Neumann János Egyetem

GAMF Műszaki és Informatikai Kar

Webprogramozás II. kurzus

Első beadandó feladat dokumentációja

Készítette:

Burka Márk

2022.04.28.

Tartalomjegyzék

A program futtatásához szükséges gépi/nyelvi igények

A program futtatásához a XAMPP programban Apach szerver és MySQL szerver indítása szükséges. A program fájlait a ../xampp/htdocs mappa alá kell elhelyezni, valamint a az sql fáljt importálni a MySQL szerverre.

Bongészőből a localhost/webro2bead1 link beírásával az oldal kezdőképernyője betöltődik, innen érhetjük el a program többi részét.

A program elérhető GitHub-ról is a következő linkről: <https://github.com/BurkaMark/webpro2bead1>

A program felépítése

Strukturálisan két fő részre oszthatóak a fájlok: a frontend oldal, melynek fájlai a public mappa alatt találhatóak, illetve a beakend oldal, melyek az app mappa alatt helyezkednek el. Továbbá mind a public, mind az app, mind a gyökérkönyvtárban található egy-egy .htaccess fájl is, melyek biztosítják, hogy a felhasználók csak a publikus fájlokhoz férhessenek hozzá. A program több része a felhasználók számára nem elérhetőek, csak a program által átadott adatok.

A public mappa tartalma

Ez a mappa tartalmazza az összes olyan fájlt, amelyeket a felhasználó is elérhet. Jelen esetben két fájlt található itt: az index.php és a css mappa alatt található styles.css fájl. A styles.css tartalmazza az összes elérhető oldal stílusát, kinézeti formázását.

A public/index.php fájl

Megnyitásakor meghívja a require.php programfájlt, így indítja el a programot.

Az app mappa tartalma

Ez a mappa tartalmazza az összes olyan fájlt, mely a program futásához, az adatbázishoz való kapcsolódáshoz, vagy épp az adatok megjelenítéséhez szükséges. Hat mappát (config, controllers, helpers, libraries, models, views) és a require.php programfájlt tartalmazza.

Az app/require.php fájl

Meghívja a három alapfájlt (Core.php, Controller.php és Database.php), a konfigurációs fájlt (config.php) és egy segéd programfájlt (session\_helper.php), valamint inicializálja a Core osztályt.

A config mappa tartalma

Itt található a configurációs fájl, a config.php.

Az app/config/config.php fájl

Globális változóknak ad meg értéket. Az első négy változó (DB\_HOST, DB\_USER, DB\_PASS és DB\_NAME) az adatbázishoz való csatlakozáshoz szükséges adatokat tárolják. Az APPROOT az aplikáció gyökérkönyvtárát rögzíti, az URLROOT az oldalak alap url-jét tárolja, míg a SITENAME a honlap elnevezését.

A helpers mappa tartalma

Ebben a mappában található a session\_helper.php fájl, amely egy kiegészítő, segédfájl, melyet a program bármely részéből elérhetünk.

Az app/helpers/session\_helper.php fájl

A session\_start() funkció elindít egy munkamenetet. Az isLoggedIn() funkció megvizsgálja, hogy $\_SESSION szuper globális változó user\_id mezője rendelkezik-e értékkel, azaz a felhasználó bejelentkezett-e. Az eredménynek megfelelően igaz, vagy hamis értékkel tér vissza.

A libraries mappa tartalma

Ebben a mappában található a három alapprogram, a Controller.php, a Core.php és a Database.php, melyek felelősek a program működésének irányításáért és az adatbázishoz való csatlakozásáért.

Az app/libraries/Controller.php fájl

A model() funkció hívja meg az input-ként megadott modelt az app/models mappa alól, majd inicializálja azt. A view() funkció hívja meg az input-ként megadott nézeti fájlt az app/views mappából, valamint átadja neki az input-ban megadott $data sort.

Az app/libraries/Core.php fájl

A konstruktor először eltárolja az url-t egy változóban a getUrl() funkció segítségével.

A getUrl() funkció a $\_GET szuper globális változó url mezőjét vizsgálja, hogy tartlmaz-e értéket. Amennyiben igen, először jobbról levágja a /-t, eltávolítja az url-be nem illő karaktereket, végül szétbontja szekciókra a /-ként. Az így visszakapott értéket átadja a konstruktorban a $url változónak.

Ezt követően a konstruktor ellenőrzi, hogy a $url első mezője tartalmaz-e értéket. Amennyiben igen, megnézi, hogy létezik-e ilyen elenevezésű kontroller programfájl. Ha igen, akkor azt állítja be, mint jelenlegi kontroller, amennyiben nem, úgy marad az alapérték, azaz a Pages, majd törli a $url első elemét.

Ugyanezt megteszi a $url második elemével, csak itt már funkciót keres a már megadott kontroller fájlban. Ha nem talál, akkor az alapértelmezett index értéket tárolja és törli a második elemet is.

Végül ellenőrzik a konstruktor, hogy maradt-e ezeken kívül még eleme a $url-nek. Ha igen, akkor azokat paraméterenként adja át a fent már megadott kontroller funkciónak.

Az app/libraries/Database.php fájl

A konstruktor összegyűjti a csatlakozáshoz szükséges adatokat, majd létrehozza a PDO kapcsolatot a program és az adatbázis között.

A query() funkció előkészíti az input-ként megkapott SQL utasítást az adatbázisnak vló átadására.

A bind() funkció összekapcsolja az input-ban megkapot paramétert, az inputban megkapot értékkel.

Az execute() funkció végrehajtja az előkészített SQL parancsot.

A resultSet() funkció összeszedi az össze olyan rekordot az adatbázisból, amely megfelel az előkészített SQL parancsban meghatározott feltételeknek.

A single() funkció megkeresi az első olyan rekordot az adatbázisban, ami megfelel az előkészített SQL parancsban meghatározott feltételeknek.

A rowCount() funkció megszámolja a rekordok számát az adatbázisban, amelyek megfelelnek az előkészített SQL parancsban meghatározott feltételeknek.

A controllers mappa tartalma

Ebben a mappában találhatóak a kontroller fájlok. Közös mindegyikben az, hogy mind a Controller.php fájl kibővítése, annak örökösei.

Az app/contollers/Mnb.php fájl

Az index() funkció üres értékekkel inicializálja a nézeti mnb fájlt.

A GetExchangeRates() funkció először kiszűri a bemeneti értékeket minden felesleges karaktertől, majd átensek egy validációs folyamaton, melyben egyrészt ellenőrzésre kerül, hogy kitöltésre kerültek-e, a megfelelő formátumban adta meg a felhasználó az értékekekt, illetve a megadott két valuta szerepel-e az adatbázisban (ValidateCurrencies() funkció). Amennyiben valamely fázisában hibára fut a validáció, a felhasználó hibaüzenetet kap.

Amennyiben hibamentesen lefut a validáció, a program kapcsolódik a MNB soap szolgáltatásához. A visszakapott string-et xml-ben tároljuk, majd egys két dimenziós mátrixba konvertáljuk át.

Az így megkapott mátrixból kikeressük a felhasználó által bekért valutákhoz tartozó adatokat és végül kiszámoljuk a két valuta közti átváltási értékeket mindkét irányba és átadjuk a nézeti fájlnak az adatokat.

A ValidateCurriencies() funkció csatlakozik az MNB soap szolgáltatásához. A visszakapott értéket egy sorba rendezi, majd megvizsgálja, hogy az input-ként megadott valuta szerepel-e a sorban. Az eredménynek megfelelően igaz, vagy hamis értékkel tér vissza.

Az app/controllers/Pages.php fájl

Az index() funkció betölti az index nézeti fájlt.

Az app/controllers/Posts.php fájl

A konstruktor meghívja a Post modelt.

Az index() funkció betölti az összes bejegyzés rekordot az adatbázisból és átadja a post/index nézeti programnak.

A create() funkció először ellenőrzi, hogy a felhasználó bejelentkezett-e már. Csak bejelentkezett felhasználók készíthetnek új bejegyzéseket.

Ezt követően eltávolítja a fölösleges karaktereket a bemeneti mezőkből és ellenőrzi, hogy minden mező kitöltésre került-e.

Amennyiben minden rendben, átadjuk a bejegyzést a Post model addPost funkciójának, az adatbázisban és megjelenítésre kerül a posts/create nézeti programban.

Az update() funkció a create() funkcióhoz hasonlóan ellenőrzi, hogy a felhasználó bejelentkezett-e, de azt is, hogy ugyanaz a felhasználó-e, mint aki létrehozta a módosítandó bejegyzést. Csak a bejegyzést eredetileg létrehozó módosíthatja annak tartalmát.

Ezt követően ugyanúgy átesik az ellenőrzéseken, mint a create() funkcióban, kiegészítve azzal, hogy ellenőrzi, hogy legalább a bejegyzés címe, vagy tartalma változott-e, illetve az input-ban megadott id-nak megfelelő bejegyzést választja ki.

Végül, ha minden rendben van, megkapja a módosított bejegyzést a Post model updatePost funkciója és megjelenítésre kerül a posts/update nézeti programmal.

A delete() funkció szintén ellenőrzi, hogy ugyanaz a felhasználó legyen bejelentkezve, mint aki létrehozta a bejegyzést. Majd megkeresve az input-ban megadott id-nak megfelelő bejegyzést átadja a Post model deletePost funkciójának.

Az app/controllers/Trails.php fájl

A konstruktor meghívja a Trail modelt.

Az index() funkció üres értékekkel inicializálja a nézeti trail fájlt.