|  |  |
| --- | --- |
|  | Soproni Egyetem  Simonyi Károly Műszaki, Faanyagtudományi és Művészeti Kar  Informatikai és Gazdasági Intézet |

|  |  |
| --- | --- |
|  |  |

**Villanyszerelők tudástára weboldal fejlesztése**

**Görcsös Dániel**

BSc Önálló laboratórium 2.

Konzulens: Szabó Tamás

2020. december 11.

|  |  |  |
| --- | --- | --- |
|  | *Soproni Egyetem* Simonyi Károly Műszaki, Faanyagtudományi és Művészeti Kar *H-9401 Sopron, Bajcsy-Zs. u. 4. Pf.: 132.*  *Tel: +36 (99) 518-101 Fax: +36 (99) 518-259* |  |

**NYILATKOZAT**

Alulírott **Görcsös Dániel** (neptun kód: **T4HCL8)** jelen nyilatkozat aláírásával kijelentem, hogy a **Villanyszerelők tudástára weboldal fejlesztése** című

**önálló laboratórium dolgozat**

(a továbbiakban: dolgozat) **önálló munkám**, a dolgozat készítése során betartottam a szerzői jogról szóló 1999. évi LXXVI. tv. szabályait, különösen a hivatkozások és idézések tekintetében.

*Hivatkozások és idézések szabályai:*

*Az 1999. évi LXXVI. tv. a szerzői jogról 34. § (1) és 36. § (1) első két mondata.)*

Kijelentem továbbá, hogy a dolgozat készítése során az önálló munka kitétel tekintetében a konzulenst illetve a feladatot kiadó oktatót **nem tévesztettem meg.**

Jelen nyilatkozat aláírásával tudomásul veszem, hogy amennyiben bizonyítható, hogy a dolgozatot nem magam készítettem, vagy a dolgozattal kapcsolatban szerzői jogsértés ténye merül fel, a Soproni Egyetem megtagadja a dolgozat befogadását és ellenem fegyelmi eljárást indíthat.

A dolgozat befogadásának megtagadása és a fegyelmi eljárás indítása nem érinti a szerzői jogsértés miatti egyéb (polgári jogi, szabálysértési jogi, büntetőjogi) jogkövetkezményeket.

Sopron, 2020. december 11.

…………….……………………………..

hallgató

**Tartalomjegyzék**

[1. Bevezetés 4](#_Toc57110596)

[2. Alkalmazott technológiák és eszközök 4](#_Toc57110597)

[2.1 HTML 4](#_Toc57110598)

[2.2. CSS 4](#_Toc57110599)

[2.3 JavaScript 5](#_Toc57110600)

[2.3.1 JavaScript példa függvény 6](#_Toc57110601)

[2.4 MySQL 7](#_Toc57110602)

[2.5 PHP 7](#_Toc57110603)

[2.6 Mathjax 7](#_Toc57110604)

[2.7 Felhasznált erőforrások 9](#_Toc57110605)

[3. Elkészült weboldal bemutatása 10](#_Toc57110606)

[3.1 Adatbázis kialakítása 10](#_Toc57110607)

[3.1.1. Táblák létrehozása 10](#_Toc57110608)

[3.1.2 Táblák feltöltése adatokkal 10](#_Toc57110609)

[3.1.3 Adatbázisban szereplő táblák részletezése 11](#_Toc57110610)

[3.1.4 Adatbázis felépítése 12](#_Toc57110611)

[3.1.5 Táblák lekérdezése 13](#_Toc57110612)

[3.2 Weboldal ismertetése 13](#_Toc57110613)

[3.2.1 Navigációs sáv 13](#_Toc57110614)

[3.2.2 Kezdőlap 14](#_Toc57110615)

[3.2.3 Menüpontok 14](#_Toc57110616)

[4. Összegzés 21](#_Toc57110617)

[5. Irodalomjegyzék 22](#_Toc57110618)

# 1. Bevezetés

Önálló labor 2. keretein belül egy weboldalt készítettem el. A motiváció az volt, hogy az egyetemi tanulmányaim mellett elkezdtem egy villanyszerelő OKJ képzést, és szomorúan vettem észre, hogy az interneten nincsen egy olyan weboldal sem, ahol megtalálná az ember a villanyszereléssel kapcsolatos fogalmakat, számolásokat. Így egy olyan weboldalt fejlesztettem, ahol megtalálhatóak a fogalmak, különféle törvények, mértékegység átváltó és villanyszereléssel kapcsolatos számolások.

# 2. Alkalmazott technológiák és eszközök

## 2.1 HTML

A HTML egy olyan nyelv, melyet weboldalak elkészítésére használhatunk. Minden weboldal mögött HTML található. Fontos tudni, hogy a HTML nem egy programozási nyelv, hanem egy jelölőnyelv, arra szolgál, hogy meghatározza egy weboldal megjelenését. A jelölőnyelv kifejezés arra utal tehát, hogy miként jelenik meg a felhasználó számára egy dokumentum, illetve weboldal.

A HTML egy szimpla szöveges fájl, egy szövegszerkesztőben megírható egy weboldal mögött álló kód. Vannak azonban HTML-kód készítésére célalkalmazások, úgynevezett HTML-szerkesztők. Ezek használatával rendszerezhetőbb, jobban átlátható lesz a kód. Ezek az alkalmazások egyből felismerik, ha egy új címkét nyit az ember, és ezeket automatikusan zárják is, hogy a későbbiekben ne okozzon problémát.

## 2.2. CSS

A CSS az angol „cascading style sheets” kifejezés rövidítése, ami magyarul „egymásba ágyazott stíluslapokat” jelent. A hangsúly a „stíluson” van – míg a HTML a weblap szerkezetét határozza meg (főcímek, bekezdések, stb.), és lehetővé teszi, hogy különböző elemeket (képek, videók) ágyazunk webes dokumentumokba, addig a CSS a weblap vizuális stílusáért felel – az oldal elrendezéséért, a színekért, a betűkészletekért, azok méretéért, és így tovább.

A webes böngészők megvizsgálják a dokumentum (weblap) CSS kódját, és ez alapján jelenítik meg a HTML elemeket.

A CSS kódot – ami a HTML-hez hasonlóan egyszerű szöveges formában van jelen a dokumentumokban – többféleképpen is lehet alkalmazni.

**External CSS**

Az external stíluslapokat .css fájlok formájában kell elmenteni. Az így elmentett stílust egy egész webhely összes oldalára alkalmazni lehet anélkül, hogy a CSS kódot minden <head> szakaszába be kellene írni. Ez sokkal átláthatóbbá teszi az oldalak forráskódját, ráadásul elegendő csak egyetlen fájlt módosítani, és a változások az összes oldalon láthatóvá válnak, amelyek ugyan azt a CSS fájlt használják.

External CSS fájl használata esetén a fájlra a weboldal <head> szakaszában kell hivatkozni:

<link rel=”stylesheet” type=”text/css” href=”../style/vill.css”>

Főként ezt az opciót használtam. Előnye, hogy egyszerű a kezelése, viszont minden oldal betölti akkor is, ha csak a fájl töredékét használjuk az oldal formázására.

**Internal CSS**

Az internal CSS alkalmazására akkor lehet szükség, amikor egyetlen oldal stílusát szeretnénk megváltoztatni.

Internal CSS használatakor a CSS kód közvetlenül a forráskódban, a <head> szakaszban helyezkedik el:

<head>

<style>

table { border: 1px #ccc solid; border-collapse:collapse; padding: 5px; color:#fff; margin-left:auto; margin-right:auto;}

table th, table td { border: 1px #ccc solid; border-collapse:collapse; padding: 20px; font-weight:normal; text-align: center; color:#fff;}

table th { text-align:center; font-weight: bold; color:#fff;}

</style>

</head>

Ezen opcióval könnyedén tudtam formázni az adatbázisból betöltött adataimat.

Az external és az internal CSS abban különbözik egymástól, hogy az external egy fájlból olvassa be magát a formázást , de lényegében mindkettő megoldás ugyanazt a funkciót látja el.

**Inline CSS**

Inline CSS-re akkor van szükség, amikor egy oldal egy-egy elemének szeretnénk csupán egyedi stílust adni

Az inline CSS szintén közvetlenül forráskódban szerepel, de nem a <head> szakaszban kell megadni, hanem annak az elemnek a címkéjében, amit meg szeretnénk változtatni.

<p style=”font-size:20px”>Cím<p>

## 2.3 JavaScript

A JavaScript egy szkriptnyelv, mellyel rendkívül sok dolog kivitelezhető. A JavaScriptet első sorban arra használják, hogy gazdagabb, felhasználóbarát élményeket teremtsenek vele az internetet böngészők számára, például dinamikusan frissülő weboldalakat, intuitív felhasználói felületeket, menüket, párbeszédpaneleket, 2D-s és 3D-s grafikákat, interaktív térképeket, videólejátszókat, és számos egyéb elemet, illetve funkciót.. Lehetővé teszi, hogy a weboldalak reagáljanak a felhasználói cselekvésekre anélkül, hogy egy új oldal töltődne be. A legtöbb asztali böngésző támogatja, így a Chrome, a Firefox, a Safari, az Edge, az Opera, stb., illetve a a mobilböngészők nagy része is.

A JavaScript egy komplett programnyelv-fordító, amely közvetlenül a webböngésző szoftverekben működik. A JavaScripttel mindent meg lehet csinálni, mint a Java-val, vagy bármilyen hasonló, általános programozási nyelvvel, például:

* Deklarálhatunk vele változókat
* Értékeket tárolhatunk és hívhatunk meg
* Függvényeket definiálhatunk és használhatunk
* Saját osztályokat hozhatunk létre
* Külső modulokat használhatunk

Egy HTML oldal is tartalmazhat JavaScript nyelven megírt kódrészletet a <script> és a </script> tag-ek között. JavaScriptet több helyen is alkalmaztam a weboldalam elkészítésekor.

### 2.3.1 JavaScript példa függvény

**Forráskód :**

<button onclick="topFunction()" id="myBtn">Lap tetejére</button>

<script>

var mybutton = document.getElementById("myBtn");

window.onscroll = function() {scrollFunction()};

function scrollFunction() {

if (document.body.scrollTop > 20 || document.documentElement.scrollTop > 20) {

mybutton.style.display = "block";

} else {

mybutton.style.display = "none";

}

}

function topFunction() {

document.body.scrollTop = 0;

document.documentElement.scrollTop = 0;

}

**Magyarázat :**

Az elkészített funkció először megnézi, hogy a lap tetejétől milyen messze vagyunk. Ha ez az érték nagyobb vagy egyenlő 20 pixelnél, akkor a jobb alsó sarokban megjelenik egy gomb, melynek a ’Lap tetejére’ címet adtam. Rákattintva 0 pixelre ugrik a weboldalunk, ami a lap tetejét jelenti. Ezzel a megoldással nem kell visszatekernünk a lap tetejéhez, elég csak megnyomni ezt a gombot.

## 2.4 MySQL

A MySQL egy nyílt forráskódú relációs adatbázis-kezelő rendszer. A MySQL központi eleme a nyílt forráskódú webes alkalmazás szoftvercsomagnak, a LAMP-nak, és más "AMP" csomagnak (WAMP, XAMPP).

Az adatbázis-kezelő szoftver széleskörű képességeivel vált népszerűvé. Ezek közül néhány fontosabb:

* Tárolt eljárások
* Adatbázis triggerek
* "View" adatbázisok (nézetek)
* Keresztplatformos elérhetőség
* Valódi VARCHAR támogatás
* INFORMATION\_SCHEMA támogatás
* "Strict" (szigorú) mód
* Többféle tárolómotor támogatás (pl.: MyISAM, InnoDB)

Azért a MySQL-t választottam, mint adatbázis szervert, mert ingyenes elérhető és nagyon jól skálázható, weboldalam elkészítéséhez pont ideális.

## 2.5 PHP

A PHP egy általános szerveroldali szkriptnyelv dinamikus weblapok készítésére. Az első szkriptnyelvek egyike, amely külső fájl használata helyett HTML oldalba ágyazható. A kódot a webszerver PHP feldolgozómodulja értelmezi, ezzel dinamikus weboldalakat hozva létre. A hagyományos HTML lapokkal ellentétben a kiszolgáló a PHP-kódot nem küldi el az ügyfélnek, hanem a kiszolgáló oldalán a PHP-értelmező motor dolgozza fel azt. A programokban lévő HTML elemek érintetlenül maradnak, de a PHP kódok lefutnak. PHP kódjainkhoz a "<?php" nyitó és a "?>" záró karaktersorokat használjuk. Weboldalam elkészítéséhez főként adatbázis-lekérdezésekhez használtam.

## 2.6 Mathjax

A MathJax egy nyílt forráskódú JavaScript-megjelenítő motor a LaTeX, MathML és AsciiMath jelöléshez, amely minden modern böngészőben működik. Nem igényel beállítást a felhasználó részéről, így az oldal szerzője matematikát tartalmazó webes dokumentumokat írhat.

Használatához el kell helyeznünk a HTML oldalunkon a következő JavaScript kódrészletet.

<script src="https://polyfill.io/v3/polyfill.min.js?features=es6"></script>

<script id="MathJax-script" async src="https://cdn.jsdelivr.net/npm/mathjax@3/es5/tex-mml-chtml.js"></script>

Ez betölti a MathJax legújabb 3.x.x verzióját az elosztott szerverről, konfigurálja a matematika felismerésére mind a TeX, mind a MathML jelöléseket, és megkéri, hogy a matematika megjelenítéséhez állítsa elő a kimenetet HTML-el és CSS-el.

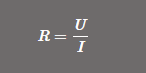
Az alapértelmezett matematikai elválasztók a „$$ ... $$” és a „\[... \] „.

Képletek elkészítésekor így néz ki a bemenet :

<p>

\[I = {Q \over t}\]

</p>

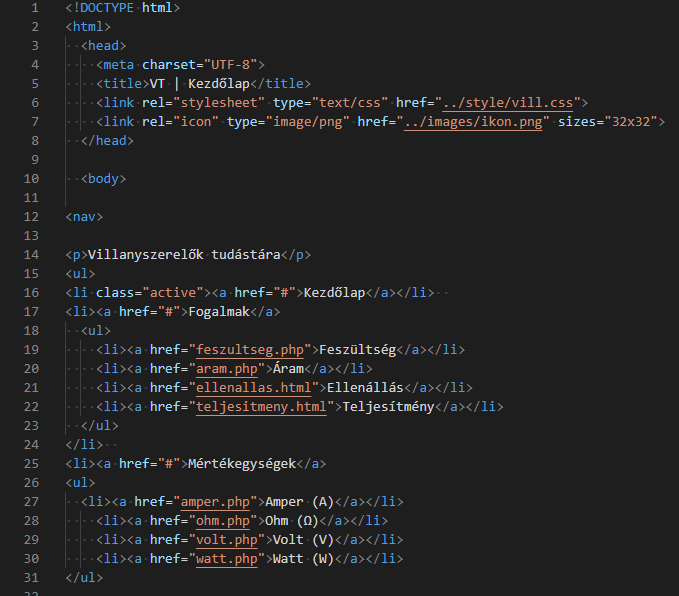
Böngészőben megnyitva a weboldalunkon pedig így néz ki a képlet:

1. ábra: Képlet a weboldalon

MathJax segítségével könnyedén tudunk matematikai képleteket megjeleníteni, így nincs szükség külön képek beszúrására, melynek nagy előnye, hogy a weboldalunk nem igényel további hálózati kéréseket, ezáltal gyorsabb lesz.

Azért ezeket a programozási nyelveket, eszközöket használtam, mert ezekben már volt egy kis tapasztalatom, nem kellett az elejétől megtanulnom őket.

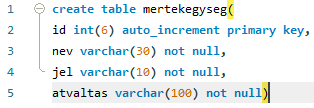
## 2.7 Felhasznált erőforrások

A HTML kódok elkészítéséhez a Microsoft Visual Studio Code HTML-szerkesztőt használtam.

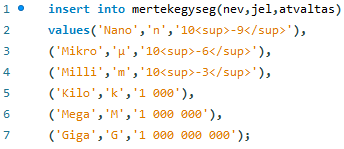
2. ábra: Kódrészlet a Microsoft VS Code szerkesztőben

Adatbázis grafikus megjelenítőjeként a MySQL Workbenchet használtam, számomra ez volt a legmegfelelőbb, mert használata nagyon egyszerű, gyorsan tanulható.

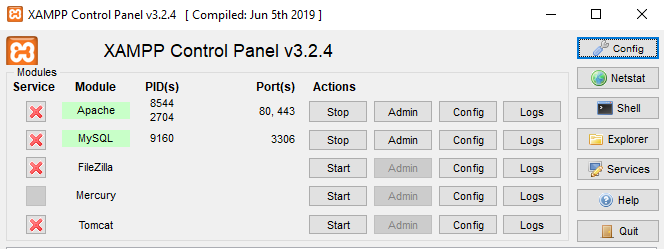
Itt látható a szerkesztőben, hogyan is néz ki a táblák létrehozása, illetve azok feltöltése adatokkal.



3. ábra: Tábla létrehozása



4. ábra: Tábla feltöltése adatokkal

Webszerverként az Apachot használtam. XAMPP segítségével könnyedén konfigurálni tudtam, mind a webszervert, mind az adatbázis szervert. Az XAMPP egy szabad és nyílt forrású platformfüggetlen webszerver-szoftvercsomag. A szoftvercsomag egy integrált rendszert alkot, amely webes alkalmazások készítését, tesztelését és futtatását célozza, és ehhez egy csomagban minden szükséges összetevőt tartalmaz

5. ábra: XAMPP felhasználói felülete

# 3. Elkészült weboldal bemutatása

## 3.1 Adatbázis kialakítása

Weboldalam összesen hat táblából tölti be az adatokat. Azért döntöttem emellett a megoldás mellet, mert így ha esetleg a későbbiekben bővíteni szeretnénk például a képleteinket, akkor nem kell belenyúlnunk a HTML kódba, elég csak egy új sort beilleszteni az adatbázis táblánkba.

Adatbázisként a MySQL-t használtam.

### 3.1.1. Táblák létrehozása

CREATE TABLE paranccsal hoztam létre a táblákat. Példának a ’kepletek’ tábla felépítését fogom bemutatni.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | mennyiseg | jeloles | osszefugges | szarmaztatas | mertekegyseg |
| int | varchar | varchar | varchar | varchar | varchar |

### 3.1.2 Táblák feltöltése adatokkal

INSERT INTO paranccsal tudunk adatokat rendelni egy táblához. Példaként a ’mertekegysegek’ táblán fogom megmutatni a műveletet.

INSERT INTO mertekegyseg (nev,jel,atvaltas)

VALUES(’Nano’,’n’,’10<sup>-9</sup>’),

(’Mikro’,’ μ’ ’10<sup>-6</sup>’),

(’Milli’,’m’,’10<sup>-3</sup>’),

(’Kilo’,’k’,’1 000’),

(’Mega’,’m’,’1 000 000’),

(’Giga’, ’G’, ’1 000 000 000’);

Azért választottam ezt a megoldást, mivel ilyenkor nem kell minden egyes mértékegység HTML kódjába belenyúlnom, elég csak egy PHP kóddal meghívni a táblát és hozzáírni a mértékegység neveit. Látható a <sup></sup> tag itt az adatbázis táblámban, ez annyit csinál, hogy egyből felsőindexbe rakja az adott számot, így azzal sem kell már foglalkoznom a HTML kódban.

### 3.1.3 Adatbázisban szereplő táblák részletezése

Az ’aramhatas’ tábla:

* id: A tábla elsődleges kulcsa
* valtakozoaram: Váltakozó áram értékei
* egyenaram: Egyenáram értékei
* hatas: Áramerősség hatása az emberre
* megjegyzes: További megjegyzések az áramütéssel kapcsolatban

A ’feszultseg’ tábla:

* id: A tábla elsődleges kulcsa
* orszag: Ország neve
* csatlakozo: Milyen csatlakozó és aljzat található meg az országban
* feszultseg: Mekkora a feszültség az adott országban
* frekvencia: Mekkora a frekvencia az adott országban

Az ’egyenfeszultseg” tábla:

* id: A tábla elsődleges kulcsa
* feszultseg: Az egyenfeszültség mértéke
* leiras: Példák és azok részletezése

A ’valtakozofeszultseg’ tábla:

* Az ’egyenfeszultseg” tábla:
* id: A tábla elsődleges kulcsa
* feszultseg: A váltakozófeszültség mértéke
* leiras: Példák és azok részletezése

A ’kepletek’ tábla:

* id: A tábla elsődleges kulcsa
* mennyiseg: Az adott képlet neve
* jeloles: Az adott képlet jele
* osszefugges: Képlet számítása
* szarmaztatas: Képlet származtatása más mértékegységekből
* mertekegyseg: Az adott képlet mértékegysége

A ’mertekegyseg’ tábla:

* id: A tábla elsődleges kulcsa
* nev: Mértékegység megnevezése
* jel: Mértékegység jelölése
* atvaltas: Mértékegységek közötti átváltás mértéke

### 3.1.4 Adatbázis felépítése

6. ábra: Adatbázis ER diagrammja

### 3.1.5 Táblák lekérdezése

A táblák lekérdezését PHP kód segítségével valósítottam meg. Egy SELECT utasítással kellett lekérdeznem a táblát.

Elsőként létre kellett hozni a csatlakozást az adatbázisommal, utána írtam meg a SELECT utasítást. Itt a PHP kódon belül valósítottam meg, hogy egy táblázatba töltse be a különböző adatokat. A táblázat formázását a <style></style> tag közé kellett beírnom a <head> részben.

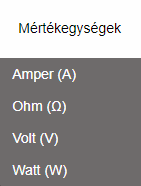
Ha létrejött a kapcsolat az adatbázissal, akkor megadtam, hogy melyik tábla adatait nézze a kód, majd egy if utasítással meg kellett mondanom neki, hogy ha nem üres a tábla akkor egy táblázatba töltse be az adatokat. Meg kellett adnom a táblázat fejlécét, majd egy while ciklussal addig töltötte be az adatokat, az adott oszlopba, míg az adatbázis táblában vannak adatok. Ha sikeresen betöltötte az adatokat a táblából zárja a kapcsolatot az adatbázissal. Hiba esetén „Connection failed „ szöveget ír ki.

A táblázat formázását amelybe az adatbázisból betöltött adatok kerülnek a következőképpen alakítottam:

* 1 pixel nagyságú fehér keret mindenhol
* A szövegek középre igazítva
* Szöveg fehér színű
* Fejlécben lévő szöveg félkövér

## 3.2 Weboldal ismertetése

### 3.2.1 Navigációs sáv

Minden egyes oldalamon megtalálható egy navigációs sáv, amely segítségével könnyedén tudunk mozogni a különböző menüpontok között. Úgy programoztam, hogy amelyik fő menüponton belül több almenüpont is található, akkor azok egy lenyíló listában jelenjenek meg.

8. ábra: Lenyíló lista



7. ábra: Navigációs sáv

### 3.2.2 Kezdőlap

A weboldal megnyitásakor ez a menüpont fogadja a felhasználót. Itt igazából csak az oldal neve a navigációs sáv található. Későbbiekben bővíthető, akár oktatóvideókkal is. Felépítése ugyanaz mint a többi menüpontnak, egyet kivéve, hogy a háttér itt nem egyszínű hanem egy kép.

9. ábra: Weboldal kezdőlapja

A navigációs sáv programozásával kellett kezdenem, amit korábban már bemutattam, majd egy képet raktam be háttérként, és a megfelelő pozícióba helyeztem a weboldal címét.

### 3.2.3 Menüpontok

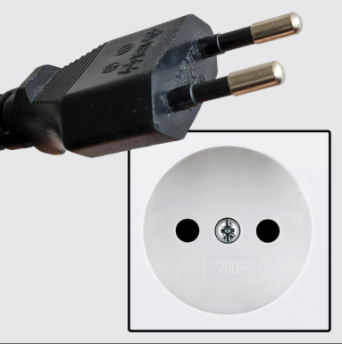
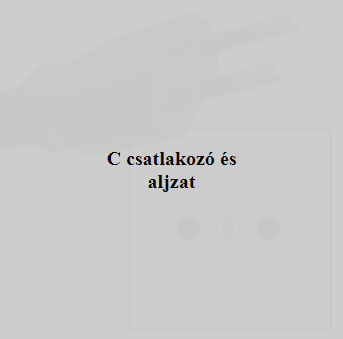
Kezdőlapon kívül hét menüpont található amelyeket most fogok bemutatni. Némelyiknél almenüpontok is találhatóak.

**Fogalmak**

Ezen menüponton belül mutattam be a villanyszerelésben használatos alapvető fogalmakat mint a feszültség, áram, ellenállás és teljesítmény.

Feszültség almenüpontnál már adatbázisból tölt be néhány adatot, így ha valamit hozzá szeretnénk adni például az ’Elektromos hálózati csatlakozók, feszültségek és frekvenciák listája európai országok lebontásában’ táblázathoz akkor nem kell hozzányúlnunk a kódhoz, elég egy új sort hozzáadnunk az adatbázis táblához és a PHP kódunk hozzárendeli a weblapunkhoz.

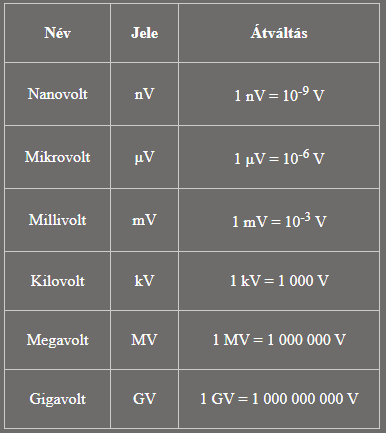
A csatlakozók típusáról raktam be képeket, így legalább a felhasználó el tudja képzelni, hogy mégis hogyan nézhetnek ki. Úgy oldottam meg, hogy ha a felhasználó a képre ráhúzza a kurzort, úgy jelenjen meg a csatlakozó típusa.



10. ábra: Csatlakozó és aljzat formázása

Az áram, ellenállás és teljesítmény menüpontoknál használnom kellett a MathJax-et, így szépen ki tudtam rajzolni a fogalmakkal kapcsolatos képleteket.

**Mértékegységek**

Mindegyik mértékegység almenüpontnál felhasználtam a ’mertekegyseg’ táblámat. Így nem kellett kézzel beírnom a különböző mértékegységek neveit, elég volt betölteni az adatbázisból az adatokat és a megfelelő helyre odaírni a mértékegység jelét, illetve nevét. Látható, hogy a név utolsó része, a jel második betűjele, illetve az átváltás mértékegysége változik csak a táblázatokban.

11. ábra: Mértékegységek táblázata

**Rajzjelek**

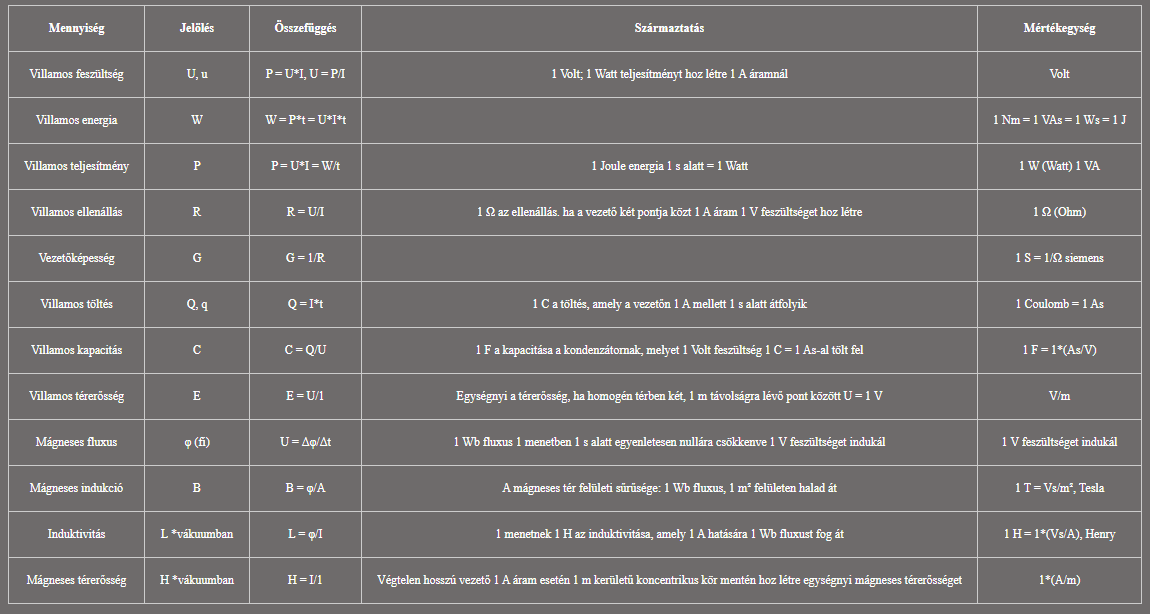
Ennél a menüpontnál csak képeket kellett beszúrnom a különféle villamosságban használt rajzjelekből. Ezeket úgy formáztam, hogy az oldal közepére kerüljenek.

**Képletek és összefüggések**

Itt is PHP kódot használtam, így nagyon egyszerű az adatok módosítása. A ’kepletek’ táblában tároltam az adatokat, szóval elég itt is az adatbázis módosítása és az adatok megváltoznak az oldalon.

A kód a ’kepletek’ táblában tárolt oszlopok adatait várja, mint paraméter, majd ezeket egy táblázat megfelelő oszlopaiba tölti be.

Az elkészült táblázat a 12. ábrán látható.



12. ábra: Képletek táblázat

A táblázat formázását itt is a <style></style> tagok között kellett elvégeznem, mint ahogy korábban már bemutattam.

Későbbiekben nagyon egyszerűen lehet további képleteket hozzáadni a táblázathoz. Elég egy INSERT INTO funkciót megírnunk MySQL-ben a megfelelő adatokkal és máris bővül a táblázat.

**Törvények**

Ohm és Kirchhoff törvényeit mutattam be, néhány képpel illusztrálva. A formázás itt is hasonló a többi menüponthoz képest.

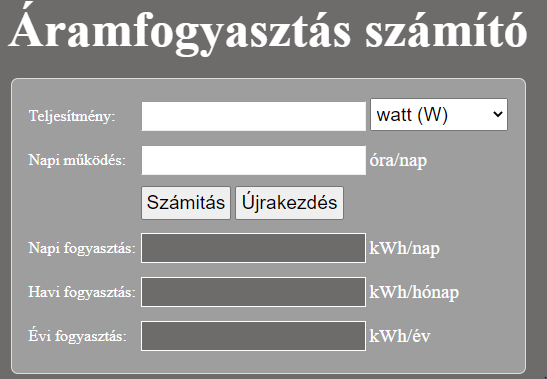
**Kalkulátorok**

Két kalkulátort készítettem el, egy amelyik az áramfogyasztás számításában segít, a másikkal pedig könnyedén kiszámolhatjuk a villanyszámlánkat.

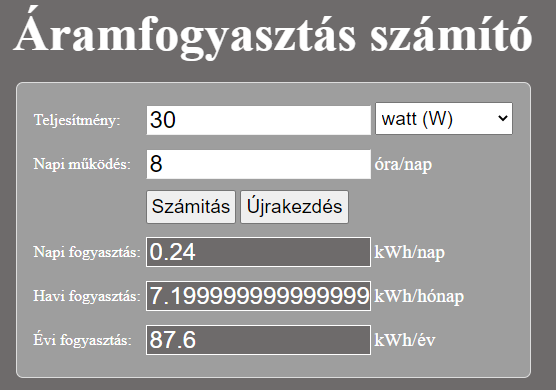
JavaScript kódok segítségével készítettem el a számításokat, majd ezeket egy táblázatba rendeztem.

**Áramfogyasztás kalkulátor**

13. ábra: Áramfogyasztás kalkulátor

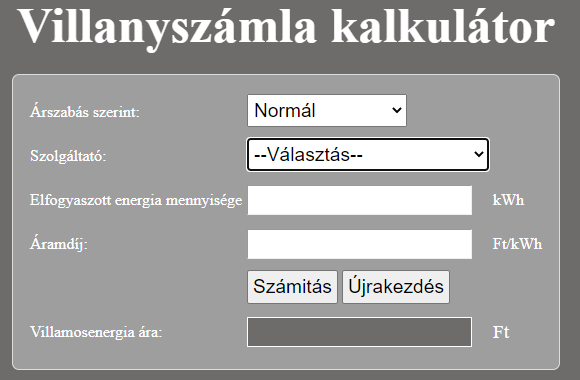


Két bemeneti adatot vár a funkció. Elsőként meg kell adnunk az adott berendezés teljesítményét. Egy lenyíló listából ki kell választanunk, hogy wattban adtuk meg az adatot vagy kilowattban ( 1000 W = 1 kW). Utána meg kell adnunk, hogy a berendezés naponta hány órán keresztül működik. Ezek az adatok alapján a számító kiszámolja a fogyasztást napra, hónapra és évre lebontva. Példaként egy 30 Wattos izzónak a fogyasztását számoltam ki, amely naponta 8 órán keresztül működik.

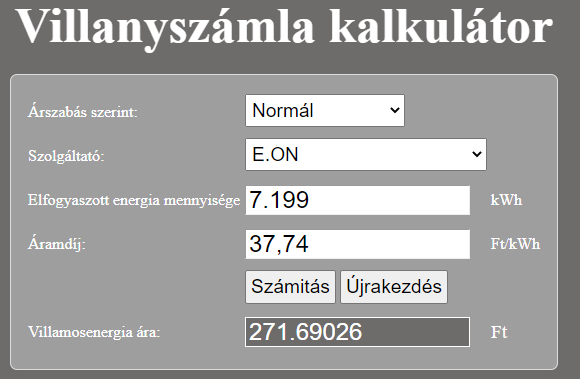


14. ábra: Példa számítás

#### 3.2.3.1 Villanyszámla kalkulátor

Itt már két funkciót kellett megírnom. Elsőként a ’kivalasztott’ funkciót mely nagyban megkönnyíti a felhasználónak a dolgát. Úgy írtam meg a kódot, hogy az oldal használójának ne kelljen megkeresnie az adott energiaszolgáltató árait az interneten, elég annyit tudnia. hogy melyik szolgáltatónál van és a lenyíló listában kiválasztania azt, és a kalkulátor automatikusan azzal az adott árral fog számolni. Továbbá azt is megadhatjuk hogy normál vagy kedvezményes áron számoljon (az árak itt is változnak). A ’villanyszamla’ funkció lekéri a táblázatba beírt adatokat és az alapján összeszorozza az elfogyasztott energia mennyiségét a szolgáltató díjával, így megkapjuk a számlánk végösszegét.

15. ábra: Villanyszámla kalkulátor

Példaként az E-ON szolgáltató, normál árszabású adatával fogom kiszámolni az előzőleg kiszámolt izzó havi fogyasztását.

16. ábra: Példa számítás

A 30 Wattos izzó napi 8 órás működéssel havonta 7.199 kW energiát fogyaszt, melynek díja 271.69 Ft.

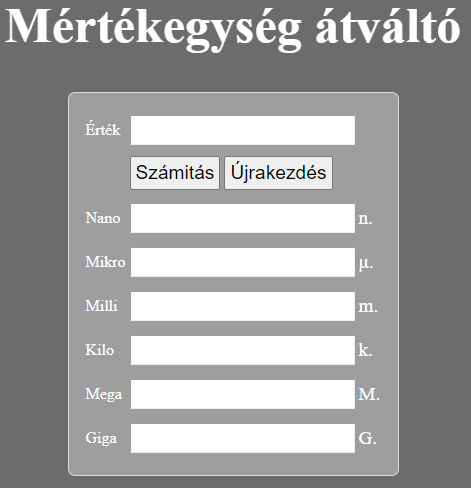
Későbbi kiegészítés lehet, hogy az energiaszolgáltatók árait a funkció lekérdezi az adott szolgáltató weboldaláról, vagy az adatokat adatbázisban tároljuk és onnan olvassa be, így nem kell majd esetleges árváltozáskor belenyúlni a JavaScript kódba.

**Mértékegység átváltó**

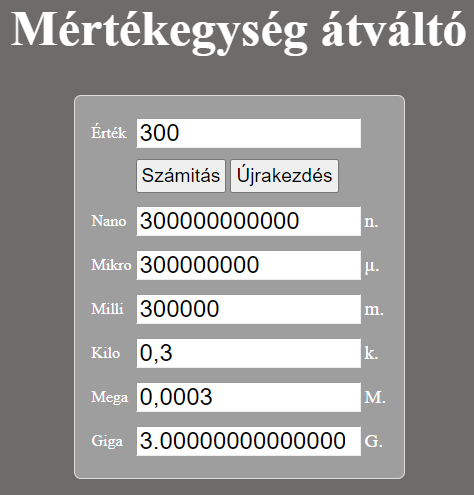
Segítségével a beírt értéket a kalkulátor automatikus átszámolja. A bemeneti érték lehet Volt, Watt, Ohm, Amper hiszen mindegyik mértékegység átváltásnál ugyanazok az átváltás nevei és értékei.

A funkció tulajdonképpen a bemeneti értéket szorozza meg a megfelelő számmal.

* nano=ertek\*1000000000
* mikro =ertek\*1000000
* milli=ertek\*1000
* kilo=ertek\*0.001
* mega=ertek\*0.000001
* giga=ertek\*0.000000001



17. ábra: Mértékegység átváltó

Példaként a bemeneti érték 300 lesz.

18. ábra: Példa számítás

# 4. Összegzés

A fentiekben bemutattam az általam elkészített weboldalt. Remélem az oldalammal picit közelebb kerülhettek a villanyszereléshez, és sok új hasznos információval lehettek gazdagabbak.

Későbbiekben a weboldalam bővíthető mind megjelenítés szempontjából, mind új funkciók hozzárendelésével, illetve új menüpontok létrehozásával.

Számomra főleg a JavaScript kódok megírása jelentett kihívást. JavaScript kódok megírásában nem volt tapasztalatom, így nagyon sokat kellett böngésznem az interneten, hogy el tudjam sajátítani. Eleinte sok hibát dobtak az általam elkészített funkciók, de végül sikerült őket kijavítanom, így a felhasználók számára hasznos funkciókat tudtam létrehozni.

Eleinte nem használtam adatbázist, így mindent a HTML kódba írtam bele. Rájöttem, hogy ha később valamit át szeretnénk írni, vagy esetleg egy új képletet hozzáadni az oldalhoz akkor bele kell nyúlni magába a HTML kódba, amely bonyolult és esetleges hibákhoz vezethet. A probléma kiküszöbölésére hoztam létre az adatbázist, ahová az adatokat feltöltöttem, majd később onnan hivatkoztam rájuk. Ezzel a megoldással elég volt csak az adatbázist módosítani nem kellet belenyúlni magába a kódba.

Továbbfejlesztési ötleteim között szerepel a mobil nézet létrehozása a weboldalhoz, illetve különféle oktató videók hozzáadása.

Későbbiekben szeretném elérhetővé tenni weboldalamat a felhasználók számára, mivel az interneten nem található semmilyen alternatíva azok számára akik jobban meg szeretnének ismerkedni a villanyszereléssel kapcsolatos tudnivalókkal.

# 5. Irodalomjegyzék

[1] Richard Blum - PHP, MySQL & JavaScript 7 könyv 1-ben (Utolsó megtekintés: 2020. október 27.)

[2] Kozmajer Viktor - PHP és MySQL az alapoktól (Utolsó megtekintés: 2020. november 3.)

[3] Mathjax dokumentáció. Mathjax. <http://docs.mathjax.org/en/latest/> (Utolsó megtekintés: 2020. november 2.)

[4] XAMPP keretrendszer használata. SZTE TTIK Informatikai Intézet. <https://www.inf.u-szeged.hu/~gnemeth/adatbgyak/exe/MySQL_XAMPP_JDBC/a_xampp_keretrendszer_s_hasznlata.html> (Utolsó megtekintés: 2020. november 7.)

[5] Dienes Zoltán – Villamosipari rajzok (Utolsó megtekintés: 2020. október 16.)

[6] Danás Miklós – Elektrotechnikai alapismeretek – villamos alapfogalmak (Utolsó megtekintés: 2020. november 10.)

[7] Dienes László – Kliment Tibor – Villanyszerelő szakmai ismeretek I. (Utolsó megtekintés: 2020. november 10.)

[8] Learn HTML. W3SCHOOLS. <https://www.w3schools.com/html/> (Utolsó megtekintés: 2020. október 19.)

[9] Learn JavaScript. W3SCHOOLS. <https://www.w3schools.com/js/default.asp> (Utolsó megtekintés: 2020. november 20.)

[10] JavaScript basics. MDN web docs. <https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics> (Utolsó megtekintés: 2020. november 22.)