Tartalomjegyzék

Témaválasztás

1. Fejlesztői dokumentáció

1.1. Fejlesztő környezet

-Visual Studio Code

1.2. Nyelvek, technológiák

- HTML  
- PHP

- Bootstrap 5.2.

- OOP

- MVC

- REST API

- XAMPP

- MySql

1.3. Adatbázis

1.3.1Adatbázis alapadatai

1.3.2.Kapcsolatok

1.3.3. Táblák

1.3.4. Funkciók

2. Felhasználói dokumentáció

Témaválasztás

Az én témám egy ács munkákkal foglalkozó cég építőanyag nyilvántartása.

Egyik ismerősöm kért fel, hogy csináljak egy olyan rendszert amiben a különböző építőnyagokat rendszerezni tudja, hogy megkönnyítse a munkáját. Szerette volna ha bármikor bárhonnan elérhető, így csináltam egy építőanyag adatbázis kezelő webes applikációt neki.

Ez a mostani igényeinek megfelelően készült, de mivel növelni akarja a vállalkozást, tovább lesz fejlesztve az oldal.

**1. Fejlesztői dokumentáció**

1.1. Fejlesztői környezet

- Visual Studio Code

A Visual Studio Code (rövidítve: VSCode vagy VS Code) ingyenes nyílt forráskódú kódszerkesztő, melyet a Microsoft fejleszt Windows, Linux és macOS operációs rendszerekhez. Támogatja a hibakeresőket, valamint beépített Git támogatással rendelkezik, továbbá képes az intelligens kódkiegészítésre az IntelliSense segítségével. A VSCode-ban a felhasználók megváltoztathatják a kinézetet (témát), megváltoztathatják a szerkesztő gyorsbillentyű-kiosztását, az alapértelmezett beállításokat és még sok egyebet. Támogatja a kiegészítőket, melyek segítségével további funkciók, testreszabási lehetőségek érhetőek el.

1.2. Nyelvek,technológiák

- HTML

A HTML angol mozaikszó, jelentése HyperText Markup Language, vagyis hiperszöveges jelölőnyelv. A weboldalakat leíró jelnyelv, vagy az a kódrendszer, amit a weboldalak elkészítéséhez használnak.

A HTML nem programozási nyelv, hanem kódnyelv, aminek segítségévek akár a Jegyzettömb segítségével is alkothatóak weboldalak. A hipertext jelenti valójában az interneten található oldalakat (dokumentumokat), amelyek szöveg, kép, videó, hang, animáció, vagy ezek valamilyen kombinációjából áll. A HTML ezeknek a dokumentumoknak az elrendezését, formázását tartalmazza saját jelölőnyelvén. Bár a honlap készítő programok grafikai felületet is biztosítanak a weblap elkészítéséhez, mellette a legtöbb programnál hipertextként is kialakítható egy oldal.

- PHP

A PHP egy szerveroldali szkriptnyelv, mely segítségével dinamikus weblapokat készíthetünk. A PHP nyelven írt kódokat a webszerver PHP feldolgozómodulja értelmezi. A PHP egy olyan programozási nyelv, mely segítségével képesek lehetünk elkészíteni egy adatbázisalapú weboldalt is. Ilyenkor kerül képbe a szerveroldal, de erről majd egy kicsit később.

Mivel a PHP nyelv egy szerveroldali szkriptnyelv, így gond nélkül végezhetünk vele adatbázis műveleteket vagy létesíthetünk kapcsolatot távoli kiszolgálókkal.

A PHP szó eredetileg a „Personal Home Page Tools” kifejezést takarta, amely arra utalt, hogy segítségével személyre szabhatjuk a weboldalainkat, azonban ez a név később módosult a „Hypertext Preprocessor„-rá, amely már inkább arra utal, hogy ez a nyelv egy hypertext előfeldolgozó.

- Bootstrap 5.2.

A Bootstrap egy nyílt forráskódú keretrendszer (framework), mely HTML, CSS, JavaScript technológiákat használ. Alapvetően arra jó, hogy nagyon könnyedén, és minimális energia befektetéssel tudjon valaki jól kinéző, bármilyen képernyőméreten szépen megjelenő weboldalakat készíteni.

-OOP

Az osztályalapú programozás, vagy általánosabban osztályorientált programozás, az objektumorientált programozásnak (OOP) egy stílusa, amelyben az öröklés az objektumok osztályainak meghatározása révén történik, ahelyett, hogy az öröklődés az objektumokon keresztül történne (összehasonlítva a prototípus-alapú programozással).

- MVC

A modell-nézet-vezérlő (MNV) (angolul model-view-controller) a szoftvertervezésben használatos programtervezési minta.[1] Összetett, sok adatot a felhasználó elé táró számítógépes alkalmazásokban gyakori fejlesztői kívánalom az adathoz (modell) és a felhasználói felülethez (nézet) tartozó dolgok szétválasztása, hogy a felhasználói felület ne befolyásolja az adatkezelést, és az adatok átszervezhetők legyenek a felhasználói felület változtatása nélkül. A modell-nézet-vezérlő ezt úgy éri el, hogy elkülöníti az adatok elérését és az üzleti logikát az adatok megjelenítésétől és a felhasználói interakciótól egy közbülső összetevő, a vezérlő bevezetésével.[2][3]

Hagyományosan asztali felhasználói felületekhez használt, de manapság már webalkalmazásokhoz is népszerűvé vált.[4] Népszerű programozási nyelvek mint a JavaScript, Python, Ruby, PHP, Java, C# és Swift már külön telepítés szükségessége nélkül rendelkeznek MNV keretrendszerekkel web- és mobilalkalmazások fejlesztésére.

- REST API

A REST API az integrációs keretrendszer része, melynek segítségével külső alkalmazásokat integrálhat a folyamatautomatizálási alrendszer alkalmazásaival. A REST alkalmazásprogramozási felület REST erőforrásokként teszi elérhetővé az üzleti objektumokat és integrációs objektumstruktúrákat. A REST API XML vagy JavaScript objektumjelölés (JSON) formátumban képes erőforrásadatokat megadni.

A REST API szabványos HTTP GET, POST, PUT és DELETE metódusok felhasználásával támogatja az erőforrásokon végrehajtott létrehozási, lekérdezési, frissítési és törlési műveleteket.

-XAMPP

A XAMPP – (kiejtése: /ˈzæmp/ vagy /ˈɛks.æmp/[1]) – egy szabad és nyílt forrású platformfüggetlen webszerver-szoftvercsomag, amelynek legfőbb alkotóelemei az Apache webszerver, a MariaDB (korábban a MySQL[2]) adatbázis-kezelő, valamint a PHP és a Perl programozási nyelvek értelmezői (végrehajtó rendszerei). Ez a szoftvercsomag egy integrált rendszert alkot, amely webes alkalmazások készítését, tesztelését és futtatását célozza, és ehhez egy csomagban minden szükséges összetevőt tartalmaz. A rendszer egyik nagy előnye az összehangolt elemek könnyű telepíthetősége.

-MySql

A MySQL egy többfelhasználós, többszálú, SQL-alapú relációs adatbázis-kezelő szerver.

A MySQL az egyik legelterjedtebb adatbázis-kezelő, aminek egyik oka lehet, hogy a teljesen nyílt forráskódú LAMP (Linux–Apache–MySQL–PHP) összeállítás részeként költséghatékony és egyszerűen beállítható megoldást ad dinamikus webhelyek szolgáltatására.

Forrás:

https://matebalazs.hu/html.html

https://webiskola.hu/php-ismeretek/mi-az-a-php-fogalma-bemutatasa/

https://gremmedia.hu/mi-az-a-bootstrap-4-hogyan-hasznaljuk

https://hu.wikipedia.org/wiki/Oszt%C3%A1lyalap%C3%BA\_programoz%C3%A1s

https://www.interviewbit.com/blog/mvc-architecture/

https://www.ibm.com/docs/hu/mam/7.6.1?topic=apis-rest-api

https://hu.wikipedia.org/wiki/XAMPP

https://hu.wikipedia.org/wiki/MySQL

1.3. Adatbázis

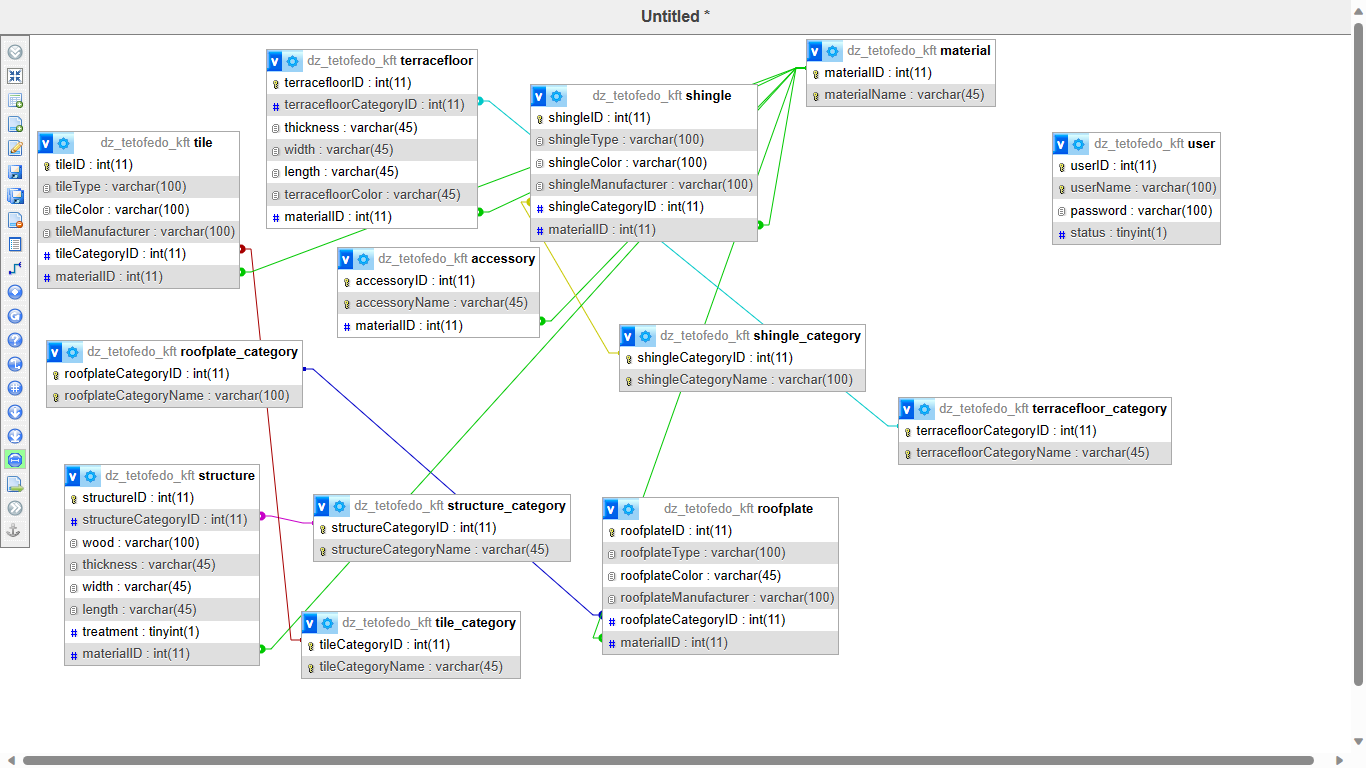
1.3.1. Adatbázis alapadatai

Sql szerver: mysql

Adatbázis neve: dz\_tetofedo\_kft

Adatbázis létrehozása: CREATE SCHEMA `dz\_tetofedo\_kft` DEFAULT CHARACTER SET utf8 COLLATE utf8\_bin ;

1.3.2.Kapcsolatok



1.3.3. Táblák

1.3.3.1. **accessory** – a kiegészítőket tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK/  PK |
| accessoryID | INT(11) | Kiegészítő azonosító | PK, FK |
| accessoryName | VARCHAR(45) | Kiegészítő neve |  |
| materialID | INT(11) | Építőanyag azonosító | FK |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`accessory` (  `accessoryID` INT NOT NULL AUTO\_INCREMENT,  `accessoryName` VARCHAR(45) NOT NULL,  `materialID` INT NOT NULL,  PRIMARY KEY (`accessoryID`), FOREIGN KEY (`materialID`) REFERENCES `material`(`materialID`) ON DELETE NO ACTION,  UNIQUE INDEX `accessoryName\_UNIQUE` (`accessoryName` ASC) ); |

1.3.3.2. **material** – az építőanyag kategóriákat tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK/  PK |
| materialID | INT(11) | Építőanyag azonosító | PK/FK |
| materialName | VARCHAR(45) | Építőanyag neve |  |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`material` (  `materialID` INT NOT NULL AUTO\_INCREMENT,  `materialName` VARCHAR(45) NOT NULL,  PRIMARY KEY (`materialID`),  UNIQUE INDEX `materialName\_UNIQUE` (`materialName` ASC) ); |

1.3.3.3. **roofplate** – a tetőlemezeket tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK/  PK |
| roofplateID | INT(11) | Tetőlemez azonosító | PK |
| roofplateType | VARCHAR(100) | Tetőlemez típusa |  |
| roofplateColor | VARCHAR(45) | Tetőlemez színe |  |
| roofplateManufacturer | VARCHAR(100) | Tetőlemez gyártója |  |
| roofplatecCategoryId | INT(11) | Tetőlemez kategória azonosító | FK |
| materialID | INT(11) | Építőanyag azonosító | FK |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`roofplate` (  `roofplateID` INT NOT NULL AUTO\_INCREMENT,  `roofplateType` VARCHAR(100) NOT NULL,  `roofplateColor` VARCHAR(45) NOT NULL,  `roofplateManufacturer` VARCHAR(100) NOT NULL,  `roofplateCategoryID` INT NOT NULL,  `materialID` INT NOT NULL,  PRIMARY KEY (`roofplateID`), FOREIGN KEY (`roofplateCategoryID`) REFERENCES `roofplate\_category`(`roofplateCategoryID`)ON DELETE NO ACTION, FOREIGN KEY (`materialID`) REFERENCES `material`(`materialID`) ON DELETE NO ACTION |

1.3.3.4. **roofplate\_category** – a tetőlemez kategóriákat törli

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK/ |
| roofplateCategoryID | INT(11) | Tetőlemez kategória azonosító | PK / FK |
| roofplateCategoryName | VARCHAR(100) | Tetőlemez kategória neve |  |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`roofplate\_category` (  `roofplateCategoryID` INT NOT NULL AUTO\_INCREMENT,  `roofplateCategoryName` VARCHAR(100) NOT NULL,  PRIMARY KEY (`roofplateCategoryID`),  UNIQUE INDEX `roofplateCategoryName\_UNIQUE` (`roofplateCategoryName` ASC) ); |

1.3.3.5. **shingle** – zsindelyeket tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK/ PK |
| shingleID | INT(11) | Zsindely azonosító | PK |
| shingleType | VARCHAR(100) | Zsindely típusa |  |
| shingleColor | VARCHAR(100) | Zsindely színe |  |
| shingleManufacturer | VARCHAR(100) | Zsindely gyártója |  |
| shingleCategoryID | INT(11) | Zsindely kategória azonosító | FK |
| materialID | INT(11) | Építőanyag azonosító | FK |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`shingle` (  `shingleID` INT NOT NULL AUTO\_INCREMENT,  `shingleType` VARCHAR(100) NOT NULL,  `shingleColor` VARCHAR(100) NOT NULL,  `shingleManufacturer` VARCHAR(100) NOT NULL,  `shingleCategoryID` INT NOT NULL,  `materialID` INT NOT NULL,  PRIMARY KEY (`shingleID`), FOREIGN KEY (`shingleCategoryID`) REFERENCES `shingle\_category`(`shingleCategoryID`) ON DELETE NO ACTION, FOREIGN KEY (`materialID`) REFERENCES `material`(`materialID`)ON DELETE NO ACTION ); |

1.3.3.6. **shingle\_category** – zsindely kategóriákat tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK/PK |
| shingleCategoryID | INT(11) | Zsindely kategória azonosító | PK / FK |
| shingleCategoryName | VARCHAR(100) | Zsindely kategória neve |  |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`shingle\_category` (  `shingleCategoryID` INT NOT NULL AUTO\_INCREMENT,  `shingleCategoryName` VARCHAR(100) NOT NULL,  PRIMARY KEY (`shingleCategoryID`),  UNIQUE INDEX `shingleCategoryName\_UNIQUE` (`shingleCategoryName` ASC) ); |

1.3.3.7. **structure** – a szerkezeteket tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK / PK |
| structureID | INT(11) | Szerkezet azonosító | PK |
| structureCategoryID | INT(11) | Szerkezet kategória azonosító | FK |
| wood | VARCHAR(100) | Fa fajta |  |
| thickness | VARCHAR(45) | Szerkezet vastagság |  |
| width | VARCHAR(45) | Szekezet szélesség |  |
| length | VARCHAR(45) | Szerkezet hosszúság |  |
| treatment | TINYINT(1) | Kezelés típusa |  |
| materialID | INT(11) | Építőanyag azonosító | FK |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`structure` (  `structureID` INT NOT NULL AUTO\_INCREMENT,  `structureCategoryID` INT NOT NULL,  `wood` VARCHAR(100) NOT NULL,  `thickness` VARCHAR(45) NOT NULL,  `width` VARCHAR(45) NOT NULL,  `length` VARCHAR(45) NOT NULL,  `treatment` TINYINT(1) NOT NULL,  `materialID` INT NOT NULL,  PRIMARY KEY (`structureID`), FOREIGN KEY (`structureCategoryID`) REFERENCES `structure\_category`(`structureCategoryID`)ON DELETE NO ACTION, FOREIGN KEY (`materialID`) REFERENCES `material`(`materialID`)ON DELETE NO ACTION ); |

1.3.3.8. **structure\_category** – a szerkezet kategóriákat tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK / PK |
| structureCategoryID | INT(11) | Szerkezet kategória azonosító | PK / FK |
| structureCategoryName | VARCHAR(45) | Szerkezet kategória név |  |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`structure\_category` (  `structureCategoryID` INT NOT NULL AUTO\_INCREMENT,  `structureCategoryName` VARCHAR(45) NOT NULL,  PRIMARY KEY (`structureCategoryID`),  UNIQUE INDEX `structureCategoryName\_UNIQUE` (`structureCategoryName` ASC) ); |

1.3.3.9. **terracefloor** – teraszburkolatokat tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK / PK |
| terracefloorID | INT(11) | Teraszburkolat azonosító | PK |
| terracefloorCategoryID | INT(11) | Teraszburkolat kategória azonosító | FK |
| thickness | VARCHAR(45) | Teraszburkolat vastagság |  |
| width | VARCHAR(45) | Teraszburkolat szélesség |  |
| length | VARCHAR(45) | Teraszburkolat hosszúság |  |
| terracefloorColor | VARCHAR(45) | Teraszburkolat szín |  |
| materialID | INT(11) | Építőanyag azonosító | FK |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`terracefloor` (  `terracefloorID` INT NOT NULL AUTO\_INCREMENT,  `terracefloorCategoryID` INT NOT NULL,  `thickness` VARCHAR(45) NOT NULL,  `width` VARCHAR(45) NOT NULL,  `length` VARCHAR(45) NOT NULL,  `terracefloorColor` VARCHAR(45) NOT NULL,  `materialID` INT NOT NULL,  PRIMARY KEY (`terracefloorID`), FOREIGN KEY (`terracefloorCategoryID`) REFERENCES `terracefloor\_category`(`terracefloorCategoryID`) ON DELETE NO ACTION, FOREIGN KEY (`materialID`) REFERENCES `material`(`materialID`)ON DELETE NO ACTION ); |

1.3.3.10. **terracefloor\_category** – teraszburkolatokat tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK / PK |
| terracefloorCategoryID | INT(11) | Teraszburkolat kategória azonosító | PK, FK |
| terracefloorCategoryName | VARCHAR(45) | Teraszburkolat kategória név |  |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`terracefloor\_category` (  `terracefloorCategoryID` INT NOT NULL AUTO\_INCREMENT,  `terracefloorCategoryName` VARCHAR(45) NOT NULL,  PRIMARY KEY (`terracefloorCategoryID`),  UNIQUE INDEX `terracefloorCategoryName\_UNIQUE` (`terracefloorCategoryName` ASC) ); |

1.3.3.11. **tile** – cserepeket tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK / PK |
| tileID | INT(11) |  | PK |
| tileType | VARCHAR(100) |  |  |
| tileColor | VARCHAR(100) |  |  |
| tileManufacturer | VARCHAR(100) |  |  |
| tileCategoryID | INT(11) |  | FK |
| materialID | INT(11) |  | FK |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`tile` (  `tileID` INT NOT NULL AUTO\_INCREMENT,  `tileType` VARCHAR(100) NOT NULL,  `tileColor` VARCHAR(100) NOT NULL,  `tileManufacturer` VARCHAR(100) NOT NULL,  `tileCategoryID` INT NOT NULL,  `materialID` INT NOT NULL,  PRIMARY KEY (`tileID`),  FOREIGN KEY (`tileCategoryID`) REFERENCES `tile\_category`(`tileCategoryID`) ON DELETE NO ACTION, FOREIGN KEY (`materialID`) REFERENCES `material`(`materialID`)ON DELETE NO ACTION ); |

1.3.3.12. **tile\_category** – cserép kategóriákat tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK / PK |
| tileCategoryID | INT(11) | Cserép kategória azonosító | PK , FK |
| tileCategoryName | VARCHAR(45) | Cserép kategória név |  |

Sql:

|  |
| --- |
| CREATE TABLE `dz\_tetofedo\_kft`.`tile\_category` (  `tileCategoryID` INT NOT NULL AUTO\_INCREMENT,  `tileCategoryName` VARCHAR(45) NOT NULL,  PRIMARY KEY (`tileCategoryID`),  UNIQUE INDEX `tileCategoryName\_UNIQUE` (`tileCategoryName` ASC) ); |

1.3.3.13. **user** – a felhasználókat tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Mező neve | Típusa | Leírás | FK / PK |
| userID | INT(11) | Felhasználó azonosító | PK |
| userName | VARCHAR(100) | Felhasználó neve |  |
| password | VARCHAR(100) | Felhasználó jelszó |  |
| status | TINYINT(1) | Jogosultság |  |

1.3.4. Funkciók

- Login

- Logout

- Építőnyagok CRUD

2. Funkciók

a. Alapfunkciók:

. Login

. Logout

. Cég alapadatok CRUD

. User CRUD

. + táblák CRUD

. Jogosultsági rendszer

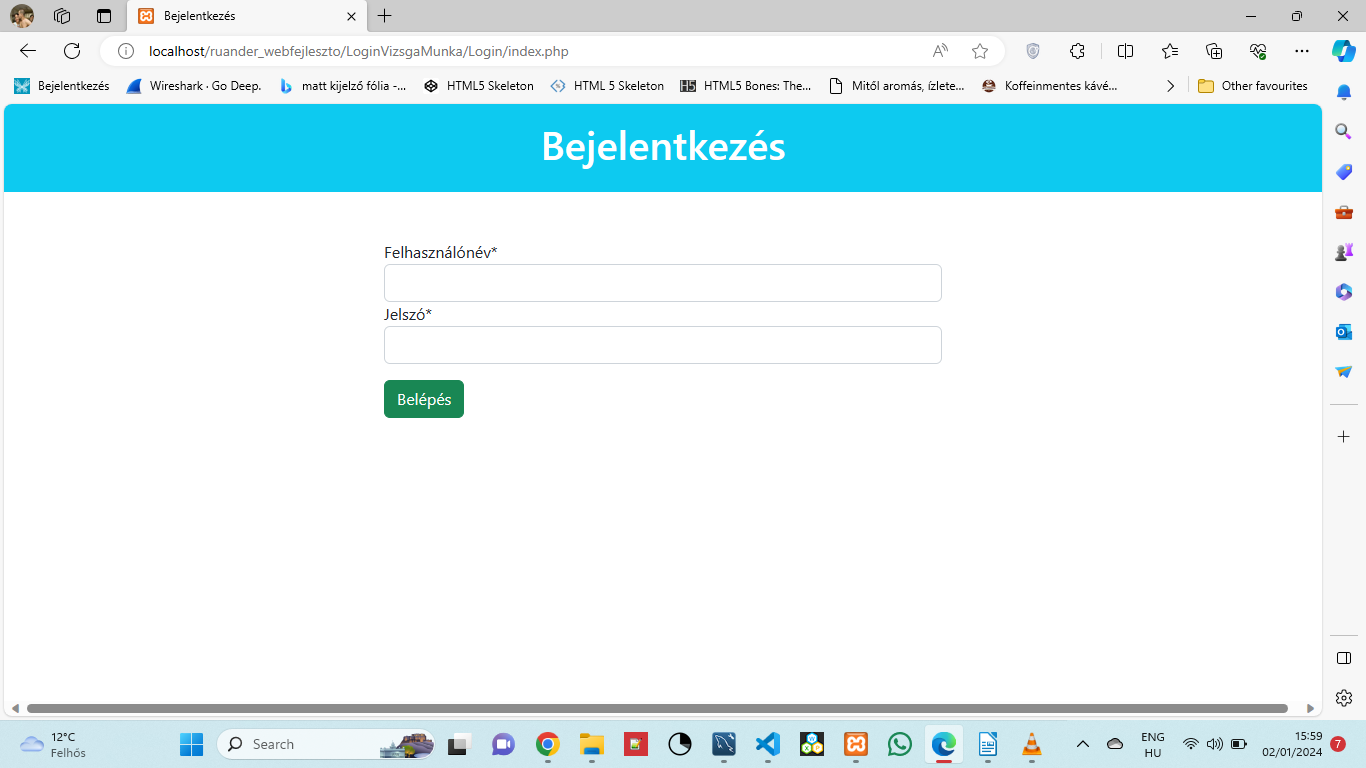
. CMS felhasználói oldal és Admin oldal elkülönítése

. menüpontok DB-ben tárolása

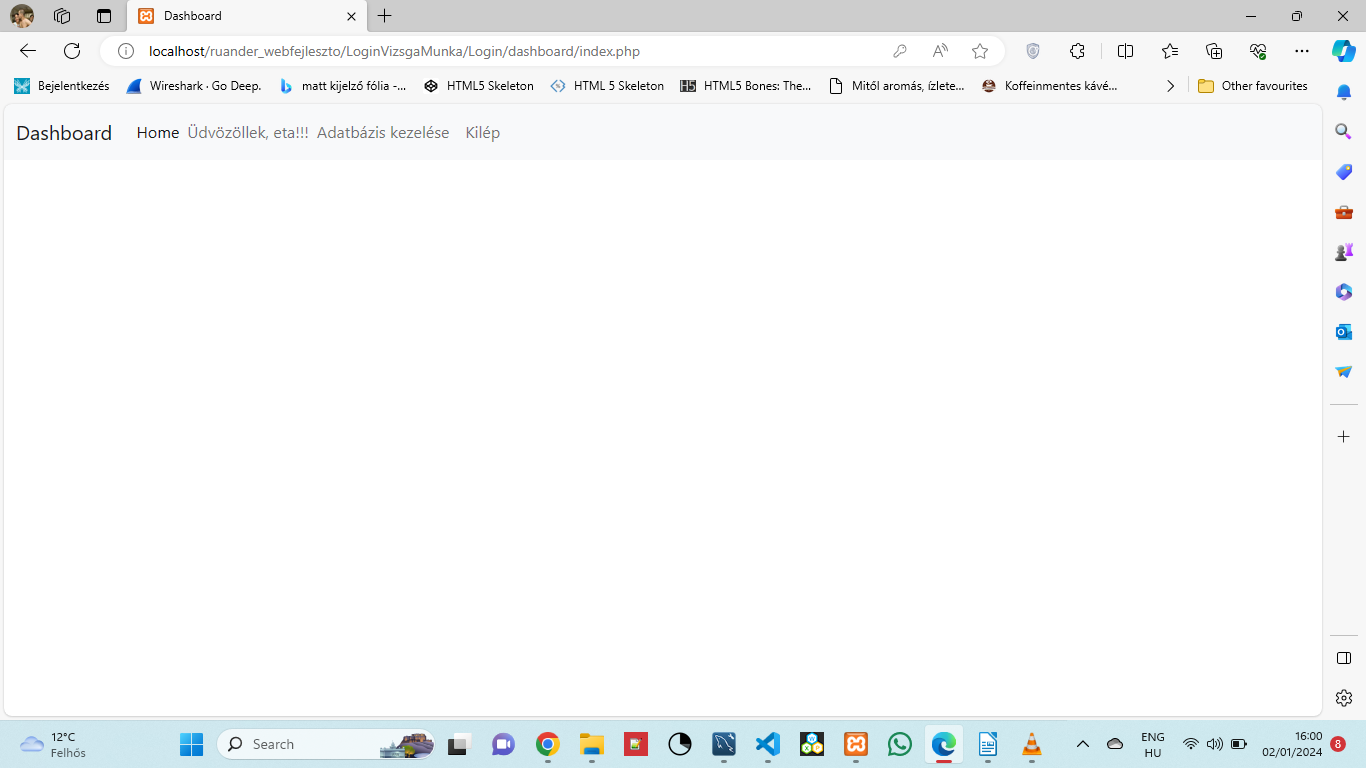
. oldalak tartalom CRUD

2. Felhasználói dokumentáció

Az alábbiakban elmagyarázom, hogyan működik az oldal a bejelentkezéstől a kilépésig.

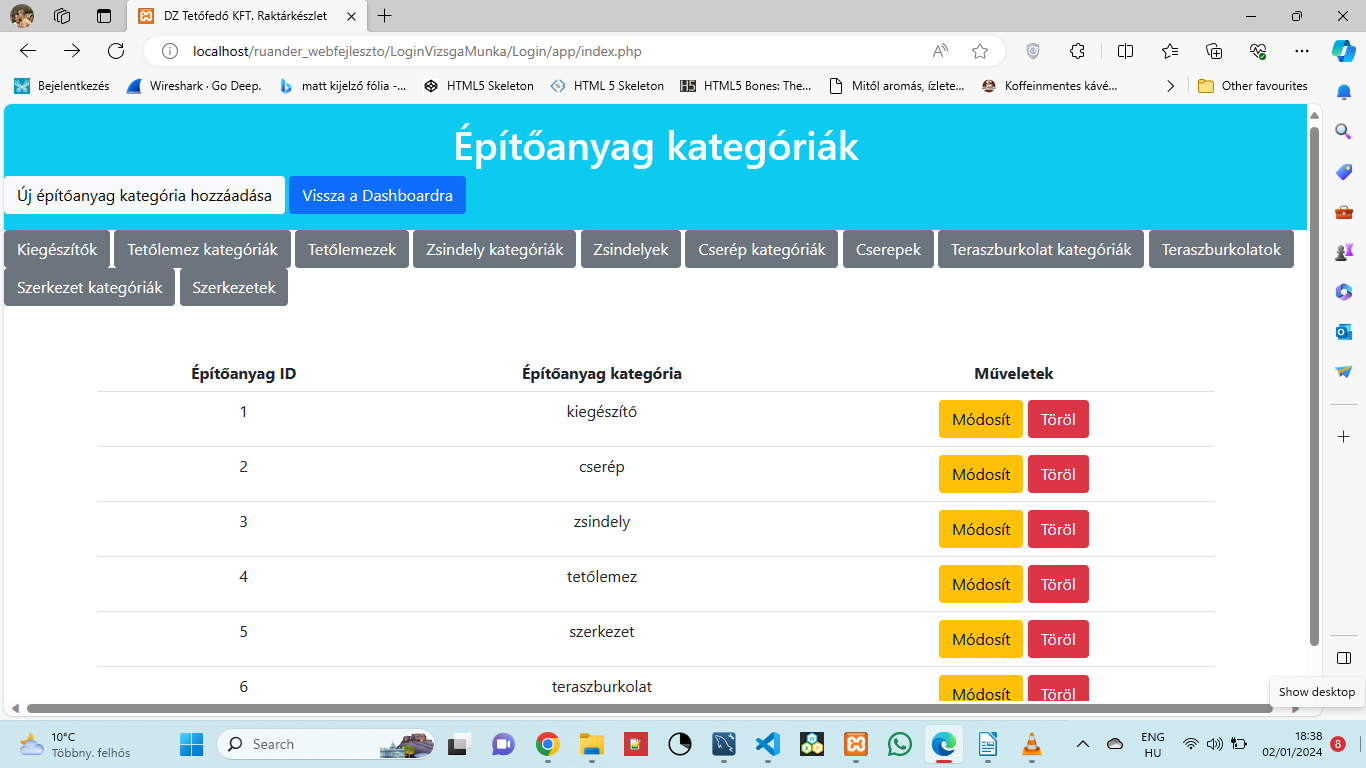


Ez a kezdőoldal. A felhasználó itt tud bejelentkezni. Miután beírja a felhasználónevet és jelszót, a belépés gombra kattintva eljut a dashboardra.

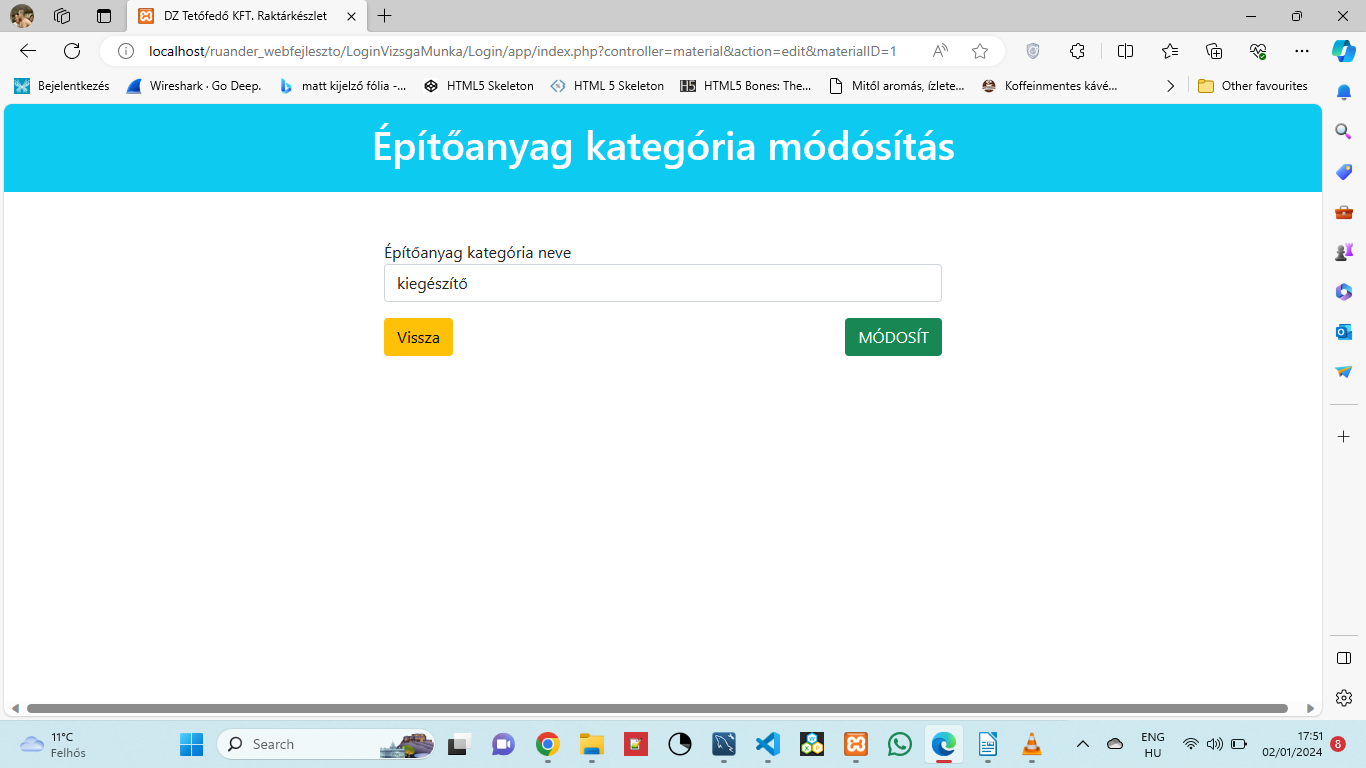


A dashboard oldalon üdvözöljük a felhasználót. Itt lehetősége van arra hogy eljusson az adatbázis kezelő rendszerhez, vagy kilépjen.

Ha az adatbázis kezelését választja megjelenik annak a főoldala. Az építőanyag kategóriák oldal, ami megjeleníti ,kilistázza az összes építőanyag kategóriát valamint a gombok segítségével elérjük az összes adatot amivel dolgozni akarunk.

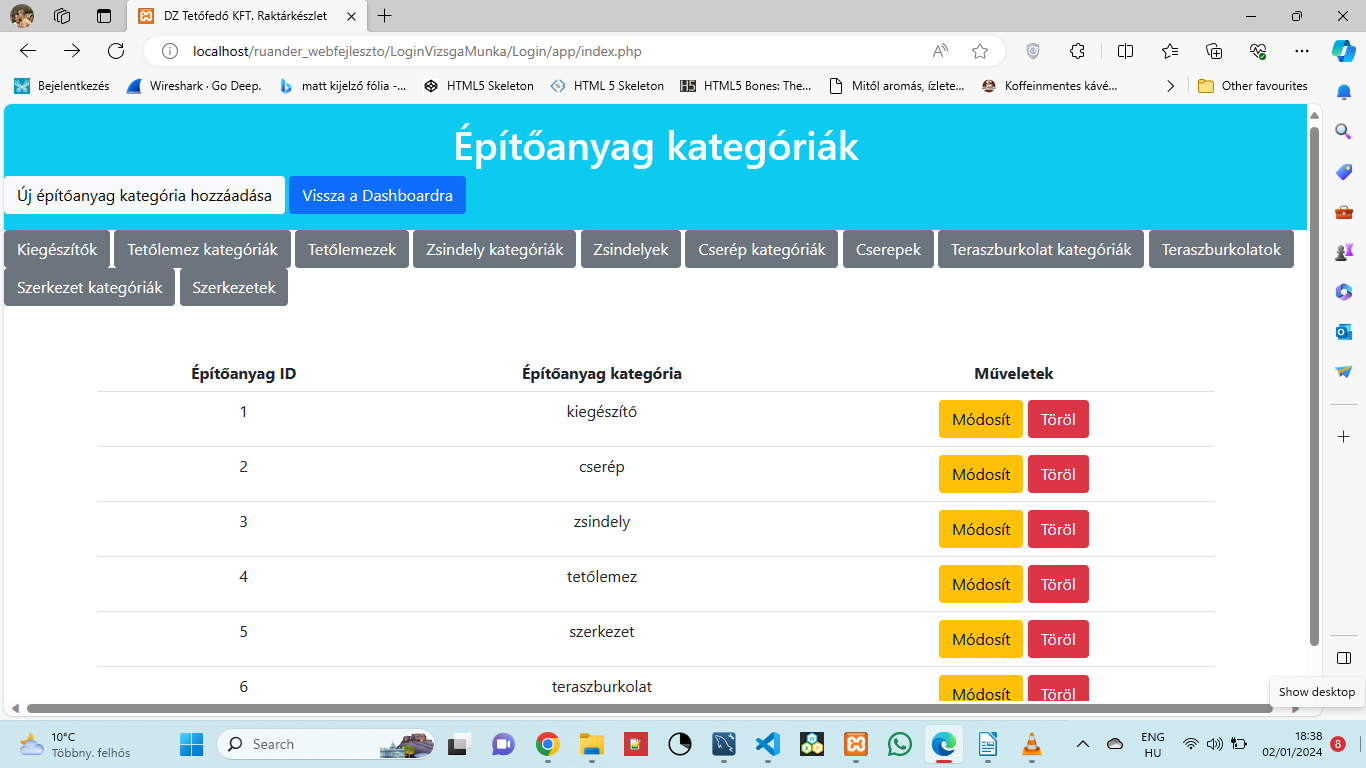


Ha a Módosít gombra kattintunk, akkor megjelenik egy űrlap, ahol meg lehet változtatni a kategória nevét. Azt is látható, hogy éppen mit akarunk módosítani.

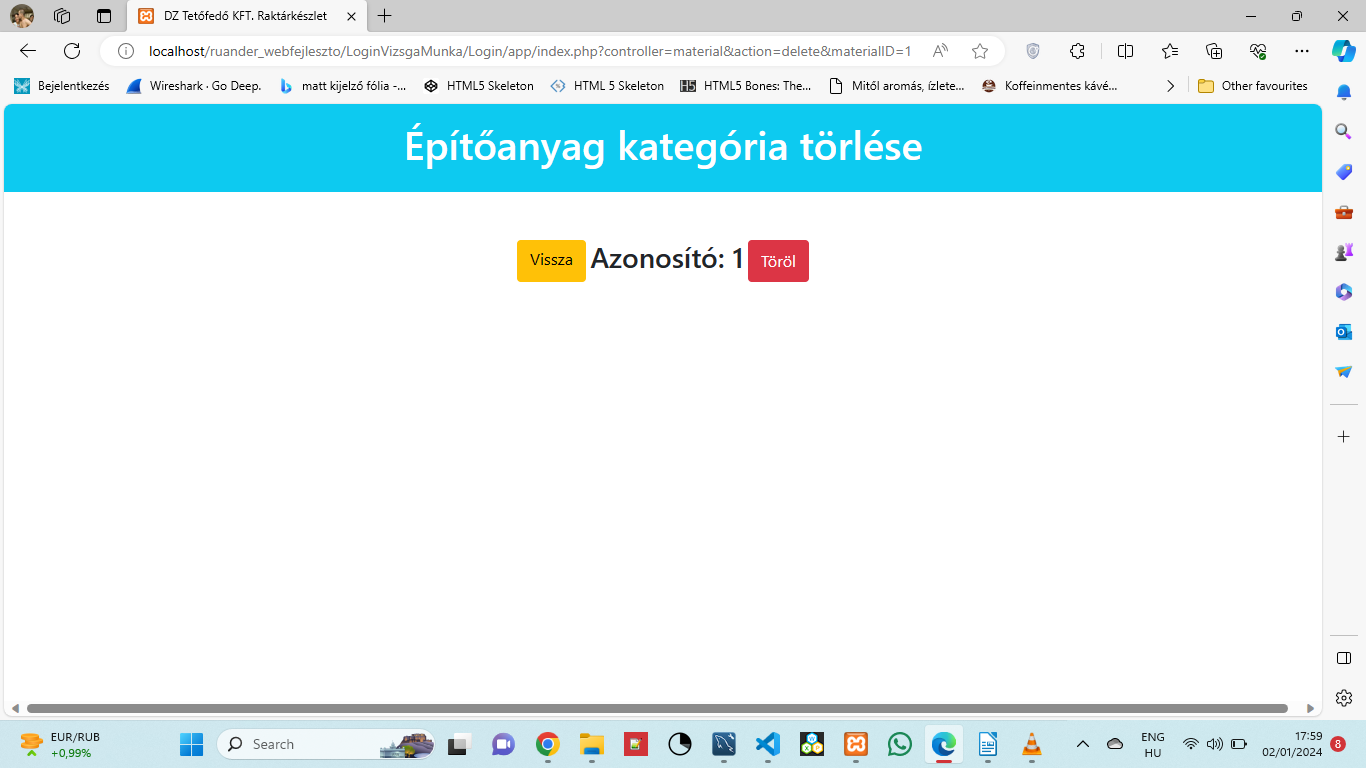


Itt ha a Módosít gombra kattintunk, akkor elmenti az adatbázisban ami frissíti az adatokat erre.

Ha a Vissza gombra kattintunk, akkor visszajutunk az Építőanyag kategóriák oldalra.

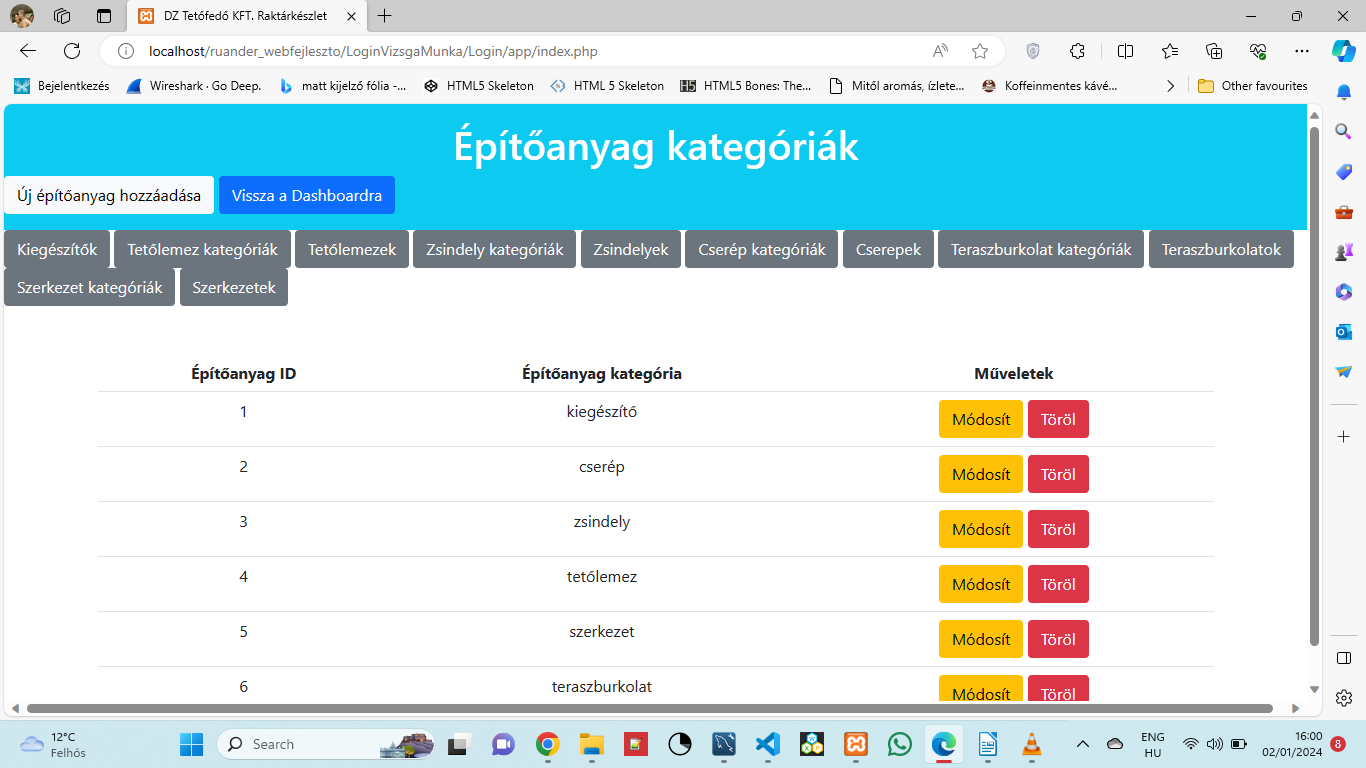


Ha a Töröl gombra kattintunk, akkor betöltődik a törléshez használt űrlap:

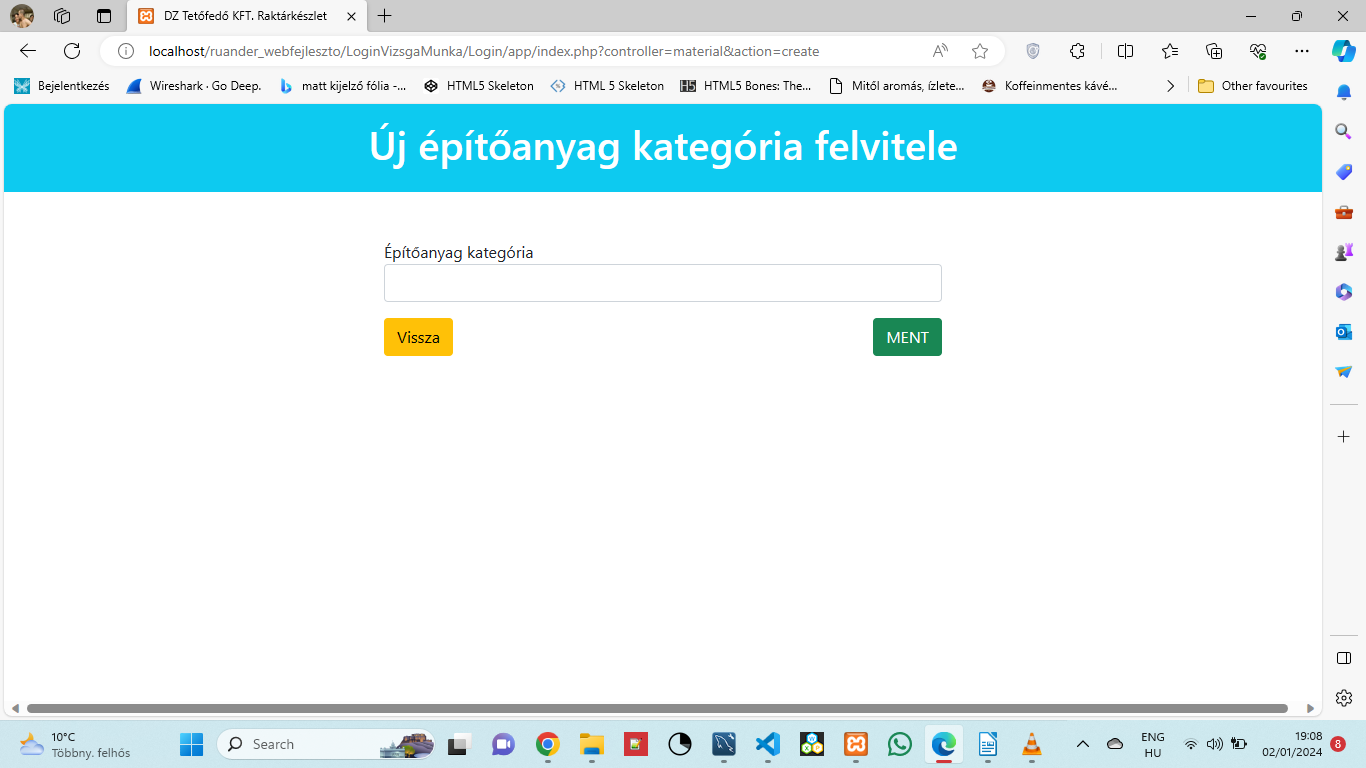


Itt láthatjuk hogy kiirja az azonosítóját is Id alapján.

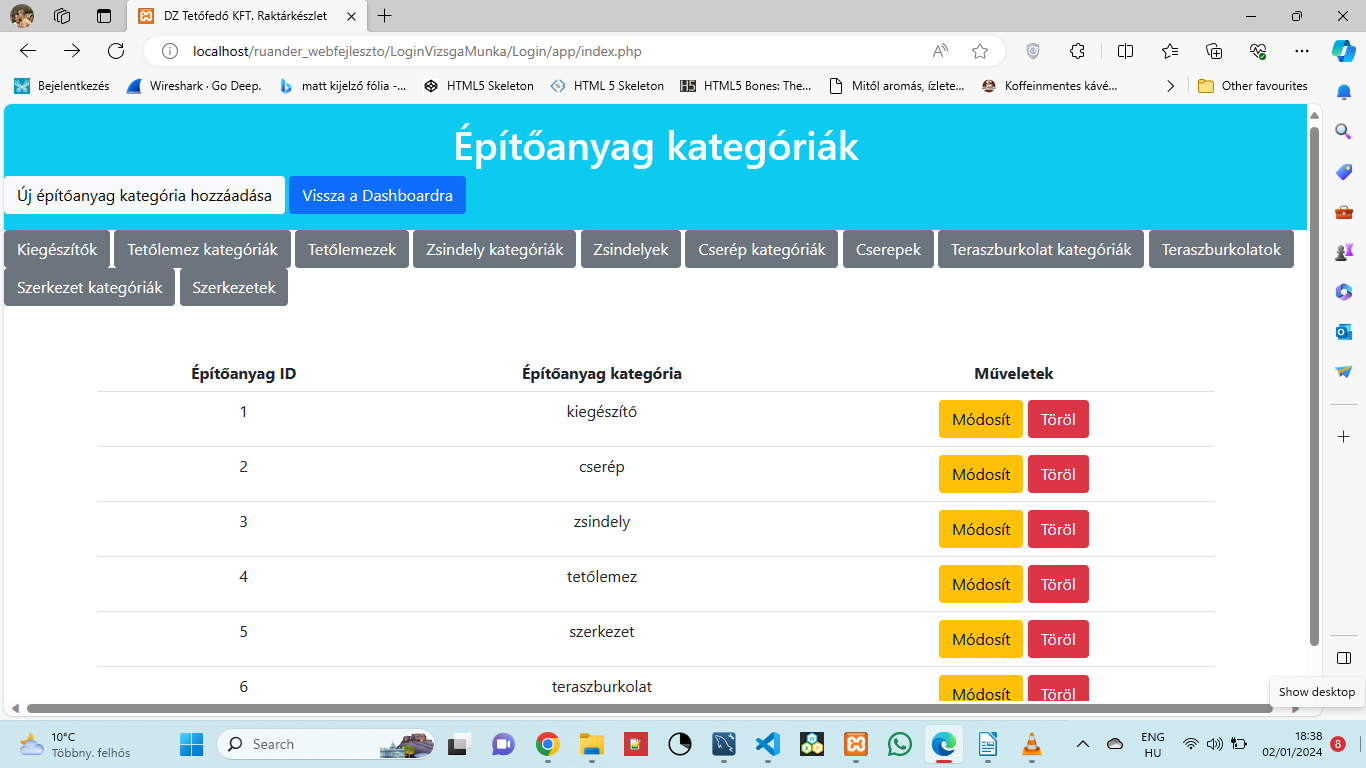
Ha a Vissza gombra kattintunk, akkor a főoldalra kerülünk visssza.



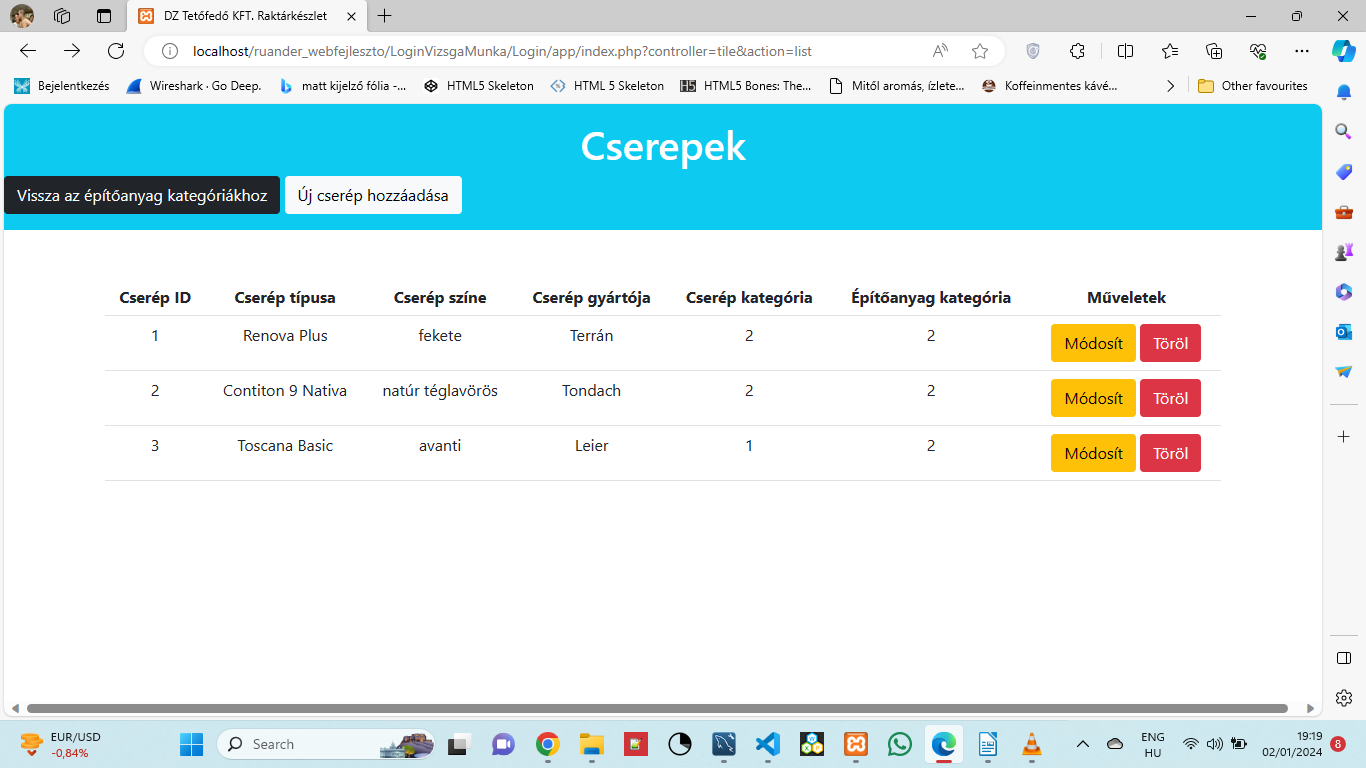
Ha az Új építőanyag kategória gombra kattintunk, akkor megjelenik az erre a célra szolgáló űrlap:



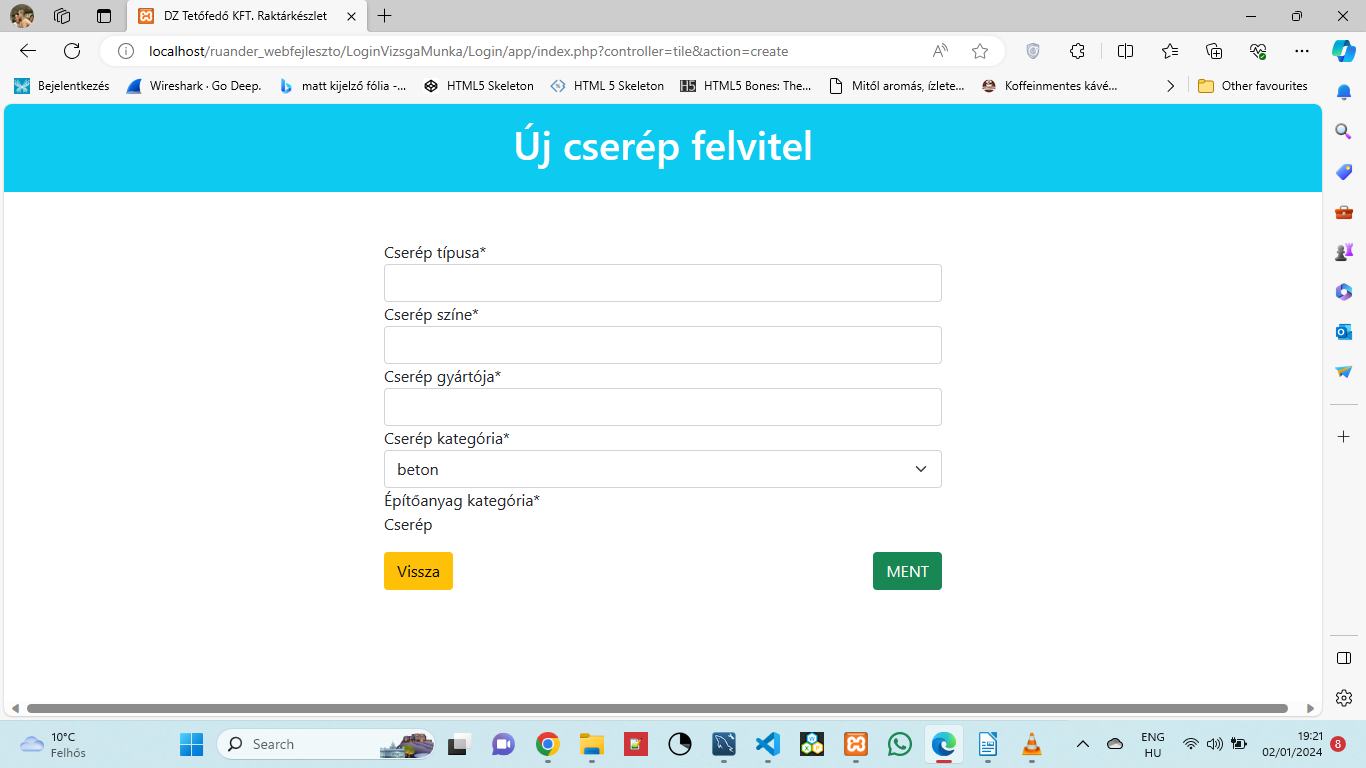
A Ment gombbal elmentjük az adatbázisba, a Vissza gombbal visszajutunk a főoldalra.



Ha pl a Cserepek gombra kattint,unk, akkor megjelenik az oldal ahol a Cserepeket listázzuk és a további műveleteket tudjuk végezni.

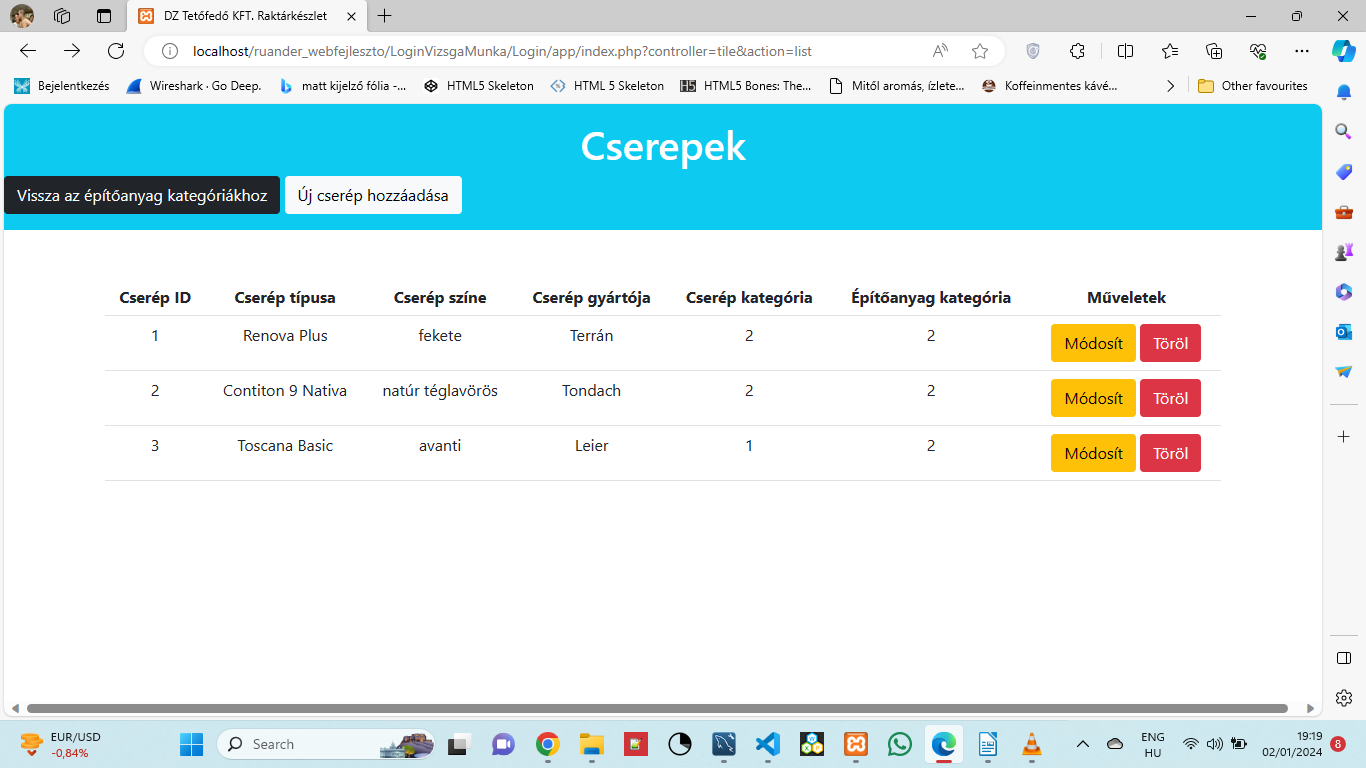


Az Új cserép hozzáadása gommbal megjelenik a hozzá tartozó űrlap:

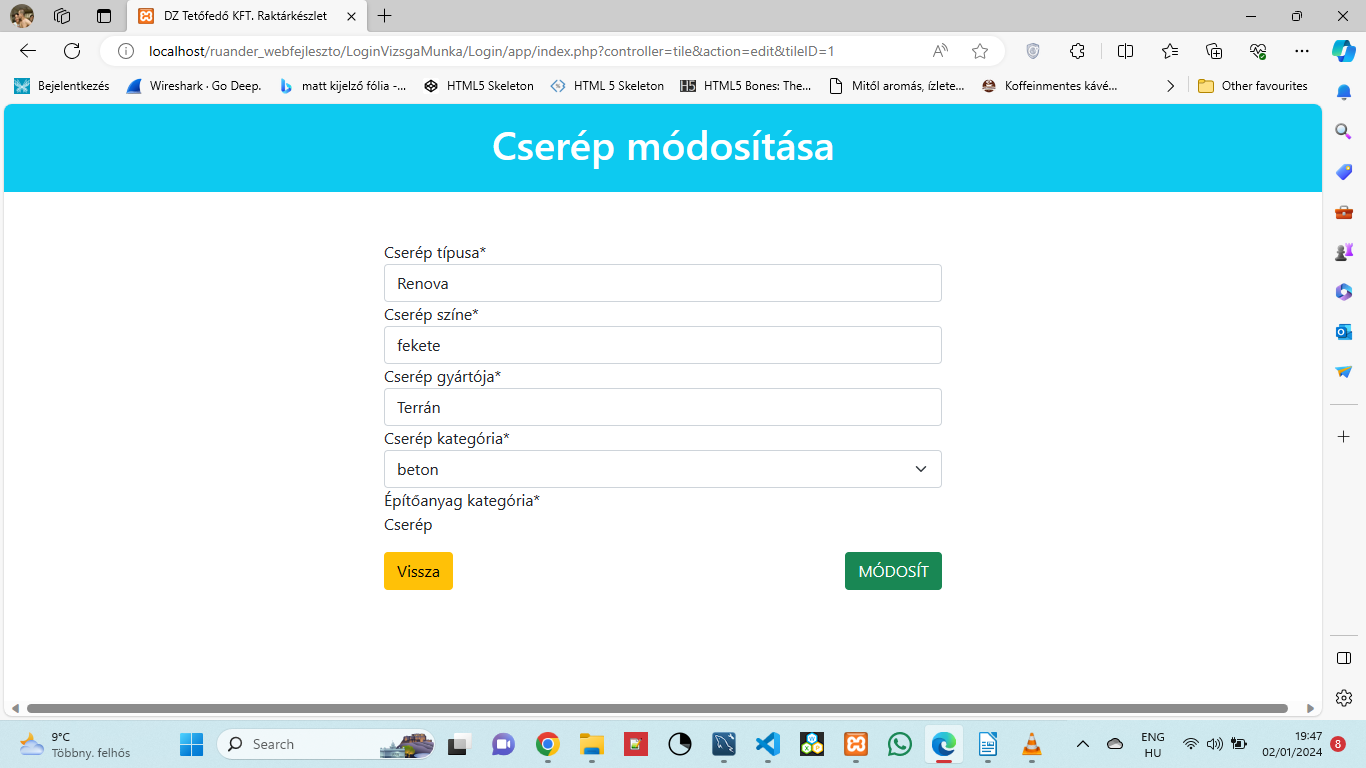


Itt látható, hogy meg kell adni a mentéshez szükséges adatokat, mint a típusa, színe, gyártója. A cserép kategóriát ki tudjuk választani. Ehhez a Cserép kategóriák táblából olvassuk be az adatokat. Az Építőanyag kategória az adott, hiszen cserép.

A Ment gombbal elmentjük az adatbázisba. Ha mégsem akarunk újat felvinni, akkor csak a Vissza gombra kattintunk, és akkor visszakerülünk a Cserepek listázó oldalára.



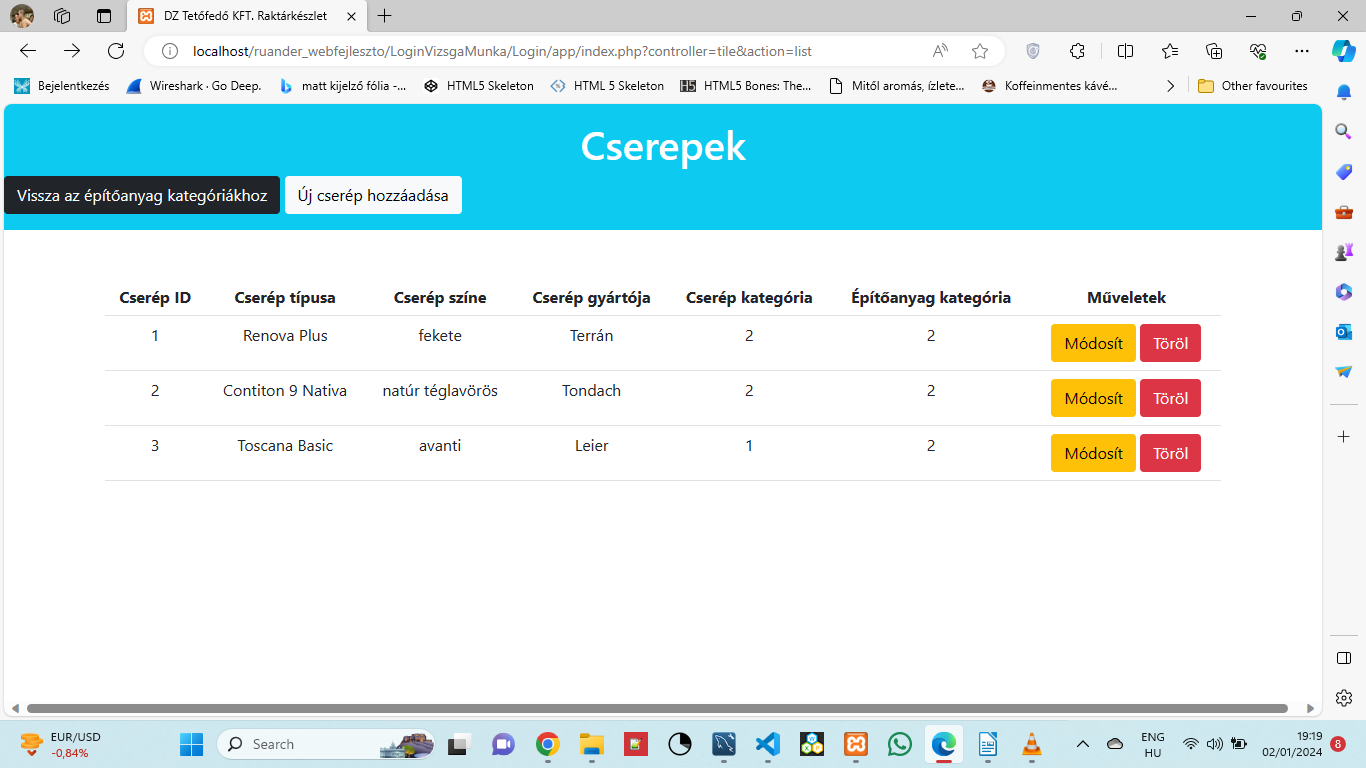
A Módosít gommbal megjelenik a hozzá szükséges űrlap:



Láthatjuk, hogy a Módósítás űrlapon megjelenik az adott Id-hoz tartozó összes adat. Így könnyebben tudjuk módosítani, azt amit szeretnénk.

A Módosít gombbal az adatok frissülnek az adatbázisban.

A Vissza gommbal pedig visszajutunk a Cserepek listázó oldalára.

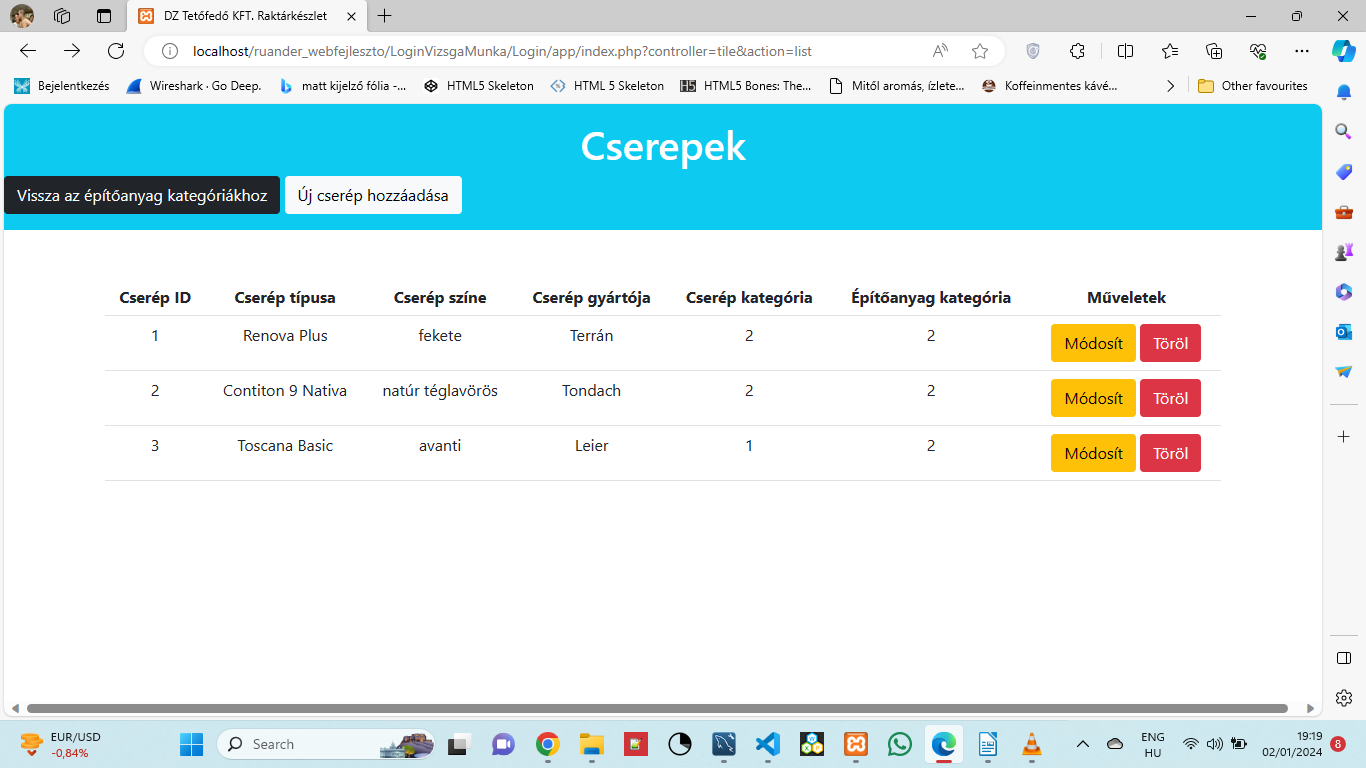


A Töröl gombbal megjelenik a törléshez használt űrlap:

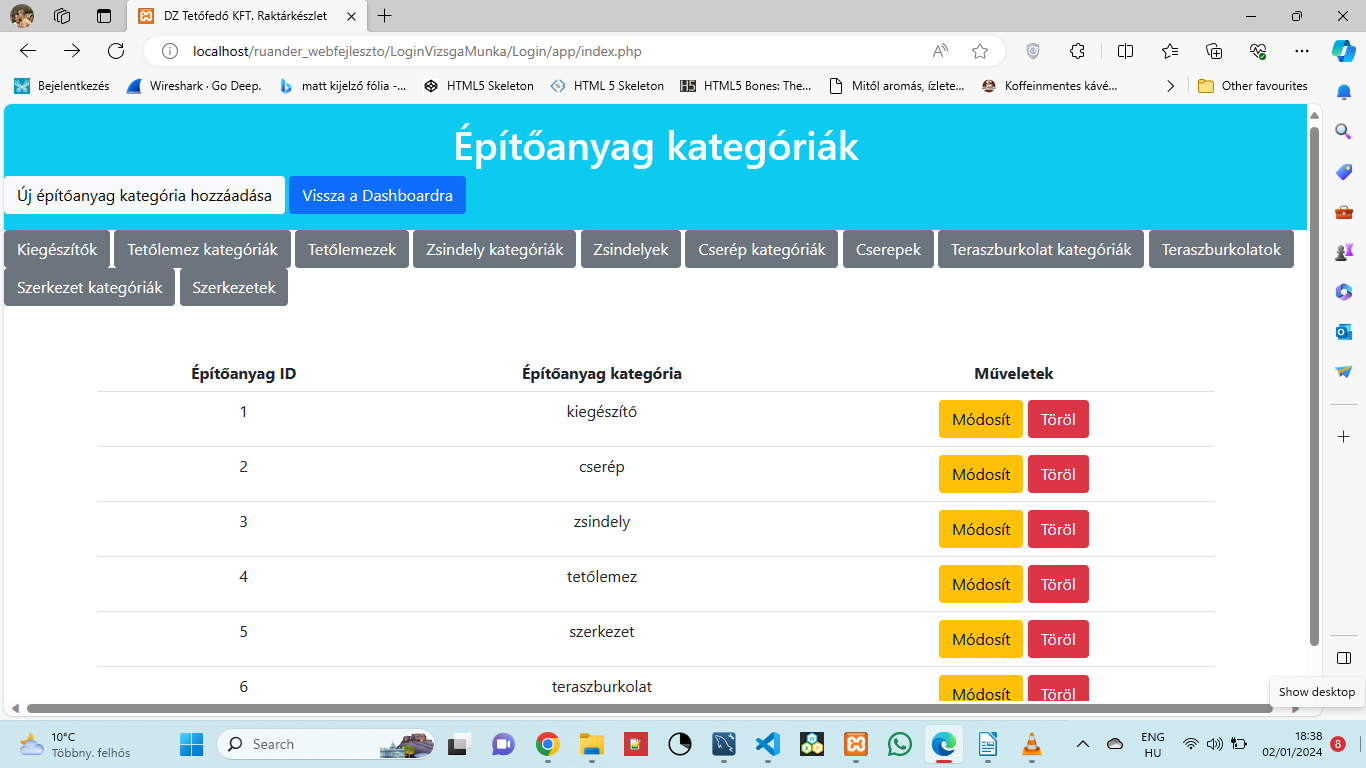


Itt is látható az azonosítója, így tudjuk melyiket töröljük a Töröl gombbal.

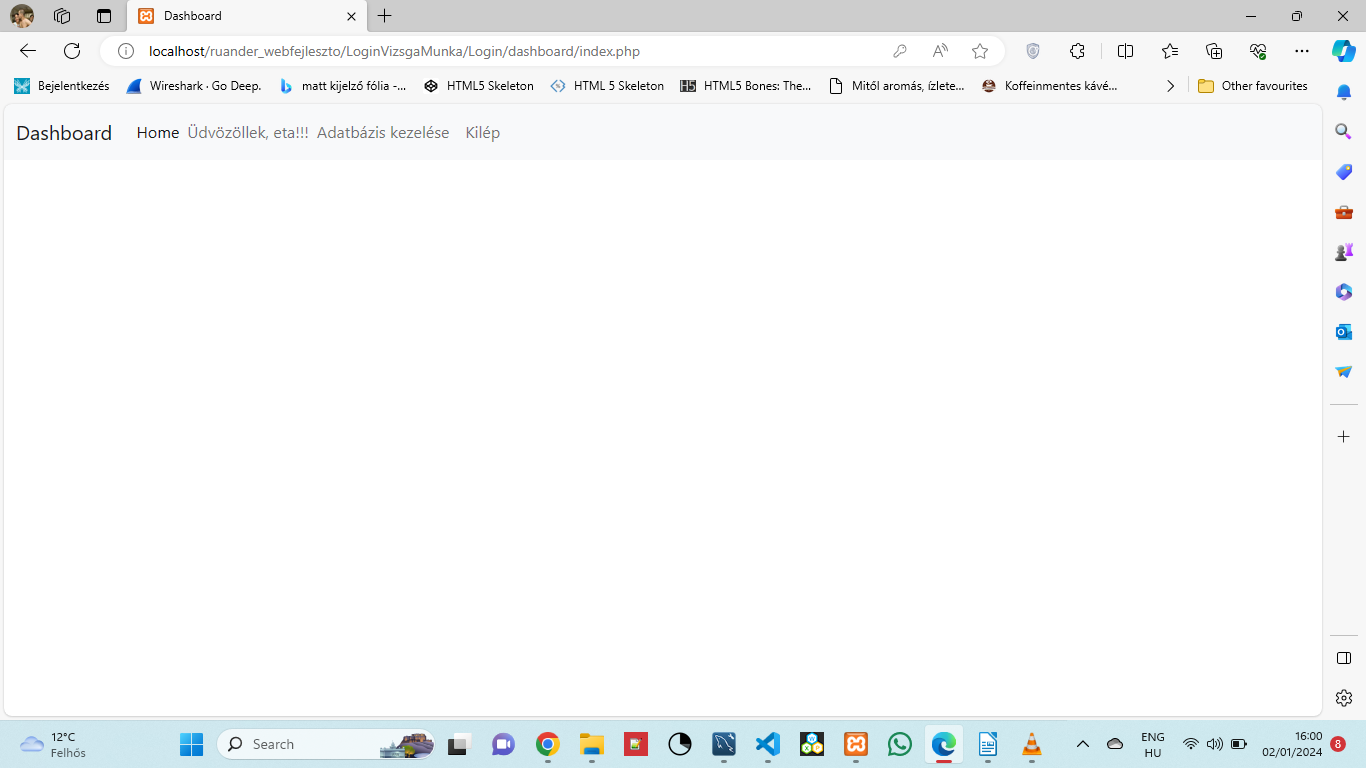
Vissza gommbal pedig visszajutunk a Cserepek listázó oldalára.



Innen vissza tudunk menni a főoldalra a Vissza az építőanyag kategóriák gombbal. Így folytathatjuk a munkát a többi adattal.



Ha nem akarunk tovább dolgozni az adatbázissal, akkor a Vissza a dashboard gombbal visszamehetünk a Dashboard oldalra.



Itt lehetőségünk van visszamenni, ha esetleg valamit elfelejtettünk.

Ha pedig ki szeretnénk lépni, akkor csak kattintsunk a Kilép-re. Ezután a Bejelentkező oldalon vagyunk újra.

