# **A 2021/22-es tanév első félévének Webprogramozás II. tárgyához készített dokumentáció**

### A feladat leírása

A feladat szerint egy fiktív szervezet, cég honlapját kellett elészíteni:

A témavezető által kiadott feladat a szélerőművekről szól ezért a főoldalra egy ezzel kapcsolatos cikk anyagát jelenítettük meg. A feladatot Kovács Dániel (KD) és Tóth Zsuzsanna (TZS) készítette el. A megoldás során használt szoftverek:

* Visual Studio Code 1.62.1 (KD) (TZS)
* XAMPP v3.3.0 (KD) (TZS)
* Opera 80.0.4170.63 (KD) (TZS)
* PhpStorm 2021.2.3. (TZS)

A kiírásban a következő szempontoknak kellett megfelni:

* Bemutatni egy céget egy reszponzív honlapon
* A honlapon legyen bejelentkezési, regisztrációs lehetőség úgy, hogy
  + A „Belépés” menüpont akkor látható, ha nincs bejelentkezve a felhasználó.
  + A „Kilépés” menüpont akkor látható, ha be van jelentkezve a felhasználó.
  + A „Belépés” menüpontra kattintva feljön egy oldal, ahol lehet bejelentkezni vagy regisztrálni.
  + d) Regisztrációkor nem léptetjük be automatikusan a felhasználót.
  + e) Bármelyik oldalon jár: a rendszer fejlécen jelenítse meg a bejelentkezett felhasználót, ha be van lépve, a következő formában: Bejelentkezett: Családi\_név Utónév (Login\_név)
* Bejelentkezett felhasználóknak lehetőséget kell nyújtani hírek írására, olvasására
* Objektum orientált PHP-ban kell megvalósítani
* A feladatot MVC modellel kell megoldani
* SOAP szolgáltatás:
  + a témavezető által kiadott feladat megvalósítása, melyben különböző adatokat kell lekérdezni a SOAP serverről
  + MNB Soap szolgáltatását igénybe véve lekérdezéseket kell végrehajtani
* A projektet fel kell tölteni egy internetes tárhelyre, valamint érdemes GITHUB-ra is.

Megvalósítás:

***KD:*** A feladatot a gyakorlati órán kiadott anyagok alapján kezdtem el vizsgálni. GitHub-ra feltöltöttem a *Web-2-Megoldasok\Gyak-03-Keretrendszer MVC-vel-PHP, Adatbázis-kezelés, Munkamenetek/* mappában található projektet (<https://github.com/kovacsdaniel16/wp2php8>).

Ezután az első dolgom az volt, hogy a honlapot kiegészítettem regisztrációs lehetőséggel, ezt úgy csináltam, hogy a **controllers** mappában létrehoztam egy **regisztracio.php** fájlt (1.kép) az alábbi paraméterekkel:

A képen szöveg látható

Automatikusan generált leírás

. kép regisztracio.php

A regisztracio.php meghívja a **regisztracio\_model.php**-t majd utána a a View\_loader-t **regisztracio\_main.php**-ra irányít. A regisztracio\_model.php-ban A **getdata()** függvény meghívása végzi el a regisztrációt. Itt példányosítjuk a includes mappában található Database osztályt ami az adatbázis kapcsolatért felelős. Azután megvizsgáljuk, hogy a regisztrálandó felhasználó kitöltötte-e a **views/belepes\_main.php**.ben található form minden mezőjét, ha igen, illetve ez megfelel a formai követelményeknek, létre tudjuk hozni a felhasználót az adatbázisban. A regisztrációs sikeres vagy sikertelen mivoltáról a **regisztracio\_main.php** oldalon lévő üzenet tájékoztat bennünket. Regisztráció során, jogosultsági szintet is adunk a felhasználónak. A példában 3 felhasználói szint létezik, *admin*, *látogató* és *felhasználó*

Sikeres belépés esetén a $\_SESSION[…] globális töltjük be View által megjelenítendő információkat, valamint a felhasználói szintet megatározó számsort. Így, belépett felhasználó esetén, a felhasználó családneve, utóneve, valamint bejelentkezési neve a page\_main.php fejlécében kerül megjelenítésre. 2. kép

A képen szöveg látható

Automatikusan generált leírás

. kép page\_main.php <header> állomány

A View\_loader a felhasználói jogosultság tükrében tölti be a bizonyos szintű felhasználóknak megjeleníthető elemeket.

Kilépés során a $\_SESSION[] tömböket kiürítjük, illetve a jogosultságot meghatározó $\_SESSION-t látogató szintűre módosítjuk.

A hírek olvasása írása feladatot egy hírfolyam szerű lapon próbáltam megvalósítani. A feladat szerint csak regisztrált felhasználók tudják használni ezt a funkciót.

Mindenek előtt létrehoztam egy **hirek** SQL adatbázist, melyben az oldal működéséhez szükséges adatokat fogja tárolni (felhasználó, hír, időpont)

Először készítettem egy **hirek.php** filet controllers mappába, ami meghívja a models mappában lévő **hirek\_model.php**-t. A hirek\_model.php tartalmaz egy getData() és egy getNews() függvényt. A getData() a views/**hirek\_main.php**-ben található form-ból begyűjtött adatokat írja bele a **hirek** SQL táblájába.

A getNews() függvényben készítünk egy lekérdezést, amelyből a kinyert adatokat $\_SESSION[] tömbbe írjuk. Itt készül egy tömb a híreknek, egy a felhasználónak, és egy a létrehozás dátumának.

A képen szöveg látható

Automatikusan generált leírás

. kép Adatbázis

A kiíratást HTML táblázat formátumban oldottam meg, egy *for* ciklus és $\_SESSION[] tömbök segítségével, mely szintén a hirek\_main.php-ben van.

A képen szöveg látható

Automatikusan generált leírás

. kép A hírek táblázatos kiíratása

A következő feladatrészhez a kapott txt-ket felhasználva létrehoztam egy **szeleromuvek** nevű adatbázist, három táblával, a megadott szempontok alapján.

SQL (TZS):

Az SQL adatbázist és táblákat többféleképpen is létre lehet hozni és különböző módosításokat lehet rajtuk végrehajtani. A feladatok megoldása során mi is egynél több módszert alkalmaztunk. Ehhez elsőnek is az **XAMPP** szoftverre volt szükségünk, melyet korábbi tantárgyak által már letöltöttünk, használtunk. A SQL adatbázishoz elegendő az Apache és a MySQL modulokat elindítani, a lépés sikerességét a feliratok hátterének zöld színre való átváltozása mutatja, ezt követően a myphpadmint localhoston keresztül megnyitjuk (<http://localhost/phpmyadmin/>). Létrehoztunk egy-egy új adatbázist, mely az én esetemben az **arfolyam\_lekerdezes.sql** nevet kapta, illesztésnek pedig meghagytam, amit a rendszer felajánlott **„utf8mb4\_general\_ci”.** Sok esetben nem ezen a grafikus felületen szokták létrehozni az adatbázisokat és a hozzájuk tartozó táblázatokat, hanem a php-n keresztül kódolással, de a grafikus felületen, az SQL fül segítségével mi is meg tudjuk valósítani kódolással az elképzeléseinket.

* Adatbázis kreálása:
  + 1. opció (kódolás nélkül): Az „Adatbázis” fülre kattintva, vagy a bal oldalt található oszlopos (adatbázis táblákon) panelon belül az „Új” ikonra kattintva egy tetszőleges név megadásával és a „Létrehozás” gomb megnyomásával automatikusan megjelenik az új adatbázis elem.
  + 2. opció (kódolással): Az „Adatbázis” mellett található „SQL”-re kattintva kezdhetjük el begépelni a kódunkat. A „**CREATE DATABASE**” utasítás segítségével tudunk létrehozni egy adatbázist, de a létrehozásával nem automatikusan következik, hogy ki is választottuk az újonnan létrehozott elemet, így szükséges még továbbá a „**USE**” kulcsszó használata is, a parancs végeztével pontosvesszőt teszünk. Törölni pedig a „**DROP DATABASE**” kifejezéssel lehet.
* Adattáblák létrehozása:
  + 1. opció (kódolás nélkül): A kiválasztott adatbázisra kattintva azon belül elhelyezni kívánt táblázatokat tudunk realizálni ezen a felületen, eleinte a nevét és az oszlopainak a számát tudjuk megadni, majd „Indítás”-t követően betöltődik további beállítási lehetőség is, mint például meg tudjuk adni az oszlopok megnevezését, a típusát (INT, VARCHAR (szöveges változó), DATE, TEXT, stb.), a hosszértékét (maximális karakterszám), hogy automatikusan növekedő legyen (A\_I) vagy sem, de a legfontosabb beállítás az Indexen belül megadható „Primary” tulajdonság, vagyis, hogy elsődleges kulccsá alakítja az elemet, mivel az adat egyértelműen meghatározza a táblázatunknak az adott oszlopát, mezejét. A törlését pedig az „Eldobás” gombbal tehetjük meg.
  + 2. opció (kódolással): Az adatbázisoknál használt kifejezéseket itt is lehet használni, csak a „datebase” szó helyett a „table”-t használtam helyette. Az **„INSERT INTO”-**val és a **„VALUES”-**zal pedig sorokat, adatokat adtam hozzá. Három táblázatot hoztam létre (**calendar\_date.sql, currency\_list.sql** és a **currency\_rate.sql**-t).

SOAP (TZS):

**A feladat elkészítése során az órai példákon, mint például mobiltelefonoson kívül az alábbi oldalakon szereplő információkat használtam fel:**

<https://www.mnb.hu/sajtoszoba/sajtokozlemenyek/2015-evi-sajtokozlemenyek/tajekoztatas-az-arfolyam-webservice-mukodeserol>;

<https://hup.hu/node/142647>;

<https://prog.hu/tudastar/190352/mnb-arfolyamok-lekerdezese>;

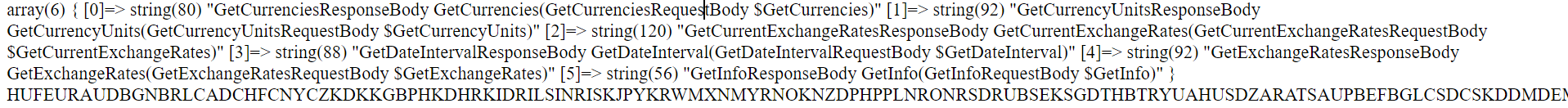
<https://www.mnb.hu/letoltes/aktualis-es-a-regebbi-arfolyamok-webszolgaltatasanak-dokumentacioja-1.pdf>;

<http://eqinfo.ucsd.edu/~rnewman/howtos/php/soap_noaa_weather/>;

Egy könyvelői cégnél, mint könyvelő dolgozok és napi rendszerességgel használok MNB vagy más központi bank hivatalos árfolyamát. Az olyan nagy programok, mint a SAP automatikusan össze van kapcsolva a rendszerrel és a feltételek beállítását követően a szükséges árfolyamot megjeleníti a könyvelési folyamatnál, a vevőszámla-kiállításnál. Emlékszek rá, hogy korábban le kellett ezt az információt a weboldalról szedni és csv formátumban, manuálisan feltölteni az adatokat a helyes kartonba. Sokszor árfolyam issue-ba is ütköztünk a hiányos feltöltések miatt, mert az árfolyamot minden munkanapokon egyszer és azt is általában dél tájékán frissítik, a könyvelői szoftver által kalkulált átlagárfolyam, így hibás lett, a nem megfelelően kiállított bizonylatokat sztornózni kellett.

A SOAP-os feladatrésznél eleinte a Magyar Nemzeti Bank (MNB) honlapján található **„Tájékoztatás az árfolyam Webservice működéséről”** szóló értesítőből indultam ki, próbáltam visszafelé megfejteni, a feladatot, hogy megérthessem, hogyan lehetséges rá megfelelően működő kódot írni. Felhasználtam a wsdl-t, mely az mnb adatszolgáltatásának leírását egy fájlként jeleníti meg, ez a link elérhető volt az mnb honlapján és a tantárgyhoz feltöltött tananyagoknál is. Az első kísérletem az összes elérhető függvény lekérdezése volt, hogy láthassam, hogy számunkra összesen 6 lehetséges függvény érhető el, melyek az alábbiak:

* GetCurrencies – Pénznemek lekérdezése
* GetCurrencyUnits – Pénzegységek lekérdezése pénznemenként
* GetCurrentExchangeRates – Aktuális árfolyamok (egy adott nap)
* GetDateInterval - Időintervallum
* GetExchangeRates – Árfolyamok (egy meghatározott időszakra vonatkozó)
* GetInfo



5. kép Az összes függvény és a pénznemek lekérdezése.

Következőként elkészítettem a pénznemek lekérdezését, mely sikeresnek bizonyult, mivel visszakaptam az ISO kódokat, ehhez meg kellett adnom, hogy milyen lehetséges értékeket vehet fel a pénznem és az elkészült lekérdezést lefuttattam. A **„for ciklus”-**t használtam, melynél az első kifejezés a ciklusváltozónak (**„i”-**nek) ad kiinduló értéket, ezt követi egy feltétel, majd végül a harmadik dönti el, hogy növeli vagy csökkenti a ciklusváltozót, mely a mi esetünkben növekedési irányt jelent. A MNB bank összesen három pénznemnél (**IDR – indonéz rúpia, JPY – japán jen és a KRW – dél-koreai von**) ad meg 100 egységet, a többinél 1 egységben vannak feltűntetve az értékek, melyekre majdnem napi rendszerességgel ad meg átváltási értéket. A legelterjedtebb közülük a japán jen, így ezt a pénznemet a „**strpos**”-szal határoztam meg, mely megkeresi a karakterláncban a részkarakterláncnak az első előfordulásának a helyét, mely ebben az esetben JPY.

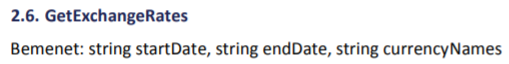
A képen szöveg, képernyőkép, képernyő látható

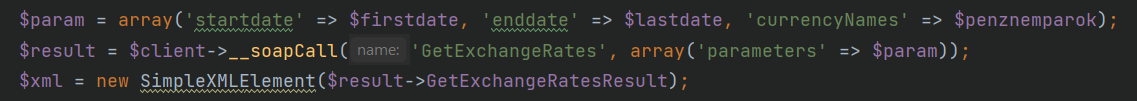
Automatikusan generált leírás

6. kép: A lekérdezések kódolása.

A phpben több feltételes utasítás is megtalálható, az **„if…else”** is ezek közé tartozik, és egy olyan utasítás, mely, ha egy feltétel igaz, akkor az első kódot végrehajtja, melyet meghatároznak számára, és ha még se tesz eleget a feltételnek, akkor az ellenkező, a másik kód fog végrehajtódni. A mi esetünkben ezt az utasítást, a melyik pénznemről szeretnénk az átváltást határoztuk meg. Nagyon fontos, hogy milyen rendszerben dolgozunk, például én jelen pillanatban a munkahelyemen US szerinti szabványok szerint dolgozok, míg a magyaroknál a „,” a tizedes vesszőnek a helyét jelöli, addig Amerikában az ezres helyiértéket jelölik vele, és a „.”-tal jelölik a tizedest, ezért **„str\_replace”-**szel kicseréltem az összes ,-t .-ra.

Ezt követően az időt és az időintervallumot vizsgáltam meg, elsőnek „**date”** függvénnyel próbáltam meghatározni a mai dátumot -> **„echo date(’Y-m-d H:i:s’);”**, majd ezt a képletet tovább gondolva meghatároztam, hogy mi egy hónappal korábbi időpont („**echo date(’Y-m-d H:i:s’, time() – 60\*60\*24\*30”**) , így képezve két dátumot, mely egy időintervallumot határoznak meg. Sajnos ez a megoldás nem vált be, próbálkoztam sql adattáblát is létrehozni mellé, lásd **calendar\_date.sql**, míg nem a MNB-s pdf alapján meg határoztam a változókat (**startdate** és **enddate**). **„Isset”** függvénnyel meghatároztuk, hogy a változó beállítottnak tekinthető-e, ami azt jelenti, hogy deklarálva van-e esetleg, és különbözik-e, mint „**null”.** A paraméterek ebben az esetben lehet mondjuk „**var”,** mely az ellenőrizendő változó, míg a „**vars”** további változókat jelenti. Megvizsgáltuk a két pénznemet és a két dátumot, a kódolást, és hogy mely információ alapján neveztük el az alábbi képeken látható. Ebben az esetben nem **„if … else”**-t használtam, hanem **„if”** függvény után betettem egy blokkot, a **„try”** kifejezéssel, mely esetén, ha a kivétel miatt aktiválódik az utasítás, akkor a kivétel törlésre kerül, de ha nem történik meg a folyamat, akkor a kód tovább folytatódik, fut. A weboldalon a **„label”**-leltudtam elérni, hogy ki tudjam választani a dátumokat és a „**button”**-nak pedig azt, hogy elindíthatom a lekérdezést.





7.-8. kép: Paraméterek elemeinek meghatározása az MNB pdf fájlja alapján

A változókat meghatároztam és megjelenítettem, hogy milyen adatokat szeretnék lehívni a soap segítségével. Az árfolyamokat megfelelően betöltötte cellákba és ki is lehet őket jelölni/választani. A websiteot, az arfolyam\_main-t php-ban készítettem el, **<!DOCTYPE>** deklarációval jelöltem, hogy a böngésző számára egyértelmű legyen, hogy html dokumentumtípusa van a fájlomnak, majd létrehoztam az osztályokat és a feliratokat, majd létrehoztam egy **formot**, melyen belül **selecttel**, **labellel**, és a **buttonnel** megalkottam a lekérdezéshez szükséges felületemet, majd a hozzájuk szükséges adatforrást a soappal biztosítottam, a lekérdezés eredményét táblázatos (**table**) formában szeretettem volna elérni. Míg a **„tr”** tag a táblázat sorait, a **„th”** a táblázat fejlécét, addig a **„td”** táblázat celláit jelöli.

A képen szöveg látható

Automatikusan generált leírás

9.kép: táblázat html-es kódolása az arfolyam\_main.phpn belül

A képen szöveg látható

Automatikusan generált leírás

10.kép: végeredmény

Weboldal publikálása (TZS):

Az elkészült weboldalunkat publikálni, vagyis mindenki számára elérhetővé tenni a GitHubon keresztül tettük meg. Azért esett az említett oldalra a választásunk, mert rendelkezik egy olyan szolgáltatással, a **GitHub Pagesnek** köszönhetően, mely az általunk korábban feltöltött fájlokat a publikálni is tudja.

A következő lépéseket kellett tennünk a 8. feladat sikeres elvégzéséhez:

* Elsőként szükségünk volt egy GitHub-os felhasználó névre, mely a mi esetünkben két lehetséges opció is volt, a **kovacsdaniel16** és a **balto1990**, majd ezt követően létre kellett hoznunk egy új adattárat (’New Repository’, mely a fejléc jobb felső sarkánál található ’+’ jelnél érhető el). Átirányított minket a ’Create a new repository’ felületre, ahol a ’Repository name\*’ alatt megadtuk az egyikünk felhasználó nevét + ’.githun.io’ (**kovacsdaniel16.github.io**) kifejezést, és kiválasztottuk a ’Public’-ot, hogy mindenki számára nyilvánossá tegyük. A célunk eléréséhez nem volt szükséges további információ megadása, így a zöld gombbal, a ’Create repository’-val elindítottuk a folyamatot, és létrejött egy üres repo. Kimásoltuk a legenerált elérhetőségi útvonalat, mely ebben az esetben a [https://github.com/ kovacsdaniel16/kovacsdaniel16.github.io.git](https://github.com/%20kovacsdaniel16/kovacsdaniel16.github.io.git) volt.
* A következő lépésként a **GitHub Desktop** alkalmazást megnyitottuk és a fejlécben található ’Repository’-ra rákattintottunk, majd ’Repository settings…’. A ’Remote’ lehetőségnél felülírjuk a szoftver által automatikusan adott elsődleges távoli adattárat a mentett elérési útvonallal. Az eddig elvégzett lépések miatt szükséges egy frissítés, melyet a ’Fetch origin’-ra kattintva tettünk meg, a folyamat lefutása után a feliratozása a gombnak megváltozott, ’Publish branch’-ra, ezt követően még egyszer rá kellett kattintanunk, hogy a kívánt folyamat végére érjünk. A továbbiakban nem volt szükségünk a szoftverre, így a lefutása után bezártuk.
* A <https://githun.com> oldalra visszatérve egy frissítést kellett végeznünk a szoftverben végzett változtatások miatt, ennek köszönhetően meg is jelentek a fájlok az általunk kreált repositoryban. Leellenőriztük, hogy minden verzió feltöltésre került-e a commitsra kattintva, majd a linkünket a címsorba beillesztettük és meg is jelent a weboldalunk a hálón.