Debreceni Egyetem
Informatikai Kar

Webes technológiákon alapuló hierarchikus feladatkezelő rendszer fejlesztése

Témavezető:
Kósa Márk Szabolcs
egyetemi tanársegéd

Készítette:
Kovács Péter
programtervező informatikus

Debrecen
2010

Tartalomjegyzék

1. Bevezetés

1.1. Napjaink társadalma és a World Wide Web

Napjaink társadalmának egyik legfontosabb jellemzője az információközpontúság. Az emberek nap, mint nap információkat szereznek és közölnek egymással. Ennek a modellnek a technikai értelemben vett sikerességéhez mindenekelőtt valamilyen rendszerben történő gyors és könnyű elérhetőség szükséges. Tim Berners-Lee és Robert Cailliau 1990-ben kiadott ajánlása[1] hatékony és könnyen használható lehetőséget adott az információk hálózaton keresztül történő, kapcsolat alapú kezelésére. A javaslatban szereplő fogalmak (világháló, böngésző, hiperszöveg, stb.) ma is a webes rendszerek fontos fogalmai. A HTTP és a HTML szabványok megalkotása, és a W3C konzorcium megalapítása után a világháló népszerűsége gyors növekedésnek indult. Mára már számos technológia elérhető a webes fejlesztők számára, valamint előtérbe került a gyors alkalmazásfejlesztés.

1.2. A szakdolgozati téma rövid áttekintése

A szakdolgozatban szerepel néhány webes technológia rövid áttekintése, majd a kiválasztott eszközökkel egy gyakorlati példán keresztül a kiválasztott eszközök használatának az ismertetésére is sor kerül.

A gyakorlati példa egy olyan rendszer megvalósítását mutatja be, ahol a rendszer szereplői egyének, akik között alá-fölérendeltségi viszony alakítható ki. Az egyes egyénekhez feladatok rendelhetők, amelyek a kialakuló hierarchia mentén menedzselhetőek. Ehhez szükséges vizsgálni a hierarchián alapuló webes rendszerek megvalósításainak lehetőségeit, amely szintén szerepet kap a szakdolgozatban.

1.3. Feltételezett ismeretek

A szakdolgozat olvasásához a következő ismeretek szükségesek:

- A procedurális és az objektum orientált paradigma fogalmainak az ismerete.
- A hálózatokkal kapcsolatos fogalmaknak és protokolloknak az ismerete.
- A relációs adatbázis kezelő rendszerek fogalmainak, és az SQL nyelvnek az ismerete.

[1] WorldWideWeb: Proposal for a HyperText Project (http://www.w3.org/Proposal.html [2010.04.28.]).

2. Webes technológiák áttekintése

2.1. A Web architektúrája

Fogalmak[2]:

- Világháló (Web): Információs tér, amely erőforrásokat tartalmaz.
- Erőforrás: Általános fogalom. Bármi lehet erőforrás, aminek azonossága van.
- Információs erőforrás: Olyan erőforrás, amely üzenetként szállítható. Információs erőforrások például a weboldalak és a képek.
- URI: Az erőforrások azonosítására szolgáló karaktersorozat. Egy URI egy erőforrást azonosít (ha ugyanaz az URI több erőforrást is azonosít, azt URI ütközésnek nevezzük), azonban több URI azonosíthatja ugyanazt az erőforrást (URI alias). Az URI perzisztens, ha mindig ugyanazt az erőforrást azonosítja. Az URI, mint azonosító vonatkozhat az erőforrás helyének (URL), vagy a nevének (URN) a meghatározására. Minden URI egy sémanévvel kezdődik. A séma specifikációja adja meg, hogy hogyan kötődik az azonosító az erőforráshoz. Az URI megadásakor felhasználható karakterek bizonyos része határoló jel. Ezek lefoglalt karakterek, speciális jelentésük van („%hexadecimális_számjegyek” formában azonban a speciális jelentés nélkül is használhatóak ezek a karakterek, ez az URL kódolás).
- Web agent: Olyan személy vagy szoftver, amely az információs téren végez tevékenységet. Az agent-ek közötti kommunikáció szabványos protokollok (pl. HTTP) segítségével, üzenetekkel történik. Web agent például a szerver.
- User agent: Speciális web agent, amely szoftver, és a felhasználó képviseletében végzi a tevékenységet. User agent például a böngésző és az e-mail kliens.
- Művelet: Az URI-val hivatkozott erőforrásokon az agent-ek műveleteket végezhetnek. Ezeket a műveleteket az URI-kat használó protokollok definiálják. Az egyik gyakran használt művelet a reprezentáció kinyerése.

[2] A fogalmak megadásához felhasznált dokumentumok: Architecture of the World Wide Web, Volume One (http://www.w3.org/TR/webarch/ [2010.04.28.]), RFC 3986 (http://tools.ietf.org/html/rfc3986 [2010.04.28.]).

- Reprezentáció: Olyan adat, amely az erőforrás állapotáról kódol információt. Egy adott URI-val több reprezentáció is elérhető. Ekkor az agent és a szerver között történik a tartalomegyeztetés, tehát annak eldöntése, hogy melyik reprezentációt kapja meg az agent. A protokollok megadják, hogy milyen formában szállítható a reprezentáció. A reprezentáció megadásához adatformátumok használhatóak. Az adatformátum specifikációja megadja a reprezentáció értelmezésének a módját. Adatformátum például az XHTML és a CSS. Az adatformátum lehet bináris vagy szöveges. Bináris az adatformátum, ha az adat közvetlenül a processzornak szól, szöveges pedig akkor, ha az adat valamilyen karakterkódolással megadott karakterek sorozata. Egy erőforrás reprezentációja tartalmazhat hivatkozást egy másik erőforrásra, az erőforrás URI segítségével. Ez a hiperhivatkozás.
- Hiperszöveg: Olyan szöveg, amely hiperhivatkozásokat tartalmaz.

2.2. MIME

A MIME az e-mailekhez használható üzenetreprezentációt leíró internetes szabvány, az RFC 822 kiterjesztése. Támogatja a US-ASCII-től eltérő karakterkódolások használatát az üzenet törzsében, valamint a nem szöveges üzenettörzs használatát is lehetővé teszi.

MIME fejléc mezők:

- MIME-Version: Az üzenet összeállításakor használt MIME verziószám.
- Content-Type: Az üzenet törzsében szereplő adat típusát adja meg. A megadáshoz használt forma: típus/altípus. A típus az adat általános típusát adja meg, az altípus pedig az adott típuson belüli specifikus formátumot. Az RFC 2046-ban szereplő néhány típus: text (szöveg), image (kép), audio (hang), video (videó), application (alkalmazások által használt tartalomtípus), stb.
- Content-Transfer-Encoding: Az üzenet törzsének a kódolási módját tartalmazza. A mező néhány lehetséges értéke:
    - 7bit: Az adat legfeljebb 998 oktet hosszúságú sorokból áll, ahol az oktetek decimális értéke legfeljebb 127 lehet, és nem lehet 0. A 13-as és a 10-es decimális értékű oktet a sor végét jelzi, máshol nem szerepelhetnek.
    - 8bit: Az adat legfeljebb 998 oktet hosszúságú sorokból áll. A 13-as és a 10-es decimális értékű oktet a sor végét jelzi, máshol nem szerepelhetnek.
    - binary: Bármilyen oktet sorozat megengedett.

    - quoted-printable: Az adat quoted-printable kódolással van kódolva.
    - base64: Az adat base64 kódolással van kódolva.
- Content-ID: Egyedi azonosító.
- Content-Description: Az üzenet tartalmának rövid leírására használható.

2.3. HTTP

A HTTP alkalmazásszintű, állapotmentes protokoll, hipermédia alapú információs rendszerekhez. A jelenleg aktuális verzió a HTTP/1.1.

A HTTP kommunikáció üzenetekkel történik. Az üzenet lehet kérés vagy válasz üzenet. A kérést küldő alkalmazás a kliens, a kérésre választ küldő alkalmazás pedig a szerver.

Az üzenet tartalmaz egy kezdő sort, amely a kérés üzenet esetében megadja az erőforráson végrehajtandó metódust, az erőforrás azonosítóját, és a használt protokoll verziószámát, a válasz üzenet esetében pedig a használt protokoll verziószámát, a státuszkódot, és a státuszkód rövid szöveges leírását. A kezdő sor után szerepelhetnek a fejléc mezők, valamint az üzenet törzse.

A szerver és a kliens között lehet köztes alkalmazás, például proxy. A kliens és a szerver rendelkezhet gyorsítótárral, amelyben a válasz üzenetek tárolhatóak. Ekkor az üzenetnek van lejárati ideje, ami után ellenőrizni kell az üzenet érvényességét, a gyorsítótárból való felhasználáshoz.

Az erőforrásokat azonosító URI-ban a http séma használata adja meg a HTTP protokoll használatát.

Néhány HTTP metódus:

- OPTIONS: Az elérhető kommunikációs opcióknak a lekérdezése.
- GET: Erőforrás reprezentációjának a lekérdezése.
- HEAD: Megegyezik a GET művelettel, de a válasz üzenet nem tartalmazza az üzenet törzsét.
- POST: Adat küldése feldolgozásra.
- PUT: A kérés üzenetben szereplő entitás tárolásának a kérése.
- DELETE: Erőforrás törlésének a kérése.

HTTP státuszkód csoportok:

- 1xx: Információs státuszkód csoport.
- 2xx: Sikeres kérés.
- 3xx: A kérés teljesítéséhez további művelet szükséges.
- 4xx: Kliens hiba.
- 5xx: Szerver hiba.

HTTP fejléc mezők:

- Általános fejléc mezők: Kérés és válasz üzenetben is szerepelhetnek, és nem a szállítandó entitásra vonatkoznak.
- Kérés fejléc mezők: A kérésre és a kliensre vonatkozó információkat tartalmazhatnak.
- Válasz fejléc mezők: A szerverre vonatkozó információkat tartalmazhatnak.
- Entitás fejléc mezők: Az üzenetben szállított entitás törzsére vonatkozó információkat tartalmazhatnak.

2.4. Apache HTTP Server

Az Apache HTTP Server egy nyílt forráskódú webszerver, amely számos operációs rendszerre elérhető. Fejlesztését az Apache Software Foundation végzi.

Az Apache HTTP szerver program a httpd.

Konfigurálása direktívák segítségével történik. Például a DocumentRoot direktíva megadja, hogy melyik könyvtárból kerüljenek kiszolgálásra a fájlok.

A fő konfigurációs fájl a http.conf nevű fájl. A .htaccess fájl speciális konfigurációs fájl, arra a mappára vonatkozik, amelyben található.

A beépített funkcionalitások modulok segítségével bővíthetőek. Apache modul például a mod_rewrite, amely URL-ek átírását teszi lehetővé megadott szabályok alapján.

```
#URL átírás engedélyezése.
RewriteEngine on
#Szabály megadása URL átíráshoz.
#Például a "cikkek/teszt_cikk/" átírás után "articles.php?name=teszt_cikk" lesz.
RewriteRule ^cikkek/([a-z0-9_]+)/?$ articles.php?name=$1 [NC,L]
```

mod_rewrite példa.

2.5. HTML

A HTML leíró nyelv információ publikálására a világhálón.

A HTML 4.01 specifikáció fogalmai:

- Elem: A HTML elemek struktúrát vagy viselkedés írnak elő. Egy elem három részből állhat: kezdő címke (<elem-név>), tartalom, záró címke (</elem-név>). A HTML dokumentum HTML elemeket tartalmaz.
- Attribútum: Az attribútum elemhez tartozó tulajdonság. Az attribútumok általában név-érték párok.
- Karakter referencia: Szimbolikus név vagy számérték karakter megadására.
- Megjegyzés: <!-- és --> között megadott megjegyzés.

A HTML dokumentum megadására szolgáló MIME típus a text/html.

2.6. XHTML

Az XHTML leíró nyelv, a HTML XML alapon történő megvalósítása. Az XML alapúság miatt az XHTML dokumentumnak jól formázottnak kell lennie.

Az XHTML 1.1 moduláris felépítésű.

Az XHTML dokumentum megadására szolgáló MIME típus az application/xhtml+xml.

2.7. CSS

A CSS stílusleíró nyelv webes dokumentumok számára.

Egy CSS stíluslap utasításokat tartalmaz, ahol az utasítások vagy „@” karakterrel kezdődő szabályok, vagy szabályhalmazok. A „@” karakterrel kezdődő szabályok a CSS feldolgozónak szólnak. A szabályhalmazok szelektorokat, és egy deklarációs blokkot tartalmaznak. A deklarációs blokk deklarációkat tartalmaz, ahol a deklaráció tulajdonságnév és érték pár. A szelektorok adják meg, hogy a dokumentum mely elemeire kell alkalmazni a deklarációs blokkbon megadott deklarációkat.

A CSS dokumentum megadására szolgáló MIME típus a text/css.

2.8. ECMAScript

Az ECMAScript objektum alapú szkriptnyelv, tehát egy létező rendszer szolgáltatásainak kezelésére szolgál. Egy ECMAScript objektum tulajdonságokat tartalmaz, ahol az egyes

tulajdonságokhoz a kezelésükre vonatkozó attribútumok tartozhatnak. A tulajdonságok objektumokat, primitív értékeket vagy metódusokat tartalmazhatnak.

Az ECMAScript prototípus alapú nyelv, minden konstruktor tartalmaz egy prototípus objektumot. A konstruktorok az objektumok létrehozására szolgálnak, a prototípus pedig az öröklődésnél használható.

A böngészőkben használt JavaScript és Jscript szkriptnyelvek az ECMAScript szabványon alapulnak.

2.9. JQuery

A JQuery ingyenes, nyílt forráskódú, könnyen használható JavaScript könyvtár, amely böngészőfüggetlen JavaScript fejlesztést tesz lehetővé.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="hu">

<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>jQuery Teszt</title>

<!-- JQuery könyvtár használata. A jquery-1.4.2.min.js fájl letöltése szükséges a
     hivatalos  JQuery honlapról. -->
<script type="text/javascript" src="jquery-1.4.2.min.js">
</script>
<script type="text/javascript">

/* Függvény betűméret megváltoztatására. */
function changeFontSize(number) {

   /* A changableFontSize osztálynévvel rendelkező elemek font-size tulajdonságának
      új értéke a második paraméterben megadott függvény visszatérési értéke lesz. */
   $(".changableFontSize").css('font-size', function (index, value) {
      var currentSize = parseFloat(value, 10);
      var newSize = currentSize*number;
      return newSize;
   })

}
```

```
/* A paraméterként megadott függvény végrehajtása a DOM betöltése után. */
$(document).ready(function() {

   /* Eseménykezelő hozzáadása. (A betűméret növelésére kattintáskor a
      paraméterként  megadott függvény kerül végrehajtásra.) */
   $(".increaseFontSize").click(function () {  changeFontSize(1.5); });

   /* Eseménykezelő hozzáadása. (A betűméret csökkentésére kattintáskor a
      paraméterként megadott függvény kerül végrehajtásra.) */
   $(".decreaseFontSize").click(function () { changeFontSize(0.5); });

});
</script>
</head>

<body>

<h1>JQuery Teszt</h1>
<div>
   <a href="#" class="increaseFontSize">Betűméret növelése</a>,
   <a href="#" class="decreaseFontSize">Betűméret csökkentése</a>
</div>
<div class="changableFontSize">Kattints a "Betűméret növelése" vagy a "Betűméret
csökkentése" feliratra, ennek a szövegnek, a betűméretének a növeléséhez vagy a
csökkentéséhez.</div>
<div>Ennek a szövegnek, a betűméretének a nagysága nem változik.</div>
...

</body>

</html>
```

JQuery példa: Betűméret növelése, csökkentése.

2.10. PHP

A PHP webes fejlesztéseknél gyakran használt, nyílt forráskódú, általános célú szkriptnyelv, amely lehetővé teszi weboldalaknak a dinamikus generálását. A PHP kód a PHP nyitó és záró címke között helyezkedik el. A PHP kód beágyazható HTML dokumentumba.

A lehetséges PHP nyitó és záró címkék:

PHP Nyitó címke	PHP Záró címke
<?php	?>
<script language="php">	</script>
<?	?>
<%	%>

A PHP kódot a PHP feldolgozó interpretálja.

2.11. PostgreSQL

A PostgreSQL ingyenes, nyílt forráskódú objektum relációs adatbázis-kezelő rendszer. Az ACID tulajdonságoknak megfelel. Támogatja a nézetek, tárolj eljárások, triggerek használatát. A PostgreSQL procedurális nyelve a PL/pgSQL.

2.12. Zend Framework

A Zend Framework nyílt forráskódú, objektumorientált PHP keretrendszer. Lazán csatolt komponensekből épül fel.

A Zend Framework-nél használt néhány tervezési minta:

- MVC: Az MVC minta az alkalmazásoknál 3 komponenst különít el:
  - Modell: A modell tartalmazza az adatok kezelését, és a hozzá kapcsolódó üzleti logikát.
  - Nézet: A nézet jeleníti meg a modellt a felhasználónak.
  - Vezérlő: A felhasználó kérése alapján kezeli a modellt és a nézetet.
- Elülső vezérlő: Közös belépési pont a kérések számára. Az elülső vezérlő feldolgozza, és továbbítja a felhasználói kérést a megfelelő vezérlőnek. A Zend Framework-ben a Zend_Controller_Front implementálja az elülső vezérlő mintát.

Zend Framework projekt:

Új Zend Framework projekt létrehozása a Zend_Tool_Project segítségével:

```
$ zf create project projekt_név
```

A létrehozott projekt felépítése:

- application: A nem publikus fájlokat tartalmazza.
    - Bootstrap.php: A Bootstrap osztály az erőforrások és a komponensek (pl.: Zend_Controller_Front) inicializálására szolgál.
    - configs/application.ini: A Zend Framework alkalmazás (Zend_Application) beállítására szolgáló direktívákat tartalmaz (például a Bootstrap osztály elérési útvonalát).
    - controllers: A vezérlő osztályokat tartalmazza.
    - models: A modell osztályokat tartalmazza.
    - views/scripts: A nézet szkripteket tartalmazza.
- library: Könyvtárakat tartalmazhat.
- public: A publikus fájlokat tartalmazza.
    - index.php: A Zend Framework alkalmazás belépési pontja. Létrehozza a Zend_Application példányt az alkalmazás inicializálásához (az inicializálás az application.ini fájlban megadott direktívák alapján történik).
- test: Egységteszteket tartalmazhat.

3. Hierarchikus modellek megvalósítási lehetőségei[3]

3.1. Bevezetés

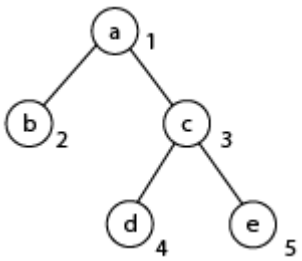
A hierarchia elemek olyan rendszere, ahol az elemek osztályozott sorrend alapján kerülnek elrendezésre.[4] A hierarchiában, mint adatszerkezetben minden elemnek 0 vagy több rákövetkezője, és legfeljebb egy megelőzője lehet. Az egyik gyakran használt hierarchikus adatszerkezet a fa. A fa elemei a csomópontok. Egy csomópont rákövetkező csomópontjai a gyermek csomópontok, a csomópont megelőző csomópontja pedig a szülő csomópont. A levél csomópont olyan csomópont, amelynek nincs gyermek csomópontja. A fában egy adott csomópont, és a csomópont leszármazottai (a csomópont gyermekei, a csomópont gyermekeinek a gyermekei, stb.) részfát alkotnak.

Néhány lehetőség hierarchikus adatok kezelésére relációs adatbázis kezelő rendszerekben:

- Szomszédsági lista modell. (Adjacency list model.)
- Beágyazott halmaz modell. (Nested set model.)

3.2. Szomszédsági lista modell

A szomszédsági lista modellben egy relációs tábla minden eleménél tároljuk az adott elem azonosítóját, és a szülő elem azonosítóját.



Elem azonosítója	Szülő azonosítója	Elem értéke
1	-	a
2	1	b
3	1	c
4	3	d
5	3	e

```
CREATE TABLE adjacency_list_example (

   id SERIAL PRIMARY KEY,
   parent_id INTEGER DEFAULT NULL REFERENCES adjacency_list_example (id),
   value CHARACTER(1)

);
```

[3] Az ebben a fejezetben szereplő példák a PostgreSQL adatbázis-kezelő rendszer segítségével, valamint a PHP nyelv segítségével készültek.

[4] http://www.thefreedictionary.com/hierarchy [2010.04.28.]

```
INSERT INTO adjacency_list_example (parent_id, value) VALUES(NULL, 'a');
INSERT INTO adjacency_list_example (parent_id, value) VALUES(1, 'b');
INSERT INTO adjacency_list_example (parent_id, value) VALUES(1, 'c');
INSERT INTO adjacency_list_example (parent_id, value) VALUES(3, 'd');
INSERT INTO adjacency_list_example (parent_id, value) VALUES(3, 'e');
```

Szomszédsági lista modell példa.

```
<?php
function child_nodes($id) {

   $result = pg_query("SELECT value FROM adjacency_list_example " .
                      "WHERE parent_id = '$id';");
   $child_nodes = array();
   while ($row = pg_fetch_array($result)) {
      array_push($child_nodes, $row["value"]);
   }
   return $child_nodes;

}
?>
```

A fa egy elemének a gyermekeinek a lekérdezése.

```
<?php
function descendants($id) {

   $result = pg_query("SELECT id, value FROM adjacency_list_example " .
                      "WHERE parent_id = '$id';");
   $descendants = array();
   while ($row = pg_fetch_array($result)) {
      $descendants[$row["value"]] = descendants($row["id"]);
   }
   return $descendants;

}
?>
```

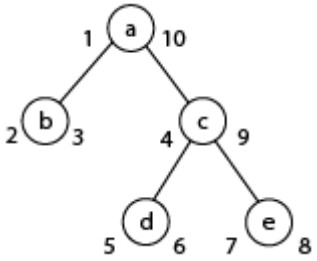
A fa egy elemének a leszármazottainak a lekérdezése.

A szomszédsági lista modell előnye, hogy a beszúrás, a módosítás és a törlés műveletét hatékonyan lehet vele végrehajtani, viszont a lekérdezés művelete a rekurzió miatt nem hatékony.

3.3. Beágyazott halmaz modell

A beágyazott halmaz modellben egy relációs tábla minden eleménél tárolunk egy bal és egy jobb oldali értéket. A bal és a jobb oldali értéket a következő algoritmus alapján kapjuk:

1. i:=1
2. Ha nincs több elem, akkor az algoritmus véget ér.
3. A gyökérelem bal oldali értéke legyen i.
4. i:=i+1
5. Alkalmazzuk az algoritmus 2-7. lépéseit a gyökérelem részfáin (a legbaloldalibb részfával kezdve, a legjobboldalibb részfa fele haladva).
6. A gyökérelem jobb oldali értéke legyen i.
7. i:=i+1



Elem azonosítója	Elem értéke	Bal oldali érték	Jobb oldali érték
1	a	1	10
2	b	2	3
3	c	4	9
4	d	5	6
5	e	7	8

```
CREATE TABLE nested_set_example (

   id SERIAL PRIMARY KEY,
   value CHARACTER(1),
   lft INTEGER NOT NULL,
   rgt INTEGER NOT NULL

);

INSERT INTO nested_set_example (value, lft, rgt) VALUES('a', 1, 10);
INSERT INTO nested_set_example (value, lft, rgt) VALUES('b', 2, 3);
INSERT INTO nested_set_example (value, lft, rgt) VALUES('c', 4, 9);
INSERT INTO nested_set_example (value, lft, rgt) VALUES('d', 5, 6);
INSERT INTO nested_set_example (value, lft, rgt) VALUES('e', 7, 8);
```

Beágyazott halmaz modell példa.

A beágyazott halmaz modellben egy elem leszármazottainak a bal és jobb oldali értéke az elem bal oldali értékénél nagyobb és az elem jobb oldali értékénél kisebb.

```
<?php
function descendants($id) {

   $result = pg_query("SELECT table1.value, table1.lft, table1.rgt " .
             "FROM nested_set_example AS table1, nested_set_example AS table2 " .
             "WHERE table2.id = '$id' " .
             "AND table1.lft > table2.lft AND table1.lft < table2.rgt " .
             "ORDER BY table1.lft;");
   $descendants = array();
   while ($row = pg_fetch_array($result)) {
      array_push($descendants, array($row["value"], $row["lft"], $row["rgt"]));
   }
   return $descendants;

}
?>
```

A fa egy elemének a leszármazottainak a lekérdezése.

Egy új elem beszúrásánál az új elem bal oldali értékével egyenlő vagy nagyobb bal oldali értékekkel rendelkező elemek bal oldali értékét, és az új elem bal oldali értékével egyenlő vagy nagyobb jobb oldali értékkel rendelkező elemek jobb oldali értékét 2-vel kell növelni.

```
<?php
function insert_node($value, $lft, $rgt) {

   $query = "UPDATE nested_set_example SET lft=lft+2 WHERE lft>=$lft;";
   $query .= "UPDATE nested_set_example SET rgt=rgt+2 WHERE rgt>=$lft;";
   $query .= "INSERT INTO nested_set_example (value, lft, rgt) " .
             "VALUES ('$value ', '$lft ', '$rgt ');";
   pg_query($query);

}
?>
```

Új elem beszúrása.

A beágyazott halmaz modell előnye, hogy a lekérdezés műveletét hatékonyan lehet vele végrehajtani, de a beszúrás és a törlés művelete nem hatékony.

3.4. PostgreSQL „WITH RECURSIVE” lekérdezés

A PostgreSQL adatbázis-kezelő rendszerben a „WITH RECURSIVE” lekérdezés segítségével rekurzív problémákat iteratívan tudunk kezelni.

A „WITH RECURSIVE” lekérdezés felépítése:

- nem rekurzív kifejezés
- UNION vagy UNION ALL
- rekurzív kifejezés

A „WITH RECURSIVE” lekérdezés működése:

1. A munka tábla, a köztes tábla és az eredmény tábla az algoritmus kezdetén üres.
2. A nem rekurzív kifejezés végrehajtásra kerül, és az eredménye a munka táblában és az eredmény táblában tárolódik (UNION megadása esetén nincsenek duplikált sorok).
3. Amíg a munka tábla nem üres, addig ismételjük az algoritmus 4-5. lépéseit (az eredményt az eredmény tábla tartalmazza).
4. A rekurzív kifejezés végrehajtásra kerül a munka tábla tartalmával, és az eredménye hozzáadásra kerül a köztes táblához, és az eredmény táblához (UNION megadása esetén nincsenek duplikált sorok).
5. A munka tábla tartalmát felülírjuk a köztes tábla tartalmával, és a köztes tábla tartalmát kiürítjük.

A szomszédsági lista modellben a lekérdezésnél használhatjuk a „WITH RECURSIVE” lekérdezést.

```
WITH RECURSIVE with_recursive_example(id, parent_id, value) AS (
   SELECT id, parent_id, value FROM adjacency_list_example
   WHERE parent_id IS NULL

   UNION

   SELECT al.id, al.parent_id, al.value
   FROM with_recursive_example, adjacency_list_example AS al
   WHERE with_recursive_example.id = al.parent_id)
SELECT value FROM with_recursive_example;
```

A fa elemeinek a lekérdezése.

```
WITH RECURSIVE with_recursive_example(id, parent_id, parent_path, value) AS (
   SELECT id, parent_id, '/', value FROM adjacency_list_example
   WHERE parent_id IS NULL

   UNION

   SELECT al.id, al.parent_id, parent_path || with_recursive_example.id || '/', al.value
   FROM with_recursive_example, adjacency_list_example AS al
   WHERE with_recursive_example.id = al.parent_id)
SELECT parent_path, value FROM with_recursive_example;
```

A fa elemeinek és az elérési útvonalnak a lekérdezése.

4. Hierarchikus feladatkezelő rendszer fejlesztése

4.1. A hierarchikus feladatkezelő rendszer rövid áttekintése

Az elkészítendő rendszer, a rendszer felhasználói számára lehetővé teszi a projektekbe szerveződést. Minden projekthez tartozik egy hierarchia, amely a projektben részt vevő felhasználók kapcsolata alapján kerül felépítésre.

A projektben részt vevő felhasználókhoz feladatokat, a feladatokhoz pedig feladat megjegyzéseket lehet rendelni. A projekt felhasználók tudnak saját maguknak, és a hierarchiában a hozzájuk tartozó leszármazottaknak feladatokat, és a feladatokhoz feladat megjegyzéseket megadni, és tudják ezeknek a projekt felhasználóknak feladatait, és a feladatok feladat megjegyzéseit figyelni.

4.2. Funkcionális követelmények

A rendszer a következő funkcionális követelményeknek kell megfeleljen:

- A rendszer rendelkezik egy főoldallal. A főoldalon a rendszer általános leírása található. A főoldalról a bejelentkezés és a regisztráció elérhető.
- A felhasználók a rendszerbe be tudnak jelentkezni. A bejelentkezéskor a következő adatok megadása szükséges:
    - Felhasználói név.
    - Jelszó.
- A felhasználók regisztrálni tudnak a rendszerbe. A regisztrációkor a következő adatok megadása szükséges:
    - Név.
    - Felhasználói név.
    - Jelszó.
    - Jelszó megerősítése.
    - E-mail cím.

  A sikeres regisztráció után a felhasználók be tudnak jelentkezni a rendszerbe.
- A rendszerbe bejelentkezés után a felhasználó módosítani tudja a nevét, a jelszavát, és az e-mail címét.
- A rendszerből a bejelentkezett felhasználó ki tud jelentkezni.

- A rendszerbe bejelentkezés után a felhasználónak a rendszer kilistázza a projekteket, amelyekben részt vesz. A projekt listát a rendszer rendezni tudja a projekt név, vagy a projekt létrehozásának a dátuma szerint. A lezárt projektek megjelenítését lehet kérni a rendszertől.
- A rendszerbe bejelentkezés után a felhasználó új projekteket létre tud hozni. Új projekt létrehozásakor a következő adatok megadása szükséges:
    - Projekt név.
    - A projekt nyitott vagy lezárt.

  A projektet létrehozó felhasználó lesz a projekt főnöke.
- A projekt főnöke módosítani tudja a projektet.
- A projekt főnöke törölni tudja a projektet, ha a projekt nem lezárt.
- A projektben részt vevő felhasználóknak a rendszer listázni tudja a projektben részt vevő felhasználókat.
- A projektben részt vevő felhasználók a saját részfájukba új felhasználókat felvehetnek, ha a felhasználó még nem szerepel a projektben, és ha a projekt nem lezárt.
- A projektben részt vevő felhasználók (a projekt főnökének a kivételével) törölhetik magukat a projektből, ha a projekt nem lezárt. A törölt projekt felhasználó saját részfájában levő projekt felhasználók a fában egy szinttel feljebb kerülnek.
- A projektben részt vevő felhasználók a saját részfájukból a projekt felhasználókat törölhetik, ha a projekt nem lezárt. A törölt projekt felhasználó saját részfájában levő projekt felhasználók a fában egy szinttel feljebb kerülnek.
- A projektben részt vevő felhasználók áthelyezhetik magukat a saját részfájukon belül a projektben, ha a projekt nem lezárt (viszont a projekt felhasználónak nem lehet saját maga a főnöke). A projekt felhasználó helyére a fában az a projekt felhasználó kerül, a saját részfájával együtt, ahová a projekt felhasználó át lett helyezve. A projekt felhasználóhoz tartozik továbbra is minden olyan projekt felhasználó, aki a projekt felhasználó saját részfájában szerepelt, viszont nem annak a projekt felhasználónak a saját részfájában, ahova a projekt felhasználó át lett helyezve. Ha a projekt főnöke lett áthelyezve, akkor a projekt új főnöke az a projekt felhasználó, akihez a projekt főnöke át lett helyezve.
- A projektben részt vevő felhasználók áthelyezhetik a saját részfájukon belül a projekt felhasználókat a projektben, ha a projekt nem lezárt (viszont a projekt felhasználónak

nem lehet saját maga a főnöke). Ha a projekt felhasználó nem a saját részfájába lett áthelyezve, akkor a projekt felhasználó saját részfájában levő projekt felhasználók továbbra is az áthelyezett projekt felhasználóhoz tartoznak. Ha a projekt felhasználó a saját részfájába lett áthelyezve, akkor a projekt felhasználó helyére a fában az a projekt felhasználó kerül, a részfájával együtt, ahová a projekt felhasználó át lett helyezve. A projekt felhasználóhoz tartozik továbbra is minden olyan projekt felhasználó, aki a projekt felhasználó részfájában szerepelt, viszont nem annak a projekt felhasználónak a saját részfájában, ahova a projekt felhasználó át lett helyezve.

- A projektben részt vevő felhasználók megnézhetik a saját, és a saját részfájukban szereplő projekt felhasználóknak a feladatait. A feladat listát a rendszer rendezni tudja a feladat név, vagy a feladat létrehozásának a dátuma szerint. A lezárt feladatok megjelenítését lehet kérni a rendszertől.
- A projektben részt vevő felhasználók megnézhetik a saját, és a saját részfájukban szereplő projekt felhasználóknak a feladatait, a leszármazottaik feladataival együtt. A feladat listát a rendszer rendezni tudja a feladat név, a feladat létrehozásának a dátuma, vagy a felhasználói név szerint. A lezárt feladatok megjelenítését lehet kérni a rendszertől.
- A projektben részt vevő felhasználók maguknak, és a saját részfájukban szereplő projekt felhasználóknak feladatokat adhatnak, ha a projekt nem lezárt. Új feladat megadásakor a következő adatok megadása szükséges:
    - Feladat név.
    - A feladat nyitott vagy lezárt.
- A projektben részt vevő felhasználók a maguk, és a saját részfájukban szereplő projekt felhasználóknak a feladatait módosíthatják, ha a projekt nem lezárt.
- A projektben részt vevő felhasználók a maguk, és a saját részfájukban szereplő projekt felhasználóknak a feladatait törölhetik, ha a projekt, és a feladat nem lezárt.
- A projektben részt vevő felhasználók megnézhetik a maguk, és a saját részfájukban szereplő projekt felhasználók feladatainak a feladat megjegyzéseit. A feladat megjegyzés listát a rendszer rendezni tudja a feladat megjegyzés, vagy a feladat megjegyzés létrehozásának a dátuma szerint.

- A projektben részt vevő felhasználók a maguk, és a saját részfájukban szereplő projekt felhasználók feladataihoz új feladat megjegyzést létre tudnak hozni, ha a projekt és a feladat nem lezárt.
- A projektben részt vevő felhasználók a maguk, és a saját részfájukban szereplő projekt felhasználók feladatainak a feladat megjegyzéseit módosíthatják, ha a projekt és a feladat nem lezárt.
- A projektben részt vevő felhasználók a maguk, és a saját részfájukban szereplő projekt felhasználók feladatainak a feladat megjegyzéseit törölhetik, ha a projekt és a feladat nem lezárt.

4.3. Tervezés

4.3.1. Adatbázis tervezése

- **users tábla**: A rendszer felhasználóinak az adatait tárolja.

  A tábla mezői:
  - id: A felhasználó azonosítója (elsődleges kulcs).
  - name: A felhasználó teljes neve.
  - username: Felhasználói név.
  - password: A felhasználó jelszava (MD5 kódolással tárolva).
  - email: A felhasználó e-mail címe.
- **projects tábla**: A rendszerben szereplő projektek adatait tárolja.

  A tábla mezői:
  - id: A projekt azonosítója (elsődleges kulcs).
  - name: A projekt neve.
  - date: A projekt létrehozásának a dátuma.
  - closed: A projekt nyitott vagy lezárt.
- **project_users tábla**: A projekt felhasználók adatait tárolja.

  A tábla mezői:
  - id: A projekt felhasználó azonosítója (elsődleges kulcs).
  - project_id: Annak a projektnek azonosítója, amelyikhez a projekt felhasználó tartozik (külső kulcs, amely a projects tábla id mezőjére hivatkozik).
  - user_id: A felhasználó azonosítója (külső kulcs, amely a users tábla id mezőjére hivatkozik).

    - parent_id: A szülő azonosítója (külső kulcs, amely a project_users tábla id mezőjére hivatkozik).
- **tasks tábla**: A projekt felhasználók feladatainak az adatait tárolja.
  A tábla mezői:
    - id: A feladat azonosítója (elsődleges kulcs).
    - project_user_id: Annak a projekt felhasználónak az azonosítója, akihez a feladat tartozik (külső kulcs, amely a project_users tábla id mezőjére hivatkozik).
    - name: A feladat neve.
    - date: A feladat létrehozásának a dátuma.
    - closed: A feladat nyitott vagy lezárt.
- **task_comments tábla**: A projekt felhasználók feladataihoz tartozó feladat megjegyzések adatait tárolja.
  A tábla mezői:
    - id: A feladat megjegyzés azonosítója (elsődleges kulcs).
    - task_id: Annak a feladatnak az azonosítója, amelyikhez a feladat megjegyzés tartozik (külső kulcs, amely a tasks tábla id mezőjére hivatkozik).
    - comment: Feladat megjegyzés.
    - date: A feladat megjegyzés létrehozásának a dátuma.

A külső kulcsoknál, kivéve a project_users tábla parent_id mezőjét, ha a hivatkozott rekord hivatkozott mezőjének az értéke megváltozik, akkor a hivatkozó mezőben is megváltozik ez az érték, ha pedig a hivatkozott rekordot törölték, akkor a hivatkozó rekord is törlésre kerül.

4.3.2. Modell, nézet, vezérlő tervezése

- Modellek:
    - Users: Modell a felhasználók táblához.
    - Projects: Modell a projektek táblához.
    - ProjectUsers: Modell a projekt felhasználók táblához.
    - Tasks: Modell a feladatok táblához.
    - TaskComments: Modell a feladat megjegyzések táblához.
- Nézetek:
    - error: Nézet a hibákhoz.
    - index: Nézetek a főoldalhoz, a bejelentkezéshez, a regisztrációhoz, és a kijelentkezéshez.

- project: Nézetek a projektek létrehozásához, a projektek törléséhez, a projektek módosításához, a projektek megtekintéséhez, és a projektek listázásához.
- projectuser: Nézetek a projekt felhasználók hozzáadásához, a projekt felhasználók törléséhez, a projekt felhasználók módosításához, a projekt felhasználók megtekintéséhez, és a projekt felhasználók és a leszármazottaik feladatainak a megjelenítéséhez.
- settings: Nézet a felhasználók beállításainak a módosításához.
- task: Nézetek a feladatok hozzáadásához, a feladatok törléséhez, a feladatok módosításához, és a feladatok megtekintéséhez.
- taskcomment: Nézetek a feladat megjegyzések hozzáadásához, a feladat megjegyzések törléséhez, a feladat megjegyzések módosításához, és a feladat megjegyzések megtekintéséhez.

- Vezérlők:
  - Error: Vezérlő a hibákhoz.
  - Index: Vezérlő a főoldalhoz, a bejelentkezéshez, a kijelentkezéshez, és a regisztrációhoz.
  - Project: Vezérlő a projektek létrehozásához, a projektek törléséhez, a projektek módosításához, a projektek megtekintéséhez, és a projektek listázásához.
  - Projectuser: Vezérlő a projekt felhasználók hozzáadásához, a projekt felhasználók törléséhez, a projekt felhasználók módosításához, a projekt felhasználók megtekintéséhez, és a projekt felhasználók és a leszármazottaik feladatainak a megjelenítéséhez.
  - Settings: Vezérlő a felhasználók beállításainak a módosításához.
  - Task: Vezérlő a feladatok hozzáadásához, a feladatok törléséhez, a feladatok módosításához, és a feladatok megtekintéséhez.
  - Taskcomment: Vezérlő a feladat megjegyzések hozzáadásához, a feladat megjegyzések törléséhez, a feladat megjegyzések módosításához, és a feladat megjegyzések megtekintéséhez.

4.4. Technológiák

A feladatkezelő rendszer fejlesztésénél használt technológiák:

- Apache HTTP Server 2.2.12

- XHTML 1.1
- CSS 2.1
- JavaScript (ECMAScript 3)
- JQuery 1.4.2
- PHP 5.2.10
- PostgreSQL 8.4.2
- Zend Framework 1.10.2

4.5. Zend Framework projekt

application/configs/application.ini

```
[production]
phpSettings.display_startup_errors = 0
phpSettings.display_errors = 0
phpSettings.date.timezone = "Europe/Budapest"
includePaths.library = APPLICATION_PATH "/../library"
bootstrap.path = APPLICATION_PATH "/Bootstrap.php"
bootstrap.class = "Bootstrap"
appnamespace = "Application"
resources.frontController.controllerDirectory = APPLICATION_PATH "/controllers"
resources.frontController.params.displayExceptions = 0
resources.db.adapter = Pdo_Pgsql
resources.db.params.host =
resources.db.params.username =
resources.db.params.password =
resources.db.params.dbname =
resources.layout.layoutPath = APPLICATION_PATH "/layouts/scripts/"

[testing : production]
phpSettings.display_startup_errors = 1
phpSettings.display_errors = 1

[development : production]
phpSettings.display_startup_errors = 1
phpSettings.display_errors = 1
resources.frontController.params.displayExceptions = 1
```

application/configs/application.ini

Az application.ini fájl direktívákat tartalmaz, amelyek az alkalmazás beállítására szolgálnak. Direktívák például:

- phpSettings.date.timezone: Az időzóna beállítása.
- bootstrap.path: A Bootstrap osztály elérési útvonalának beállítása.
- appnamespace: Az alkalmazás névterének a beállítása.

Az adatbázis elérése megadható az application.ini fájlban a következő direktívák segítségével:

```
resources.db.adapter = Pdo_Pgsql
resources.db.params.host = hoszt_név
resources.db.params.username = felhasználói_név
resources.db.params.password = jelszó
resources.db.params.dbname = adatbázis_név
```

Az adatbázis elérés beállítása.

application/controllers

A tervezésnél megadott vezérlőket tartalmazza: ErrorController.php, IndexController.php, ProjectController.php, ProjectuserController.php, SettingsController.php, TaskcommentController.php, TaskController.php.

Az IndexController.php tartalmazza például a login műveletvezérlőt a bejelentkezéshez. A login műveletvezérlő a bejelentkezéshez szükséges űrlapot elérhetővé teszi a nézetnek. Ha a felhasználó elküldte az űrlapban kitöltött adatokat, akkor megvizsgálja, hogy a beírt adatok megfelelőek-e, és hogy az adatbázisban létezik-e a megadott felhasználói név és jelszó. Ha hibásak a beírt adatok, akkor hibaüzenet lesz az űrlaphoz megadva, egyébként pedig a rendszer tárolja a bejelentkezett felhasználó azonosítóját, és átirányít a projektek listázásához.

```
public function loginAction()
{

   $request = $this->getRequest();
   $form = new Application_Form_Login();

   if ($request->isPost()) {

      if ($form->isValid($request->getPost())) {

         $adapter =
            new Zend_Auth_Adapter_DbTable(Zend_Db_Table::getDefaultAdapter());
```

```
        $adapter->setTableName('users')
                ->setIdentityColumn('username')
                ->setCredentialColumn('password')
                ->setCredentialTreatment('MD5(?)');

        $formValues = $form->getValues();

        $adapter->setIdentity($formValues['username'])
                ->setCredential($formValues['password']);

        $auth = Zend_Auth::getInstance();
        $result = $auth->authenticate($adapter);

        if ($result->isValid()) {

           $auth->getStorage()
                ->write($adapter->getResultRowObject('id'));

           $this->_helper->redirector('list','project');

        } else {

           $error = "Hibás felhasználói név vagy jelszó.";
           $form->getElement('username')->addError($error);
           $form->getElement('password')->addError($error);

        }

      }

   }

   $this->view->form = $form;

}
```

IndexController.php: login műveletvezérlő a bejelentkezéshez.

application/forms

Az űrlap osztályokat tartalmazza: Login.php (Űrlap osztály a bejelentkezéshez.), Project.php (Űrlap osztály a projektekhez.), ProjectUser.php (Űrlap osztály a projekt felhasználókhoz.), ProjectUserEdit.php (Űrlap osztály a projekt felhasználók módosításához.), Register.php (Űrlap osztály a regisztrációhoz.), Settings.php (Űrlap osztály a beállításokhoz.), Task.php (Űrlap osztály a feladatokhoz.), TaskComment.php (Űrlap osztály a projekt megjegyzésekhez.).

Például a Login.php tartalmaz egy felhasználói név mezőt, egy jelszó mezőt, és egy bejelentkezés gombot. Minden adat kitöltése szükséges a bejelentkezéshez. A felhasználói név csak betűket, és számokat tartalmazhat.

```
<?php

class Application_Form_Login extends Zend_Form
{

   public function init()
   {

      $this->setMethod('post');

      $this->addElement('text', 'username', array(
         'label' => 'Felhasználói név:',
         'required' => true,
         'filters' => array('StripTags', 'StringTrim'),
         'validators' => array('NotEmpty', 'Alnum')
      ));

      $this->addElement('password', 'password', array(
         'label' => 'Jelszó:',
         'required' => true,
         'filters' => array('StripTags', 'StringTrim'),
         'validators' => array('NotEmpty')
      ));
```

```
        $this->addElement('submit', 'submit', array(
            'ignore' => true,
            'label' => 'Bejelentkezés'
        ));

    }


}
```

Application/Form/Login.php

application/layouts/scripts/layout.phtml

A nézeteknél használt layout.

application/models/DbTable

A tervezésnél megadott modelleket tartalmazza: Projects.php, ProjectUsers.php, TaskComments.php, Tasks.php, Users.php.

Például a Users.php tartalmaz egy addUser metódust egy felhasználó hozzáadásához a rendszerbe.

```
public function addUser( $post )
{

    $data = array( 'name' => $post['name'],
                   'username' => $post['username'],
                   'password' => MD5($post['password']),
                   'email' => $post['email']);

    $this->insert($data);

}
```

Users.php: Metódus egy felhasználó hozzáadásához a rendszerbe.

application/models/Acl.php

Tartalmazza annak megadását, hogy a bejelentkezett, és a nem bejelentkezett felhasználók mely vezérlők mely műveletvezérlőihez férhetnek hozzá.

```
<?php

class Model_Acl extends Zend_Acl
{

   public function __construct()
   {

      $this->addRole(new Zend_Acl_Role('guest'));

      $this->addRole(new Zend_Acl_Role('user'), 'guest');

      $this->add(new Zend_Acl_Resource('error'));
      $this->add(new Zend_Acl_Resource('index'));
      $this->add(new Zend_Acl_Resource('project'));
      $this->add(new Zend_Acl_Resource('projectuser'));
      $this->add(new Zend_Acl_Resource('settings'));
      $this->add(new Zend_Acl_Resource('task'));
      $this->add(new Zend_Acl_Resource('taskcomment'));

      $this->allow('guest', 'error');
      $this->allow('guest', 'index');

      $this->allow('user', 'index', 'logout');
      $this->allow('user', 'project');
      $this->allow('user', 'projectuser');
      $this->allow('user', 'settings');
      $this->allow('user', 'task');
      $this->allow('user', 'taskcomment');
```

```
        $this->deny('guest', 'index', 'logout');
        $this->deny('user', 'index', array('index', 'login', 'register'));

    }


}
```

application/models/Acl.php

application/views/scripts

A tervezésnél megadott nézeteket tartalmazza: error, index, project, projectuser, settings, task, taskcomment.

Például az index/login.phtml tartalmazza a „Bejelentkezés” címet, és a bejelentkezéshez szükséges űrlapot.

```
<h2>Bejelentkezés</h2>

<?php
$this->form->setAction($this->url());
echo $this->form;
```

index/login.phtml

application/bootstrap.php

A Bootstrap osztály az erőforrások és a komponensek inicializálását tartalmazza. Például a nézet inicializálása tartalmazza a dokumentum típus, a tartalom típus, és az oldal címének a beállítását.

```
protected function _initView()
{

    $view = new Zend_View();

    $view->doctype('XHTML11');

    $view->headMeta()->appendHttpEquiv('Content-Type', 'text/html; charset=UTF-8');

    $view->headTitle('Feladatkezelő Rendszer');
```

```
    $viewRenderer =
        Zend_Controller_Action_HelperBroker::getStaticHelper('ViewRenderer');
    $viewRenderer->setView($view);

    return $view;
}
```

A nézet inicializálása.

library/TaskManager/Controller/Plugin/Acl.php

```
<?php

class TaskManager_Controller_Plugin_Acl extends Zend_Controller_Plugin_Abstract
{

    private $_acl = null;
    private $_auth = null;

    public function __construct(Zend_Acl $acl, Zend_Auth $auth) {

        $this->_acl = $acl;
        $this->_auth = $auth;

    }

    public function preDispatch(Zend_Controller_Request_Abstract $request) {

        if ($this->_auth->hasIdentity()) {
            $role = 'user';
        } else {
            $role = 'guest';
        }

        $controller = $request->getControllerName();

        $action = $request->getActionName();
```

```
      if(!$this->_acl->isAllowed($role, $controller, $action)) {

        if (!$this->_auth->hasIdentity()) {
          $request->setControllerName('index')
                  ->setActionName('index');
        } else {
          $request->setControllerName('project')
                  ->setActionName('list');
        }

      }

    }


}
```

library/TaskManager/Controller/Plugin/Acl.php

Ha a felhasználó nincs bejelentkezve, és egy vezérlő olyan műveletvezérlőjéhez szeretne hozzáférni, amihez be kell jelentkezni, akkor a főoldalra lesz átirányítva. Ha a felhasználó be van jelentkezve, és egy vezérlő olyan műveletvezérlőjéhez szeretne hozzáférni, amihez ki kell jelentkezni, akkor a projektek listázásához lesz átirányítva.

library/Zend

A Zend Framework 1.10.2 könyvtárat kell tartalmazza. A szakdolgozathoz mellékelt forráskódokat tartalmazó fájl ezt a könyvtárat nem tartalmazza, ezért a rendszer működéséhez a Zend Framework 1.10.2 letöltése szükséges a hivatalos Zend Framework honlapról.

languages/Hungarian.php

Az alkalmazás által használt magyar nyelvű üzeneteket tartalmaz.

public/css/global.css

Az alkalmazás által használt CSS stíluslap.

public/js/jquery/jquery-1.4.2.min.js

A JQuery 1.4.2 JavaScript fájl. A szakdolgozathoz mellékelt forráskódokat tartalmazó fájl ezt a JavaScript fájlt nem tartalmazza, ezért a rendszer működéséhez a JQuery 1.4.2 letöltése szükséges a hivatalos JQuery honlapról.

public/js/jquery/plugins/jquery-treeview

A Treeview[5] JQuery beépülőt kell tartalmazza, amely listák fa szerkezetben történő megjelenítését teszi lehetővé. A szakdolgozathoz mellékelt forráskódokat tartalmazó fájl ezt a JQuery beépülőt nem tartalmazza, ezért a rendszer működéséhez ennek a letöltése szükséges. A feladatkezelő rendszer a jquery.treeview.js, és a jquery.treeview.css fájlokat használja, ezért ezeket a fájlokat tartalmaznia kell.

public/js/projectusers.js

A projekt projekt felhasználóinak a megjelenítésekor használt JavaScript fájl.

public/.htaccess

A Zend_Tool_Project által létrehozott. htaccess fájl az URL átíráshoz.

public/index.php

A Zend_Tool_Project által létrehozott belépési pont az alkalmazásba.

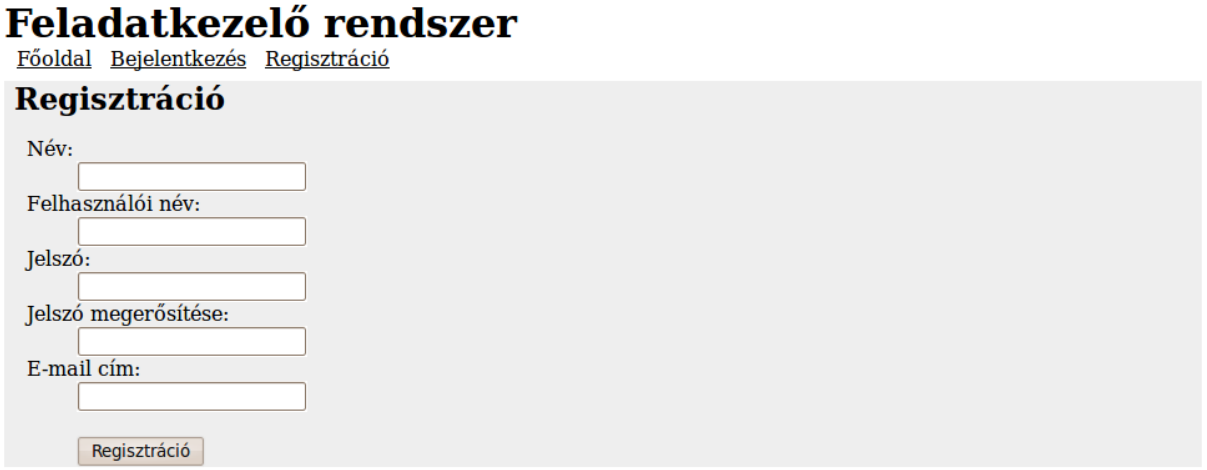
[5] http://bassistance.de/jquery-plugins/jquery-plugin-treeview/ [2010.04.28.]

5. Néhány kép a feladatkezelő rendszerről

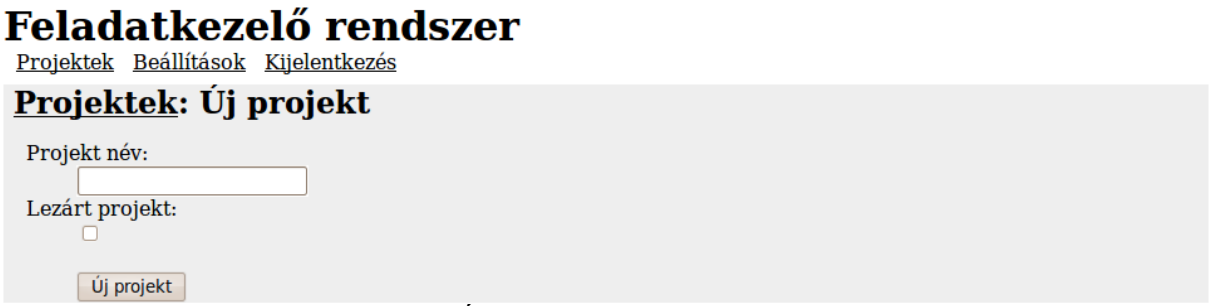
A weboldal megjelenítésére használt böngésző: Firefox 3.5.9.

Főoldal.

Bejelentkezés.

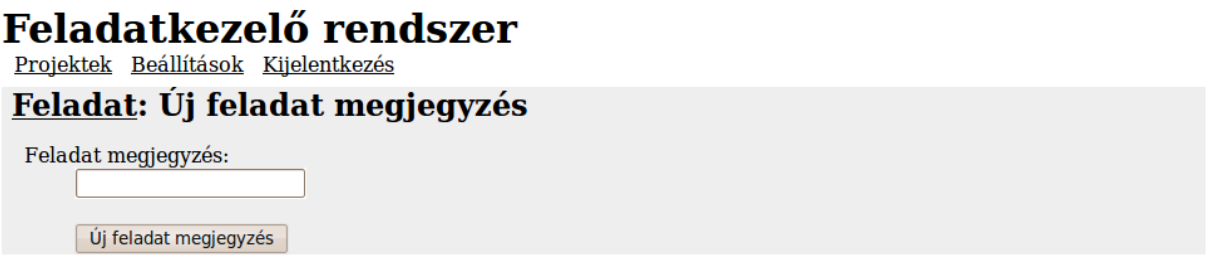


Regisztráció.



Új projekt létrehozása.

Új feladat hozzáadása a projekt felhasználóhoz.



Új feladat megjegyzés hozzáadása a feladathoz.

6. Összefoglalás

A szakdolgozatban egy feladatkezelő rendszer webes technológiák segítségével történő megvalósítását ismertettem.

A következő témaköröket mutattam be:

- A szakdolgozat témájának a bemutatása.
- A rendszer elkészítésénél felhasznált webes technológiák bemutatása.
- A hierarchikus adatok kezelésének lehetőségei relációs adatbázis kezelő rendszerekben.
- A feladatkezelő rendszer fejlesztésének a bemutatása.

Úgy gondolom, hogy a rendszer fejlesztésénél felhasznált webes technológiák jól strukturált lehetőséget adnak a fejlesztőknek webes alkalmazások készítéséhez.

7. Köszönetnyílvánítás

Köszönöm mindenkinek, aki segítséget nyújtott, hogy a szakdolgozat elkészülhessen. Köszönöm témavezetőmnek, Kósa Márk Szabolcs egyetemi tanársegéd úrnak a szakdolgozat elkészítésében nyújtott segítségét, továbbá köszönöm a Debreceni Egyetem tanárainak az oktatást.

8. Irodalomjegyzék

[1] Tim Berners-Lee
Information Management: A Proposal
http://www.w3.org/History/1989/proposal.html [2010.04.28.]

[2] T. Berners-Lee, R. Cailliau
WorldWideWeb: Proposal for a HyperText Project
http://www.w3.org/Proposal.html [2010.04.28.]

[3] Architecture of the World Wide Web, Volume One
http://www.w3.org/TR/webarch/ [2010.04.28.]

[4] RFC 3305
http://tools.ietf.org/html/rfc3305 [2010.04.28.]

[5] RFC 3986 Uniform Resource Identifier (URI): Generic Syntax
http://tools.ietf.org/html/rfc3986 [2010.04.28.]

[6] RFC 2045 Multipurpose Internet Mail Extensions (MIME) Part One:
Format of Internet Message Bodies
http://tools.ietf.org/html/rfc2045 [2010.04.28.]

[7] RFC 2046 Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types
http://tools.ietf.org/html/rfc2046 [2010.04.28.]

[8] RFC 2616 - Hypertext Transfer Protocol -- HTTP/1.1
http://tools.ietf.org/html/rfc2616 [2010.04.28.]

[9] Apache HTTP Server Version 2.2 Documentation
http://httpd.apache.org/docs/2.2/ [2010.04.28.]

[10] HTML 4.01 Specification
http://www.w3.org/TR/1999/REC-html401-19991224/ [2010.04.28.]

[11] XHTML 1.1 - Module-based XHTML
http://www.w3.org/TR/xhtml11/ [2010.04.28.]

[12] XHTML Modularization 1.1
http://www.w3.org/TR/xhtml-modularization/ [2010.04.28.]

[13] Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification
http://www.w3.org/TR/CSS21/ [2010.04.28.]

[14] ECMAScript Language Specification Edition 3 Final
http://www.mozilla.org/js/language/E262-3.pdf [2010.04.28.]

[15] JQuery
http://jquery.com/ [2010.04.28.]

[16] PHP
http://www.php.net/ [2010.04.28.]

[17] PostgreSQL
http://www.postgresql.org/ [2010.04.28.]

[18] Zend Framework
http://framework.zend.com/ [2010.04.28.]

[19] Storing Hierarchical Data in a Database
http://articles.sitepoint.com/article/hierarchical-data-database [2010.04.28.]
[20] PostgreSQL 8.4: WITH Queries
http://www.postgresql.org/docs/8.4/static/queries-with.html [2010.04.28.]
[21] Postgresql 8.4 Recursive Queries
http://www.storytotell.org/blog/2009/08/11/postgresql84-recursive-queries.html [2010.04.28.]
[22] jQuery plugin: Treeview
http://bassistance.de/jquery-plugins/jquery-plugin-treeview/ [2010.04.28.]
[23] Zendframework tutorial: Developing a Blog using version 1.9
http://www.harikt.com/content/simple-blog-using-zend-framework-19 [2010.04.28.]
[24] Zend Framework sign up and authentication
http://zendgeek.blogspot.com/2009/07/zend-framework-sign-up-and.html [2010.04.28.]
[25] Zend Framework Forum
http://www.zfforums.com/ [2010.04.28.]
[26] Zend Framework Acl with example
http://zendguru.wordpress.com/2008/11/05/zend-framework-acl-with-example/ [2010.04.28.]
[27] Zend_Db_Select multiple table joins explained
http://think-robot.com/2009/04/zend_db_select-multiple-table-joins-explained/ [2010.04.28.]
[28] Zend Framework Application Progress
http://www.ecitadel.net/blog/2010/01/02/zend-framework-application-progress [2010.04.28.]
[29] Example Zend Framework Blog Application Tutorial - Part 7: Authorisation with Zend_Acl and Revised Styling
http://blog.astrumfutura.com/archives/362-Example-Zend-Framework-Blog-Application-Tutorial-Part-7-Authorisation-with-Zend_Acl-and-Revised-Styling.html [2010.04.28.]
[30] Example Zend Framework Blog Application Tutorial: Part 8: Creating and Editing Blog Entries with a dash of HTMLPurifier
http://blog.astrumfutura.com/archives/365-Example-Zend-Framework-Blog-Application-Tutorial-Part-8-Creating-and-Editing-Blog-Entries-with-a-dash-of-HTMLPurifier.html [2010.04.28.]