

ZÁRÓDOLGOZAT

Készítették:  
Mecsei Péter – Sándor Kristóf Illés

Konzulens:  
Kerényi Róbert Nándor

Miskolc

2024.

Miskolci SZC Kandó Kálmán Informatikai Technikum

Miskolci Szakképzési Centrum

SZOFTVERFEJLESZTŐ- ÉS TESZTELŐ SZAK  
  
  
  
  
  
  
  
Webshop  
*Bemutatása*

Mecsei Péter – Sándor Kristóf Illés

2023-2024

Tartalomjegyzék

Miért webshopot választottunk?:

Azért a webshopot választottunk mert szerintünk egy hétköznapi szoftver a webshop de mégis megmérettetés számunkra, illetve hasznos tapasztalatot tudunk szerezni frontend-en és backend-en.

A szoftver célja:  
 Az elsődleges cél egy olyan webshop készítése, amely egyszerű és könnyen kezelhető felhasználói felülettel rendelkezik.Ez a felhasználók számára megkönnyíti a böngészést és a vásárlást.

**Adatbázis és annak kezelése**

**Fejlesztői környezet:**

Az adatbázisunk létrehozására a **phpMyAdmin** felületét választottuk.

A phpMyAdmin egy olyan webes alkalmazás, amelyet adatbázis kezelésére és adminisztrációjára használnak, különösen **MySQL** és MariaDB adatbázisok esetén.

Ez az alkalmazás lehetővé teszi az adatbázisok létrehozását, módosítását, törlését és karbantartását egy felhasználói felületen keresztül, ami egy böngésző segítségével elérhető. Népszerűségét elsősorban az egyszerű használat és a széles körű funkcionalitás jellemzi.

**Néhány fő ok amiért hasznos, illetve amik miatt mi is ezt választottuk:**

**Grafikus felhasználói felület:** Egy olyan grafikus felhasználói felületet kínál, ami lehetővé teszi az adatbázisok és azok tábláinak kezelését virtuálisan. Ez főleg azért hasznos mert meggyorsítja a munkánkat azzal, hogy nem kell parancssoros interfészt használnunk.

**Adatbázisok karbantartása:** Az adatbázisok karbantartása, például az indexek újragenerálása, a táblák optimalizálása vagy az adatok exportálása és importálása is könnyen elvégezhető pár kattintással.

**Felhasználók és jogosultságok kezelése:** A phpMyAdmin lehetőséget biztosít a felhasználók és a hozzájuk kapcsolódó jogosultságok kezelésére az adatbázisokon belül. Ennek köszönhetően alap szintű ellenőrzést biztosíthatunk az adatbázisokhoz való hozzáférések felett.

**Adatbázisok és táblák kezelése:** Ennek a felületnek köszönhetően könnyedén hozhatunk létre és módosíthatunk adatbázisokat, azok tábláit, illetve adatokat is hozzáadhatunk és módosíthatunk vagy törölhetünk.

**Importálás és exportálás:** Az Importálás és Exportálás funkciókkal könnyedén lehet adatokat importálni az adatbázisba, vagy exportálni adatokat más formátumokba, például SQL vagy CSV.

**Összességében:** A phpMyAdmin egy olyan alkalmazás, amely lehetővé teszi az adatbázisok kezelését egy intuitív webes felületen keresztül. Könnyen használható funkcióival és grafikus felületével segít a felhasználóknak hatékonyan kezelni és karbantartani az adatbázisokat anélkül, hogy közvetlenül parancssoros műveleteket kellene végezniük. Ezek miatt különösen hasznos minden olyan személy vagy szervezet számára, akik MySQL vagy MariaDB adatbázisokat használnak webes alkalmazások vagy projektek adatainak tárolására.

**Az elképzelés:**

A tervezés alatt egy olyan adatbázist szerettünk volna létrehozni melynek felépítése segíti az adatbázis tábláiban lévő adatok **jó átláthatóság**át és **kezelhetőség**ét. Emellett a megvalósítás majd a tesztelés során létrejövő hibák, esetleges hiányosságok könnyen orvosolhatóak legyenek.

**A megvalósítás:**

A megvalósítás menetében próbáltunk az előző pont alapján létrehozott tervhez ragaszkodni. A terv elég jól sikerült, ugyanis csak apró módosításokat kellett eszközölnünk.

Természetesen a fejlesztés során még adódtak **apróbb gondok**, például hozzákellett adnunk egy plusz mezőt egy táblához, vagy éppen kikellett törölni, de alapjába véve egy nagyon jó adatbázist sikerült létrehoznunk.

A screenshot of a computer

Description automatically generated A táblák **összekötése** is remek lett mivel a backend fejlesztésének menetében egyértelmű volt hogy milyen lekérdezéseket kell alkalmaznunk, azokat hogy kell megvalósítanunk, valamint miként kell feltöltenünk adatokat bizonyos táblákba.

1. ábra Az adatbázis felépítése, kapcsolatok a táblák közt

A mellékelt ábrán jól látható pontosan az adatbázisunk felépítése, hogy milyen táblákat hoztunk létre és az adott táblák milyen mezőket tartalmaznak.

Igyekeztünk csak a **szükséges** és **fontos** dolgokat eltárolni az adatbázisban, ilyen például hogy minden rekord **egyedi azonosító**val rendelkezik, ami megakadályozza a rekordok duplikálását, illetve hogy lekérdezésekkor és adatok feltöltésekor ne akadjon össze az adatbázis.

Fontos megjegyezni hogy a **termékek**nek a **képeit** is adatbázisban tároljuk, gondolkodtunk rajta hogy a képeket ftp szerveren tároljuk el, de egyszerűbbnek találtuk így hogy minden egy helyen. Elég volt csak írni pár függvényt arra hogy a képeket megtudjuk jeleníteni frontenden és hogy az admin felületünkről tudjunk hozzáadni vagy módosítani képet az adatbázisban.

Jól látható hogy például a ’felhasznalo’ táblában az ’email’ mező **egyedi kulcs**csal van ellátva. Ez azért jó nekünk mert adatbázis szinten is védelmet nyújt a redundanciával szemben.

**Normál forma:** Az adatbázisban lévő táblák és a köztük lévő kapcsolatok struktúrája alapján megállapítható hogy az adatbázisunk harmadik normál formában van(3NF). A harmadik normál forma azt foglalja magába hogy **minden kulcs attribútum tranzitív függőségektől mentes** a kulcsokon belül. Ez azt jelenti, hogy nincs olyan nem kulcs attribútum amely más nem kulcs attribútumtól függ.

Emellett ha egy adatbázis a harmadik normál formában van, az azt jelenti, hogy automatikusan megfelel az első és a második normál formáknak is. Az adatbázis normál formái egymásra épülnek, tehát az alacsonyabb rendű normál formák követelményeit is teljesíteni kell ahhoz hogy magasabb normál formában legyen egy adatbázis.

A screenshot of a computer

Description automatically generated A normál formákra való felosztás a tervetés során segít optimalizálni az adatszerkezetet, minimalizálni a redundanciát és biztosítani az adatintegritást.

2. ábra Példa a normál formára

A ’’felhasznalo’’ tábla attribútumai (Nev,Email,Jelszo,Megerositve) között nincs tranzitív függőség. A tranzitív függőség azt jelenti, hogy két nem-kulcs mező függ egymástól és közvetetten függ a tábla elsődleges kulcsától.

A ’’felhasznalo\_cim’’ táblában a címek az egyedi felhasználókhoz vannak rendelve a ’’Felhasz\_Id’’ segítségével.

**WebAPI backend**

**Mi a WebAPI?**

A **WebAPI (Web Application Programming Interface)** egy olyan programozási interfész, amely lehetővé teszi a különböző szoftveralkalmazások közötti kommunikációt a világhálón keresztül.

Gyakran használják alkalmazások közötti adatcserére és kommunikácóra. A WebAPI-k általában **HTTP (Hypertext Transfer Protocol) protokol**lt használnak üzenetek küldésére és fogadására.

**Fontos tulajdonságai, amiért választottuk:**

**Kommunikáció kliens és szerver között:** A WebAPI-k lehetővé teszik a kliensalkalmazások számára, hogy kéréseket küldjenek a szervernek és válaszokat kapjanak. Ez lehetővé teszi az alkalmazások közti kommunikációt.

**RESTful szolgáltatások:** Sok WebAPI implementálja a **REST (Representational State Transfer)** architektúrát, ami egy olyan tervezési stílus amely egyszerű könnyen és jól érthető interfészt biztosít az alkalmazások közti kommunikációhoz. A RESTful API-k erőforrásokat kezelnek, amelyeket **egyedi URI**-kon keresztül érhetünk el, különböző HTTP módszerekkel **(GET, POST, PUT, DELETE).**

**Adatcsere formátumok:** A WebAPI-k támogatják a különböző adatcsere formátumokat, például **JSON (JavaScript Object Notation)** vagy **XML (Extensible Markup Language).** Ezek a formátumok lehetővé teszik az adatok strukturált és egyszerű átvitelét alkalmazások közt.

**Platformfüggetlenség:** Mivel a HTTP-t használja alapvető kommunikációs protokollként, a WebAPI-k platformfüggetlenek, mivel szinte minden modern alkalmazás támogatja a HTTP-t. Ez azt jelenti, hogy az alkalmazások különböző platformokon ,például webböngészőkön, mobilalkalmazásokban, asztali alkalmazásokban, képesek kommunikálni egymással.

**Amik szintént WebAPI-t használnak:**

**Szociális hálózatok,** például Facebook vagy Twitter, API-ja lehetővé teszi az alkalmazások számára, hogy posztokat, felhasználói adatokat stb. küldjenek, módosítsanak vagy lekérjenek.

**Pénzügyi szolgáltatók**, például bankszámla kezelési rendeszerek, API-ja a tranzakciók végrehajtását, egyenlegek lekérdezését teszi lehetővé.

**Időjárás alkalmazások,** például BBC Weather, az időjárással kapcsolatos adatokat (időkép, hőmérséklet, szél erőssége stb.) képesek lekérdezni különböző helyszínekre.

Ezek mellet van még egy fontos szempont ami miatt WebAPI-t használtunk, ez pedig a mi fejlesztőkörnyezetünkben elérhető **ASP.NET Core**-t használó WebAPI.

**Mi is az az ASP.NET Core?**

Az ASP.NET Core egy **nyílt forráskódú**, **cross-platform** keretrendszer amelyet **webalkalmazások** és szolgáltatások fejlesztésére használnak. Az ASP.NET Core-t a Microsoft fejleszti és a .NET Core keretrendszer része. Ez a **keretrendszer** lehetővé teszi a modern és hatékony alkalmazások létrehozását különböző környezetekben, Windows, macOS és Linux rendszereken. **Modern**, könnyűsúlyú és **teljesítményorientált** keretrendszer, amely széles körű felhasználási lehetőséget kínál.

**Főbb jellemzői:**

**Cross-platform támogatás:** Teljes mértékben cross-platform, ami azt jelenti hogy képes futni Windows, macOS és Linux rendszeren is. Ez lehetővé teszi a fejlesztők számára, hogy webalkalmazásokat üzemeltethessenek és hozzanak részre bármilyen operációs rendeszen.

**Modularitás**: Az ASP.NET Core egy moduláris keretrendszer, ez azt jelenti hogy a fejlesztők csak azokat a komponenseket használhatják amelyekre szükségük van. Ez javítja a teljesítményt és csökkenti az alkalmazás méretét.

**Nyílt forráskód:** A teljesen nyílt forráskód lehetővé teszi a közösség számára, hogy hozzájáruljon a fejlesztéshez és a javításokhoz. Emellett ez növeli az átláthatóságot és az alkalmazások függetlenségét.

**Korszerű architektúra:** Támogatja a modern fejlesztési elveket, például az **MVC (Model-View-Controller)** tervezési mintát, valamint könnyen integrálható a fejlett fejlesztői eszközökkel, mint a Visual Studio IDE.

**Tejlesítményorientált:** Optimalizált és hatékony teljesítményt nyújt a nagy terhelésű webalkalmazások számára.

**Kompatibilitás:** Lehetővé teszi a korábbi ASP.NET alkalmazások migrálását és frissítését a legújabb verzióra, miközben a régebbi technológiákat is támogatja.

**Főbb felhasználási területei:**

Webalkalmazások és weboldalak fejlesztése, például kereskedelmi vagy adminisztrációs felületek.

API-k készítése és üzemeltetése, amelyeket más alkalmazások használnak, mint a mi projektünk esetében is.

Valós idejű alkalmazások (real-time apps) készítése, például chat alkalmazások és élő közvetítések.