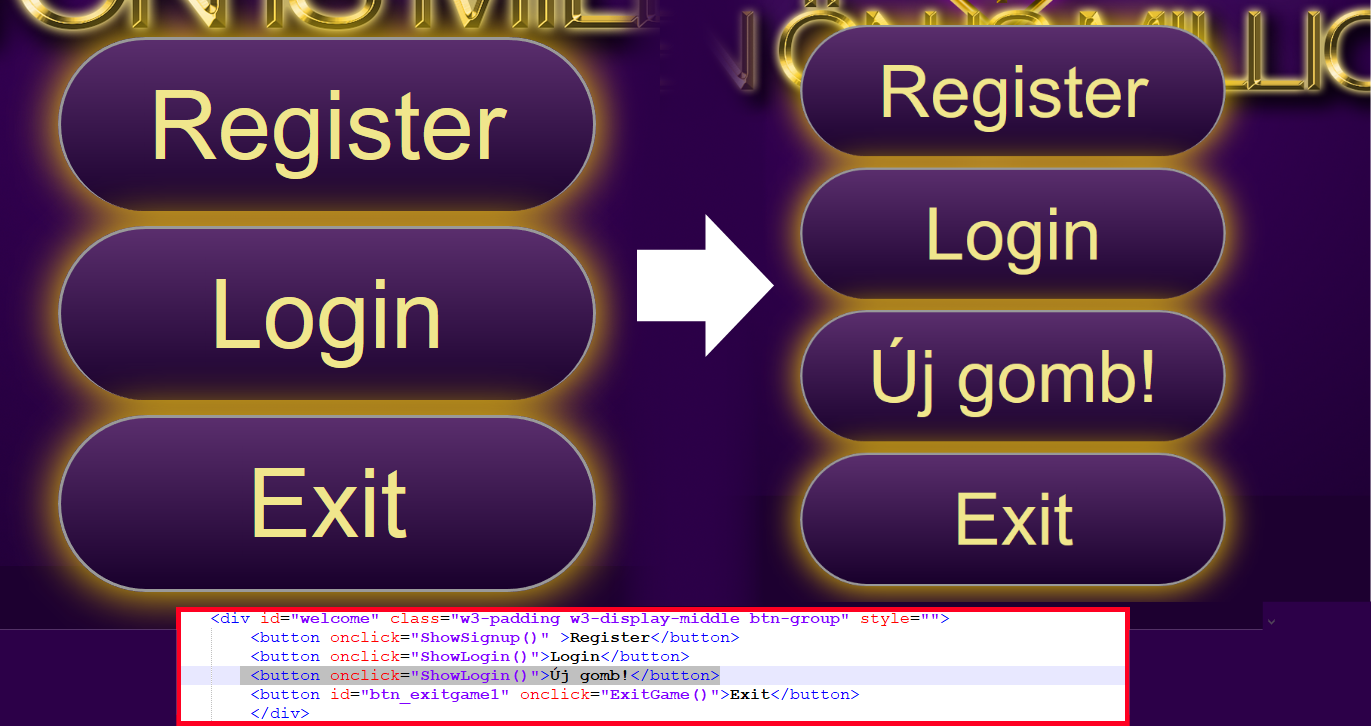
A frontend fejlesztését lehetővé teszi a Cordova, mivel a HTML web programozási nyelvet használja. A Cordova egy futó web applikáció, ami képes a telefonos verzió kezelésére. A Cordova képes mindenféle funkciót kihasználni a mobilokból, ezalatt a billentyűzetet, kamerát, GPS és forgatást képes kezelni. A Legyen ön is Milliomos applikáció esetében a billentyűzet és megjelenésen van a hangsúly.

A fejlesztéshez szükség van egy IDE [[1]](#footnote-1)szoftver letöltéséhez. A legyen ön is milliomos project esetében a Visual Studio 2015 Community Edition-t használta a csoport, és ingyenes is. A Visual Studio emellett nagyon hasznos lesz a „valós idejű” tesztelésre is. A HTML kódot viszont lehet bármilyen más szövegszerkesztővel is használni, például Notepad++, ami egy ingyenes és programozók számára ajánlott szöveg szerkesztő.[[2]](#footnote-2)



A legnagyobb probléma, ami felmerült a fejlesztés során az a megfelelő arányok eltalálása, mivel az applikáció nem csak számítógépen, de telefonon is használható. A képernyő arányához kell viszonyulnia a képernyőnek, és a rajta lévő objektumoknak, például a gombok és szöveg dobozok. Az arányok mellett a HTML korlátozásai is problémát jelenthet, mivel nem minden web böngésző támogatja az összes funkciót. [[3]](#footnote-3)

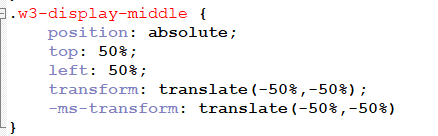
A megfelelő arányokat a HTML képes betartani, ha az objektumokat a képernyőhöz viszonyítva számoljuk. Ezt úgy lehet elérni, hogy folyamatosan rendezzük a csoportokat, például a képernyő közepére, szélére, legaljára rendezzük az objektumokat. Innentől fogva az applikáció minden platformon megfelelő méretezéssel jelenik meg. Ez viszont rengeteg tesztelést igényel, mivel az applikációt folyamatosan bővíteni szeretnénk. Szerencsére a rendezés elősegíti a bővíthetőséget, mivel képesek vagyunk rendezni az elemeket tetszés szerint, a meglévő elemek befolyásolása nélkül. Például, ha új gombot szeretnénk berakni a főmenübe, akkor elég egy új gomb objektumot felvenni a HTML file-ba, és utána az megfelelően rendeződik.



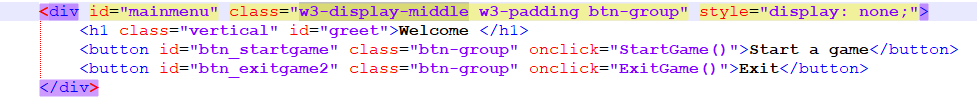
*Új gomb a HTML kódba, megfelelő rendezéssel, csak egy darab új sor kellett*

* 1. Formázás

A HTML és Cordova használata emiatt előnyös, mivel megfelelő felépítés lehetővé teszi az applikáció gyors fejlesztését és tesztelését, egyszerre teszi lehetővé a webes verzió és a telefonos verzió fejlesztését. A megfelelő formázást a HTML által használt .CSS file tartalmazza, itt található különböző rendezés típusok, amikre hivatkozhatunk a HTML-en belül. Ezt a .CSS file-t egy másik mappába lett téve jobb rendezés érdekében. Kifejezetten hasznos a képernyő alapú rendezés, ami %-osan rendezi a tartalmát a HTML elemeknek. Ezek fel vannak osztva több típusú rendezésre, például felfelé rendezés, lefelé, jobbra és balra.[[4]](#footnote-4)



*A középre rendezés tulajdonságai*



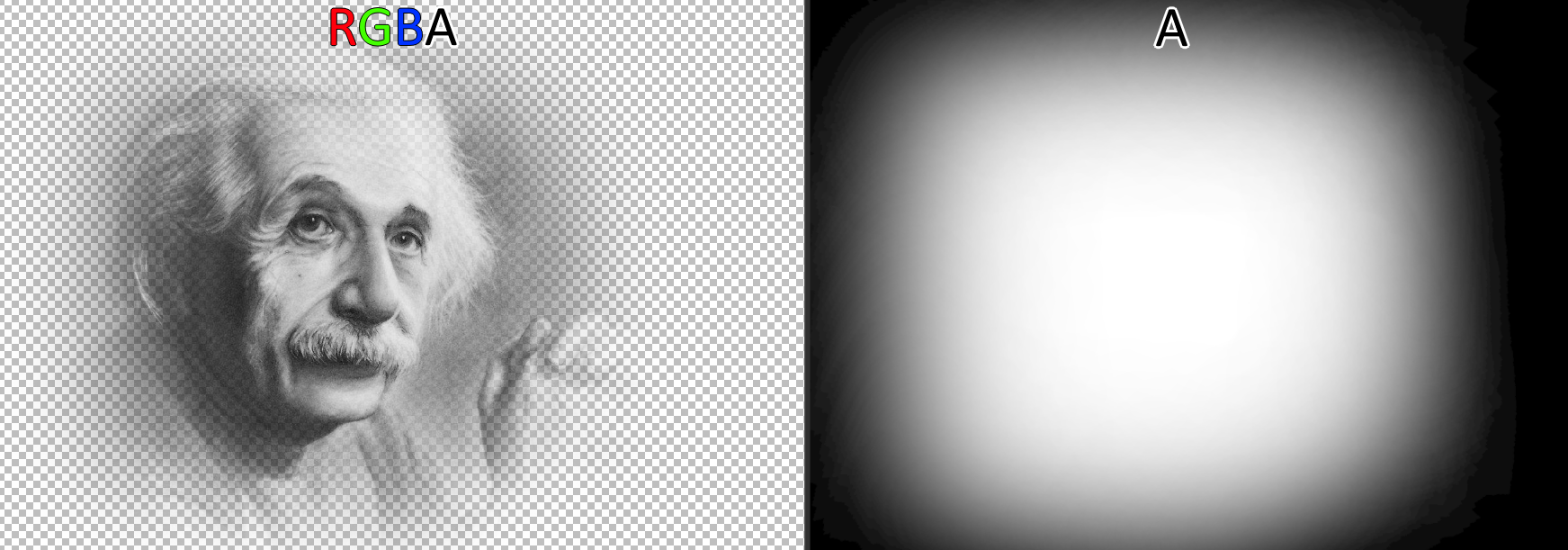
*HTML-en belül, csoport (div) középre van rendezve*

A CSS emellett tartalmazza a többi formai tulajdonságot, például a betűtípus és méretezést, színeket. A CSS lehet akár egy file az interneten is, de ajánlott saját példányt létrehozni, mivel nem biztos, hogy mindig van internet szolgáltatása az applikációnak.

* 1. Formázás, Média

A HTML használatával lehetséges képeket és videókat is importálni, még akár zenét is. Az applikációban használt média elemeket mappába rendezve vannak tárolva, így könnyebb megtalálni a megfelelő file típust. Támogatott file típusok között van: MP4, PNG, WAV. Fontos tudni, hogy ezek a média elemek nem minden böngészőben működnek ugyan úgy, ezért a HTML kódban meg kell szabni, hogy böngészőtől függően hogyan működjön.

A PNG file típus kifejezetten fontos, mivel lehetővé teszi a PNG negyedik szín csatornájának a használatát: Alpha. Az alpha alatt az áttetszőségről beszélünk, a HTML az áttetszőséget a háttérrel való egybeolvasztást teszi lehetővé. Például szeretnénk, hogy a kép a pereme körül átmetszőbb legyen.

 [[5]](#footnote-5)

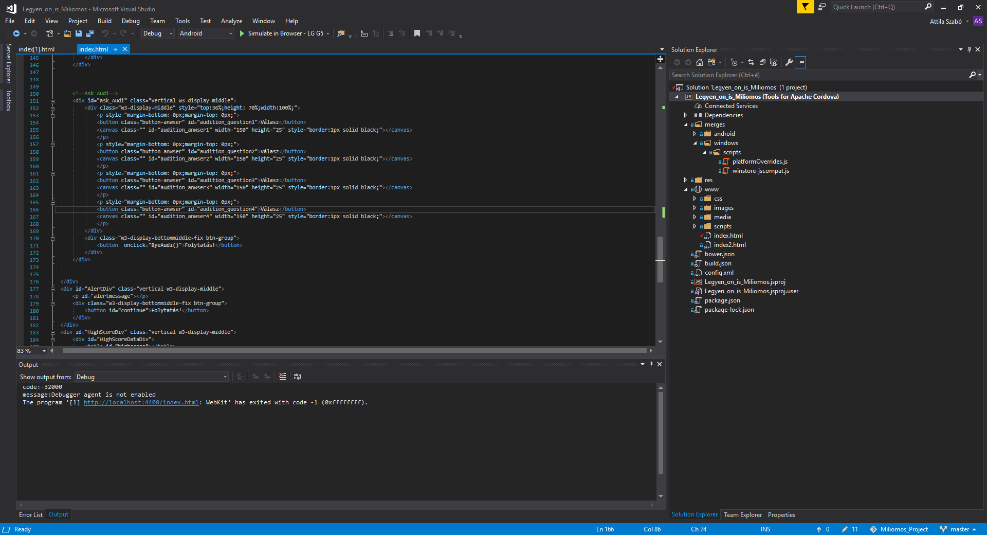
*Alpha Fekete rész = Áttetsző, összeolvad a weboldal színével*

Ez az effektus csak a PNG, vagy TGA file típussal lehetséges, mivel ezek a típusok használnak csak Alpha színe. A hátránya az, hogy több helyet foglal el. Előnye viszont, hogy bármikor, ha változtatjuk a weboldal témáját, az ilyen elemek folyamatosan jól fognak illeszkedni a weboldalra, mivel egyéni áttetszésük van. Ha viszont olyan képformátumra van szükségünk, ami viszonylag jó kép minőséget ad, és file méretet, akkor ajánlott a JPG.

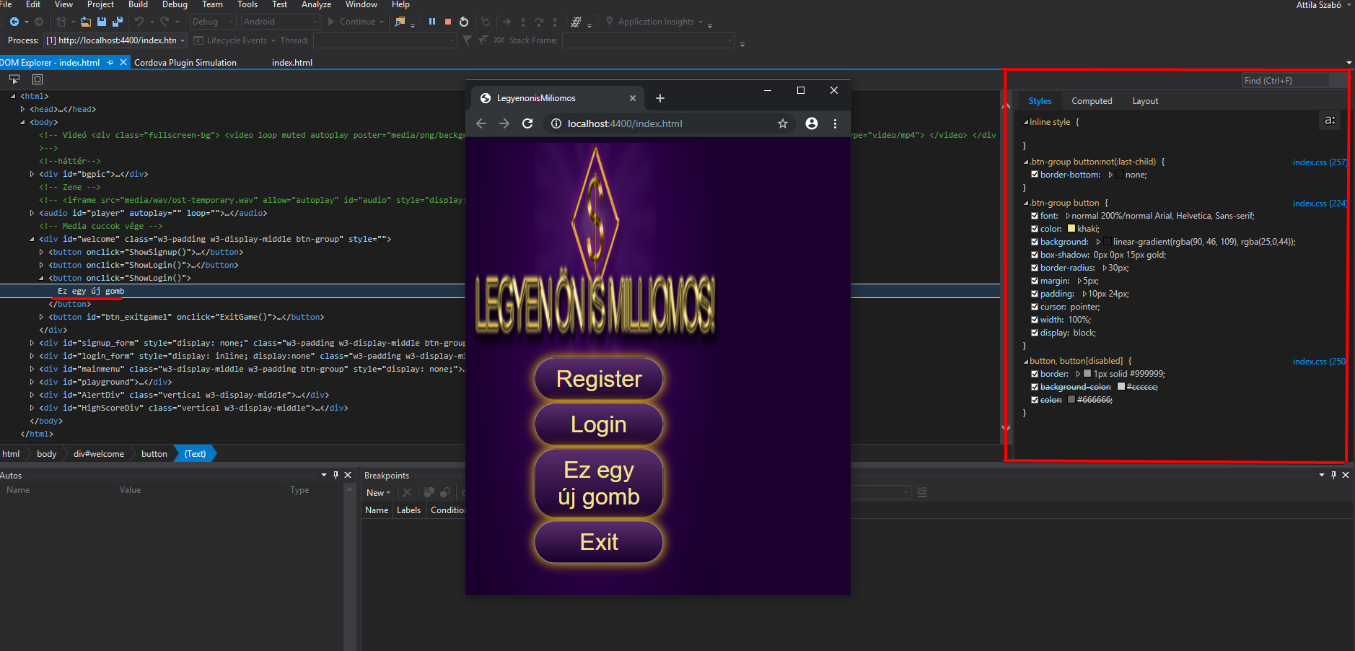
1. Frontend tesztelés

Mivel az applikáció folyamatosan bővül, ezért állandóan tesztelni kell. A tesztelés során a program használata során előforduló hibákat kell megkeresni, általában új funkciók esetében korábban működő részek is elromolhatnak. A tesztelés emellett jó a még félig kész funkciók tesztelésére is gyakorlatban. Mivel webfejlesztésről van szó, ezért gyorsabb a White Box Testing metódus, ami röviden azt jelenti, hogy mi közvetlenül a weboldal fejlesztés során, a metódusok meghívásával kapunk visszajelzést a program működéséről. Így gyorsan tesztelhető az alkalmazás működése, mivel egyből kapunk visszajelzést a működésről[[6]](#footnote-6). A tesztelésre több módszer is lehetséges:

* 1. Visual Studio

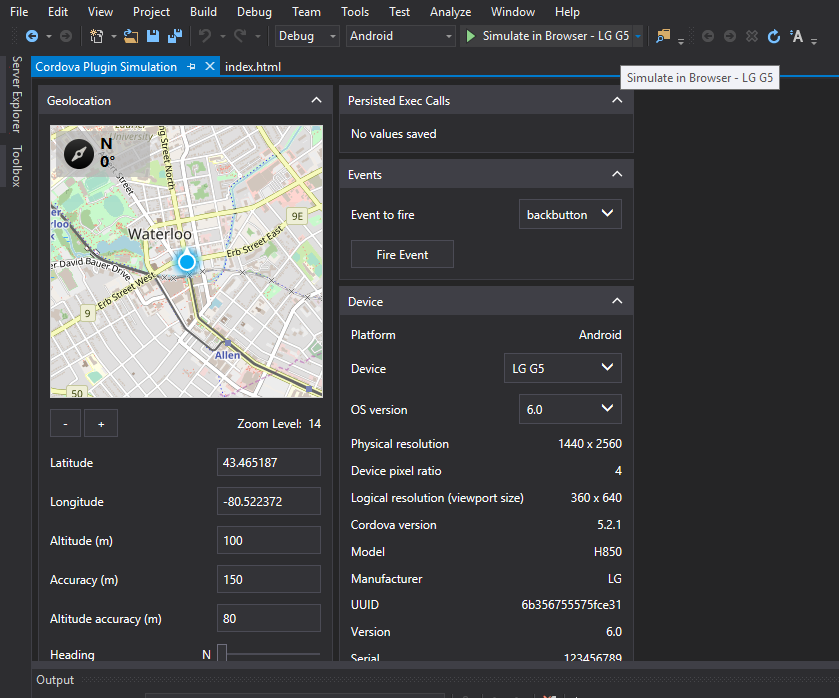


*A Visual Studio file, miután letöltöttük Gitről*



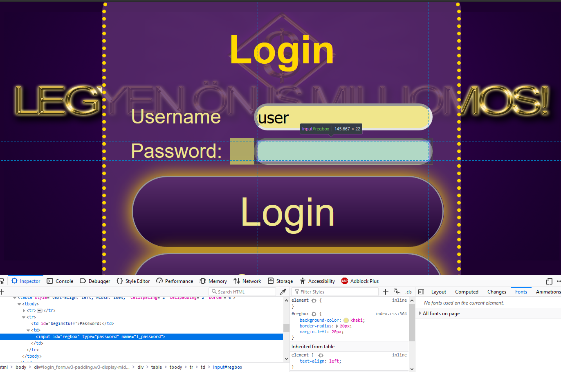
*Valós idejű tesztelésre példa, új sorok egyből megjelennek az applikációban, Visual Studio sajna nem engedi a virtuális képernyő forgatását*

Az applikációt a Visual Studio használatával lehet akár egy telefonos készülékre telepíteni is. A telefonon előbb aktiválni kell a fejlesztői opciókat, így adunk engedélyt a számítógépnek, hogy a telefonra közvetlenül telepítsen alkalmazásokat. A telepítés mellett több lehetőségünk is van, például a debugolás. A debug alatt azt értjük, hogy számítógépünk és a telefonunk össze van kötve, és a kódot látjuk hogyan hívódig meg, még akár változtathatunk is a kódon valós időben. Ez rendkívül hasznos tud lenni, mivel nem kell újra megnyitni az alkalmazást minden alkalommal, amikor változik a kód. Az emuláció [[7]](#footnote-7)emellett rengeteg lehetőségünk is van, akár Android, vagy IOS rendszeren is tudunk emulálni. De ha szimulálni akarunk egy mobil készüléket, akkor annak a beállításainál is lehetőség van az operációs rendszer, és egyéb mobilos paraméterek beállítására, például, hogy mekkora a képarány. Így könnyű tesztelni az alkalmazást, anélkül, hogy vásárolnánk tesztelésre készüléket.



*Android Operációs rendszer beállítása, vagy képernyő arány*

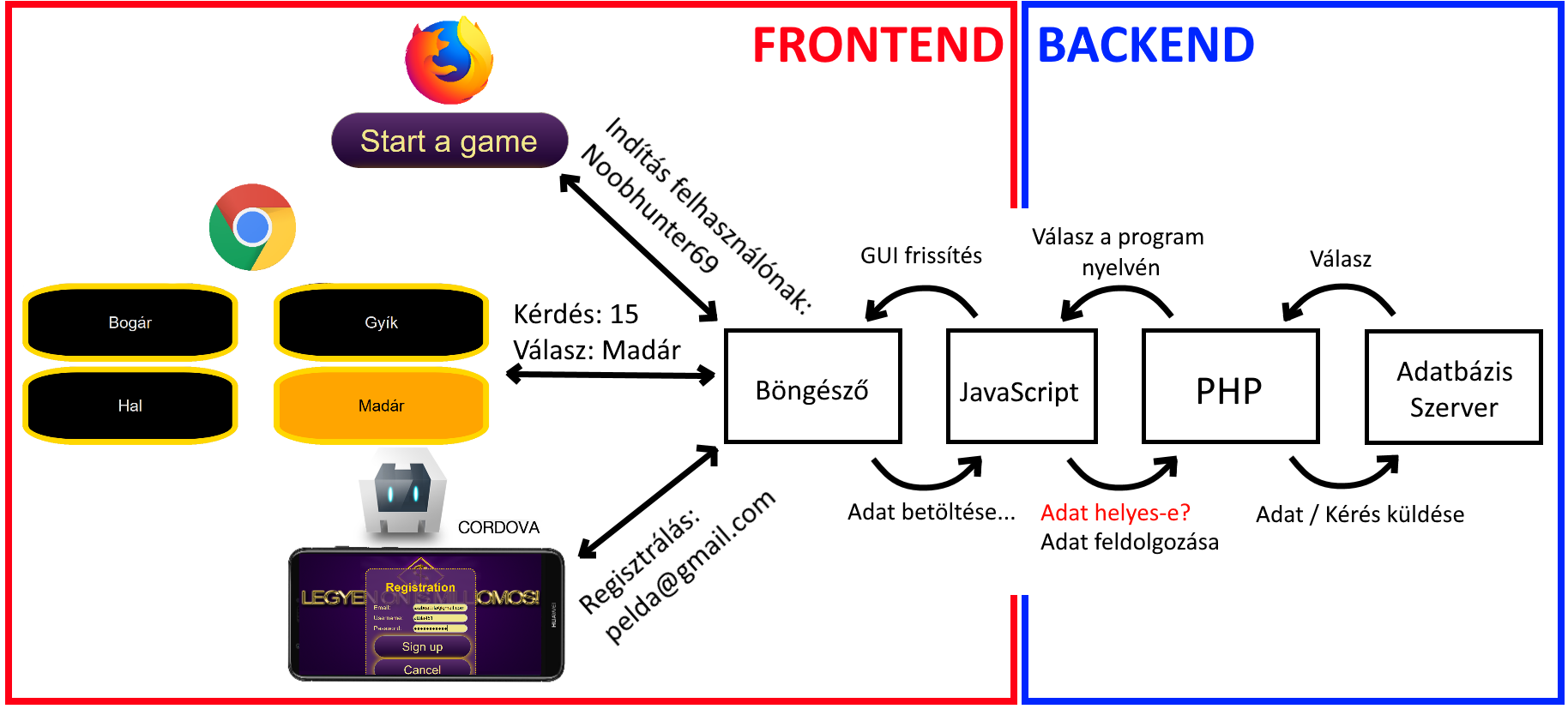
* 1. Böngésző fejlesztői opció



*Firefox böngésző használatával valós idejű szerkesztés*

Lehetséges az applikáció tesztelése Visual Studio nélkül is. A böngészők általában rendelkeznek fejlesztői eszközökkel, ha mi megnyitjuk a HTML file-t, amin dolgoztunk, akkor lehetőségünk van előre megtekinteni a weblap működését. A HTML file megnyitása után a weblap bezárásáig lehetőségünk van bármilyen módosításra. Fontos tudni viszont, hogy ezek a módosítások csak a böngésző példányán jelennek meg, ezeket el kell menteni a HTML file-ba is. A fejlesztői opciók is rendelkezhetnek Androidos, vagy IOS „emulátorral”, ami viszont csak azt csinálja, hogy a weblap telefonos verzióját tölti be, míg a képernyő arányt nem változtatja.

# Fejlesztés



1. Fejlesztés menete

Az alkalmazás bővítésének legfontosabb része a számolások helyes elvégzése, és kiírása. Ennek a fejlesztési rétegnek jól kell kommunikálnia az alkalmazás backend részével. Egy adatbázisba csak helyes értékeket szabad felvinni, különben hibába ütközünk. Ha pedig a felhasználó nem megfelelő értékeket visz fel, akkor azt ki kell jelezni is. Ez nem különbözteti meg azt, hogy a backend ettől függetlenül tartalmazhat hibakezelést, de ezt természetesen tudatni kell a felhasználóval. A fejlesztésnek folyamatosan figyelembe kell vennie a platform függőséget is, például Firefox, Chrome, és telefonra a Cordova.

A programra használt navigálás és műveletek inicializálását a HTML és a JavaScript végzi el. A HTML-ben szereplő objektumok, például gombok által kerülnek meghívásra különböző műveletek.

Ezek között a műveletek között van az adatbázisból való lekérdezés is. Ezekkel az adatokkal pedig úgy kell bánni, hogy azok felhasználhatóak legyenek a megjelenítésre. A feldolgozott adatokat pedig a backend részére, a szerver adatbázisára küldjük további feldolgozásra. Az innen érkező válasz alapján pedig folytatódik a program élet ciklusa. Tehát fontos feltételezni, hogy ezek között a pontok között bármikor lehetnek hibák, és ezeket kezelni kell. Ezek közé a hibajelzések közé sorolható a szövegdobozos hibaüzenetek, amik visszajelzést adnak arról, hogy nem megfelelő adatot vitt fel a felhasználó, vagy ha esetleg éppen nem elérhető a szerver.

1. Backend fejlesztés

Elsősorban az applikáció „agya” kell, ahol tárolni lehet a játékos teljesítményét, és bővíteni lehet a játék során előforduló kérdéseket. Ezt megelőzi az adatbázis tervezése. Fontos tudni, hogy milyen adatokkal akarunk dolgozni, később ezeket kiegészíteni már nehezebb. A szerveroldali kommunikációnak emellett kezelnie kell a helytelen adatokat, máskülönben a felhasználók visszaélhetnek a program hibájával.

1. Frontend fejlesztés

Egy működő alappal már lehet fejleszteni a frontendet. Itt jön figyelembe a tesztelés hangsúlya, mivel van lehetőség arra, hogy a kód futtatása közben fejlesszük a felhasználói felületet. Az adatbázist ilyenkor nem kell már szerkeszteni, később esetleg lehet bővíteni, de több lehetőségünk van a frontenden hasznosítani ezeket. Ugyan azok az adatok több helyen is előfordulnak, és többször lehet újra hasznosítani őket. Ilyenre példa a felhasználó nevének kiírása, vagy a „kérdések” lekérdezése.

1. Integrated Development Environment [↑](#footnote-ref-1)
2. https://notepad-plus-plus.org/ [↑](#footnote-ref-2)
3. https://www.w3schools.com/cssref/css3\_browsersupport.asp [↑](#footnote-ref-3)
4. https://www.w3schools.com/w3css/w3css\_display.asp [↑](#footnote-ref-4)
5. https://opimedia.azureedge.net/-/media/images/utr/editorial/articles/online-articles/2014/04-01/albert-einstein-the-humanitarian/albert-einstein-jpg.jpg [↑](#footnote-ref-5)
6. http://softwaretestingfundamentals.com/white-box-testing/ [↑](#footnote-ref-6)
7. Egy másik készülék használatát utánozza [↑](#footnote-ref-7)