

Ninjastic

Felhasználói
dokumentáció

Nyilatkozat

Jancsik Balázs Zoltán, Erdős Attila és Csendes Marcell nyilatkozunk arról, hogy a szakdolgozatot mi hárman csináltuk, nem másoltuk, a hivatkozásokat megfelelően használtuk, máshol még nem adtuk be és meg nem engedett segédanyagot nem használtunk.

Tartalomjegyzék

Nyilatkozat.....	2
Bevezetés, témafelvetés	5
Szakirodalmi áttekintés	5
Apache.....	5
MySQL	5
PHP	6
Fórum jelentősége.....	7
Anyag és módszer.....	7
Adatbázis tervezés elmélete	7
Programozás elmélete.....	8
Adatbázis tervezése.....	8
Admin felület.....	12
Bejelentkezés.....	12
Dashboard	13
Felhasználók	13
Listázás	13
Létrehozás/Szerkesztés	14
Törlés	14
Topikok	15
Listázás	15
Létrehozás/Szerkesztés	15
Törlés	16
Hozzászólások.....	16
Listázás	16
Létrehozás / Szerkesztés	17
Törlés	17
Csúnya szavak.....	18
Listázás	18
Létrehozás / Szerkesztés	18
Törlés	19
Adminok	19
Listázás	19
Létrehozás / Szerkesztés	20
Törlés	20

Bevezetés, témafelvetés

A veszprémi egyetem hallgatójaként a számítástechnika és a programozói világ számos ágával volt lehetőségünk találkozni. Bővül a lexikális tudásunk és szerencséseknek mondhatjuk magunkat, hogy tanáraink rengeteg időt fordítottak gyarkorlati képzésünkre is. A szakdolgozat elkészítése során elsősorban az így megszerzett ismereteket, a magyar és angol szakirodalmat, valamint internetes szakanyagokat használtam fel. Dolgozatunk témájának egy fórum elkészítését választottuk.

Napjainkban a világ egyik legelterjedtebb kommunikációs csatornája az interneten a fórum, ahol mindenkinek lehetősége van akár segítség kérésre vagy nyújtásra. Például visszanézhetőek korábbi beszélgetések, amely a mai fiatal generációnak segítséget nyújthat a pályakezdésükben és a felmerülő kezdetleges problémákban.

Rengeteg fórumot láttunk már, de magunktól még sosem készítettünk el egyet sem. Úgy gondoltuk, ezzel a feladattal érdemes foglalkozni. Mindemelett HTML, PHP, MySQL, JavaScript, JQuery, Ajax, Symfony, Docker, Vue Js, Twig és sok egyéb technológia felhasználását bevettük.

A jövőben fórum, webshop, blog, ERP és egyéb típusú web és kózoszolos alkalmazások fejlesztésével szeretnénk foglalkozni. Ezen belül a PHP frameworkok, JavaScript frameworkok, Microservicek és egyéb kommunikációs megoldások megvalósításával.

Szakirodalmi áttekintés

Apache

Az Apache HTTP Server (röviden Apache) nyílt forráskódú webkiszolgáló alkalmazás. Az Apache program jellemzőivel párhuzamba hozható az amerikai indián törzsek találékonysága és alkalmazkodási képessége. Ez egy ingyenes program, amely fontos szerepet játszott az Internet elterjedésében. A projekt célja, olyan webszerver szoftver megalkotása és fejlesztése, amely eleget tesz a rohamosan fejlődő világ és az Internet igényeinek, biztonságos, üzleti-, vállalati felhasználásra is alkalmas és emellett szabadon használható. Az Apache a régi NCSA HTTPd szerverre épül és az Apache Szoftver Licenc feltételeivel terjesztik. Az első webszervert egy angol fizikus, Tim Berners-Lee, alkotta meg. Az Apache sok szabványt támogat, melyeknek nagy része lefordított modulok formájában áll rendelkezésre. Ezek a modulok sok területet lefednek, a kiszolgáló oldali programnyelv támogatástól kezdve a hitelesítési sémáig. Az ismertebb, támogatott programnyelv modulok a Perl, Python, Java, Tcl és a Php. A legnépszerűbbek modulok a mod_access, mod_auth és a mod_digest. Statikus és dinamikus weboldalak közzétételére egyaránt használják. Sok webalkalmazást az Apache által nyújtott környezethez és szolgáltatásokhoz terveznek. Az Apache alkotja a webszerver komponenst a népszerű LAMP alkalmazáscsomagban, melynek további komponensei a MySQL adatbázisszerver és a PHP/Perl/Python programozási nyelvek.

MySQL

A MySQL egy több felhasználós, többszálú, SQL-alapú relációs adatbázis-kezelő szerver. A szoftver eredeti fejlesztője a svéd MySQL AB cég, amely kettős licencezéssel tette elérhetővé a MySQL-t. Választható módon vagy a GPL, vagy egy kereskedelmi licenc érvényes a felhasználásra. 2008 januárjában a Sun felvásárolta 800 millió dollárért a céget. 2010. január 27-én a Sun az Oracle Corporation tulajdonába került. A MySQL az

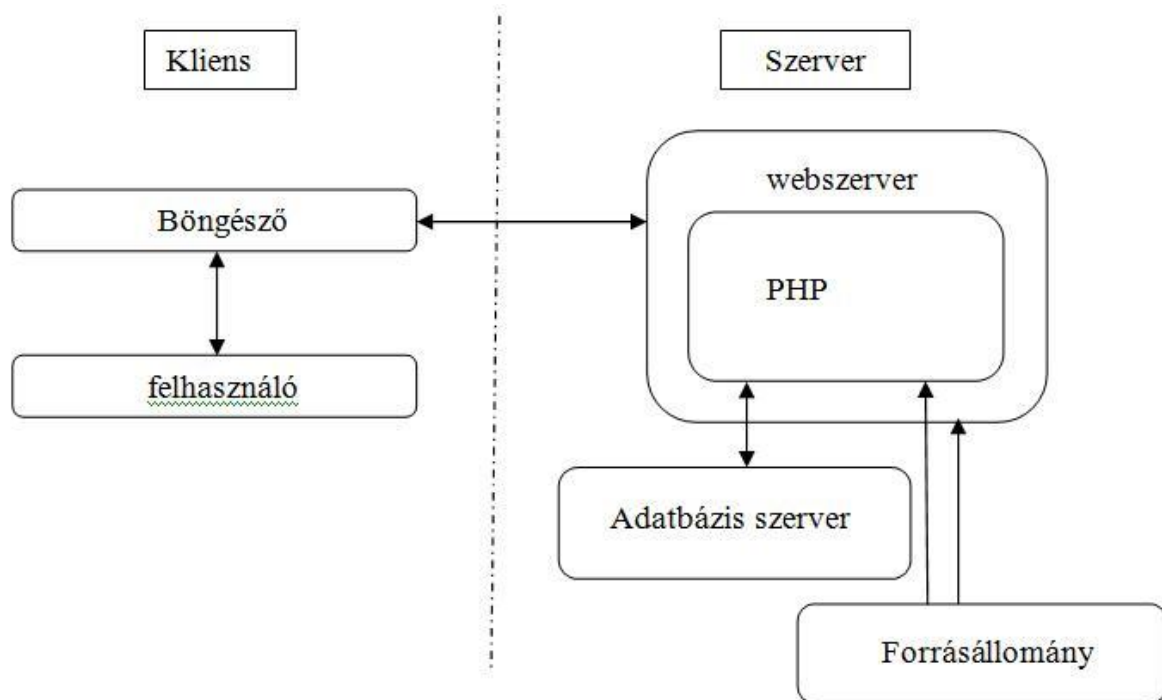
egyik legelterjedtebb adatbázis-kezelő, aminek egyik oka lehet, hogy a teljesen nyílt forráskódú LAMP összeállítás részeként költséghatékony és egyszerűen beállítható megoldást ad dinamikus webhelyek szolgáltatására.

PHP

A PHP általános szerver oldali szkript nyelv, melyet dinamikus weblapok megalkotására hoztak létre. Az első szkript nyelvek egyike, amely külső fájl használata helyett HTML oldalba ágyazható. A kódot a webszerver PHP processzor modulja értelmezi, ezzel dinamikus weboldalakot hoz létre. Rasmus Lerdorf 1995-ben indította útjára. Ma a The PHP Group tartja fenn és fejleszti. A PHP szabad szoftver, de licence nem csereszabatos a GNU licensszel, mivel megkötéseket tartalmaz a PHP név használatára. A PHP nyelv túlnőtt eredeti jelentőségén. Születésekor csupán egy makrókészlet volt, amely személyes honlapok karbantartására készült. Innen ered a neve is: Personal Home PageTools. Később a PHP képességei kibővültek, így egy önállóan használható programozási nyelv alakult ki, amely képes nagyméretű webes adatbázis alapú alkalmazások működtetésére is.

A böngésző és a szerver közötti kommunikáció folyamata

A szerver-kliens kapcsolat kétirányú. A böngészőbe a felhasználó beírhat egy URL-t, rákattint egy linkre, ekkor egy úgynevezett GET metódusú kérést küld el a szervernek. Ezen felül az űrlap kitöltése után az elküldés gombra kattintás lehetőséget ad arra, hogy POST (vagy GET) metódusú kérést küldjünk a szervernek illetve onnan választ is kapjunk a böngészőn keresztül. A kérés tulajdonképpen egy adatcsomag, ami több lépésből áll. Először a kérésben elmegy az URL, a metódus fajtája POST vagy GET, a használandó protokoll HTTP 1.0 vagy HTTP 1.1. Ezt követően POST metódus esetén a kérésben elmennek az űrlapban kitöltött adatmezők adatai, majd a böngésző neve, és speciális tulajdonságai is. Ezután a böngésző által lekezelni képes speciális adatfajta szabványos nevei kerülnek továbbításra. Majd a böngészőben tárolt és az adott domainhez tartozó COOKIE-k adatai következnek. Ha az URL a szerveren .html, .htm, .gif, .mpeg, stb... file-t jelöl meg, akkor a webszerver kikeresi a saját háttértárán a megfelelő fájlt és elküldi a böngészőnek különböző plusz, úgynevezett fejléc adatok (HEADER) kíséretében. A HEADER-ben lévő adatok a következők lehetnek. A kódolás nyelve, a cache-ek működésére vonatkozó parancsok, az esetlegesen elküldendő COOKIE-k, a SESSION-ökhöz tartozó úgy nevezett SESSION COOKIE-k. Ha az URL .php kiterjesztésű file-t jelöl, akkor a webszerver átadja az URL-t a PHP motornak. Itt legelőször a PHP motor megkeresi a háttértáron a megadott file-t. Ezután betölti, szintaktikailag ellenőrzi a betöltött php file tartalmát, majd elkezd értelmezni és végrehajtani azt. Ha szükséges elindít a PHP-hez konfigurált egyéb modulokat, kommunikál velük, mint egy SQL szerver. A PHP kód segítségével a kimenetre lehet küldeni HEADER parancsokat, COOKIE-k és SESSION-ök adatait. Ez a folyamat a PHP motor HTML kód küldése előtt történik. A programírás során, a PHP oldalakon keverni lehet a HTML és a PHP kódot. Ha az értelmező HTML kód részt talál, akkor ezt a kódot küldi a kimenetre változtatás nélkül. A PHP motor működése során kiíró utasításokkal előállítja a HTML forráskódot és elküldi a kimenetre. Ezt az információt a webszerver kapja meg. A webszerver a tartalmat tovább juttatja a böngészőnek. A böngésző a megkapott információt betölti és értelmezi. Ha a kódban jó JavaScript, Flash, Java vagy egyéb nem kifejezetten HTML kód található, akkor a böngésző elindítja a kliens számítógépre telepített megfelelő feldolgozó modult.



1. ábra: Szerver-kliens kapcsolata (forrás: saját ábra)

Fórum jelentősége

A fórumok lehetőséget biztosítanak a felhasználók számára arra, hogy szöveges párbeszédet folytassanak egymással. Itt az üzeneteket témák szerint csoportosítják, ami a jobb átláthatóságot segíti elő. A „bulletin board” az Internet elterjedését megelőző korszak legnépszerűbb elektronikus adatcsere rendszere volt, amelyben egy adott téma iránt érdeklődő felhasználók cserélhettek üzeneteket és állományokat egymással. Képernyőn való megjelenítése kizárólag szöveges jellegű volt, elérésére közvetlenül a kiszolgálóra történő telefonos betárcsázás útján nyílt lehetőség. Manapság az angol "bulletin board" kifejezést a magyar nyelvben "fórum" név helyettesíti, amit általában webes alapú üzenetcsere lehetővé tevő rendszerek megnevezésére használnak. A fórumok többféleképpen épülhetnek fel. Többnyire minimálisan két szintből állnak. Ezekben általában az a közös vonás, hogy először különböző kulcsszavak, címek alapján bontják szét a tartalmat, majd ezeken belül találhatjuk meg a hozzászólásokat.

Anyag és módszer

Adatbázis tervezés elmélete

Az adatbázisok tervezését és kivitelezését egy összetett rendszerszervezési és elemzési folyamat előzi meg. Nagyon lényeges, hogy a tervező megismerje az adatbázisban eltárolandó adatok természetét. A normalizálás eredményét általában három lépéssel érhetjük el. A normalizálás lényegében táblázatsztémbontó relációs műveletek sorozata, amelyek eredményeként egymással kapcsolatban álló, az eredetinel kisebb tárolási igényű relációkat kapunk. A szétbontás haszna nagyjából így foglalható össze. Csökken a tárolási igény (bizonyos adatokat nem tárolunk feleslegesen többször is), megszűnnek a törlési, módosítási és beszúrási problémák (anomáliák) és logikailag áttekinthetőbb lesz a táblázat. A normalizálás folyamata egzakt matematikai elveken alapuló művelet. Fontos megismerni néhány matematikai fogalommal, mely a normalizálás folyamán előfordulnak. Funkcionális függőség P-attribútum részhalmaz funkcionálisan meghatározza Q-attribútum halmazt, ha minden P-hez tartozó

értékhez (a P attribútumainak oszlopaiban álló értékekhez) pontosan egy Q-hoz tartozó értéket tudunk hozzárendelni. Tehát P értékei egyértelműen meghatározzák Q értékeit. Teljes funkcionális függőség: Azt mondjuk, hogy Q teljes funkcionális függőségben van P-vel, ha P-ből képzett attribútum részalmazoktól nem függ a Q attribútum halmaza. Ellenkező esetben részleges függőségről beszélünk. Transzitiv funkcionális függés "A" attribútum részalmaztól transzitiv funkcionális függőségben van "C" attribútum részalmaz, ha létezik egy "B" részalmaz, amely "A"-tól függ, a "C" pedig "B" től. Tehát mondhatjuk, hogy a két funkcionálisan függő attribútum halmaz mellett található egy harmadik részalmaz, mely a két halmaz közötti funkcionális függőséget átviszi (tehát direkt módon is függnek egymástól és áttételesen is). A normalizálás végrehajtása során egymásra épülő, egyre kevesebb redundanciát tartalmazó normálformák keletkeznek. 6 egymásra épülő normálformát ismerünk: 1NF, 2NF, 3NF, BCNF, 4NF, 5NF azaz első, második, harmadik, Boyce-codd normálforma, negyedik és ötödik normálforma. Az első három normálforma a funkcionális függőségekben található redundanciák, míg a negyedik és ötödik a többértékű függőségekből eredő redundanciák megszüntetésére szolgál. Gyakorlatilag a 3. normálformában lévő adatbázis redundancia mentesnek tekinthető. Az esetek többségében elegendő a normalizálást a 3. normálformáig elvégezni. 0NF-ben az adatbázis nem normalizált alakját szokás 0.-dik normálformának nevezni. A tervezett adatbázisunk akkor van első normálformában, ha minden sora különböző, az oszlopok száma, sorrendje minden sorban azonos, minden oszlop csak egy attribútum értéket vesz fel és minden sorhoz egy egyedi kulcs tartozik, amitől az összes többi attribútum funkcionálisan függ. Második normálformában akkor van, ha első normálformában van (előfeltétel) és a nem kulcs attribútumok funkcionálisan teljesen függenek az elsődleges kulcstól. Harmadik normálformában akkor van, ha második normálformában van és funkcionális függés csak az elsődleges kulcsból indul ki; vagyis megszüntettük a közvetett (transzitiv) függéseket. A normalizálási folyamat során a funkcionális függőségek már csak a kulcstól valósulnak meg. A redundancia csökken, a relációk száma nő.

Programozás elmélete

Programtervezésen a leendő program szerkezetének megalkotását értjük. A program elemeinek megfelelő sorrendbe rakása alapvető. A szerkezetében változók, műveletek, ciklusok, utasítás-sorok, függvények, eljárások, alprogramok találhatóak. Programtervezés során az első dolgunk, hogy a problémát meghatározzuk. Ehhez fel kell mérnünk a feladatot, mit szeretnénk megvalósítani és mik lehetnek az akadályok. Ha tudjuk a feladatunkat, akkor a következő teendőnk a rendszertervezés és modellalkotás. A program struktúráját itt készítjük el vagy képzeljük el. Ha a struktúra jó, akkor azt követi a harmadik lépés, ahol a kódolás történik, azaz a programírás. A programírást követően a tesztelés következik, ahol próbáljuk megtalálni a hibákat olyan értékekkel, amik elképzelhetetlenek, de azokra is működnie kell a programnak. Ha hibákat is sikerült kijavítani, akkor az utolsó lépés a dokumentáció elkészítése. A programunknak mindenféleképpen megbízhatónak kell lennie, megfelelően kell működnie. A jó terv lecsökkenti az idő- és ez által költségigényes tesztelési fázist. Egyébként a jól megtervezett program módosítása is egyszerűbb.

Adatbázis tervezése

A weblapomhoz készítettem egy adatbázist, ehhez a symfonynak a migrációját használtam. Ebben van lehetőség adatbázisok létrehozására és mód van a hozzá szükséges táblák, valamint az azokat tartalmazó mezők generálására. Adatokkal fel tudjuk tölteni az adatbázisunkat. Az adatbázisomat „app” néven lehet elérni. Öt darab táblája van, melyek a következők: „admin”, „comment”, „dirty_word”, „topic”, „user”. Minden egyes tábla és mező nevében használtam azt a szabályt, hogy a neve nem tartalmazhat csak angol karaktereket, számokat és pár egyéb karaktert. Mivel igyekeztem beszédes neveket választani, ezért ezek nem okoznak értelmezési

nehézségeket. Így a későbbiekben - akár más által is - könnyen áttekinthető és ez által továbbfejleszthető szerkezetet készítettünk.

Az „admin” tábla azért szükséges, hogy itt tudjunk új adminisztrátorokat rögzíteni. Akik segíteni tudják a weboldal működését. Az admin felhasználók tudnak csak belépni az admin oldalra és csak azok akik aktívak. Az „admin” táblánk oszlopai a következők:

1. **id:** Ez egy azonosító, ami egy int típusú érték (egész szám). Az értékét automatikusan növeli minden egyes új rekordnál. Elsődleges kulcs. Erre lehet hivatkozni minden egyes lekérdezésnél, amikor a „admin” táblának egy adott rekordjából információt szeretnénk kinyerni.
2. **name:** Ez szöveg mező, 255 karaktert tartalmazhat. Ez egy adott adminisztrátornak a nevét tárolja. Ez azért szükséges, hogy megtudjuk mutatni neki, hogy ő van éppen belépve az admin felületre.
3. **email:** Ez szöveg mező, 160 karaktert tartalmaz, mert hivatalosan ennél hosszabb nem lehet. Ez a adminisztrátor email címét tárolja, amivel be tud jelentkezni az admin felületre.
4. **status:** Ennek a mezőnek a típusa „int”. Ez határozza meg, hogy adminisztrátor beléphet ez az admin felületre vagy sem. Ha az értéke 0 akkor nem léphet be, amennyiben 1 akkor engedélyezzük a belépést.
5. **created_at:** Ennek a mezőnek a típusa „datetime”. Egy adminisztrátor mikor került be a táblába.
6. **updated_at:** Ennek a mezőnek a típusa „datetime”. Egy adminisztrátor mikor frissült az adata utoljára táblába.
7. **password:** Ennek a mezőnek a típusa „password”. Amikor ebbe a mezőbe írunk, nem látjuk, hogy milyen karaktert írtunk be, csillagok jelzik a karakterek számát. A jelszó hossza 255 karakter lehet maximum. A felhasználó jelszavát sha256 kódolással tároljuk. Ez biztosítja az admin felület védelmét.

A „comment” táblára azért van szükségünk. Itt gyűjtjük a hozzászólásokat az egyes topicokhoz. Az „comment” táblánk oszlopai a következők:

1. **id:** Ez egy azonosító, ami egy int típusú érték (egész szám). Az értékét automatikusan növeli minden egyes új rekordnál. Elsődleges kulcs. Erre lehet hivatkozni minden egyes lekérdezésnél, amikor a „admin” táblának egy adott rekordjából információt szeretnénk kinyerni.
2. **user_id:** Egy foreign_key, ami int típusú (egész szám), a User táblához. Ezzel van összekapcsolva a két tábla egymással. Tudni szeretnénk melyik User-nek volt ez a hozzászólása.
3. **message:** Ez szöveg mező, amely tartalmazza azt a szöveget, amely már nem tartalmaz ofcén szavakat. Az ofcén szavak csillaggal vannak jelölve. A rész egyezést is csillagozzuk.
4. **status:** Egy adott üzenet megjelenhet-e, a felületen. Moderálási lehetőség, amennyiben 1 akkor megjelenhet egyébként meg nem.

5. **created_at**: Ennek a mezőnek a típusa „datetime”. Egy comment mikor került be a táblába.
6. **updated_at**: Ennek a mezőnek a típusa „datetime”. Egy comment mikor frissült az adata utoljára táblába.
7. **topic_id**: Ez egy foreign_key, amely egy topicot azonosít. Ez mutatja meg számunkra, hogy ez az üzenet melyik topic-hoz tartozik.
8. **original**: Az eredeti szöveg, amely az ofcén szavakat is tartalmaz.

A „dirty_word” tábla. Ebben a táblában az ofcén szavakat használjuk, amelyeket letöltöttünk xml formában és feltöltöttünk az adatbázisban. Ebben az xml-ben 2 érték szerepelt egy típus és maga a szó. A „dirty_word” tábla oszlopai a következők:

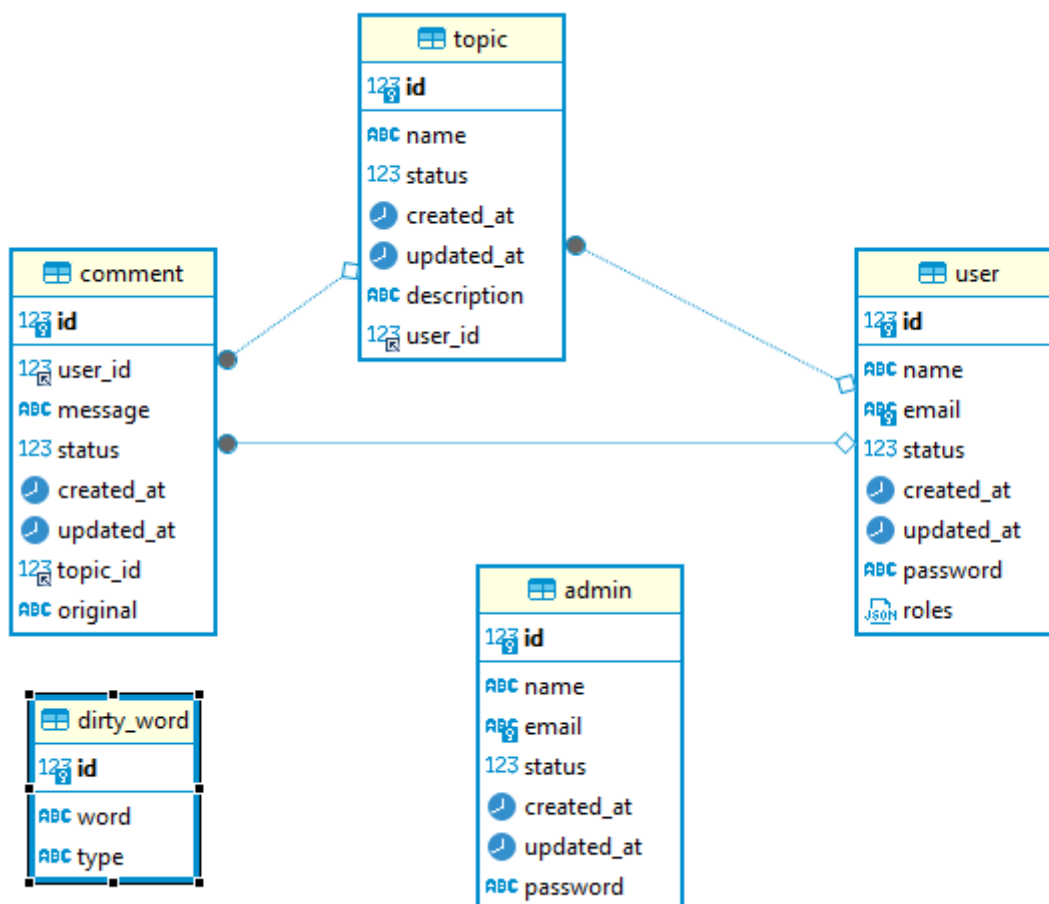
1. **id**: Ez egy azonosító, ami egy int típusú érték (egész szám). Az értékét automatikusan növeli minden egyes új rekordnál. Elsődleges kulcs. Erre lehet hivatkozni minden egyes lekérdezésnél, amikor a „admin” táblának egy adott rekordjából információt szeretnénk kinyerni.
2. **word**: varchar típusú oszlop, amelynek a hossza 255 karakter lehet. Ez tartalmazza konkrétan a trágár szót. Ezeket a szavakat kell kiszűrni a megjelenésből.
3. **type**: varchar típusú oszlop, amely „f” és „m” betűket tartalmazhat kizárólag.

A „topic” táblára azért van szükségünk. Itt gyűjtük a topic-kokat, amelyekhez hozzászólhatnak a felhasználók. Az „topic” táblánk oszlopai a következők:

1. **id**: Ez egy azonosító, ami egy int típusú érték (egész szám). Az értékét automatikusan növeli minden egyes új rekordnál. Elsődleges kulcs. Erre lehet hivatkozni minden egyes lekérdezésnél, amikor a „admin” táblának egy adott rekordjából információt szeretnénk kinyerni.
2. **user_id**: Egy foreign_key, ami int típusú (egész szám), a User táblához. Ezzel van összekapcsolva a két tábla egymással. Tudni szeretnénk melyik User hozta létre.
3. **name**: Ez szöveg mező, amely tartalmazza azt a szöveget, amely meghatározza a topic témáját.
4. **status**: Egy adott topic megjelenhet-e, a felületen. Moderálási lehetőség, amennyiben 1 akkor megjelenhet egyébként meg nem.
5. **created_at**: Ennek a mezőnek a típusa „datetime”. Egy topic mikor került be a táblába.
6. **updated_at**: Ennek a mezőnek a típusa „datetime”. Egy topic mikor frissült az adata utoljára táblába.
7. **description**: A topic leírása, hogy miről kell szólnia, ha név (cím) nem lenne egyértelmű.

Az „user” tábla azért szükséges, hogy itt tudjunk új felhasználókat rögzíteni. Akik ténylegesen fogják felhasználni a weboldalt. Ők tudnak új topicokat létrehozni és a topicokon belül kommunikálni másokkal. Az „user” táblánk oszlopai a következők:

1. **id**: Ez egy azonosító, ami egy int típusú érték (egész szám). Az értékét automatikusan növeli minden egyes új rekordnál. Elsődleges kulcs. Erre lehet hivatkozni minden egyes lekérdezésnél, amikor a „admin” táblának egy adott rekordjából információt szeretnénk kinyerni.
2. **name**: Ez szöveg mező, 255 karaktert tartalmazhat. Ez egy adott felhasználó a nevét tárolja. Ez azért szükséges, hogy megtudjuk mutatni neki, hogy ő van éppen belépve a weboldalon.
3. **email**: Ez szöveg mező, 160 karaktert tartalmaz, mert hivatalosan ennél hosszabb nem lehet. Ez a felhasználó email címét tárolja, amivel be tud jelentkezni az admin felületre.
4. **status**: Ennek a mezőnek a típusa „int”. Ez határozza meg, hogy felhasználó beléphet-e a weboldalra vagy sem. Ha az értéke 0 akkor nem léphet be, amennyiben 1 akkor engedélyezzük a belépést.
5. **created_at**: Ennek a mezőnek a típusa „datetime”. Egy felhasználó mikor került be a táblába.
6. **updated_at**: Ennek a mezőnek a típusa „datetime”. Egy felhasználó mikor frissült az adata utoljára táblába.
7. **password**: Ennek a mezőnek a típusa „password”. Amikor ebbe a mezőbe írunk, nem látjuk, hogy milyen karaktert írtunk be, csillagok jelzik a karakterek számát. A jelszó hossza 255 karakter lehet maximum. A felhasználó jelszavát sha256 kódolással tároljuk. Ez biztosítja az weboldal felület védelmét.



2. ábra: Harmadik normálforma (forrás: saját ábra)

Schema Name:	app	SQL Path:	
Default Charset:	utf8mb4	Database size:	240K
Default Collation:	utf8mb4_0900_ai_ci		

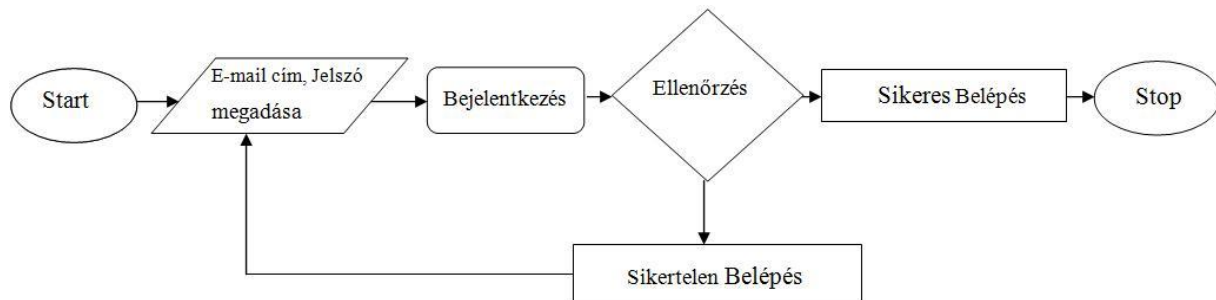
	Table Name	Engine	Auto Increment	Data Length	Description	Partitioned
Tables	admin	InnoDB	11	16K		[]
Views	comment	InnoDB	32	16K		[]
Indexes	dirty_word	InnoDB	366	16K		[]
Procedures	doctrine_migration_versions	InnoDB	0	16K		[]
Triggers	messenger_messages	InnoDB	0	16K		[]
Events	topic	InnoDB	5	16K		[]
Source	user	InnoDB	12	16K		[]

3. ábra: Adatbázistábla (forrás: saját ábra)

Admin felület

Bejelentkezés

A felületre domain után a /admin utvonalon érhető el. Ahol az admin email cím és jelszó párossal van lehetőségünk bejelentkezni, amelyet ha sikeresen megadunk akkor egy új felületen találjuk magunkat, a dashboardon. Ha helytelenül akkor hiba üzeneteket tapasztalhatunk.



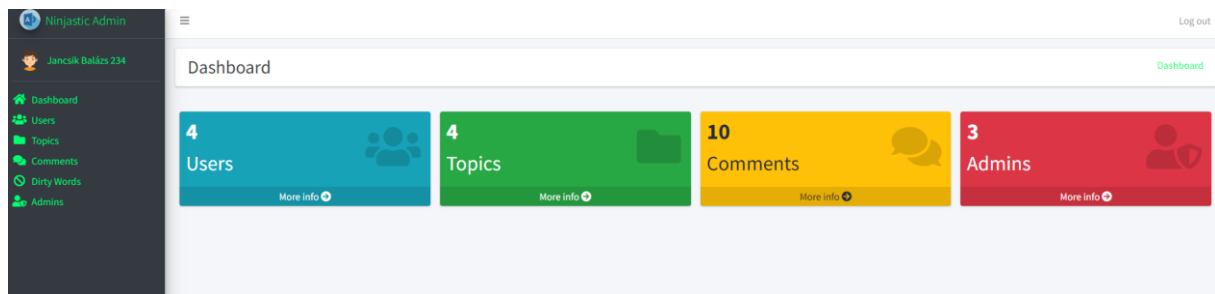
4. ábra: Belépési folyamatábra (forrás: saját ábra)

The screenshot shows the login interface for 'Ninjastic'. It features a central white box with a light gray border. Inside this box, there are two input fields: 'E-mail' and 'Password'. Each field has a red eye icon to toggle visibility and a user icon or lock icon to the right. Below the password field is a blue 'Login' button. The background is a solid light gray.

5. ábra: Bejelentkezési lap (forrás: saját ábra)

Sikeres bejelentkezés esetén a felhasználó adatait SESSION-ben tároljuk le, amíg ott megtalálható addig az admin bejelentkezve marad.

Dashboard



6. ábra: Dashboard (forrás: saját ábra)

Ez egy rövid át tekintő, egyből az adminisztrátor számára, ahol egyből láthatja, hogy hány regisztrált felhasználó van éppen, hány topikot hoztak már létre, hány hozzászólás történt már összesen és hány admin felhasználó van regisztrálva.

Baloldalon található egy menü, amelyen menü pontokat találunk.

A jobb felső sarkban található a „Log out” → Kijelentkezés. Amelyre kattintást követően kijelentkezünk, a SESSION-t töröljük és visszatérünk a Bejelentkezési felületre.

Felhasználók

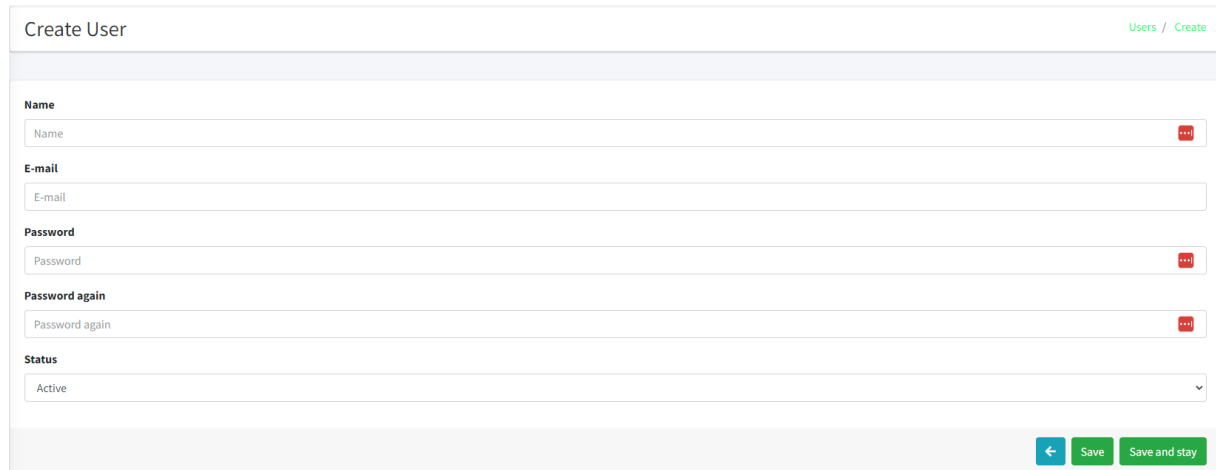
Listázás

Users				
<div><div>+ Create</div><div>Search <input type="text"/></div><div>Status <input type="text"/></div></div> <div>Result per page 25</div>				
ID	Name	E-mail	Status	Event
1	JBZ	jancsik.balazs@gmail.com	Active	<div><div></div><div></div></div>
2	Erdős Attila 2	erdos.attila@bitninja.io	Active	<div><div></div><div></div></div>
6	Marci	sqpp@proton.me	Active	<div><div></div><div></div></div>
11	test	attila.erdos@bitninja.io	Active	<div><div></div><div></div></div>

7. ábra: Users lista oldal (forrás: saját ábra)

A felhasználókat találjuk ezen a felületen. A create gomb megnyomása esetén tudunk új felhasználót létrehozni. Keresésre van lehetőség amely a névben és az e-mail címben tud keresni. A status választó meg egy filter. Ezen kívül a táblázat jobb oldalán van lehetőségünk állítani, hogy hány elem jelenhet meg egyszerre a felületen. Users-zel egysorban található zöld betűvel kisebbben szintén ez a breadcrumb. Amely, ha mélybire megyünk létrehozás vagy szerkesztésre akkor lehet róla vissza navigálni.

Létrehozás/Szerkesztés



Create User Users / Create

Name
Name

E-mail
E-mail

Password
Password

Password again
Password again

Status
Active

Save Save and stay

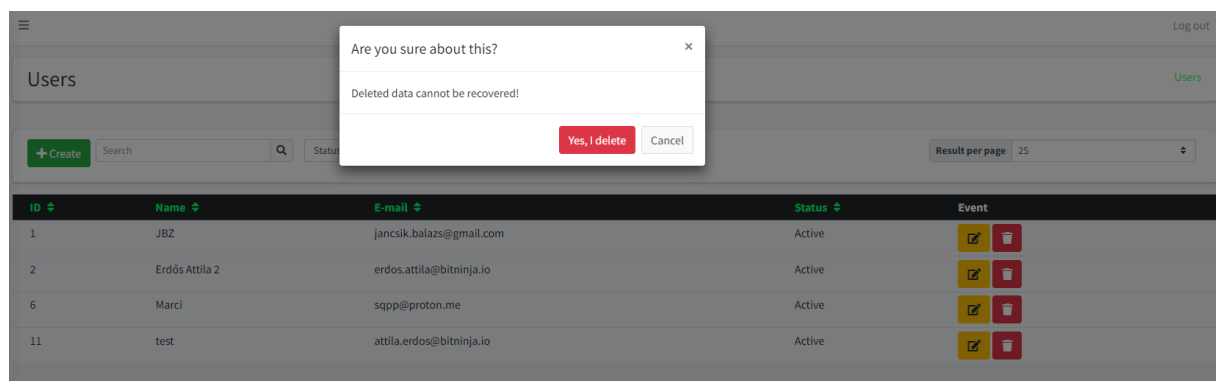
8. ábra: Users létrehozás/szerkesztés oldal (forrás: saját ábra)

Ezen felületen lehet létrehozni vagy szerkeszteni egy felhasználót. A felhasználó mentése esetén amely a „save” vagy „save and stay” gomb megnyomásával történik. A „save” gomb használata esetén visszavigágunk a listázási oldalra. A „save and stay” megnyomása esetén maradunk ezen a felület, amennyiben szeretnénk folytatni a felhasználó adatainak a módosítását. A vissza nyilacska gomb mentés nélkül visszavigág a listázási oldalra. Az adatok mielőtt a mentés megtörténne ellenőrzés történik, amelyek a következők.

1. „Name”: Nem lehet üres mező
2. „Email”: E-mail formátumnak megfelelőnek kell lennie.
3. „Password”: Minimum 8 karakter hosszúnak kell lennie és password again input mezővel megegyező tartalommal kell bírnia.

Amennyiben egy email cím már használva van, arról hiba üzenetet kapunk.

Törlés



Are you sure about this?

Deleted data cannot be recovered!

Yes, I delete Cancel

ID	Name	E-mail	Status	Event
1	JBZ	jancsik.balazs@gmail.com	Active	✎ 🗑
2	Erdős Attila 2	erdos.attila@bitninja.io	Active	✎ 🗑
6	Marci	sqpp@proton.me	Active	✎ 🗑
11	test	attila.erdos@bitninja.io	Active	✎ 🗑

9. ábra: Users törlés popup (forrás: saját ábra)

Törlés esetén mindig egy felugró ablak jelenik meg, amely egy megerősítés ellenében végrehajta a műveletet.

Topikok

Listázás

Topics					Topics
<div> + Create <input type="text" value="Search Username"/> <input type="text" value="Search"/> <input type="text" value="Status"/> </div> <div>Result per page 25</div>					
ID	UserName	Name	Status	Event	
1	JBZ	Első topic 2	Active		
2	Erdős Attila 2	Második topic	Active		
3	JBZ	api test	Active		
4	JBZ	api test updated	Active		

10. ábra: Topics lista oldal (forrás: saját ábra)

A topikokat találjuk ezen a felületen. A create gomb megnyomása esetén tudunk új topikot létrehozni. „Search Username” placeholder mutatja nekünk azt, hogy abban az input mezőben kizárólag a felhasználó nevére lehet keresni.

„Search” placeholder mutatja nekünk azt, hogy keresni tudunk a topik nevére.

A „Status” választó meg egy filter. Ezen kívül a táblázat jobb oldalán van lehetőségünk állítani, hogy hány elem jelenhet meg egyszerre a felületen.

Létrehozás/Szerkesztés

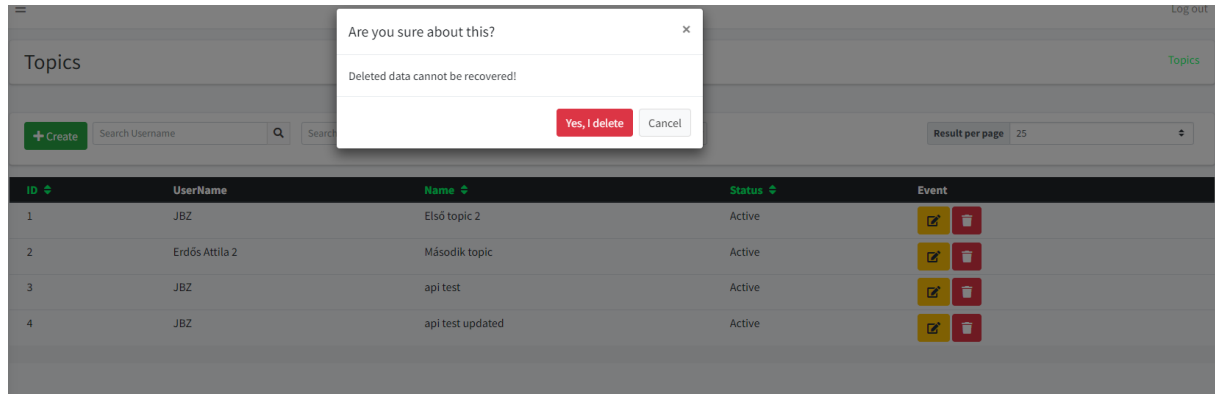
Create Topic		Topics / Create
<div> <div>Name</div> <input type="text" value="Name"/> </div> <div> <div>Description</div> <input type="text" value="Description"/> </div> <div> <div>User</div> <input type="text" value="Not selected user"/> </div> <div> <div>Status</div> <input type="text" value="Active"/> </div>		
<div> ← Save Save and stay </div>		

11. ábra: Topics létrehozás/szerkesztés oldal (forrás: saját ábra)

Ezen felületen lehet létrehozni vagy szerkeszteni egy topikot. A topik mentése esetén amely a „save” vagy „save and stay” gomb megnyomásával történik. A „save” gomb használata esetén visszavigálunk a listázási oldalra. A „save and stay” megnyomása esetén maradunk ezen a felület, amennyiben szeretnénk folytatni a topik adatainak a módosítását. A vissza nyilacska gomb mentés nélkül visszavigál a listázási oldalra. Az adatok mielőtt a mentés megtörténne ellenőrzés történik, amelyek a következők.

1. „Name”: Nem lehet üres mező
2. „Description”: Nem lehet üres mező
3. „User”: Kötelező felhasználót hozzárendelni egy topikhoz

Törlés



12. ábra: Topics törlési oldal (forrás: saját ábra)

Törlés esetén mindig egy felugró ablak jelenik meg, amely egy megerősítés ellenében végrehajta a műveletet.

Hozzászólások

Listázás

Comments						Comments
<div><div>Create</div><div>Search Username</div><div>Search Topic</div><div>Search</div><div>Status</div><div>Result per page 25</div></div>						
ID	User Name	Topic Name	Message	Status	Event	
1	JBZ	Első topic 2	Ez egy érdekes message és sok ...	Active		
3	Erdős Attila 2	Második topic	Hosszabb szöveg Hosszabb szöve...	Active		
4	Erdős Attila 2	Első topic 2	hi, sup?	Active		
13	JBZ	Első topic 2	Your comment message here	Active		
18	JBZ	Második topic	tesztelés	Active		
20	JBZ	Második topic	sggsg	Active		
27	JBZ	Első topic 2	Postman seems to pass the mess...	Active		
28	JBZ	Első topic 2	Na utolsó	Active		

13. ábra: Comments lista oldal (forrás: saját ábra)

A hozzászólásokat találjuk ezen a felületen. A create gomb megnyomása esetén tudunk új hozzászólást létrehozni.

„Search Username” placeholder mutatja nekünk azt, hogy abban az input mezőben kizárólag a felhasználó nevére lehet keresni.

„Search Topic” placeholder mutatja nekünk, hogy a topic nevére tudunk keresni.

„Search” placeholder mutatja nekünk azt, hogy keresni tudunk az üzenet szövegében.

A „Status” választó meg egy filter. Ezen kívül a táblázat jobb oldalán van lehetőségünk állítani, hogy hány elem jelenhet meg egyszerre a felületen.

Létrehozás / Szerkesztés

14. ábra: Comments létrehozás/szerkesztés oldal (forrás: saját ábra)

Ezen felületen lehet létrehozni vagy szerkeszteni egy hozzászólást. A hozzászólás mentése esetén amely a „save” vagy „save and stay” gomb megnyomásával történik. A „save” gomb használata esetén visszanyitjuk a listázási oldalt. A „save and stay” megnyomása esetén maradunk ezen a felületen, amennyiben szeretnénk folytatni a hozzászólás adatainak a módosítását. A vissza nyílcska gomb mentés nélkül visszanyitja a listázási oldalt. Az adatok mentése előtt a mentés megtörténne ellenőrzés történik, amelyek a következők.

1. „Original”: Nem lehet üres mező
2. „User”: Kötelező felhasználót hozzárendelni egy hozzászóláshoz
3. „Topic”: Kötelező Topicot hozzárendelni egy hozzászóláshoz

Törlés

ID	User Name	Topic Name	Message	Status	Event
1	JBZ	Első topic 2	Ez egy érdekes message és sok ...	Active	[Edit] [Delete]
3	Erdős Attila 2	Második topic	Hosszabb szöveg Hosszabb szöve...	Active	[Edit] [Delete]
4	Erdős Attila 2	Első topic 2	hi, sup?	Active	[Edit] [Delete]
13	JBZ	Első topic 2	Your comment message here	Active	[Edit] [Delete]
18	JBZ	Második topic	tesztelés	Active	[Edit] [Delete]
20	JBZ	Második topic	sggsg	Active	[Edit] [Delete]

15. ábra: Comments törlés oldal (forrás: saját ábra)

Törlés esetén mindig egy felugró ablak jelenik meg, amely egy megerősítés ellenében végrehajta a műveletet.

Csúnya szavak

Listázás

Dirty Words				Dirty Words
<div> + Create <input type="text" value="Search"/> <input type="button" value="Q"/> <input type="text" value="Status"/> </div>				Result per page 25
ID	Word	Type	Event	
1	Aberált	Maybe		
2	Aberált	Maybe		
3	Abortuszmaradék	Force		
4	Abszolút hülye	Maybe		
5	Agyalágyult	Maybe		
6	Agyatlan	Maybe		
7	Agybatetovált	Maybe		

16. ábra: Dirty Words lista oldal (forrás: saját ábra)

A ofcén szavak listáját találjuk ezen a felületen. A create gomb megnyomása esetén tudunk új ofcén szót létrehozni.

„Search” placeholder mutatja nekünk azt, hogy keresni tudunk trágár szóra.

A „Status” választó meg egy filter. Ezen kívül a táblázat jobb oldalán van lehetőségünk állítani, hogy hány elem jelenhet meg egyszerre a felületen.

Létrehozás / Szerkesztés

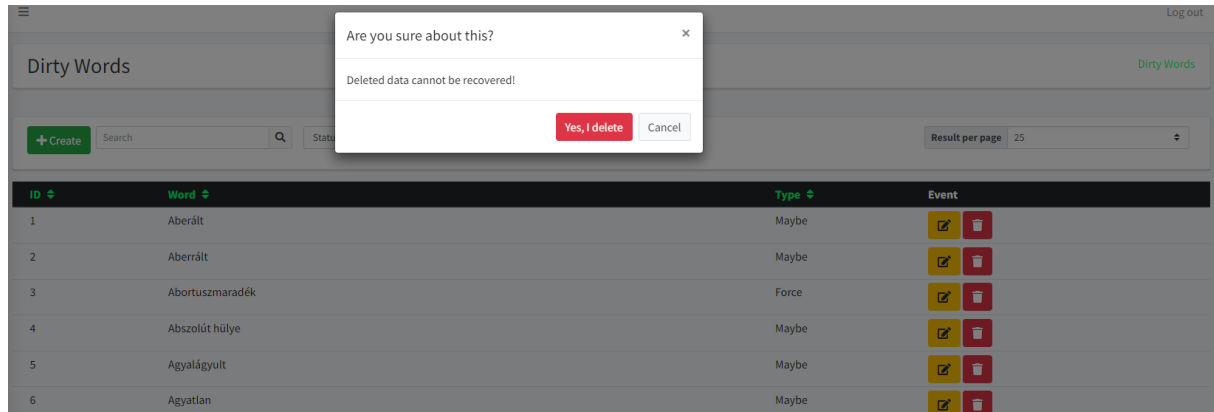
Create Dirty Word		Dirty Words / Create
Word		
	<input type="text" value="Name"/>	
Type		
	<input type="text" value="Force"/>	
		<input type="button" value="←"/> <input type="button" value="Save"/> <input type="button" value="Save and stay"/>

17. ábra: Dirty Words létrehozás/szerkesztés oldal (forrás: saját ábra)

Ezen felületen lehet létrehozni vagy szerkeszteni egy csúnya szót. A csúnya szó mentése esetén amely a „save” vagy „save and stay” gomb megnyomásával történik. A „save” gomb használata esetén visszavigálunk a listázási oldalra. A „save and stay” megnyomása esetén maradunk ezen a felület, amennyiben szeretnénk folytatni a csúnya szó adatainak a módosítását. A vissza nyilacska gomb mentés nélkül visszavigál a listázási oldalra. Az adatok mielőtt a mentés megtörténne ellenőrzés történik, amelyek a következők.

1. „Word”: Nem lehet üres mező
2. „Type”: Nem lehet üres mező, kötelező választani valamit

Törlés

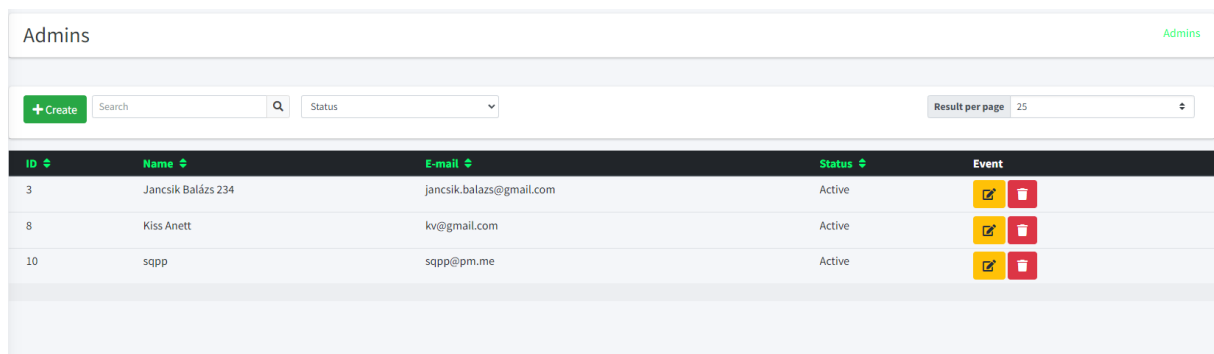


18. ábra: Dirty Words törlés oldal (forrás: saját ábra)

Törlés esetén mindig egy felugró ablak jelenik meg, amely egy megerősítés ellenében végrehajta a műveletet.

Adminok

Listázás



19. ábra: Admins lista oldal (forrás: saját ábra)

Az adminok listáját találjuk ezen a felületen. A create gomb megnyomása esetén tudunk új admint létrehozni.

„Search” placeholder mutatja nekünk azt, hogy keresni név és e-mail cím alapján.

A „Status” választó meg egy filter. Ezen kívül a táblázat jobb oldalán van lehetőségünk állítani, hogy hány elem jelenhet meg egyszerre a felületen.

Létrehozás / Szerkesztés

Create Admin Admins / Create

Name

E-mail

Password

Password again

Status
 Active

20. ábra: Admins lista oldal (forrás: saját ábra)

Ezen felületen lehet létrehozni vagy szerkeszteni egy admint. Az admin mentése esetén amely a „save” vagy „save and stay” gomb megnyomásával történik. A „save” gomb használata esetén visszavigágunk a listázási oldalra. A „save and stay” megnyomása esetén maradunk ezen a felület, amennyiben szeretnénk folytatni az admin adatainak a módosítását. A vissza nyilacska gomb mentés nélkül visszavigágál a listázási oldalra. Az adatok mielőtt a mentés megtörténne ellenőrzés történik, amelyek a következők.

1. „Name”: Nem lehet üres mező
2. „Email”: E-mail formátumnak megfelelőnek kell lennie.
3. „Password”: Minimum 8 karakter hosszúnak kell lennie és password again input mezővel megegyező tartalommal kell bírnia.

Amennyiben egy email cím már használva van, arról hiba üzenetet kapunk.

Törlés

Are you sure about this? ×

Deleted data cannot be recovered!

ID	Name	E-mail	Status	Event
3	Jancsik Balázs 234	jancsik.balazs@gmail.com	Active	
8	Kiss Anett	kv@gmail.com	Active	
10	sqpp	sqpp@pm.me	Active	

21. ábra: Admins lista oldal (forrás: saját ábra)

Törlés esetén mindig egy felugró ablak jelenik meg, amely egy megerősítés ellenében végrehajta a műveletet.

Irodalom jegyzék

http://hu.wikipedia.org/wiki/Forum_Romanum (letöltés dátuma: 2024.04.16)

http://www.pellerd.hu/index.php?option=com_content&view=article&id=225%3Aami-a-forum-rendeltetese&catid=28%3Ahasznalat&Itemid=41&lang=hu (letöltés dátuma:

2024.04.16)

http://hu.wikipedia.org/wiki/Apache_HTTP_Server (letöltés dátuma: 2024.04.16)

<http://hu.wikipedia.org/wiki/MySQL> (letöltés dátuma: 2024.04.16)

<http://hu.wikipedia.org/wiki/PHP> (letöltés dátuma: 2024.04.16)

<http://apatoktol.atw.hu/Koz/Adatbaziskezeles/Normpelda.doc> (letöltés dátuma:

2024.04.16)

<https://github.com/stifolder/kretainsult/blob/master/src/assets/dirtywords.xml> (letöltés dátuma: 2024.04.16)

<https://github.com/ColorlibHQ/AdminLTE/tree/v3.0.3> (letöltése dátuma: 2024.01.12)