**ZÁRÓDOLGOZAT**

**Müller Ottó**

**Szoftverfejlesztő**

**BAJAI SZC TÜRR ISTVÁN TECHNIKUM**

**SZoftverfejlesztő**

ZÁRÓDOLGOZAT

Kripto Bázis

**Müller Ottó**

2024

**NYILATKOZAT A ZÁRÓDOLGOZATRÓL**

Alulírott .......................................................................... (név) tanuló **kijelentem, hogy** ...................................................................................... című záródolgozatomat (nyomtatott és elektronikus formában) a Bajai SZC Türr István Gazdasági Szakgimnáziumának pedagógusai és tanulói:

· felhasználhatják (pl. hivatkozás alapjául, olvasótermi használatra) későbbi mun-kájukhoz a szerzői jogok tiszteletben tartása mellett).

· nem használhatják fel (titoktartási nyilatkozat csatolása mellett).

Ugyanakkor kijelentem, hogy a záródolgozat **saját munkám eredménye.**

Baja, 20.......................

.................................................................

aláírás

Tartalom

[Tartalom 4](#_Toc165445528)

[1. Bevezetés 1](#_Toc165445529)

[1.1. Témaválasztás 1](#_Toc165445530)

[1.2. Célkitűzések 1](#_Toc165445531)

[2. Felhasználói dokumentáció 2](#_Toc165445532)

[2.1. Röviden az oldalról 2](#_Toc165445533)

[2.2. Rendszerkövetelmények 2](#_Toc165445534)

[2.3 Weboldal felépítése 3](#_Toc165445535)

[3. A fejlesztői dokumentáció 10](#_Toc165445536)

[3.1. Az alkalmazott fejlesztői eszközök 10](#_Toc165445537)

[3.3. Adatbázis felépítése 13](#_Toc165445538)

[3.4. Forráskódok 15](#_Toc165445539)

[3.5 Főbb szkriptek 16](#_Toc165445540)

[3.5.1. currenciesCtrl.js 16](#_Toc165445541)

[3.5.2. logCtrl.js 18](#_Toc165445542)

[3.6. Mappastruktúra 21](#_Toc165445543)

[4. Összegzés 23](#_Toc165445544)

# Bevezetés

## Témaválasztás

A szakdolgozatom témáját erősen inspirálta a kriptovaluták általi érdeklődésem, ami az elmúlt évek során alakult ki. Mindig is a webfejlesztés érdekelt a programozáson belül, úgyhogy nem volt kétség, hogy egy weboldallal valósítom meg elképzeléseimet. Maga a weboldal egy egyszerű, felhasználóbarát, fórumszerű weboldal hírekkel, árfolyammal, profilok testreszabásával és maga a fórum, ahol a felhasználók megoszthatják véleményeiket különböző témákról, amiket saját maguk vetnek fel.

A weboldal készítéséhez HTML5, CSS, Javascript(NodeJS) nyelveket használtam különböző keretrendszerekkel együtt mint a Bootstrap 5.3, AngularJS, az adatbázist a MySQL és XAMPP segítségével hoztam létre.

## 1.2. Célkitűzések

A weboldalammal egy olyan felületet kívánok létrehozni, amely ismeretterjesztő, felhasználókra fókuszált közeget teremt. Regisztrációval és profilok testreszabásával adok szabad kezet a felhasználóknak, amit lementhet és módosíthat is A legfontosabb szempont, hogy a felhasználó testreszabhatja a saját igényei szerint az oldalt.

Szeretném a felhasználókat rávenni arra, hogy komolyan vegyék a kriptovalutákat és jövőt lássanak benne, mivel ez segíthet az embereknek biztos pénzügyi alapokra helyezni jövőjüket.

A weboldal az alábbi elemeket tartalmazza:

* Kezdőlap – Hírek megjelenítése, ami bárki számára látható (nem szükséges regisztráció)
* Regisztráció – Egy panel, amivel regisztrálhat a felhasználó
* Bejelentkezés – Egy panel, amivel a felhasználó bejelentkezhet és jogosultságot szerez a főbb funkciókhoz
* Profilmódosítás – Testreszabhatják profiljukat
* Tudásbázis – Különböző kriptovaluták rövid leírása
* Árfolyam – Legfrissebb árak megjelenítése
* Fórum – felhasználók által létrehozott posztok
* Kijelentkezés – kijelentkezteti a felhasználót, így visszakerül a bejelentkezés panelre

# Felhasználói dokumentáció

## 2.1. Röviden az oldalról

Az oldalt úgy terveztem, hogy bárki számára használható legyen, de főleg azoknak ajánlanám, akik még kezdők a témában és szeretnék minél több ember véleményét hallani több szempontból is. Ha valaki bejelentkezik azonnal hozzáfér mindenhez, ami az oldalon van. Árfolyamot nézhet, fórumot böngészhet, posztot hozhat létre és mindig elolvashatja a legújabb híreket. Mivel felhasználóbarát az oldal senkinek nem lesz nehezére navigálni az oldalon és aktív tagja lenni annak. Ha az oldal kikerülne az internetre, pontosan úgyanúgy lehetne használni, mint bármelyik másik közösségi oldalt. Van regisztráció, amihez tartozik validáció és az adatok elmentésre kerülnek adatbázisba, tehát hosszútávon is működik.

Amennyiben nem regisztrál a felhasználó, csak az alapvető oldalakat tudja megtekinteni, mint a szabályzat vagy a hírek.

## 2.2. Rendszerkövetelmények

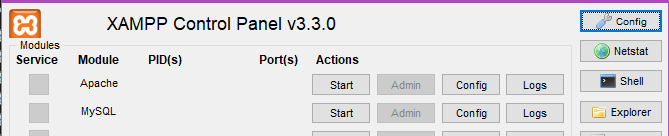
A weboldal futtatásához egy asztali számítógépre vagy laptopra van szükségünk, ami képes egy friss verziójú böngészőt futtatni, mivel az oldal arra van fejlesztve.

**Szoftver követelmények**

Windows 10 vagy későbbi operációs rendszer verziót ajánlott használni, mivel a böngészők, amikre fejlesztve lett, ezeket támogatják.

XAMPP programra lesz még szükségünk, amit le kell töltenünk legalább 8.0-ás verziót vagy annál újabbat. Ebben található a phpMyAdmin és az Apache webszerver telepítése.

A weboldal futtatásához le kell töltenünk az XAMPP-ot, feltelepíteni, majd újraindítani a számítógépet. Az újraindítás után keressük meg az „xampp” mappát és futtasuk a(z) „xampp-control.exe” fájlt, amivel elindul a program. Ha megjelent a program ablaka, nyissuk meg és nyomjunk rá a két start gombra, ami az „Apache” és „MySQL” mellett jobbra található.



. ábra XAMPP

Ha elindultak, zöldre fognak váltani és a két admin gomb elérhető lesz. Nyomjunk rá a MySQL sorjában lévő admin gombra, ami a phpMyAdmin oldalát nyitja meg böngészőben. A felső menüsorból az ’Importálás’ menüpontot választva továbblép az importáláshoz szükséges oldalra. Itt a harmadik sorban található a „Számítógép Tallózása” felirat, ami mellett egy Fájl kiválasztása gomb található. Ezt lenyomva megjelenik, az ablak ahol ki lehet választani a database mappából a „kriptobazis.sql” nevű állományt. A kiválasztás után az ’Indítás’ gombra kattintva lefut az adatbázist létrehozó parancssor. Ezután az XAMMP nevű programot bezárni nem szabad, és benne az Apache és a MySQL programoknak futnia kell.

Ezek után még szükségünk lesz a „Visual Studio Code” nevű programra. Ezt is töltsük le az internetről és telepítsük fel a legfrissebb verziót. Ha ez megvan, nyissuk meg a programot és keressük meg az „Open Folder” funkciót a „File” menüpontban. A fájlkezelő ablakban most keressük meg a weboldal mappáját és nyissuk meg a programban. Ha minden elemet beimportált jobb klikkel nyomjunk rá az API nevű mappára az explorerben (bal oszlop) és nyomjunk rá bal klikkel az „Open in integrated Terminal” pontra. Most a képernyő alján megjelent a terminál. Ide írjuk be azt, hogy „npm install”, amivel letölthetjük a szükséges fájlokat a program futásához. Ha minden letöltés sikerült indítsuk el a szervert „npm run dev” paranccsal. Ilyenkor üzenetet is kapunk, hogy fut a szerver. Ezután a bal oszlopban (Explorer) keressük meg az „index.html” fájlt jobb gombbal kattintsunk rá és majd az „Open with Live Server” pontra. Ilyenkor megnyílik a weboldal az alapértelmezett böngészőben.

## 2.3 Weboldal felépítése

A weboldal alapja egy oldalsó navigációs oszlop köré épült, ami egy kinyitható „template” nagyjából viszont kicsit átalakításra került. A navigációs gombokat vagy linkeket rövid tervezés alapján hoztam létre, leginkább olyanokat, amikről úgy gondoltam, hogy jól meg lehet oldani és értelme is van az oldalon.

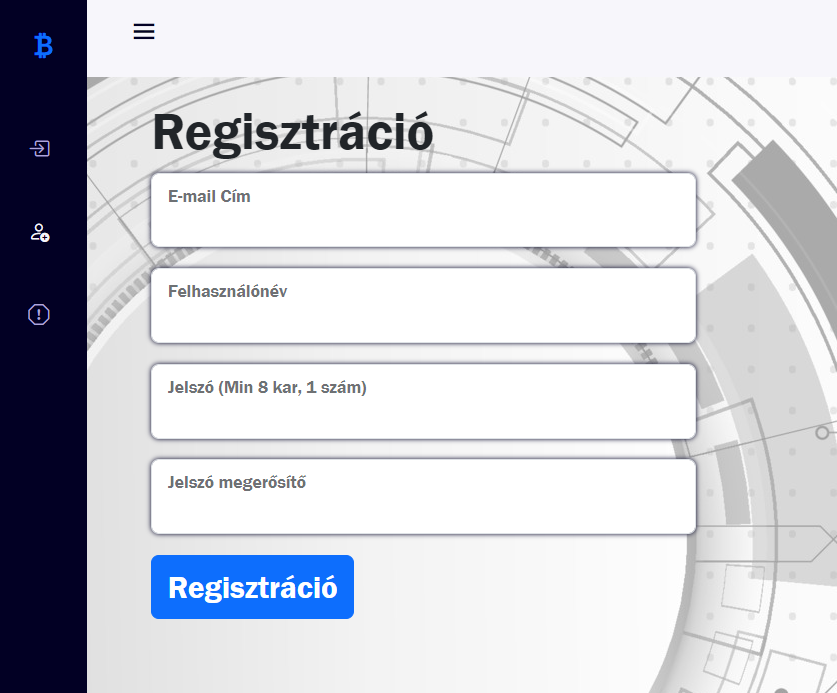
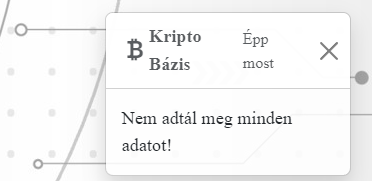
A weboldal átláthatósága és kinézete a színekben, a betűtípusban és az elrendezésben rejlik. Nagyon fontos, hogy ne legyen túl sok zavaró tényező egy nézeten, mert zavarja az átláthatóságot. Nem bejelentkezett állapotban csak a kezdőlaphoz, regisztrációhoz és bejelentkezéshez férünk hozzá.

**Kezdőlap**

A kezdőlapon egy gomb fogad minket „hírek frissítése” néven, amire ha rányomunk előjön egy mozgó kis töltőikon, ami pár másodpercig látható, majd megjelennek a hírek és az oldal a jobb alsó sarokban még értesít is minket, hogy mi történt.

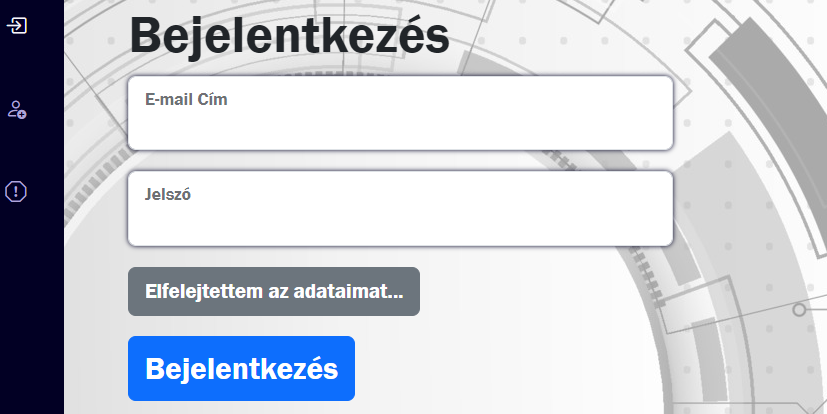
**Regisztráció**

A regisztráció kinézete megegyezik a bejelentkezéssel. Amikor a regisztrációra nyomunk az oldalsó sávban elvisz minket a regisztráció nézethez, ami 4 mezővel fogad minket.



. ábra Regisztrációs Panel

A mezőkre rá van írva, hogy milyen adatokat kell megadni illetve az is, hogy azoknak miből kell állnia (jelszó mező). A kék regisztráció gombbal tudunk kommunikálni az oldallal, hogy szeretnénk regisztrálni és hogy ellenőrizze a bevitt értékeket. Ha bármilyen hiba lép fel mint mondjuk nem töltöttük ki az összes mezőt vagy a jelszó nem tartalmazza a kért karaktereket és nem is egyezik meg a jelszó megerősítővel vagy ha az E-mail mező nem tartalmaz „@” –ot akkor az oldal a jobb alsó sarokban figyelmeztet, hogy mi történt vagy mi a gond. Ezekből a hibaüzenetekből a regisztrációnál összesen öt darab van. Abban az esetben, ha minden mező jól van kitöltve, kapunk szintúgy egy üzenetet, hogy sikeresen regisztráltunk. Ezek után a regsztrált E-mail címmel és jelszóval átmehetünk a bejelentkezés nézetre. Bejelentkezés nézetben két mező és két gomb fogad. Az első mező az E-mail címet kéri míg a másik a jelszót. Ezt az „Elfelejtettem az adataimat…” című gombbal tehetjük meg. Ha beírjuk az E-mail címűnket az első mezőbe és rányomunk a szürke gombra értesítést kapunk hogy az E-mail el lett küldve. Ha olyan E-mail címet adunk meg ami nincs regisztrálva akkor egy hibaüzenet fogad minket hogy nincs ilyen regisztrált E-mail cím.



3. ábra Bejelentkezés Panel

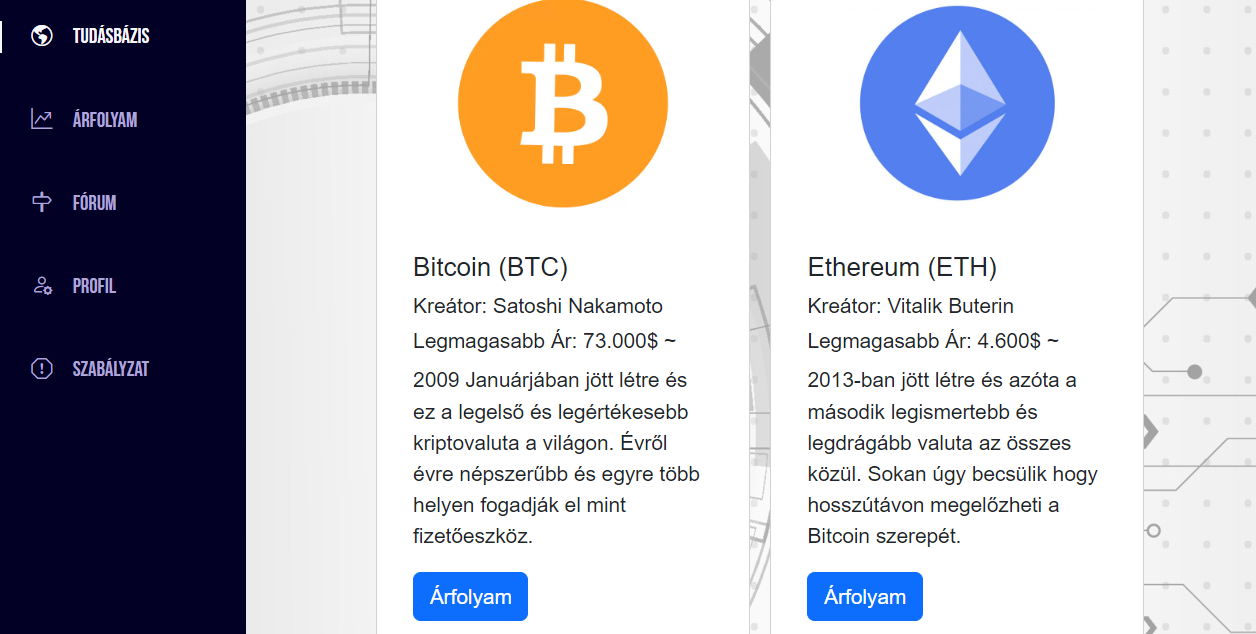
Bejelentkezés gomb lenyomásakor az oldal ellenőrzi a bevitt értékeket és annak megfelelőlen küldd egy választ a felhasználónak. Üres mezőnél figyelmeztet, hogy üresen hagyta a felhasználó őket, rossz E-mail vagy jelszó kombinációnál jelez hogy rossz adatokat adott meg.

Sikeres bejelentkezésnél betekintést nyerünk az oldalra, mint felhasználó. Itt már számos navigációs link áll rendelkezésre.

**Tudásbázis nézet**

A tudásbázis nézetre navigálva egy enciklopédia szerű oldalon kötünk ki, ami információval lát el számos kriptovalutáról. Ezek mind-mind úgynevezett kártyákon helyezkednek el, leírást és egy gombot tartalmaznak. Ha a gombra rányomunk az oldal rögtön átirányít minket a következő nézetre, ami az árfolyam nézet.

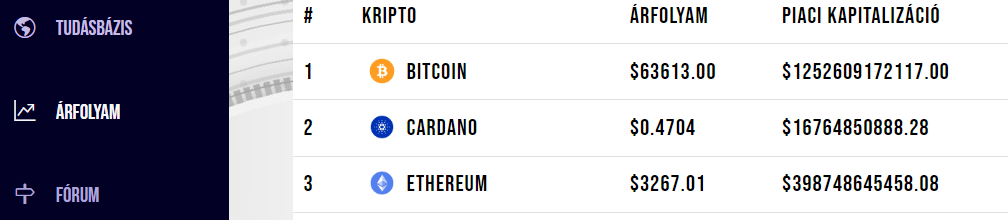
**Tudásbázis kinézete**



4. ábra Tudásbázis nézet

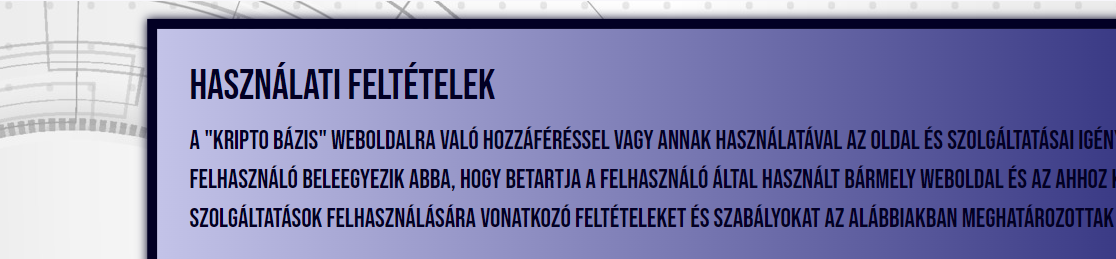
**Árfolyam nézet**

Az árfolyam nézet magáért beszél hiszen itt kriptovaluták árfolyamát tekinthetjük meg rövid várakozás után, mivel mindig a legújabb adatokat jeleníti meg az oldal, amihez szüksége van egy kis időre. Ezt egy töltőikon jelzi, ami addig tölt, amíg meg nem jelenik az árfolyamtábla.

Itt számos adatot kapunk. A kriptovaluta nevét, jelenlegi árát, a piaci kapitalizációját, azt hogy mekkora forgalma volt az elmúlt 24 órában, és azt is hogy mennyi belőle a teljes kínálat. Akárhányszor visszalátogatunk ide mindig lefrissül és a legújabb számokat kapjuk.

5. ábra Árfolyam nézet

**Szabályzat nézet**

A szabályzat nézet egy szabályzatról szóló leírás.

6. ábra Szabályzat nézet

Ahhoz hogy az oldalon regisztrált fiókunkat megtarthassuk, be kell tartanuk ezeket a szabályokat különben az adminok intézkednek és törlik a felhasználót.

**Profil nézet**

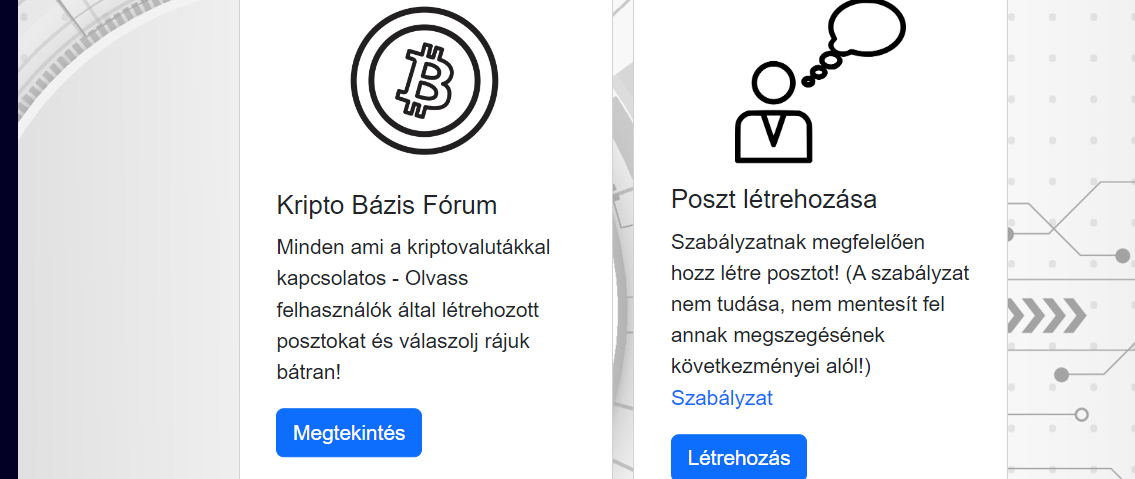
Ebben a nézetben testreszabhatjuk profilunkat: megadhatjuk a lakhelyünk vagy becenevünk.



7. ábra Profil nézet

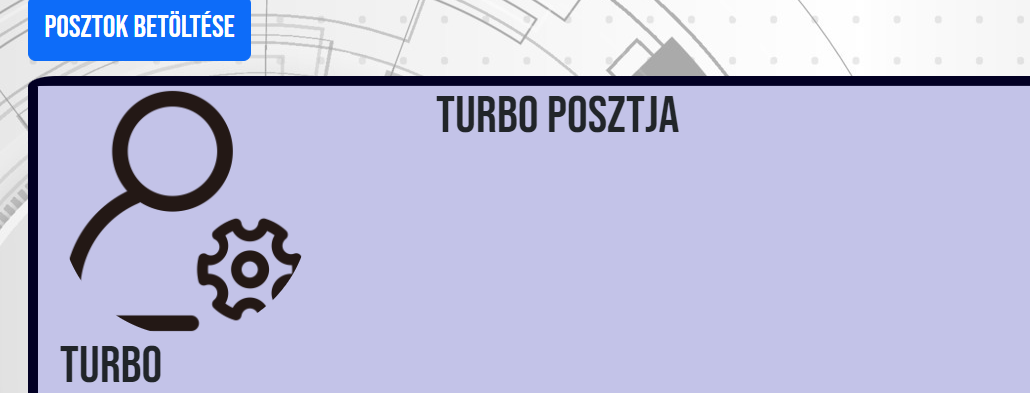
Itt is hasonlóan zajlanak le a dolgok mint a regisztrációnál. A „Mentés” gombbal tudjuk a mezőkbe felvitt adatokat lementeni a fiókhoz, amik a következők: Becenév, egyéni állapotüzenet (gondolatok), Lakhely, és egy mező ahol megadhatod, hogy milyen más közösségi platformon vagy jelen. Itt érvényesülnek a szabályzatban megírt pontok miszerint valós adatokat adjunk meg, amik nem sértőek, nem okoznak kellemetlenséget, nem aggresszívak, nincs másra negatív hatással.

**Fórum nézet**

A fórum nézeten láthatunk két kártyát, amin van két gomb. Az egyik elvisz minket a posztokhoz, ahol láthatjuk a felhasználók által létrehozott posztokat. A másik elvisz minket egy oldalra ahol saját posztunkat hozhatjuk létre és még leírást is kapunk, hogy ügyeljünk a szabályzatra.

8. ábra Fórum nézet

A fórum betöltéséhez csak nyomjunk rá a nagy kék gombra, hogy előhozzuk a posztokat.



9. ábra Fórum lista nézet

Itt minden posztot egymás alá sorol be, ahol megjeleníti a poszt címét, szerzőjét, leírást, és a készítés dátumát.

**Poszt létrehozása**

Egy űrlapot látunk magunk előtt pontosan három mezővel. Ezeket kell megadnunk: Poszt címe, Poszt írója, Poszt leírása (max 800 karakter). Ha üresen hagyjuk valamelyik mezőt természetesen a poszt nem fog létrejönni és üzenetet kapunk hogy töltsünk ki minden mezőt.



10. ábra Hibaüzenet

Ha minden mezőt kitöltöttünk, eldönthetjük, hogy létrehozzuk a posztot vagy inkább a bevitt értékeinket töröljük és újra írjuk. Ha létrehozzuk a posztot, akkor az felmegy az adatbázisba és legközelebbi fórumlátogatásunknál már mi is látni fogjuk.

# A fejlesztői dokumentáció

## 3.1. Az alkalmazott fejlesztői eszközök

A weboldalt a Visual Studio Code nevű programban készítettem el. Ez az egyik legelterjedtebb fejlesztői környezet a webfejlesztésre és ingyenesen használható. A Javascript, CSS, HTML részét mind ebben készítettem el. A weboldal megnyitásához a Brave és Google Chrome alkalmazásokat használtam, a Google Chrome-ot azért, mert az a legelterjedtebb a Brave-t pedig azért, mert szeretem. Adatbázis eléréséhez a „XAMMP” programot használtam mivel ez tartalmaz „Apache” webkiszolgálót és „phpMyAdmin” modult is, ami segíti a „MYSQL” menedzselését. A weboldalon felhasznált ikonokat a Bootstrap hivatalos oldaláról töltöttem le, míg a képeket a Google képkeresőjéből. A dokumentumban található képeket a Windows 10 képmetszőjével készítettem és illesztettem be. A munkám folyamatos haladását a Git verziókövető szoftverrel biztosítottam, hogy a GitHubról mindig mindenhol elérjem és lássam, hogy mikor melyik modulban ejthettem esetleg hibát.

**XAMPP[[1]](#footnote-1)**

A XAMPP egy ingyenes és nyílt forráskódú szoftvercsomag, amely teljes körűen támogatja a következő webtechnológiákat: Apache HTTP szerver, MySQL adatbáziskezelő rendszer, PHP és Perl programozási nyelvek, valamint phpMyAdmin adatbáziskezelő eszköz.

Az XAMPP segítségével könnyedén telepíthetünk egy teljes körű fejlesztői környezetet lokális gépünkre, amely lehetővé teszi számunkra, hogy webalkalmazásokat hozzunk létre és teszteljünk, még mielőtt azokat a valódi szerverre telepítenénk.

**Apache[[2]](#footnote-2)**

Az Apache webszerver egy ingyenes, nyílt forráskódú webszerver szoftver, amelyet a Apache Software Foundation fejleszt. A világ egyik legnépszerűbb webszerver alkalmazása, amelyet a legtöbb webhely üzemeltetésére használnak.

Az Apache webszerver támogatja a HTTP és HTTPS protokollokat, valamint rendelkezik további funkciókkal, mint például virtuális hosztok, SSL titkosítás, URL újraírás, hozzáférési jogosultságok kezelése, naplózás és sok más.

**phpMyAdmin[[3]](#footnote-3)**

A phpMyAdmin egy ingyenes, webalapú adatbázis-kezelő eszköz, amelyet PHP-ban írtak és MySQL adatbázisokhoz terveztek. Segítségével könnyedén kezelhetjük az adatbázist, létrehozhatunk, törölhetünk, módosíthatunk táblákat és rekordokat, valamint futtathatunk SQL lekérdezéseket és parancsokat.

**Visual Studio Code[[4]](#footnote-4)**

A Visual Studio Code egy ingyenes forráskód szerkesztő és fejlesztőeszköz, amelyet a Microsoft fejlesztett ki. Az alkalmazás keresztplatformos, és támogatja a kód szerkesztését és hibakeresését számos programozási nyelven, például JavaScript, TypeScript, Python, Ruby, C++.

A VS Code könnyű és gyors, valamint rengeteg kiegészítővel és funkcióval rendelkezik, mint például beépített Git integráció, kódszerkesztési segédletek, automatikus kódformázás, debuggolási eszközök, és sok más. Azért a Visual Studio Code-ot választottam, mivel rengeteg tapasztalatom van benne, hiszen közel hat éve dolgozom benne majdnem minden nap. Könnyen kiismerem magam a felületen és tetszik a stílusa.

**Git verziókövető[[5]](#footnote-5)**

A Git egy elosztott verziókezelő rendszer, amelyet a forráskód verziók kezelésére használnak. A Git lehetővé teszi a fejlesztők számára, hogy nyomon kövessék a változásokat a projekt különböző verziói között, hatékonyan együttműködjenek más fejlesztőkkel, illetve könnyedén vissza tudjanak térni korábbi változatokhoz.

**Github[[6]](#footnote-6)**

A GitHub egy online verziókezelő és egy projekt-menedzsment platform, amely segít a fejlesztőknek megosztani kódot, együttműködni másokkal és nyomon követni a változtatásokat a kódban. A GitHub lehetővé teszi, hogy programozók tárolják kódjukat a felhőben, nyomon kövessék a változtatásokat, javítsák hibákat és együtt dolgozzanak más fejlesztőkkel a különböző projektekben. A platformot rendkívül széles körben használják a fejlesztők világszerte, és számos lehetőséget kínál a kódkészítés és projektmenedzsment hatékonyabbá tételére.

**NodeJS[[7]](#footnote-7)**

A Node.js egy nyílt forráskódú, JavaScript-alapú környezet, amely lehetővé teszi a fejlesztők számára a szerveroldali programozást. A Node.js a Google Chrome V8 JavaScript motorjára épül, amely gyors és hatékony futást biztosít a JavaScript kódoknak.

A Node.js-t általában alkalmazások készítésére használják, például webalkalmazások, API-k, szerveroldali alkalmazások és mikroszolgáltatások fejlesztésére. A Node.js segítségével könnyen lehet aszinkron és eseményvezérelt kódokat írni, amelyek lehetővé teszik a hatékony és skálázható alkalmazásfejlesztést.

**Bootstrap (5.3)[[8]](#footnote-8)**

A Bootstrap egy nyílt forráskódú front-end keretrendszer, amely segíti a webfejlesztőket a weboldalak és webalkalmazások készítésében. A Bootstrap tartalmaz egy sor előre elkészített HTML és CSS stílusokat, amelyek segítségével könnyedén és gyorsan készíthetünk reszponzív, esztétikus és mobilbarát dizájnokat. Emellett tartalmaz további komponenseket, mint például gombok, űrlapok, navigációs sávok és modálos ablakok, amelyek segítik a fejlesztés folyamatát.

**Javascript[[9]](#footnote-9)**

Javascript egy interpretált programozási nyelv, melyet általában webfejlesztéshez használnak. Az elsődleges feladata a weboldalak dinamikus és interaktív funkcióinak létrehozása. A nyelv széles körű támogatást élvez, és számos keretrendszerrel és könyvtárral rendelkezik, amelyek segítenek a fejlesztés során. A nyelvet könnyű megtanulni és használni, így ideális választás kezdő fejlesztők számára is.

**CSS (Cascading Style Sheets)[[10]](#footnote-10)**

CSS egy stíluslap nyelv a weboldalak és webalkalmazások szép megjelenésének és elrendezésének definiálására. A CSS segítségével lehet formázni a szöveget, háttérképet, elrendezést, színeket és egyéb vizuális elemeket a weboldalon.

**HTML 5[[11]](#footnote-11)**

Az HTML 5 egy modern webfejlesztési technológia, amely lehetővé teszi interaktív és dinamikus weboldalak létrehozását. Az HTML 5 támogatja a multimédiás elemeket, a videókat, az audiókat és a grafikákat, valamint lehetőséget ad az offline alkalmazások használatára is.

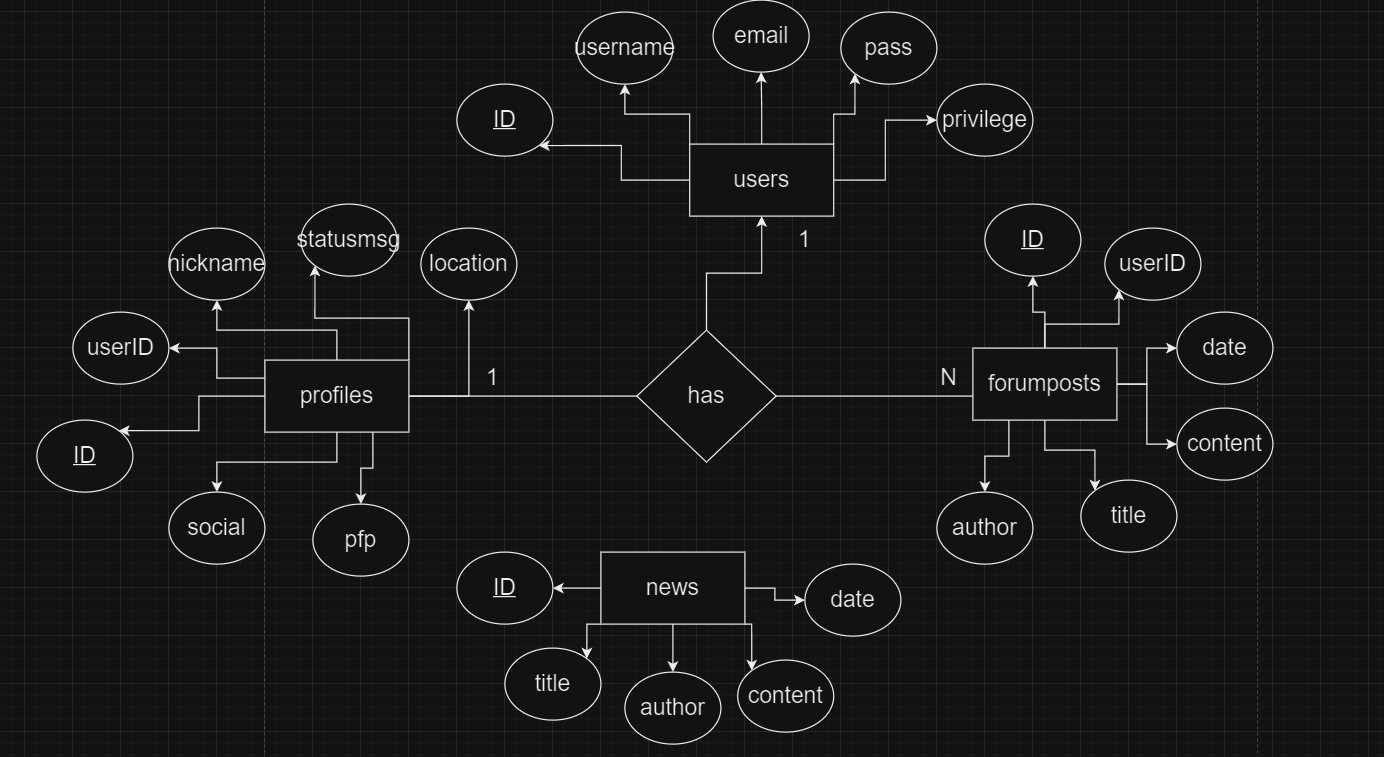
**AngularJS[[12]](#footnote-12)**

Az AngularJS egy nyílt forráskódú JavaScript keretrendszer, amelyet a Google fejlesztett ki a webes alkalmazások fejlesztésére. Az AngularJS segítségével könnyedén készíthetünk dinamikus, egyoldalas alkalmazásokat (SPA), melyek interaktívak és felhasználóbarátak. Az AngularJS segít az adatok és felhasználói felület közötti kapcsolatok könnyű kezelésében, valamint támogatja a komponensalapú fejlesztést is.

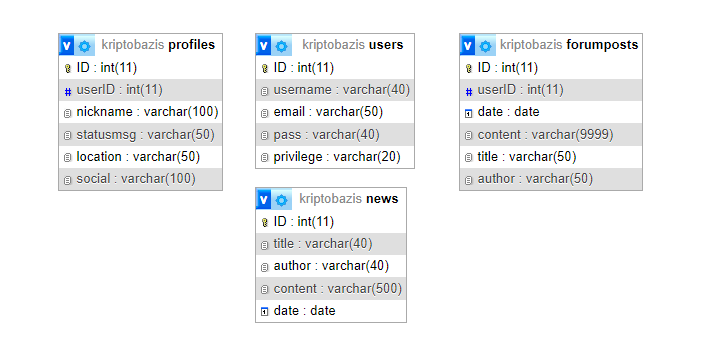
## 3.3. Adatbázis felépítése

Legelső feladatom volt, hogy megtervezzem az adatbázist mivel eköré épűlnek a funkciók.

Alapvetőleg minden a users tábla köré épül tehát a profilok, és posztok a bejelentkezett felhasználó ID-ját átveszik és azt illesztik be a saját userID mezőjükbe.

**Adatbázis ER modell**

. ábra ER modell



. ábra Relációs modell

Táblák:

users tábla

* ID – int típusú, automatikusan növekvő minden újonnan regisztrált felhasználó ID-ja 1-el nagyobb mint az előző.
* username – varchar típusú, max 40 karakter
* email – varchar típusú, max 50 karakter
* pass – varchar típusú max 40 karakter
* privilege – varchar típusű max 20 karakter

profiles tábla

* ID – int típusú, automatikusan növekvő
* userID – int típusú a users tábla ID-ját veszi át script segítségével.
* nickname – varchar típusú max 100 karakter
* statusmsg – varchar típusú max 50 karakter
* location – varchar típusú max 50 karatker
* social – varchar típusú max 100 karakter

forumposts tábla

* ID – int típusú, automatikusan növekvő mindig +1 posztokként.
* userID – int típusú a users tábla ID-ját veszi át script segítségével.
* date – dátum típusú, script segítségével veszi át az akkori dátumot amikor bekerül
* content – varchar típusú 9999 karakter max, limitálva script segítségével
* title – varchar típusú 50 karakter max
* author – varchar típusú 50 karakter max

news tábla

* ID – int típusú, automatikusan növekvő
* title – varchar típusú max 40 karakter
* author – varchar típusú max 40 karakter
* content – varchar típusú max 500 karakter
* date – dátum típusú

## 3.4. Forráskódok

Minden script Javascript nyelven van írva ,amik működtetik az alábbi kontrollereket és Backend hívásokat.

* Regisztráció-kezelés – regCtrl.js és app.js összetétele
* Bejelentkezés-kezelés – logCtrl.js és app.js összetétele
* Kezdőlap-kezelés – newsCtrl.js és app.js összetétele
* Főoldalstruktúra (bal oldali bár) – MainCtrl.js és app.js összetétele
* Profilkezelés – profileCtrl.js
* Hírek kezelése – newsCtrl.js
* Fórumkezelés – forumCtrl.js

API modulok

* Adatbázis-kezelés & endpointok – database.js, server.js
* E-mail küldés – email.js
* fájfeltöltés – fileupload.js
* egyéb funkciók – functions.js

Stílusfájlok

* main.css – minden elem kinézete itt van megcsinálva
* bootstrap.min.css – a bootstrap kinézetek innen kerülnek meghívásra

Ikonok, képek, fontok

* Bootstrap-icons-1.11.2 – az ikonok könyvtára
* img mappa – felhasznált képek könyvtára
* textfonts mappa – egyénileg használt font stílus a google fonts-ról

Egyéb JS fájlok

* angular.min.js – angularJS forrásfájlja
* angular-route.min.js – angularJS route rendszer forrásfájlja
* bootstrap.min.js – bootstrap JS fájlja
* sha1.js – CryptoJS fájlja (jelszó titkosítása)

HTML fájlok

* index.html – mindennek az alapja, az oldal alváza (ebbe kerülnek be a nézetek)
* wiki.html – tudásbázis nézete
* rules.html – szabályzat nézete
* login.html – bejelentkezés panel nézete
* register.html – regisztrációs panel nézete
* news.html – hírek felvétele nézet
* main.html – kezdőlap nézete
* currencies.html – árfolyam nézete
* forum.html – fórum választás nézete
* mainforum.html – fórumlista nézete
* createforum.html – fórumposzt létrehozása nézet
* profile.html – profil nézet

## 3.5 Főbb szkriptek

A főbb szkriptekről egy igényes leírást csináltam kódrészlettel bemutatva, mivel ezek ritkán használtak vagy komplexebb felépítésűek.

### 3.5.1. currenciesCtrl.js

Az árfolyam kontrollerét a coinmarketcap.com API segítségével készítettem el, aminek az ingyenes verziója lehetővé teszi, hogy havonta 10000 kreditnyi hívást csinálhatok csak a legújabb árakról, mennyiségekről a kriptovalutákkal kapcsolatban. Egy kripto adatainak hívása egy kredit.

$scope.cryptos = [];

    $scope.priceUpdate = async function() {

        await axios.get(`${$rootScope.serverUrl}/db/get\_crypto\_data/btc`).then(res =>{

            const cryptoData = res.data.data.data.BTC

            $scope.cryptos.push({

                imageName: 'bitcoin',

                name: 'Bitcoin',

                price: Math.round(cryptoData.quote.USD.price).toFixed(2),

                marketcap:  Math.round(cryptoData.quote.USD.market\_cap).toFixed(2),

                volume:  Math.round(cryptoData.quote.USD.volume\_24h).toFixed(2),

                totalSupply: cryptoData.max\_supply

            })

        })

Amikor az árfolyam nézetre navigál a felhasználó, az összes előre megadott kriptovaluta adatait lekérdezem az API segítségével és a kiválasztott adatokat eltárolom egy tömbben, majd a tömb adatait megjelenítem egy táblázatban. Az, hogy a táblázatban melyik kép jelenjen meg az adott kripto-hoz úgy működik, hogy elmentem a kripto nevét egy „imageName” változóba, majd a képek forrásához beillesztem azt.

A Backend hívás

router.get('/get\_crypto\_data/:crypto', (req, res) => {

    const crypto = req.params.crypto

    const url = `https://pro-api.coinmarketcap.com/v1/cryptocurrency/info?symbol=${crypto}`

    const url2 = `https://pro-api.coinmarketcap.com/v1/cryptocurrency/quotes/latest?symbol=${crypto}`

    request.get({

      url: url2,

      json: true,

      headers: {

        'X-CMC\_PRO\_API\_KEY': process.env.APIKEY

      }

    }, (error, response, data) => {

      if (error) {

        return res.send({

          error: error

        });

      }

      res.send({

        data: data

      });

    });

});

A hívásba paraméterből kiveszem az adott kripto „slug” értékét és az alapján kéri le az adatokat. Az „url2” változót használva kapjuk meg az adatokat, amihez kell egy API-KEY, amit az oldalról lehet megszerezni. Ezt az API-KEY-t titkosan tárolom a .env fájlban.

### 3.5.2. logCtrl.js

A logCtrl.js fájlban történik meg a bejelentkezés és elfelejtett adatok küldése E-mail-ben. A validálás a Backend-ben zajlik le „Access token” segítségével. Lekéri adatbázisból a users adatait majd létrehozza a tokent. Ha jók a bejelentkezési adatok akkor egy egy napig érvényes token jön létre ami a sessionstoragebe kerül tárolásra viszont ha rossz akkor egy üres tokent küldd vissza a backend és nem engedi bejelentkezni a felhasználót mondván hogy rossz adatokat adott meg. Sikeres bejelentkezés után lementésre kerül a felhasználó adatai kliensoldalon, amit az access tokenből szerez meg. Ezekkel az adatokkal lehet a többi funkciót működtetni mint például a profil módosítást ami a userID alapján kerül feltöltésre vagy frissítésre. Frontend kódrészlet:

 $scope.bejelentkezes= function() {

        toastcontent.innerText = "";

        const { email, pass } = $scope.user;

        if (email == null || pass == null){

            toastcontent.innerText = "Nem adtál meg minden adatot!"

            toastBootstrap.show()

            return;

        }

        else{

            let data = {

                'email': email,

                'pass': CryptoJS.SHA1(pass).toString()

            }

            axios.post($rootScope.serverUrl + '/db/logincheck', data).then(res=>{

                console.log(res.data.data)

                if (res.data.data[0].token != ''){

                    sessionStorage.setItem('access\_token', JSON.stringify(res.data.data[0].token));

                    console.log("Van token!")

                    token = JSON.parse(sessionStorage.getItem('access\_token'));

                    axios.defaults.headers.common['Authorization'] = `Bearer ${token}`;

                    $rootScope.loggedUser = $rootScope.getLoggedUserData(token);

                    $rootScope.loggedIn = true;

                    $location.path('/profile');

                    $scope.$apply();

                }

                else{

                    console.log("Nincs token!")

                    toastcontent.innerText = "Nem jó adatokat adott meg!"

                    toastBootstrap.show()

                }

            });

        }

    }

Backend kódrészlet

router.post('/logincheck', (req, res)=>{

    let table = 'users';

    let field1 = 'email';

    let field2 = 'pass';

    let value1 = req.body.email;

    let value2 = req.body.pass;

    pool.query(`SELECT \* FROM ${table} WHERE ${field1}='${value1}' AND ${field2}='${value2}'`, (err, results)=>{

       if (results.length > 0){

           let user = {

               ID: results[0].ID,

               name: results[0].username,

               email: results[0].email,

               privilege: results[0].privilege

            }

            results[0] ={ token: jwt.sign(user, process.env.ACCESS\_TOKEN\_SECRET, {expiresIn: '1d'})};

        }

        else

        {

            results[0] = { token: '' }

        }

        if (err) return res.send({message: 'Hiba történt!'})

        res.send({message: 'Sikeres adatkérés.', data: results})

    });

});

A lementett felhasználói adatok az app.js-ben kerül mentésre $rootscope.loggedUser változóba hogy bárhonnan el lehessen érni.

//access token validálása

    if (sessionStorage.getItem('access\_token')) {

        token = JSON.parse(sessionStorage.getItem('access\_token'));

        axios.defaults.headers.common['Authorization'] = `Bearer ${token}`;

        $rootScope.loggedIn = true;

        $rootScope.loggedUser = $rootScope.getLoggedUserData(token);

    }

//bejelentkezett felhasználó adatainak lekérése

    $rootScope.getLoggedUserData = function(token) {

        var base64Url = token.split('.')[1];

        var base64 = base64Url.replace('-', '+').replace('\_', '/');

        user = JSON.parse($window.atob(base64));

       loggedUser = {

            'ID': user.ID,

            'name': user.username,

            'email': user.email,

            'privilege': user.privilege

        }

        return loggedUser;

    }

## 3.6. Mappastruktúra

A projekt az MVC modell alapján épült fel.

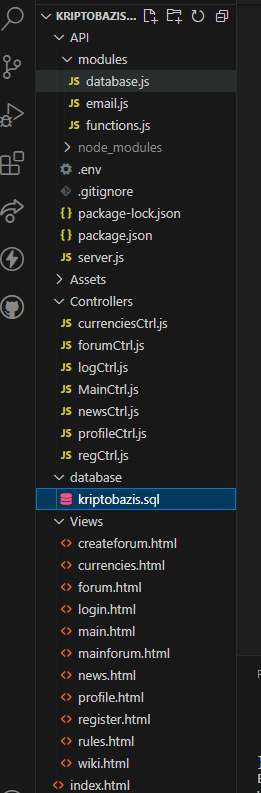
A MVC (Model-View-Controller) modell egy szoftvertervezési minta, amely segít a kód strukturálásában és szervezésében. A modell (modules) az alkalmazás logikáját, adatelérését és adatkezelését tartalmazza. A nézet (View) felelős a felhasználó felület megjelenítéséért és felhasználói interakció kezeléséért. A kontroller (Controller) pedig a logika és a felület közötti kapcsolatot biztosítja, és reagál a felhasználói interakciókra.

A MVC modell alkalmazásával könnyebbé válik a kód újrafelhasználása, karbantarthatósága és tesztelése is, mivel a különböző feleket elkülönítve kezeli a különféle funkciókat.

Az MVC modell elemein kívűl még van az Assets mappa ahol megtalálhatóak a felhasznált ikonok, könyvtárak, szövegstílusok, képek és az oldal alvázának a javascript fájlja, az app.js.

A backend az API mappában helyezkedik el két részre osztva. A modules mappában találhatóak meg a Backend endpointjai, az adatbázis létrehozása és a szükséges funkciók meg a „dependency-k”. A másik részében vannak a node\_modules, a .env, a package.json és a server.js. A .env felelős a fontosabb adatok tárolására és titkosítására, míg a server.js a Node.js szerver létrehozására és a szükséges modulok meghívására. Valamint el van rejtve a .gitignore fájl amivel be lehet állítani hogy „commitolás” esetén melyik fájlokat vagy mappákat hagyja figyelmen kívül.

Az utolsó mappa a database mappa ahol megtalálható a „kriptobazis.sql” fájl amivel a „kriptobazis” adatbázist tudjuk feltölteni.



. Mappaszerkezet VS-codeban

# Összegzés

Ebben a pontban ki fogom fejezni hogy miket tapasztaltam át a fejlesztés közben és hogy mik lettek a következményei.

## A kigondolt cél és az elkészült projekt különbségei

Mivel egy általam újonnan tanult keretrendszerben dolgoztam (angularJS) így akadtak kezdeti nehézségek amiket idővel oldottam meg. Emellett ahhoz hogy a kriptovaluták árfolyamát meg tudjam szerezni ahhoz egy másik cég API-ját kellett felhasználnom. A probléma az hogy az ingyenes API csak a legfrissebb adatokat tudja lekérni tehát akárhányszor lehívás történik az mindig a lehető legújabb. Az eredeti tervben lett volna az árfolyam nézetben egy „flowchart” vagyis grafikon ami mutatta volna az egyes valuták értékének változását évről évre.

Sajnos nem találtam megfelelő alternatívát erre a problémára úgyhogy maradtam annál a megoldásnál hogy egy táblázatban írja ki az adatokat és mindig a legfrissebeket.  
Ezen kívűl nem volt komolyabb változás az eredeti kigondoltságban és a vége között maximum a kinézet amit talán kicsit máshogy képzeltem el, de így sem aggódok miatta.

## Személyes tapasztalatok és problémák

A legtöbb gondot a keretrendszerek kombinálása okozta mivel még nem volt bennük túl sok tapasztalatom. Az oldalsó oszlop egy előre megadott minta alapján készült ami mint kiderült, sok más dologra is hatásra van. Például a „bootstrap toast” helyett modal ablakokban gondolkodtam viszont azok formázását szinte lehetetlen volt megoldani hiszen több mindenbe is beleütközött. Ezek után még az angularJS és Backend közötti kapcsolat volt még nehézkes az elején. Sokat kellett „debug”-olnom vagyis hibát keresnem mivel sokszor előfordult hogy egy két kisebb kihagyás vagy extra karakter került mondjuk az SQL lekérdezésekbe vagy csak szimplán rossz adatot próbáltam elérni. De próbáltam higgadtan kezelni a problémákat ami később arra vezetett hogy minden egyszerűbb lekérdezést vagy tárolást könnyedén megoldottam mivel tudtam hogy hol keressem a probléma forrását. Nagy segítségemre volt a böngésző fejlesztői eszköze ahol nyomon követhettem a Backend hívásokat, sessionstorage elemeit, és magát a kódot is hogy mi hogyan változik.

Úgy gondolom hogy ez a projekt rengeteget segített abban hogy megértsem az angularJS keretrendszert és pontosítsam a Backend-el kapcsolatos tudásomat is. Ha jóval mélyebben beleásom magam és sok időt rááldoztam volna, egy komolyabb végeredmény születhetett volna.

## Továbbfejlesztési lehetőségek

Számottevő lehetőség van ebben a kategóriában mint például a korábban említett grafikon megvalósítása ami egy kis pénz áldozásával meg is valósulhat. A táblázat helyett minden kriptonak lenne egy saját kiválasztható kártyája vagy oszlopa ami előhozná a grafikont ami valahogy így nézne ki.



. Coinmarketcap árfolyam grafikon

Amit még érdemes lehetne továbbfejleszteni az a jogosultságkezelés. Szimplán lenne egy nézet ahol megjelenítődik az összes felhasználó adatai és jogosultsága legfőképp. Itt egy admin felhasználó tudná beállítani hogy kinek milyen jogosultsága legyen illetve hogy az adott felhasználó nevét vagy jelszavát lehetne változtatni ha szüksége van rá.

## Tesztelés

A Tesztelés fájljai itt találhatóak:

# Források

## Források

Micah – Building your coinmarketcap API query link – How to guide

<https://www.youtube.com/watch?v=SC2q-_9GI1w&t=109s>

Ekkor: 2024.04.15.

Bootstrap 5.3 – Components

<https://getbootstrap.com/docs/5.3/components/accordion/>

Ekkor: 2024.02.25

Nodemailer

<https://www.npmjs.com/package/nodemailer>

Ekkor: 2024.04.19

Access token – Creating and viewing access tokens NPM

<https://docs.npmjs.com/creating-and-viewing-access-tokens>

Ekkor: 2024.04.20

CSS Loader keyframe

<https://www.w3schools.com/howto/howto_css_loader.asp>

Ekkor: 2024.04.16

Bootstrap Sidebar template

<https://freefrontend.com/bootstrap-sidebars/>

Ekkor: 2024.01.29

# Ábrajegyzék

(képek sora,neve, navigálható)

1. https://www.apachefriends.org/index.html [↑](#footnote-ref-1)
2. https://httpd.apache.org/ [↑](#footnote-ref-2)
3. https://www.phpmyadmin.net/ [↑](#footnote-ref-3)
4. https://code.visualstudio.com/ [↑](#footnote-ref-4)
5. https://git-scm.com/ [↑](#footnote-ref-5)
6. https://github.com/ [↑](#footnote-ref-6)
7. https://nodejs.org/en [↑](#footnote-ref-7)
8. https://getbootstrap.com/docs/5.3/getting-started/introduction/ [↑](#footnote-ref-8)
9. https://www.javascript.com/ [↑](#footnote-ref-9)
10. https://www.w3.org/Style/CSS/Overview.en.html [↑](#footnote-ref-10)
11. https://html.com/ [↑](#footnote-ref-11)
12. https://angularjs.org/ [↑](#footnote-ref-12)