UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Urška Valenčič

Razvoj poslovne spletne skupnosti z orodjem Drupal

DIPLOMSKO DELO NA UNIVERZITETNEM ŠTUDIJU

Mentor: doc. dr. Matjaž Kukar

Ljubljana, 2010

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01686/2010
Datum: 01.09.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **URŠKA VALENČIČ**

Naslov: **RAZVOJ POSLOVNE SPLETNE SKUPNOSTI Z ORODJEM DRUPAL**
BUSINESS-ORIENTED SOCIAL NETWORKING FOR DRUPAL

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Spletne skupnosti (socialna omrežja) v zadnjem času postajajo močno orodje za povezovanje in sodelovanje posameznikov in podjetij. Vendar pa prevladujoče spletne rešitve ne ustrezajo vedno potrebam posameznih skupnosti. Razvijte dodatek za podporo konkretnim zahtevanim funkcionalnostim poslovne spletne skupnosti v odprtokodnem sistemu za upravljanje s spletnimi vsebinami Drupal. Opredelite prednosti, ki jih ponuja vaš pristop pred prevladujočimi spletnimi socialnimi omrežji (Facebook, LinkedIn, ...), opišite implementacijo v Drupalu in njene zmogljivosti, ter praktične izkušnje končnih uporabnikov.

Mentor:
doc. dr. Matjaž Kukar

Dekan:
prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Namesto te strani vstavite original izdane teme diplomskega dela s podpisom mentorja in dekana ter žigom fakultete, ki ga diplomant dvigne v študentskem referatu, preden odda izdelek v vezavo!

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani/-a Urška Valenčič,

z vpisno številko 63040173,

sem avtor/-ica diplomskega dela z naslovom:

Razvoj poslovne spletne skupnosti z orodjem Drupal

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom

  doc. dr. Matjaža Kukarja

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne 15.12.2010 Podpis avtorja/-ice:

Zahvala

Najprej bi se zahvalila mentorju doc. dr. Matjažu Kukarju za vse napotke, nasvete in strokovno pomoč.

Zahvaljujem se podjetju Tesla, ki mi je ponudilo študijsko prakso, v okviru katere je kasneje nastalo to diplomsko delo.

Najlepša hvala staršem, ki so mi omogočili študij, me tekom študija moralno in materialno podpirali, predvsem pa, ker so zaupali vame in v moje odločitve. Vem da to velikokrat ni bilo lahko, še posebej ob mojemu odhodu v tujino. Hvala vama.

Posebno zahvalo bi namenila možu, ki me je bodril in spodbujal ob zaključku študija.

Kazalo

Seznam uporabljenih kratic in simbolov

ACID	Atomicity, Consistency, Isolation, Durability
AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
CMS	Content Management System
CSS	Cascading Style Sheets
CVS	Concurrent Versions System
DOM	Document Object Model
GPL	General Public Licence
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
IRC	Internet Relay Chat
PHP	Hypertext Pre-Processor
URL	Uniform Resource Locator
WCMS	Web Content Management System
WYSIWYG	What You See Is What You Get
XML	Extensible Markup Language

Povzetek

Diplomsko delo obravnava področje zaprtih virtualnih skupnosti. Namen dela je predstaviti obstoječe socialne mreže in spletne sisteme, ki omogočajo sodelovanje uporabnikov znotraj zaprtih skupin ter implementirati ustrezno rešitev, ki bo zadovoljila uporabniške zahteve zaprte poslovne spletne skupnosti. Analiza uporabniških zahtev je predstavljena z uporabniškimi zgodbami in uporabniškimi primeri. Manjša zaprta poslovna spletna skupina bi lahko zaživela kot del trenutno dosegljivih poslovnih socialnih mrež, kot sta LinkedIn in Xing. Vendar nam te, z rastjo skupnosti in njenih potreb, ne omogočajo možnosti prilagoditve in razširitve z nadgradnjami funkcionalnosti znotraj posamezne skupine. V ta namen je potrebno preučiti sisteme za upravljanje s spletnimi vsebinami, ki bi bili ustrezni za razvoj nove poslovne spletne skupnosti.

Cilj diplomskega dela je razviti zaprto poslovno spletno skupnost s sistemom za upravljanje s spletnimi vsebinami. Izbrala sem sistem Drupal, ki je preprosto razširljiv in prilagodljiv CMS sistem z močno skupnostjo uporabnikov in razvijalcev. Implementacija rešitve obsega poleg osnovne namestitve sistema Drupal 5, namestitev potrebnih dodatnih modulov in razvoj novega modula. Rezultat dela je zaprta poslovna spletna skupnost, ki jo uporablja množica podjetnikov v Goteborgu. Sistem nudi veliko možnosti za razširitev in nadgradnjo v prihodnosti. Trenutna rešitev bo delovala brezhibno tudi, ko bo število članov skupnosti naraslo na tisoč. V prihodnosti pa je priporočljiva nadgradnja celotnega sistema na Drupal 6 oziroma Drupal 7, ki prinašata optimizacijo kode, povečano varnost in večje možnosti za skalabilnosti sistema.

Ključne besede:

CMS, Drupal, poslovna spletna skupnost, sistemi za upravljanje spletnih vsebin

Abstract

This thesis studies closed virtual communities. The aim of the thesis is to present existing social networks and online community systems that enable collaboration within closed groups and finaly to implement a business-oriented community web page. The web community has to satisfy user requirements. User requirements analysis is performed and presented with user stories and use cases. Closed group within existing business social networks like LinkedIn and Xing, could be a simple solution in the beginning. But, with the growth of group's members and their needs, these systems do not offer enough flexibility, customization or extensibility for the group. In order to build a new web community, a study of existng CMS systems is conducted.

The goal of the thesis is to develop a closed business web community using web content management system. I have chosen Drupal, because it is easily extensible, flexible and designed to be customized. Drupal has a strong online community of users and web developers, which provides a good support. The implementation of my solution includes the basic installation of Drupal 5 system, installation of needed add-on modules and the development of a new Drupal module. The result is an active business web community for entrepreneurs in Goteborg. The solution offers high level of extensibility and upgrade possibilities in the future. The current system can handle up to thousand members of the community. Although an upgrade to Drupal 6 or Drupal 7 is highly recommended in the future. New versions bring optimized code, increased security and greater scalability options.

Key words:

CMS, Drupal, business-oriented online community, web content management systems

1 Uvod

Razvoj spletnih tehnologij nas je v zadnjih dveh desetletjih pripeljal od statičnih HTML strani preko strežniških skriptnih jezikov in aplikacijskih strežnikov, ki omogočajo gradnjo dinamičnih spletnih strani, do pojava pojma Web 2.0. Razvoj spletnih strani je od samega začetka težil k hitrejši odzivnosti z izmenjavo manjših količin podatkov s strežnikom v ozadju. Web 2.0 predstavlja nov pogled na namembnost spletnih strani in prispeva k interakciji uporabnikov. Uporabniki lahko na spletu sodelujejo med sabo, si izmenjujejo informacije, kreirajo vsebino in so del virtualne skupnosti. Uporabniki so torej vključeni v proces nastanka same spletne vsebine in niso zgolj pasivni opazovalci. Produkti Web 2.0 tehnologij so spletne skupnosti, socialne mreže, blogi in forumi. Z uporabo AJAX tehnologij, ki avtomatično posodobijo ustrezne dele strani, se je povečalo zadovoljstvo uporabnika.

Socialne mreže danes niso več namenjene zgolj za družabne namene. Njihove prednosti so začela koristiti tudi podjetja. Razširil se je virusni marketing. Podjetja so začela uporabljati spletne sisteme kot so Wiki, poslovne socialne mreže in zaprte poslovne spletne skupnosti. Prednosti poslovne spletne skupnosti so vzdrževanje in zbiranje intelektualnega kapitala, povečana kreativnost, podpora odločanju, ohranjanje znanja organizacij, večja interaktivnost in sodelovanje znotraj organizacije.

Podjetje ALMI, ki ima pod okriljem 17 podjetij na Švedskem, nudi malim in srednje velikim podjetjem pomoč in podporo pri razvoju ter vzpodbuja novonastala podjetja pri inovaciji. Iskali so spletni sistem, ki bi mladim podjetnikom nudil različne možnosti sodelovanja, komunikacije in izmenjave znanja z bolj izkušenimi podjetniki. Švedsko podjetje Tesla, ki nudi tehnološke rešitve pri razvoju spletnih strani, jim je predstavilo spletno skupnost KosmAgora. Pričujoče diplomsko delo je rezultat projekta KosmAgora, ki sem ga razvila za potrebe podjetja Tesla. Cilj diplomske naloge je razvoj poslovne spletne skupnosti v sistemu za upravljanje z vsebinami. Potrebno je najti ustrezen sistem, ki bo omogočal zahtevane funkcionalnosti spletne skupnosti. V prvem poglavju analiziram potrebe uporabnikov in opišem zahteve spletnega sistema z uporabniškimi zgodbami in uporabniškimi primeri. Glede na uporabniške zahteve bom raziskala obstoječe rešitve in utemeljila izbiro CMS sistema Drupal. Četrto poglavje opisuje sisteme za upravljanje spletnih vsebin Drupal, njegov uporabniški vmesnik, administracijski del in glavne gradnike. Predstavila bom osnovne komponente sistema, povezanost med njimi in njihovo uporabnost. Opisala in utemeljila bom izbiro dodatnih modulov, ki so prispevali k implementaciji rešitve. V petem poglavju se poglobim v zaledni del sistema. Najprej predstavim datotečno strukturo in podatkovno bazo, nato pa pojasnim posamezne dele sistema, kot so kljuke in programski vmesnik Form API. To poglavje je podprto s primerom razvitega modula *Mypage*, ki je nastal v praktičnem delu diplomske naloge. Šesto poglavje opisuje korake testiranja spletne skupnosti. Podan je test razvitega modula in orodja, ki beležijo zmogljivost sistema. Nato analiziram rezultate obremenitvenih testov in zaključim z opisom uporabniške izkušnje. Diplomsko delo je zaključeno s sklepnimi ugotovitvami in predstavitvijo možnosti za nadaljni razvoj.

2 Zajem zahtev

Analiza potreb in zahtev uporabnikov predstavlja osnovo za nadaljni proces razvoja spletne strani. Potrebno je definirati cilje in omejitve ter razumeti potrebe vseh vpletenih v projektu. Najprej moramo razmisliti kdo vse bo vpleten v naš projekt in na kakšen način ter kateri tipi uporabnikov bodo uporabljali našo spletno skupnost in kako bodo ti sodelovali med seboj. Poleg tega bo potrebno analizirati in zasnovati potek strani preko katerega bo uporabnik prispel do svojega cilja. [2]

Za pridobitev vseh potrebnih informacij sem izvedla analizo uporabniških primerov (angl. use case analysis) in uporabniških zgodb (angl. user stories). Uporabniške zgodbe ponavadi napiše naročnik sistema, v praksi pa se pogosto izoblikujejo iz zapiskov rednih sestankov z naročnikom. Tako je bilo tudi pri projektu KosmAgora.

Da bomo pravilno definirali uporabniške vloge, moramo vedeti kdo so uporabniki spletne skupnosti. To so podjetniki, ki v skupnosti predstavljajo svoje podjetje. Poleg njih bo tudi nekaj izkušenih podjetnikov, vodij spletne skupnosti. Ne smemo pozabiti tudi na nekaj uporabnikov, ki bo zašlo na našo spletno stran, njim bomo rekli anonimni obiskovalci. Pravkar smo definirali štiri uporabniške vloge:

- Anonimni obiskovalec
- Administrator
- Član
- Vodja

2.1 Uporabniške zgodbe

Z uporaniškimi zgodbami se analizirajo različni primeri uporabe in funkcionalnosti spletne strani.

Uporabniška zgodba 1: Kontrola in način dostopa za različne tipe uporabnikov

Administrator ima možnost določiti dostop do vozla vsebine posameznim uporabniškim vlogam. Določene vsebine (vozli) naj bodo dostopni vsem članom, ostali vozli pa samo članom z uporabniško vlogo vodje. Naslovna stran *www.kosmagora.com/se* naj prikazuje samo osnovne informacije o spletni skupnosti KosmAgora in polje za prijavo (angl. login). Naslovna stran bo imela ločeno podobo od vseh ostalih strani ter ne bo vsebovala menijev in ostalih blokov. Samo prijavljeni uporabniki imajo dostop do ostalih vsebin. Potrebno je poskrbeti, da ne bo prišlo do vdorov anonimnih uporabnikov v spletno skupnost.

Uporabniška zgodba 2: Včlanitev in prijava

Nov uporabnik spletne skupnosti lahko postane član z vabilom (angl. invitation) enega od članov. Ko član spletne skupnosti pošlje povabilo novemu uporabniku, se ta s povabilom registrira in postane član. V povabilu je povezava s ključem na stran vpisa. Uporabnik na strani vpisa najprej sprejme pogoje uporabe, vnese potrebna polja in potrdi prijavo. Obvezna polja naj bodo uradno ime podjetja in davčna številka, neobvezna pa prevladujoč jezik, naslov in telefon, sektor ter leto ustanovitve. Vsak član naj ima možnost povabljanja neomejenega števila uporabnikov. Vodje naj imajo pregled nad številom povabljenih članov s strani posameznega uporabnika. Povabilo naj bo enostavno elektronsko sporočilo, ki ga lahko vodje poljubno urejajo. Administrator naj ima še vedno možnost stvaritve novega člana. Ob registraciji mora vsak član obvezno izpolniti naslednja polja: uporabniško ime in geslo, veljaven elektronski poštni naslov, profilna slika, ime podjetja, šifra podjetja, leto ustanovitve in potrditev, da se strinja s pogoji uporabe spletne skupnosti. Poleg tega mora obvezno izbrati vsaj dva jezika, v katerem deluje organizacija. Kasneje pa lahko uporabnik izpolni še polja stroka, naslov podjetja in njegovo telefonsko številko. Zaradi varnosti naj bo dodatek vpisnega lista tudi polje za preverjanje človečnosti (angl. captcha).

Uporabniška zgodba 3: Meniji in URL naslovi

Spletna skupnost naj ima dve vrsti menijev, ena naj bo namenjena upravljanju lastnih vsebin. Na primer povezave na moj blog, uporabniški račun, lastno predstavitveno stran itd. Temu meniju bomo rekli »osebni meni«. Drugi meni, pa naj vsebuje povezave na sklope vsebine skupnosti: *Home, AboutKosmagora, Members, Forums, Groups, Polls* in *Blogs*. Temu meniju rečemo tudi tudi »meni primarnih povezav«. Hierarhija menijski elementov ne sme biti pregloboka, oz. če že je, naj se vgnezdeni meniji kar se da hitro prikažejo.
URL naslovi naj bodo čimbolj intuitivni, če se nahajamo na meniju *Mypage/Transporters/Edit* naj bo tudi URL naslov temu primeren, npr. *http://www.kosmagora.com/mypage/transporters/edit*.

Uporabniška zgodba 4: Blog

Vsak uporabnik lahko piše svoj blog znotraj skupnosti. Blog naj bo zelo preprost in služi za ustvarjanje objav posameznih uporabnikov.

Uporabniška zgodba 5: Forum

Eden od elementov menija naj bo forum. Forum naj bo dosegljiv za vse člane. Uporabnik z vlogo člana lahko objavi komentar, uporabnik z vlogo vodje pa poleg tega lahko ustvari tudi novo temo. Naj bo del foruma zaprt za vodje.

Uporabniška zgodba 6: Ankete

Eden od elementov menija naj bodo ankete (angl. polls). Vodje naj imajo možnost ustvarjanja in pregledovanja anket, ki jih izpolnjujejo vsi člani. Ankete naj se nahajajo med primarnimi povezavami.

Uporabniška zgodba 7: Urejanje vsebine

Uporabnik naj ima na voljo za vnos vsebine urejevalnik besedila, ki bo vključeval vse osnovne funkcije urejanja kot so velikost, tip in barva pisave, poravnava besedila, oblika besedila (krepko, ležeče, podčrtano, prečrtano), tabela in seznami. Poleg tega pa naj ima uporabnik možnost vključevanja slik in upravljanja z lastnimi slikami. Če je slika prevelika, naj se avtomatično pomanjša na ustrezno velikost.

Uporabniška zgodba 8: Oglaševanje

Slikovni oglasi so namenjeni pridobivanju večjega števila članov. Vodje naj imajo možnost določitve kateri oglas se bo postavil kam in kdaj se bo oglas prikazal. Oglasi se bodo nahajali na desnem robu strani. Maksimalno število dovoljenih oglasov lahko spreminja le administrator. Naj se meri tudi statistika uporabe oglasov.

Uporabniška zgodba 9: Skupine

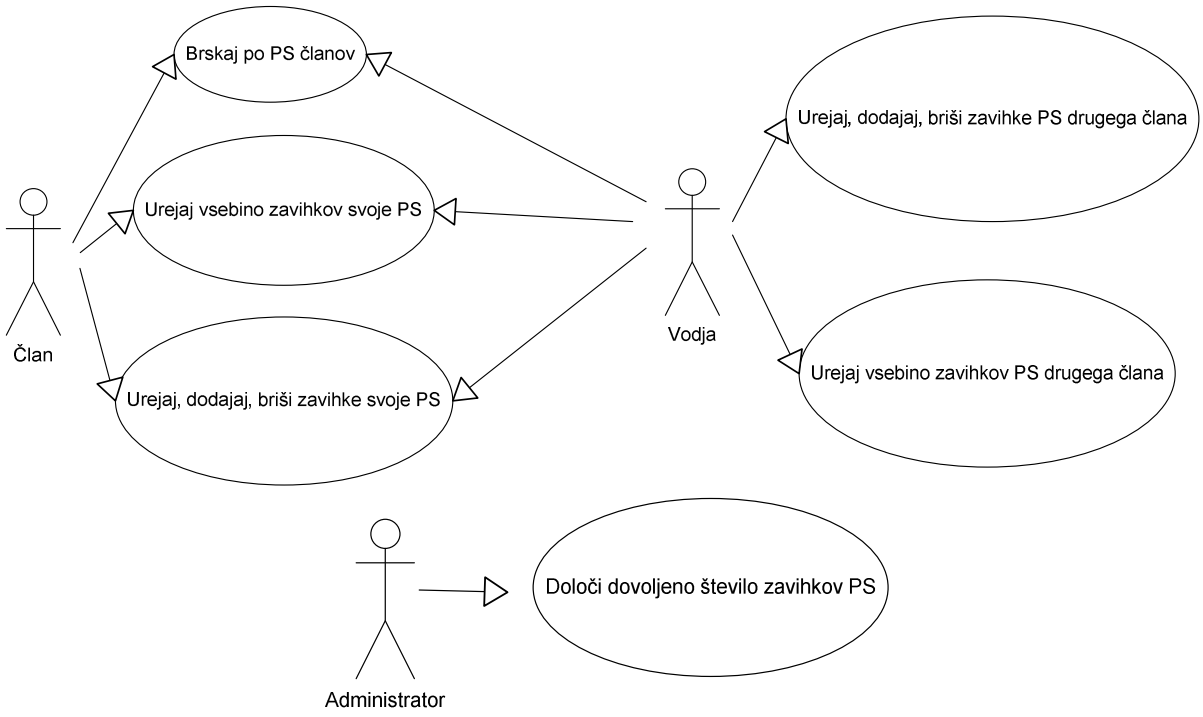
Želimo imeti več zaprtih skupin in eno odprto skupino za vse člane. Skupina *Active group* je odprta skupina, ki bo namenjena objavi informacij, novic in podobno. Poleg nje imamo dve skupini, ki sta namenjeni povezavi udeležencev mentorskih programov, to sta skupini *MP March 2010* in *MP May 2010*. Poleg teh skupin imamo še zaprto skupino za administratorje in vodje, z imenom *KosmAgora Development*. Člani skupine lahko naložijo (angl. upload) in prenesejo (angl. download) datoteke in objavijo komentarje. Vodje skupin imajo možnost ustvarjanja novih skupin, upravljanja članov, objavo prispevkov, datotek in obrazcev. Obrazce, ki jih ustvarijo vodje za določeno skupino morajo nato izpolniti njeni člani. Vodje naj imajo pregled nad izpolnjenimi obrazci in možnost izvoza v Excel dokument. Ustvarjanje obrazcev naj vključuje možnost izbora iz spustnega seznama, vnos teksta (z omejitvijo znakov) in datuma.

Uporabniška zgodba 10: Jezik

Spletna skupnost deluje v dveh jezikih, švedskem in angleškem. Vsak član naj ima možnost izbire jezika ob registraciji.

2.2 Uporabniški primer: Predstavitvene strani podjetja

Ker se v spletni skupnosti za vsakim članom v resnici skriva podjetje, potrebujemo manjšo predstavitveno stran posameznega podjetja (v nadaljevanju PS podjetja), ki bo delovala kot neka spletna stran podjetja znotraj skupnosti. Zahteva po predstavitveni strani članov spletne skupnosti je bila s strani naročnika zelo specifična, zato smo to uporabniško zahtevo opisali z uporabniškimi primeri. Uporabniški primer definira zaporedje akcij sistema, ki pripelje do vidnega rezultata določenemu uporabniku. Uporabniški primeri so nam v pomoč pri doseganju ciljev uporabnikov, uporabniški scenariji pa podrobno opisujejo kako dosežemo določen cilj. [1] Uporabniški diagram PS podjetij prikazuje cilje treh vrst uporabnikov: vodje, člana in administratorja.



Slika 1. Uporabniški diagram PS podjetij.

Uporabniški scenarij: »Brskaj po PS članov«

Opis: Člani oz. uporabniki spletne skupnosti lahko brskajo po predstavitvenih straneh ostalih članov.
Akterji: Član spletne skupnosti (v nadaljevanju uporabnik) z vlogo *Član* ali/in vlogo *Vodja*, sistem.
Predpogoji: /
Rezultat: Prikazana vsebina predstavitvene strani izbranega člana.
Osnovni potek:

1. Sistem prikaže pod primarno povezavo *Members* seznam članov s povezavami na njihove predstavitvene in kontaktne strani.

2. Uporabnik izbere enega od članov in klikne na povezavo do njegove predstavitvene strani.
3. Sistem prikaže predstavitveno stran izbranega člana z zavihki in vsebino.

Uporabniški scenarij: »Urejaj vsebino zavihkov svoje PS«

Opis: Vsak član spletne skupnosti ima možnost urejanja vsebine lastne predstavitvene strani. Ureja lahko vsebino pod zavihki svoje PS in ima možnost pregleda svoje PS. Uporabniku naj bo olajšano delo vnašanja teksta, slik in urejanja besedila.
Akterji: Uporabnik spletne skupnosti z vlogo *Član* ali/in vlogo *Vodja*, sistem.
Predpogoji: Uporabnik je kliknil na element navigacijskega menija *My webpage*.
Rezultat: Prikazana privzeta zavihka *Home* in *About* ter njuna vsebina.
Osnovni potek:

1. Sistem prikaže predstavitveno stran člana z opcijami *View* in *Edit* ter vsemi zavihki strani, privzeto *Home* in *About*.
2. Uporabnik izbere opcijo *Edit* ter zavihek, katerega vsebino želi urejati.
3. Sistem prikaže dve tekstovni polji, eno za spreminjanje naslova zavihka, drugo pa za spreminjanje vsebine zavihka. Polje za spreminjanje vsebine je obogateno z urejevalnikom besedila in vmesnikom za nalaganje slik.
4. Uporabnik vnese tekst v naslovno polje in tekst v polje vsebine ter pritisne gumb za potrditev.
5. Sistem prikaže stran z vsebino zavihka, ki ga je član uredil.

Alternativni potek:

4. a) Uporabnik v polje vsebine vstavi sliko.
   b) Sistem shrani sliko v mapo, ki je deligirana posameznemu članu spletne skupnosti. Sledi korak 5.
   c) Če je slika prevelika, jo sistem avtomatično zmanjša na ustrezno velikost. Sledi korak 4 b.
   d) Če slika ni v enem od formatov .gif, .jpg, .png, sistem javi napako.

Dodatne zahteve:

Vsebina naj se ureja z bogatim urejevalnikom besedila, tako da bo član z lahkoto spremenil velikost, barvo in tip pisave, postavitev teksta in vključil smeškote. Uporabnik naj ima možnost enostavnega nalaganja in urejanja fotografij.

Uporabniški scenarij: »Urejaj, dodajaj, briši zavihke svoje PS«

Opis: Uporabnik lahko svoji predstavitveni strani ureja, dodaja in briše zavihke. Doda lahko le določeno število zavihkov. Izbriše lahko le toliko zavihkov, da bosta na strani vedno vsaj dva zavihka.
Akterji: Uporabnik spletne skupnosti z vlogo *Član* ali/in vlogo *Vodja*, sistem.
Predpogoji: Uporabnik je kliknil na element navigacijskega menija *My webpage* in na zavihek *Settings*.
Rezultat: Prikazani na novo ustvarjeni in urejeni zavihki na predstavitveni strani uporabnika.

Osnovni potek:

1. Sistem prikaže nastavitveno stran s seznamom naslovov trenutnih zavihkov, ki imajo poleg povezavi za urejanje in brisanje.
2. Če je trenutno število zavihkov na uporabnikovi PS manjše od dovoljenega števila zavihkov, sistem prikaže polje za vnos naslova novega zavihka.
3. Uporabnik vpiše naslov novega zavihka v polje.
4. Sistem prikaže polji za urejanje vsebine in naslova ravnokar ustvarjenega zavihka.

Alternativni potek:

2.a) Če je trenutno število zavihkov enako dovoljenemu številu zavihkov, sistem ne prikaže polja za vnos novega zavihka.

2. b) Če je trenutno število zavihkov večje od dovoljenega števila zavihkov, sistem ne prikaže polja za vnos novega zavihka, prikaže pa opozorilo, naj uporabnik izbriše odvečne zavihke.

3. a) Uporabnik klikne na povezavo za urejanje ob poljubnem naslovu obstoječega zavihka.

Sistem prikaže polji za urejanje vsebine in naslova zavihka.

3.b) Uporabnik klikne na povezavo za brisanje ob poljubnem naslovu obstoječega zavihka.

Sistem izbriše zavihek in prikaže nastavitveno stran z novim seznamom naslovov zavihkov in obvestilom o uspešnem brisanju zavihka.

Dodatne zahteve: Vsaka PS uporabnika mora vedno imeti vsaj dva zavihka.

Uporabniški scenarij: »Urejaj vsebino zavihkov PS drugega člana«

Opis: Uporabnik z vlogo *Vodja* ima pravico urejanja vsebine zavihkov kateregakoli člana. Spreminja lahko naslove zavihkov ter njihovo vsebino.
Akterji: Uporabnik spletne skupnosti z vlogo *Vodja*, sistem.
Predpogoji: Uporabnik je kliknil na primarno povezavo *Members*.
Rezultat: Prikazana vsebina urejenega zavihka.
Osnovni potek:

1. Sistem prikaže seznam vseh članov spletne skupnosti, s povezavami na njihove PS in kontakt.
2. Uporabnik klikne na povezavo PS poljubnega člana.
3. Sistem prikaže predstavitveno stran člana, z opcijami *View*, *Edit* in *Settings* ter vsemi zavihki strani.
4. Sledijo koraki od 2. koraka naprej uporabiškega scenarija »Urejaj vsebino zavihkov svoje PS«.

Uporabniški scenarij: »Urejaj, dodajaj, briši zavihke PS drugega člana«

Opis: Vodja lahko na predstavitveni strani kateregakoli člana ureja, dodaja in briše zavihke. Doda lahko le določeno število zavihkov. Izbriše lahko le toliko zavihkov, da bosta na strani vedno vsaj dva zavihka.
Akterji: Uporabnik spletne skupnosti z vlogo *Vodja*, sistem.

Predpogoji: Uporabnik je kliknil na primarno povezavo *Members*.
Rezultat: Prikazani na novo ustvarjeni in urejeni zavihki na predstavitveni strani izbranega člana.
Osnovni potek:

1. Sistem prikaže seznam vseh članov spletne skupnosti, s povezavami na njihove PS in kontakt.
2. Uporabnik klikne na povezavo do PS določenega člana.
3. Sistem prikaže predstavitveno stran člana, z opcijami *View*, *Edit* in *Settings* ter vsemi zavihki strani.
4. Uporabnik izbere opcijo *Settings*.
5. Sledijo koraki od prvega koraka uporabniškega scenarija »Urejaj, dodajaj, briši zavihke svoje PS« naprej.

Uporabniški scenarij: »Določi dovoljeno število zavihkov«

Opis: Administrator lahko spreminja število, ki določa maksimalno dovoljeno število zavihkov PS.
Akterji: Administrator spletne skupnosti, sistem.
Predpogoji: Administrator se nahaja na strani *Administer/Site configuration/My Page module settings*
Rezultat: Spremenjeno dovoljeno število zavihkov.
Osnovni potek:

1. Sistem prikaže polje za vnos števila dovoljenih zavihkov.
2. Administrator vnese število in potrdi vnos.
3. Sistem spremeni dovoljeno število zavihkov na vseh PS članov.

Alternativni potek:

2.a) Administrator pritisne gumb za ponastavitev vrednost.
Sistem ponastavi dovoljeno število zavihkov na 7 vsem PS članov.

3 Analiza obsotječih rešitev

Potrebno je najti rešitev, ki bo zadovoljila uporabniške zahteve spletne skupnosti. Sistem mora v osnovi podpirati integracijo profilov uporabnikov, foruma, blogov, anket, večjezičnost, oglaševanje, poleg tega pa mora omogočati tudi izmenjavo dokumentov med uporabniki. Ker gre za zaprto spletno skupnost moramo nadzorovati in upravljati pravice dostopa do različnih vsebin, različnim tipom uporabnikov s sistemom za upravljanje z uporabniki in sistemom za upravljanje z vsebino. Večino teh funkcionalnosti podpira množica spletnih sistemov za upravljanje z vsebino. Realizacija predstavitvenih strani podjetij s prilagodljivimi zavihki je uporabniška zahteva, specifična za poslovno spletno skupnost KosmAgora. Sicer obstajajo poslovne socialne mreže, ki ponujajo njenim članom uporabo zaprtih skupin znotraj njih in ustvarjanje predstavitvenih strani, a imajo določene pomanjkljivosti.

3.1 Spletne socialne mreže

V zadnjem desetletju so se močno razširile spletne storitve v obliki socialnih mrež, spletnih blogov, virtualnih skupnosti, zaprtih spletnih strani in forumov. Virtualna skupnost (angl. virtual community, online community) je skupnost, ki jo povezujejo interesi in ima možnost medsebojnega komuniciranja npr. v klepetalnicah in na forumih. [26] Virtualna skupnost ima lahko podobo informacijskega sistema, kamor lahko kdorkoli objavi vsebino (npr. sistem oglasnih desk) ali pa omogoča dostop le omejenemu številu oseb (npr. Weblog). Spletna skupnost (angl. web community) je virtualna skupnost realizirana s spletno stranjo. Lahko ima obliko storitev socialne mreže, internetnega foruma, skupino blogov ali pa drugih spletnih družbenih aplikacij. Socialne mreže so nadgradnja spletnih skupnosti. Spletno socialno mrežo (angl. online social network) lahko opišemo kot spletno storitev, ki omogoča posameznikom povezati se z drugimi uporabniki in pregledovati sezname povezav drugih uporabnikov v sistemu. Glede na sklope zadovoljevanja sociooloških potreb lahko zgradbo spletnih socialnih mrež razčlenimo na identiteto, prisotnost, odnose in razmerja, pogovor, skupnost, ugled in souporabo [27]. Skupnost znotraj socialne mreže predstavlja skupino, ki jo lahko ustvari posameznik in vanjo povabi svoje prijatelje. Skupina podpira komentiranje, nalaganje fotografij in enostaven sistem forumov za izmenjavo mnenj. Skupina je lahko zaprta, katere članstvo določa stvarnik skupine, lahko pa je odprta vsem članom socialne mreže.

Prve socialne mreže so bile namenjene zgolj navezovanju stikov med uporabniki, torej izvedbo profila in prijateljevanja z drugimi uporabniki (npr. SixDegrees.com, Friendster). Kmalu dobro zgrajen profil ni bil dovolj, uporabniki so dobili možnost dodajanja fotografij, avdio in video posnetkov z zunanjih strani (Myspace). Danes najbolj razširjena socialna mreža, Facebook, pa z vmesnikom API Facebook platforme razvijalcem omogoča vključitev podatkov na zunanje strani in namizne aplikacije ter s tem povečanje obsega socialne povezanosti. [11]

Kmalu so se pojavile spletne strani usmerjene v interesna področja, spletne skupnosti pa so nastale tudi iz spletnih strani, katerih prvotni namen je bil deljenje multimedijskih vsebin, kot sta Flickr in YouTube. Z njimi so zrasle tudi poslovne socialne mreže, kot so Ryze.com (prva socialna mreža za podjetnike), LinkedIn in Xing. Ravno zadnji dve najbolje zadostita zahtevam uporabnikov poslovne spletne skupnosti. Pri LinkedIn na primer, se podjetje lahko predstavi z zavihki *Carrers*, *Employes*, *Products and Services*. Xing pa ponuja predstavitveno stran podjetja z dvema statičnima zavihkoma: *About* in *Employees*. S tem predstavlja dokaj solidno rešitev zahteve po predstavitvenih straneh podjetij. Obe mreži podpirata skupine, ki vsebujejo forume (pri Xing) oz. diskusije (pri LinkedIn) in pregled članov.

Socialne mreže so v osnovi namenjene grajenju povezav med njenimi člani. V socialno mrežo se lahko včlani kdorkoli brez povabila in začne graditi povezave med ostalimi člani. Zaprto virtualno skupnost lahko realiziramo z zaprto skupino znotraj obstoječe socialne mreže, v katero se lahko nov uporabnik včlani le s povabilom. Pri poslovnih socialnih mrežah so člani lahko tudi podjetja. Xing in LinkedIn nudita vrsto funkcionalnosti prilagojeno podjetjem, med drugim tudi profil podjetja, predstavitveno stran podjetja, zaposlitvene ponudbe in promocije. Z njimi približno zadovoljimo večino uporabniških zahtev spletne skupnosti KosmAgora, razen zahtev po izmenjavi dokumentov, blogu in anketah. Ravno izmenjava dokumentov je pomembna funkcionalnost, ki je socialne mreže ne podpirajo in je bolj značilna za spletne sisteme za upravljanje z bazo znanja, kot je Wiki. Poleg tega je naročnik težil k samostojni rešitvi, bazirani na lastni domeni *www.kosmagora.com*. Poslovna spletna skupnost je namenjena podjetjem ki imajo sedež v Goteborgu. Ker je želja naročnika, da bi se skupnost razširila v prihodnosti tudi na ostala Švedska mesta, s katerimi bodo verjetno prišle tudi dodatne zahteve, je potreben sistem, ki bo podpiral veliko možnosti razširitev.

3.2 Wiki

Wiki je sistem spletnih strani, ki omogoča preprosto dodajanje in urejanje medsebojno povezanih spletnih strani. Vsebino Wiki strani lahko dodaja in ureja vsak obiskovalec s pomočjo označevalnega jezika (angl. markup language) ali WYSIWYG urejevalnika besedila. Wiki sistem je lahko odprt za vse obiskovalce strani, ali pa zaprt za določeno skupino uporabnikov, ki sodelujejo med seboj. Wiki je orodje za upravljanje z znanjem. Omogoča preprosto urejanje, organiziranje in iskanje vsebine in je tako odlično orodje za hranjenje dokumentacije. Osnovne funkcionalnosti vsakega Wiki sistema so urejanje vsebine, povezovanje vsebine, zgodovina, zadnje spremembe, peskovnik in iskalnik. [6] Poleg teh nekateri sistemi podpirajo tudi blog, forum in izmenjavo dokumentov. Danes lahko izbiramo med množico odprtokodnih Wiki sistemov kot so Media Wiki, TWiki, pmWiki in Dokuwiki. Vse več podjetij jih uporablja za hranjenje interne dokumentacije.

Rešitev z enim od odprtokodnih Wiki sistemov bi doprinesla učinkovito upravljanje z dokumenti, gradnjo baze znanja članov ter povečanje sodelovanja med člani z diskusijami. Glavna pomanjkljivost Wiki strani je izgubljanje avtorstva. Sistem temelji na ideji da vsakdo lahko ureja določeno stran. Pri zaprtih sistemih se seveda da omejiti, da določene strani

urejajo le določeni uporabnik. Vendar je samo upravljanje z uporabniki in njihovimi pravicami dostopa do različnih vsebin strani dosti bolj težavno in dopušča manj možnosti prilagoditev kot ostali CMS sistemi. Prav tako imamo manj možnosti pri samem upravljanju vsebine strani. Gradnja menijev in drugih gradnikov strani, ki jih ponujajo sistemi WCMS, ni mogoča. Wiki sistemi nimajo določenih funkcionalnosti socialnih mrež, kot so prilagojeni profili uporabnikov, kontaktiranje uporabnikov, klepet in oglaševanje. Spletna skupnost KosmAgora prav tako potrebuje sistem, ki bi omogočal ustvarjanje obrazcev, ankete in predstavitvene strani podjetij. Ravno možnost urejanja vsebine samo lastne spletne strani znotraj sistema strani je v nasprotju s poglavitno Wiki funkcionalnostjo neprestanega urejanja skupnih strani.

3.3 Sistemi za upravljanje spletnih vsebin

Sistem za upravljanje spletnih vsebin (angl. Web Content Management Systems), kratko WCMS in pogosto tudi CMS, je programska rešitev, implementirana kot spletna aplikacija, ki na uporabniku prijazen način omogoča ustvarjanje, upravljanje, urejanje, objavljanje in preiskovanje različnih spletnih vsebin. Uporabnikom nudi avtorska orodja, s pomočjo katerih lahko ustvarjajo in urejajo vsebino brez znanja o programskih in označevalnih jezikih. Vse delo s sistemom poteka preko spletnega vmesnika. Prezentacijski sloj spletne aplikacije je ločen od vsebine in je določen s t.i. predlogami (angl. template), kar omogoča enostavno spreminjanje zunanjega izgleda. [10] Na spletu se pojavlja veliko število odprtokodnih CMS sistemov, ki so za razliko od komercialnih seveda brezplačni in ne nudijo tehnične podpore, imajo pa zato široko skupnost uporabnikov in s tem zagotovljeno podporo. Pri izboru ustreznega odprtokodnega CMS sistema je potrebno preučiti kako razširjen je sistem, torej število njegovih uporabnikov. Odločamo se tudi glede na podporo njegove skupnosti, osnovnih zmogljivosti, vgrajenih funkcionalnosti, ki jih sistem ponuja, stopnjo prilagodljivosti in stopnjo razširljivosti. Dober CMS sistem poskrbi za redno nadgradnjo ter zagotavlja varnost in skalabilnost. Najpomembnejši kriterij pri izboru ustreznega CMS sistema za spletno skupnost je stopnja razširljivosti sistema. S čim več obstoječimi programskimi dodatki, kot so vtičniki in moduli, želimo iz običajne spletne strani zgraditi stran, ki bo omogočala glavne funkcionalnosti virtualne skupnosti.

Željene vgrajene funkcionalnosti CMS sistema za postavitev spletne skupnosti KosmAgora:

- upravljanje kontrole dostopa
- blog
- forum
- ankete
- oglaševanje
- skupine
- nalaganje slik in dokumentov
- večjezičnost

- profil
- enostavno urejanje vsebine (WYSIWYG)

CMS sistemi, ki nudijo omenjene funkcionalnosti in imajo tudi veliko število uporabnikov, so: Typo3, Drupal, Joomla, XOOPS in Plone.

Joomla[1] z nekaj več kot 6000 podaljški (angl. extension) nudi vse potrebne funkcionalnosti. Je ena najbolj priljubljenih CMS sistemov in zelo preprosta za uporabo. Na ta račun pa nudi manj možnosti za razširljivost kot Drupal.

XOOPS[2] je modularen sistem, ki zadosti osnovnim potrebam spletne skupnosti, vendar ne podpira skupin. Ponuja pa modul za urejanje osebne strani uporabnikov, vendar ima le ta statičen izgled, podoben profilu uporabnika. XOOPS ima malo število dodatnih modulov in tudi slabo podporo in dokumentacijo za razvoj novih modulov.

Typo3[3] ima večino funkcionalnosti v samem jedru sistema. Manjkata le anketa in skupine, ki pa ju dobimo s podaljški. Typo3 ima namreč zelo močno skupnost razvijalcev in veliko množico prostodostopnih podaljškov (angl. extensions).

Plone[4] je primeren za poslovne spletne strani in intranete. Nudi zelo dobro podporo skupinam in z dodatki nudi vse željene funkcionalnosti spletne skupnosti. Število dodatkov (angl. add-ons) je skoraj 1500. Plone je eden redkih popularnih CMS sistemov, ki je napisan v programskem jeziku Python. Pri ostalih prevladuje programski jezik PHP.

Vsi od zgoraj naštetih sistemov nudijo ustrezno rešitev uporabniškim zahtevam spletne skupnosti KosmAgora. Iskala sem sistem, ki nudi veliko možnosti razširitev in podpira čim več funkcionalnosti. Drupal že z osnovno namestitvijo zadovolji večino uporabniških zahtev, njegovih 7000 modulov pa ponuja širok spekter funkcionalnosti. Strmo krivuljo privajanja na sistem olašja dobra dokumentacija in podpora Drupal skupnosti v obliki forumov.

[1] http://www.joomla.org
[2] http://xoops.org
[3] http://typo3.org
[4] http://plone.org

4 Drupal

Drupal je odprtokodna platforma za upravljanje z vsebino na spletu. Je sistem z visoko stopnjo modularnosti, preprosto razširljiv, podpira množico standardov in teži k čisti kodi. Drupal sestoji iz osnovnih (angl. core) in dodatnih (angl. contrib, add-on) modulov. S tem omogoča prilagajanje različnim potrebam uporabnikov, poenostavi proces ustvarjanja, upravljanja in izdajanja vsebine na spletu. Gre za množico skript napisanih v jeziku PHP z množico programskih vmesnikov, ki omogočajo hiter in efektiven razvoj prilagojene spletne aplikacije. Uporabniški vmesnik loči upravljanje vsebine in predstavitev vsebine. [12]

Za Drupalom stoji močna spletna skupnost z množico poštnih seznamov, uporabniških skupin, neprofitnih organizacij, manjših podjetij in podjetnikov.
Drupal Association je neprofitna organizacija, ki nudi podporo in posodablja Drupalovo spletno stran [19], organizira Drupal konference in dogodke. Drupalova spletna skupnost uporabnikov, administratorjev, oblikovalcev in spletnih razvijalcev, ki neprestano nadgrajujejo programsko opremo, nudi močno podporo. Gre za odprtokodni projekt, ki je distribuiran pod licenco GPL. [12]

Orodje Drupal omogoča posamezniku, skupini ali pa podjetju, da preprosto objavi, upravlja in organizira spletno vsebino. Z njim je nastala množica osebnih in poslovnih spletnih strani, spletnih portalov, e-trgovin, spletnih časopisov, spletnih galerij, intranetov, socialnih mrež in mnogo več. [19]

Projekt KosmAgora se je začel razvijati v verziji Drupal 5.20 decembra 2009. Trenutno je najbolj razširjena in varna različica Drupal 6, razvija pa se tudi že Drupal 7. Ko se odločamo katero verzijo sistema bomo namestili, je najbolj pomembno, da pregledamo kateri moduli in funkcionalnosti so na voljo v tej različici ter kako velika je skupnost, ki uporablja to verzijo, saj to vpliva na podporo, testiranje in varnost sistema.

Opis tehničnih zahtev in namestitev sistema se nahaja v dodatku A. Do poslovne spletne skupnosti dostopamo na naslovu *http://www.kosmagora.com*. Spletni sistem je podprt s podatkovno bazo, poimenovano *kosmagora*.

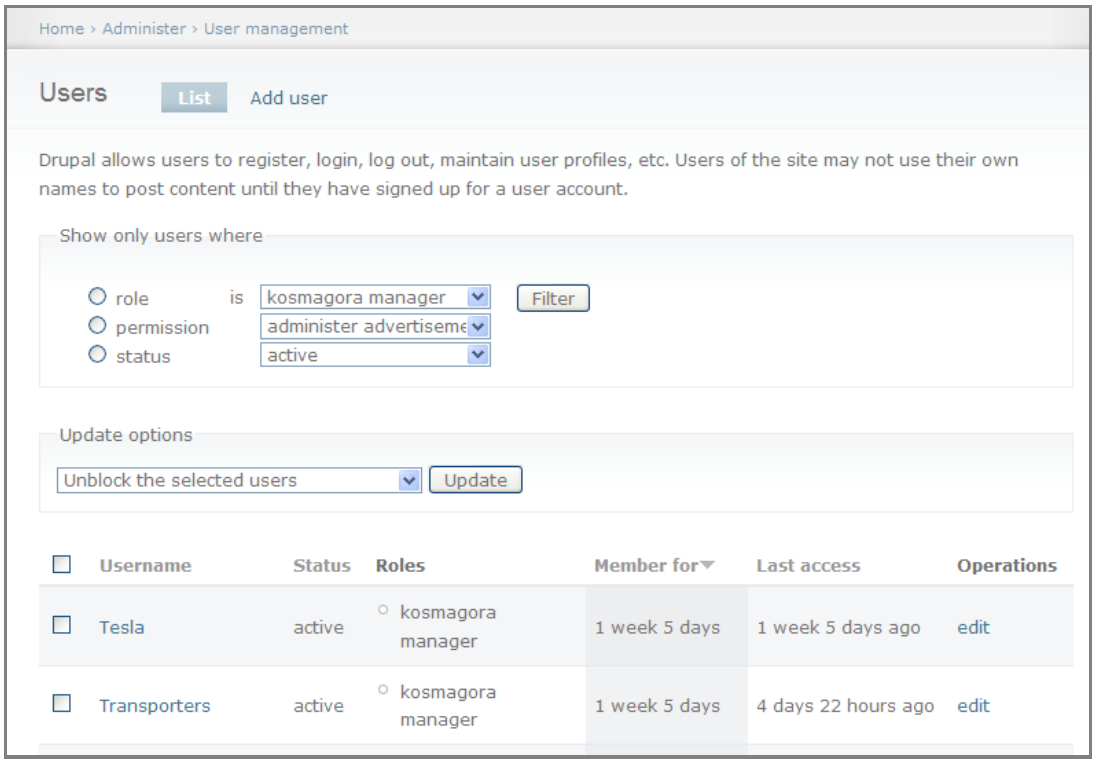
Sistem Drupal ima dva dela; obličje in zaledje. Čelni del sistema je del spletne strani, kjer je predstavljena vsebina, ki jo vidi obiskovalec strani. Zaledni del pa je namenjen nastavitvam, kontroli in namestitvam novosti strani. Zalednemu delu CMS sistema pravimo tudi administracijska plošča (angl. admin panel). [7] Najprej bom predstavila administracijski menu zalednega dela, ki nam ponuja vrsto nastavitev. Na kratko bom opisala osnovne funkcionalnosti zalednega sistema in na koncu predstavila rezultat naših nastavitev s slikami čelnega dela sistema (slike 8, 9, 10, 13).

Osnovni administracijski meni vsebuje naslednje razdelke:

- *Content management* (Upravljanje z vsebino)
- *Site building* (Gradniki sistema)
- *Site configuration* (Namestitvene strani)
- *User management* (Upravljanje z uporabniki)
- *Logs* (Izpisi)

4.1 Upravljanje z uporabniki

Upravljanje z uporabniki zajema poleg samega brisanja, dodajanja in urejanja uporabniških računov, še uporabniške pravice in vloge. Vsak uporabnik ima določeno uporabniško vlogo in vsaka uporabniška vloga ima določeno množico pravic dostopa. Drupal ima tri uporabniške vloge: *administrator, authenticated user* in *anonymous user*. Za poslovno spletno skupnost smo definirali novo uporabniško vlogo *kosmagora manager*, ki bo imela več pravic kot uporabniška vloga avtentificiranega uporabnika. S tem smo ustregli uporabniški zahtevi po štirih uporabniških vlogah. To storimo pod *User Management/Roles*. Posameznim uporabniškim vlogam dodeljujemo pravice s kontrolo dostopa, preko *User Management/Access control*. Vsaki vlogi dodelimo le toliko pravic, da lahko opravlja željene naloge in nič več. Ko dodajamo nove module, postane razpredelnica z nekaj sto polji precej nepregledna in hitro se zgodi da smo nekateri vlogi dodali kako pravico preveč. [9] V našem primeru moramo še posebej paziti na pravice anonimnih uporabnikov, ki morajo imeti le pravice pogleda nad vstopno stranjo. Zato smo namestili nekaj dodatnih modulov, kot sta *Contet Access* in *Path Access*, ki nam omogočata boljš kontrolo nad pravicami dostopa vsebine. Opis teh modulov se nahaja v razdelku 4.3.4. Ko imamo enkrat definirane vse uporabniške vloge z ustreznimi pravicami, lahko posameznemu uporabniku dodelimo več vlog. Slika 2 prikazuje del sistema za upravljanje z uporabniki, ki se nahaja v razdelku *User Management* administracijskega menija.



Slika 2. Upravljanje z uporabniki.

4.2 Upravljanje z vsebino

4.2.1 Vozli

Vsi tipi vsebine (angl. content types) izhajajo iz osnovnega tipa vsebine, vozla (angl. node). Tak način kreiranja tipov vsebine nudi veliko možnosti razširitve. Vsak vozel ima osnovno množico vedenjskih lastnosti, ki jih ostali tipi vsebine podedujejo. Vozli jedra so stran (angl. page), zgodba (angl. story), anketa (angl. poll), objava na forumu (angl. forum post) in blog. Vsak vozel, ki ga ustvarimo ima naslov, avtorja, datum stvaritve in možnost sledenja revizijam. Vsak vozel lahko napreduje na platnico (angl. front page), lahko je izdan (angl. published) ali neizdan in pa iskan (angl. searched). S pomočjo dodatnega modula CCK lahko ustvarimo nove tipe vsebine.

4.2.2 Kategorizacija vsebine

Modul *Taxonomy* služi za kategorizacijo (angl. category, taxonomy) vsebine. S termini lahko označimo posamezno vsebino, tako da jo bodo uporabniki lažje našli. Sistem kategorij vsebuje slovarje (angl. vocabularies) in termine (angl. terms). Slovarji označujejo skupino terminov in so nekaka nadpomenka. Termin je posamezna beseda v slovarju, ki se uporablja za kategorizacijio vsebine. Modul *View* uporablja kategorizacijo vsebine in s tem omogoča gradnjo novih pogledov vsebine glede na kategorije.

4.3 Gradniki sistema

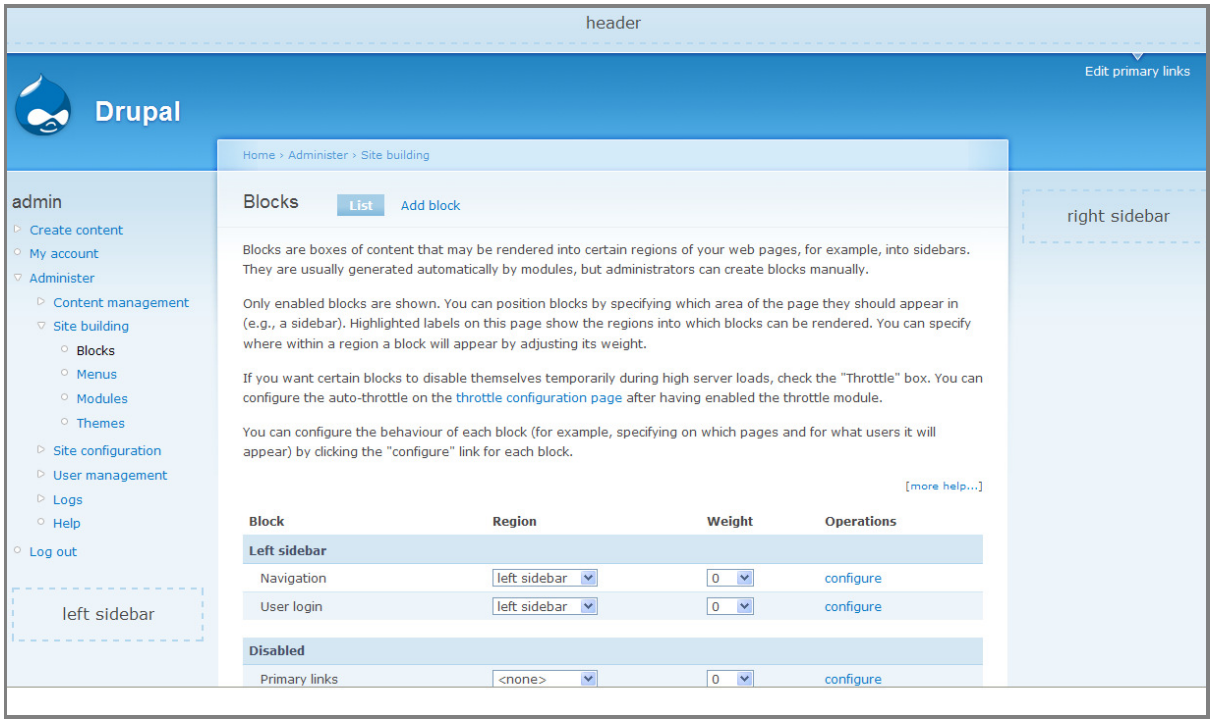
4.3.1 Teme

Drupal ima ločene elemente vsebine in predstavitve vsebine. Tako lahko posamezno vsebino predstavimo z različno vizualno zasnovo (angl. design) oz. temo. Z osnovno namestitvijo Drupala dobimo množico dosegljivih tem. Na Drupalovih spletnih straneh pa najdemo na stotine novih tem, ki jih lahko enostavno uporabimo. S konfiguracijo posamezne teme lahko spremenimo tematske barve, poleg tega pa tu določimo tudi kateri elementi bodo vidni na strani. Naložimo lahko novo sliko loga in ikone bližnjice (angl. shortcut icon, favicon), prikažemo ime strani in slogan strani.

Teme so sestavljene iz PHP, HTML in CSS skript. Drupalov gonilnik tem (angl. theme engine) nudi razvijalcem funkcionalnost, ki omogoča kreiranje lastnih unikatnih tem [9]. Ker naročnik ni imel specifičnih zahtev glede vizualne podobe in ker gre za resno, zaprto spletno skupnost, se samemu oblikovanju strani nisem posvetila.

4.3.2 Bloki

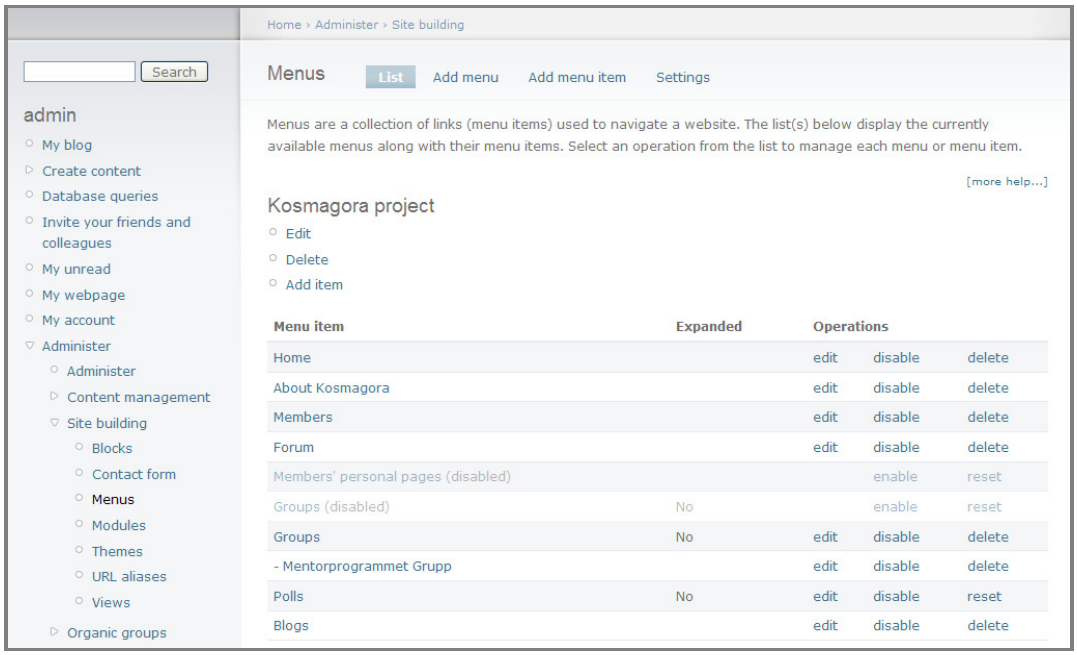
Blok je enota vsebine, ki jo lahko postavimo v pet različnih regij postavitve strani (angl. layout): levo in desno orodno vrstico, glavo, nogo in v vsebinski del (angl. content). Posamezen blok generira določen modul. Bloke lahko po potrebi onemogočimo ali pa jim določimo specifične nastavitve. Te so naslov bloka, spreminjanje vidnosti bloka (posameznim uporabnikom lahko dovolimo, da sami skrijejo ali prikažejo blok), določitev vidnost glede na uporabniške vloge in določitev vidnosti glede na URL naslov. Na administracijski plošči pod *Administer/Site building/Blocks* vidimo, da je navigacijski menu v bistvu blok, ki je prikazan v levi orodni vrstici (*left sidebar* na sliki 3).



Slika 3. Prikaz blokov sistema.

4.3.3 Menu

Na razdelku *Site building/Menus* administracijskega menija lahko hierarhično uredimo primarne povezave, sekundarne povezave in povezave navigacije. Navigacijski meni je glavni meni spletne strani in se dinamično spreminja glede na stran na kateri se nahajamo in glede na vlogo uporabnika. Na sliki 3 lahko vidimo v glavi primarno povezavo, *Edit primary links*. Namesto te bomo ustvarili povezave, ki bodo predstavljale glavne dele naše strani (slika 4).

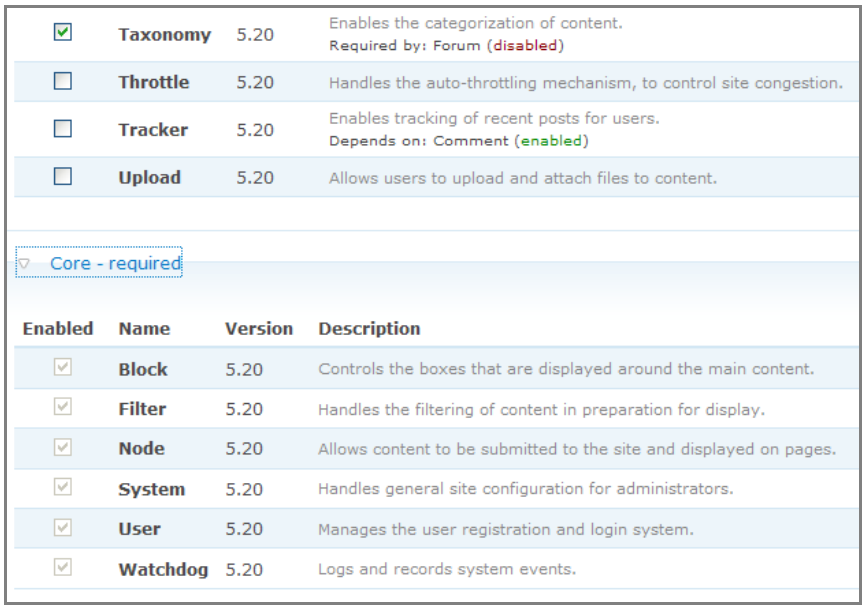


Slika 4: Sistem menijev.

Sekundarne povezave uporabimo za manj pomembne strani, kot so informacije o lastništvu. Ponavadi se le-te nahajajo v nogi spletne strani. Za poslovno spletno skupnost jih nismo uporabili.

4.3.4 Moduli

Drupal je visoko modularen sistem. Vsaka funkcionalnost je zapakirana v modul, ki ga lahko omogočimo in onemogočimo. Med moduli lahko obstajajo tudi odvisnosti, npr. modul *Forum* je odvisen od modula *Taxonomy* (slika 5). Obvezni moduli jedra (angl. core required modules) nudijo osnovno delovanje sistema za upravljanje z vsebino.

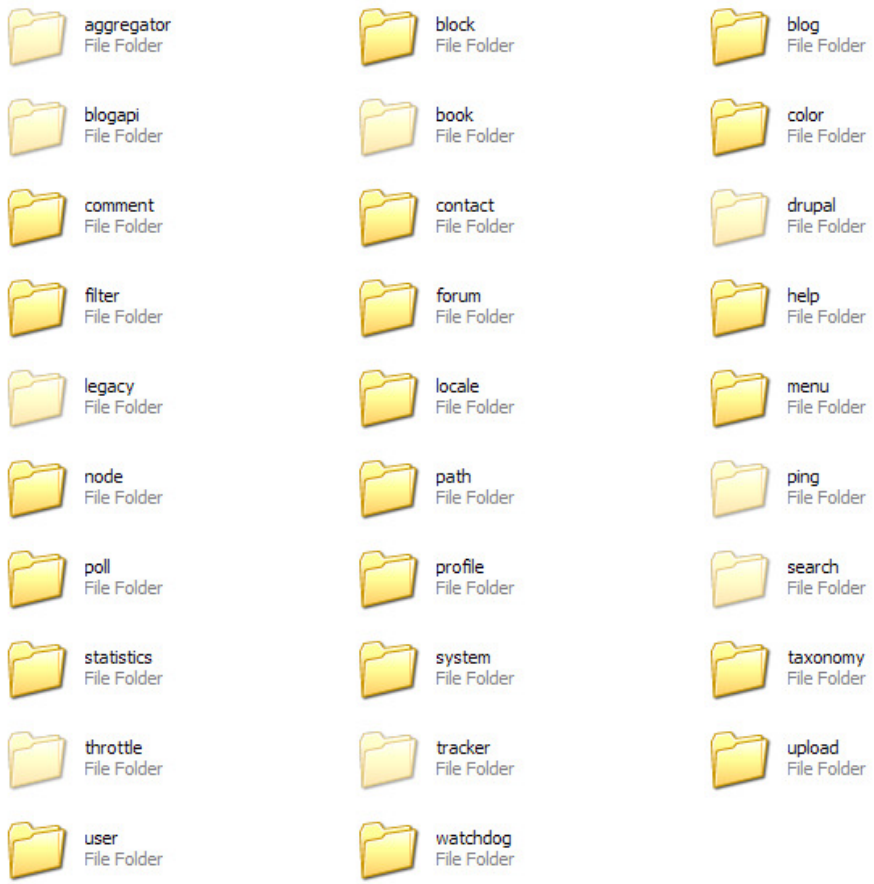


Slika 5. Del seznama modulov jedra (zgoraj opcijski in spodaj obvzeni).

Na spletni strani organizacije Drupal najdemo množico modulov, ki razvijalcem spletnih strani nudijo širok spekter funkcionalnosti in značilnosti (angl. feature), kot so npr. novi tipi vsebine, opozorila elektronske pošte, galerija slik in še mnogo več. Dodatni moduli (angl. contrib modules, add-on modules) se nenehno razvijajo. Preden se odločimo za namestitev dodatnega modula, ki smo ga našli na spletu, je pametno preveriti koliko uporabnikov ima, na kateri stopnji razvoja je in če nima morda preveč poročil o hroščih. Če imamo specifične zahteve, ki jim dodatni moduli ne ustrezajo, pa lahko ustvarimo tudi nov modul z želeno funkcionalnostjo. Več o razvoju novih modulov v razdelku 5.

4.3.5 Moduli jedra

Jedro označuje ograda (angl. framework) sistema Drupal in nudi osnovne funkcionalnosti. Ko prenesemo namestitveni paket Drupala v bistvu dobimo jedro, ki ne zadošča le za osnovno delovanje sistema, ampak nudi tudi najbolj priljubljene funkcionalnosti modernih spletnih strani in spletnih skupnosti. Moduli jedra se ločijo na obvezne in neobvezne. Na sliki 6 vidimo seznam obveznih modulov, ki poskrbijo za upravljanje z uporabniki in vsebino. Slika 7 prikazuje mape modulov jedra. Mape, ki so nevidne, ponazarjajo module, ki so del jedra, a jih nisem uporabila pri razvoju poslovne spletne skupnosti.



Slika 6: Moduli jedra Drupala.

Pri razvoju poslovne skupnosti sem uporabila naslednje neobvezne module.

- *Comment* (komentar)
- *Blog* (blog)
- *Contact* (kontaktni obrazec člana)
- *Poll* (anketa)
- *Profile* (prilagajanje profilov uporabnikov)
- *Upload* (prenos dokumentov)
- *Color* (spreminjanje barve tem)
- *Help* (strani za pomoč administratorjem)
- *Taxonomy* (kategorizacija)
- *Locale* (večjezičnost)

Ti moduli ne zahtevajo dodatne razlage. Opisala pa bom na hitro lastnosti modulov jedra *Path* in *Statistics*.

Path

Modul *Path* omogoča kreiranje URL vzdevkov (angl. alias) za poljubne strani. S tem modulom bomo lahko spletni naslov *www.kosmagora.com/?q=node/34* preimenovali v *www.kosmagora.com/?q=about* oz. z uporabo čistih URL-jev v *www.kosmagora.com/about*. Drupal normalno generira URL-je, ki na spletni strani *www.kosmagora.com* izgledajo nekako takole *www.kosmagora.com/?q=node/34*. Ta privzet način internetnega naslova je težko berljiv, poleg tega pa tudi onemogoča nekaterim iskalnikom indeksiranje strani. Uporabniška zgodba 3 vključuje zahtevo po preprostih in intuitivnih URL naslovih. Spletni strani lahko omogočimo čiste URL naslove (angl. clean URLs), ki ne bodo imeli več oznake »?q=«. Preden omogočimo čiste URL naslove preko Drupalovega vmesnika, moramo na strežniku namestiti, da jih bo ta omogočal. Potrebne so spremembe datotek *httpd.conf* ali *.htaccess*. Na administracijski plošči se lahko šele nato omogoči čiste URL naslove v razdelku S*ite configuration/Clean urls*. [15]

Statistics

Modul, ki beleži osnovno statistiko uporabe spletne strani. Z njim dobimo poročila o zadnjih naslovih, ki so bili prikazani (*Recent hits*), vidimo katere strani so bile največkrat prikazane (*Top pages*), kateri uporabniki nas največkrat obiščejo (*Top visitors*) in kdo preusmerja obiskovalce na našo stran (*Top referrers*). Modul žal ne omogoča pregleda nad IP naslovom obiskovalcev ali pa državo IP naslovov. Statistične podatke se hrani do štirih mesecev. Če bi želeli hraniti celotno zgodovino, pa moramo nastaviti *cron* funkcijo, ki bo podatke shranjevala na posebnem mestu na strežniku.

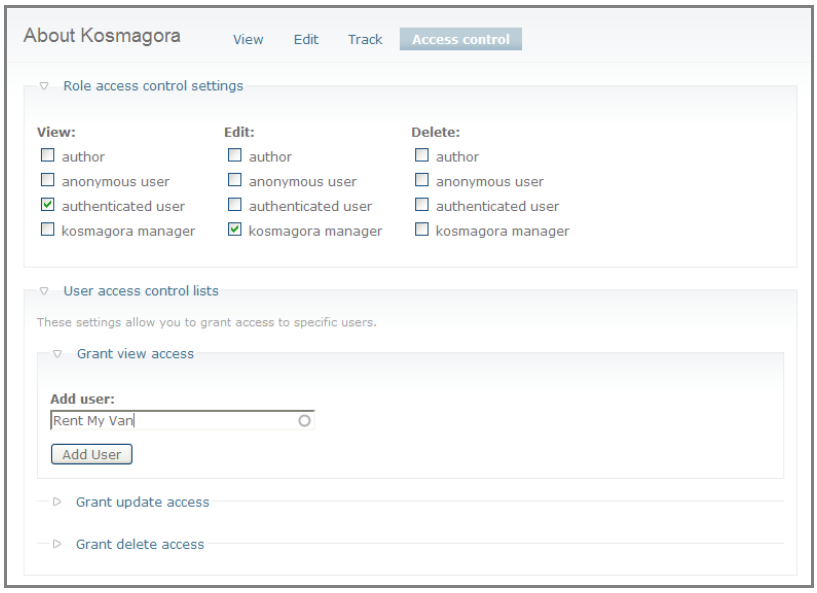
4.3.6 Dodatni moduli

Glede na uporabniške zahteve smo namestili štiriindvajset dodatnih modulov. Izmed množice skoraj 7000 Drupal modulov, je večino takih, ki niso kompatibilni z vsemi verzijami Drupala. Nekateri moduli so tudi v fazi razvoja in so zato nestabilni. Vsi dodatni moduli, ki sem jih namestila so stabilni in imajo veliko množico uporabnikov. Izbiro nameščenih modulov bom utemeljila z opisom rešitev uporabniških zahtev. Na slikah, ki ponazarjajo funkcionalnosti posameznih modulov poslovne spletne skupnosti je vsebina strani izmišljena. Prav tako je, zgolj za ponazoritev funkcionalnosti, uporabljena osnovna predloga vizualne zasnove strani.

Kontrola in način dostopa za različne tipe uporabnikov

Dodani moduli: *Content Access, Front Page, Path Access* (odvisen od *Role Weights*)

Ker imamo več različnih tipov vsebine v spletni skupnosti in več vlog uporabnikov, je bilo potrebno najti modul s katerim bi lahko določali pravice dostopa do posameznih vsebin in delov skupnosti. Modul *Content Access* doda vsakemu tipu vsebine možnosti nastavitve pravic načina dostopa za vsako uporabniško vlogo ter avtorja vsebine. Iste nastavitve se lahko omogočijo tudi za posamezne vozle vsebine. To je modul, ki je nujno potreben za vsako spletno skupnost in socialno mrežo. Za lažje upravljanje vsebine in dostopa vstopne strani smo namestili modul *Front Page*. Z njim se določi drugačna postavitev in tema za vstopno stran, kar daje uporabniku vtis zaprte spletne strani. Da ne bi prišlo do vdorov na stran, kar bi se lahko hitro zgodilo že samo z eno napačno nastavitvijo kontrole dostopa vsebin, pa se je potrebno prepričati, da zares na nobenem spletnem mestu, razen domače strani, nimajo dostopa anonimni obiskovalci strani. Za to poskrbi modul *Path Access*, s katerim se omeji dostop do določenih ali pa vseh URL naslovov z uporabniško vlogo anonimnež, razen tistih, ki jih posebej določimo, na primer *www.kosmagora.com/front*. *Path Access* je odvisen od modula *Role Weights*.



Slika 7. Kontrola dostopa z dodatnim modulom.

Včlanitev in prijava

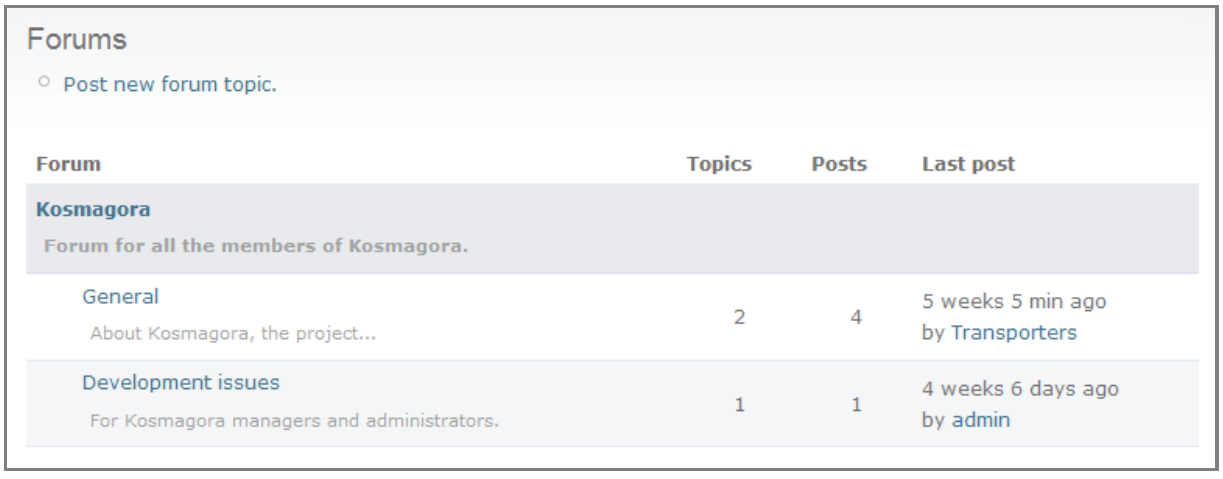
Dodani moduli: *Profile, CAPTCHA Image* (odvisen od *CAPTCHA*), *Legale*, (odvisen od *Checkbox Validate*), *Token, Invite, Invite Statistics*

Modul *Invite* nudi rešitev včlanitve s povabili in pripomore k hitri rasti skupnosti. Član lahko po želji uredi vsebino povabila, kjer je povezava na stran za včlanitev. *Invite Statistics* pa hrani informacije o številu povabljenih uporabnikov na posameznega člana in stanju povabil (*pending, expired, accepted*). Administrator ima še nekaj dodatnih nastavitev, kot je določitev maksimalnega dovoljenega števila povabil na člana. Stran za včlanitev je določena z opcijskim modulom jedra *Profile*. Za vključitev polja z obvezno potrditvijo smo namestili preprost modul *Legale*. Modul *CAPTCHA* pa preverja človečnost uporabnika. Odločili smo se za preverjanje z vnosom besedila s slike z modulom *CAPTCHA Image*.

Forum

Dodani moduli: *Forum, Forum Access* (odvisen od *ACL*)

Forum je sicer del Drupalovega jedra, ki pa ne prinese nastavitev pravic urejanja in ustvarjanja novih tem (angl. topic) le določenim uporabniškim vlogam. Za to poskrbita modula *Forum Access* in *ACL*. Z njima lahko določimo pravice branja, pisanja, urejanja, ustvarjanja odgovorov in tem posameznim uporabniškim vlogam. Poleg tega pa za posamezen forum določimo seznam oseb, ki dobijo pravice moderatorja (privzete pravice urejanja in brisanja tem in prispevkov).



Slika 8. Forum.

Oglaševanje

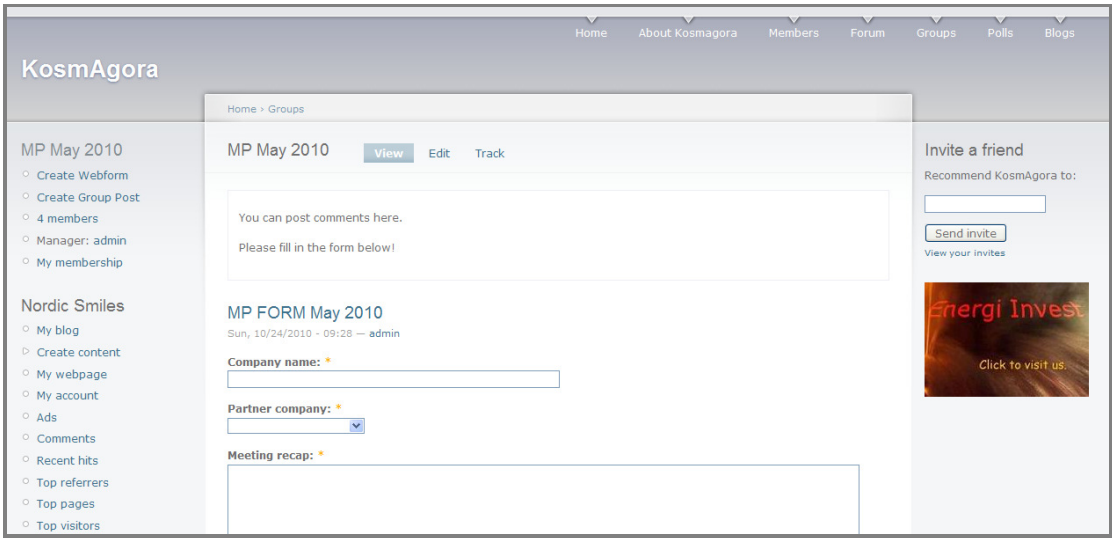
Dodani modul: *Ad, Image Ad*

Paket *Ad* vsebuje vrsto podmodulov, znotraj modula *Ad*, ki omogočajo raznorazne vrste prikaza oglasov, poročil o oglasih in predpomnjenja oglasov. Glede na uporabniške zahteve je bil dovolj podmodul *Image Ad*, s katerim lahko uporabnik z uporabniško vlogo *Vodja* (v nadaljevanju *Vodja*) naloži sliko, ki predstavlja oglas, ji določi spletno mesto, kamor bo oglas kazal in čas veljavnosti oglasa. Ker imajo z oglasi opravka le *Vodje* in administrator, nas malo bolj zapleten potek nastavitev oglasov ni motil. Z rastjo skupnosti pa bi bilo pametno razviti nov modul, ki bo nudil preprostejše nastavitve za člane, kot je naprimer nalaganje slike oglasa in čakanje na odobritev. To sicer ni del uporabniških zahtev našega projekta.

Skupine

Dodani moduli: *Organic Groups* (odvisen od *Views, Views RSS*), *Organic Groups Access Control, Webform*

Skupine so realizirane s pogledi. Gre torej za novo vrsto pogleda, ki omogoča pogled določene vsebine določenim članov. Z modulom *Organic Groups* imamo celoten nadzor nad skupino ter možnost upravljanja skupin in njihovih članov. Modul *Organic Groups Access Control* nudi dodatno varnost za zaprte skupine. *Vodje* lahko ustvarijo novo skupino v glavnem meniju *Create content/Group.* Prav tako lahko dodajajo in brišejo člane, ustvarjajo novo vsebino znotraj skupine v oblike komentarjev ali pa prispevkov (angl. coments, posts). Vodje lahko objavijo tudi prispevek z dokumentom, ki ga naložijo. Dovoljene tipe in velikost dokumentov določi administrator. Za vsak prispevek lahko avtor posebej določi pravice nad njim. Pri osnovni nastavitvi imajo člani le možnost pogleda, lahko pa se jim določi tudi možnost urejanja. Člani lahko sodelujejo v skupini zgolj z objavo komentarjev na prispevke. Izjemoma lahko urejajo določen prispevek in tudi naložijo novo datoteko, če je kreator prispevka dodelil ustrezne pravice članom.



Slika 9. Spletni obrazec v skupini.

Vodje lahko za posamezno skupino ustvarijo obrazec. To funkcionalnost nam da modul *Webform*. Ta poskrbi, da *Vodje* na preprost način ustvarijo polja, dovolijo določeno število oddaj in celo omogočijo avtomatično pošiljanje izpolnjenega obrazca določenemu uporabniku. Slika 9 prikazuje pogled vsebine skupine, med katero je tudi spletni obrazec. Polja, ki jih imamo na voljo ne zavzemajo vseh možnosti za potrebe spletne skupnosti KosmAgora, zato smo morali spremeniti kodo modula, kar ni najbolj varno. Več o programskih popravkih modulov v razdelku 4.3.7.

Meniji

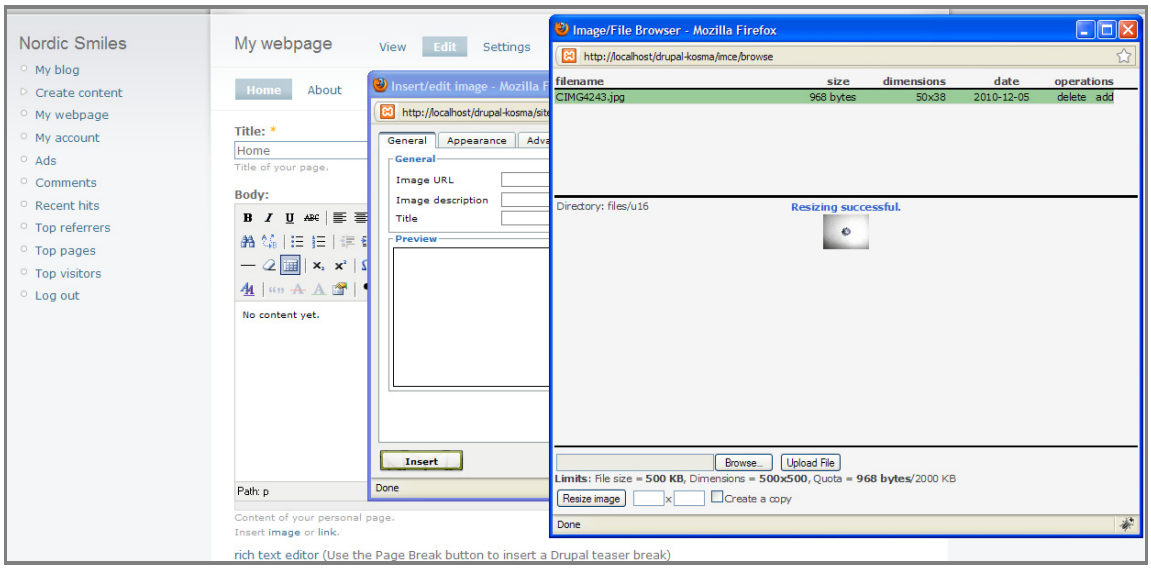
Dodan modul: *DHTML Menu*

S pomočjo JavaScript-a *DHTML Menu* dinamično prikaže vgnezdene menije s klikom na element menija in ni potrebno čakanje na osvežitev strani. Modul, ki enostavno pripomore k boljši uporabniški izkušnji.

Urejanje vsebine

Dodana modula: *IMCE, Tiny Tiny MCE*

Na spletu najdemo množico odprtokodnih urejevalnikov besedila. Najprej sem naletela na modul *WYSIWYG*, s skupino urejevalnikov besedila in kode. Rešitev ni pripeljala do željenih rezultatov, saj je bila integracija z moduli za upravljanje s slikami zelo težavna. Po vrsti napak sem našla rešitev, ki je zadovoljila uporabniške zahteve. *Tiny Tiny MCE* modul nam prinese urejevalnik besedila s širokim spektrom funkcij, ki jih lahko po potrebi odstranimo. *IMCE* modul je kompatibilen z njim in vsebuje uporabniški vmesnik za nalaganje slik. Slika 10 prikazuje delo z obema.



Slika 10: Urejevalnik vsebine in upravljanje s slikami.

4.3.7 Prilagoditve in programski popravki modula

Če želimo spremeniti del funkcionalnosti modula, ali pa omogočiti kako novo funkcijo, dodati nov tip in tega ne moremo storiti zgolj s kljukami, ampak je potrebna sprememba kode v obstoječem modulu, pravimo, da modul prilagodimo (angl. tweak) oz. programsko popravimo. Ena od uporabniških zahtev (del uporabniške zgodbe 9) je bila , da obrazcu dodamo spustni seznam (angl. drop down list) iz katerga lahko izberemo določene vrednosti (npr. vsa uporabniška imena). V prejšnjem poglavju smo spoznali modul *Webform*, ki pa ne ponuja te komponente. Na spletu najdemo množico programskih popravkov za raznorazne module. Tam sem našla tudi popravek *webform_2.7_extended_components.patch* za modul *Webform*, ki omogoča dodajanje lastnih, prilagojenih komponent.

Izvedbo programskih popravkov omogoča množica samostojnih programov, kot je UnixUtils[5] in vrsta razvojnih orodij (IDE-jev), kot je Eclipse[6]. Ko smo pognali programski popravek nad modulom webform, smo lahko dodali novo komponento *dynamic_select.inc*. To je nova komponenta, ki nam prinese spustni seznam, ki ga lahko napolnimo s podatki iz baze.

Rešitev s programskimi popravki ni najbolj varna, saj ti popravki niso dovolj dobro stestirani, tako kot so dodatni moduli, ki jih najdemo na Drupalov spletni strani. Ko izvajamo programske popravke, moramo razumeti katere spremembe bo popravek prinesel in se prepričati, da z njim ne bo ogrožena stabilnost sistema. [13]

[5] http://unxutils.sourceforge.net/
[6] http://www.eclipse.org

5 Razvoj modulov za Drupal

Za razvoj Drupalovega modula je sicer dovolj že malo boljši urejevalnik besedil, ki prepozna programske jezike. Kadar pa posegamo v kodo jedra sistema Drupal ali morda obsežnejšega obstoječega modula, pa imamo kar nekaj razvojnih orodij, ki podpirajo PHP, npr. Eclipse[7], Komodo IDE[8], Zend Studio[9], NetBeans[10] s Xdebug[11]. Pri dodatnih modulih, ki so še v fazi razvoja je dostikrat pametno pognati kodo z enim od zgoraj naštetih orodij, saj tako lahko odkrijemo potencialno napako.

Za izgradnjo modula je potrebno osnovno znanje programskega jezika PHP ter znanje upravljanja s podatkovnimi bazami. Pri pisanju kode se moramo držati določenih standardov kodiranja, ki so osnovani na kodirnem standardu jezika PHP. [16] Standard PHP oznak priporoča opustitev znaka *?>* na koncu datoteke. Če se tega standarda razvijalec ne drži, se hitro pojavijo napake.

5.1 Zakaj smo se odločili za izgradnjo novega modula

Za realizacijo zahtev uporabniškega primera »Predstavitvene strani podjetja« smo iskali modul, ki bi omogočal uporabnikom gradnjo vsebine z bolj zapleteno strukturo kot je le navadna objava. Po raziskavi sem našla dva modula, ki omogočata strukturirano objavo vsebine. Modul *MySite* sicer omogoča uporabnikom prikaz vsebine pod različnimi zavihki, ampak ima eno omejitev. Vsebina, ki je na ta način prikazana, je povzeta iz že ustvarjenih vsebin. Pod istim zavihkom lahko na primer prikažemo vse naše objave v blogu in forumih, ne moremo pa ustvariti nove vsebine in jo objaviti pod poljubnim zavihkom. Ob pregledu razvijajočih se modulov sem zasledila tudi modul *Monster Menus*, ki omogoča več možnosti urejanja in postavitve vsebine s strani uporabnika, ampak žal ni prosto dosegljiv.

5.2 Datotečna organizacija

Znotraj mape sistema Drupal se nahaja naslednja struktura:

- Mapa *includes* vsebuje knjižnice pogostih Drupal funkcij.
- Mapa *misc* vsebuje JavaScript skripte in poljubne ikone ter slike.
- Mapa *modules* vsebuje module jedra. Ko dodajamo nove module, jih ne vstavimo v to mapo, ampak na ustrezno mesto znotraj mape *sites*.

[7] http://www.eclipse.org/
[8] http://www.activestate.com/komodo-ide
[9] http://www.zend.com/products/studio/
[10] http://netbeans.org/
[11] http://www.xdebug.org/

- Mapa *profiles* vsebuje različne profile za namestitev spletne strani. Namestitveni moduli nam omogočajo avtomatično namestitev določenih modulov jedra in dodatnih modulov.
- Mapa *scripts* vsebuje skripte za preverjanje sintakse, za zagon iz ukazne vrstice in za upravljanje s funkcijo *cron*.
- Mapa *sites* vsebuje vse dodatne nastavitve, module (*sites/all/modules*) in teme, ki jih uporabnik namesti tekom razvoja strani. Mapa *default* znotraj te mape vsebuje konfiguracijsko datoteko *settings.php*. Ob uporabi določenih modulov potrebujemo tudi mapo *files*, ki bo hranila slike logov, avatarjev in podobno. To mapo kreiramo znotraj *sites/default/files*.
- Mapa *themes* vsebuje predloge in osnovne teme. Vse dodatne teme, ki smo jih naknadno prenesli hranimo v mapi *sites/all/themes*. [12]

5.2.1 Datotečna struktura modula

Vsak modul mora vsebovati vsaj dve datoteki *.info* z informacijami in glavno datoteko *.modul*. Če naš modul zahteva spremembe podatkovne baze, le-te shranimo v datoteko *.instal*. Datoteka *.info* je informacijska datoteka, ki jo potrebuje vsak modul. Spodaj je primer vsebine *mypage.info* datoteke.

```
; $Id$
name = My personal web page
description = "This module enables users to create their personal web
pages."
core = 5.x
```

Prva vrstica je namenjena možnim informacijam o CVS verzioniranju. Spremenljivka `name` nosi uporabniku berljivo ime modula. Spremenljivka `description` nosi opis modula, ki bo viden administratorju na Drupal strani. Spremenljivka `core` pa predstavlja minimalno zahtevano verzijo Drupala, pod katero bo modul deloval. [3]

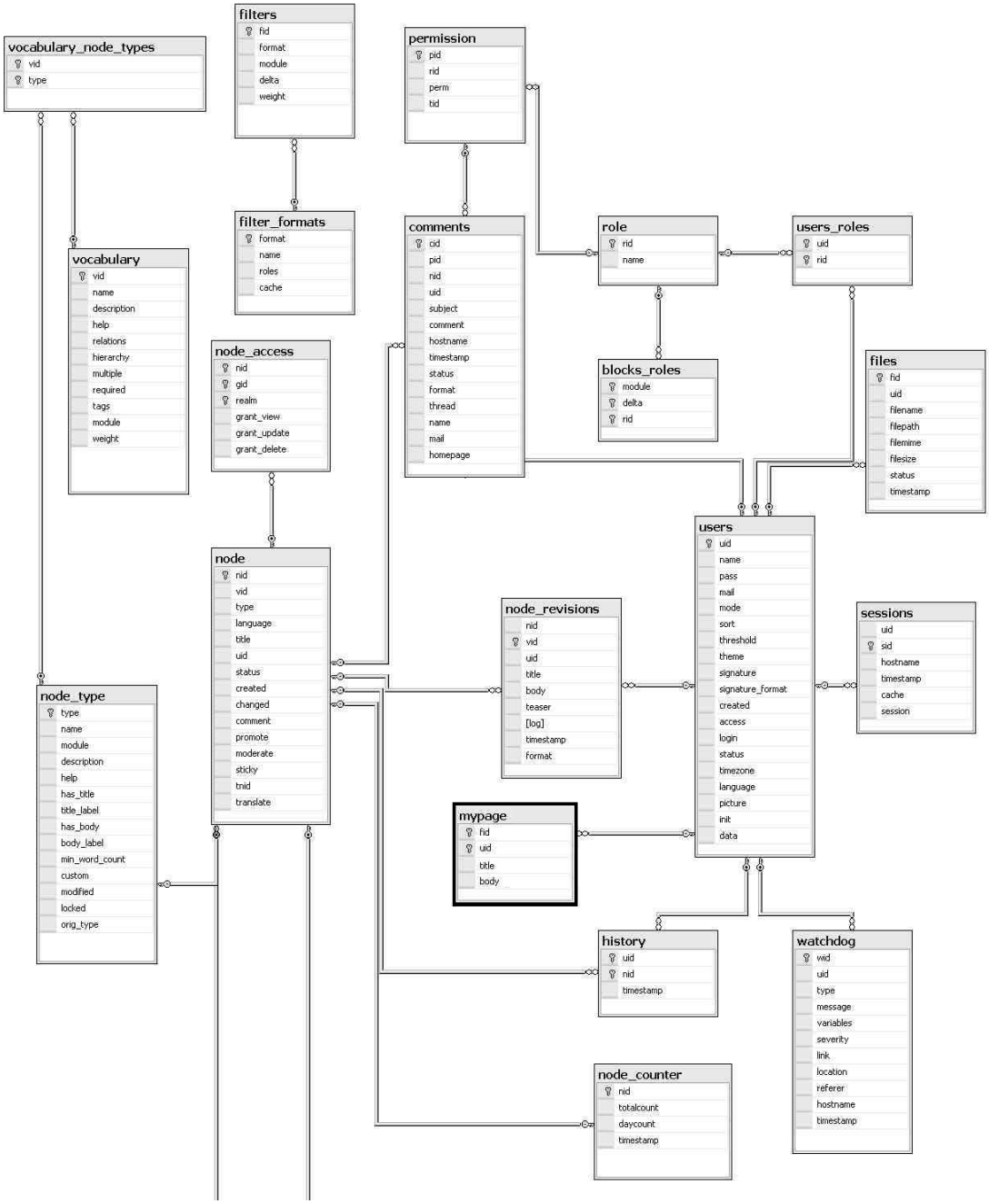
Datoteka *.install* vsebuje namestitveno skripto dodanih oz. spremenjenih tabel podatkovne baze Drupala.

Datoteka *.module* vsebuje glavno kodo modula. Po želji lahko kodo razmestimo na več *.inc* datotek, ki pa bodo prebrane ob ustreznem klicu v *.module* datoteki. [17]

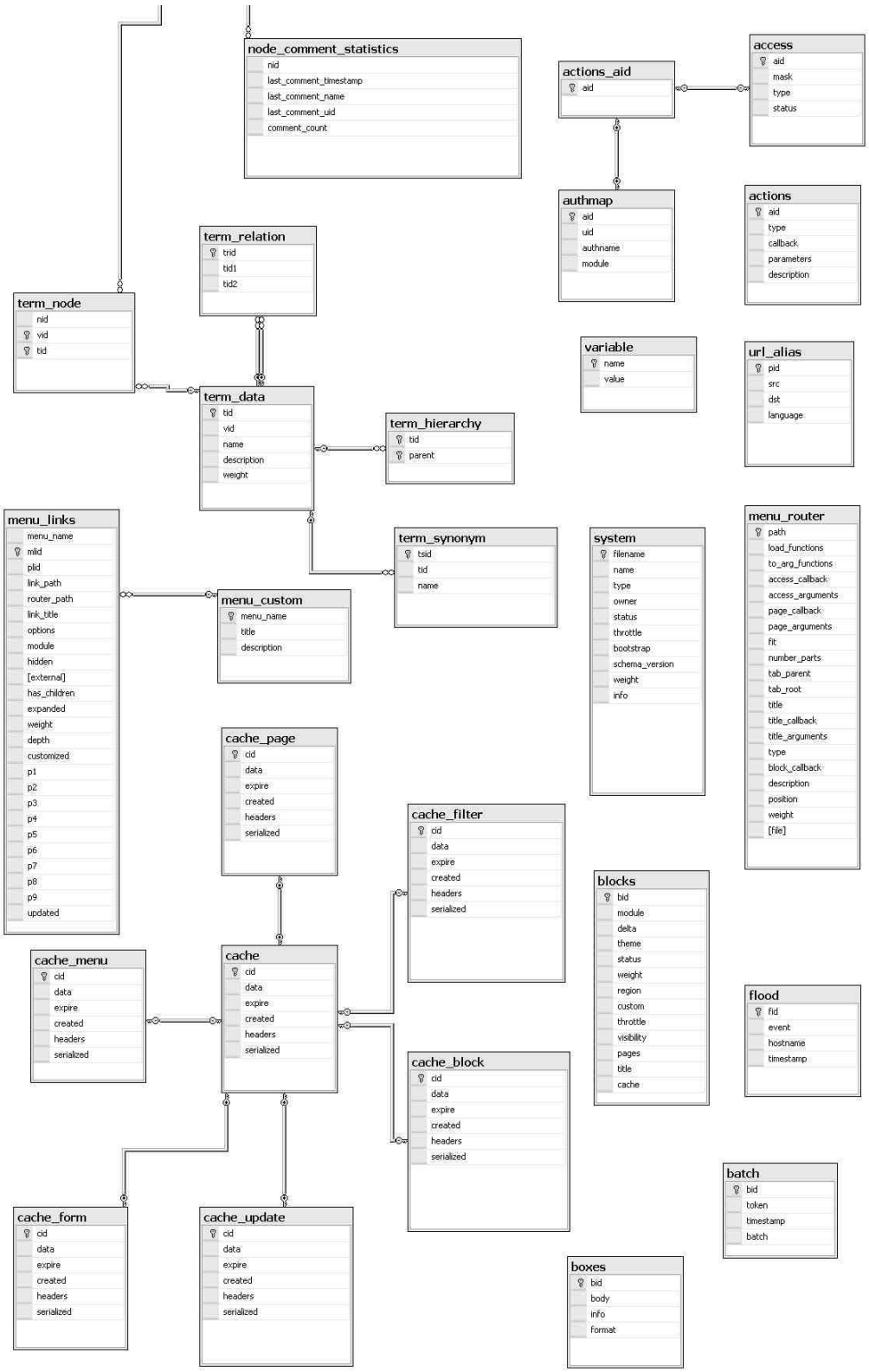
5.3 Delo s podatkovno bazo

Drupal podpira MySQL in PostgreSQL podatkovni bazi. Poslovna spletna skupnost KosmAgora je bila postavljena na MySQL bazi. Na slikah 11 in 12 je prikazan podatkovni model jedra Drupala z dodano tabelo *mypage*, ki je del modula *Mypage* . Kot je vidno na modelu sta nekako glavni tabeli *node* in *users*. Ker gre za zaprt sistem za upravljanje s spletno vsebino, je logično, da sta najbolj uporabljeni tabeli, ki vsebujeta podatke o vsebini in

uporabnikih. Z gradnjo spletne skupnosti smo prišli na velikost podatkovne baze sto treh tabel. Večina modulov vključuje tudi dodatne tabele, ki se ustvarijo znotraj podatkovne baze. Drupalova baza ni omejena z referencialno integriteto, saj tako omogoča več svobode dodanim modulom. Vendar trenutno razvijalci Drupala 7 obljubljajo uvedbo referencialnih integritet v jedru sistema.



Slika 11: Model podatkovne baze jedra (1. del)



Slika 12: Model podatkovne baze jedra (2.del).

Ob stvaritvi novega modula v namestitveni datoteki definiramo spremembe v podatkovni bazi. Spodaj je koda datoteke *mypage.install*, ki podatkovni bazi doda tabelo *mypage*.

```
<?php
/**
* Implementation of hook_install().
      */
      function mypage_install() {
            db_query("CREATE TABLE if not exists {mypage} (
                  uid int unsigned NOT NULL default '0',
                  fid int unsigned NOT NULL default '0',
                  title varchar(127) NOT NULL default '',
                  body longtext,
                  PRIMARY KEY  (uid,fid)
                  ) /*!40100 DEFAULT CHARACTER SET utf8 */;");
            }
      /**
      * Implementation of hook_uninstall().
      */
      function mypage_uninstall() {
            db_query('DROP TABLE {mypage}');
      }
```

Namestitvena datoteka vsebuje kodo za kreiranje in brisanje tabele. Za modul *Mypage* potrebujemo tabelo, ki bo shranjevala vsebino (naslov in glavni del) posameznega uporabnika pri posameznem zavihku. Tabela vsebuje polja *uid* (identifikacijska številka uporabnika), *fid* (identifikacijska številka zavihka), *title* (naslov) in *body* (telo oz. glavni del vsebine.). Polje *uid* ustreza polju *uid* v Drupalovi tabeli *users* in je del primarnega ključa.

Drupal dostopa do podatkovne baze preko plasti abstrakcije podatkovne baze (angl. database abstraction layer). Vmesnik abstrakcije podatkovne baze tako ponuja funkcije kot so `db_query()`, ki nadomesti `mysql_query()` in `pg_query()`. Tako lahko z isto funkcijo dostopamo do katerekoli podatkovne baze. Poleg tega, pa na ta način tudi onemogočimo napade v podatkovno bazo. Če uporabljamo katero drugo podatkovno bazo, pa lahko preprosto napišemo podporo zanjo. [12]

Celoten seznam funkcij najdemo na Drupalovih spletnih straneh [18]. Nekaj primerov funkcij programskega vmesnika podatkovne baze (angl. database abstraction API):

- `$db_query($query)` vrne rezultat poizvedbe, ki je podana kot parameter funkcije.
- `$db_fetch_object($result)` vrne objekt, ki predstavlja vrstico rezultata poizvedbe.
- `$db_fetch_array($result)` vrne seznam, ki predstavlja vrstico rezultata poizvedbe.
- `$db_lock_table ($table)` zaklene tabelo, podano s parametrom. Uporabi se *mysql* ukaz `LOCK TABLES t WRITE`.
- `$db_unlock_tables()` odklene vse tabele. Uporabi se *mysql* ukaz `UNLOCK TABLES`.

Funkcije Drupalovega vmesnika za ravnanje s podatkovno bazo so definirane v datoteki *drupal\includes\database.mysql.inc*. Spodnji del kode modula *Mypage* ponazarja uporabo funkcij vmesnika podatkovne baze.

```
$sql = "SELECT title,body FROM {mypage} m INNER JOIN {users} u WHERE
u.uid = m.uid AND u.name = '%s' AND fid = %d";
$result = db_query($sql, $uname, $page);
$obj = db_fetch_object($result);
```

MySQL ponuja od verzije 5.1. naprej dva najpogostejša tipa tabel (angl. table types, tudi storage engine), MyISAM in InnoDB. Drupal uporablja MyISAM, ki je privzeti tip za sisteme relacijske podatkovne baze MySQL verzije 5.1 in nižje. MyISAM uporablja zaklepanje na nivoju tabel. Ko je dodana vrstica v tabelo, ali pa se ta ureja, se tabela zaklene in je ni mogoče brati. To predstavlja problem, če bi naša spletna aplikacija izvajala več urejanj in dodajanj kot pa branj. V našem primeru gre večinoma za branja iz podatkovne baze. S povečano aktivnostjo uporabnikov, pa lahko postane število pisanj in urejanj v podatkovno bazo nezanemarljivo. MyISAM-ova največja pomanjkljivost je, da ne zagotavlja omejitev referencialne integritete podatkov. To pomanjkljivost odpravi tip tabel InnoDB, ki ga uporablja že Drupal 6 verzija. InnoDB zagotavlja omejitve referencialne integritete in omogoča višjo stopnjo sočasnosti, saj opravlja zaklepanje na nivoju vrstic tabel. Pri MyISAM lahko v primeru nesreče pride do nekonsistentnosti podatkov in je potrebna obnovitev z ukazom `REPAIR TABLE`. InnoDB zagotavlja sočasnost izvajanja transakcij in preprosto obnavljanje podatkovne baze po nesreči. InnoDB ima ACID lastnosti, to pomeni, da zagotvalja atomarnost, konsistentnost, izolacijo in trajnost podatkov.

Če zaklepanje tabel vpliva na zmogljivost spletne skupnosti, preverimo z analizo spremenljivk *Table_locks_immediate* in *Table_locks_waited*, ki jih dobimo z ukazom `SHOW STATUS LIKE 'Table'`. Če je vrednost spremenljivke *Table_lock_waited* prevelika, lahko optimiziramo spletno stran s pretvorbo določenih tabel v InnoDB. To lahko storimo z ukazom `ALTER TABLE accesslog TYPE = 'InnoDB'`. Pretvorba je priporočena le za določene tabele. Pretvorba vseh tabel lahko pripelje do neskladja. [12]

5.4 Kljuke

Sistem Drupal z moduli doseže inverzijo kontrole. Različne funkcionalnosti modulov so klicane ob primernem času. Klic teh funkcionalnosti je realiziran s kljukami. Kljuke so notranji dogodki sistema, konvencionalno določeni z imenom funkcij. PHP funkcija z imenom *foo_bar*, kjer je *foo* ime modula, *bar* pa ime kljuke, je kljuka. Rečemo jim tudi funkcije s povratnim klicem (angl. callback functions, oz. callbacks). Vsaka kljuka ima določeno množico parametrov ter določen tip vrnjenega rezultata. Ko kljuko priključimo Drupalovemu dogodku (dogodek je npr. posodobitev uporabniškega profila, logiranje ali pa izbris vozla), se bo ob klicu tega dogodka izvršila koda naše kljuke. Kljuke tako omogočajo modulom, da se priključijo osnovnim Drupalovim funkcijam. [12, 17] Celoten seznam kljuk je tudi objavljen na spletu [23].

5.4.1 Sistem *menu*

Drupalov sistem za menije vključuje navigacijski sistem za uporabnike in sistem funkcij povratnega klica, ki ga Drupal uporabi za odgovor na URL zahteve spletnega brskalnika. Za kreiranje novega modula je potrebno dobro razumevanje sistema *menu*. Preprosta hierarhija je definarana s potmi (angl. paths). Implementacija kljuke *hook_menu()* definira razdelke menija (angl. menu item) in jim določi pot. Sistem *menu* potem agregira te razdelke in določi hierarhijo menija glede na poti.

Primeri poti: *a*, *a/b*, *e*, *a/b/c/d*, *f/g,* in *a/b/h* in njihova hierarhija menija:

- *a*
    - *a/b*
        - *a/b/c/d*
        - *a/b/h*
- *e*
- *f/g*

Število elementov, ki sestavlja pot, še ne določa globino razdelka menija v drevesu. Ko pride zahteva strani, sistem *menu* pregleda, če je zahtevana pot registrirana kot razdelek menija s povratnim klicem. Če ni, potem sistem pregleda drevesno strukturo menija in vrne najbližji razdelek s povratnim klicem. Za pot *a/b/i* bi sistem vrnil povratni klic za *a/b*.

Sistem *menu* tudi nadzira dostop do funkcij s povratnim klicem. Razdelek menija lahko nosi atribut *access*. Če je ta `TRUE`, je dostop dovoljen, sicer pa ne. Če razdelek nima atributa *access*, se dostop dovoli glede na stanje *access* atributa njegovega prednika.

Na osnovni Drupalovi strani lahko opazimo, da je precej povezav prikazanih v obliki zavihkov. Ti se v sistemu *menu* imenujejo lokalne naloge (angl. local tasks). Nekako je dogovor, da so imena teh lokalnih nalog kratki glagoli. Za vsako množico lokalnih nalog mora obstajati tudi privzeta lokalna naloga (angl. default local task). Starš razdelka menija z lokalno nalogo bo vrnil privzeto lokalno nalogo. Privzeta lokalna naloga kaže torej na pot starša razdelka. Ko je izbrana lokalna naloga razdelka, se bo dejansko prikazala privzeta lokalna naloga otroka razdelka. Sama pot privzete naloge je uporabljena zgolj za primerno umestitev v hierarhiji menijev. Sistem lokalnih nalog omogoča uporabniku izkušnjo dela z običajnimi zavihki. [25]

5.4.2 Kljuke modula *Mypage*

Modul *Mypage*, ki je bil razvit glede na uporabniški scenarij »Predstavitvene strani podjetij« vsebuje množico kljuk. Kljuke so opisane tako, kot so bile realizirane v kodi modula. Ime kljuke s pripono *hook_* v opisu predstavlja lastnosti splošne Drupalove kljuke. Ime kljuke s pripono *mypage_* pa se nanaša specifično na kljuke modula *Mypage*.

mypage_help()

Funkcija, ki nudi pomoč uporabnikom. Z implemetacijo kljuke *hook_help()* postane dokumentacija modula dosegljiva gonilniku (angl. engine) Drupala in ostalim modulom. Parameter `$section` posreduje kljuki URL naslov, kjer je bila zahtevana pomoč. Kljuka *mypage_help* se bo odzvala na sekcijo `admin/help#mypage`, kjer bo vsebina pomoči prikazana na *admin/help* strani, deskriptor # pa bo poskrbel za povezavo na pomoč modula *mypage*.

hook_perm()

Kljuka *hook_perm()* definira pravice uporabnikov. Ta kljuka v resnici definira le imena pravic, ki jih bomo potem lahko izbrali na strani pravic uporabnikov. Te iste pravice potem uporabimo pri omejitvi dostopa določenih akcij *(angl. actions)* modula. Pravice so nato preverjene s funkcijo *user_access()*. *Mypage* modul ima tri pravice uporabnikov: *view mypage, edit mypage* in *manage all mypages*. Uporabniki, ki imajo pravico *view*, lahko le pregledujejo predstavitvene strani ostalih uporabnikov. Pravica *edit mypage* omogoča uporabniku da ureja lastno predstavitveno stran ter pregleduje predstavitvene strani ostalih uporabnikov. Uporabniki z *manage all mypages* pravico pa lahko urejajo predstavitvene spletne strani vseh uporabnikov.

hook_menu()

Ta funkcija definira elemente menija in povratne klice strani. Modul s to kljuko določi poti, na podlagi katerih se potem sistem odloči katere zahteve bodo izvedene. Glede na tip določitve poti se kreira povezava v navigacijskem bloku in/ali pa se kreira nov element v meniju strani administracije (*q=admin/menu*).

Funkcija *hook_menu()* vrne seznam elementov menija. Vsak element je asociativen seznam, ki lahko vsebuje naslednje vrednosti:

- `path`, obvezna vrednost. Pot, kjer je uporabnik izbral element menija.
- `title`, obvezna vrednost. Naslov elementa menija.
- `callback`, funkcija, ki se bo poklicala za prikaz strani, ko uporabnik klikne na povezavo. Če povratna funkcija ni določena, se bo izvedla povratna funkcija starša elementa menija.
- `callback arguments`, seznam argumentov funkcije povratnega klica.
- `access`, logična vrednost, ki dovoli oz. nedovoli dostop do elementa menija uporabniku. Če je ta vrednost izpuščena, bo določen dostop glede na pravice dostopa starša elementa menija.
- `weight`, celo število, ki relativno določa uvrstitev elementa v meniju. Privzeta vrednost je 0, višje vrednosti se umestijo nižje v meniju. Kadar `weight` ni določen se elemente razvrsti po abecednem redu.

- `type`, bitna maska zastavic, ki opisuje značilnosti elementa menija. Vsaki bitni maski je določena tudi konstanta s pomenljivim imenom v datoteki *menu.inc*. Poznamo naslednje tipe elementov:

    - `MENU_NORMAL_ITEM` (privzet tip)
      Običajni elementi menija se prikažejo v meniju, lahko pa jih administrator tudi skrije in premakne
    - `MENU_ITEM_GROUPING`
      Skupine elementov se uporabljajo za strani kot so node/add in nudijo skupino podstrani, ki jih lahko uporabnik obišče.
    - `MENU_CALLBACK`
      Povratnemu klicu določi pot, tako da se ob dostopu URL mesta sproži ustrezna funkcija.
    - `MENU_DYNAMIC_ITEM`
      Dinamični elementi se hitro spreminjajo in zato tudi naj ne bi bili shranjeni v podatkovni bazi za administracijo.
    - `MENU_SUGGESTED_ITEM`
      Moduli lahko priporočijo elemente, da jih administratorji potem omogočijo.
    - `MENU_LOCAL_TASK`
      Lokalne naloge so prikazane kot zavihki.
    - `MENU_DEFAULT_LOCAL_TASK`
      Vsaka množica lokalnih nalog mora imeti tudi privzeto lokalno nalogo, ki kaže na isto stran kot njen starš. [24]

Tabela 1 ponazarja strukturo menijev modula *Mypage*. Pove na katerih URL naslovih bo določena vsebina, s katero funkcijo kličemo to vsebino ter pravice uporabnikov, katerim bo omogočen dostop do posamezne strani.

V kljuki *mypage_menu()* najprej kličemo funkcijo `mypage_get_page_num($uname)`, ki preveri, če ima izbran uporabnik že ustvarjeni privzeti dve predstavitveni strani (podzavihka *Home* in *About*). Če jih nima, se kreirata nova dva zapisa v tabeli *mypage*, ki predstavljata ti dve strani. Če pa uporabnik že ima predstavitveno stran, se vrne seznam s ključem *fid* in *title*, torej identičnim ključem podzavihka in njegovim nazivom.

Celoten potek strani gradimo torej z elementi menija, v katerih se kličejo posamezene funkcije, kot lahko vidimo v tabeli 1. Dva glavna dela URL naslovov, kjer se modul pojavlja, sta *mypage*, ki omogoča uporabnikom urejanje lastne spletne strani ter *mypage-all*, ki vsebuje pregled nad vsemi člani in njihovimi predstavitvenimi stranmi. Iz obeh naslovov kličemo iste funkcije. Npr. funkcijo *mypage_edit()* kličemo iz elementa s potjo *mypage/uporabnik/edit* ter naslova *mypage-all/uporabnik/edit*. Znotraj funkcije potem ločimo, zaradi same narave uporabnikom prijaznega naslavljanja, dostop do baze glede na globalno spremenljivko `$user->name` (v primeru urejanja lastne strani) ali pa dostop do baze glede na drugi argument poti, ki predstavlja ime uporabnika.

URL naslov *(www.kosmagora.com/...)*	Vsebina	Funkcija s povratnim klicem	Dostop
/mypage-members	Urejen seznam vseh članov skupnosti, ki vključuje uporabniško ime s povezavo na ogled profila, povezavo na njegovo predstavitveno stran, če le ta obstaja ter povezavo na kontakt.	*mypage_view_members*	*view mypage, edit mypage, manage my pages*
/mypage/ime_uporabnika/view/st_podzavihka	Vsebina podzavihka z določeno številko predstavitvene strani uporabnika. Če *st_podzavihka* ni določena, se prikaže vsebina prvega podzavihka.	*mypage_view*	*view mypage, edit mypage, manage my pages*
/mypage/ime_uporabnika/edit/st_podzavihka	Obrazec za urejanje vsebine podzavihka z določeno številko predstavitvene strani uporabnika. Če *st_podzavihka* ni določena, se prikaže obrazec za urejanje vsebine prvega podzavihka.	*mypage_edit*	*edit mypage, manage my pages*
/mypage/ime_uporabnika/delete/st_podzavihka	Izbris podzavihka s ustrezno številko izbranega uporabika in preusmeritev na */mypage/ime_uporabnika.*	*mypage_delete*	*edit mypage, manage my pages*
/mypage/ime_uporabnika/settings	Seznam podzavihkov uporabnika s povezavami *Delete* in *Edit* ter obrazcem za ustvarjanje novega zavihka, če je število trenutnih zavihkov pod določeno mejo.	*mypage_own_main*	*edit mypage, manage my pages*
/mypage-all/ime_uporabnika/view/st_podzavihka	Vsebina podzavihka z določeno številko predstavitvene strani uporabnika. Če *st_podzavihka* ni določena, se prikaže vsebina prvega podzavihka.	*mypage_view*	*view mypage, edit mypage, manage my pages*

Tabela 1. Prikaz strukture menija modula *Mypage*(1.del).

URL naslov (www.kosmagora.com/...)	Vsebina	Funkcija s povratnim klicem	Dostop
/mypage-all/ime_uporabnika/edit/st_podzavihka	Obrazec za urejanje vsebine podzavihka z določeno številko predstavitvene strani uporabnika. Če *st_podzavihka* ni določena, se prikaže obrazec za urejanje vsebine prvega podzavihka.	*mypage_edit*	*manage my pages*
/mypage-all/ime_uporabnika/delete/st_podzavihka	Izbris podzavihka s ustrezno številko izbranega uporabika in preusmeritev na */mypage-all/ime_uporabnika.*	*mypage_delete*	*manage my pages*
/mypage-all/ime_uporabnika/settings	Seznam podzavihkov uporabnika s povezavami *Delete* in *Edit* ter obrazcem za ustvarjanje novega zavihka, če je število trenutnih zavihkov pod določeno mejo.	*mypage_settings*	*manage my pages*
/mypage-all/ime_uporabnika/blog	Povzetek blogov določenega uporabnika (del jedra sistema).	*mypage_view*	*view mypage, edit mypage, manage my pages*
/mypage-all/ime_uporabnika/contact	Kontaktni obrazec določenega uporabnika (del jedra sistema).	*mypage_contact*	*view mypage, edit mypage, manage my pages*
/admin/settings/mypage	Obrazec za določitev maksimalnega dovoljenega števila podzavihkov.	*mypage_admin*	*admin*

Tabela 1. Prikaz strukture menija modula *Mypage* (2.del).

Predstavitvena stran podjetja ima tri glavne zavihke *View*, *Edit* in *Settings*. Vsak zavihek je realiziran z ustrezno funkcijo. *View* poskrbi za prikaz vsebine in gre le za pridobivanje podatkov iz tabele *mypage* v podatkovni bazi. *Edit* vsebuje obrazec (angl. form) za vnos naslova podzavihka in vsebine. Tu sem uporabila vmesnik Form API in vsebino zakodirala s kodirnikom *base64_encode*. Samega urejevalnika vsebine in vmesnika za prenos in obdelavo slik mi ni bilo potrebno razvijati, saj Drupal ponuja kar nekaj možnosti. Te so opisane v razdelku 4.4. Zavihek *Settings* pa vsebuje prikaz podzavihkov z možnostmi urejanja in brisanja obenem pa v funkciji `mypage_own_main()` preverimo tudi število dovoljenih

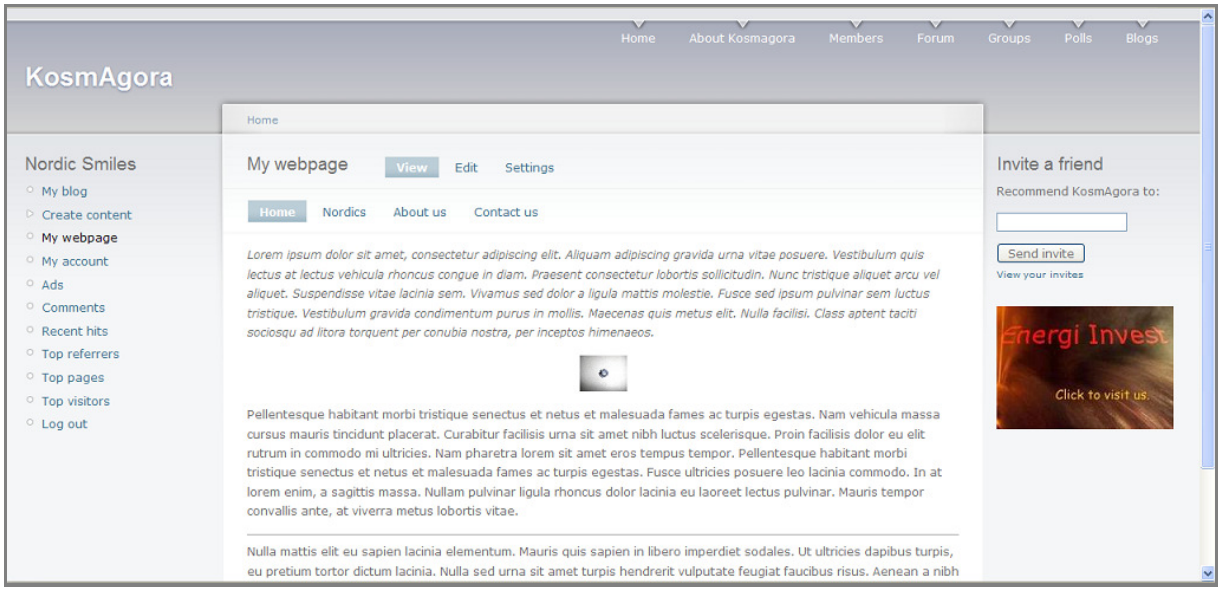
podzavihkov. Če meja dovoljenih podzavihkov še ni bila prekoračena, je prikazan tudi obrazec za ustvarjanje novega podzavihka.

Poleg vmesnika za urejanje lastne predstavitvene strani vsebuje modul *Mypage* tudi pregledno stran, kjer se nahaja seznam uporabnikov s povezavami na njihove predstavitvene strani, povzetki blogov in kontaktno stranjo. Ta stran je realizirana s funkcijo `mypage_view_members()`.

Z obrazcem za administracijo, ki se nahaja na naslovu */admin/settings/mypage* lahko administrator spremeni dovoljeno število podzavihkov. Privzeta vrednost spremenljivke *mypage_maxtabs* je 7. Element menija, ki je bil dodan s kljuko *mypage_admin()*, je realiziran s kodo:

```
$items[] = array(
  'path' => 'admin/settings/mypage',
  'title' => t('Mypage module settings'),
  'description' => t('Description of your Mypage settings control'),
  'callback' => 'drupal_get_form',
  'callback arguments' => 'mypage_admin',
  'access' => user_access('access administration pages'),
  'type' => MENU_NORMAL_ITEM,
);
```

Funkcija `drupal_get_form`, ki je tu uporabljena, je opisana v razdelku 5.5. Zgornji del kode zgolj ponazarja kako se ustvari preprost element menija. Kljuka *mypage_menu()* vsebuje precej bolj zapleteno hierarhijo menijev, ki jih uporabniki lahko dodajajo, urejajo in brišejo. Na sliki 13 je primer pogleda PS pojetij, z zavihki, ki so produkt kljuke *mypage_menu()*.



Slika 13. Predstavitvene strani podjetij v poslovni spletni skupnosti.

hook_user()

Ta kljuka omogoča modulom, da se odzovejo, ko so izvedene določene operacije nad uporabniškimi računi. Za konsistentnost podatkov moramo poskrbeti sami. Tako programsko ob izbrisu uporabnika, izbrišemo tudi njegovo predstavitveno stran. Spodaj je del kode *Mypage* modula, ki ponazarja kljuko *mypage_user()*.

```
function mypage_user($op, &$edit, &$user) {
  if ($op == 'delete') {
      db_query('DELETE FROM {mypage} WHERE uid = %d', $user->uid);
  }
}
```

5.5 Form API

Vmesnik Form API je Drupalov programski vmesnik za generiranje, validacijo in obdelavo HTML obrazcev. Form API definirane obrazce pripravi za predstavitev in ob predložitvi validira, obdela in shrani podatke obrazca. Obrazec je definiran kot vgnezden seznam lastnosti (angl. properties) in vrednosti (angl. values). Ta seznam se nato postopoma pretvori v ustrezno HTML obliko obrazca. Form API poskrbi za obravnavo, obdelavo in manipulacijo obrazca, ki ga je razvijalec definiral. Razvijalci tako lahko brez težav dodajamo in spreminjamo obrazce. Prav tako lahko enostavno dodamo validacijo obrazca, saj se za validacijo uporablja ista podatkovna struktura kot se je ob generiranju obrazca. [12, 4]

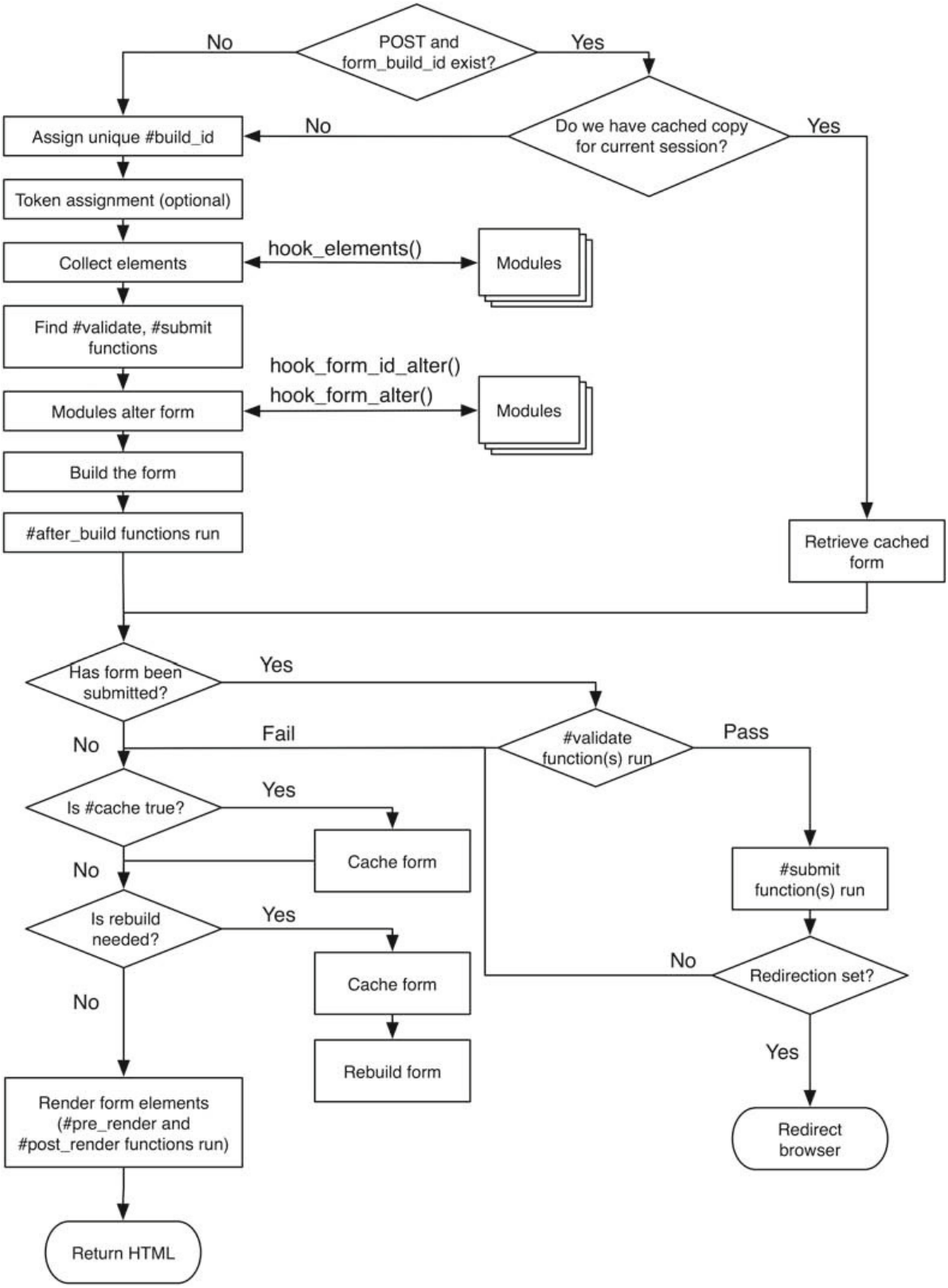
5.5.1 Funkcija `drupal_get_form`

Parameter funkcije `drupal_get_form($form_id)` je unikaten niz, ki identificira obrazec. Gre za funkcijo s povratnim klicem. Vsi opcijski dodatni parametri pa predstavljajo parametre funkcije s povratnim klicem, če jih le-ta zahteva. Funkcija `drupal_get_form()` tako uporabi rezultat podane funkcije s povratnim klicem, da ustvari strukturo obrazca, obdela obrazec in ga vrne v HTML obliki. Spodaj je delček kode, ki bo generiral obrazec z identifikacijo `'mypage_add_new_form'`. Dodatni parameter `$uname` služi kot parameter funkcije `mypage_add_new_form()`.

```
$page_content .= drupal_get_form('mypage_add_new_form', $uname);
```

Na sliki 14 (vir[12]) lahko vidimo kako Drupal s funkcijo `drupal_get_form()` obdela obrazce. [1, 22] Najprej zgradi obrazec, ga sprocesira, preoblikuje ali pa preusmeri na ustrezno mesto. Obrazec se procesira glede na definicijo obrazca v predhodnem koraku in na koncu preoblikuje v zahtevano HTML obliko. Funkcija poskrbi, da so obrazci prikazani in

varno predloženi (angl. submitted) preko treh faz: validacija, oddaja (angl. submission) in preusmeritev (angl. redirection).



Slika 14: Form API

5.5.2 Konstruktor obrazca

Konstruktor obrazca je funkcija, s katero ustvarimo obrazec. Termina konstruktor si tu ne smemo razlagati kot konstruktor v smislu objektno-orientiranega programiranja. Konstruktor obrazca zgradi podatkovno strukturo obrazca in deluje kot kljuka.
Opisala bom preprost obrazec *Create new page*, ki je del modula *Mypage* (sliki 15 in 16).

Slika 15: Obrazec “Dodaj novo PS”.

Spodaj je konstruktor obrazca s slike 15.

```
function mypage_add_new_form($uname){
      $form['mypage_add_new']['title'] = array(
            '#type' => 'textfield',
            '#title' => t('New page title'),
            '#default_value' => '',
            '#weight' => 0,
            '#size' => 40,
            '#maxlength' => 80,
            '#required' => TRUE,
            '#description' => t('Naming each page ...')
      );
      $form['mypage_add_new']['submit'] = array(
            '#type' => 'submit',
            '#submit' => TRUE,
            '#value' => t('Add new page')
      );
      $form['mypage_add_new']['uname'] = array(
            '#type' => 'value',
            '#value' => $uname
      );
return $form;
}
```

Spremenljivka `$form` je asociativni seznam, ki hrani elemente obrazca. Posamezen element je tudi definiran z asociativnim seznamom, ki opisuje konfiguracijo elementa. Funkcija `mypage_add_new_form()` vrne obrazec s tremi elementi. Ključ elementa `['mypage_add_new']['title']` definira prvi element. Vrednost tega elementa je asociativni seznam z množico lastnosti, ki ga opisujejo. Opisala bom le nekaj najbolj uporabljenih lastnosti. Lastnost `#type` določa tip elementa. Poleg osnovnih tipov HTML obrazcev imamo na voljo še nekaj tipov, značilnih za Drupal (npr. `#value`, ki sem ga uporabila v tretjem

elementu). Lastnost `#value` hrani vrednost elementa. Lastnost `#required = TRUE` pove, da je polje obvezno. To je razvidno tudi v HTML obliki, kjer je poleg polja prikazana zvezdica (slika 15). Celoten seznam lastnosti obrazcev z opisi lahko najdemo na Drupalovi spletni strani [21].

Element `['mypage_add_new']['submit']` je tipa `#submit` in kliče funkcijo `mypage_add_new_form_submit`. Ta element bo upodobljen kot gumb za potrditev. Ker bomo potrebovali spremenljivko `$uname` ob procesiranju obrazca, jo želimo z obrazcem posredovati naprej. Element `['mypage_add_new']['uname']` za razliko od prvih dveh, ni upodobljen v HTML obliki, ampak samo hrani referenco na spremenljivko `$uname`. Ponavadi obrazci za ta namen uporabljajo lastnost `#hidden`, Drupal pa ima ravno za te namene lastnost `#value`, ki elementa nikoli ne upodobi v HTML obliki. Do objekta `$uname` dostopamo s klicem `$form_values['uname']`, kot smo naredili v predloženi funkciji `mypage_add_new_form_submit()`.

Funkcija `drupal_get_form()` bo najprej preverila ali so vsa polja, ki so definirana v konstruktorju obrazca, veljavna in v pravilni obliki. Potem pa bo izvedla še dve funkciji s povratnim klicem: `validate()` in `submit()`.

5.5.3 Funkcija `validate`

Namen validacije je, da preprečimo nesmiselne vnose v polja obrazca. Če pride do napake, bo opis le-te prikazan uporabniku. Če pa je bil vnos pravilen, bo Drupal sprocesiral dejanske predložene (angl. submitted) vrednosti. Validacijsko funkcijo zapišemo z imenom obrazca in pripono `_validate()`. Za naš obrzec sicer ne potrebujemo te funkcije, ker vso osnovno validacijo opravi že Drupalova funkcija `_form_validate`. Če bi želeli dodati še kake specifične omejitve našemu obrazcu, bi morali napisati funkcijo `mypage_add_new_form_validate()`.

Poznamo več vrst validacij, validacija žetona, vgrajena validacija (npr. `#maxlength`), validacija vezana na element (`#element_validate`). Naš obrazec vključuje tekstovno polje, ki ima lastnost `#maxlength`, ki poskrbi da polje vsebuje le določeno število znakov. Za validacijo vnosa tega polja že poskrbi Drupal s svojo `_form_validate` funkcijo. Temu rečemo vgrajena validacija.

5.5.4 Funkcija `submit`

Po validaciji funkcija `drupal_get_form()` izvede klic naslednje funkcije, z imenom obrazca in pripono `_submit()`. Potrditvena funkcija določi, kaj se bo zgodilo s podatki obrazca. Ali bomo podatke obrazca shranili v podatkovno bazo, ali jih bomo posredovali vozlu Drupala? V naši funkciji predložitve dodamo nov zapis v tabelo *mypage*.

Parametra funkcije `_submit` sta `$form_id` in `$form_values`. Spremenljivka `$form_values` vsebuje asociativen seznam obrazca, ampak tokrat s podatki, ki jih je vnesel uporabnik. Tako s `$form_values['title']` dobimo naslov strani, ki ga je vpisal uporabnik v

tekstovno polje obrazca. Glede na spremenljivko `$form_values['uname']` pa vpišemo ustrezno ime uporabnika z imenom strani v tabelo *mypage*. Na koncu poskrbimo še za ustrezno preusmeritev strani s funkcijo `drupal_goto()`.

```
function mypage_add_new_form_submit($form_id, $form_values) {
      global $user;
      $uname = $form_values['uname'];
      $pages = mypage_get_page_num($uname);
      $i = 1;
      foreach ($pages as $key => $value) {
            if ($value['fid'] != $i) {
                  $page = $i;
            }
            $i++;
      }
      if(empty($page)) $page = $i;
      $body = base64_encode('No content yet.');
      if ($uname == $user->name) {
            $sql = "INSERT INTO {mypage} (uid, fid, title, body)
            VALUES (%d, %d, '%s', '%s')";
            db_query($sql, $user->uid, $page, $form_values['title'],
            $body);
            $goto = 'mypage/'. $user->name . '/edit/' . $page ;
      }
      else {
            $sql = "SELECT uid FROM {users} WHERE name = '%s'";
            $unameid = db_fetch_object(db_query($sql, $uname));
            $sql = "INSERT INTO {mypage} (uid, fid, title, body)
            VALUES (%d, %d, '%s', '%s')";
            db_query($sql, $unameid->uid, $page,
            $form_values['title'], $body);
            $goto = 'mypage-all/'. $uname . '/edit/' . $page ;
      }
      drupal_goto($goto);
}
```

Zgornja koda je primerna le za verzijo Drupal 5. Drupal 6 in višje verzije imajo za parametra funkcije `_submit $form` in `&$form_state`. Ker se vrednosti `$form_state` spremenljivke uporabijo pogosto v drugih dele kode, so se razvijalci Drupala 6 odločili, da je bolj smiselno spremenljivko `$form_state` podati z referenco. Na ta način bodo vse spremembe te spremenljivke dosegljive izven sklopa funkcije. Vrednosti obrazca sedaj dobimo preko `$form_state['values']`.

Za preusmeritev, bi lahko uporabila tudi lastnost `#redirect`, a Drupal 5 omogoča le nastavitev poti znotraj definicije elementa obrazca. Tako v našem primeru, ko določamo pot preusmeritve glede na ime uporabnika in številko strani, ne moremo redefinirati lastnosti `#redirect` v `_submit` funkciji, ampak za preusmeritev uporabimo funkcijo `drupal_goto()`. Drupal 6 nam omogoča dinamično preusmeritev. Tako bi lahko preusmeritev realizirali s `$form_state['redirect'] = sprintf($goto);` [4, 20]

6 Vrednotenje razvitega sistema

Ob zaključku razvoja posamezne enote programa (v sistemu Drupal je to modul), je potrebno enoto testirati (angl. unit testing). Temu manjšemu testu sledi še celovito testiranje in vrednotenje sistema, ki podpira poslovno spletno skupnost. Poznamo več vrst testiranj:

1. Testiranje varnosti sistema poskrbi za zaščito pred nepooblaščenim notranjim in zunanjim dostopom in namernim povzročanjem škode. Pri spletni skupnosti, ki je osnovana na sistemu Drupal to pomeni, da testiramo ali so bile pravice dostopa uporabnikov pravilno nameščene. Z množico dodatnih modulov se veča tudi tabela uporabniških pravic in s tem postane težji pregled nad kontrolo dostopa. Dodatni modul *Path Access* poskrbi za dodatno kontrolo dostopa, saj omogoča, da omejimo dostop določenim uporabnikom na vseh URL naslovih, razen tistih, ki smo jih posebej definirali. Za spletno skupnost KosmAgora dovolimo dostop anonimnim uporabnikom le do naslova *www.kosmagora.com/se* in nikamor drugam. S tem zagotovimo, da je spletna skupnost zaprta za avtentificirane uporabnike. Poleg tega pa je potrebno opraviti še vrsto testov, ki zagotovijo pravo vrsto dostopa (branje, urejanje, pisanje ali brisanje) za posamezno vlogo uporabnikov.

2. Testiranje obremenitve oz. zmogljivosti sistema. Sistem obremenimo z večjim številom transakcij, da vidimo katerih virov mu najprej zmanjka. Glede na rezultate se lahko odločimo za optimizacijo programske kode ali zamenjavo strojne opreme.

3. Testiranje uporabnosti. Preučimo uporabniške navade. Pogovor oz. intervju z uporabniki. [8]

6.1 Testiranje razvitega modula

Že med samim razvojem modula, pa če je ta še tako majhen, je potrebno testiranje. Poleg testov osnovnega delovanja modula, je potrebno preveriti tudi, če modul vpliva na druge dele sistema. V našem primeru, če je imel nek uporabnik ustvarjeno svojo predstavitveno stran in smo potem tega uporabnika izbrisali, moramo preveriti, da se v tabeli *mypage* ne nahajajo več podatki vezani na neobstoječega uporabnika. Potrebno je tudi opraviti test namestitve in odstranitve modula, saj se tu hitro pojavijo manjše sintaktične napake, če niso upoštevali pravil kodiranja. Novejše verzije Drupala že ponujajo modul *SimpleTest*[12], ki podpira testiranje modulov. Ker je naš modul napisan za Drupal 5.20, smo morali postopoma tekom razvoja opraviti naslednje teste.

[12] http://drupal.org/handbook/modules/simpletest

Test integracije modula

Pogoj: Namestitev/odstranitev modula.
Posledica: Nameščen/odstranjen modul ter ustvarjena/izbrisana tabela *mypage* v podatkovni bazi.

Test namestitevene strani administratorja

Pogoj 1: Na strani *admin/settings/page* administrator ponastavi vrednost *Max default tabs* na vrednost *n*.
Posledica 1: Ko je pri poljubnem članu kreiranih *n* podzavihkov, obrazca za kreiranje novega zavihka ni več na voljo.

Pogoj 2: Nek član *x* ima kreirano *n* podzavihkov. Administrator spusti mejo na $n < m$.
Posledica 2: Član *x* nima več na voljo obrazca za kreiranje novega podzavihka, poleg tega pa se izpiše opozorilo naj uporabnik izbriše presežene podzavihke.

Test strani pregleda članov

Pogoj: Izbris člana.
Posledica: Član se ne pojavi na seznamu vseh članov, poleg tega pa se izbrišejo vsi zapisi, ki se nanašajo nanj v tabeli *mypage*.

Test pravic dostopa

Pravice dostopa določimo pod *Administer/User management/Access control.*

Pogoj 1: Vloga nima obkljukane nobene pravice modula *Mypage*.
Pogoj 2: Vloga ima obkljukano pravico *view mypage*.
Posledica 1 in 2: Člani te vloge imajo pregled nad vsemi člani in njihovimi predstavitvenimi stranmi, nimajo pa lastne predstavitvene strani.

Pogoj 3: Vloga ima obkljukano pravico *edit mypage*.
Posledica 3: Člani te vloge imajo pregled nad vsemi člani in njihovimi predstavitvenimi stranmi, lahko pa tudi urejajo lastno predstavitveno stran.

Pogoj 4: Vloga ima obkljukano pravico *manage all mypages*.
Posledica 4: Člani te vloge imajo pregled nad vsemi člani in njihovimi predstavitvenimi stranmi ter možnost urejanja strani ostalih članov. Člani lahko tudi urejajo lastno predstavitveno stran.

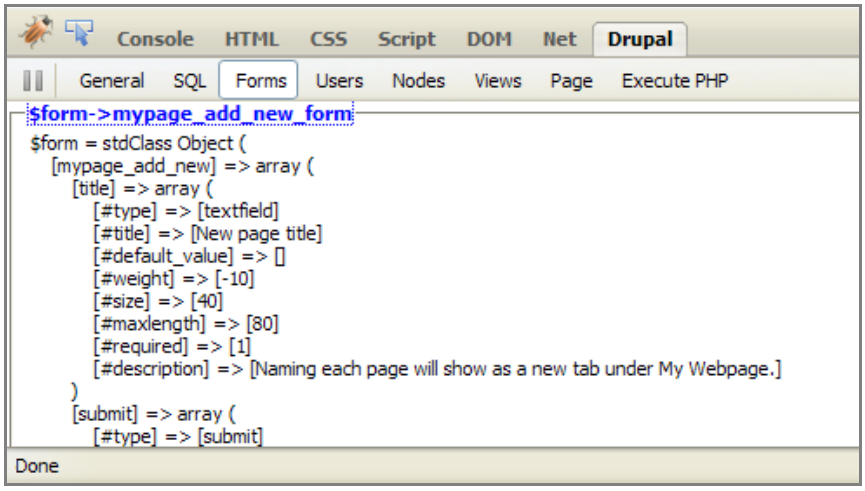
Test *TinyTinyMCE* in *IMCE* nastavitev

Pogoj: Člani imajo omogočen *TinyTinyMCE* način vnosa pri urejanju predstavitvenih strani. Prav tako je omogočen *IMCE* modul z ustreznimi nastavitvami, ki dodelijo posameznemu uporabniku datoteko na stražniku za hranjenje slik.
Posledica: Člani lahko uporabijo *TinyTinyMCE* urejevalnik besedila. Ko kliknejo na ikono za nalaganje slik, se prikaže vmesnik *IMCE*, preko katerga imajo tudi dostop do slik, ki so jih morda že naložili.

6.2 Testiranje zmogljivosti sistema

6.2.1 Drupal for Firebug

Pri razvoju spletnih strani nam je v pomoč Firefoxovo orodje Firebug[13], ki nam omogoča pregled nad HTML, CSS, Script, DOM in Net elementi. Firebug ima poseben podaljšek (angl. extension) Drupal For Firebug[14], ki ga lahko namestimo. Ta nadgradnja podpira iskanje specifičnih napak Drupala in njegovih statusnih sporočil. Imamo možnost pregleda procesiranja vozlov, obrazcev, objektov uporabnikov ter SQL poizvedb. Poleg nadgradnje orodja Firebug moramo namestiti tudi Drupalov modul *Drupal for Firebug*[15], da bo podaljšek deloval.



Slika 16. Izpis strukture obrazca “Dodaj novo PS” z orodjem *Drupal For Firebug*.

6.2.2 Devel

Dodatni modul *Devel*[16] je pravzaprav sklop štirih modulov: *Devel*, *Devel Node Acces* (prikaz pravic na trenutni strani), *Generator* (nam omogoča kreiranje tisoče uporabnikov) in

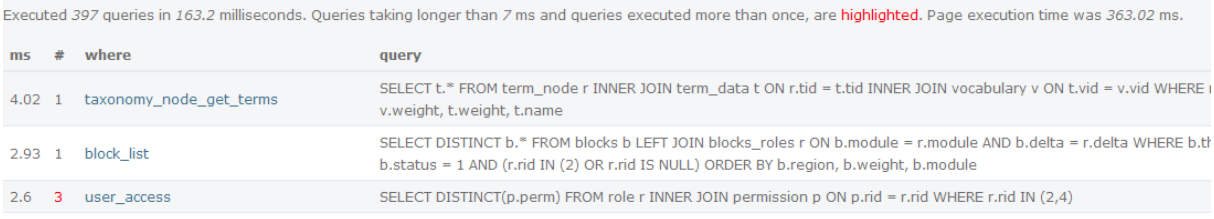
[13] http://getfirebug.com/
[14] https://addons.mozilla.org/en-US/firefox/addon/8370/
[15] http://drupal.org/project/DrupalForFirebug
[16] http://drupal.org/project/devel

Performace Logging. *Devel* je nepogrešljiv modul pri razvoju novih modulov ali pa spreminjanju obstoječe kode, saj vsebuje orodja za razhroščevanje in pregledovanje delčkov kode.

Seznam funkcionalnosti:

- *Empty cache* izvede `drupal_flush_all_cahes()` funkcijo, ki počisti predpomnilnike. Na novo stisnjene CSS in JavaScript datoteke dobijo nova imena, ponovno se zgradi register tem in meniji, tabela *node_type* je posodobljena in počiščene so predpomnilniške tabele v podatkovni bazi.
- *Execute PHP Code* nam omogoča izvedbo manjših izsekov PHP kode. S funkcijami `dpm()` , `dvm()` , `dpr()` in `dvr()` lahko izpišemo sporočila.
  - `dpm()` funkcija izpiše preprosto spremenljivko na mesto sporočil strani.
  - `dvm()` izpiše kompleksne spremenljivke, kot so objekti in seznami s funkcijo `var_dump()` na mesto sporočil.
  - `dpr()` izpiše kompleksne spremenljivke na vrh strani s pomočjo rekurzivne funkcije `dprint_r()`.
  - `dvr()` izpiše `var_dump()` na vrh strani.
- *Function reference* nam vrne seznam uporabniških funkcij, ki so bile določene s trenutno zahtevo. Drupal tu uporabi PHP funkcijo `get_defined_functions()`.
- *Reinstall modules* ponovno namesti module s kljukama *hook_uninstall()* ter *hook_install()*.
- *Reset menus* počisti in ponovno zgradi tabelo *menu_router* ter posodobi tabelo *menu_links*.
- *Variable editor* nam omogoča urejanje spremenljivk in njihovih vrednosti, ki so trenutno shranjene v tabeli *variables*. Do teh spremenljivk ponavadi dostopamo s funkcijama `variable_get()` in `variable_set()`.
- *Session viewer* prikaže vrednost spremenljivke `$_SESSION`.
- *Node access summary* prikaže povzetek dostopov do vsebine. [12]

Devel modul skriva množico funkcionalnosti, do katerih večinoma lahko dostopamo šele, ko omogočimo njegov blok. Pripomočki, ki se nahajajo na *admin/settings/devel* vključujejo dnevnik izvršenih poizvedb, urejenih po času izvedbe, prikaz časa izvedbe strani (angl. page execution time), koliko pomnilnika je bilo porabljeno za prikaz trenutne strani, prikaz preusmeritev strani, ki so bile preusmerjene s funkcijo `drupal_goto()`ter prikaz elementov obrazca z utežmi. Vse te informacije so zelo uporabne pri optimizaciji saj lahko tako opazimo počasne poizvede ter poizvedbe, ki se morda kličejo prevečkrat na isti strani (slika 17).


Executed 397 queries in 163.2 milliseconds. Queries taking longer than 7 ms and queries executed more than once, are highlighted. Page execution time was 363.02 ms.

ms	#	where	query
4.02	1	taxonomy_node_get_terms	SELECT t.* FROM term_node r INNER JOIN term_data t ON r.tid = t.tid INNER JOIN vocabulary v ON t.vid = v.vid WHERE … v.weight, t.weight, t.name
2.93	1	block_list	SELECT DISTINCT b.* FROM blocks b LEFT JOIN blocks_roles r ON b.module = r.module AND b.delta = r.delta WHERE b.th… b.status = 1 AND (r.rid IN (2) OR r.rid IS NULL) ORDER BY b.region, b.weight, b.module
2.6	3	user_access	SELECT DISTINCT(p.perm) FROM role r INNER JOIN permission p ON p.rid = r.rid WHERE r.rid IN (2,4)

Slika 17. Primer izpisa izvedenih poizvedb z modulom *Devel*.

6.2.3 Obremenitveni testi

Danes najdemo kar nekaj prosto dostopnih orodij za merjenje zmogljivosti spletnega strežnika. Med njimi so najbolj razširjeni ApacheBench[17], Siege[18], OpenWebLoad[19] in httperf[20] z dodatkom Autobench, ki avtomatizira teste in generira analizo rezultatov. Orodja za testiranje zmogljivosti v bistvu merijo hitrost procesiranja HTTP zahtev, ki so poslana spletnemu strežniku. Zahteve se pošiljajo s stalno hitrostjo, meri pa se hitrost odgovorov nanje. Če poženemo teste z monotono naraščajočo hitrostjo pošiljanja zahtev, lahko tako hitro najdemo hitrost, pri kateri strežnik postane preobremenjen. Ker je ApacheBench že nameščen s spletnim strežnikom Apache, sem opravila teste kar z njim.

Tehnične specifikacije strežnika

Poslovna spletna skupnost *www.kosmagora.com* je postavljena na navideznem zasebnem strežniku (angl. Virtal Private Server) z dodeljenim 5 GB trdega diska, 1024 MB RAM-a in procesorjem Intel Pentium DualCore 2.4 Ghz 1 MB. Na strežniku je nameščen operacijski sistem Linux Ubuntu 9.04. Programska oprema: XAMPP 1.7.1., ki vključuje Apache 2.0, PHP 5.2.9 in MySQL 5.1.33.

ApacheBench

ApacheBench[21] je program za obremenitveno testiranje (angl. command line load testing utility) in je del namestitvenega paketa spletnega strežnika Apache. ApacheBench poženemo z ukazom `ab` v ukazni vrstici. Preprost ukaz za testiranje spletnega mesta s 1000 zahtevami in stopnjo konurenčnosti 10 izgleda takole:

```
ab –n 1000 –c 10 http://www.kosmagora.com
```

Opcija `-n` označuje število vseh zahtev, opcija `-c` pa število istočasnih zahtev. Podroben opis opcij se nahaja na spletni strani orodja. Rezultat tega testa je nekaj več kot 2000 zahtev na sekundo. Vsebino odgovorov zahtev lahko vidimo, če izberemo opcijo `-v 4`. Opazila sem, da je šlo le za zahteve na preusmeritev, ki so bile zato tako hitro obdelane. V prvem primeru je ApacheBench pošiljal zahteve kot 10 anonimnih uporabnikov, ki se jim prikaže le začetna vstopna stran. V našem primeru pa je velika večina spletne vsebine dostopna le za avtentificirane uporabnike. Da bi stestirali zmogljivost spletnega strežnika za avtentificirane uporabnike, moramo testu `ab` posredovati informacijo o piškotku Drupalove seje. (angl. Drupal session cookie information). V nastavitvah brskalnika poiščemo piškotek avtentificiranega uporabnika, ki se začne z imenom *SESS*. Pomembni sta spremenljivki *Name* in *Content* [14]. Sledi ukaz za test seje avtentificiranega uporabnika.

[17] http://httpd.apache.org/docs/2.0/programs/ab.html
[18] http://www.joedog.org/index/siege-home
[19] http://openwebload.sourceforge.net/
[20] http://www.hpl.hp.com/research/linux/httperf/
[21] http://httpd.apache.org/docs/2.2/programs/ab.html

```
ab –n 10000 –c 5 –C ' SESS8720439dcd6a713becc72a8fd041936e =
d53b9bc5390a73f956ab5c5974eb194f ' http://www.kosmagora.com/se/mypage-
members
```

Po množici testov na spletnem strežniku Apache poslovne spletne skupnosti, sem prišla do naslednjih rezultatov.

št. sočasnih zahtev	št. zahtev / s	št. neobdelanih zahtev
1	17.71	0
5	21.53	0
10	21.70	0
15	20.00	0
20	20.56	0
25	21.42	0
30	20.58	0
35	20.73	0
40	20.73	0
45	20.80	0
50	20.68	0
55	38.11	706

Tabela 2. Rezultat ApacheBench testov.

V povprečju strežnik obdela 20,87 zahtev na sekundo, preobremenjen pa postane pri 55 sočasnih dostopih. Če predpostavimo, da opravi aktiven uporabnik v spletni skupnosti 10 klikov na minuto, to pomeni, da lahko naša spletna skupnost prenese vsaj 300 sočasnih avtentificiranih uporabnikov. Rezultat ni najboljši, vendar ne smemo pozabiti, da gre za starejšo verzijo Drupal 5, s katero večina spletnih strežnikov obdela nekje od 20 do 100 zahtev na sekundo le z osnovnimi moduli. Z večjim pomnilnikom bi lahko stregli večjemu številu uporabnikov naenkrat. Ker je trenutno v spletni skupnosti dvajset aktivnih članov in so napovedi povečanja v prihodnjem letu do nekaj sto članov, bo razvit sistem popolnoma zadostoval potrebam uporabnikov.

Predpomnjenje

Večina spletnih strani, ki so nastale z orodjem Drupal, je z uporabo predpomnjenja (angl. caching) za anonimne uporabnike desetkrat bolj učinkovita. Učinkovitost se meri v številu zahtev na sekundo. Sistem predpomnjenja doseže tako učinkovitost s shrambo rezultatov določenih poizvedb v posebnih predpomnilniških tabelah. S tem se izognemo ponavljajočim se zahtevnim poizvedbam in tako prihranimo na času izvedbe.

Drupal 5 ne uporablja predpomnjenja za avtentificirane uporabnike. Drupalov sistem za pravice omogoča prikaz različnih vsebin, glede na različno vlogo uporabnikov. Vsebina je torej specifična za posameznega uporabnika in nam prepomnjenje tu ne koristi veliko. Ker je naša spletna skupnost namenjena avtentificiranim uporabnikom in imajo anonimni uporabniki

dostop le do vstopne strani, uporaba orodij za predpomnjenje APC in eAccelerator ne prinese večjih hitrosti, kot bi si želeli. Za verzijo Drupal 6 je na voljo modul *Authcache*[22], ki omogoča predpomnjenje za avtentificirane uporabnike. Ta modul zahteva za pravilno delovanje tudi določene spremembe na strani.

6.3 Uporabniška izkušnja

Že v prvih fazah razvoja spletne skupnosti je pomembna komunikacija z uporabniki. Že pred uradno objavo spletne skupnosti je pametno preizkusiti stran na testnih uporabnikih. Pomembno je, da ti uporabniki predstavljajo končno skupino uporabnikov. Uporabniki spletne skupnosti KosmAgora so podjetniki, ki niso tako tehnično podkovani. Prvi odziv testnih uporabnikov je bil precej slab. Na to je vplivalo pomanjkanje same vsebine strani in množica funkcionalnosti, ki na začetku ni bila najbolj organizirana. Za nekoga precej preprosta operacija dodajanja slik na lastno predstavitveno stran, je drugemu povzročala probleme. V ta namen se je strani dodala pomoč, ki nudi uporabnikom informacije o strukturi vsebin spletne skuposti in urejanju posameznih vsebin. Tudi sama struktura menijev se je v začetnih fazah izoblikovala, tako da sedaj levi stranski meni ponuja možnosti urejanja lastnih vsebin in nastavitev uporabnika, glavni meni na vrhu strani pa vsebuje množico povezav na posamezne dele spletne skupnosti, ki so namenjeni vsem uporabnikom. Odziv testnih članov spletne skupnosti je bil odličen in začeli so povabljati nove uporabnike. Trenutno ima skupnost okoli dvajset aktivnih članov, ki predstavljajo manjša podjetja v Goteborgu in nekaj podjetij, ki je pod okriljem organizacije ALMI. Ker spletna skupnost ponuja tako širok spekter funkcionalnosti, so bili nekateri uporabniki na začetku malo zmedeni. Precej funkcionalnosti je med seboj podobnih in zaradi tega tudi odveč. Zaenkrat se še ni izkazala potreba po forumu, saj uporabniki dosti raje objavljajo komentarje znotraj skupin. Prav tako ni noben član ustvaril svojega bloga, ampak se raje poslužujejo predstavitvene strani, ki se je izkazala za izredno priljubljeno funkcionalnost. Uporabniki spletne skupnosti so v splošnem zadovoljni in stran tedensko obiščejo.

[22] http://drupal.org/project/authcache

7 Zaključek

Rezultat diplomskega dela je poslovna spletna skupnost KosmAgora, ki ponuja njenim članom možnosti sodelovanja, izmenjave znanja in različnih načinov komunikacije. Posebna funkcionalnost te spletne skupnosti so predstavitvene strani podjetij, ki jih omogoča modul, ki sem ga razvila. Največji izziv diplomskega dela je bil ravno razvoj Drupal modula. Potrebno se je bilo poglobiti v zaledje sistema in razumeti njegove funkcije. S pomočjo vrste literature o sistemu Drupal, ki je na voljo in podpore Drupalove virtualne skupnosti, v obliki forumov in IRC kanalov, lahko napreden uporabnik izdela preprost modul v dobrem tednu dni. Razvoj modula, ki prinese novo funkcionalnost in njegova integracija s sistemom, kot je modul predstavitvenih strani podjetij, pa je vzelo dober mesec.

Ugotovila sem, da je izjemno pomembna temeljita analiza uporabniških zahtev. Pravtako je zaželjeno vključiti testne uporabnike že v prvih fazah razvoja. Testni uporabniki nam dajo povratno informacijo o uporabnosti sistema, na podlagi katere se odločimo o morebitnih spremembah. Težav pri osnovni namestitvi sistema ni bilo, pojavile pa so se manjše težave ob namestitvi nekaterih dodatnih modulov. Dostikrat se nameščen modul ni izkazal za najbolj učinkovitega oziroma ni nudil funkcionalnosti, ki bi jih pričakovali. Pri namestitvi dodatnih modulov dostikrat vzame čas neintuitivna nastavitev le teh. Določen modul je na primer potrebno omogočiti in nato nastaviti njegove lastnosti in pravice na več koncih. Na primer pod moduli, bloki, meniji in posameznih nastavitvenih straneh novega modula. Poleg tega je potrebno dodati ustrezne pravice dostopa in preveriti odvisnost od drugih modulov.

Ko je postavljen sistem, ki nudi vse potrebne funkcionalnosti poslovne spletne skupnosti, je potrebno motivirati uporabnike, da se bodo včlanili v skupnost, v njej ostali in rasli znotraj skupnosti. Uporabnike mora skupnost prepričati, da bodo z včlanitvijo nekaj pridobili, da se bodo znotraj nje lahko udejstvovali in vplivali na njeno vsebino. To smo dosegli z aktivno objavo novih vsebin, ki opisujejo spletno skupnost in nudijo pomoč uporabnikom. Najbolj aktivnim uporabnikom je bila omogočena tudi možnost oglaševanja.

Rešitev, ki sem jo predstavila, uporablja trenutno okoli dvajset »Start-up« podjetij v Goteborgu, med katerimi je večina novih, mladih podjetij in nekaj podjetij, ki spadajo pod krovnim podjetjem ALMI. Poslovna spletna skupnost se stalno razvija in izboljšuje, saj zahteve uporabnikov rastejo s številom članov skupnosti. Delo na projektu KosmAgora sem zaključila meseca maja 2010. Od takrat je stran dobila svojo vizualno podobo in pa klepetalnico. V prihodnosti se bodo gotovo odstranile določene funkcionalnosti, ki so trenutno manj priljubljene in se dodale nove. Taka je na primer blog, ki se ga zaradi obstoja predstavitvenih strani podjetij, praktično ne uporablja. Trenutna rešitev je popolnoma ustrezna za nekaj sto članov, če predpostavimo, da se njihove potrebe ne bodo drastično spreminjale. V prihodnosti, ko pa bo število uporabnikov naraslo iz nekaj sto na nekaj tisoč, pa bo nujna nadgradnja sistema na Drupal 6, oz. Drupal 7, ki se trenutno hitro razvija. Organizacija Drupal nudi podporo le dvem verzijam naenkrat, tako se že sedaj novi moduli za Drupal 5 verzijo ne razvijajo več. Ravno nadgradnja na Drupal 6 prinese vrsto izboljšav, kot so optimizacija kode, boljša varnost in skalabilnost sistema. Nadgradnja celotnega sistema

vključuje nadgradnjo modulov, od katerih morda ne bodo vsi dosegljivi v novejši različici. Potrebna bo tudi nadgradnja razvitega modula *Mypage,* predvsem zaradi novosti sistema *menu* in vmesika Form API.

Seznam slik

Seznam tabel

Dodatek A

Namestitev sistema Drupal 5

A.1 Tehnične zahteve sistema

- Spletni strežnik Apache 1.3, Apache 2.x ali Microsoft IIS, izmed katerih je najbolj preizkušen spletni strežnik Apache.
- Strežnik podatkovne baze. Pri uporabnikih sistema Drupal sta najbolj razširjena strežnika MySQL ter PostgreSQL, MySQL 3.23.17 ali novejše verzije za Drupal 5 ter PostgreSQL 7.4 ali novejše različice za Drupal 5 in Drupal 6.
- PHP 4.4.0 ali novejše različice. Drupal 5 še ne podpira PHP 5.3., podpirajo pa ga različice Drupal 6.14 in novejše. [28]

A.2 Namestitev strežnika XAMPP

XAMPP[23] je akronim za večplatformni (anlg. cross-platform) spletni strežnik, ki vključuje HTTP strežnik Apache, podatkovno bazo MySQL in interpreter jezikov PHP in Perl ter še množico dodatnih modulov, med katerimi sta najbolj uporabljena OpenSSL in phpMyAdmin. Za Drupal 5.20 je potrebna namestitev paketa XAMPP 1.7.1, ki vsebuje PHP 5.2.9. Z aplikacijo phpMyAdmin lahko preprosto dostopamo do podatkovne baze, ustvarimo novo bazo in urejamo pravice uporabnikov. Po zagonu namestitvene datoteke *setup_xampp.bat*, lahko z datotekama *apache_start.bat* in *mysql_start.bat* poženemo storitve strežnikov Apache in MySQL. Če je lokalna namestitev uspela, bi moral brskalnik na naslovu *http://localhost* prikazati začetno XAMPP stran. Zaradi varnosti je potrebno takoj dodati geslo za dostop do podatkovne baze ter ostale varnostne nastavitve, ki jih lahko določimo na naslovu *http://localhost/security/xamppsecurity.php*.
S pomočjo vmesnika phpMyAdmin se lahko preprosto ustvari podatkovna baza, ki bo služila ogrodju Drupal.

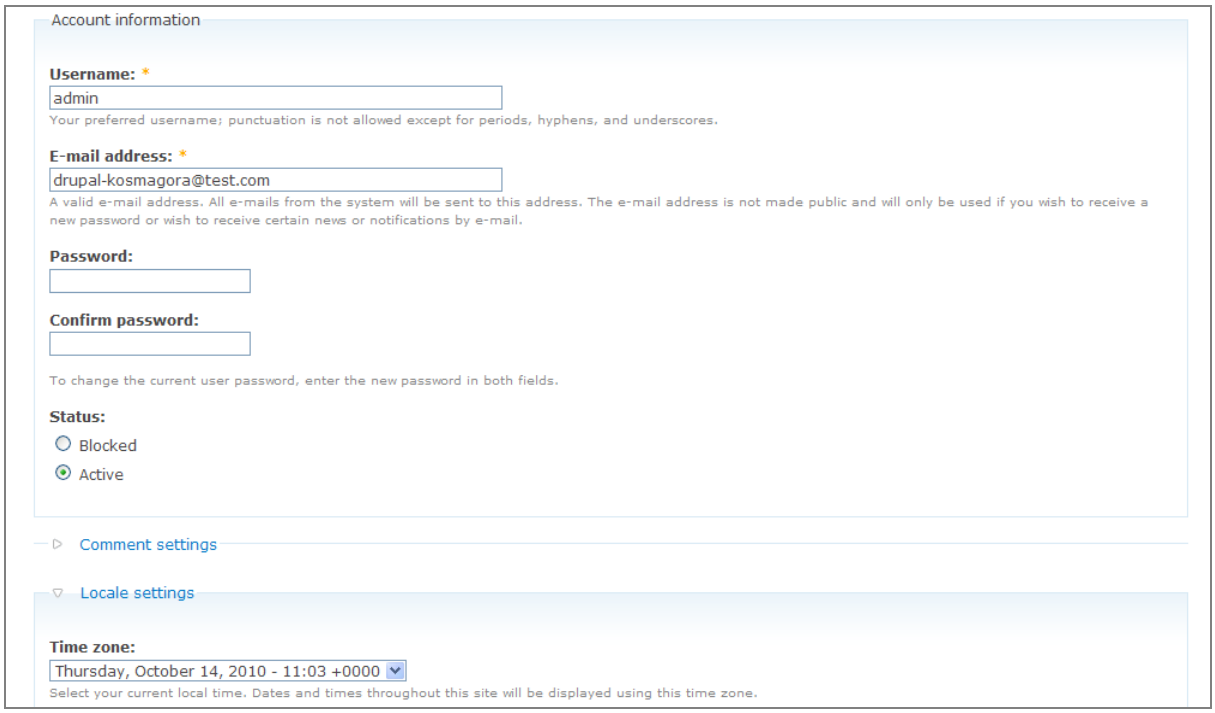
A.3 Namestitev sistema Drupal 5

Namestitveni paket Drupal, ki ga lahko prenesmo s spletnega naslova http://www.drupal.org. Paket vsebuje glavno mapo, znotraj katere se nahajajo datoteke celotnega sistema. Ime te mape predstavlja tudi del spletnega naslova, na katerem se bo nahajala Drupalova spletna stran. Za primer podamo ime *drupal_kosma*. Mapo vstavimo znotraj mape *htdocs*, na primer *C:\xampp\htdocs*. V mapi *C:\xampp\htdocs\drupal_kosma\sites\default* se nahaja nastavitvena datoteka *settings.php*, kjer so določene naslednje nastavitve:

[23] http://www.apachefriends.org/en/xampp.html

```
$db_url = 'mysql://username:password@localhost/databasename';
$base_url = 'http://www.example.com';
```

Na naslovu *http://localhost/drupal_kosma/install.php* se izvede namestitvena skripta. Preko brskalnika tako preprosto vnesemo osnovne podatke kot so ime spletne strani, elektronski naslov spletne strani ter ustvarimo prvi račun (slika 18). Prvi račun, s `uid` enako 1, ima določeno uporabniško vlogo administratorja, ki ima pravico dostopa do vseh delov strani. [5]



Slika 18. Pogled Drupalove strani: Ustvaritev prvega računa.

Literatura

[1] I. Alexander, N. Maiden, *Scenarios, Stories, Use Cases: Through the Systems Development Life-Cycle*, John Wiley & Sons, 2004, pogl. 3.

[2] T. Brinck, D. Gergle, *Usability for the web, designing web sites that work*, Academic Press, 2002.

[3] M. Butcher, *Drupal 6 JavaScript and jQuery*, Packt Publishing, 2009.

[4] M. Butcher, *Learning Drupal 6 Module Development: A Practical Tutorial for CreatingYour First Drupal 6 Modules with PHP*, Packt Publishing, 2008.

[5] R. T. Douglass, M. Little, J.W. Smith, *Building Online Communities with Drupal, phpBB, and WordPress*, Apress, 2006, pogl. 1.

[6] A. Ebersbach, M. Glaser, R. Heigl, A. Warta, *Wiki Web Collaboration*, Springer, 2008

[7] J. A. Green, *Drupal 6 Content Administration: Maintain, Add, and Edit the Content of Your Drupal site with Ease*, Pack Publishing, 2009

[8] M. Lapajne, *Testno voden razvoj programske opreme v Javi*, diplomsko delo, Ljubljana, Fakulteta za računalništvo in informatiko, 2010, pogl. 2.

[9] D. Mercer, *Building Powerful and Robust Websites with Drupal 6*, Packt Publishing, 2008.

[10] M. Pucelj, *Sistemi za upravljanje spletnih vsebin*, diplomsko delo, Ljubljana, Fakulteta za računalništvo in informatiko, 2009.

[11] M. Trtnik, *Razvoj aplikacij spletnih socialnih mrež za izvedbo virusnega marketinga na primeru Facebook platforme*, diplomsko delo, Fakulteta za računalništvo in informatiko, 2009.

[12] J. K. VanDyk, *Pro Drupal Development*, 2nd Edition, Apress, 2008.

[13] (2009) *Applying patches.* Dostopno na: http://drupal.org/patch/apply

[14] (2008) *Benchmarking Authenticated Drupal Users with ApacheBench.* Dostopno na: http://ezra-g.com/blog/20080229/benchmarking-authenticated-drupal-users-with-apachebench

[15] (2010) *Clean URLs.* Dostopno na: http://drupal.org/node/15365

[16] (2010) *Coding standards*. Dostopno na: http://drupal.org/node/318

[17] (2009) *Creating modules for 5.x - a tutorial.* Dostopno na: http://drupal.org/node/82920

[18] (2010) *Database abstraction layer.* Dostopno na: http://api.drupal.org/api/drupal/includes--database.inc/group/database/5

[19] (2010) *Drupal, Community plumbing.* Dostopno na: http://drupal.org/

[20] (2009) *Drupal 5.x to 6.x FormAPI changes.* Dostopno na: http://drupal.org/node/144132

[21] (2009) *Forms API Reference.* Dostopno na: http://api.drupal.org/api/drupal/developer--topics--forms_api_reference.html/5

[22] (2010) *Form API drupal_get_form*. Dostopno na: http://api.drupal.org/api/function/drupal_get_form/5

[23] (2010) *Hooks*. Dostopno na: http://api.drupal.org/api/group/hooks/5

[24] (2010) *Hook menu*. Dostopno na: http://api.drupal.org/api/function/hook_menu/5

[25] (2010) *Menu System*. Dostopno na: http://api.drupal.org/api/group/menu/5

[26] (2010) *Slovar informatike*. Dostopno na: http://www.islovar.org/iskanje_enostavno.asp

[27] (2007) *Social software building blocks*. Dostopno na: http://nform.ca/publications/social-software-building-block

[28] (2010) *System requirements*. Dostopno na: http://drupal.org/requirements