UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Matej Božič

Razvoj razširitve za opis in ocenjevanje restavracij za sistem za upravljanje vsebin Joomla

DIPLOMSKO DELO VISOKOŠOLSKEGA STROKOVNEGA ŠTUDIJA

LJUBLJANA, 2012

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Matej Božič

Razvoj razširitve za opis in ocenjevanje restavracij za sistem za upravljanje vsebin Joomla

DIPLOMSKO DELO VISOKOŠOLSKEGA STROKOVNEGA ŠTUDIJA

Mentor:
viš. pred. dr. Damjan Vavpotič

LJUBLJANA, 2012

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00552/2012
Datum: 02.02.2012

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

Kandidat: **MATEJ BOŽIČ**

Naslov: **RAZVOJ RAZŠIRITVE ZA OPIS IN OCENJEVANJE RESTAVRACIJ ZA SISTEM ZA UPRAVLJANJE VSEBIN JOOMLA**

DEVELOPMENT OF AN EXTENSION FOR DESCRIPTION AND EVALUATION OF RESTAURANTS IN JOOMLA CMS

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija

Tematika naloge:

V teoretičnem delu diplomske naloge predstavite sistem za upravljanje vsebin Joomla in primerjajte različne možnosti, ki so na voljo za razširitev funkcionalnosti sistema. V praktičnem delu naloge se posvetite razvoju razširitve za opis in ocenjevanje restavracij. Najprej predstavite problematiko in motiv za razvoj takšne razširitve, nato pa s pomočjo diagramskih tehnik UML pripravite analizo in načrt razširitve. Na podlagi pripravljenega načrta izdelajte programsko kodo razširitve ter predstavite njeno delovanje v okviru sistema Joomla.

Mentor:

viš. pred. dr. Damjan Vavpotič

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

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

diplomskega dela

Spodaj podpisani **Matej Božič**,

z vpisno številko **63060396**,

sem avtor diplomskega dela z naslovom:

Razvoj razširitve za opis in ocenjevanje restavracij za sistem za upravljanje vsebin Joomla

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom (naziv, ime in priimek)

  viš. pred. dr. Damjan Vavpotič

  in somentorstvom (naziv, ime in priimek)

  /

- 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 15.4.2012 Podpis avtorja: _____

ZAHVALA

Zahvaljujem se mentorju, viš. pred. dr. Damjanu Vavpotiču za pomoč in nasvete pri izbiri teme in izdelavi tega diplomskega dela.

Seveda ne smem pozabiti tudi na svojo družino, ki mi je bila v oporo skozi celoten potek študija.

Zahvaljujem se tudi vsem prijateljem in sošolcem, ki so mi vsa ta leta stali ob strani.

Kazalo vsebine:

Kazalo slik:

Seznam uporabljenih kratic in simbolov:

MVC	(ang. Model-View-Controller) arhitektura Model-Pogled-Krmilnik za ločitev domenske logike od predstavitve.
MOS	(ang. Mambo Open Source) predhodnik sistema Joomla.
CMS	(ang. Content Management System) sistem za upravljanje s spletnimi vsebinami.
PHP	(ang. PHP Hypertext Preprocessor) odprtokodni programski jezik za strežniške uporabe.
MySql	Je odprtokodna implementacija relacijske podatkovne baze.
CSS	(ang. Cascading Style Sheets) so podloge, predstavljene v obliki preprostega slogovnega jezika, ki skrbi za prezentacijo spletnih strani.
HTML	(ang. Hyper Text Markup Language) označevalni jezik za izdelavo spletnih strani.
XML	(ang. Extensible Markup Language) razširljiv označevalni jezik, podoben HTML-ju.
SEF	(ang. Search Engine Friendly) iskalnikom prijazne povezave, ki vsebujejo ključne besede, povezane s spletno stranjo.
PDF	(ang. Portable Document Format) odprt standard za izmenjavo elektronskih dokumentov (neodvisen od platforme).
URL	(ang. Uniform Resource Locator) je naslov spletnih strani v svetovnem spletu.
(X)HTML	(ang. Extensible HyperText Markup Language) je označevalni jezik, ki ima enak namen kot HTML, vendar je usklajen s sintakso XML.

POVZETEK

Diplomska naloga je predstavljena v dveh delih. V prvem, teoretičnem, delu diplomske naloge predstavimo sistem za upravljanje z vsebinami na spletu Joomla in primerjamo različne možnosti, ki so na voljo za razširitev funkcionalnosti sistema. Razlog odločitve za prav ta CMS sistem je preprost. Sistem se uporablja pri veliko postavitvah spletnih strani, saj nudi enostaven dostop do vsebine spletne strani tudi za nevešče uporabnike. Uporabniki se tako izognejo neprijetnostim, kot so vključevanje spletnega programerja ob vsakodnevni posodobitvi vsebine.

V drugem, praktičnem, delu naloge pa se posvetimo razvoju razširitve za opis in ocenjevanje restavracij. Najprej predstavimo problematiko in motiv za razvoj takšne razširitve, nato pa s pomočjo diagramskih tehnik UML pripravimo analizo in načrt razširitve. Na podlagi pripravljenega načrta izdelamo programsko kodo razširitve ter predstavimo njeno delovanje v okviru sistema Joomla. Ker je sistem odprtokodni, ima skoraj vsak z zadovoljivim znanjem možnost razvijati lastne razširitve za lastno ali širšo uporabo.

Pri razvoju razširitve smo upoštevali najnovejše smernice razvoja, ki naj bi se jih razvijalci držali. Ena izmed takih je npr. arhitektura model – pogled – krmilnik (model – view – controller ali krajše MVC) in je prav tako predstavljena v nadaljevanju dela. V diplomskem delu se osredotočamo predvsem na razvoj glavnih treh tipov razširitev, komponente, modula in vtičnikov, ter na pravilno pripravo teh razširitev po končanem razvoju za namestitev v sistem Joomla.

Ključne besede: sistem za upravljanje z vsebinami Joomla, razvoj razširitve, razširitev za opis in oceno restavracij, komponenta, modul, vtičnik

SUMMARY

The thesis is presented in two parts. The first, theoretical part, deals with the content management system on the Internet Joomla and describes how it works. We also compare different options that are available to extend the functionality of the system. Reason for decision for Joomla CMS system is simple. The system is widely used on a lot of webpages, it offers easy access to website content and even users without programming skills can easy manage the content of the website in Joomla.

In the second, practical part of the thesis, we focus on development of an extension for description and evaluation of restaurants in Joomla CMS. First, we present the problem and motivation for the development of such an extension, then we provide an analysis and plan of the extension with help of UML diagrams. Based on the diagrams we prepare programming code of the extension and present it in Joomla CMS. Because Joomla is open source software, virtually anyone with enough knowledge has the possibility to develop their own extensions for their own use.

We obtain latest recommendations in the development of this extension. One of these recommendations is Arhitecture Model – View – Controller (MVC), details are presented below. The thesis also addresses the development of three types of extensions, components, modules and plugins, and the proper preparation of such extensions, after developing to integrate them with installation to system Joomla.

Keywords: content management system, the development of extensions, extension for description and evaluation of restaurants, component, module, plugin

1. UVOD

Internet je najhitreje rastoči medij, ki omogoča izmenjavo informacij. Seveda pa je potrebno te informacije redno posodabljati in nadgrajevati. In ravno zato je nastala potreba po enostavnem urejanju spletnih vsebin, ki so dostopne vsakomur, ne le programerjem in spletnim razvijalcem.

Leta 2000 je podjetje Miro International Pvt Ltd. začelo z razvojem »Mambo Open Source« (MOS) sistema za enostavno urejanje vsebin na internetu. Avgusta 2005 so takratni projektni vodja ter še nekateri razvijalci ustvarili spletno stran, imenovano OpenSourceMatters.org z namenom širjenja informacij med uporabnike, razvijalce, izdelovalce spletnih strani in celotno skupnost na splošno. Razvili so novo vejo projekta Mambo in jo poimenovali Joomla! Gre za latinsko črkovanje besede »jumla«, ki v svahiliju pomeni »vsi skupaj« oziroma »kot celota«.

Joomla je torej sistem za upravljanje z vsebinami na spletu. Omogoča nam izgradnjo spletnih strani ter spletnih aplikacij, ki poleg vseh ostalih lastnosti vključuje tudi dobro možnost razvoja razširitev. Je odprtokodni sistem in tako prosto dosegljiv vsakomur za uporabo in razvoj razširitev. Sistem deluje pod verzijo 2 licence GNU/GPL.

Sistem za upravljanje z vsebino (CMS) je programska oprema, ki beleži vsak del vsebine na spletni strani, kot na primer knjižnica hrani knjige. Vsebina je lahko preprosto besedilo, fotografije, glasba, video posnetki, dokumenti in še veliko več. Prednost takega sistema je, da ne zahteva tehničnega znanja za upravljanje z njim.

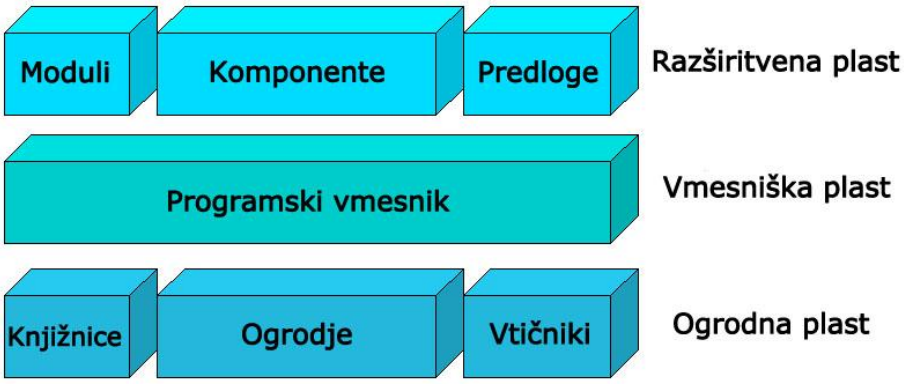
Za različico Joomla 1.0.0, ki je bila izdana kot prva različica sistema Joomla leta 2005, je bila leta 2008 izdana različica 1.5, ki je nadgradila uporabnost sistema na področju oblikovanja uporabniškega vmesnika in še bolj na področju logične strukture podatkovne baze.

Joomla je kodirana v jeziku PHP in podpira PHP 4.3 vse do najnovejše različice PHP 5. Joomla uporablja objektno usmerjeno razvojno tehniko in programsko razvojne standarde, kot je na primer arhitektura MVC (Model-Pogled-Kontroler), ki nam omogoča ločitev poslovne logike od prikaza vsebine. Podatke shranjuje v podatkovno bazo MySql. Vključene funkcije so med drugim tudi: predpomnenje, RSS, tiskanje posameznih strani, ankete, podpora za prilagoditev sistema za različne jezike in regionalne razlike …

V začetku leta 2011 je bila izdana Joomla 1.6, ki pa jo je po prehodu na šest-mesečni cikel izdajanja novih različic, julija leta 2011 nadomestila različica 1.7. V času pisanja te diplomske naloge je aktualna različica 2.5.4, a je že za september 2012 napovedana različica 3.0. Različica 2.4 je prva, ki deluje tudi z drugimi podatkovnimi bazami in ne samo z MySql.

Ker je obravnavana razširitev v tem diplomskem delu napisana za različico 1.5, se bomo največ posvečali tej še vedno precej razširjeni različici Joomle.

Namestitev sistema Joomla je preprosta. Potreben je le spletni strežnik, ki podpira PHP in MySql podatkovno bazo. Datoteke iz namestitvenega paketa razširimo v mapo na spletnem strežniku in sledimo navodilom preko spletnega brskalnika, kjer vpišemo tudi vse potrebne parametre za dostop do podatkovne baze in urejevalnika. Različica 1.5 je razdeljena na tri plasti, ki so prikazane na sliki 1:



Slika 1: Arhitektura Joomle

1 Vrhnjo razširitveno plast (Extension layer) sestavljajo razširitve ogrodja in njegovega vmesnika.

2 Srednja – vmesniška plast (Application layer) je sestavljena iz aplikacij, ki razširjajo osnovni razred Japplication.

Vključene so štiri aplikacije v distribuciji Joomla:

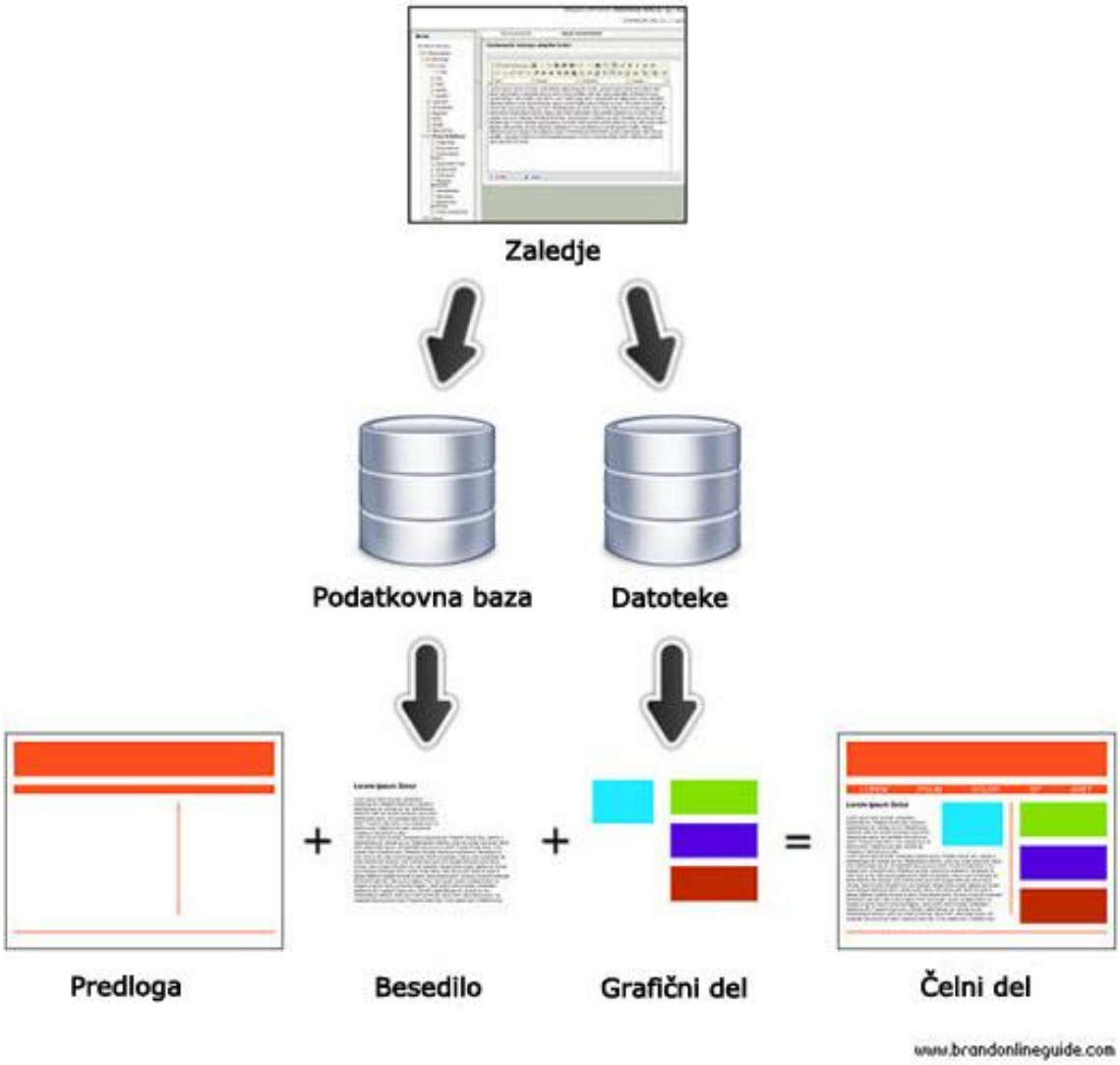
- Jinstallation, ki je odgovorna za pravilno namestitev sistema na spletni strežnik in je izbrisana po uspešni namestitvi (mapa installation).
- Jadministrator, ki je odgovorna za pravilni prikaz zalednega dela sistema za skrbnika strani oziroma administratorja.
- Jsite ima za razliko od Jadministrator-ja odgovornost pravilnega prikaza prednjega - čelnega dela sistema.
- XML-RPC podpira oddaljeno skrbništvo spletne strani Joomla.

3 Spodnjo – ogrodno plast pa sestavljajo ogrodje (Framework), knjižnice (Libraries), ki jih ogrodje potrebuje, ali pa so nameščene s strani uporabnikov, ter vtičniki (Plugins), ki razširjajo funkcionalnost, dosegljivo v ogrodju sistema.

2. PRINCIP DELOVANJA JOOMLE

2.1. Osnovna zgradba sistema CMS

Joomla ločuje vsebino spletne strani od poslovne logike. Vsa vsebina se nahaja v podatkovni bazi (MySql) na spletnem strežniku, podatki o tem, kako prikazati vsebino, pa so zbrani v predlogah. Ko brskalnik pokliče sistem z namenom prikazovanja informacij, Joomla pridobi vsebino svoje strani v podatkovni bazi in s pomočjo datotek, v katerih so informacije o tem, kako naj bodo podatki predstavljeni, oblikuje stran ter pošlje informacije nazaj brskalniku (Slika 2).



Slika 2: Delovanje sistema Joomla

Urejanje vsebine v zalednem delu (backend) sistema Joomla pomeni urejanje vsebine v podatkovni bazi. Predloga je skupina datotek, ki za sistem opredeljujejo način postavitve elementov na spletni strani in s tem določajo obliko predstavitve te vsebine v brskalniku. V Joomli predloga določa ogrodje, kam postaviti menije, vsebinska področja, glavo in nogo strani, kakšno grafično ozadje in katere oblike pisave naj ta uporabi, in podobno, kar je še lahko definirano v predlogi, kot je na primer poljubna Javascript koda. Predloga je sestavljena iz osnovne datoteke PHP, datoteke CSS, mape, ki vsebuje datoteke s slikami ter datoteke XML, ki vsebujejo informacije o predlogi, ki so potrebne pri namestitvi predloge v sistem. To pa je samo osnovna sestava. Vsebina se v brskalniku prikaže šele, ko je sistem oblikoval že celotno spletno stran.

Za Joomlo so na voljo že mnoge izdelane predloge, med njimi najdemo tudi veliko brezplačnih. Uporabnik lahko predlogo izdela sam ali pa jo da izdelati. Za izdelavo manj zahtevnih, a lastnih predlog obstajajo tudi preprosti programi za oblikovanje predlog, kot je npr. Artisteer, ki generira vse potrebne datoteke kakor tudi sam izgled strani.

2.2. Varnost v Joomli

S preko 8600 razširitvami, ki so na voljo za Joomlo, lahko celoten sistem postane precej ranljiv. Še posebno, če so te razširitve napisane površno in brez razmišljanja o varnosti. Za večino ranljivosti v Joomli so običajno krive razširitve. Rek, da je sistem tako varen, kot je njegov najšibkejši člen, tu še kako drži. Pri nameščanju razširitev je potrebna zadostna mera previdnosti, še posebej moramo paziti pri uporabi razširitev razvijalcev, ki niso uradni Joomla razvijalci. Dobro sliko nam da že razvijalčeva spletna stran, sploh če ima ta tudi forum, kjer lahko preverimo, če imajo uporabniki kakšne resnejše težave z razširitvijo. Pred uporabo novih, nepoznanih razširitev je te dobro namestiti na dvojnik sistema. Tam jih lahko dodobra stestiramo. Ustanovljena je tudi posebna skupina (Joomla Security Strike Team), ki se ukvarja s problemi na področju varnosti jedra Joomle. Poleg opravljanja revizij sistema preverjajo tudi vsako poročilo uporabnikov o morebitnih ranljivostih, ki pride do njih. Le tako se varnost sistema ohrani na najvišji možni ravni.

```
// no direct access
defined( '_JEXEC' ) or die( 'Restricted access' );
```

Slika 3: Prepovedan neposredni dostop do datoteke

Na sliki 3 vidimo primer stavka, ki preveri, če je datoteka klicana znotraj Joomla seje. To zaščiti našo stran pred neposrednim dostopom do PHP datotek z vpisom njihovega URL naslova.

Druga stvar, ki jo preverimo, je, ali so podatki, ki so bili uvoženi v to razširitev, »očiščeni«. Za to smo uporabili razred *JRequest*, ki je vgrajen v strukturi Joomle. Poleg tega lahko uporabljamo tudi funkcije, kot so *getInt*, in tako preverimo, da je edina vrednost, ki lahko prihaja iz *$id*, integer. V primeru neposrednega dostopa tega seveda ne moremo delati. Brez tega preverjanja puščamo razširitev ranljivo za tako imenovano »SQL injekcijo« (vstavljanje SQL). Ker je Joomla široko razširjen sistem za upravljanje s spletnimi vsebinami in je dnevno uporabljan po vsem svetu, je pogosto tarča napadov hekerjev, ki skušajo najti načine za vdor v sistem. Uporabnik Joomle mora v prvi vrsti sam poskrbeti za varnost, kjer je to mogoče.

Priporočilo razvijalcev je, da Joomlo redno posodabljamo na zadnjo različico, ki je na voljo. Ko je odkrita ranljivost v sistemu, lahko posodobitev pričakujemo v enem ali dveh dneh. Prednost posodobitev Joomle je ta, da ne spreminjajo podatkovne baze, ampak samo datoteke na strežniku in zato vsebina ostane nedotaknjena. Stare datoteke tako le preprosto prepišemo z novimi. Seveda pa je kljub temu zelo pomembna izdelava varnostnih kopij. Na našo srečo imamo na voljo dobro rešitev, zelo priljubljeno in nagrajeno komponento z imenom Akeeba Backup, s pomočjo katere lahko izdelujemo varnostne kopije celotnega sistema kot tudi posameznih delov sistema. S pomočjo nje je tudi prenos sistema na drugo gostovanje precej poenostavljen, in to z vključeno vsebino, vsemi drugimi razširitvami in predlogami spletne strani. Oglejmo si, kaj lahko za boljšo varnost Joomle storimo sami:

- **Izbira primernega gostovanja**
  Veliko napak in ranljivosti lahko povzročijo slabo vzdrževani strežniki ter razna deljena gostovanja (shared hosting). Če je tako deljeno gostovanje slabo, je lahko naša spletna stran napadena preko druge strani, ki si z nami deli gostovanje.

- **Uporaba .htaccess datoteke**
  Privzeto .htaccess datoteka ni nastavljena.Ta nam omogoča uporabo iskalnikom prijaznih spletnih naslovov (SEF) in tudi uporabo pravil za razne preusmerite na naši strani (Preusmeritve 301, napaka 404). Z dodajanjem spodnjih atributov v to datoteko pa blokiramo tudi nekaj pogostih ranljivosti:

  ##Začetek – Prepisna pravila za blokado nekaterih izkoriščevalskih programov # # Blokiraj kakršnekoli skripte ki poskušajo določiti mainframe vrednosti prek URL-ja
  RewriteCond %{QUERY_STRING} mainframe->[a-zA-Z_]{1,21}(=|%3D) [OR]
  # Blokiraj kakršnekoli skripte, ki poizkušajo s funkcijo base64_encode pošiljati ¬podatke« preko URLja

```
RewriteCond %{QUERY_STRING} base64_encode.*(.*) [OR]
# Blokiraj kakršnekoli skripte ki vsebujejo oznako < script> v URL-ju
RewriteCond %{QUERY_STRING} (<|%3C).*script.*(>|%3E) [NC,OR]
# Blokiraj kakršnekoli skripte ki poskušajo določiti PHP GLOBALS spremenljivke
prek URL-ja
RewriteCond %{QUERY_STRING} GLOBALS(=|[|%[0-9A-Z]{0,2}) [OR]
# Blokiraj skripte, ki poizkušajo spremeniti _REQUEST spremenljivko prek URL-ja
RewriteCond %{QUERY_STRING} _REQUEST(=|[|%[0-9A-Z]{0,2}) [OR]
# Pošlji vse blokirane zahteve na domačo stran z napako 403 Forbidden!
RewriteRule ^(.*)$ index.php [F,L]
##Konec -Rewrite Prepisna pravila za blokado nekaterih izkoriščevalskih programov#
```

- **Nastavitve pravic datotek na strežniku**
  Po končani namestitvi moramo vsem datotekam nastaviti pravice CHMOD na 644, mapam pa na 755. Izjema je datoteka configuration.php, kateri nastavimo pravice na 604.

- **Sprememba »jos_« predpone v podatkovni bazi**
  Veliko napadov SQL poskuša pridobiti podatke iz tabele, kjer so shranjeni podatki o uporabnikih, in s tem pridobiti uporabniško ime ter geslo uporabnikov tipa Super Administrator, ki imajo popoln nadzor nad sistemom. Sprememba predpone spremeni večino, če ne skoraj vsa vstavljanja SQL. Nove različice Joomle (od 1.7 naprej) imajo to podprto samodejno že med namestitvijo, saj nam vsakič izberejo drugo, naključno predpono podatkovne baze.

- **Odstranitev številke različice ali imena razširitve**
  Veliko ranljivosti se nahaja le v določenih različicah posameznih razširitev. Zato je lahko tudi prikaz različice v kodi razširitve kot npr. //Komponenta za oceno in opis restavracij v1.0 nevarnost za potencialni napad. Z odstranitvijo te številke je teže ugotoviti, katera različica razširitve je nameščena.

- **Redno posodabljanje Joomle in razširitev**
  Kot smo že omenili, je nujno potrebno vedno nameščati zadnje verzije sistema Joomla ter vseh nameščenih razširitev. Veliko prvotnih ranljivosti je v novejših različicah odpravljenih.

- **Izbris neuporabljenih datotek**
  Ko je nameščena razširitev, ki se ne uporablja, je dobra praksa, da se to razširitev odstrani in ne samo onemogoči. Samo onemogočena razširitev lahko preko svojih datotek še vedno predstavlja ranljivost sistema.

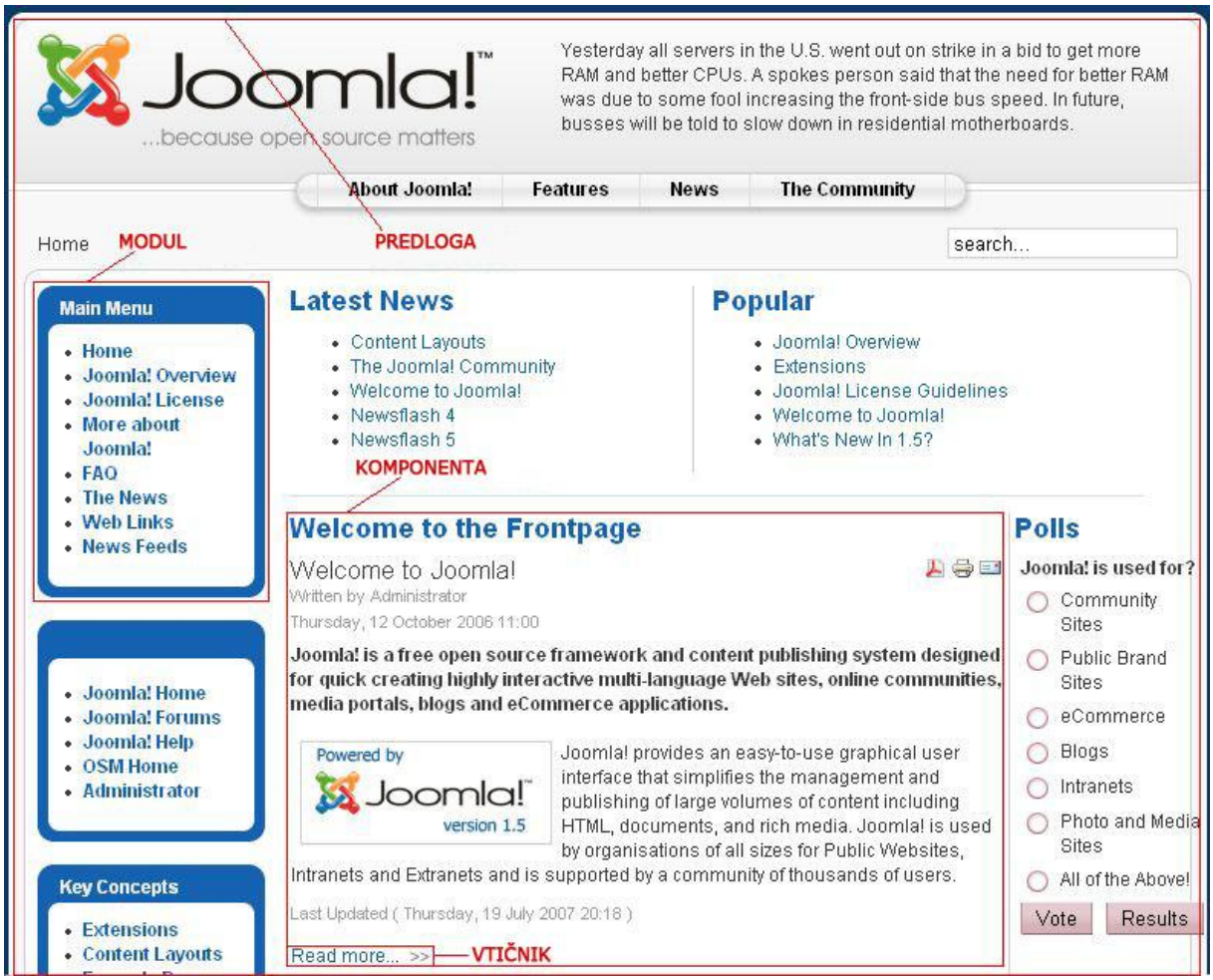
Seveda je za večjo mero varnosti, tako kot tudi povsod drugje, odvisna tudi pravilna izbira gesel uporabnikov. Gesla lahko izboljšamo že s preprosto zamenjavo določenih črk s številkami (uporaba ti. Leet pisave). Pomembno je tudi, da je geslo za dostop do podatkovne baze različno od gesla, ki ga uporabljamo za dostop do sistema. Vzdrževanje varnosti je stalen proces, ki ga je potrebno vedno imeti v mislih, in ni samo trenutno stanje.

2.3. Vrste razširitev v Joomli

Joomla ni namenjena samo upravljanju s članki (vsebino), ampak omogoča tudi uporabo veliko drugih kompleksnih aplikacij, ki jih integriramo vanjo. Z njimi povečamo funkcionalnost sistema. Mednje sodijo nakupovalne košarice, forumi, podpore socialnim omrežjem, podpora večjezičnim spletnim stranem, prikazovanje oglasov ali opisov itd. Prednost teh razširitev je, da vse delujejo znotraj Joomle, v eni podatkovni bazi, z eno predlogo in znotraj enega jedra. Ko napišemo razširitev za Joomlo, bo podedovala izgled in postavitev, ki je definirana v privzeti predlogi sistema. Vsak program, ki ga lahko napišemo v jeziku PHP, je potencialna razširitev, ki čaka na prilagoditev za Joomlo. Prednost razširitev je tudi, da jih je mogoče zlahka prenašati med sistemi. Ko jih pravilno sprogramiramo, jih bomo zlahka nameščali na druge Joomla namestitve na povsem drugih strežnikih, z drugačnimi predponami podatkovne baze itd. Namestitev lahko opravijo tudi manj vešči uporabniki, saj ne potrebujejo nobenega programerskega znanja, niti jim ni treba poznati podatkovne baze.

Koda Joomle je narejena za razširjanje in ne neposredno spreminjanje. Namesto spreminjanja jedrne kode je bolje napisati razširitev. Ko so izdane posodobitve za Joomlo, bo jedrna koda posodobljena (prepisana), a razširitve bodo ostale nedotaknjene. Joomlina koda omogoča razširitvam deljenje virov in včasih si razširitve med seboj izmenjujejo dejanja in podatke. Tak primer je tudi naša komponenta, ki nudi podatke vtičniku za iskanje po opisih restavracij ter modulu za prikaz opisa restavracije po meri.

Med razširitve (Slika 4) v Joomli sodijo komponente, moduli, vtičniki ter tudi jezikovni paketi in seveda predloge. Vse bomo na kratko predstavili v nadaljevanju.



Slika 4: Razširitve v Joomli

2.3.1. Komponente

Ključen tip razširitev v Joomli so komponente. Joomla je narejena tako, da na vsaki generirani strani naloži in izvaja samo eno komponento. Komponente imajo običajno tudi napredno zaledje, kjer so nam na voljo nastavitve in parametri komponente. V našem primeru komponente za opis in oceno restavracij lahko v zaledju preko orodne vrstice dodajamo, brišemo ali urejamo posamezen opis restavracije.

2.3.2. Moduli

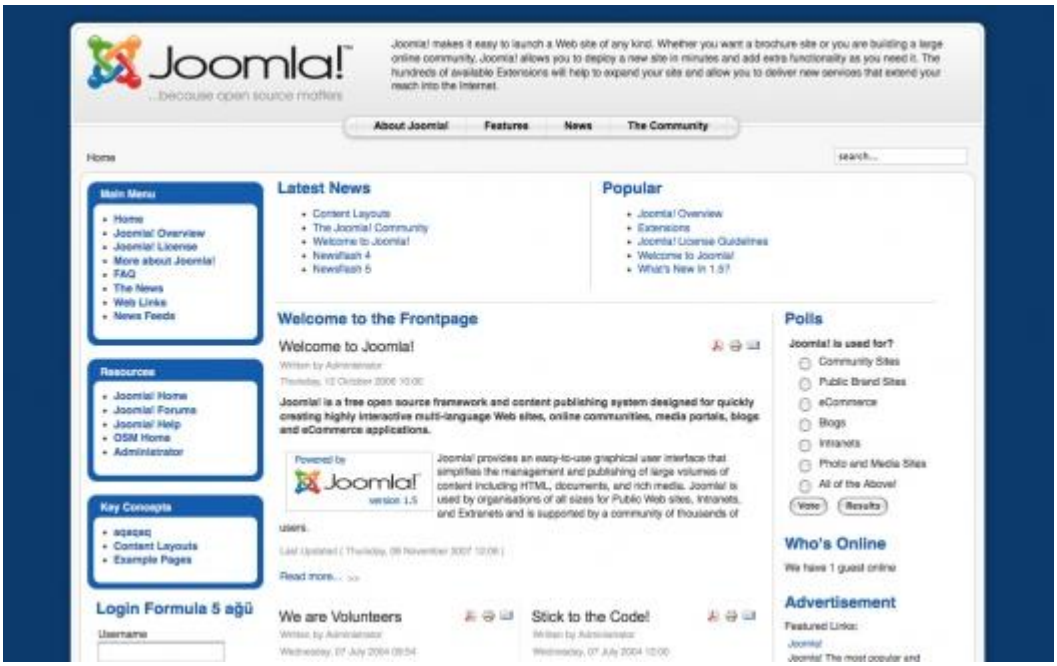
V nasprotju s komponentami imamo na eni strani lahko poljubno število modulov. Moduli običajno dopolnujejo vsebino komponent in služijo za prikaz stranskih vrstic ali menijev. Niso pa namenjeni za osrednjo predstavitev. V našem primeru je modul za prikaz skrajšanega opisa restavracij povezan s komponento za opis in oceno restavracije. Moduli so lahko tudi samostojno postavljeni, ne da bi bili s čim povezani. Lahko vsebujejo tudi le statično HTML ali navadno besedilo. Nastavitve v zaledju modulov so običajno bolj okrnjene in vključujejo le nekaj osnovnih nastavitev za prikaz modula.

2.3.3. Vtičniki

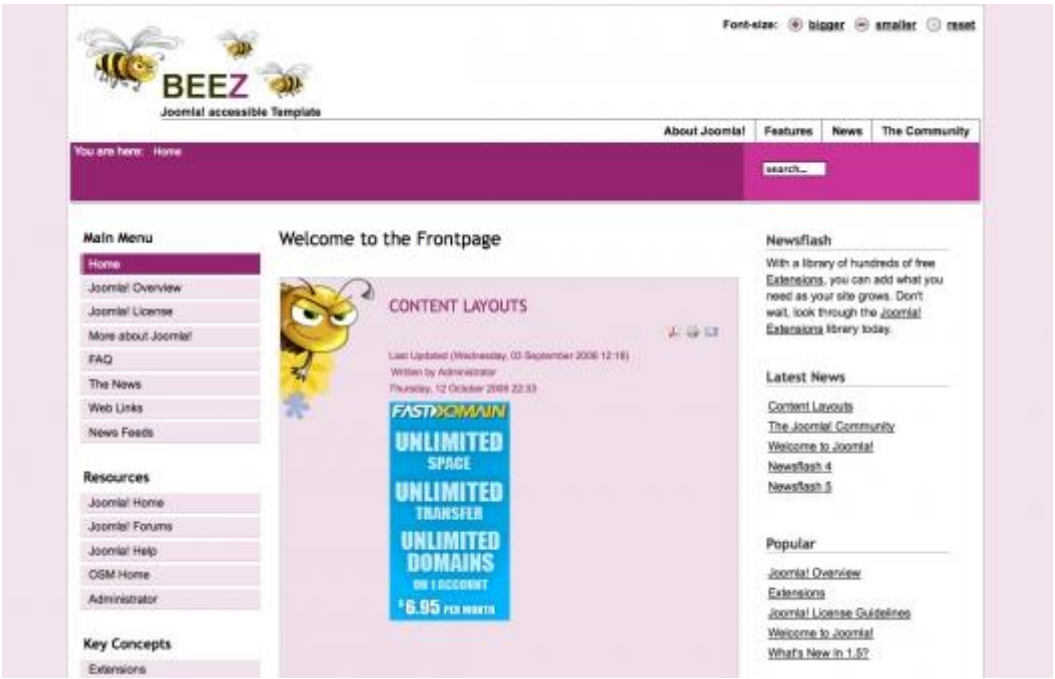
Kadar potrebujemo del kode, ki deluje na celotni spletni strani, skozi vse komponente in module, je to kodo najbolje implementirati kot vtičnik. V starejših različicah Joomle so se vtičniki imenovali Mamboti. Vtičniki imajo običajno nalogo oblikovanja izhoda določene komponente ali modula, ko je stran v zaključni fazi pred prikazom. Recimo poudarjanje vsebine z barvnim ozadjem, okno komentarja določenega članka … V našem primeru je eden od vtičnikov namenjen iskanju imena restavracije v poljubnem besedilu v člankih. Ko najde ime restavracije, ki je že opisana v komponenti za opis in oceno restavracij, ga spremeni v povezavo (link) do tega opisa. Nadzor vtičnika v zaledju je precej podoben nadzoru modulov, saj je tudi ta omejen le na osnovne parametre.

2.3.4. Predloge

Predloge v sistemu Joomla določajo obliko spletne strani. Z več različnimi predlogami lahko spremenimo videz Joomline spletne strani. V predlogi so tudi določena mesta, kjer so prostori za prikaz komponente in modulov. Stil predlog je definiran v eni ali več CSS datotekah. Ena predloga ima lahko tudi več variacij z različnimi barvnimi shemami. Njihovo oblikovanje je relativno enostavno in nudijo veliko fleksibilnosti pri določanju načina prikaza spletne strani. Na sliki 5 in 6 je prikazana ista vsebina spletne strani v dveh različnih predlogah.



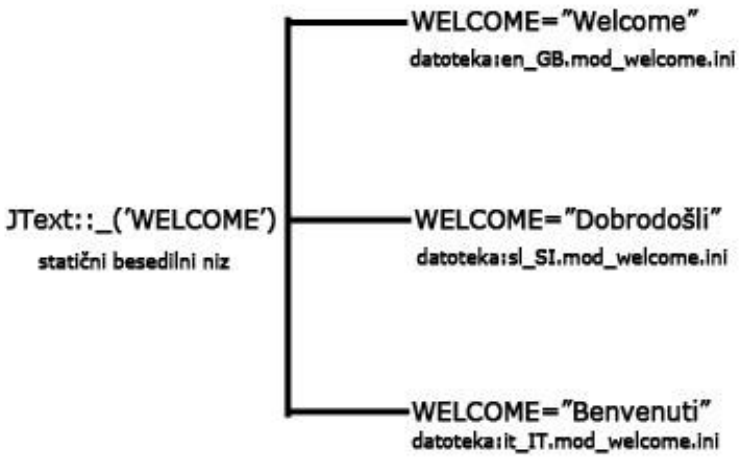
Slika 5: Privzeta vsebina ob namestitvi Joomle, privzeta predloga



Slika 6 Privzeta vsebina ob namestitvi Joomle, Beez predloga

2.3.5. Jezikovni paketi

Najpreprostejša razširitev v Joomli so jezikovni paketi. Shranjeni so kot jedrni paket ali kot razširitveni paket. Gre za datoteke, ki so sestavljene iz parov ključ – vrednost. Tak par zagotavlja prevod ključa, ki je statični tekstovni niz, definiran v izvorni kodi. Jezikovni paketi so namenjeni obem stranem, zaledju ter sprednjemu čelnemu delu, ki ga vidijo obiskovalci strani. Vsebujejo lahko tudi XML datoteko, ki definira jezik in tekstovno obliko za uporabo generiranja vsebine v PDF obliki. Ta možnost je sicer v novejših različicah ukinjena, ker zanjo ni bilo dovolj zanimanja.



Slika 7: Ključ in tri različne vrednosti zanj

3. RAZVOJ RAZŠIRITEV

3.1. Razvoj komponente za opis in ocenjevanje restavracij

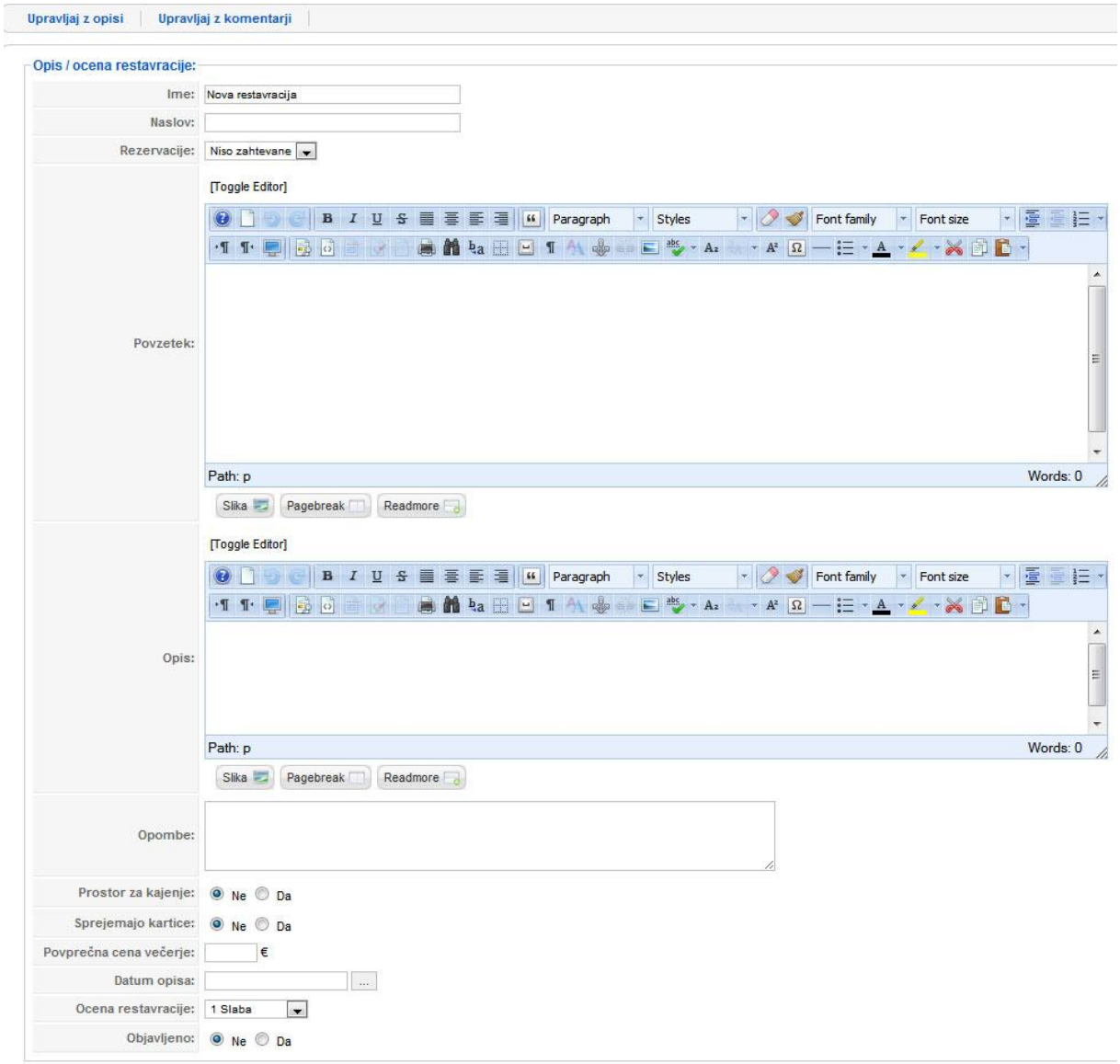
Motiv za razvoj te komponente je nastal ob pojavu potrebe po vodenju seznama vseh bližnjih restavracij za lokalni TIC (Turistično-informacijski center). Preko namenske komponente bo vnos opisa ter ocene posamezne restavracije ali gostilne lažji, kot če bi za vsako posebej pisali svoj članek. Poleg tega komponenta omogoča tudi uporabniške komentarje, katere je mogoče v zaledju tudi urejati ali ob neprimernosti izbrisati. Vzrok za odločitev za razvoj lastne komponente je bil tudi ta, da se tako komponento v prihodnosti laže nadgrajuje, razširja in prilagaja lastnim potrebam, saj nismo omejeni z avtorskimi pravicami ali plačilom licence. Naloga komponente je torej ta, da v podatkovno bazo shrani vsak opis restavracije ter pripadajoče komentarje uporabnikov, če ti obstajajo. Administrator oz. registrirani uporabnik spletne strani ima preko prijave v zaledje popoln nadzor nad vsemi opisi ter komentarji.

3.1.1. Zajem zahtev za komponento

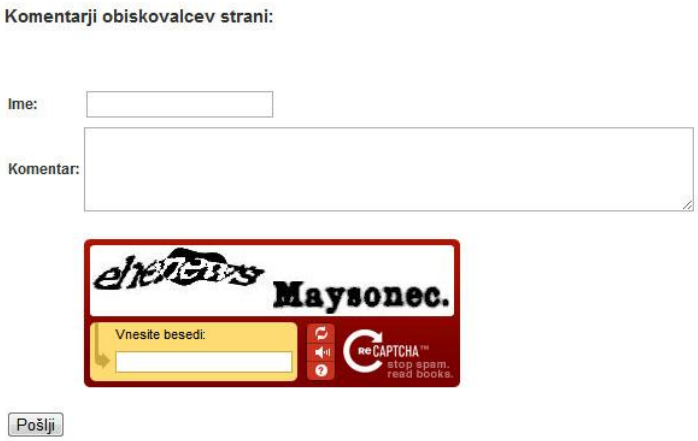
Za komponento za opis in ocenjevanje restavracij se zahteva, da v zaledju preko prijave v sistem omogoča vnos, urejanje ter brisanje opisov restavracij. Dostop je privzeto omogočen administratorju spletne strani ter registriranemu uporabniku (uredniku). Zaradi lažje navigacije mora komponenta nuditi seznam vseh opisov, ki vsebuje samo osnovne podatke vsakega opisa (ime, naslov, rezervacije, sprejemanje kreditnih kartic, povprečno ceno, oceno ter status objave). Ob kliku na posamezen opis se ta odpre v urejevalnem načinu, kjer ga administrator lahko popravi in nazaj shrani. Ob kliku na gumb »Nov« v orodni vrstici se odpre prazno vnosno polje (Slika 8), ki ima za ime privzeto vneseno Nova restavracija, da se vnos ne shrani prazen (brez imena). Administrator lahko briše posamezni vnos ali več vnosov hkrati. Enako določa tudi status objave vsakega vnosa posebej (objavljen ali neobjavljen). Če je opis neobjavljen, ni viden v čelnem delu spletne strani. Poleg administratorja ima do zaledja dostop tudi uporabnik urednik (registrirani uporabnik).

V ločenem meniju imamo na voljo tudi urejanje ter brisanje komentarjev. Komentarji so prikazani v seznamu, ki vsebuje ime opisa, kateremu komentar pripada, ime komentatorja, datum komentarja ter besedilo komentarja. Ob kliku na posamezen komentar se ta odpre in ga lahko urejamo. Komentarje lahko brišemo enako kot opise restavracij, le da nas mora pred izbrisom na to opozoriti obvestilno okno, ki ga lahko potrdimo ali prekličemo.

Uporabniki imajo v prikazu na spletni strani (v čelnem delu) prikazan seznam vseh objavljenih opisov restavracij. S klikom na posamezni opis se ta prikaže v podrobnem načinu. Ocena restavracije mora biti prikazana z zvezdicami od 1 do 5. Pod njo je vsem obiskovalcem omogočena možnost komentiranja opisa preko vpisa komentarja. Vnosno polje za komentar je prikazano na sliki 9. Uporabniki morajo pred poslanim komentarjem tudi prepisati varnostno Captcha kodo, ki ima namen preprečiti spam komentarje. Vpisani komentarji so nato prikazani pod komentiranim opisom restavracije. Komentiranje je dovoljeno tudi prijavljenemu uporabniku in administratorju, na primer kot odgovor na kakšen komentar. Polje »Ime« se ob prijavljenem uporabniku izpolni samodejno iz tabele Ime v uporabniški tabeli sistema Joomla. Komponenta mora podpirati tudi SEF povezave (iskalnikom prijazne povezave).

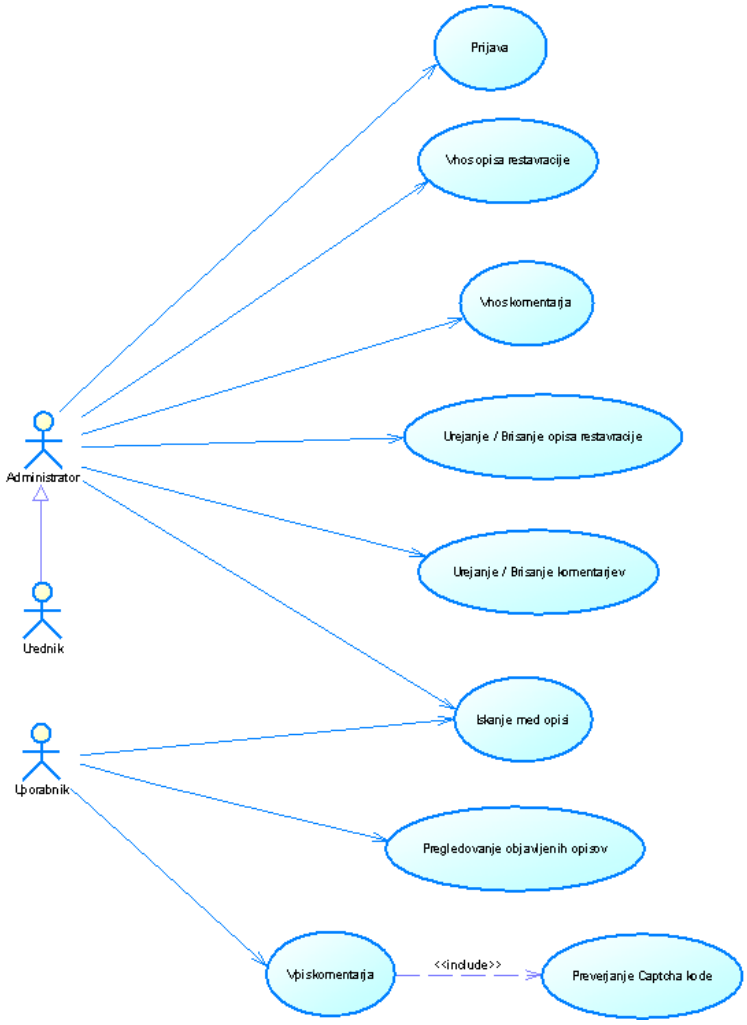


Slika 8: Obrazec za vnos opisa restavracije



Slika 9: Vnosno polje za vpis komentarja

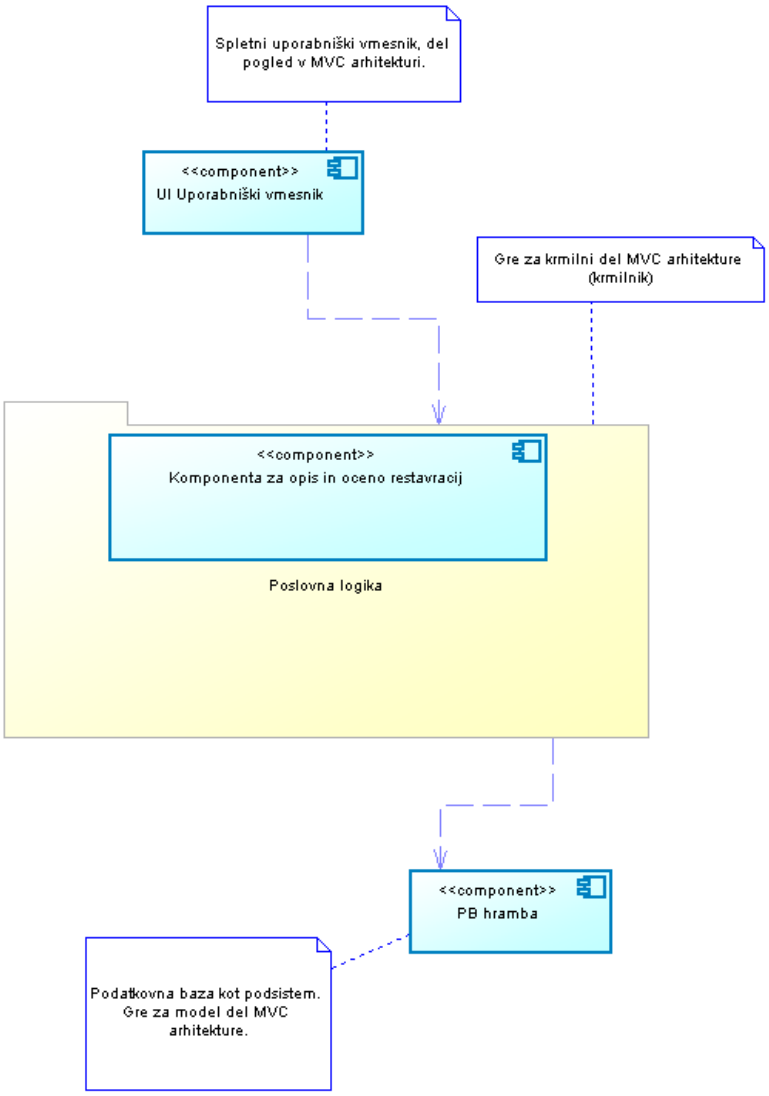
3.1.2. Diagram primerov uporabe



Slika 10: Diagram primerov uporabe

Na sliki 10 vidimo primere uporabe komponente za opis in ocenjevanje restavracij. Administrator se prijavi v zaledje, kjer lahko vnaša nove opise restavracij, ureja ter briše že vpisane opise restavracij, lahko ureja in briše tudi vnesene komentarje ali pa preko čelnega dela sam poda komentar kot odgovor na neki predhodni komentar obiskovalca. Lahko tudi išče med opisi restavracij. Poleg administratorja ima dostop do zaledja tudi registriran uporabnik (urednik), ki ima na voljo enake funkcionalnosti kot administrator. Uporabniki oz. obiskovalci spletne strani imajo na voljo le iskanje med opisi restavracij, vnos komentarjev na objavljene opise ter ročno pregledovanje teh opisov. Ob vsakem vnosu komentarja se preveri tudi pravilnost Captcha kode.

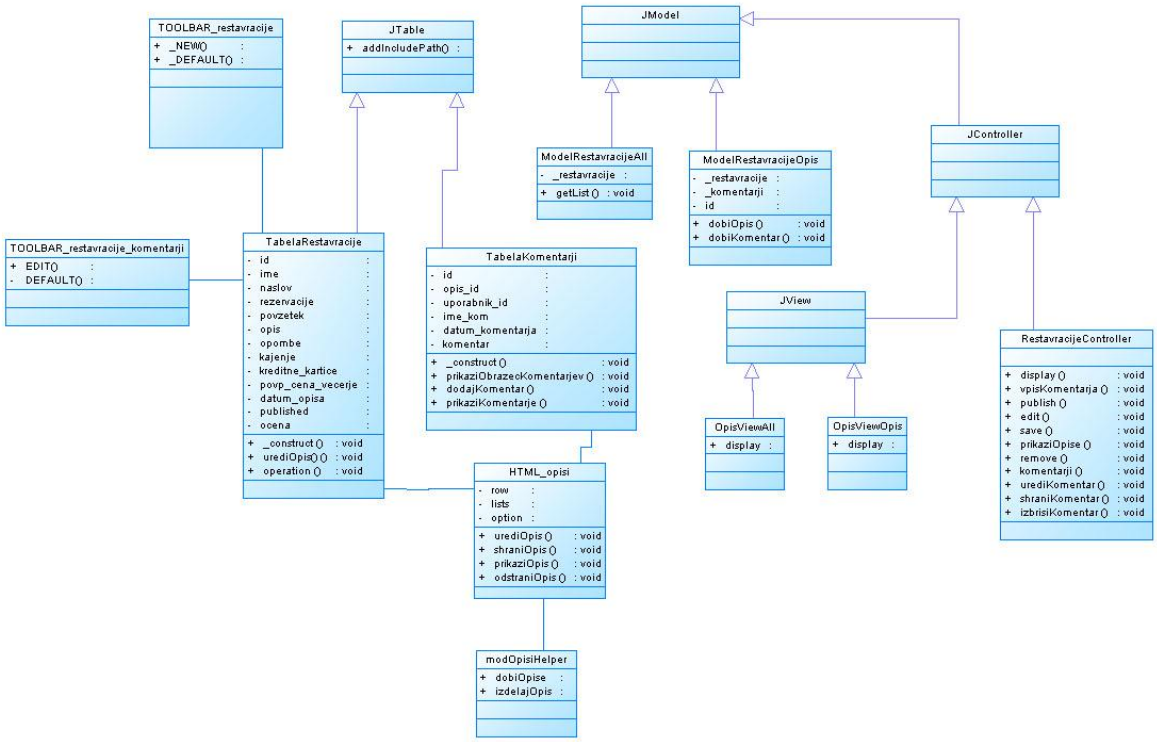
3.1.3. Komponentni diagram



Slika 11: Komponentni diagram

Na sliki 11 je prikazan komponentni diagram komponente za opis in ocenjevanje restavracij. Komponenta je razvita v MVC arhitekturi, zato je uporabniški vmesnik, ki je del pogleda v MVC arhitekturi, ločen od poslovne logike (krmilnega dela MVC arhitekture) in od podatkovne baze, ki predstavlja model del v MVC arhitekturi.

3.1.4. Razredni diagram



Slika 12: Razredni diagram

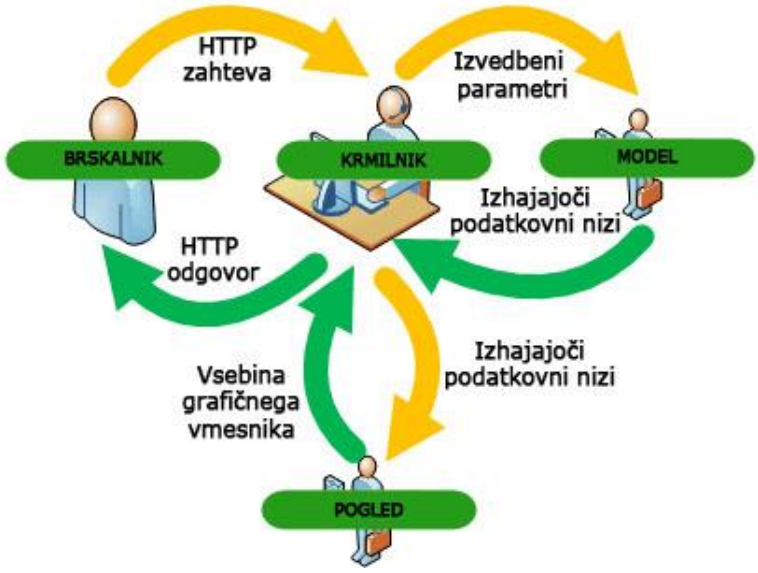
Na sliki 12 je prikazan razredni diagram komponente za opis in ocenjevanje restavracij. Gre za diagram, ki opisuje zgradbo sistema s prikazom sistemskih razredov, njihovih atributov, operacij (ali metod) in razmerja med razredi.

Razred *TOOLBAR_restavracije* je namenjen implementaciji gumbov za dodajanje, urejanje ter status objave posameznega opisa v zaledju. Podobno je tudi razred *TOOLBAR_restavracije_komentarji* namenjen gumbom za urejanje ter brisanje komentarjev v zaledju. Razred *TabelaRestavracije* razširja razred *JTable*, kateri omogoča metode za ustvarjanje, branje, posodabljanje in brisanje vnosov iz ene tabele v podatkovni bazi in ga tako prilagodi za naše potrebe. S tem se izognemo pisanju metod za vsako poizvedbo posebej.

Enako tudi razred *TabelaKomentarji* razširja razred *JTable* za potrebe shranjevanja, urejanja ter brisanja komentarjev. Razred *ModelRestavracijaAll* je model (v MVC arhitekturi), ki razširja razred *JMode*l. Metoda *getList()* preveri, če je bil seznam opisov že naložen, drugače opravi poizvedo za pridobitev seznama objavljenih opisov iz podatkovne baze. Razred *ModelRestavracijeOpis* je model za posamezne opise restavracij. Gre za enak koncept, kot pri razredu *ModelRestavracijeAll*, le da namesto nalaganja seznama vseh opisov, metoda *dobiOpis()* naloži le posamezno vrstico iz podatkovne baze. Metoda *dobiKomentar()* je skoraj enaka metodi *getList()*, le da opravi poizvedbo v tabeli *restavracije_komentarji*. Razreda *OpisViewAll* in *OpisViewOpis* razširjata razred *JView* in sta pogled del MVC arhitekture, ker skrbita za prikaz vseh opisov (OpisViewAll) ter za prikaz posameznega opisa restavracije (OpisViewOpis). Razred *RestavracijaController* (krmilnik) razširja razred *JController* in služi upravljanju logičnega toka komponente.

3.1.5. Arhitektura MVC (Model-Pogled-Krmilnik)

MVC je programska arhitektura, ki jo uporabljamo z namenom boljšega organiziranja programske kode na način, ki loči domensko logiko aplikacije (izmenjavo podatkov med podatkovno bazo in uporabnikom) od vnosa in predstavitve podatkov. Prednost tega pristopa je ta, da se celotna logika aplikacije združi v en del in je tako ni treba reprogramirati vsakič, ko se spremeni vmesnik, ki uporablja podatke, in obratno. Gre za predvidljiv način nadzora logičnega toka v aplikaciji. Ločimo tri glavne dele MVC arhitekture, katerih potek je prikazan tudi na sliki 13.



Slika 13: Potek MVC arhitekture

Model

Model je domensko specifična predstavitev podatkov, na podlagi katerih aplikacija deluje. Upravlja podatke in o spremembi svojega stanja obvešča svoje pridružene poglede. Domenska logika v modelu spreminja podatke v informacije. V našem primeru komponente za opis in ocenjevanje restavracij imamo dva modela. Eden služi za pridobivanje seznama vseh objavljenih opisov, drugi pa za pridobivanje informacij opisa posamezne restavracije. Modeli se uporabljajo za definiranje različnih načinov, s katerimi dostopamo do podatkov.

Pogled (View)

Pogled pridobi podatke iz modela in jih pošlje v predlogo, ta pa jih predstavi uporabniku na način, ki je primeren za želeno interakcijo. Pogled nikoli ne spreminja podatkov ali ne vpliva nanje. Ima le nalogo pridobitve podatkov preko krmilnika ter njihov prikaz na želeni način. V primeru spletnih aplikacij je pogled generirana HTML koda, ki je poslana brskalniku z namenom prikaza uporabniku na zaslonu.

Tako kot pri modelu naše razširitve smo ustvarili tudi različne poglede za predstavitev opisov restavracij. En pogled je namenjen predstavitvi seznama vseh objavljenih opisov, drugi pa za prikaz posameznega opisa restavracije.

V pogledih so omogočene tudi različne maske. Za te maske ni nikakršnega predpisa o poimenovanju. Običajno je dobra praksa kreiranje ene maske za en pogled, ni pa to nujno. Tako ohranimo enostavnost izvajanja kode.

Krmilnik (Controller)

Krmilnik je zelo pomemben del MVC arhitekture, saj se odziva na uporabniška dejanja. V primeru spletnih aplikacij je uporabniška zahteva običajno zahteva po neki določeni strani. Krmilnik bo ugotovil vrsto zahtevka s strani uporabnika (iz URL naslova) in temu primerno ukrepal s klicem pravega modela, ki bo nato pridobil zahtevane podatke ter jih oblikovane poslal naprej pravemu pogledu. Ta bo te podatke na podlagi predloge tudi prikazal. V Joomli je MVC implementiran z uporabo treh razredov: JModel, JView in JController.

3.2. Razvoj modula za komponento

Modul je v osnovni obliki zgrajen iz dveh datotek. To sta konfiguracijska datoteka (XML) in krmilniške datoteke (PHP). Datoteka XML vsebuje splošne informacije o modulu (kako bo ta predstavljen v zalednem delu upravljalca modulov) kakor tudi parametre modula, ki jih lahko vključimo za izboljšavo videza ali funkcionalnosti modula, krmilniška datoteka pa zagotavlja nadzor nad logiko modula.

Enako kot pri razvoju komponente lahko tudi pri razvoju modula izkoristimo arhitekturo MVC. V ta namen dodamo datoteki PHP še datoteko helper.php, v kateri se izvajajo vse operacije in dostop do podatkov. Tako ločimo logiko od same predstavitve, katero zapišemo v datoteko default.php, v mapi tmpl/. Ta datoteka vsebuje (X)HTML kodo za predstavitev modula v sprednjem (čelnem) delu.

Prednost takega datotečnega sistem je tudi ta, da je tako omogočeno, da je HTML oz. predstavitev zlahka povožena s strani katerekoli predloge za sistem Joomla. S tem podpremo optimalni prikaz modula na kateri koli spletni strani.

Pri poimenovanju modulov moramo paziti, da modul nima enakega imena kot neki drugi že nameščeni modul. Drugače se tak modul ne bo namestil pravilno. Dobra praksa pri poimenovanju modulov je ta, da pred ime dodamo predpono »mod_«. V našem primeru je modul poimenovan kot »mod_ restavracije«.

Enako kot v komponento je tudi v modul mogoče vključiti jezikovno datoteko (npr. en-GB.mod_restavracije.ini za angleški jezik). Ta se lahko s pomočjo konfiguracijske datoteke namesti v jezikovni del sistema. S tem podpremo večjezičnost našega modula. Jezikovno datoteko lahko doda tudi vsak uporabnik sam in ne nujno razvijalec.

Ker se je pokazala želja po skrajšanem prikazu nekaj opisov restavracij, ki vsebuje le ime ter povzetek opisa, smo to zahtevo najlepše rešili s pomočjo modula. Tega lahko prikažemo na poljubnih predvidenih mestih na spletni strani, in to celo večkrat.

3.2.1. Postopek razvoja modula

Tudi pri modulu je vstopna točka modula krmilniška datoteka PHP (Slika 14). Ta pridobiva potrebne parametre, nastavljene s strani uporabnika Joomle, ter jih podaja logičnemu delu, ki izvaja operacije in upravlja podatke. Metode logičnega dela vračajo rezultate, ki jih ta datoteka hrani za predstavitev.

```php
<?php
defined('_JEXEC') or die('Restricted access');

require(dirname(__FILE__).DS.'helper.php');
$random = $params->get('random', 0);
$style = $params->get('style','default');

    if($random)
        {
            $list = modOpisiHelper::dobiNakljucniOpis();
        }
    else
        {
            $list = modOpisiHelper::dobiOpise($params);
        }

require(JModuleHelper::getLayoutPath('mod_restavracije', $style));

?>
```

Slika 14: Krmilniška koda modula

V naš modul smo vključili še parameter za prikaz naključnih opisov v modulu. V zaledju modula lahko izberemo, ali se opisi prikazujejo naključno ali ne, ter koliko opisov se prikaže v modulu. Izdelali smo tudi dva različna pogleda. Eden je klasični prikaz, drugi pa vsebuje prikaz z alinejami. Vrsto prikaza prav tako izberemo v zaledju modula. Privzeto je nastavljen normalen prikaz zadnjega vpisanega opisa restavracije.

Konfiguracijska datoteka vsebuje v svojem prvem delu predstavitev podatkov o modulu ter razvijalcu. V drugem delu so informacije o datotečnem sistemu modula, jezikovnih izbirah ter parametri, ki so potrebni, da modul deluje po željah uporabnika. Te parametre lahko uporabnik kasneje spreminja v nastavitvah modula v zaledju. Na sliki 15 je prikazana koda konfiguracijske datoteke za skrajšan prikaz opisa restavracij preko modula.

```xml
<?xml version="1.0" encoding="utf-8"?>
<install type="module" version="1.5">
    <name>Opisi restavracij</name>
    <author>Matej Božič</author>
    <creationDate>Marec 2012</creationDate>
    <copyright>(C) 2012</copyright>
    <license>Commercial</license>
    <authorEmail>matej.bozic@gmail.com</authorEmail>
    <authorUrl>www.bozic17.blogspot.com</authorUrl>
    <version>1.0</version>
    <description>Modul za predstavitev opisov restavracij.</description>
    <files>
        <filename module="mod_reviews">mod_restavracije.php</filename>
        <filename>helper.php</filename>
        <filename>tmpl/_restavracije.php</filename>
        <filename>tmpl/bulleted.php</filename>
        <filename>tmpl/default.php</filename>
    </files>
    <params>
    <param name="random" type="radio" default="0" label="Naključno" description="Opise prikazuj v naključnem vrstnem redu">
    <option value="0">Ne</option>
    <option value="1">Da</option>
    </param>
    <param name="@spacer" type="spacer" default="" label="" description="" />
    <param name="items" type="text" default="1" label="Prikaži #" description="Število prikazanih opisov" />
    <param name="style" type="list" default="default" label="Stil prikaza" description="Uporabljeni stil za prikaz opisov">
        <option value="default">Privzet</option>
        <option value="bulleted">Alineje</option>
    </param>
    </params>
</install>
```

Slika 15: XML konfiguracijska datoteka modula

Logiko modula predstavlja, kot smo že omenili, razred helper.php (Slika 16), ki vsebuje metodo za pridobivanje zadnjih vpisanih opisov restavracij, metodo za izdelavo posameznega opisa ter metodo za pridobitev naključno izbranega opisa.

```php
<?php
defined('_JEXEC') or die('Restricted access');
class modOpisiHelper
    {
        function dobiOpise(&$params)
        {
            $items = $params->get('items', 1);
            $db =& JFactory::getDBO();
            $query = "SELECT id, ime, povzetek FROM #__opisi_restavracij WHERE published = '1' ORDER BY
datum_opisa DESC";
            $db->setQuery( $query, 0, $items );
            $rows = $db->loadObjectList();
            return $rows;
        }

        function izdelajOpis(&$restavracije, &$params)
            {
                $link = JRoute::_("index.php?option=com_restavracije&view=opis&id=" .$restavracije->id);
                require(JModuleHelper::getLayoutPath ('mod_restavracije', '_restavracije'));
            }

        function dobiNakljucniOpis()
            {
                $db =& JFactory::getDBO();
                $query = "SELECT id, ime, povzetek FROM #__opisi_restavracij";
                $db->setQuery( $query );
                $rows = $db->loadObjectList();
                $i = rand(0, count($rows) - 1 );
                $row = array( $rows[$i] );
                return $row;
            }
    }
?>
```

Slika 16: Razred helper.php z metodami

Ker je bistvo arhitekture MVC ločitev logike od predstavitve, moramo dodati še predlogo, v kateri je datoteka, ki vsebuje informacije o predstavitvi modula v sprednjem delu sistema. Ta del je enakovreden pogledu v komponenti. Enako lahko spreminjamo maske, zato se v ta namen ustvari mapa tmpl/, ki vsebuje te maske. Privzeta maska se poimenuje kar »default.php«, njena koda pa je sestavljena kot mešanica kode PHP in HTML. V našem primeru imamo dve maski, default.php (Slika 17) ter bulleted.php za prikaz opisov z alinejami. Poleg teh dveh imamo tudi masko _restavracije, ki pa je namenjena interni rabi in ni na voljo uporabniku v zaledju modula.

```php
<?php
defined('_JEXEC') or die('Restricted access');

foreach ($list as $restavracije){
modOpisiHelper::izdelajOpis($restavracije, $params);
}
?>
```

Slika 17: Maska default.php za prikaz opisov v modulu

3.3. Razvoj vtičnikov za komponento

V Joomli so vtičniki implementirani s pomočjo razreda JPlugin. Tako jedro Joomle kot nameščena razširitev, razvita s strani tretje osebe, lahko sprožita dogodek, na katerega se vtičniki odzovejo. Implementacija vtičnika temelji na principu opazovalca, ki je predstavljen z razredom JPlugin, in opazovanega, ki je predstavljen z razredom JEventDispatcher. Vtičniki nudijo način izvajanja dela kode, ko se pojavijo določeni dogodki. Med drugim lahko vtičnike uporabljamo za zagon HTML urejevalnikov, izvajanje iskanj, predstavitev vsebine in prijavo uporabnikov v več sistemov hkrati. Vtičniki lahko sodelujejo s komponentami in moduli, ne da bi spreminjali njihove izvorne kode. V našem primeru trije vtičniki sodelujejo s komponento za opis in oceno restavracij.

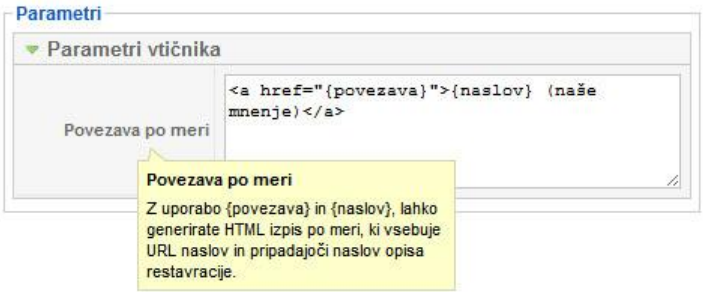
3.3.1. Vtičnik za ustvarjanje povezave iz imena restavracije

Naloga tega vtičnika je samodejno ustvarjanje povezave ob zaznavi imena restavracije v besedilu članka. S klikom na to povezavo nas preusmeri na podroben opis restavracije s tem imenom.

Uporabili smo funkcijo *registerEvent()* objekta *$mainframe* za dodelitev vtičnika dogodku *onPrepareContent.* Ko Joomla naloži vsebino iz podatkovne baze, se sprožijo vse funkcije, dodeljene *onPrepareContent* dogodku.

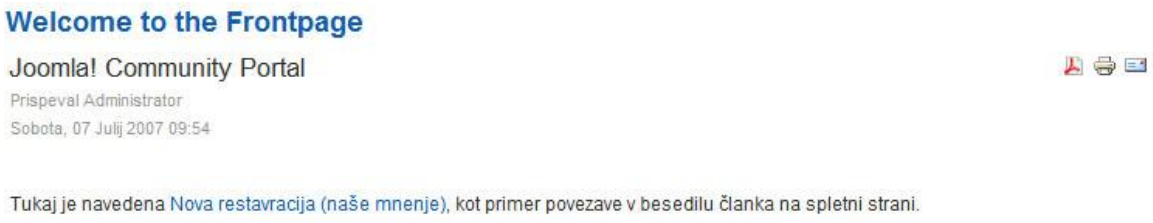
Izdelamo dve tabeli, eno z vzorcem, ki ga iščemo, in eno z nizom za zamenjavo. Metoda contentRestavracije_izdelajLink nam na podlagi pridobljenega id-ja izdela povezavo.

Vtičnik preko parametra v zaledju (Slika 18) podpira tudi dodajanje besedila po meri povezavam do opisov restavracij, tako da lahko bralce opozorimo, kam jih ta povezava preusmeri. Omogoča nam tudi dodatne parametre povezave, kot je na primer odpiranje strani v novem oknu ali zavihku, in podobno.



Slika 18: Parametri vtičnika za ustvarjanje povezave iz imena opisa restavracije

Prikaz delovanja vtičnika za ustvarjanje povezave iz imena restavracije, s parametri s slike 18, je prikazan na sliki 19.



Slika 19: Primer delovanja vtičnika s parametri s slike 18

3.3.2. Vtičnik za prikaz bistvenih informacij opisa restavracije

Ta vtičnik prikaže samo bistvene informacije posameznega opisa restavracije. Za njegovo uporabo moramo vpisati *restavracijeinfo* ter ime restavracije v zavite oklepaje. Na primer {restavracijeinfo Nova restavracija}.

Zakaj zaviti oklepaji ?

Veliko jedrnih vtičnikov uporablja zavite oklepaje za označitev »značk vtičnikov« v vsebini spletne strani, da ti niso zamešani s HTML-jem ali XML-jem. Te značke lahko relativno enostavno zaznamo v vsebini z metodami za regularne izraze.

Enako kot pri prvem vtičniku tudi temu registriramo metodo *pluginRestavracijeInfo* (Slika 20), da se sproži ob dogodku *onPrepareContent:*

```
function pluginRestavracijeInfo ( &$row, &$params )
{
    $plugin =& JPluginHelper::getPlugin('content', 'restavracijeinfo');
    $pluginParams = new JParameter( $plugin->params );
    preg_match_all('/\{restavracijeinfo (.*)\}/U',$row->text, $matches);

    foreach( $matches[1] as $name )
    {
        $restavracije = contentRestavracijeInfo_dobiOpisPoImenu($name);
        $html = contentRestavracijeInfo_izdelajHTML($restavracije, $pluginParams);
        $row->text = str_replace("{restavracijeinfo $name}", $html, $row->text);
    }
    return true;
}
```

Slika 20: Metoda vtičnika pluginRestavracijeInfo z detekcijo značke v besedilu

Na sliki 21 vidimo primer delovanja vtičnika v čelnem delu, ki nam prikaže samo določene bistvene elemente opisa restavracije (Ime restavracije, naslov, Cenovni razpon, Rezervacije, Kreditne kartice, Kadilnica ter ocena z zvezdicami).

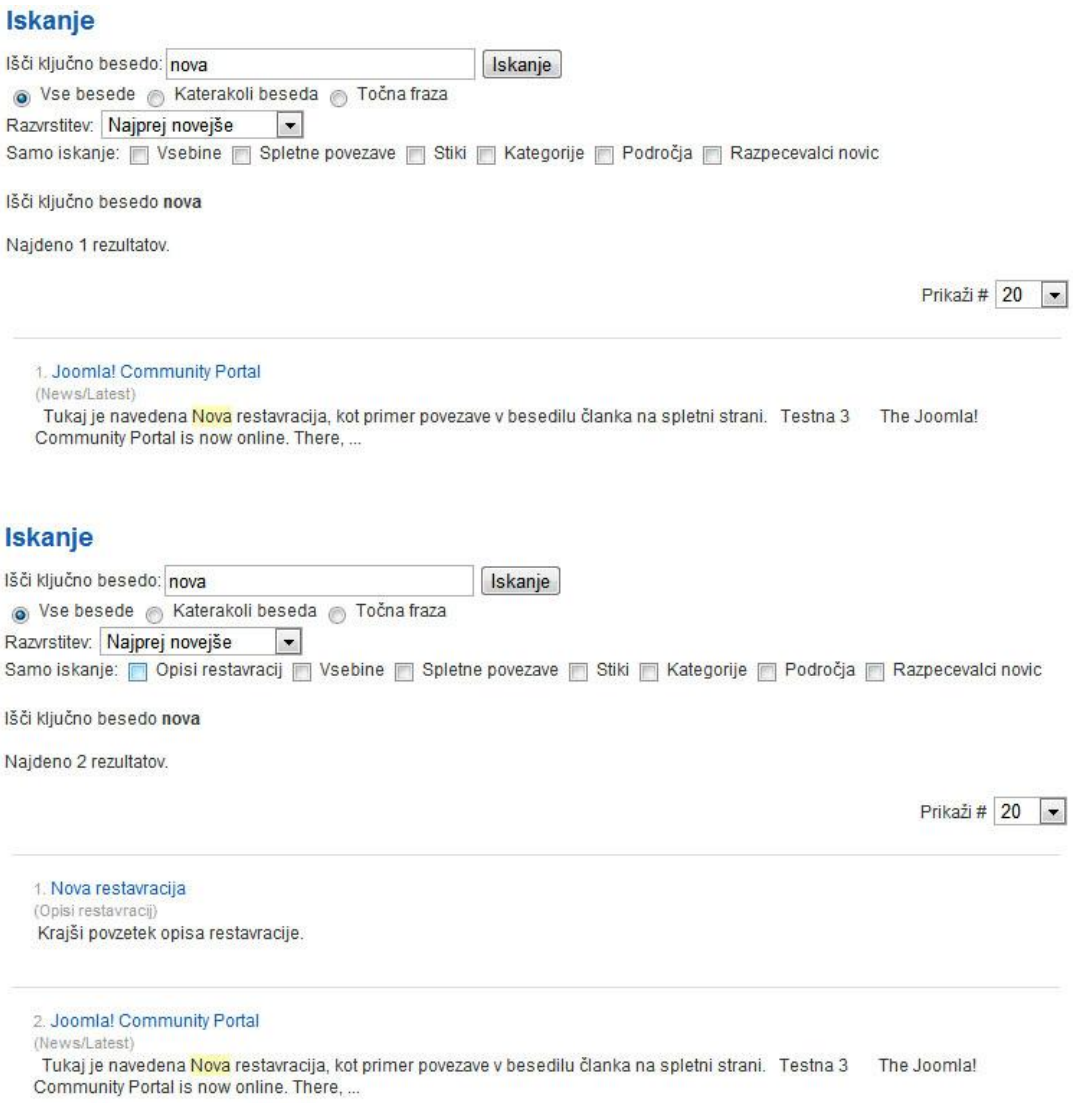
Slika 21: Primer prikaza bistvenih informacij za Nova restavracija

3.3.3. Vtičnik za iskanje po opisih restavracij

Privzeto iskanje v Joomli ne deluje tudi za iskanje po opisih restavracij v naši komponenti. Zato smo ustvarili še tretji vtičnik za iskanje med opisi vseh restavracij.

Za iskanje moramo registrirati metode za dva dogodka, *onSearch* in *onSearchAreas*. Prvi se sproži, ko komponenta za iskanje išče za dano frazo. Sprožitev drugega pa se zgodi, ko uporabniki v iskalnem obrazcu izberejo, po katerih področjih naj poteka iskanje.

Na sliki 22 je v zgornjem delu prikazano iskanje brez vključenega vtičnika in v spodnjem delu slike z vključenim vtičnikom za iskanje. Z omogočenim vtičnikom je bil najden tudi iskani opis restavracije, medtem ko ga z onemogočenim nismo našli. Ob uporabi tega vtičnika ima uporabnik tudi možnost, da išče samo po opisih restavracij (Izbirno polje Opisi restavracij pod Samo iskanje – slika 22 spodaj).



Slika 22: Primer iskanja z vklopljenim vtičnikom in brez njega za iskanje po opisih

3.3.4. Izdelava SEF povezav

Privzete povezave, ki jih ustvarja naša komponenta za opis in ocenjevanje restavracij, so take: *http://localhost/index.php?option=com_opisi_restavracij&id=1&task=view&Itemid=1*

Ti naslovi so dolgi, težko zapomljivi in slabo optimizirani za spletne iskalnike, ki bodo indeksirali našo stran. Veliko lepše bi bilo imeti povezavo v stilu: *http://www.domena.si/restavracije/view/1*. Da smo to lahko dosegli, smo definirali usmerjevanje (routing), ki bo generiralo in dekodiralo SEF – iskalnikom prijazne povezave. Še prej pa moramo v zaledju, v globalnih nastavitvah Joomle, omogočiti SEF povezave ter uporabo *mod_rewrite* modula (uporaba tega nam odstrani index.php iz povezav). Na koncu še

preimenujemo datoteko htaccess.txt v .htaccess, ki se nahaja v korenskem imeniku na spletnem strežniku.

Za izdelavo notranjih povezav v Joomli smo klicali *JRoute:_()* metodo. Ta metoda vzame relativni naslov kot parameter in vrne SEF različico povezave. Za izdelavo te različice naslova *JRoute* najprej posreduje relativno povezavo v tabelo, nato odstrani *option* element in doda svojo vrednost kot prvi segment novega URL-ja. Ta metoda bo nato poiskala datoteko router.php v mapi komponente. Vsebina datoteke router.php je prikazana na sliki 23.

```php
<?php
defined( '_JEXEC' ) or die( 'Restricted access' );

//Gradnja SEF povezav
    function restavracijeBuildRoute(&$query)
    {
        $segments = array();
        if (isset($query['view']))
            {
                $segments[] = $query['view'];
                unset($query['view']);
            }
        if(isset($query['id']))
            {
                $segments[] = $query['id'];
                unset($query['id']);
            }
        return $segments;
    }

//Parsanje SEF povezav ob kliku na opis

    function restavracijeParseRoute($segments)
    {
        $vars = array();
        $vars['view'] = $segments[0];

        if(count($segments) > 1)
        {
            $vars['id'] = $segments[1];
        }
    return $vars;
    }
?>
```

Slika 23: Koda datoteke router.php za gradnjo SEF povezav

4. NAMEŠČANJE RAZŠIRITEV V JOOMLI

Ko smo končali z razvojem teh razširitev, je treba vse mape in datoteke, ki so nastale med razvojem, primerno shraniti. Razširitve lahko nameščamo ročno, tako da kopiramo vse datoteke v datotečni sistem Joomle preko FTP odjemalca in ročno spremenimo tabele v podatkovni bazi, vendar tak način ni najboljši. Pa še parametri pri taki namestitvi niso pravilno nastavljeni. Recimo v meniju komponente v zaledju bi pogrešali vnos za nameščeno komponento. Veliko boljši način je kreiranje paketa datotečnega sistema razširitve, ki ga za namestitev potrebuje razred *JInstaler*.

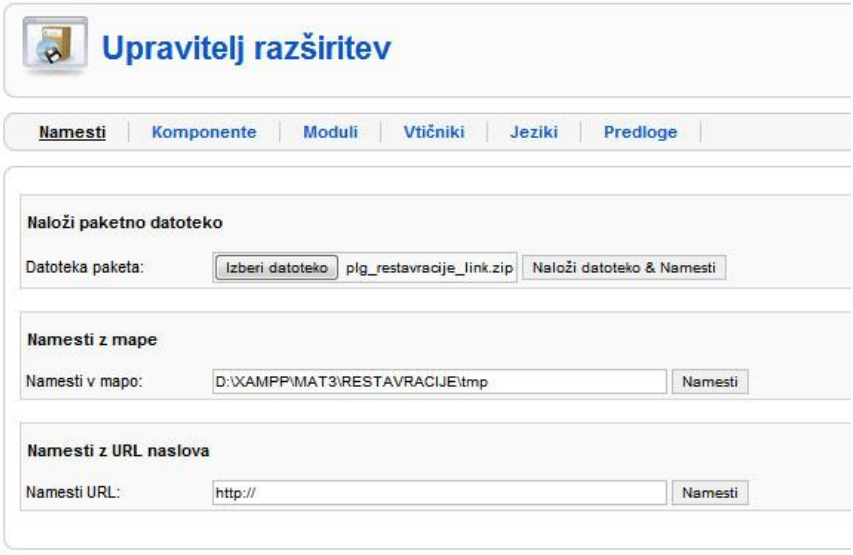
Joomla ob nameščanju razširitev zahteva določene informacije o razširitvi, katere podamo v obliki datoteke XML. Ta vsebuje vse podatke, ki so potrebni za samodejno namestitev izbrane razširitve. Datoteko napišemo razvijalci sami. Če ta datoteka ne obstaja, nam razred *JInstaler,* s pomočjo katerega sistem namešča razširitve, zavrne namestitev in vrne sporočilo o napaki med namestitvijo naše razširitve.

Datoteko moramo poimenovati enako kot razširitev, na primer *restavracijeinfo.xml.* Če jo poimenujemo drugače, bo namestitev sicer uspešno zaključena, vendar sistem Joomla ne bo prikazal vseh parametrov. Seveda če teh parametrov ni, ime datoteke ne vpliva na samo namestitev.

V tej datoteki so zapisane naslednje informacije:

- vse osnovne opisne podrobnosti razširitve (ime, elektronski naslov …) in po izbiri opis, avtorske pravice ter informacije o licenci,
- seznam vseh datotek, ki jih sistem kopira med namestitvijo (datoteke poimenujemo z malimi črkami, da se izognemo težavam na določenih strežnikih),
- izbirno tudi datoteke PHP, ki opravljajo dodatno namestitev in odstranitev operacij,
- kadar razširitev sodeluje s podatkovno bazo, pa tudi .sql datoteko, ki vsebuje poizvedbe zbirke podatkov, ki so izvedeni ob namestitvi ali odstranitvi.

Ko smo tako datoteko ustvarili, jo dodamo k paketu razširitve in vse skupaj primerno shranimo v .zip datoteko. Tak paket lahko enostavno namestimo preko zalednega vmesnika (Slika 24). Če smo XML datoteko napisali brez napak, nam sistem sporoči uspešno namestitev in razširitev nam je tako na voljo. Iste razširitve ne moremo dvakrat namestiti, saj nam sistem javi napako.



Slika 24: Zaledni vmesnik za nameščanje razširitev

5. SKLEP

Sistem Joomla nam ponuja veliko prednosti in predvsem z različnimi razširitvami, ki so nam na voljo, lahko zadovolji potrebe različnim uporabnikom. Ker gre za odprtokodni sistem, ki nam je na voljo vsem, je razširjenost in podprtost še toliko večja. Tudi sama namestitev na spletni strežnik je dokaj enostavna in jo lahko opravi tudi nekdo, ki ni nujno vešč razvijalec in mu ni treba poznati podrobnosti podatkovne baze ali jezika PHP. Potrebuje samo vse dostopne podatke, postavljeno podatkovno bazo in uporabniška imena ter gesla. Z izdelavo predloge po meri ali predelavo kakšne že obstoječe nam sistem nudi tudi unikatnost postavitve naše vsebine.

V sklopu razširitve za sistem Joomla smo izdelali komponento za opis in ocenjevanje restavracij. Ta nam nudi vodenje seznama restavracij, ki vsebuje vse bistvene informacije o posamezni restavraciji, njen opis, cenovni rang ter oceno opisovalca. Z njo smo želeli olajšati vodenje seznama okoliških restavracij in njihovo predstavitev ter tudi boljšo promocijo. Kot dodatno funkcionalnost komponente smo razvili še modul za skrajšan prikaz opisov zadnjih ali pa naključno vnesenih restavracij. Za boljšo povezavo komponente z vsebino spletne strani smo razvili tudi tri vtičnike, kateri preko izdelave povezav iz imena restavracije v besedilu, prikazom bistvenih informacij opisa restavracije ter možnostjo iskanja med opisi restavracij naredijo komponento še uporabnejšo.

Kot vsak sistem je tudi ta prilagojen za izboljšave in nadgradnje. V prihodnosti je ob izkazani potrebi mogoče vse elemente razširitve spremeniti ali prilagoditi novim zahtevam. In ravno to je prednost razvoja lastne komponente, saj nismo kakorkoli omejeni z avtorskimi pravicami ali drugimi omejitvami in tako razširitev tudi dobro poznamo. Razširitev ima lahko z majhno predelavo širok spekter uporabnosti tudi v druge namene.

VIRI IN LITERATURA

[1] James Kennard, Mastering Joomla! 1.5 Extension and Framework Development, Birmingham UK: Packt Publishing, 2007.

[2] James Kennard, Joomla! 1.5 Development Cookbook, Birmingham UK: Packt Publishing 2009.

[3] Ric Shreves, Joomla! Bible, Indianapolis US: Whiley Publishing Inc. 2010.

[4] Joomla sestava. Dostopno na: http://en.wikipedia.org/wiki/Joomla

[5] Arhitektura Joomle. Dostopno na: http://en.wikipedia.org/wiki/Joomla

[6] Razvoj Joomla komponente: Dostopno na: http://docs.joomla.org/Component_Development

[7] 13 nasvetov za boljšo varnost Joomle. Dostopno na: http://www.cio.com/article/703841/13_Tips_for_Better_Joomla_CMS_Security

[8] Varnost v sistemu Joomla. Dostopno na: http://developer.joomla.org/security

[9] Vrste Joomla razširitev. Dostopno na: http://docs.joomla.org/Extension_types_%28general_definitions%29

[10] Model – Pogled – Krmilnik. Dostopno na: http://en.wikipedia.org/wiki/Model-view-controller

[11] Izboljšave Joomle 1.5 http://www.joomla.org/announcements/release-news/4483-joomla-15-overview.html