Blog weboldal

készítette: Bogáncs Dániel Sándor

Tartalom

Előszó

Vannak emberek, akik szeretnek időnként elgondolkodni az élet nagy dolgain, és ezeket a gondolatokat le is írják. Némelyek papírra, mások az internet eldugott weboldalain. Sok blogolási lehetőséget nyújtó weboldal létezik, de nem tökéletesek. Nem áll rendelkezése a bejegyzések kategorizálásának és felcímkézésének széles tárháza, nincs részletesen skálázható hozzáférési rendszer, amit a felhasználó személyekre lebontva állíthat be. Ezt a világban tátongó űrt hivatott betölteni a weboldal, amit ez a leírás dokumentál.

A weboldal tulajdonosközpontú, ami annyit tesz, hogy a naplót egyetlen személy írhatja, a többi csak olvasó. Ez egy kis munkával továbbfejleszthető közösségi naplózássá, de ez még nem valósult meg, mivel a weboldalnak nem ez az elsőrendű célja, csupán a készítő saját személyes gondolatait hivatott tárolni. Ez a naplókezelő Bogáncs Dániel Sándor saját felhasználásra készült önszorgalomból, hogy a PHP szerveroldali programozást alapszinten elsajátítsa, továbbá hogy a mérnöki tervezést és dokumentálást gyakorolja.

Röviden áttekintve a következő főbb funkciókat valósítja meg az oldal:

- Van regisztrációs lehetőség. (A blogbejegyzésekhez tartozó hozzáférések regisztrációhoz kötöttek.)
- A szerkesztés nélkülözhetetlen eleme a virtuális naplónak. Funkcióiban gazdag, címkézések, témaválasztások, részletes hozzáférési szint beállítások állnak itt rendelkezésre.
- A legtöbb hangsúly arra került, hogy az olvasók számára a lehető legáttekinthetőbben legyenek a bejegyzések prezentálva. Sokféle szűrési lehetőség áll az olvasók rendelkezésére. Áttekinthető a teljes tartalom, az egyes bejegyzéscímekre kattintva további részleteket olvashatunk róluk.

Rövid áttekintés után egy szótár következik, hogy a dokumentációban használt kifejezések egyértelműek legyenek, utána a weboldallal szemben állított követelmények listája található, majd a főbb modulok részletes kifejtésére kerül sor az egyes fejezetekben.

Szótár

Állapot (regisztrációé)	A korábban megadott adatok módosítás esetén nem vesznek el. Ha a felhasználó módosítja az adatait, új regisztráció-variáns jön létre. Ha a felhasználó fél évig nem lép be, a weboldal tulajdonosával újra kell aktiváltatnia a regisztrációját. Ezeknek megfelelően a regisztrációknak állapotaik vannak, például: aktív, inaktív, módosított… stb..
Bejegyzés	A napló tartalmának alap logikai egysége. Ez egy hosszabb szöveg, időnként szakaszokra van bontva. Továbbá kiegészítő információk is tartoznak hozzá (fejléc).
Címke	A címke a bejegyzés tartalmának jellemzésére szolgál, de szűkebb értelemben, mint a téma meghatározása. Címke sokszor csak fogalmak említését jelenti. Példa: Család, szórakozás, érzések… stb..
Fejléc	Vannak a bejegyzés egészére vonatkozó adatok, mint cím, létrehozás dátuma, sorszám… stb.. Így különböztetjük meg azoktól az adatoktól, amik a bejegyzés szöveges tartalmának kisebb részeire vonatkoznak.
Felhasználó	Minden olyan személy, aki a weboldallal bármilyen interakcióba kerül.
Hozzáférési szint	Két értelemben beszélünk a hozzáférés szintjéről. A bejegyzéseknek és azok szövegrészeinek hozzáférési szintje egy korlát, ami tiltja a hozzáférést a kisebb szintű regisztrációval rendelkező felhasználóknak. A regisztrációhoz tartozó hozzáférési szint viszont egy jog hozzáférni a nem nagyobb hozzáférési szintű bejegyzésekhez.
Hozzászólás	A felhasználóknak lehetőségük van hozzászólni a naplóhoz. A küldött szöveg nem publikus, nincs komment modul. Lényegében egy egyszerű, az írónak szóló üzenetről van szó.
Író	Bejegyzés létrehozásának és szerkesztésének, illetve az összes korábbi bejegyzés olvasásának jogával rendelkező regisztrációjú felhasználó. Ameddig a weboldal kibővítésre nem kerül, ez ekvivalens a weboldal tulajdonosával.
Lekérdezés	Felhasználói szinten lekérdezésnek számít egy bejegyzés megnyitása, aminek a sorszáma naplózásra is kerül. Programozói szinten a hagyományos adatbázis-lekérdezéseket értjük alatta.
Napló	A weboldal tulajdonosa a napjainkban használatos ’blog’ szó helyett a ’naplót’ szót szereti használni a saját oldalára. A weboldal címében is ez szerepel.
Olvasó	Bejegyzés létrehozásának és szerkesztésének a jogával nem rendelkező regisztrációjú felhasználó. Az olvasó csak olvasni tudja a bejegyzéseket, hozzáférési szintjétől függően többet vagy kevesebbet.
Regisztráció	Általános értelemben regisztrációnak mondjuk a felhasználónév és jelszó párost, hozzájuk rendelve egy hozzáférési szintet. A regisztráció nem ekvivalens a felhasználóval, egy felhasználónak több regisztrációja is lehet. Technikai szempontból regisztráció-variánsok összességéről beszélhetünk, melyeknek az ’első id’ attribútuma megegyezik.

Regisztráció variánsai	A felhasználó által adott regisztráción belül bármikor megadott adatok összessége. Adatmódosítás esetén a régi adatok mentve maradnak, és az újaknak létrejön az adatbázisban egy új rekord. Minden rekord egy variánsnak számít, ami ugyanazon regisztrációban jött létre. Közös jellemzőjük, hogy mindegyik hordozza ugyanazt az azonosítót, ami regisztráláskor lett generálva.
Robotcheck	Ez a megnevezése a weboldal azon részének, ami ellenőrzi, hogy a regisztráló felhasználó nem robot. Az adatbázisból kiválaszt egy kérdést, aminek a helyes megválaszolásával lehet csak regisztrációt létrehozni.
Szakaszok	A bejegyzés szövegének egyes részei privátabbak lehetnek a többinél. Ezért hozzáférési szint nem csak a bejegyzésnek adható, hanem az egyes szövegrészeinek is külön-külön. Egy ilyen saját hozzáférési szinttel rendelkező szövegrészt nevezünk szakasznak. Az adatbázisban 'szövegek' néven szerepel, de szakaszokat kell érteni alatta. (Ez a szakasz nem ekvivalens az író által a bejegyzések szöveges tartalmában elkülönített szakaszokkal.)
Téma	A téma a címkéknél tágabb értelemben határozza meg a bejegyzés tartalmát, a bejegyzés egészére általánosan érvényes jellemzés. Példák: Probléma, történet, vélemény, értelmezés… stb..
Weboldal tulajdonosa	Bogáncs Dániel Sándor, a weboldal készítője és karbantartója, maga az író.

Követelmények

A weboldalnak sok követelménynek kellett megfelelnie. A weboldal tulajdonosának konkrét elképzelési voltak, hogy milyen funkciókra van szükség. Most felsorolásra kerül minden követelmény, amik életre hívták a programozói megoldások többségét. Minden követelmény kap egy azonosítót és egy megnevezést, amivel hivatkozni lehet rá.

BJ00	Bejelentkezés	Regisztrációval rendelkező felhasználónak lehetőséget kell biztosítani a bejelentkezésre.
BJ01	Bejelentkezési adatok	Bejelentkezni felhasználónév és jelszó megadásával lehetséges.
BJ02	Jelszó-emlékeztető	Amennyiben a felhasználó elfelejtette a jelszavát, felhasználónév és email cím megadásával jelszóemlékezetőt kaphat az email címére.
BJ03	Inaktivitás	Amennyiben egy felhasználó fél évig nem lép be, a regisztrációja váljon inaktívvá. A weboldal tulajdonosával kell aktiváltatni a régi hozzáférési szint visszaszerzéséhez. (Erre azért van szükség, mert nem biztos, hogy fél évvel később ugyanazok a hozzáférési jogok lesznek érvényesek. Mikor a felhasználó regisztrációja újra aktív lesz, lehet a réginél több, de akár kevesebb joga is.)
FI00	Filter	A felhasználónak legyen lehetősége szűrési feltételeket megadni, hogy könnyebben tudjon keresgélni a bejegyzéscímek között.
FI01	Feltételek	Lehessen keresni egy adott címke vagy egy téma kiválasztásával (vagy akár mindkettővel egyszerre), továbbá lehessen szűrni, hogy vagy csak az ajánlott bejegyzéseket jelenítse meg vagy az összeset.
FI02	Feltételek megjegyzése	A filterben beállított feltételek megmaradnak mindaddig, míg a felhasználó át nem állítja őket, vagy rá nem kattint az 'Összes' gombra.
FI03	Feltételek fejlécből	A bejegyzések fejléceiben felsorolt címkékre lehessen kattintani. Az oldal újratöltődésével állítódjon be szűrőfeltételnek a kiválasztott címke, és ennek megfelelően szűrje meg a címlistát.
HS00	Felhasználó hozzáférési szint	Minden a weboldalra látogató felhasználónak kell legyen hozzáférési szintje.
HS01	Bejegyzés hozzáférési szintje	Minden bejegyzésnek lennie kell saját hozzáférési szintjének. Ez a legkisebb hozzáférési szintű szakasz szintje. Erre szükség van a HS05 kódú követelménymiatt.
HS02	Szakasz hozzáférési szintje	Egy bejegyzés szövegének egy részlete lehet privátabb a többinél. Ez a szakasz kaphat magasabb hozzáférési szintet. Ha a szöveg közepén van egy szövegrész, ami privátabb megjegyzést tartalmaz a többinél, akkor a teljes szöveget három szakaszra lehet bontani, az elsőnek és a harmadiknak alacsonyabb, a másodiknak magasabb hozzáférési szintet adva.
HS03	Felhasználó hozzáférési joga	A felhasználók csak azokat a szakaszait láthatják a bejegyzéseknek, amiknek a hozzáférési szintje nem nagyobb az övéknél.
HS04	Cenzúra	Abban az esetben, ha egy felhasználó túl alacsony hozzáférési szinttel rendelkezik egy szakasz elolvasásához, annak a helyén '[…]' jelölést találjon.
HS05	Elrejtés	Amennyiben egy bejegyzés minden szakasza magasabb hozzáférési szinttel rendelkezik, mint a felhasználó, akkor azt teljesen el kell rejteni

		előle.
HS06	Tulajdonos hozzáférési joga	A weboldal tulajdonosának legyen az elérhető legmagasabb hozzáférési szintje.
HS07	Írás	Új bejegyzést létrehozni vagy régit módosítani/szerkeszteni csak a weboldal tulajdonosának legyen joga.
IN00	Információk	Legyen külön oldal fenntartva, ami tájékoztatást ad a felhasználóknak az egyes témakörök, címkék és hozzáférési szintek konkrét jelentéséről. Ez az oldal világosítsa fel a felhasználót az ő jelenlegi saját hozzáférési szintjéről is.
IN01	Köszöntő	A főoldalon legyen egy rövid köszöntő, ami eligazítást ad a weboldalra tévedő felhasználóknak.
IN02	Jelmagyarázat	A főoldalon szerepeljen egy rövid jelmagyarázat, ami lényegre törően ismerteti a bejegyzés címein használt jelöléseket (vastagítás, csillagozás, témaszínek és elnevezések).
KO00	Kommentálási lehetőség	Az olvasónak legyen lehetősége kommentálni a naplót.
KO01	Időkorlát	Egy olvasó egy nap csak egyszer szólhasson hozzá a naplóhoz. Ha megtette, ne jelenjen meg számára az ezzel kapcsolatos modul.
KO02	Terjedelmi korlát	Egy hozzászólás nem lehet több ezer karakternél.
NA00	Napló olvasásának lehetősége	A meglévő bejegyzésekhez legyen egy modul, amivel azok előhívhatóak és olvashatóak.
NA01	Címlista	Legyen egy lista a bejegyzések címeivel az oldal szélén, ahonnan gyorsan elérhetőek a weboldal bejegyzései.
NA02	Szabványos bejegyzéscím	A szabványos bejegyzéscím két részből áll: a bejegyzés címe, aztán a megírásának a dátuma (YYYY-mm-dd formátumban) szóközzel elválasztva. Példa: Egy jó nap 2000-01-23
NA03	Címek	A címlistában szereplő címek szabványos bejegyzéscímek. (NA02)
NA04	Címek színezése	Minden szabványos bejegyzéscímet meg kell formázni. Legyen a témának megfelelő színe, és legyen vastagított, ha ajánlott bejegyzés, szerepeljen csillag az elején, ha leellenőrizetlen. Amennyiben a felhasználó hozzáférési szintje alacsony, ne láthassa a sorszámot.
NA05	Címlista felépítése	A címlistában szereplő bejegyzéscímek fentről lefelé a legfrissebbtől a legrégebbiig legyenek felsorolva és sorszámozva. Fel kell tüntetni a naplónak az éveit, és mindegyik alá fel kell sorolni az abban az évben készült bejegyzéseket. (A napló kezdetekor a bejegyzések a napló első évében készültek.) Továbbá vannak fejezetek, ezek az évekkel azonos struktúrában jelenjenek meg a listában. Példa: 2. Év 4. fejezet 6. Még egy jó nap 2002-06-01 5. Egy jó nap 2002-05-01 3. fejezet 4. Állatkert 2002-04-01 1. Év 2. fejezet 3.Rossz nap 2001-03-01

1. fejezet
    2. Jól vagyok 2001-02-01
    1. Beteg vagyok 2001-01-01

NA06	Elő-bejegyzések	Az író készített bejegyzéseket, amik a naplóírás előtti időszakot mutatják be felvezetésként. Ezek több korábbi évből származnak, de együttesen a napló nulladik évében, nulladik fejezetében vannak számon tartva. A számozásuk a többitől eltérő 0.1, 0.2, 0.3 sémát követi. A weboldal ezt is legyen képes kezelni.
NA07	Hivatkozások elhelyezése	Amennyiben a megjelenített bejegyzés szövegében szabványos bejegyzéscím található, azt témájának megfelelő színnel, és az adott bejegyzés tartalmára mutató hivatkozással kell ellátni.
NA08	Bejegyzés felépítése	Legyen két külön panel: egyik a bejegyzés szövegének, másik külön a fejlécnek. A szöveg paneljében csak a lényegi szöveg illetve a bejegyzés címe szerepeljen. Minden egyéb adat egy fejléc panelben foglaljon helyet. A megjelenítendő adatok a következők: Bejegyzés címe és rövid tartalma, a megjelölt címkék, és megemlített más bejegyzések címei formázva, hivatkozással ellátva. A címkék és a bejegyzéscímek egymást kövessék egy felsorolásban, külön-külön mindkettő legyen rendezve.
RE00	Regisztrációs lehetőség	A felhasználóknak legyen lehetőségük a weboldalon regisztrálni egy űrlap kitöltésével.
RE01	Robotcheck	Regisztráció csak akkor sikeres, ha a felhasználó egy kérdés helyes megválaszolásával igazolni tudja, hogy nem robot.
RE02	Visszajelzés	Sikeres regisztráció esetén a felhasználó számára kerüljenek kiírásra az adatai ellenőrzés céljából.
RE03	Feltételek elfogadása	A felhasználónak regisztráció alkalmával el kell fogadnia a felhasználási feltételeket. Ha nem teszi, nem jöhet létre a regisztráció.
RE04	Adatok megőrzése	Adatok módosítása esetén a korábbi adatok ne vesszenek el. A regisztráció az adott állapotban maradjon elmentve. Jöjjön létre új regisztráció a módosított adatokkal. A törölt regisztráció adatai úgyszintén ne vesszenek el.
RE05	Adatok módosítása	Érvényes regisztrációval rendelkező felhasználónak lehetőséget kell biztosítani a megadott adatainak a módosítására.
RE06	Állapotok	Az adott regisztráció módosítások okozta variánsai között az állapotok tartanak rendet. Minden regisztrációhoz és variánsához tartozzon egy állapot, ami meghatározza, melyik az aktív, az inaktív, a módosított vagy a törölt. Egyszerre egy regisztrációnak mindig csak egy aktív variánsa lehet.
RE07	Módosítás jelszóval	Bármilyen módosítás végrehajtásához szükséges az aktuális jelszó megadása.
RE08	Adatok mentése	A regisztrációról minél több egyéb információ legyen számon tartva. Konkrétan a következők: regisztráció dátuma, utolsó hozzászólás ideje, utolsó módosítás ideje, utolsó belépés ideje, belépések száma, lekérdezések száma, megnyitott bejegyzések sorszáma dátummal.
RE11	Kitöltés megőrzése	Ha a regisztrációnál megadott adatokról kiderül szerveroldalon, hogy hibásak, lehetőséget kell biztosítani az űrlap újbóli kitöltésére. Mikor az űrlap újra betölt, a korábban beírt adatok legyenek megőrizve.
RE12	Biztonság regisztrációnál	Regisztrációnál a speciális karaktereket szöktetni kell, egy nap 30 regisztrációnál ne lehessen több, egy munkafolyamatban legfeljebb 10 regisztráció lehetséges.
SZ00	Új bejegyzés	Legyen lehetőség új bejegyzést írni.

SZ01	Bejegyzés szerkesztése	Legyen lehetőség szerkeszteni a meglévő bejegyzéseket.
SZ02	Rendelkezésre állás	Az írónak minél kevesebbet kelljen foglalkoznia a fejléc adatainak megadásával. Írás helyett a témákat és hozzáférési szinteket legördülő menüből, a címkéket jelölőnégyzet kipipálásával lehessen kiválasztani. A dátum, fejezet és év mezőkre tegyen kísérletet az oldal kitölteni. Röviden szólva álljon rendelkezésre annyi adat készen, amennyi csak lehet.
SZ03	Szakaszokra bontás	Bejegyzés szövegének írásakor az írónak legyen lehetősége a szöveget több szakaszra felbontani, és ezekhez különböző hozzáférési szintet rendelni.
SZ04	Napi mennyiség	Hagyományból a naplóban egy nap csak egy bejegyzés írható. (Amennyiben az író mégis egy napon akar több bejegyzést írni, akkor azoknak későbbi dátumokat kell adni. Egy dátum csak egyszer szerepelhet az adatbázisban!)
ÜL00	Űrlap kliensoldali ellenőrzése	Űrlapon megadott adatok kerüljenek ellenőrzésre már kliensoldalon *JavaScript* segítségével. (Regisztrálás és adatmódosítás esetén.)
ÜL01	Űrlap szerveroldali ellenőrzése	Űrlapon megadott adatok kerüljenek ellenőrzésre szerveroldalon. (Regisztrálás és adatmódosítás esetén.)
ÜL02	Kötelező mezők	Ellenőrizni kell, hogy a felhasználó kitöltötte-e az űrlap kötelezően kitöltendő mezőit. (Regisztrálás és adatmódosítás esetén.)
ÜL03	Karakterlánc hossza	Ellenőrizni kell, hogy a felhasználó a megengedett karakterlimitet nem haladta meg a mezőkben. (Regisztrálás és adatmódosítás esetén.)
ÜL04	Szabályos email	Ellenőrizni kell, hogy a felhasználó szabályos formában adta-e meg az email címét. (Regisztrálás és adatmódosítás esetén.)
ÜL05	Jelszavak egyezése	Ellenőrizni kell, hogy a megadott jelszó és annak megerősítése egyezik-e. (Regisztrálás és adatmódosítás esetén.)
ÜL06	Foglalt felhasználónév	Ellenőrizni kell, hogy a megadott felhasználónév nem foglalt-e már.
ÜL07	Foglalt email	Ellenőrizni kell, hogy a megadott email cím nem foglalt-e már.

A követelményeknek részét képezi, hogy miről milyen adatokat kell lementeni az adatbázisba. Ez az ’Adatbázis’ fejezet ’Adatok leírása’ alfejezetében kerül kifejtésre.

Adatbázis

Amiről az előszóban nem esett szó, az a weboldal mögött álló adatbázis. A bejegyzések ugyanis nem külön legenerált weboldalak, mert azok a lekéréseknek megfelelően dinamikusan töltődnek fel tartalommal. Hatékonyan megtervezni egy adatbázist nem könnyű feladat. A tervezés főbb szempontjai kerülnek most kifejtésre, amivel már az oldal főbb működése is körvonalazódni fog.

Adatok leírása

A napló bejegyzésekből áll. Minden bejegyzésnek van egyedi sorszáma, egyedi címe és egyedi dátuma (hagyományból egy nap egy bejegyzés írható). Kötelezően megadandó egy fő témakör, illetve tetszőlegesen sok címke adható hozzá. Minden bejegyzéshez hozzárendeljük a napló aktuális korát években, illetve egy fejezetszámot. Meg lehet jelölni a bejegyzéseket ajánlottként, megadható rövid tartalom, és jelezhető az olvasónak, hogy a bejegyzés ugyan meg lett írva, de még leellenőrizetlen. Lehetőség van rá, hogy az író hozzáférési szinttel lássa el a bejegyzést, akár bekezdésenként eltérőekkel is.

Regisztrálhatnak felhasználók, melyeknek tárolni kell a felhasználónevét, jelszavát, továbbá megadható becenév, email cím és jelmondat. Automatikusan kap mindenki egy később átállítható hozzáférési szintet, automatikusan mentődik a regisztráció ideje és dátuma. Naplózásra kerül az utolsó belépés, hozzászólás és adatlap módosításának dátuma, továbbá a belépések és lekérdezések száma és a lekérdezett bejegyzések sorszáma dátummal együtt. Minden adat mentve marad, az adatlap módosításával új regisztráció jön létre, ezzel a régi adatokat megőrizve. Az aktív regisztráció kiválasztásához mindegyikhez egy állapot van rendelve (és aktív állapotú mindig csak egy van). Lehetőség van az olvasónak az író számára megjegyzést küldeni, amik szintén lementésre kerülnek. Regisztrációkor egy robotszűrő kérdést tesz fel a felhasználónak, amit helyesen kell megválaszolni. Ezek a kérdések és hozzá a válaszok véletlenszerűen választódnak ki az adatbázisból.

Mint a weboldal bevezetőszövege, a regisztrációt segítő leírás és még pár hasonló, funkcióval nem rendelkező szöveg nem kell az adatbázisba kerüljön, közvetlenül a weboldalra írhatóak.

Adatbázis kialakítása

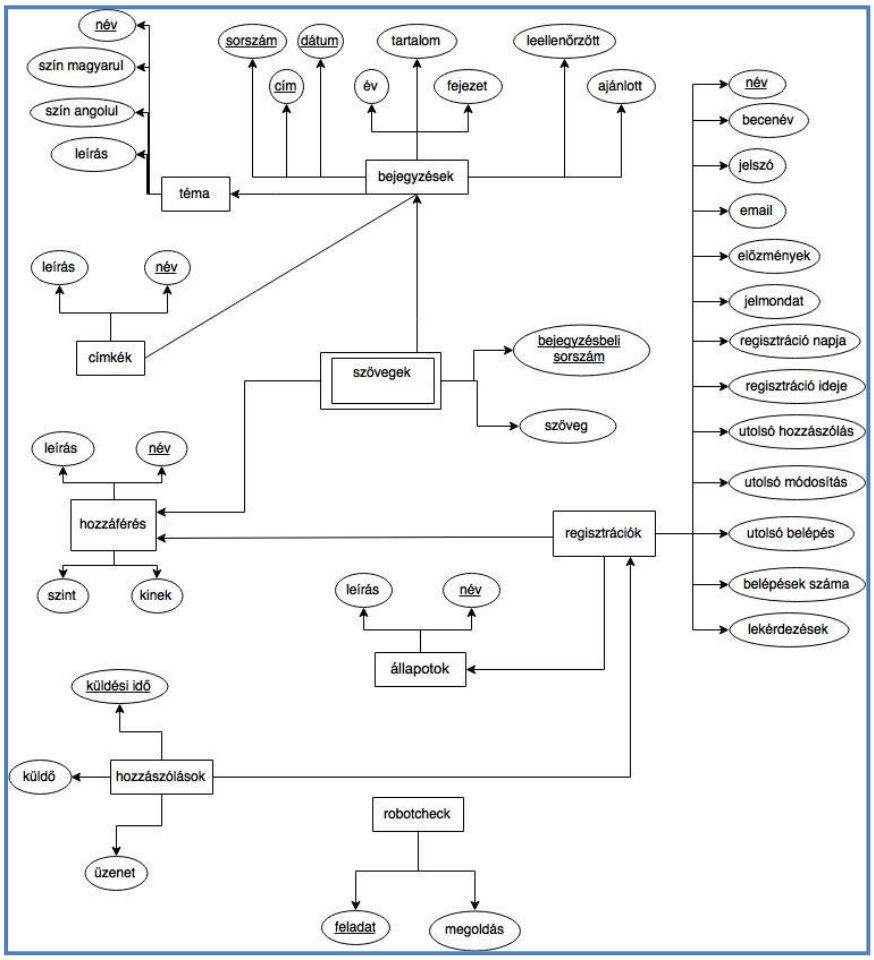
Főbb szempontok

- Az egy bejegyzésre globálisan vonatkozó adatok (sorszám, cím, dátum… stb. mondhatni a fejléc) kerülnek egy táblába, a bejegyzés szövege viszont szakaszonként egy másikba, ahol minden rekord saját hozzáférési szinttel rendelkezik
- A bejegyzés hozzáférési szintje a legalacsonyabb hozzáférési szintű szakasz szintje. Ez tárolásra kerül a fejlécadatok mellett, még ha ez redundanciát is jelent. Sok számítás megspórolható vele. Fontos szerepe van a HS05 kódú követelmény értelmében.
- Ha a szöveg közepén van egy szakasz, ami privátabb megjegyzést tartalmaz a többinél, akkor azt három szakaszra lehet bontani, az elsőnek és a harmadiknak alacsonyabb, a másodiknak magasabb hozzáférési szintet adva. Ez három rekord lesz a szövegek táblájában, és mindegyik tartalmazza a bejegyzés fejlécének azonosítóját.
- A hozzáférési szintek külön táblába kerülnek, amiket ugyan el lehet nevezni, de használat során ezeket egy konkrét szám szimbolizálja, hogy össze lehessen hasonlítani a bejegyzéseknél megadott szintekkel, és el lehessen dönteni, melyik a magasabb. A szintek száma nem egyezik az azonosítójukkal, ami lehetővé teszi a bővítést. (Ha van például szint 1

és szint 2, ezeknek az azonosítója 1 és 2. Ha bekerül egy új szint a kettő közé, akkor sorra szint 1, 2 és 3 lesz, de az azonosítójuk 1, 3 és 2. Kódban szigorúan csak azonosítókat szabad megadni karbantarthatóság és bővíthetőség érdekében.)

- A regisztrációk táblájában az állapotok azonosítóját mentjük, az állapotok szöveges elnevezései külön táblába kerülnek. Ugyanez igaz a hozzáférési szintre és a témakörökre.
- Több címke is rendelhető egy bejegyzéshez, emiatt kapcsolótáblát kell alkalmazni.
- Az adatbázis minden fontosabb eleméhez (témakörök, címkék és hozzáférési szintek) külön leírás is tartozik, hogy a ne csak az elnevezésük utaljon a betöltött szerepükre. Ez a használatot nagyban megkönnyíti írói, olvasói és programozói oldalról egyaránt.
- A bejegyzésszöveg szakaszaihoz külön le kell menteni a szakasz sorszámát, hogy helyesen tudjuk visszanyerni a teljes szöveget.
- A szakaszokat a bejegyzés címe és a szövegbeli sorszáma együttesen azonosítja.
- A bejegyzések címei témakörtől függően más színűek a tartalomjegyzékben. (NA04) A témakörök színeinek a neve magyarul és angolul, illetve a hexadecimális kódja is lementésre kerül. Ez segít, hogy a PHP-vel összeállított HTML oldalon színezéseket lehessen elhelyezni. (Ez karbantarthatóság végett van így. Elég az adatbázisban átírni egy témakör színét, és a weboldalon mindenhol az új színezés lesz érvényben.)

Entitás-reláció diagram



Sémák megvalósítása

Megfontolások alapján az adatbázis sémái a következők:

```
allapotok (
  id, nev, leiras
)

 bejegyzesek (
  sorszam, cim, datum, tartalom, tema_id, ev, fejezet,
  hozzaferesi_szint, leellenorzott, ajanlott
)

cimkek (
  id, nev, leiras
)

hozzaferes (
  id, nev, szint, kiknek, leiras
)

hozzaszolasok (
  id, kuldo, kuldes_ido, uzenet
)

kapcs_bej_cim (
  bejegyzes_sorszam, cimke_id
)

regisztraciok (
  id, elso_id, nev, becenev, jelszo, elozmenyek, email, jelmondat,
  reg_nap, reg_ido, utolso_hozzaszolas, utolso_modositas,
  utolso_belepes, belepesek, lekerdezesek, hozzaferesi_szint,
  allapot_id
)

robotcheck (
  id, feladat, megoldas
)

szovegek (
  id, bejegyzes_sorszam, bejegyzesbeli_sorszam, hozzaferesi_szint,
  szoveg
)

temak (
  id, nev, hun_szin, eng_szin, hexa_szin, leiras
)
```

Adatbázis tábláinak részletei

A könnyen áttekinthető sémák után következzenek az adatbázis részletei, melyben dokumentálásra kerülnek az egyes táblák és azok attribútumainak beállításai. A leírások kódrészletek az exportált *MySQL* fájlból. Továbbá néhány tábla tartalma is bekerül ebbe az alfejezetbe a kódban használt azonosítók könnyebb értelmezéséhez.

Állapotok

```
`allapotok` (
  `id` int(11) NOT NULL,
  `nev` varchar(16) COLLATE utf8_hungarian_ci NOT NULL,
  `leiras` text COLLATE utf8_hungarian_ci NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

id	nev
1	aktív
2	inaktív
3	módosított
4	törölt

Bejegyzések

```
`bejegyzesek` (
  `sorszam` int(4) NOT NULL,
  `cim` varchar(32) NOT NULL,
  `datum` varchar(16) CHARACTER SET utf8 COLLATE utf8_hungarian_ci NOT NULL,
  `tartalom` text CHARACTER SET utf8 COLLATE utf8_hungarian_ci NOT NULL,
  `tema_id` int(2) NOT NULL,
  `ev` int(2) NOT NULL,
  `fejezet` int(4) NOT NULL,
  `hozzaferesi_szint` int(2) NOT NULL,
  `leellenorzott` tinyint(1) NOT NULL,
  `ajanlott` tinyint(1) NOT NULL DEFAULT '0'
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Címkék

```
`cimkek` (
```

```
  `id` int(11) NOT NULL,
  `nev` varchar(32) COLLATE utf8_hungarian_ci NOT NULL,
  `leiras` text COLLATE utf8_hungarian_ci NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=32 DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

Hozzáférés

```
`hozzaferes` (
  `id` int(2) NOT NULL,
  `nev` varchar(32) COLLATE utf8_hungarian_ci NOT NULL,
  `szint` int(2) NOT NULL,
  `kiknek` text COLLATE utf8_hungarian_ci NOT NULL,
  `leiras` text COLLATE utf8_hungarian_ci NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

id	nev	szint
1	teljesen publikus	1
2	regisztrálva publikus	2
3	bizalmasan kezelendő	4
4	privát	6
5	szigorúan titkos	7
6	szerkeszthető	8
7	komolyabb érdeklődőknek publikus	3
8	különleges bizalommal kezelendő	5

Hozzászólások

```
`hozzaszolasok` (
  `id` int(11) NOT NULL,
  `kuldo` varchar(32) COLLATE utf8_hungarian_ci NOT NULL,
  `kuldes_ido` datetime NOT NULL,
  `uzenet` text COLLATE utf8_hungarian_ci NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

Kapcsolótábla bejegyzések és címkék között

```
`kapcs_bej_cim` (
  `bejegyzes_sorszam` int(4) NOT NULL,
  `cimke_id` int(2) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

Regisztrációk

```
`regisztraciok` (
  `id` int(4) NOT NULL,
  `elso_id` int(11) NOT NULL,
  `nev` varchar(32) COLLATE utf8_hungarian_ci NOT NULL,
  `becenev` varchar(32) COLLATE utf8_hungarian_ci NOT NULL,
  `jelszo` varchar(32) COLLATE utf8_hungarian_ci NOT NULL,
  `elozmenyek` text COLLATE utf8_hungarian_ci NOT NULL,
  `email` varchar(64) COLLATE utf8_hungarian_ci NOT NULL,
  `jelmondat` varchar(256) COLLATE utf8_hungarian_ci NOT NULL,
  `reg_nap` date NOT NULL,
  `reg_ido` time NOT NULL,
  `utolso_hozzaszolas` date NOT NULL,
  `utolso_modositas` datetime NOT NULL,
  `utolso_belepes` datetime NOT NULL,
  `belepesek` int(32) NOT NULL,
  `lekerdezesek` int(32) NOT NULL,
  `hozzaferesi_szint` int(32) NOT NULL,
  `allapot_id` int(2) NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

Robotcheck

```
`robotcheck` (
```

```
  `id` int(11) NOT NULL,
  `feladat` text CHARACTER SET utf8 COLLATE utf8_hungarian_ci NOT NULL,
  `megoldas` text CHARACTER SET utf8 COLLATE utf8_hungarian_ci NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=25 DEFAULT CHARSET=utf8;
```

Szövegek

```
`szovegek` (
  `id` int(4) NOT NULL,
  `bejegyzes_sorszam` int(4) NOT NULL,
  `bejegyzesbeli_sorszam` int(4) NOT NULL,
  `hozzaferesi_szint` int(2) NOT NULL,
  `szoveg` text COLLATE utf8_hungarian_ci NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=1544 DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

Témák

```
`temak` (
  `id` int(2) NOT NULL,
  `nev` varchar(16) COLLATE utf8_hungarian_ci NOT NULL,
  `hun_szin` varchar(16) COLLATE utf8_hungarian_ci NOT NULL,
  `eng_szin` varchar(16) COLLATE utf8_hungarian_ci NOT NULL,
  `hexa_szin` varchar(16) COLLATE utf8_hungarian_ci NOT NULL,
  `leiras` text COLLATE utf8_hungarian_ci NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=utf8 COLLATE=utf8_hungarian_ci;
```

Regisztráció

A regisztráció fontos része a weboldalnak (RE00), hogy a hozzáférési szinteket kezelni lehessen. A regisztráció a következő lépésekből áll:

1. A felhasználó kitölti az űrlapot szándékának megfelelően.
2. Ekkor még nem küldhető el a regisztráció. Gombnyomásra **kliensoldalon** egy *JavaScript* először ellenőrzi a kitöltött adatokat. Ha egy mező nincs helyesen kitöltve, felugró ablakkal lesz tájékoztatva a teendőkről a felhasználó.
3. A kliensoldali ellenőrzés természetesen nem biztonságos, ezért **szerveroldalon** ez újra szükséges pár egyéb szempontot is hozzávéve.
4. **Biztonsági intézkedések** is szükségesek.
5. Ha minden adat helyes, akkor azok az adatbázisban **lementés**re kerülnek.
6. Ezek után képes a felhasználó **bejelentkezni**.
7. A felhasználónak később módjában áll a megadott adatait **módosít**ani.

Kliensoldali ellenőrzések

reg.php

A szerver terhelésének csökkentése érdekében a hibásan kitöltött űrlapok döntő többsége már kliensoldalon detektálva lesz a *reg.php* oldalon. (ÜL00) Az ellenőrzést *JavaScript* végzi, és felugró ablakkal jelzi a felhasználónak a hibát. A mezők valid értékeihez a következő függvények definiálása volt szükséges:

- `regCheck(nev, becenev, jelszo1, jelszo2, email, jelmondat, megoldas, elfogad):`
  - Ezt a függvény fut le gombnyomáskor. Feladata a többi függvény meghívása.
- `hosszCheck(mezo, hossz):`
  - Az adatbázisban korlátozott karakterszám áll rendelkezésre az adatok tárolásához, ezért ellenőriznünk kell a bevitt értékek hosszát. Ha egy érték túl hosszú, hamis értékkel tér vissza. (ÜL03)
- `uresCheck(mezo):`
  - Regisztráció alkalmával mindenhol vannak kötelezően kitöltendő mezők, így ezen a weboldalon is. Ilyen a név, a jelszó és annak megerősítése, illetve a robotcheck kérdésére adandó válasz. Ha egy mező üres, igaz értékkel tér vissza. (ÜL02)
- `emailCheck(mezo):`
  - Amennyiben az email cím kitöltésre kerül, érdemes megvizsgálni a helyes formátumot. Ha egy email formátuma hibás, hamis értékkel tér vissza. (ÜL04)
- `jelszoCheck(jelszo1, jelszo2):`
  - A jelszót kétszer szükséges megadni. Ez a függvény karakterenként összehasonlítja a megadott karakterláncokat, és csak teljes egyezés esetén fogadja el. Ha a jelszavak nem egyeznek, hamis értékkel tér vissza. (ÜL05)
- `elfogadta(elfogad):`
  - A weboldalnak erkölcsi felhasználási feltételei vannak. Regisztrációhoz ezeket el kell fogadni a megfelelő jelölőnégyzet kipipálásával. Ha a felhasználó nem fogadta el a feltételeket, hamis értékkel tér vissza. (RE03)

Amennyiben az űrlap kitöltése helyes volt a *reg.php* oldalon, a regisztráció elküldésére alkalmas gomb aktívvá válik, amire kattintva POST üzenet változóiban az értékek átkerülnek a *reg2.php* oldalra.

```
// regisztralo gomb tiltasa
$(document).ready(function() {
    $('#Button').attr('disabled', 'disabled');
});

// osszes beviteli adat leellenorzese
function regCheck(nev, becenev, jelszo1, jelszo2, email, jelmondat, megoldas,
elfogad){
    // elfogadta-e a felhasznalasi felteteleket
    if(!elfogadta(elfogad)) return false;
    // ures-e
    if(uresCheck(nev)) return false;
    if(uresCheck(jelszo1)) return false;
    if(uresCheck(jelszo2)) return false;

    if(uresCheck(megoldas)) return false;
    // tul hosszu-e
    if(!hosszCheck(nev, 32)) return false;
    if(!hosszCheck(becenev, 32)) return false;
    if(!hosszCheck(jelszo1, 32)) return false;
    if(!hosszCheck(jelszo2, 32)) return false;
    if(!hosszCheck(email, 64)) return false;
    if(!hosszCheck(jelmondat, 256)) return false;
    if(!hosszCheck(megoldas, 32)) return false;
    // jo e-mail-e
    if(email.value != "") if(!emailCheck(email)) return false;
    // jelszavak egyeznek-e
    if(!jelszoCheck(jelszo1, jelszo2)) return false;
    // helyes kitöltes eseten regisztracio kuldesenek engedelyezese
    $('#Button').removeAttr('disabled');
    return true;
}
```

Szerveroldali ellenőrzések

reg2.php

A *reg.php* oldalról, a *reg2.php* oldalra navigálhatunk tovább. A regisztráció során megadott adatok POST üzenetben kerülnek átadásra. Itt ugyanazok az ellenőrző függvények hajtódnak végre, mint a *reg.php* oldalon kliensoldali ellenőrzéskor, de most szerveroldalon *PHP* nyelven megírva. (ÜL01) Ezeken felül további ellenőrzésekre is szükség van, amelyek csak szerveroldalon ellenőrizhetőek:

- Felhasználónév foglalt. (RE09)
- Email cím már regisztrálva van. (RE10)
- A kérdésre adott válasz hibás. (RE01)

```
// EGYEZO NEV KERESESE
$vanmarnev;
if($result = $conn->query("SELECT nev FROM regisztraciok WHERE nev=\"".$nev."\",
AND (allapot_id=\"1\" OR allapot_id=\"2\") LIMIT 1")){
    $row = $result->fetch_assoc();
    $vanmarnev = $row["nev"];
    $result ->free();
}
else $vanmarnev = "";
```

Ezekhez adatbázis-lekérdezések szükségesek. Amennyiben az értékek hibásak, visszatérhetünk a *reg.php* oldalra. A kitöltött adatok POST üzenetben visszaküldésre kerülnek, így nem szükséges a teljes űrlapot újra kitölteni. (RE11)

```
// van-e mar ilyen felhasznalonev
else if(!empty($vanmarnev)){
    print "
    <br/>
    <center>
        Ezzel a felhasználónévvel  már van regisztrált felhasználó.
        A linkre kattintva térj vissza, és adj meg másik nevet!
        <br/>
        <br/>
    </center>";

    $gombszoveg = "Megpróbálom még egyszer!";
}
```

```
// VISSZANAVIGALO GOMB
print "
<form action=\"reg.php\" method=\"POST\">
    <input type=\"hidden\" name=\"nev\" value=\"".$_POST["nev"]."\">
    <input type=\"hidden\" name=\"becenev\" value=\"".$_POST[
    "becenev"]."\">
    <input type=\"hidden\" name=\"email\" value=\"".$_POST["email"].
    "\">
    <input type=\"hidden\" name=\"jelmondat\" value=\"".$_POST[
    "jelmondat"]."\">
    <center>
        <input type=\"submit\" class=\"postlink\" style=\"color:
        blue\" value=\"".$gombszoveg."\"/>
    </center>
</form>";
```

Biztonsági intézkedések

reg2.php

Ugyan a honlap készítője nem ismeri kellő részletességgel a támadások módját, de bekerült pár óvintézkedés (RE12):

- Az adatokból szöktetni kell a speciális karaktereket.
- Egy felhasználó egy munkafolyamatban legfeljebb tízszer regisztrálhat.
- Egy nap legfeljebb 30 alkalommal regisztrálhatnak.

Adatok lementése az adatbázisba

reg2.php

Az űrlap helyes kitöltése esetén az űrlap adatai lementésre kerülnek:

- Név
- Becenév
- Jelszó
- Email
- Jelmondat

Ezeken felül további adatok kerülnek automatikusan kitöltésre:

- Regisztráció ideje
- Regisztráció dátuma
- Állapot (aktív)
- Hozzáférési szint (teljesen publikus szint +1)

Miután az adatok lementésre kerültek, kiíródnak még egyszer az oldalra, hogy a felhasználó ellenőrizni tudja. (RE02)

Bejelentkezés

naplo.php, emlekezteto.php, emlekezteto2.php, logout.php

A *naplo.php* egy univerzális oldal. Ez felel a bejegyzések lekérdezéséért és helyes megjelenítéséért. Az oldal tetején azonban a bejelentkező modul is helyet kapott. (BJ00) Amikor egy felhasználó bejelentkezik, a következő események zajlanak le:

- Függetlenül a bejelentkezéstől az oldal mindig elmenti a megtekintett bejegyzések sorszámát és a megtekintés dátumát. Mikor valaki még nem jelentkezik be, a ’Névtelen’ nevet kapja. Annak érdekében, hogy ez a döntés ne okozzon problémát, létre lett hozva egy ’Névtelen’ felhasználó, így másoknak ez a név már nem elérhető. (Ha bejelentkezett felhasználóról van szó, akkor az ő regisztrációjánál lesz naplózva az esemény.)

```
//////////////// ELOZMENY ////////////////////

    if(isset($_SESSION["felhasznalo"]) AND isset($_REQUEST["bejegyzes"])){

        $tabla_korabbielozmeny = "SELECT * FROM regisztraciok WHERE nev=\"".$_SESSION[
        "felhasznalo"]."\" and allapot_id=\"1\" LIMIT 1";
        if ($result_korabbielozmeny = $conn->query($tabla_korabbielozmeny)) {
            $row_korabbielozmeny = $result_korabbielozmeny->fetch_assoc();
        }

        $tabla_hozzaszolasok = "UPDATE regisztraciok SET elozmenyek=\"".$_REQUEST[
        "bejegyzes"]." ".date("Y-m-d H:i:s")."; ".$row_korabbielozmeny["elozmenyek"].
        "\" WHERE nev=\"".$_SESSION["felhasznalo"]."\" and allapot_id=\"1\"";
        mysqli_select_db($conn, "naplo");
        $retval_hozzaszolasok = mysqli_query($conn, $tabla_hozzaszolasok);
        if(! $retval_hozzaszolasok ) die('Hiba lépett fel: ' . mysqli_error($conn));

    }
    // nem regisztralt felhasznalo
    else if(isset($_REQUEST["bejegyzes"])){
        $tabla_korabbielozmeny = "SELECT * FROM regisztraciok WHERE nev=\"Névtelen\"
        and allapot_id=\"1\" LIMIT 1";
        if ($result_korabbielozmeny = $conn->query($tabla_korabbielozmeny)) {
            $row_korabbielozmeny = $result_korabbielozmeny->fetch_assoc();
        }

        $tabla_hozzaszolasok = "UPDATE regisztraciok SET elozmenyek=\"".$_REQUEST[
        "bejegyzes"]." ".date("Y-m-d H:i:s")."; ".$row_korabbielozmeny["elozmenyek"].
        "\" WHERE nev=\"Névtelen\" and allapot_id=\"1\"";
        mysqli_select_db($conn, "naplo");
        $retval_hozzaszolasok = mysqli_query($conn, $tabla_hozzaszolasok);
        if(! $retval_hozzaszolasok ) die('Hiba lépett fel: ' . mysqli_error($conn));
    }
```

- Bejelentkezéskor ellenőrzésre kerül a név és a jelszó. (BJ01)

```
        // BELEPESI KISERLET

        // felhasznalo kikeresese
        $tabla = "SELECT * FROM regisztraciok WHERE nev=\"".$nev."\" AND (allapot_id=1
        OR allapot_id=2) LIMIT 1";
        if ($result = $conn->query($tabla)) {
            $row = $result->fetch_assoc();

            // jelszo egyeztetese
            if($row["jelszo"]==$jelszo){
```

- Ha az adatok helyesek, ellenőrzésre kerül az időkorlát. Ha a felhasználó 16 200 000 másodpercnél, azaz fél évnél régebben lépett be, a regisztráció inaktív állapotba kerül. (BJ03)

```
                // utolso belepes ota eltelt ido: sikeres belepes
                if(strtotime(date("Y-m-d"))-strtotime($row["utolso_belepes"])< 16200000 )
```

```
// inaktiv allapotba helyezes
$tabla = "UPDATE regisztraciok SET allapot_id=\"2\" WHERE nev=\"".
$nev."\" and (allapot_id=\"1\" or allapot_id=\"2\")";
mysqli_select_db($conn, "naplo");
$retval = mysqli_query($conn, $tabla);
if(! $retval ) die('A bejelentkezés technikai okok miatt nem
lehetséges: ' . mysqli_error($conn));
```

- Ha az időkorlát sincs túllépve, a bejelentkezés sikeres. SESSION változókban mentésre kerül a felhasználónév és a hozzáférési szint (illetve annak azonosítója), továbbá frissülnek olyan adatok, mint utolsó belépés, lekérdezések száma... stb.. (RE08)

```
// BELEPESSZAM NOVELESE

$belepesek_szama = $row["belepesek"]+1;
$tabla = "UPDATE regisztraciok SET belepesek=\"".$belepesek_szama.
"\", utolso_belepes=\"".date("Y-m-d H:i:s")."\" WHERE nev=\"".
$_SESSION["felhasznalo"]."\" and allapot_id=\"1\"";
mysqli_select_db($conn, "naplo");
$retval = mysqli_query($conn, $tabla);
if(! $retval ) die('A hozzászólás technikai okok miatt nem menthető
el: ' . mysqli_error($conn));
```

- Amennyiben a felhasználónév-jelszó páros hibás, a felhasználó hibaüzenetet kap és egy linket, ami továbbvezet az emlekezteto.php oldalra, ahol meg kell adni a nevet és az email címet. Innen az *emlekezteto2.php* oldalra kerül a felhasználó. Amennyiben ez a név-email páros megtalálható az adatbázisban, email generálódik benne a jelszóval, ami a megadott email címre el is lesz küldve. (BJ02)

```
// valtozok biztositasa
$nev = mysql_escape_string($_POST["nev"]);
$email = mysql_escape_string($_POST["email"]);

// felhasznalo kikeresese
$tabla = "SELECT * FROM regisztraciok WHERE nev=\"".$nev."\" LIMIT 1";
if ($result = $conn->query($tabla)) {
    $row = $result->fetch_assoc();
    // email egyeztetese
    if($row["email"]==$email){
        mail($email,
         "Jelszóemlékeztető",
         "A regisztrációhoz a következő jelszót adtad meg: ".$row_email["jelszo"]);
    }
    $result->free();
}
```

Kijelentkezés

Bejelentkezés után a regisztráció linkje eltűnik, és megjelentik helyette a kijelentkezés gombja. Erre kattintva az oldal továbbnavigálja a felhasználót a *logout.php* oldalra, ahol törlődik a munkafolyamatból a felhasználó neve és hozzáférési szintje. A weboldalon ezután ismételten csak a publikus tartalmakat láthatja.

```php
// FELHASZNALO KIJELENTKEZTETESE
unset($_SESSION["felhasznalo"]);
unset($_SESSION["felhasznalo_hozzaferesi_szintje"]);
```

Adatok módosítása

beallitasok.php, beallitasok2.php

Az adatok módosítása a *beallitasok.php* oldalon lehetséges. (RE05) Az adatok ellenőrzése azonos a regisztrációnál alkalmazottal, viszont nincs robotcheck és feltételelfogadás, de szükséges a régi jelszó megadása az adatok módosításához. (ÜL**, RE07)

Az űrlap helyes kitöltésével betöltődik a beallitasok2.php oldal, megtörténnek a szerveroldali ellenőrzések, majd új regisztráció jön létre. Az új kapja az 'aktív' állapotot, a régi pedig a 'módosított' állapotot. Az első regisztrációkor kapott azonosító ilyenkor öröklődik. (ÜL**, RE04, RE06)

```php
// felhasznaloi adatok archivalasa
mysqli_select_db($conn, "naplo");
$retval_hozzaszolas_ideje = mysqli_query($conn, "UPDATE
regisztraciok SET allapot_id=\"3\" WHERE nev=\"".$_SESSION[
"felhasznalo"]."\"");
if(! $retval_hozzaszolas_ideje ) die('Üzenetküldés során
hiba lépett fel: ' . mysqli_error($conn));

// adatok kimentese
$tabla1 = "INSERT INTO regisztraciok (nev, elso_id,
becenev, jelszo, email, jelmondat, reg_nap, reg_ido,
utolso_hozzaszolas, utolso_modositas, utolso_belepes,
belepesek, lekerdezesek, hozzaferesi_szint, allapot_id)
VALUES (\"".$nev."\", \"".$elso_id."\", \"".$becenev."\",
\"".$jelszo."\", \"".$email."\", \"".$jelmondat."\", \"".
$reg_nap."\", \"".$reg_ido."\", \"".$utolso_hozzaszolas.
"\", \"".$mod_ido."\", \"".$utolso_belepes."\", \"".
$belepesek."\", \"".$lekerdezesek."\", \"".
$hozzaferesi_szint."\", 1)";

mysqli_select_db($conn, "regisztraciok");
$retval = mysqli_query($conn, $tabla1);
```

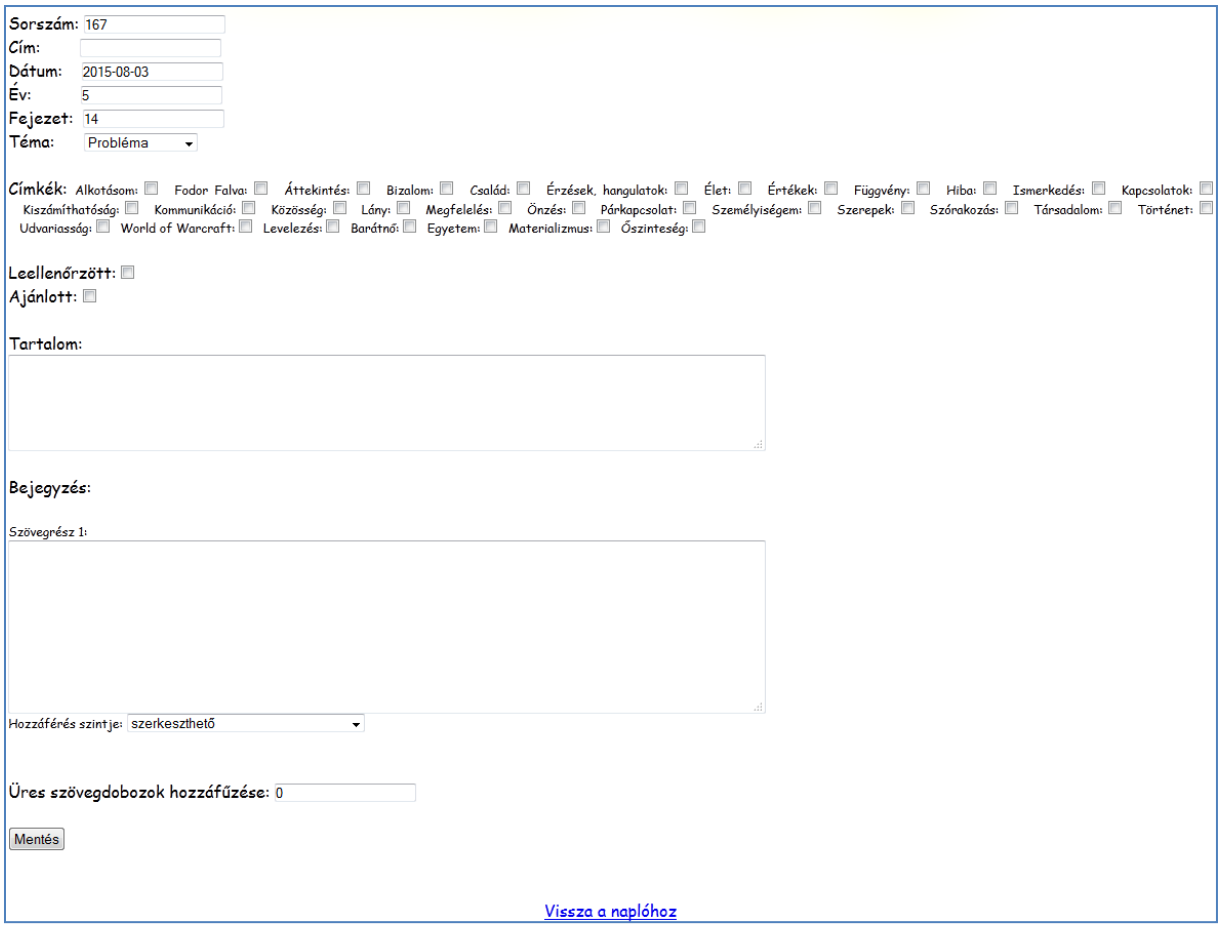
Bejegyzések szerkesztése

A *naplo.php* oldalon bejelentkezés után megjelennek linkek a bejelentkező modul helyén, az egyikkel lehetőség van új bejegyzést írni. (SZ00) Egy megnyitott bejegyzés címe mellett úgyszintén van egy link, mellyel szerkeszteni lehet azt. (SZ01) Mindkét helyről a *szerkesztes.php* oldalra vezetnek a linkek.

Az oldal betöltődése előtt minden esetben végbemennek a következő előkészületek:

- Ellenőrzi, hogy a felhasználó az elérhető legnagyobb hozzáférési szinttel rendelkezik. (HS07)
- Definiál tömérdek változót.
- A témák, címkék és hozzáférési szintek kiválasztandóak bejegyzésíráskor, nem kell kézzel beírni. Ennek megfelelően az adatbázisból a lehetséges értékeket le kell kérdezni. (SZ02)
- Végül szükség van egy input mezőre, hogy a bejegyzést tartalommal töltsük meg.

Ezek alapján következőképpen épül fel az oldal: Legelöl szerepelnek a főbb adatok beviteli mezői (sorszám, cím, dátum, év, fejezet). Ezt követi egy combobox, amiből a passzoló témát lehet kiválasztani. Utána következnek a címkék jelölőnégyzetekkel felsorolva. Utána van két további jelölőnégyzet, melyekkel a bejegyzés tartalmának leellenőrzöttségét erősíthetjük meg, illetve ajánlhatjuk a bejegyzést az olvasóknak. Ezek alatt található a rövid tartalom szövegdoboza, ahova értelemszerűen a rövid tartalmat lehet írni. A bejegyzésnek a szövegének a szövegdoboza következik, a szövegdoboz alatt pedig egy combobox, melyből kiválasztható a szöveg hozzáférési szintje. Szerepel az oldal alján még egy mentés gomb is. Erre kattintva az adatok lementésre kerülnek és az oldal újra betöltődik. Amennyiben a szöveget több szakaszra szeretnénk bontani, lehetőségünk van megmondani, hogy az oldal újratöltésekor hány extra szövegdobozra van szükségünk. Mindegyik szövegdobozhoz tartozik saját hozzáférési szint beállítására alkalmas combobox. (HS02)



Az oldal működési módja POST és REQUEST üzenetektől függően háromféle lehet: új bejegyzés létrehozása, meglévő bejegyzés előhívása, módosítások mentése. Ezen esetek következnek most részletesen kifejtve.

Új bejegyzés

szerkesztes.php

Első lehetőségünk, hogy egy teljesen új bejegyzést nyitunk. A *szerkesztes.php* oldal ezt úgy észleli, hogy nem érkezik bejegyzéssorszám se REQUEST se POST üzenet formájában. Nem kérdezünk le meglévő bejegyzést, tehát új bejegyzést kell létrehozni.

```
// UJ BEJEGYZES

if(empty($_REQUEST["bejegyzes"]) AND count($hozzaferes_nevei_szintbol)==$_SESSION[
"felhasznalo_hozzaferesi_szintje"] AND !isset($_POST["sorszam"])){
    $sysmsg = "uj bejegyzes";
```

Ilyenkor a mezők üresen maradnak, kivéve párat, amik automatikusan kitöltődnek az írói munkát segítve (SZ02):

- Új sorszámot generál úgy, hogy az eddigi legnagyobb sorszámot megnöveli eggyel.

```php
// uj sorszam
$tabla = "SELECT count(*) FROM bejegyzesek";
if ($result = $conn->query($tabla)){
    $row = $result->fetch_row();
    $sorszam = $row[0] - 11;
    $result->free();
}
```

- Kitölti a dátumot aznapra vonatkozóan.

```php
// uj datum
$datum = date("Y-m-d");
```

- A napló aktuális korát években és a legutolsó fejezetsorszámot átmásolja a legutóbbi bejegyzésből.

```php
// uj ev uj fejezet legfrissebb bejegyzes alapjan
$tabla = "SELECT * FROM bejegyzesek WHERE sorszam=\"".($sorszam - 1)."\"";
if ($result = $conn->query($tabla)){
    $row = $result->fetch_assoc();
    $ev = $row["ev"];
    $fejezet = $row["fejezet"];
    $result->free();
}
```

Az adatokat az író módosíthatja, megadhat címet, és a bejegyzés tartalmának hozzáférési szintet. (HS01)

Mentés

szerkesztes.php

Ha az író befejezte a szerkesztést és menti a munkáját, a mentés gombra kattintva meghívja ugyanazt a *szerkesztes.php* oldalt. Ezúttal POST változó formájában (nem REQUEST) érkezik a bejegyzés sorszáma. Ezt az oldal felismeri, és tudja, hogy mentés történt. Ilyenkor a következő lépések történnek:

- Speciális karakterek szöktetése után lementi az értékeket
- A legkisebb szintű szakasz szintjét külön lementi a fejléc adatai közé. (HS01)
- Módosítja a már meglévő bejegyzés adatait az adatbázisban.
- A szövegrészek (szakaszok) felosztása mivel idővel változhat, ezért ezeket a rekordokat nem lehet frissíteni, mindig törölni kell a régieket, és aztán beszúrni az újakat. (Ez a címkékre is igaz, törölni kell az összes korábban jelölt címkét, és a friss jelöléseket kell menteni helyette.)

Megnyitás

szerkesztes.php

Amennyiben REQUEST változóban érkezik bejegyzéssorszám, a *naplo.php* oldalon egy bejegyzés szerkesztésének a linkjére kattintottak. Ez azt jelenti, hogy a mentés esetével ellentétben nincsenek POST változókba mentve a bejegyzés adatai. Ilyenkor a következő teendők vannak:

- Be kell tölteni az adatbázisból a kiválasztott bejegyzés adatait és szövegét,

- és le kell kérni a címkéket a kapcsolótáblán keresztül.

```
// REGI BEJEGYZES MODOSITASA

else if(isset($_REQUEST["bejegyzes"]) AND count($hozzaferes_nevei_szintbol)==
$_SESSION["felhasznalo_hozzaferesi_szintje"] AND !isset($_POST["nev"])){
    // valtozok kiolvasasa es helyreallitasa
    $sorszam = $_REQUEST["bejegyzes"];
    $tabla_bejegyzes = "SELECT * FROM bejegyzesek WHERE sorszam=\"".$_REQUEST[
    "bejegyzes"]."\"";
    if ($result_bejegyzes = $conn->query($tabla_bejegyzes)){
        $row_bejegyzes = $result_bejegyzes->fetch_assoc();

        // valtozok biztositasa
        $fejezet = $row_bejegyzes["fejezet"];
        $tema_id = $row_bejegyzes["tema_id"];
        $cim = $row_bejegyzes["cim"];
        $datum = $row_bejegyzes["datum"];
        $ev = $row_bejegyzes["ev"];
        $tartalom = $row_bejegyzes["tartalom"];
        $tartalom = str_replace("\\","",$tartalom);
        $bejegyzes_hozzaferesi_szintje = $hozzaferes_szintjei_idbol[$row_bejegyzes[
        "hozzaferesi_szint"]];
        $leellenorzott = $row_bejegyzes["leellenorzott"];
        $ajanlott = $row_bejegyzes["ajanlott"];

        // cimkek azonositoinak lekerese
        $tabla_cimkeid = "SELECT * FROM kapcs_bej_cim WHERE bejegyzes_sorszam=\"".
        $_REQUEST["bejegyzes"]."\"";
        if ($result_cimkeid = $conn->query($tabla_cimkeid)) {
            while ($row_cimkeid = $result_cimkeid->fetch_assoc()) {
                $tabla_cimkenev = "SELECT nev FROM cimkek WHERE id=\"".$row_cimkeid[
                "cimke_id"]."\"";
                if ($result_cimkenev = $conn->query($tabla_cimkenev)) {
                    $row_cimkenev = $result_cimkenev->fetch_assoc();
                    $cimkek[$row_cimkeid["cimke_id"]] = $row_cimkenev["nev"];

                    $result_cimkenev->free();
                }
            }
            $result_cimkeid->free();
        }
        // szovegreszek kigyujtese
        for($i=1; ; $i++){
            $tabla_szoveg = "SELECT * FROM szovegek WHERE bejegyzes_sorszam=\"".
            $_REQUEST["bejegyzes"]."\" AND bejegyzesbeli_sorszam=\"".$i."\"";
            if ($result_szoveg = $conn->query($tabla_szoveg)) {
                if($row_szoveg = $result_szoveg->fetch_assoc()){
                    $szovegreszek[] = str_replace("\\","",$row_szoveg["szoveg"]);
                    $szoveg_hozzaferes[] = $hozzaferes_szintjei_idbol[$row_szoveg[
                    "hozzaferesi_szint"]];
                }
                else break;
```

Napló

naplo.php

A *naplo.php* oldal a leglényegesebb eleme az egész weboldalnak. Működése meglehetősen összetett:

- felelős a bejelentkeztetésért (ahogy az a 'Regisztráció' fejezet 'Bejelentkezés' alfejezetében részletesen ki lett fejtve),
- megszűri a felhasználó számára hozzáférhető bejegyzéseket,
- szűr a felhasználó egyéni beállításai alapján,
- összeállítja a kiválasztott bejegyzés tartalmát,
- a szövegeket megfelelően formázza és hivatkozásokkal látja el,
- végül egy egyszerű üzenetküldő funkcióval is rendelkezik.

A fejezet további részében ezek a fő modulok lesznek részletezve.

Lekérdezések és változók

Az oldal sok változóval dolgozik, melyek értékeit az adatbázis tárolja. Ezeket érdemes külön megemlíteni, hogy később érthető legyen, milyen értékű változók állnak rendelkezésünkre. Az adatbázis-lekérdezések a következőkre vonatkoznak:

- ***Hozzáférési szintek***: A szintek kezelése meglehetősen összetett probléma. Funkció szempontjából sokszor azonosító szerepel a kódban, viszont az összehasonlításokat a szintek konkrét értéke alapján kell végezni. Ezért két hash táblát kell alkotni. Az egyik az azonosító alapján megadja a szint értékét, a másik a szint értékéből az azonosítót. (A kód többszöri újradolgozáson van túl, így „örökölt" egy hash táblát, mely megadja a szintekhez a hozzájuk tartozó elnevezést. Ez már nélkülözhető tábla, de pár helyen használatban van, mikor is az összes elemének a számát kérdezzük le.)
  - `$hozzaferes_nevei_szintbol`
  - `$hozzaferes_id_nevbol`
  - `$hozzaferes_szintjei_idbol`
- ***Felhasználó adatai***: Bejelentkezéskor le kell kérdezni a felhasználónevet, jelszót, az utolsó bejelentkezés dátumát, a regisztráció állapotát és az eddigi lekérdezések számát. Ezek vagy közvetlenül felhasználásra kerülnek, vagy SESSION változóban lesznek elmentve a későbbi felhasználáshoz.
- ***Témakörök***: A szűrőbeállításokhoz és a bejegyzéscímek színezéséhez le kell kérdezni a témákat, azok azonosítóját és a hozzájuk tartozó színeket.
  - `$jelek`
  - `$jelszin`
  - `$temakereso`
  - `$temakereso_idk_nevbol`
- ***Címlista***: A weboldal szélén található a bejegyzések címeinek a listája. A listában szerepelnek az napló évei, továbbá a fejezetek és bejegyzések sorszámai.
  - `$lista[][]`
  - `$lista_ev_sorszambol`
  - `$lista_fejezet_sorszambol`
  - `$lista_bejegyzes_sorszambol`
  - `$lista_sorszamok`

- ***Címkék listája***: A szűrőbeállításokhoz le kell kérdezni az elérhető címkéket és a hozzájuk tartozó azonosítókat.
    - `$cimkekereso_idk_nevbol`
    - `$cimkekereso_nevek_idbol`
    - `$cimkekereso_nevek`
- ***Kiválasztott bejegyzés adatai***: Amikor kiválaszt a felhasználó egy bejegyzést olvasásra, nem csak a szövege, de sok mellékinformáció is megjelenik a fejlécben. Ezeket is le kell menteni.
    - `$cim`
    - `$datum`
    - `$tartalom`
    - `$bejegyzes_ev`
- ***Kiválasztott bejegyzés szövege***: A bejegyzés szövege szakaszokból áll. Mindet le kell kérdezni és le kell menteni őket egy változóba. A szakaszok egymásután fűzhetőek. Az adott szakasz hozzáférési szintjét nem szükséges lementeni, de felhasználásra kerül. Minden szakasz szintje össze lesz hasonlítva a felhasználó szintjével.
    - `$szovegreszek;`
    - `$szovegszam;`
- ***Dátumból cím***: Előfordul, hogy egy bejegyzésnek csak a dátuma van meg. Ez alapján kell megtalálni a hozzá tartozó címet.

Funkciók

A *naplo.php* főbb funkciói kerülnek most bemutatásra az előző alfejezetben taglaltakhoz hasonló sorrendben.

Bejelentkezés

ld.: 'Regisztráció' fejezet 'Bejelentkezés' alfejezete.

Információk

Egy táblázat rövid eligazítást ad az olvasónak a napló használatához. Ennek része egy statikusan kódba írt szöveg, illetve a használatos jelek felsorolása, amiknek a részét képezik a színezett témakörök. (IN02) Ez utóbbi több helyen is megjelenik, ezért adatbázisból kell lekérdezni a karbantarthatóság érdekében. Témánként egy karaktersorozat kerül mentésre, ami érvényes HTML kód. Ezeket később rendezni kell. A kódban alapvetően nem a téma elnevezése lenne elől, hanem a színezés kódja, ami hibás rendezést eredményezne. Ezért a karaktersorozat a téma megnevezésével kezdődik érvényes *HTML* kommentbe helyezve:

```
//////////////// JELMAGYARAZAT OSSZEALLITASA + TEMASZINEK LEMENTESE ///////////////

    // temak kigyujtese
    $tabla = "SELECT * FROM temak";
    if ($result = $conn->query($tabla)) {
    $jelek=array();
    $jelszin=array();
    $temakereso=array();
    // temanevek megszinezese es mentese
        while($row = $result->fetch_assoc()){
                $jelek[] = "<!-- ".$row["nev"]." --><span style=\"color: #".$row[
                "hexa_szin"].";\">".$row["nev"]."</span>   ";
                $jelszin[$row["id"]] = $row["hexa_szin"];
                $temakereso[]=$row["nev"];
                $temakereso_idk_nevbol[$row["nev"]]=$row["id"];
        }
        // eroforras felszabaditata
        $result->free();
    }
    usort($jelek, "hunsort");
```

Szűrő

A felhasználónak lehetősége van szűrési feltételeket megadni (témát ésvagy konkrét címkét), hogy könnyebben megtalálhassa a kedvére való bejegyzéseket. (FI00, FI01) Mivel az oldal minden újratöltéskor lekérdezi a bejegyzéseket, elegendő SESSION változókba elmenteni a megadott szűrési feltételeket, amik felhasználhatóak ebben a lekérdezésben közvetlenül. A változók a teljes munkafolyamat alatt megmaradnak, törlésük az 'Összes' gomb megnyomásával lehetséges. (FI02)

MySQL parancs kibővítésével és a szűrési feltételek elküldésével könnyedén megvalósítható a szűrés:

```
        $szuro="";

        if(isset($_SESSION["tema"])) if($_SESSION["tema"]!=0) $szuro.=" AND
        tema_id=\"".$_SESSION["tema"]."\"";
        if(isset($_SESSION["cimke"])) if($_SESSION["cimke"]!=0) $szuro.=" AND
        cimke_id=\"".$_SESSION["cimke"]."\"";
        if(isset($_SESSION["ajanlott"])) if($_SESSION["ajanlott"]==true) $szuro
        .=" AND ajanlott=\"1\"";

        $tabla_bejegyzeslista .= $szuro;
```

Címlista

A címek listája az 'Információk' alfejezethez hasonló módon *HTML* kódokból áll. Jelen esetben a bejegyzés sorszáma, címe és dátuma a témakörének megfelelő színezéssel, sorszámmal, plusz hivatkozással. Külön oda kell figyelni a nulladik év tartalmának sorszámozására, illetve, hogy egyes bejegyzéseket a felhasználó nem láthat. (NA00, NA01, NA03, NA04, NA06, HS05)

```
// CIMLISTA OSSZEALLITASA
for ($i=0; $row_bejegyzes = $result_bejegyzes->fetch_assoc(); $i++) {
    $ev = $row_bejegyzes["ev"];

    if($_SESSION["felhasznalo_hozzaferesi_szintje"]>=$hozzaferes_szintjei_idbol[
    3])
        // sorszamozas (nulladik evnek kulon)
        if($row_bejegyzes["ev"]<1){
            $realsorszam=$row_bejegyzes["sorszam"]+12;
            $veglegessorszam="0.".$realsorszam.". ";
        }
        else $veglegessorszam=$row_bejegyzes["sorszam"].". ";
    else $veglegessorszam="";

    // hivatkozas hozzaadasa ha megtekintheti
    if($hozzaferes_szintjei_idbol[$row_bejegyzes["hozzaferesi_szint"]]<=
    $_SESSION["felhasznalo_hozzaferesi_szintje"]){
        $referencia = "href=\"naplo.php?bejegyzes=".$row_bejegyzes["sorszam"].
        "#bejegyzes\"";
        if($row_bejegyzes["ajanlott"]==1) $kiemeles="font-weight: bold;";
        else $kiemeles="";
    }
    else $referencia = "";

    // ellenorizetlen bejegyzes megjelolese
    if($row_bejegyzes["leellenorzott"]==0){
        $ell="*";
    }
    else $ell="";
```

Megjelenítés során évek és fejezetsorszámok ékelődnek a címek közé. Ehhez hasznosak a lekérdezésekkor lementett hash táblák és változók. Akkor kerül kiírásra az év és a fejezetet, ha változást észlelünk, így minden év és fejezet csak egyszer íródik ki. (NA05)

```
if(!empty($lista_sorszamok))    {
    sort($lista_sorszamok);

    $ev=-1;
    $fejezet=-1;
    $evek=$lista_ev_sorszambol[$lista_sorszamok[count(
    $lista_sorszamok)-1]];

    for($i=count($lista_sorszamok)-1; $i>=0; $i--){

        $j=$lista_sorszamok[$i];

        if($ev!=$lista_ev_sorszambol[$j]){
            $ev=$lista_ev_sorszambol[$j];
            print "<br/>";
            print "<b><a name=\"bejegyzes\" id=\"".$ev."\">".$ev
            .". Év tartalma </a></b>";
            print "<br/>";
        }
        if($fejezet!=$lista_fejezet_sorszambol[$j]){
            $fejezet=$lista_fejezet_sorszambol[$j];
            print "<br/>".$fejezet.". fejezet<br/><br/>";
        }
        print $lista_bejegyzes_sorszambol[$j];
    }
}
```

Az alábbi képeken egy csak publikus és egy teljes címlista részlete látható:

Bejegyzések

3. Év tartalma

11. fejezet

Funkciók 2014-05-16
Vágyak III. 2014-04-26

2. Év tartalma

9. fejezet

Okosság 2013-07-06

8. fejezet

Lélek 2013-03-13
Igazság 2013-03-03

1. Év tartalma

6. fejezet

Emlékek 2012-07-08
Vártság 2012-07-04
Szavak 2012-07-02

5. fejezet

Vágyak II. 2012-05-16
Vágyak 2012-05-06
Világnézetek 2012-05-02
Empátia 2012-04-30
Beszélgetés 2012-04-19

3. Év tartalma

12. fejezet

156. Évzárás III. 2014-07-23
155. Publikusság IV. 2014-07-22
154. Önértékelés III. 2014-07-21
153. Szórakozás III. 2014-07-20
152. Értékrend II. II. 2014-07-19
151. Ellentmondás II. 2014-07-18
150. Antiszoci IV. 2014-07-17
149. Lazítani II. 2014-07-16
148. Tiszta lappal 2014-07-15

11. fejezet

147. Funkciók 2014-05-16
146. Publikusság III. 2014-05-11
145. Lehetőségek 2014-05-10
144. Kijutás 2014-05-03
143. Halál 2014-04-27
142. Vágyak III. 2014-04-26
141. Célok 2014-04-06
140. Határok 2014-04-01
139. Izgalmak 2014-03-29
138. Borítókép 2014-03-17
137. Jellemzőim II. 2014-03-16
136. Életpálya II. 2014-03-15
135. Életpálya 2014-03-14

Kiválasztott bejegyzés fejléce

A kiválasztott bejegyzés adatainak egy része sima szöveg, mint a cím, a dátum vagy a tartalom, de a címkék speciálisak. Ha egy címkére kattintunk, az oldal újratöltődik, de ez alkalommal csak azok a bejegyzések jelennek meg a címlistában, amik az adott címkével rendelkeznek. Ez lényegében a szűrő egy közvetett használata. (FI03, NA08)

Amikor rákattintunk egy címkére, az POST üzeneteket küld a *naplo.php* oldalnak. Ezt csak form segítségével sikerült megoldani. Olyan címkelista van összeállítva, ami egy érvényes HTML form egyben. (A témák kigyűjtéséhez hasonló módon itt is HTML kommenttel kezdődik minden sor, hogy helyesen lehessen sorrendbe tenni az elemeket.) CSS használatával lehet elérni, hogy ne gombnak látszódjon, csak egy egyszerű szövegnek (class=”postlink”).

```
<!-- Kiszámíthatóság -->
<form action="naplo.php?bejegyzes=107#bejegyzes" method="POST">
<input type="hidden" name="cimke" value="13">
<input type="submit" class="postlink" value="- Kiszámíthatóság"/>
</form>
```

A címkék után a sorban következnek a bejegyzés szövegében megemlített bejegyzéscímek. Ezeknek a megvalósítása a ’Kiválasztott bejegyzés szövege’ alfejezetben található.

Kiválasztott bejegyzés szövege

Talán a legegyszerűbb feladat a kiválasztott bejegyzés tartalmának beillesztése. Az adatok lekérdezésekor a szöveg szakaszai egyesítve lettek, a privát részeket kicenzúrázva. (HS03)

A kihívás abban rejlik, hogy az író gyakran hivatkozik egyéb bejegyzéseire. Ezekre érdemes hivatkozásokat elhelyezni, hogy a felhasználó gyorsabban megtalálja a kapcsolódó bejegyzéseket. A bejegyzéscím felismerésének a titka, hogy a cím mellett szerepel a dátum is. Így elegendő az általános „kötőjeles” formátumú dátum kötőjeleit megtalálni, és kinyerhető a dátum, amivel már lekérdezhető a cím (hiszen mindkettő egyedi). Ezek után már csak ki kell cserélni a szövegben található cím-dátum párost egy formázott változatra, ami aztán a címkék végére is bekerül alfanumerikus sorrendben. (NA07, NA08)

```
// HIVATKOZASOK ELHELYEZESE A SZOVEGEKBEN

$elofordult_mar[]=array();
$elofordult;
$cimkek2=array();

for($i=0; $i<strlen($szovegreszek); $i++){

    if($szovegreszek[$i]=="-"){
        if($szovegreszek[ ($i - 3) ] == "-"){
            // bejegyzesre utalo datumot talalt
            $tabla = "SELECT * FROM bejegyzesek WHERE datum=\"".substr(
            $szovegreszek, $i-7, 10)."\"";
            // kikeresi a datumhoz tartozo bejegyzest
            if ($result = $conn->query($tabla)){
                $row = $result->fetch_assoc();

                if(true){
                    // korabban elhelyezett azonos hivatkozast keres
                    $elofordult=0;
                    for($j=0; $j<count($elofordult_mar); $j++)
                    {
                        if($elofordult_mar[$j]==$row["datum"]) $elofordult=1;
                    }
                    // akkor csereli le ha meg nem volt
                    if($elofordult==0 and isset($lista2[$row["ev"]][$row[
                    "fejezet"]][$row["sorszam"]])){
                        // elso ev miat korrekcio szukseges
                        $linkelt_bejegyzes = $lista2[$row["ev"]][$row["fejezet"
                        ]][$row["sorszam"]];
                        $cimkek2[]=$linkelt_bejegyzes; //listaba menti
                        $elofordult_mar[]=$row["datum"]; //megjegyzi
                        // elhelyei a hivatkozasokat
                        $szovegreszek = str_replace($row["cim"]." ".$row["datum"
                        ] ,$linkelt_bejegyzes , $szovegreszek);

                    }
                }
                $result->free();
            }


        }
    }
}

$szovegreszek = str_replace("\\","",$szovegreszek);

// rendezes
usort($cimkek2, "hunsort");
usort($cimkek, "hunsort");
```

Hozzászólás

Az olvasónak lehetősége van üzenni az írónak. (KO00) Egy egyszerű modul közepén egy szövegdobozzal. Az üzenet megírása után gombra kattintással a szöveg lementődik az adatbázisba, és frissül az utolsó üzenetküldés dátuma.

Az üzenésnek korlátai vannak. Egy nap csak egyszer lehet írni. Ha már megtörtént, a modul nem jelenik meg többet aznap. (KO01) Ebből az okból kifolyólag a HTML kód PHP segítségével kerül kiírásra, ha a feltétel teljesül.

```
                // HOZZASZOLAS MEZOJE

                if($_SESSION["felhasznalo_hozzaferesi_szintje"]>
                $hozzaferes_szintjei_idbol[1]){


                $tabla_hozzaszolas = "SELECT utolso_hozzaszolas FROM regisztraciok
                WHERE nev=\"".$_SESSION["felhasznalo"]."\" LIMIT 1";
                    if ($result_hozzaszolas = $conn->query($tabla_hozzaszolas)) {
                        $row_hozzaszolas = $result_hozzaszolas->fetch_assoc();

                        // ma meg nem kuldott uzenetet
                        if($row_hozzaszolas["utolso_hozzaszolas"]!=date("Y-m-d")){

                            print
```

A szöveg nem lehet 1000 karakternél hosszabb, ez külön vizsgálatra kerül elküldés után. Ha az üzenetküldő helytelenül járt el, hibaüzenetet kap. (KO02)

```
////////////////// HOZZASZOLAS ////////////////////////

    // kuldve lett ervenyes hozzaszolas
    if(!empty($_POST["hozzaszolas"]) and !empty($_SESSION["felhasznalo"])){

        // valtozo biztositasa
        $hozzaszolas = $_POST["hozzaszolas"];

        // HOZZASZOLASI IDO ELLENORZESE

        $tabla = "SELECT utolso_hozzaszolas FROM regisztraciok WHERE nev=\"".$_SESSION[
        "felhasznalo"]."\" LIMIT 1";

        // komplikalt vizsgalata az uzenet hosszanak: vacakolt, de igy most MUKODIK!
        $hossz=strlen($hozzaszolas);
        $hosszabb=strlen($hozzaszolas) - 1000;
        $sysmsg=$hossz;
        if($hossz < 1001) $sysmsg=1;
        else $sysmsg=0;
        if($sysmsg==0 ){
            $sysmsg="Sajnos az üzeneted ".$hossz." hosszú, ami ".$hosszabb."
            karakterrel több a megengedettnél!";
        }
```

Weboldal formázása

A weboldal minimális *CSS* formázást is használ. Ez a *forma.css* fájlban került lekódolásra. A különböző formázások felhasználás gyakorisága alapján a vannak csoportosítva.

Általános formázások

Vannak mindenre kiterjedő beállítások. Ez a honlap BODY részének formázását jelenti. (Továbbá minden margó és szegély eltűntetésre került.)

```
BODY {
    font-family: "Comic Sans MS", Arial, sans-serif;
    font-size: 16px;
    margin:0 0 0 0;
    padding: 0 0 0 0;
    background: #080;
    text-align: center;
}

*{
    margin: 0;
    padding: 0;
}
```

Gyakori formázások

Vannak olyan beállítások, amik elszórtan a weboldal teljes területén előfordulnak: Minden oldalon szerepel a weboldal címe. A főbb egységek keretbe vannak foglalva. A keretekben időnként elválasztó vonal is előfordul. Vannak sötét háttérszínnel kiemelt fejlécek, amik általában a keret tetején helyezkednek el. A főbb egységek kitölthetik a képernyőt teljes szélességben, de lehet vékonyított is. Ezek alapján a formázások a következőek:

```
.focim {
    font-size:96px;
    color:#909;
    font-weight: bold;
    text-align: center;
    text-shadow: 0 0 70px #FF0;
}

.keret {
    border: 5px ridge #909;
    background: #FFF;
    text-align: justify;
}

.belsoszegely{
    padding: 5;
}

.kitoltes {
    border: 5px #909;
    background: #909;
    color: #FF0;
    font-weight: bold;
    font-size: 16px;
    text-align: center;
    padding:5;
}

.globalcontainer {
    width: 90%;
    min-width: 800px;
    margin: 0 auto;
}
.vekonycontainer {
    width: 50%;
    min-width: 400px;
    margin: 0 auto;
}
```

Speciális formázások

Vannak ritkábban használt speciális formázások. Helyenként előfordul, hogy az oldal egyes információkat szürkített apró betűs formázással közöl. Ezek mellékes információk, ilyen például regisztrációkor a maximálisan megadható karakterek száma.

Másik eset, amikor nem elrejteni, hanem kiemelni akarunk egy információt. Ehhez rendelkezésre áll figyelemfelkeltő formázás, amivel általában a hibaüzenetek szoktak megjelenni.

Végül a fejlécben felsorolt címkéknél használt trükk is ebbe a kategóriába tartozik. Gombon a formázás eltűnteti annak textúráját. A gomb továbbra is megőrzi funkcióját, de a kinézete egyszerű szöveg.

```
.aprobetus{
    font-size: 12px;
}

.figyelem{
    color: #F00;
}

.postlink{
    background: transparent;
    border: none !important;
    cursor: pointer;
}
```

Weboldal fő felépítése

Az utolsó a weboldalt teljes egészében felépítő, az alapelrendezését megadó formázási típus. Ez hierarchiában a nagy egésztől a legkisebb modulig igyekszik elrendezni az oldal teljes tartalmát.

```
.fejlec {

}
    .felhasznalo {
        height: auto;
    }
        .belepes {
            width: 600px;
            float: left;
            text-align: left;
            height: auto;
        }
        .regisztracio {
            margin-left: 600px;
            text-align: right;
            height: auto;
        }
    .szabalyok {

    }
        .koszonto {

        }
        .segitseg {font-family: "Times New Roman", sans-serif;
            border-bottom: 5px ridge #909;
            padding: 5;
        }
        .jelmagyarazat {
            padding: 5;
        }
```

```
.torzs {
    font-size: 14px;
}
    .lista {
        float: left;
        width:290px;
    }
    .bejegyzes {
        margin-left: 315px;
    }
        .tartalom {
            border-bottom: 5px ridge #909;
            padding: 5;
        }
        .cimeressor{

        }
            .cimkek {
                min-height:250px;
                margin-right: 255px;
                border-right: 5px ridge #909;
            }
            .cimer {
                margin-left: 220px;
                float:top;
            }
        .bejegyzesszoveg {
            font-family: "Times New Roman", sans-serif;
            float: bottom;
            padding: 5;
        }

        .balratart {
            width: 90%;
            float: left;
            height: auto;
        }
        .jobbratart{
            text-align: left;
        }
```

Egyéb oldalak

A weboldalnak vannak különösebb funkcióval nem rendelkező oldalai, illetve kiegészítő kódjai, amik külön fájlba lettek megírva.

index.php

Az író címerével díszített kezdőoldal. Egy linkre kattintva jutunk a *naplo.php* oldalra. A link szövege erkölcsi alapon igyekszik kiszűrni a nem kívánt vendégeket.

info.php

Tájékoztató oldal a felhasználók számára. Minden hozzáférési szint, témakör és címke felsorolásra kerül táblázatos formában, és mindegyikhez leírás is tartozik. Az információk mindegyike kivétel nélkül az adatbázisból kerül lekérdezésre. Az oldal alján a felhasználó megtalálhatja a saját hozzáférési szintjét is. (IN00)

Kiegészítő oldalak

adatbazisnyitas.php

Az esetek többségében az új oldal betöltése adatbázis-lekérdezéseket igényel. Az adatbázis nyitásának egységes kezelése érdekében ez külön fájlban kap helyet, ami hozzáfűzendő minden adatbázist használó oldalhoz.

```
// CSATLAKOZAS
$conn = new mysqli("...", "...", "...", "...");

// KAPCSOLAT ELLENORZES
if ($conn->connect_errno) {
    printf("Hiba történt az adatbázishoz csatlakozás közben. Hibaüzenet: %s\n", $conn->
    connect_error);
}

// KODOLASI BEALLITASOK
mysqli_query($conn, "SET NAMES 'UTF8'");
mysqli_query($conn, "SET CHARACTER SET UTF8");
```

fejlecextra.php

Minden oldal fejlécében előforduló kódok kerülnek ide kihelyezésre. Ilyen a CSS használata és a karakterkódolás beállítása.

```
print "

<!-- CSS hasznalat -->
<style type=\"text/css\">
    @import url(\"forma.css\");
</style>

<!-- karakterkodolas -->
<meta charset=\"utf-8\" />";
```

hunsort.php

Gyakori probléma, hogy rendezés során az egyszerű rendezés nem képes kezelni a magyar karaktereket. Ez az oldal olyan rendező függvényt tartalmaz, amivel már a magyar karakterek is helyes sorrendbe kerülnek.

```
function hunsort($a, $b)
{
    static $Hchr = array('á'=>'az', 'é'=>'ez', 'í'=>'iz', 'ó'=>'oz', 'ö'=>'ozz', 'ő'=>
    'ozz', 'ú'=>'uz', 'ü'=>'uzz', 'ű'=>'uzz',  'Á'=>'az', 'É'=>'ez', 'Í'=>'iz', 'Ó'=>
    'oz', 'Ö'=>'ozz', 'Ő'=>'ozz', 'Ú'=>'uz', 'Ü'=>'uzz', 'Ű'=>'uzz');

    $a = strtr($a,$Hchr);    $b = strtr($b,$Hchr);
    $a=strtolower($a); $b=strtolower($b);

    return strcmp($a, $b);
}
```

mindenelott.php

Bármit is kezdenénk kódolni, mindenekelőtt el kell indítani a munkafolyamatot, és ellenőrizni kell, hogy van-e a felhasználónak hozzáférési szintje. Ha nincs, megkapja a legalacsonyabbat. (HS00)

```
session_start();

$sysmsg=""; // ide lehet a felhasznalo fele tovabbitando uzeneteket irni

if(empty($_SESSION["felhasznalo_hozzaferesi_szintje"])) $_SESSION[
"felhasznalo_hozzaferesi_szintje"] = 1;
```