UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Matej Govek

PRIMERJAVA RAZVOJA SISTEMA ZA UPRAVLJANJE SPLETNIH VSEBIN V TEHNOLOGIJAH ASP.NET IN PHP

DIPLOMSKO DELO NA
VISOKOŠOLSKEM STROKOVNEM ŠTUDIJU

MENTOR: višji pred. dr. Igor Rožanc

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: 00468/2009
Datum: 01.09.2009

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

Kandidat: **MATEJ GOVEK**

Naslov: **PRIMERJAVA RAZVOJA SISTEMA ZA UPRAVLJANJE SPLETNIH VSEBIN V TEHNOLOGIJAH ASP.NET IN PHP**

COMPARISION OF WEB CONTENT MANAGEMENT SYSTEM DEVELOPED BY ASP.NET AND PHP

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija

Tematika naloge:

Različne sodobne tehnologije za razvoj spletnih aplikacij omogočajo učinkovit razvoj spletnih rešitev, vendar v primeru določenih zahtev obstajajo z vidika razvijalca med njimi pomembne razlike tako pri zahtevnosti izvedbe kot pri kakovosti končne rešitve.

V diplomski nalogi predstavite razvoj sistema za upravljanje spletnih vsebin (WCMS) v dveh spletnih tehnologijah: ASP.NET in PHP. Pri razvoju od zahtev do izdelanega sistema uporabite ustrezen metodološki pristop, težišče naloge pa naj bo na prikazu in primerjavi razvoja istih delov sistema na dva različna načina. V nalogi prikažite izgled in uporabo sistema, zaključite pa jo s primerjavo ugotovljenih značilnosti tehnologij ASP.NET in PHP.

Mentor:

viš. pred. dr. Igor Rožanc

Fakulteta za računalništvo in informatiko
2
UNIVERZA V LJUBLJANI

Dekan:

prof. dr. Franc Solina

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: 00468/2009
Datum: 01.09.2009

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

Kandidat: **MATEJ GOVEK**

Naslov: **PRIMERJAVA RAZVOJA SISTEMA ZA UPRAVLJANJE SPLETNIH VSEBIN V TEHNOLOGIJAH ASP.NET IN PHP**

COMPARISION OF WEB CONTENT MANAGEMENT SYSTEM DEVELOPED BY ASP.NET AND PHP

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija

Tematika naloge:

Različne sodobne tehnologije za razvoj spletnih aplikacij omogočajo učinkovit razvoj spletnih rešitev, vendar v primeru določenih zahtev obstajajo z vidika razvijalca med njimi pomembne razlike tako pri zahtevnosti izvedbe kot pri kakovosti končne rešitve.

V diplomski nalogi predstavite razvoj sistema za upravljanje spletnih vsebin (WCMS) v dveh spletnih tehnologijah: ASP.NET in PHP. Pri razvoju od zahtev do izdelanega sistema uporabite ustrezen metodološki pristop, težišče naloge pa naj bo na prikazu in primerjavi razvoja istih delov sistema na dva različna načina. V nalogi prikažite izgled in uporabo sistema, zaključite pa jo s primerjavo ugotovljenih značilnosti tehnologij ASP.NET in PHP.

Mentor:

viš. pred. dr. Igor Rožanc

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Franc Solina

IZJAVA O AVTORSTVU
diplomskega dela

Spodaj podpisani Matej Govek, z vpisno številko 63020213, sem avtor diplomskega dela z naslovom:

PRIMERJAVA RAZVOJA SISTEMA ZA UPRAVLJANJE SPLETNIH VSEBIN V TEHNOLOGIJAH ASP.NET IN PHP

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom dr. Igorja Rožanca,
- so elektronska oblika diplomskega dela, naslov (slov., ang.), povzetek (slov., ang.) ter ključne besede (slov., ang.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 15.03.2010 Podpis avtorja:

ZAHVALA

Zahvaljujem se mentorju diplomske naloge, višjemu predavatelju dr. Igorju Rožancu, za vodenje in prijazno pomoč pri nastanku diplomske naloge.

Posebno bi se zahvalil družini, ki me je v času študija psihično in finančno podpirala in vseskozi spodbujala.

KAZALO

KAZALO SLIK

KRATICE, OKRAJŠAVE, SIMBOLI

AJAX	Asynchronous JavaScript and XML – je skupina med seboj povezanih tehnologij, uporabljena na strani odjemalca za izdelavo interaktivnih spletnih strani.
ASP	Active Server Pages – strežniški skriptni jezik.
ASP.NET	Spletno aplikacijsko ogrodje za izdelavo spletnih aplikacij in spletnih servisov.
BCL	Base Class Library – standardna knjižnica za vse programske jezike ki uporabljajo .NET Framework.
CLR	Common Language Runtime – jedro v Microsoft .NET Framework ogrodju za izvajanje aplikacij.
CMS	Content Management System – sistem za upravljanje s spletnimi vsebinami (programske aplikacije, ki podpirajo generiranje in urejanje spletnih vsebin).
C#	Microsoftov programski jezik.
HTML	HyperText Markup Language – standarden jezik za razvoj spletnih strani. Z jezikom HTML torej označujemo in določamo lastnosti besedila.
HTTP	HyperText Transfer Protocol – komunikacijski protokol med odjemalci in strežniki. Protokol je namenjen objavljanju in prejemanju informacij na spletu preko HTML strani.
JS	JavaScript – je skriptni jezik, ki omogoča izvajanje določenih operacij na uporabnikovi strani in tako omogoča hitrejše delovanje aplikacije.
OOP	Object oriented programming – objektno programiranje.
PHP	Hypertext Preprocessor, izvirno Personal Home Page tools – strežniški skriptni jezik.
SQL	Structured Query Language – strukturni poizvedovalni jezik.
T-SQL	Transact – SQL – serija programskih razširitev za SQL ki omogoča nadzor nad transakcijami, ter rokovanje z napakami.
URL	Uniform Resource Locator – naslov lokacije objekta, običajno internetne strani.
UML	Unified Modeling Language – je splošen vizualni jezik za modeliranje.
WCF	Windows Comunicatin Fundation – enoten programski model za gradnjo porazdeljenih sistemov na Microsoftovi platformi.

WCMS	Web Content Management System – sistem za upravljanje s spletnimi vsebinami (spletne aplikacije, ki podpirajo generiranje in urejanje spletnih vsebin).
WF	Windows Workflow Fundation – je programski model, ki razvijalcem ponuja že izdelano infrastrukturo za uporabo delovnih tokov znotraj aplikacije, ki jo razvijajo.
WPF	Windows Presentation Fundation – za prikazovanje grafičnih vmesnikov v Windows aplikacijah.

POVZETEK

Cilj diplomske naloge je primerjava dveh najbolj uporabljenih tehnologij za razvoj spletnih aplikacij: PHP in ASP.NET z jezikom C#. Primerjava je izvedena na primeru izdelave sistema za upravljanje s spletnimi vsebinami (ang. WCMS).

Eden najučinkovitejših načinov, da našo dejavnost spozna širši krog ljudi, je predstavitev preko spletne strani na svetovnem spletu. V diplomski nalogi smo predstavili razvoj in delovanje sistema za upravljanje s spletnimi vsebinami ter primerjavo za to uporabljenih tehnologij PHP in ASP.NET.

ASP.NET je Microsoftovo spletno aplikacijsko ogrodje, ki omogoča izdelavo dinamičnih spletnih aplikacij in spletnih servisov. ASP.NET je zgrajen na CLR jedru, ki omogoča kodiranje v kateremkoli jeziku, ki je podprt z .NET ogrodjem.

PHP je odprtokodni programski jezik za razvoj dinamičnih spletnih vsebin. Teče na spletnem strežniku, kjer jemlje PHP izvorno kodo kot vhod in generira spletno stran kot izhod.

Sistem za upravljanje spletnih vsebin je razvit tako v ASP.NET kot v PHP različici. Ima dva modula in sicer uporabniški in administratorski. V uporabniškem se izdelujejo spletne strani, administratorski pa služi za razvrščanje uporabnikov v skupine ter določanje njihovih pravic. Za uporabo sistema je potrebna registracija. Administrator mora uporabniku določiti pravice, na podlagi katerih uporabnik lahko izdeluje spletno stran za določeno opcijo menija. Možno je določiti tudi obdobje, v katerem bo spletna stran objavljena.

Za izdelavo aplikacije z uporabljeno infrastrukturo se je izkazala kot boljša rešitev ASP.NET tehnologija. Kot glavno prednost naj navedemo lažje kodiranje in razhroščevanje. Prednosti PHP-ja bi se izkazale predvsem v primeru, ko bi morala aplikacija delovati na različnih operacijskih sistemih ali različnih strežnikih.

Ključne besede:
ASP.NET, PHP, .NET Framework, CMS, WCMS

ABSTRACT

The main aim of the diploma thesis is the comparison of the two very popular technologies for the development of web applications. The comparison of PHP and ASP.NET with C# language is done on example of web content management system (WCMS) development.

Web site publications are one of the most efficient ways to present our activity to the general public. This diploma work describes the development and performance of web content management system. The comparison of used technology PHP and ASP.NET is presented as well.

ASP.NET is a web application framework developed by Microsoft. It allows programmers to build dynamic web applications and web services. ASP.NET is built on the Common Language Runtime (CLR) which allows programmers to write code using any supported .NET language.

PHP is an open source programming language for building dynamic web applications. PHP generally runs on a web server, where output web site is generated from PHP code.

Web content management system is built in PHP and ASP.NET version. It has two modules. The users module is designed to make web sites, while the administrator module is used to group users and define permissions. Registration is required to use the system. The administrator has to define the permissions for user. Those are the base for creation of defined option. It is also possible to define the period in which the web site will be operational.

The best technology for building applications with given infrastructures proved to be ASP.NET. His advantage is based on easy coding and debugging. PHP has advantage when applications runs on the different servers or different operating systems.

Keywords:
ASP.NET, PHP, .NET Framework, CMS, WCMS

1. UVOD

Začetki predstavitev na internetu so potekali z uporabo statičnih spletnih strani. Te so izvajalci načrtovali in izvedli z namenom, da se ne bodo spreminjale oz. se bodo spreminjale v manjšem obsegu, ker pogosto spreminjanje vsebine privede do dodatnih stroškov. S spreminjanjem vsebine v kratkih časovnih intervalih in z naraščanjem količine vsebine se lahko pojavi tudi problem nadzora vsebine. V današnjem času se količina vsebine, ki jo želijo tako pravne kot fizične osebe predstaviti na internetu hitro povečuje, zato se povečuje tudi čas potreben za postavljanje in vzdrževanje strani. Pojavili so se različni sistemi, ki omogočajo enostavno izvedbo in postavitev spletne strani, najbolj znani so CMS, WCMS, Wiki in Blog. Ti sistemi za razliko od statičnih spletnih strani vsebine ne hranijo v dokumentih, temveč v podatkovni bazi.

Sistemi za upravljanje s spletnimi vsebinami CMS in WCMS so aplikacije, ki podpirajo izdelavo in urejanje spletnih vsebin. Zaradi velike podobnosti med njimi v nadaljevanju uporabljam kratico CMS. Uporabnikom omogočajo da se izognejo urejanju HTML kode, hkrati pa nudijo kompleksne rešitve za dodajanje, posodabljanje, strukturiranje, povezovanje, arhiviranje, iskanje in komuniciranje spletnih vsebin. Z uporabo CMS lahko uporabniki novice dodajo kar v urejevalniku podobnem Wordu, ne da bi se ukvarjali z oblikovanjem novice v HTML kodi ali njenim prenosom na odgovarjajoči strežnik.

Glavna dilema pri vzpostavitvi določenega spletnega mesta je izbira ustreznega CMS. Na prvem koraku je izbira med izdelavo lastnega, ali uporabo že izdelanega CMS, kjer se odločamo med odprtokodnimi ali komercialnimi rešitvami. Pri komercialnih rešitvah je velikokrat težava visoka cena ter nejasno specificirani stroški vzdrževanja. To pripelje tudi do problema navezanosti, kajti z naraščanjem zahtevnosti spletne storitve, se zaradi že plačane cene težje odločimo za menjavo CMS, čeprav je ta neustrezen ali nepovezan z informacijskim sistemom organizacije. Pri odprtokodnih rešitvah so problemi predvsem zaradi neprijaznosti sistema, kar zahteva veliko vlaganje lastnega dela in vaje, nihče nam ne zagotavlja delovanje sistema, poleg tega pa razvoj lahko zaide v slepo ulico. Na tržišču je veliko različnih sistemov za upravljanje s spletnimi vsebinami, ki temeljijo na različnih programskih jezikih, uporabljajo različne spletne in aplikacijske strežnike, ter različne podatkovne baze. Pri izbiri sistema naletimo še na težave, ker se določene vrste funkcionalnosti med sistemi prekrivajo ali izključujejo, odločitev pa otežuje še možnost razširitve sistema, ki ga sistem prilagodi posameznemu uporabniku. Poleg tega pa je za izbiro potrebno veliko dela, da se iz raznih virov dokumentacij potencialnih sistemov izlušči želene informacije. Zaradi omenjenega smo se odločili izdelati lasten CMS.

Tako se je pojavilo vprašanje, katero tehnologijo uporabiti. Na spletu je cel kup forumov, kjer privrženci ene ali druge tehnologije podajajo svoja mnenja, a nobeno se ne zdi preveč

objektivno. Tako smo se tega večnega vprašanja lotili s preizkusom. Sklenili smo razviti sistem v dveh tehnologijah ASP.NET in PHP, ki ju v zaključku primerjamo ter predstavimo njune prednosti in slabosti.

Namen in cilji diplomske naloge:

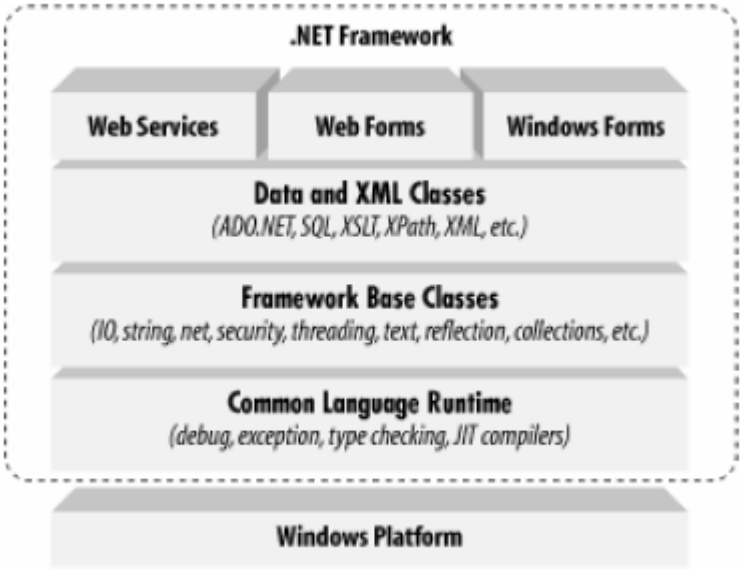
- opisati in predstaviti ustrezne tehnologije in orodja,
- izdelati uporabno spletno aplikacijo za upravljanje s spletnimi vsebinami v dveh tehnologijah ASP.NET in PHP,
- primerjati oba rezultata.

2. UPORABLJENA TEHNOLOGIJA IN ORODJA

Poglavje je namenjeno predstavitvi uporabljenih tehnologij in orodij. Za enak ali podoben izdelek bi lahko uporabili tudi druge tehnologije in orodja.

2.1. Ogrodje .NET

Ogrodje .NET (ang. .NET Framework) je programska komponenta, ki jo naložimo na Microsoftove operacijske sisteme [10]. Ponuja obširen nabor vnaprej pripravljenih rešitev za standardne programske potrebe. Ogrodje (slika 1) je s svojo enostavno uporabo postalo ključni element za razvijanje programskih rešitev, ki delujejo na platformah Microsoft. Sestavljata ga dva ključna dela: knjižnice in CLR virtualni stroj. Virtualni stroj skrbi za izvajanje programske kode napisane v enem od jezikov ki so podprti z ogrodjem .NET. Ogrodje .NET predstavlja največji premik iz DOS okolja v Windows NT. Navaja odhod COM standarda in pozdravlja izvajanje aplikacij znotraj skupnega izvajalnega okolja CLR.



Slika 1 : Ogrodje arhitekture .NET Framework

Osnovni razredi ogrodja .NET so razvrščeni po imenskih prostorih (ang. namespaces). Pomembni imenski prostori so:

- `System` – standardni razredi, ki jih uporabljajo skoraj vsi programi,
- `System.Data` – upravljanje s podatkovnimi bazami (ADO.NET),
- `System.IO` – ukazi za delo z datotečnim sistemom,

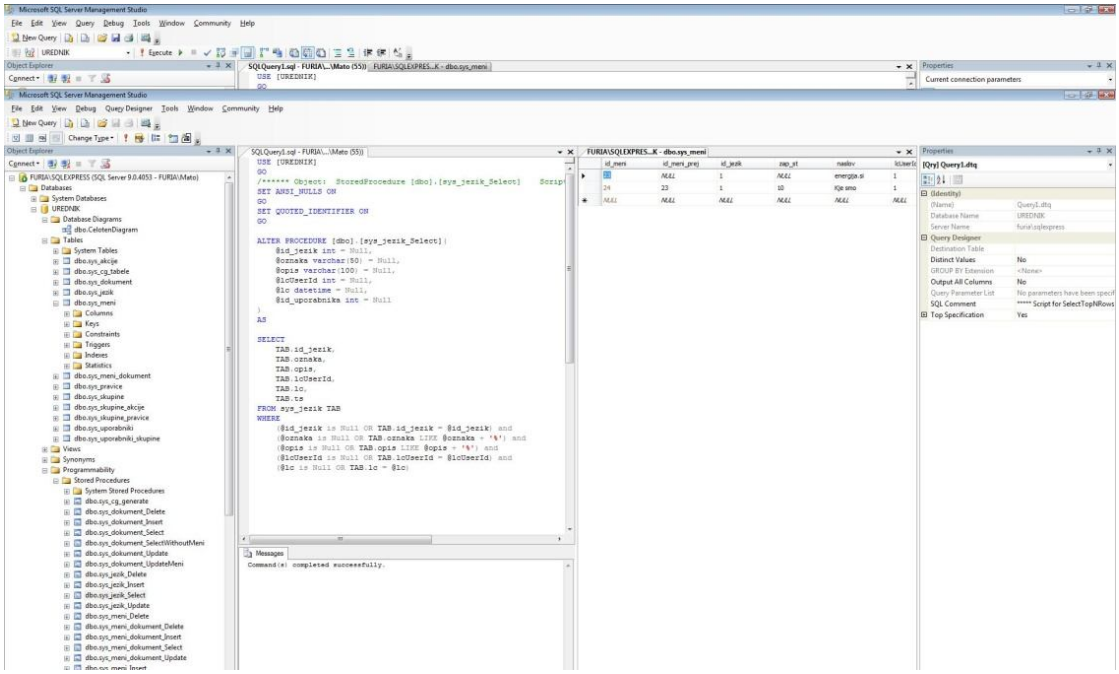
- `System.Windows.Forms` – razredi za delo z Windows uporabniškim vmesnikom,
- `System.Net` – mrežno programiranje,
- `System.Web.UI` – spletne strani ASP.NET,
- `System.Collections` – razredi, ki omogočajo upravljati s podatkovnimi zbirkami objektov,
- `System.Security` – razredi, ki omogočajo kriptografijo in pravice,
- `System.Threading` – razredi, ki omogočajo delo z nitmi.

.NET Framework 3.5 je zadnja različica ogrodja .NET. Prinaša nove funkcionalnosti in izboljšave kot so:

- LINQ (Language Integrated Query), ki omogoča uporabo SQLu sorodnih stavkov neposredno v jezikih ogrodja .NET,
- ASP.NET AJAX,
- novi protokoli za izgradnjo WCF servisov,
- nova orodja za upravljanje z WF, WCF in WPF,
- novi razredi v BCL.

2.2. Microsoft SQL Server 2008 Express

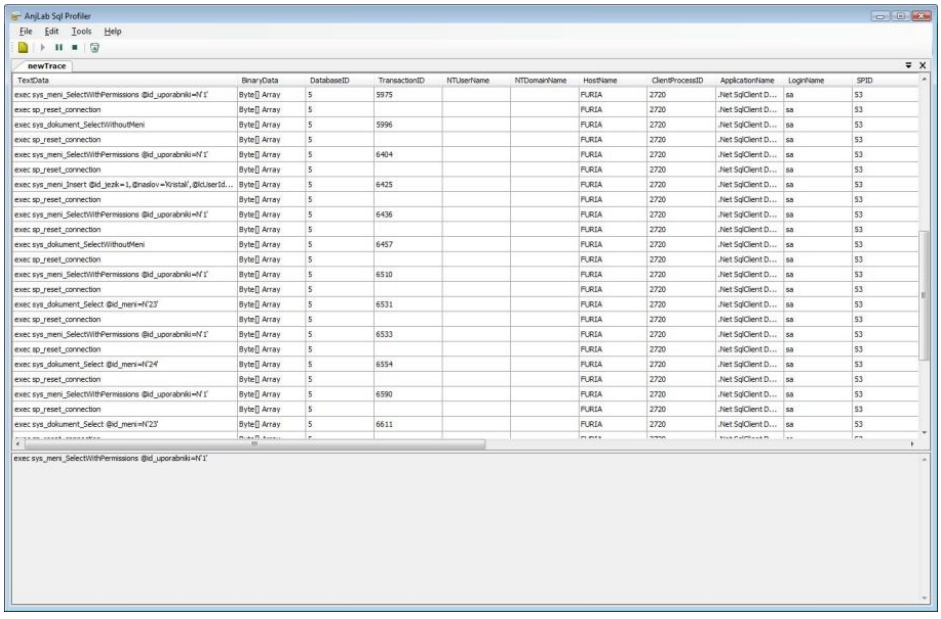
Microsoft SQL Server 2008 Express je brezplačna tehnologija in orodja za upravljanje z relacijskimi podatkovnimi bazami (ang. Relational Database Management System- RDBMS) [11]. Podpira shranjene procedure, funkcije in poglede. Uporablja T-SQL razširitev povpraševalnega jezika SQL. SQL Server Management Studio je orodje vsebovano v programskem paketu Microsoft SQL Server 2008 Express ki na enostaven in pregleden način (slika 2) omogoča upravljanje podatkovnih baz in njihovih komponent.



Slika 2: SQL Server Management Studio

2.3. AnjLab SQLProfiler

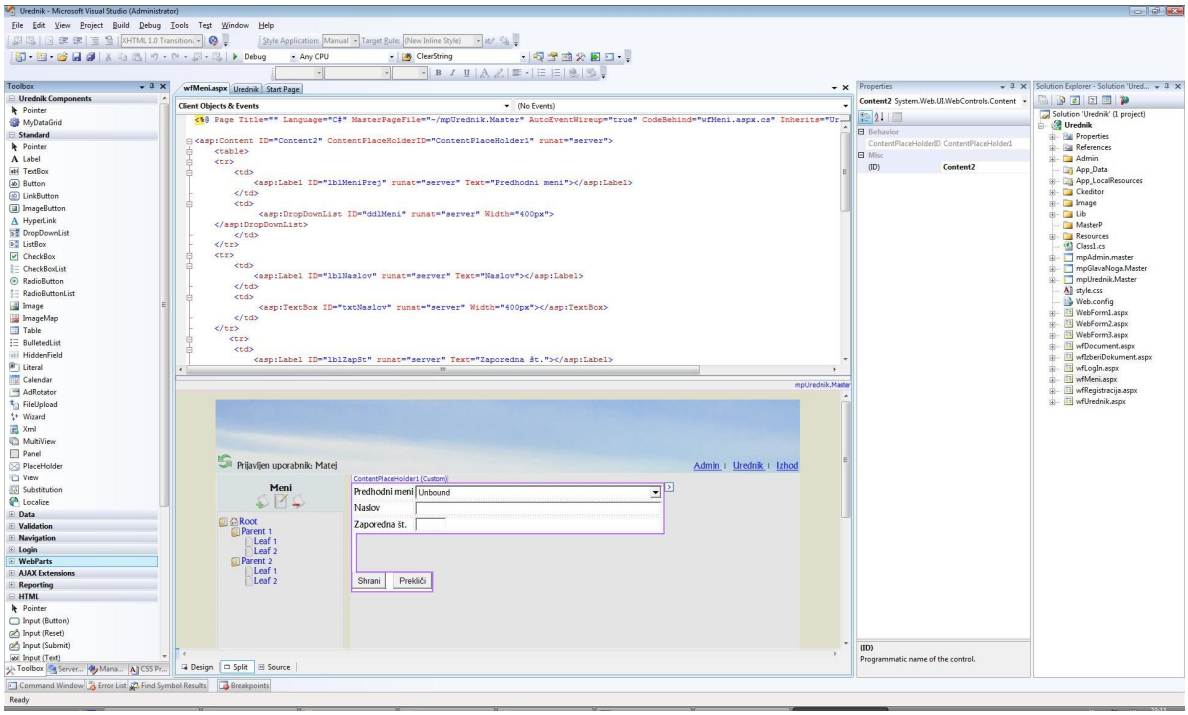
AnjLab SQLProfiler (slika 3) je brezplačno grafično orodje za pregledovanje dogodkov na Microsoft SQL strežniku. Ponuja možnost izbire dogodkov in filtrov. Enostavno lahko nadzorujemo procedure, ki se izvajajo na SQL strežniku, saj je razvidno katera procedura se izvaja, na kateri bazi, kdo jo izvaja, njen začetek in konec izvajanja, s katerimi parametri se kliče in podobno.



Slika 3: AnjLab SQLProfiler

2.4. Microsoft Visual Studio 2008

Microsoft Visual Studio 2008 je zbirka različnih orodij za razvijalce programske opreme. Razvojno okolje (slika 4) je namenjeno za profesionalne razvijalce, ki razvijajo aplikacije za splet (ASP.NET, AJAX), Windows, Windows Server, sistem Microsoft Office, SQL Server, igre (tako za PC kot Xbox) in naprave Windows Mobile [10].



Slika 4: Razvojno okolje Microsoft Visual Studio 2008

Razvojno okolje nudi podporo različnim programskim jezikom: Visual C#, Visual C++, Visual Basic in drugim. Združuje zmogljiv urejevalnik kode, prevajalnik, razhroščevalnik, orodja za dokumentiranje programov in druga orodja, ki pomagajo pri pisanju programskih aplikacij. Aplikacije, ki so razvite v tem okolju, tečejo na vseh platformah, ki podpirajo ogrodje Microsoft.NET.

Bistvene prednosti Visual Studia 2008 pred prejšnjimi verzijami se kažejo v:

- lastni izbiri .NET Frameworka za razvoj,
- novi verziji jezika C# in Visual Basic.NET
- izboljšavi sistema Intelisense (sistem za intuitivno programiranje),
- razhroščevanju JavaScript kode,
- vgrajeni podpori AJAX,
- Windows Communication Fundation (WCF),
- Windows Presentation Fundation (WPF),
- Windows Workflow Fundation (WF),

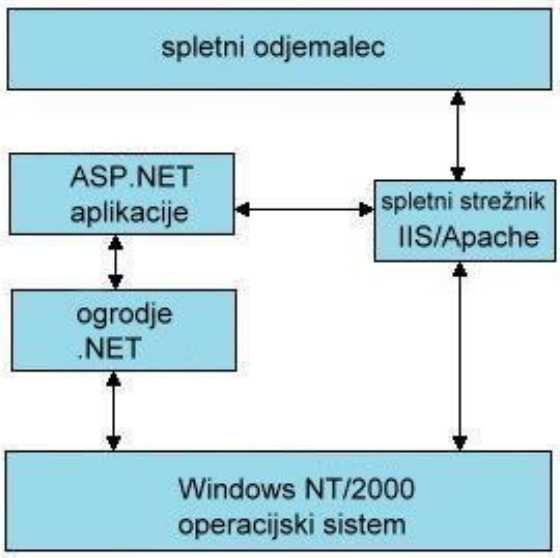
- CardSpace – je instanca identifikacijskega razreda `Identity Selector`. CardSpace shranjuje reference digitalnih uporabniških identitet.
- komponentnem testiranju posameznih modulov (ang. unit testing).

2.5. Spletni strežnik IIS

Spletni strežnik IIS (ang. Internet Information Services), nekdaj tudi Internet Information Server, je niz internetnih storitev za strežnike, ki jih ponuja Microsoft v operacijskem sistemu Microsoft Windows [1]. Je drugi najbolj priljubljeni spletni strežnik na svetu.

2.6. ASP.NET

ASP.NET je brezplačna spletna tehnologija, katere predhodnik je ASP [8]. Brezplačno razvojno okolje za razvoj v tej tehnologiji je Visual Web Developer Express, ki je okrnjena različica Visual Studia. Tehnologija ni nadgradnja starejše, ampak zajema nove inovativne gradnike za spletno programiranje. Omogoča razvoj dinamičnih, interaktivnih spletnih aplikacij in spletnih servisov. Ogrodje vsebuje vrsto pripravljenih storitev in kontrol, ki pospešijo izdelovanje projekta in olajšujejo delo. ASP.NET deluje v integraciji z ogrodjem .NET (slika 5). Zgrajen je na CLR-ju, ki omogoča programerjem pisanje ASP.NET kode z uporabo kateregakoli .NET jezika. Izvaja ga HTTP strežnik IIS ali Apache.

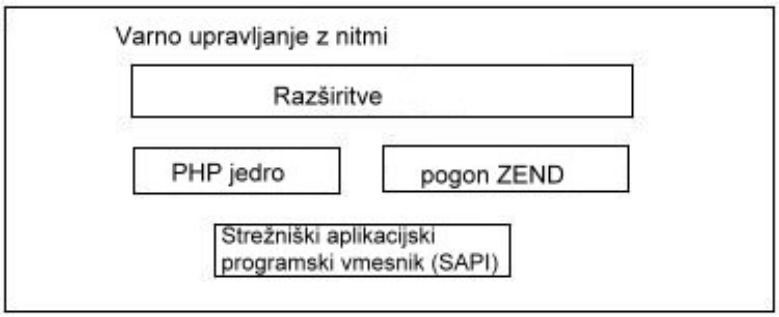


Slika 5: Arhitektura ASP.NET

ASP.NET spletna stran je fizično razdeljena na tri datoteke. Razdeli označevalni (HTML) jezik, ASP.NET kodo (ang. code behind) in deklaracijo kontrol, ki jih prevajalnik (ang. compiler) potem združi. Tako je spletna stran razdeljena na tri datoteke s končnicami `.aspx, .aspx.cs` in `.aspx.designer.cs.`

2.7. PHP 5

PHP (ang. PHP Hypertext Preprocessor, izvirno Personal Home Page tools) je strežniški skriptni jezik odprtega izvora, ki se pogosto uporablja v povezavi s HTML-jem [9]. Kot del PHP-ja sta tudi možnost zaganjanja skript v ukaznem načinu in kreiranje grafičnih aplikacij. V začetku je bil zamišljen kot niz makrojev, ki bi piscem kode pomagali pri vzdrževanju osebnih domačih strani, zato tudi tako izvorno ime. Napisal ga je dansko-kanadski programer Rasmus Lerdorf leta 1994, da bi zamenjal nekaj v Perlu napisanih skript, ki jih je uporabljal za upravljanje svoje spletne strani. Od takrat so bile zmožnosti PHP-ja razširjene iz niza pripomočkov v programski jezik s številnimi možnostmi, s katerimi je mogoče upravljati velika spletna okolja, ki jih poganjajo zbirke podatkov. Sam pojem PHP se nanaša na celoten sistem, sestavljen iz petih delov, ki so vidni na sliki 6 in pojasnjeni v nadaljevanju:



Slika 6: Zgradba sistema PHP

Strežniški aplikacijski programski vmesnik

Strežniški aplikacijski programski vmesnik (ang. Server Application Programming Interface - SAPI) koordinira življenjski cikel PHP-ja in je običajno vgrajen (ang. embeded) v spletni strežnik, npr. Apache http Server. Spletni strežnik skrbi za komunikacijo med njima; sprejema zahtevke po straneh PHP (datotekah) in jih posreduje sistemu PHP.

Pogon Zend

Pogon Zend (ang. Zend Engine) je skriptni pogon, na katerem delujejo moduli povezani s PHP-jem. Zend razčlenjuje (ang. parse) in prevaja skripte PHP v strojno kodo pa tudi skrbi za njeno izvajanje. Zagotavlja tudi vmesnik (ang. application programming interface - API) za

njegove razširitve, kar omogoča uporabo raznih merilnikov (ang. profiler), razhroščevalnikov ipd.. Bil je optimiziran tako, da zagotavlja močno izboljšano učinkovitost delovanja in razširljivost. PHP 5 prinaša nove bistvene izboljšave z uvedbo pogona Zend 2. Pogon Zend 2 zelo izboljša podporo objektnemu programiranju. Prvič so objekti in objektno usmerjeni načrt v središču PHP-ja, zaradi česar je še primernejša osnova za aplikacije za velika podjetja.

PHP jedro

PHP jedro (ang. PHP Core) omogoča izvajanje osnovnih funkcij kot so: delo z datotekami, napakami, tokovi, ipd., vendar te funkcije niso na voljo programerjem, ampak razširitvam (ang. extensions). Skrbi tudi za njihovo inicializacijo in omogoča njihovo uporabo.

Razširitve

Tu se nahajajo razne razširitve (delo s sejo, nizi, bazo itd.), katerih funkcije so na voljo programerjem.

Varno upravljanje z nitmi

Varno upravljanje z nitmi (ang.Thread Safe Resource Management) zagotavlja, da posamezna instanca sistema PHP pravilno obdeluje več zahtev hkrati.

V nasprotju z običajno HTML stranjo strežnik PHP skripte ne pošlje neposredno odjemalcu, ampak jo razčleni sam pogon PHP. Elementi HTML v skripti so izpuščeni, koda PHP pa je prevedena in izvedena. S kodo PHP v skripti lahko poizvedujemo po zbirkah podatkov, izdelujemo slike, beremo in zapisujemo datoteke, komuniciramo z oddaljenimi strežniki. Rezultat kode PHP je povezan s HTML-jem v skripti in poslan uporabniku [2].

Zadnja verzija je PHP 5, ima pa več podverzij. Za izdelavo diplomske smo uporabili verzijo PHP 5.2.11. Izboljšave PHP 5:

- ima vgrajeno podporo za XML. Vse različne funkcije in razredi, ki zagotavljajo obdelavo XML na različne načine, zdaj uporabljajo isto vgrajeno knjižnico. Tako so funkcije XML bolj stabilne in medsebojno izvedljive,
- Knjižnica SQL za SQLLite je zdaj del PHP, skupaj z vsemi funkcijami, ki jih potrebujemo za delo z njo,
- podpira zasebne in zaščitene metode v razredih,
- podpira konstante razredov,
- Objekti so posredovani funkcijam in metodam po referenci. To zelo zmanjša verjetnost napak v objektno usmerjeni kodi,
- PHP podpira statične metode in lastnosti, zaradi česar lahko napredneje objektno načrtujemo,

- Metode je mogoče deklarirati tako, da zahtevajo določene vrste parametrov,
- Primerjalni operator (===)preverja, ali dva sklica kažeta na isti objekt. Prej je bilo težko preizkušati objekte na ta način,
- PHP zdaj podpira abstraktne razrede in vmesnike.

PHP je mogoče uporabljati na različnih platformah. Izvaja se v operacijskem sistemu Windows, večini različic Unixa, vključno z Linux-om ter v operacijskem sistemu Macintosh OS X. Obstaja podpora za niz spletnih strežnikov, tudi za Apache (ki je odprtega izvora in deluje na različnih platformah), Microsoft IIS (Internet Information Services), WebSytePro, iPlanet Web Server, Microsoft Personal Web Server. V večini primerov je PHP mogoče namestiti kot strežniški modul. Z drugimi besedami, raje kot ločena aplikacija se izvaja kot del procesa strežnika. Uporabljamo ga lahko tudi kot samostojno aplikacijo, ki se izvaja iz ukazne vrstice.

Datoteka s PHP kodo ima končnico `.php`. Ta lahko vsebuje tako PHP kot HTML kodo. PHP kodo lahko tako vrinemo med HTML oznake, ali pa jo ločimo od HTML jezika v posebno datoteko. Ločevanje kode je prepuščeno presoji programerja.

2.8. CKEditor

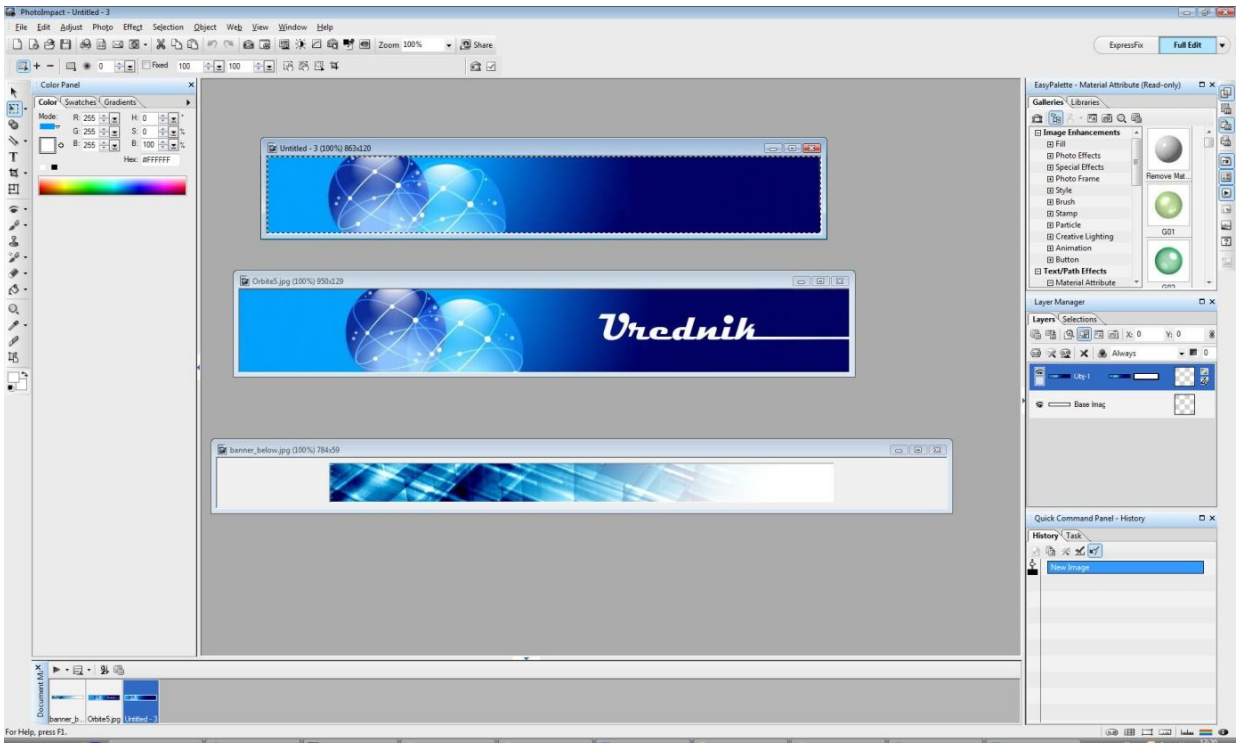
CKEditor je urejevalnik besedila za splet. Je WYSIWYG (ang. What You See Is What You Get) urejevalnik, kar pomeni, da je tekst, ki se ureja, enak končnemu izgledu dokumenta. Uporabniški vmesnik (slika 7) spominja na standardne urejevalnike besedil, kot sta Microsoft Word ali Open Office Writer [12].

Slika 7: Orodna vrstica CKEditorja

2.9. Photoimpact

Photoimpact je rastrski grafični program za urejanje slik [13]. Razvilo ga je podjetje Ulead. Prva verzija je bila izdana leta 1996 in od takrat so skoraj vsako leto izdali novo boljšo različico programa. Program se lahko primerja z bolj znanim Photoshop-om. Njegova prednost je predvsem prijaznejši uporabniški vmesnik. Združitev Uleada s podjetjem Corel je pripeljala do ustavitve razvoja Photoimpacta, z namenom osredotočanja na Corelov Paint Shop Pro. Novica je bila objavljena ravno med pisanjem diplomske naloge. Zadnja verzija programa Photoimpact X3 je bila izdana leta 2008 in bo kot kaže zadnja.

Orodje Photoimpact smo uporabili za izdelavo grafične podobe spletne aplikacije. Slika 8 prikazuje delovno okolje. Levo se nahaja orodna vrstica in kontrolna plošča, na sredini je slika v obdelavi, spodaj so vsi dokumenti, desno pa so plasti, ki tvorijo sliko, galerija in zgodovina.



Slika 8: Prikaz programa Photoimpact

3. RAZVOJ APLIKACIJE ZA UPRAVLJANJE S SPLETNIMI VSEBINAMI

V prejšnjem poglavju smo opisali orodja in tehnologije, ki smo jih uporabili za razvoj aplikacije. V tem poglavju pa bomo predstavili razvoj aplikacije od analize zahtev do končnega izgleda narejene aplikacije ter spletne strani.

3.1. Razvoj aplikacije po metodologiji RUP

To poglavje je namenjeno predstavitvi metodologije razvoja aplikacij, katere smernice smo uporabili. Prednost se je izkazala predvsem v uporabi iterativnega razvoja, uporabili pa smo tudi diagrame primerov uporabe.

Metodologija RUP (ang. Rational Unified Process) zajema vse, kar počnemo, da bi dosegli želen rezultat, torej izdelek ali storitev, ki je cilj našega dela. Velja za težko metodologijo, saj zajema in podrobno opisuje številne elemente. Kljub temu se je metodologija izkazala kot primerna izbira. RUP se izvaja iterativno ter za vizualizacijo uporablja jezik UML [3]. Glavni namen uporabe te metodologije je uspešen razvoj programske opreme.

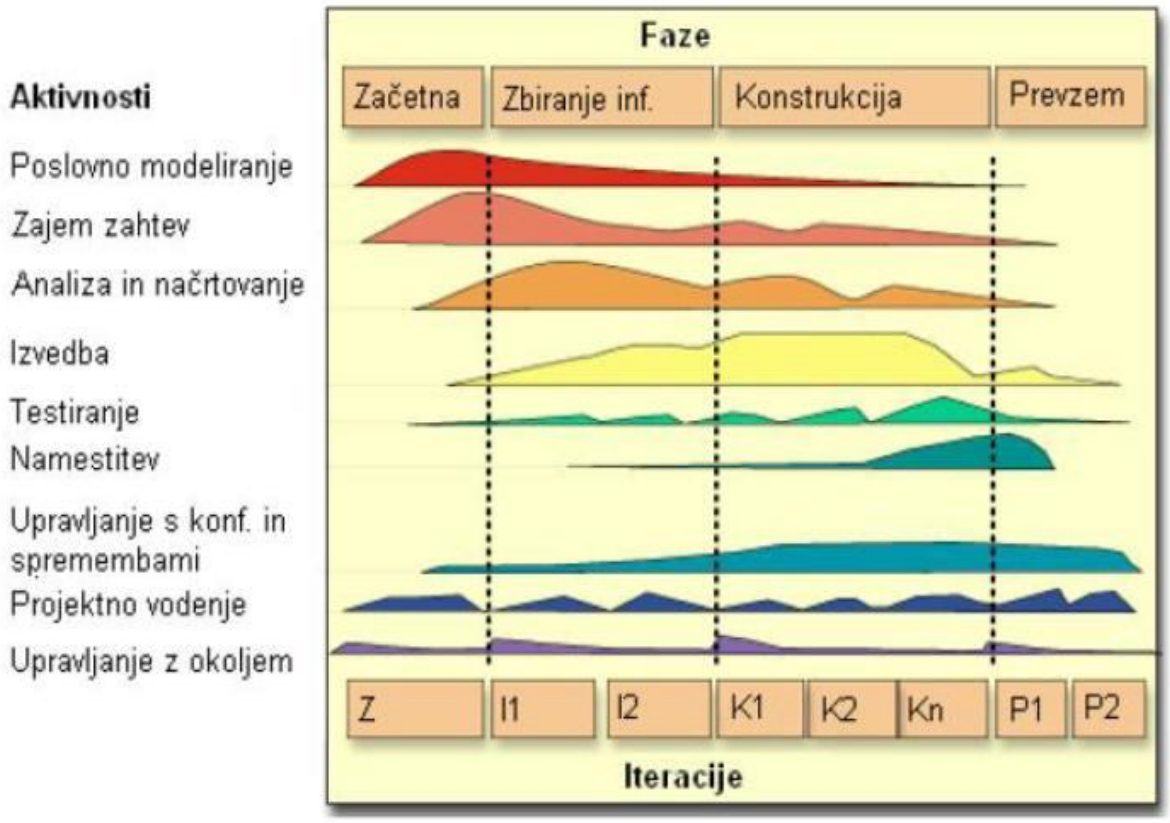
Metodologija RUP je baza znanja, ki je nastala kot rezultat prizadevanja za bolj učinkovito objektno usmerjen razvoj programske opreme na ravni organizacije. Opisuje kako učinkovito uporabiti šest najboljših izkušenj s področja razvoja programske opreme [4]:

- **iterativni razvoj** – temelji na izvajanju več iteracij oziroma ponovitev, kjer v vsaki iteraciji razvijemo del funkcionalnosti sistema,
- **obvladovanje zahtev** – tukaj gre za sledenje zahtev od začetka, preko sprememb, do zaključka,
- **uporabo komponente arhitekture** – razbijanje razvoja na več delov, kar zmanjša zapletenost sistema
- **vizualno modeliranje** – prikaz modela sistema z uporabo jezika UML,
- **preverjanje kakovosti** – gre za ocenjevanje kakovosti izdelka z vidika funkcionalnosti, zanesljivosti in zmogljivosti sistema,
- **nadzorovanje sprememb** – tukaj gre za redno spremljanje in poročanje. Pri nadzoru pomagajo struktura načrtov in vnaprej določena poročila.

RUP zajema štiri faze življenjskega cikla, pri čemer lahko vsako fazo razdelimo na več iteracij (slika 9):

- **Začetna faza:** začetek projekta, zbiranje informacij, opredelitev okvirjev obravnavanega področja, študija izvedljivosti, definicija obsega projekta in utemeljitev poslovne pridobitve,

- **Zbiranje informacij:** analiza in načrtovanje obravnavanega področja, specifikacija značilnosti, načrtovanje arhitekture,
- **Konstrukcija:** konstrukcija zajema kodiranje in izdelavo izdelka. Konstrukcija lahko poteka v več iteracijah, če gre za velik projekt,
- **Prevzem:** priprava za prenos končnemu uporabniku, namestitev pri uporabniku.



Slika 9: Shema procesa razvoja po metodologiji RUP

V posameznih fazah ali iteracijah imamo opravka z večjim številom aktivnosti, ki tečejo vzporedno (slika 9). V začetni fazi imamo opravka predvsem s poslovnim modeliranjem in zajemom zahtev. V drugi fazi prevladuje aktivnost analize in načrtovanja. Testiranje, upravljanje z okoljem in projektno vodenje so aktivnosti, ki so prisotne skozi vse štiri razvojne faze.

3.2. Analiza zahtev in načrtovanje

Za uspešno izvedbo razvoja aplikacije sta analiza in načrtovanje najpomembnejša dejavnika. Z njima lahko dosežemo boljše razumevanje sistema, ga poenostavimo ter zagotovimo ponovno uporabo, vizualizacijo in nadzor nad arhitekturo sistema.

V tem poglavju bomo predstavili:

- funkcionalne in nefunkcionalne zahteve sistema,
- diagram primerov uporabe za celotni sistem in
- podatkovni model.

3.2.1. Nefunkcionalne zahteve

V tem poglavju bomo podali nekaj najpomembnejših nefunkcionalnih zahtev, ki se nanašajo na tehnične ter ostale nevsebinske zahteve sistema.

1. Strojna in programska oprema
Strojna oprema zahteva osebni računalnik, ki ima dostop do interneta. V okviru programske opreme potrebujemo IIS, PHP, Microsoft .NET Framework 3.5, ASP.NET in Microsoft SQL Server. Zagotoviti moramo tudi JS podporo za brskalnik.

2. Upravljanje s podatki
Sistem mora zagotoviti učinkovitost tudi v primeru obravnavanja večje količine podatkov.

4. Performanse
Sistem naj omogoča tekoče delovanje (paziti je potrebno na zahtevnejše algoritme in poizvedbe). Ostale performanse so pogojene s performansami strežnika in odjemalca vključno s hitrostjo internetne povezave.

5. Varnost
Zaradi občutljivosti in pomembnosti podatkov je varnost zelo pomembna. Vsi podatki bodo zaradi varnosti shranjeni v podatkovni bazi, z izjemo slik, ki jih za delo s CKEditorjem rabimo v datotečnem imeniku. Sistem mora omogočati varno dostopanje do podatkov na bazi. Vsaka poizvedba v bazi naj bo klic shranjene procedure.

6. Uporabniški vmesnik
Uporabniški vmesnik mora biti zanimiv za pogled, po možnosti bolj pomirjujočih barv. Prilagojen naj bo za čim hitrejše delo.

3.2.2. Funkcionalne zahteve

Funkcionalne zahteve bomo predstavili s pomočjo diagrama primerov uporabe, podroben opis in predpogoje pa bomo opisali tekstovno s pomočjo kartic z zgodbami (ang. story card).

Kartice z zgodbami po navadi napiše naročnik sistema. Vsaka kartica vsebuje določeno funkcionalnost sistema, ki jo je potrebno realizirati. Pred opisom kartic z zgodbami, moramo predstavili akterje sistema.

Akterji sistema so:

- Uporabnik: neprijavljen obiskovalec spletne strani,
- Urednik: prijavljen obiskovalec v uredniški vmesnik z določenimi pravicami,
- Administrator : prijavljen obiskovalec v uredniški vmesnik, ki lahko določa pravice.

Funkcionalnost sistema za uporabnika velja tudi za vse druge akterje, ravno tako velja funkcionalnost urednika tudi za administratorja, torej velja generalizacija po navedenem vrstnem redu akterjev.

1. KARTICA
Uporabnik lahko pregleduje spletne strani, ki so jih naredili uredniki.

2. KARTICA
Urednik ima možnost registracije, preko katere dobi veljaven uporabniški račun. Pri tem mora podati ime, priimek, uporabniško ime, elektronski naslov ter geslo.

3. KARTICA
Po uspešni registraciji se uporabnik lahko prijavi v sistem. Za prijavo potrebuje uporabniško ime in geslo.

4. KARTICA
Uredniki in administratorji imajo možnost odjave iz sistema. To preprečuje možnost zlorabe njihovih računov.

5. KARTICA
Uredniku in/ali administratorju se po prijavi izriše njegovo delovno okolje, v katerem so vsi meniji, na katerih ima pravice. Na podlagi pravic se omogočijo gumbi za dodajanje, urejanje ali brisanje menija ali dokumenta.

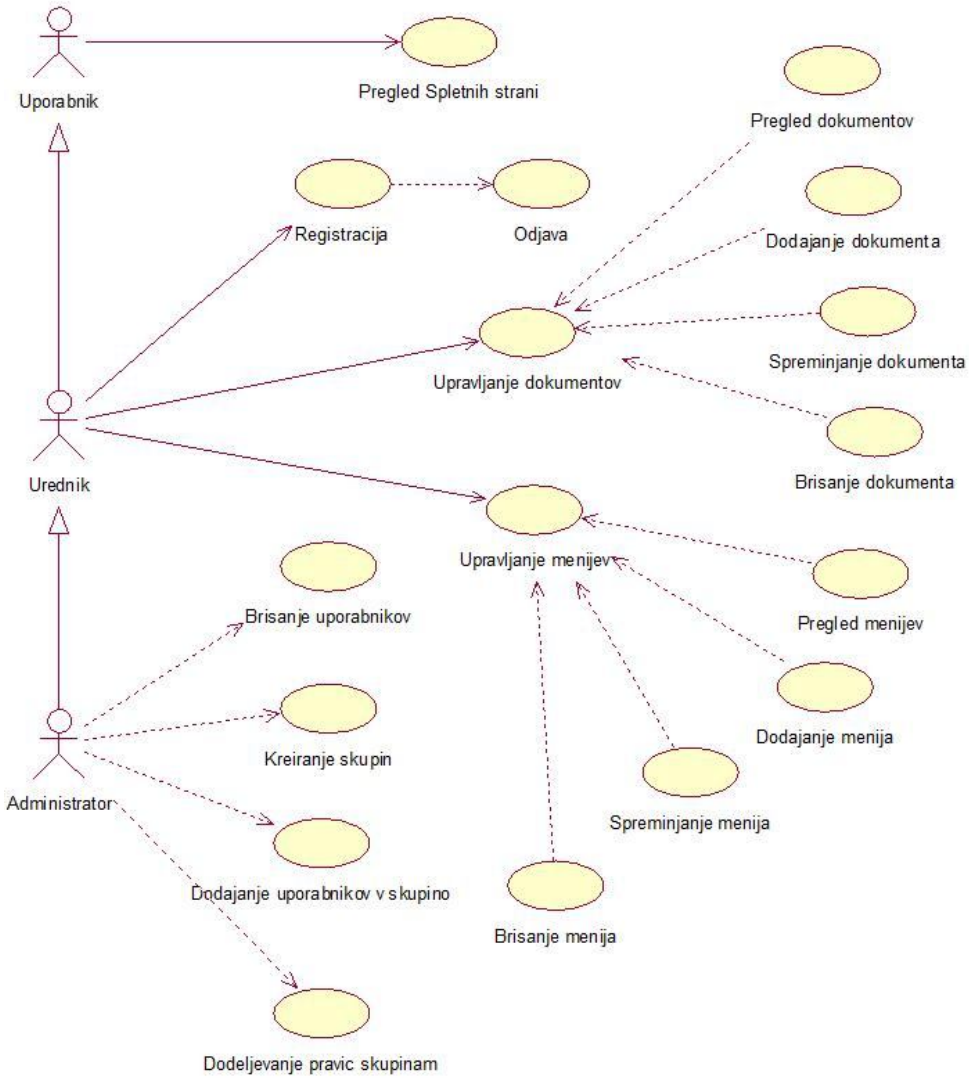
6. KARTICA
Administrator lahko dostopa do posebnega modula, kjer ima možnost izbrisati uporabnika (urednika), narediti nove skupine, razvrščati uporabnike po skupinah ter skupinam dodeljevati pravice.

7. KARTICA
Urednik lahko (na podlagi pravic) naredi nov dokument ali meni, doda dokument meniju, spreminja ali briše stare dokumente ali menije.

8. KARTICA
Pri dodajanju dokumenta mora urednik poleg vsebine dokumenta navesti tudi naslov ter določiti ali naj bo dokument objavljen ali ne.

9. KARTICA
Pri dodajanju menija mora urednik podati naslov menija, zaporedno številko (služi za prikaz po vrstnem redu), morebitni fizični naslov strani, veljavnost strani ter izbrati jezik.

Za lažjo predstavitev si poglejmo diagram primerov uporabe (ang. use case diagram) (slika 10).



Slika 10: Diagram primerov uporabe Uredniškega vmesnika

Prioriteta realizacije posameznih kartic glede na izdaje je prikazana na spodnji sliki (slika 11). V prvi izdaji naredimo registracijo urednika v sistem, prijavo ter odjavo iz sistema. V drugi izdaji izdelamo administratorski modul z možnostjo razvrščanja uporabnikov ter dodeljevanje pravic. Tako lahko v tretji izdaji izvedemo izris uporabniškega okolja, ki ga v četrti izdaji

dopolnimo s funkcionalnostmi. Peta izdaja pa je izven meja sistema in služi za izdelavo prototipa prikaza delujoče spletne strani, ki jo sestavimo. Peta izdaja je lahko individualna za vsako stranko.

IZDAJA	KARTICA
1.	2, 3, 4
2.	6
3.	5
4.	7,8,9
5.	1

Slika 11: Prioriteta realizacije posameznih kartic glede na izdajo

3.2.3. Podatkovni model sistema

Podatkovni model zajema 8 entitet ter 40 shranjenih procedur. Vsaka podatkovna entiteta vsebuje primarni ključ, ki se samodejno povečuje in je v relacijah z drugimi entitetami. Vsaka entiteta ima vsaj štiri shranjene procedure, ki omogočajo osnovne operacije nad podatki (branje, dodajanje, urejanje in brisanje).

Slika 12: Relacijski podatkovni model sistema

Fizični podatkovni model lahko naredimo s pisanjem skripte (priloga a) ali pa naredimo relacijski podatkovni model s pomočjo diagrama (slika 12). V drugem primeru se fizični podatkovni model avtomatsko kreira s pomočjo preprostega uporabniškega vmesnika.

Podatkovni entiteti meni (`sys_meni`) in dokument (`sys_dokument`) sta hrbtenica sistema za upravljanje s spletnimi vsebinami. Vsebujeta tako vsebinske kot povezovalne atribute, preko katerih se povezujeta s podatki v drugih tabelah.

Vsi atributi, razen prvega, ki se začnejo s predpono `id_`, predstavljajo dejanske povezave, oziroma tuje ključe na druge podatkovne entitete.

Vsebovani atributi entitete `sys_meni`:

- `id_meni:` primarni ključ, ki služi kot identifikacijska številka menija,
- `id_meni_prej:` predhodni meni (identifikacijska št. menija),
- `id_jezik:` jezik,
- `zap_st:` vrstni red menija na istem nivoju,
- `naslov:` naslov menija,
- `lcUserId:` identifikator uporabnika, ki je zadnji spremenil meni,
- `lcDatum:` datum zadnje spremembe,
- `ts`: timestamp (časovna oznaka tvorbe),
- `datum_od:` začetni datum objave menuja,
- `datum_do:` končni datum objave menuja,
- `url:` url naslov.

Entiteta `sys_meni` vsebuje povezave s:

- `sys_dokument:` vsebuje podatke o dokumentih,
- `sys_meni:` povezava sama nase,
- `sys_pravice:` podatki o pravicah za določen meni (dodajanje, brisanje, urejanje, branje),
- `sys_jezik:` jezik.

Vsebovani atributi entitete `sys_dokument:`

- `id_dokument:` primarni ključ, ki služi kot identifikacijska številka dokumenta,
- `id_meni:` meni,
- `dokument:` vsebina dokumenta,
- `naslov:` naslov dokumenta,
- `objavi:` ali naj bo dokument viden zunanjemu uporabniku,
- `id_uporabniki:` ključ uporabnika, ki je kreiral dokument,
- `lcUserId:` ključ uporabnika ki je zadnji spreminjal dokument,

- `lc:` datum zadnje spremembe,
- `ts:` timestamp (časovna oznaka tvorbe)..

Entiteta `sys_dokument` vsebuje povezave s:

- `sys_uporabniki:` vsebuje podatke o uporabnikih (kdo je kreiral in kdo je zadnji spremenil dokument),
- `sys_meni:` vsebuje podatke o menijih.

3.3. Izgled in delovanje

Aplikacija je razdeljena na dva dela in sicer na administratorski in uredniški modul. Pri tem je uredniški modul namenjen kreiranju spletnih strani, administratorski pa dodeljevanju pravic urednikom. Za prikaz spletne strani smo razvili dodatno aplikacijo oziroma njen prototip. Ta aplikacija ni del končnega sistema, služi nam samo za prikaz narejenega. Predvidena je prilagoditev posameznim željam uporabnikov (tu je mišljena predvsem postavitev strani in izgled sam). Za aplikacijo je predvidenih še več nadgradenj, v končnih fazah razvoja pa razširitev in avtomatizacija prototipa tako, da ga ne bo več treba prilagajati temveč bo imel uporabnik grafični vmesnik in bo tako sam skrbel za postavitev strani, na voljo mu bo tudi nekaj primerov postavitev in dizajnov.

3.3.1. Prijava

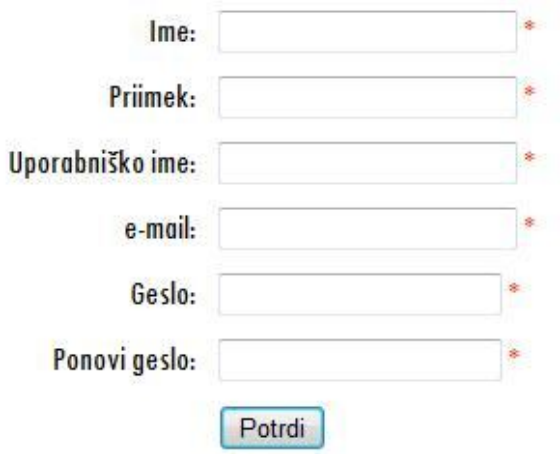
Za prijavo v sistem (slika 13) je potrebna registracija. Pri prijavi v sistem potrebujemo uporabniško ime in geslo. Imamo možnost tudi odkljukati polje »zapomni si me«, pri čemer se ob prijavi pošlje piškotek uporabniku in se ob naslednji prijavi iz njega prebere uporabniško ime.

Slika 13: Vnosni obrazec za prijavo v sistem

Prijava je uspešna, ko se nahaja v podatkovni bazi natanko eno uporabniško ime in geslo, ki ga je uporabnik vpisal v vnosni obrazec za prijavo v sistem.

3.3.2. Registracija

Registracija je namenjena kreiranju novega uporabniškega računa. Pri registraciji (slika 14) mora uporabnik vnesti ime, priimek, uporabniško ime, elektronsko pošto in geslo.



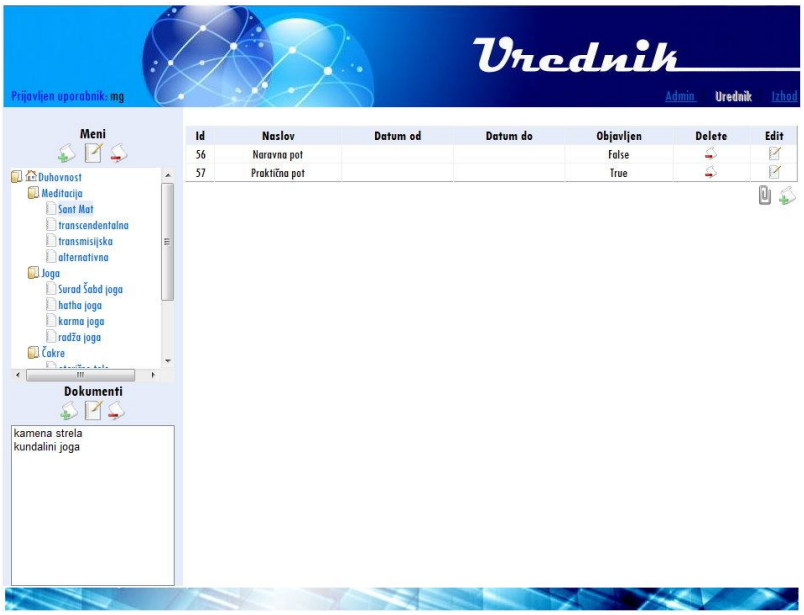
Slika 14: Vnosni obrazec za registracijo uporabnika

Kot je razvidno iz slike 14 mora uporabnik geslo vtipkati dvakrat. Tako preprečimo, da bi se uporabnik zatipkal pri vnosu gesla. Pri registraciji lahko pride do napak. Sporočila o možnih napakah:

- sporočilo o napaki v primeru manjkajočega vnosa kateregakoli vnosnega polja,
- sporočilo o napaki zaradi neustrezne dolžine uporabniškega imena in gesla,
- sporočilo o napaki zaradi napačnega elektronskega naslova,
- sporočilo o napaki zaradi zasedenega uporabniškega imena in
- sporočilo o napaki zaradi neujemanja gesel.

3.3.3. Delovno okolje

Po uspešni prijavi v sistem se iz baze preberejo vse pravice, ki so dodeljene skupini ali skupinam urednika. Na podlagi teh pravic se izriše delovno okolje (slika 15). Na levi strani so meniji (nad katerimi ima uporabnik pravice) ter vsi dokumenti, ki niso vezani na noben meni. Ob kliku na meni se prikažejo podatki o vseh vsebovanih dokumentih.



Slika 15: Delovno okolje

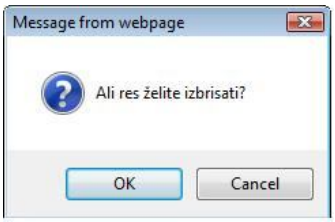
3.3.4. Izdelava in urejanje menijev

Pravice so dodeljene vsakemu vozlu menija posebej in določajo osnovne operacije (branje, dodajanje, urejanje in brisanje). Uporabnik vidi meni, če ima pravico za branje. Ob kliku na določen meni ima tako uporabnik s pravicami za branje, dodajanje in urejanje, izklopljen gumb za brisanje, ostale pa vklopljene (slika 16) in (slika 17).

Slika 16: Vklopljen gumb Spremeni meni

Slika 17: Izklopljen gumb Izbriši meni

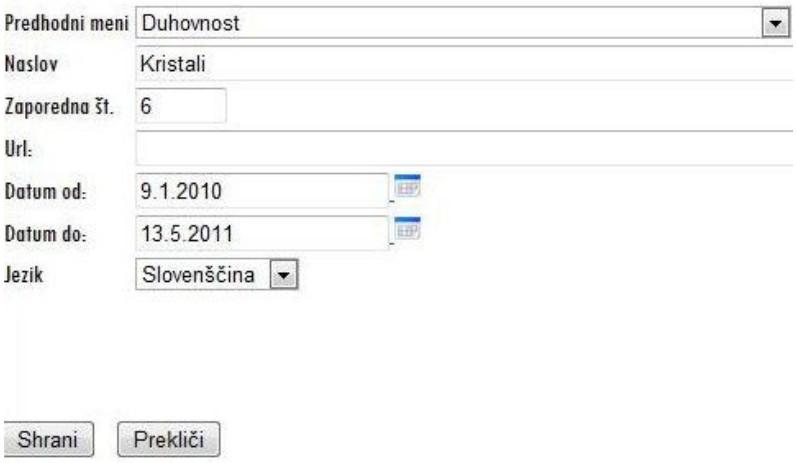
Pri brisanju menija ob kliku na gumb »`izbriši meni`« nas sistem opozori in hoče potrditev izbrisa (slika 18). Po potrditvi se meni izbriše in njegov priklic ni več možen. Podobno opozorilo se prikaže tudi ob brisanju dokumentov (velja isti princip brisanja).



Slika 18: Potrditev za brisanje menija

Pri izdelavi novega menija (slika 19) ob kliku na gumb `Nov meni` določimo:

- `predhodni meni` (očeta vozla),
- `naslov` menija,
- `zaporedna št:` zaporedna številka, ki nam služi za vrstni red (v primeru, ko ima oče več sinov),
- `url` spletne strani (ko želimo, da nam meni služi kot povezava na drugo spletno stran),
- `datum_od` (začetni datum objave menija)
- `datum_do` (končni datum objave menija)
- `jezik` (možnost prikaza spletne strani v več jezikih)



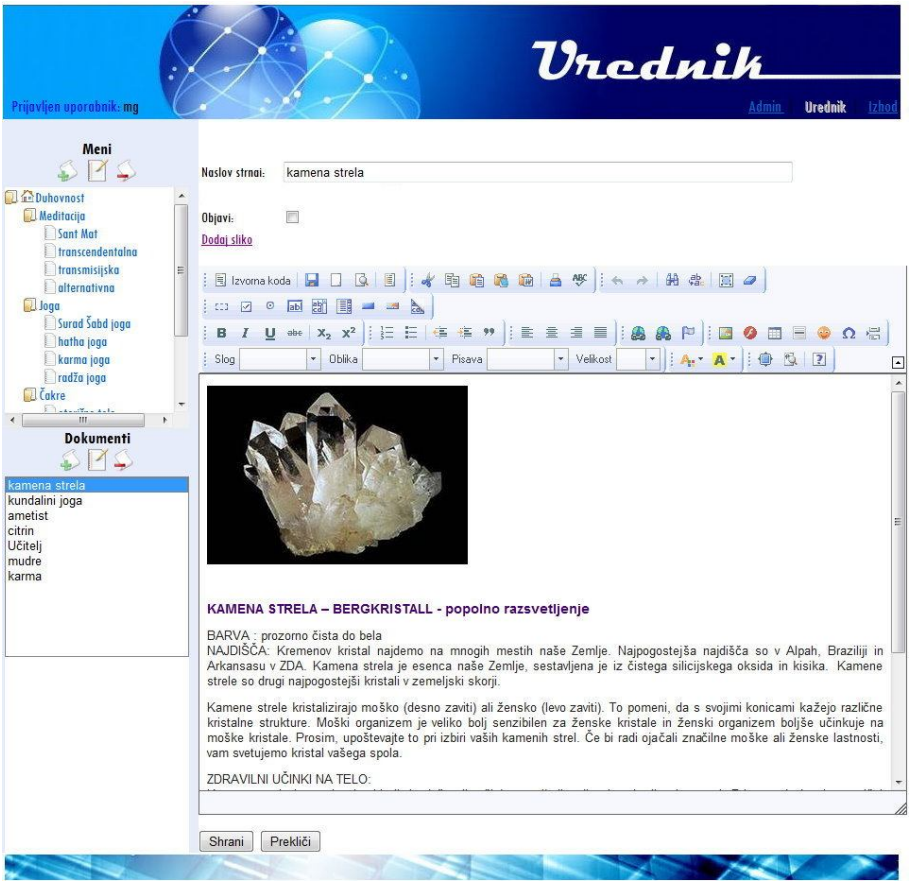
Slika 19: Dodajanje novega menija

Z `datumom_od` in `datumom_do` smo dosegli, da se lahko strani avtomatsko osvežujejo oziroma prikazujejo nove.

3.3.5. Izdelava in urejanje dokumentov

Dokumente lahko doda v skupno shrambo vsak uporabnik, za dodajanje dokumenta meniju pa mora ta imeti pravice. Dokumenti, ki so dodani meniju, podedujejo pravice menija. Tako lahko uporabnik dodaja dokumente in podmenije le menijem nad katerimi ima pravice. Enako velja za urejanje in brisanje. Dokumente lahko izdelamo na meniju, ali pa kot samostojne dele in jih kasneje pripnemo meniju. Ob kliku na nov dokument se nam odpre okno za izdelavo dokumenta (slika 20). Potrebno je napisati naslov in odkljukati, če želimo, da bo dokument viden in bo pri pregledovanju spletnih strani prikazan ob kliku na meni. Tako je na en meni lahko vezanih več dokumentov, samo en dokument pa je viden za določen meni. S tem dosežemo, da pri preurejanju strani pripravimo novo stran in jo vklopimo, ko nam stara ne

služi več. Sam dokument izdelamo v CKEditorju, ki nam omogoča enostavno izdelovanje in urejanje dokumenta. Za dodajanje slike je potrebno vpisati url naslov, zato imamo tudi možnost dodati sliko na strežnik. Dokument pripnemo meniju tako, da izberemo meni in dokument ter kliknemo na gumb `Pripni` (sponka). Dodajanje dokumenta direktno v meni izvedemo s klikom na `Nov Dokument` v desnem kotu (slika 15).



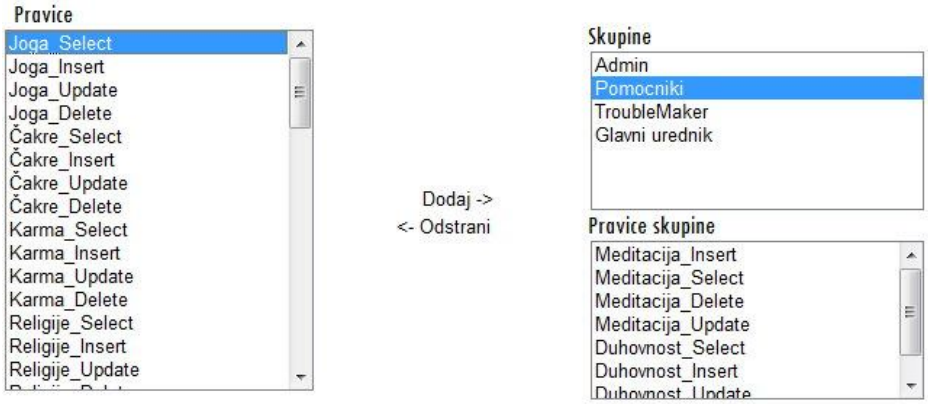
Slika 20: Dodajanje dokumenta

3.3.6. Dodeljevanje pravic

Uporabniki skupine `Admin`, imajo v zgornjem desnem kotu povezavo do administratorskega vmesnika (slika 21). Tu lahko kreiramo skupine ter le tem dodajamo pravice, nato pa skupinam dodamo uporabnike. Uporabnik je lahko v več skupinah, ob prijavi bo imel tako vse pravice, ki jih imajo skupine katerim pripada. Tako lahko naredimo določene menije in dokumente za nekatere skupine (uporabnike) nevidne, ali pa jim damo samo pravico za branje ter tako onemogočimo spreminjanje. Nezaželenim uporabnikom lahko administrator tudi izbriše račune.

Slika 21: Administratorski modul

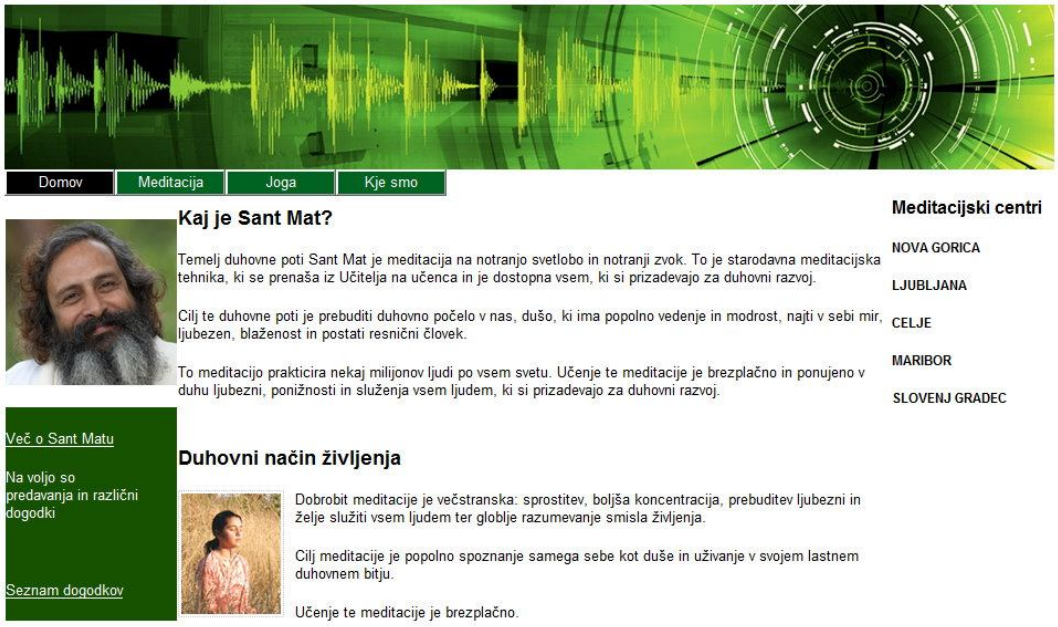
Ob kreiranju novega menija se avtomatsko generirajo pravice za ta meni (za prej naštete operacije). Ime pravice je sestavljeno iz imena menija, podčrtaja in imena operacije. Naloga administratorja je, da dodeli skupini določene pravice (slika 22).



Slika 22: Dodeljevanje pravic skupini

3.4. Prikaz izgleda spletne strani

Prikaz spletne strani je kot prototipna aplikacija (myCms) realizirana samo v ASP.NET različici. Za prikaz delovanja smo izbrali enostaven meni. myCms je sestavljen iz treh komponent, ene za izrisovanje gumbov ostali dve pa za prikaz glave in telesa spletne strani. Vse kar naredimo tu je, da iz baze preberemo in prikažemo domačo stran ter menije (slika 23). Ob kliku na posamezen meni se sproži dogodek, ob katerem iz baze preberemo in prikažemo dokument (spletno stran), ki je vezana na ta meni. Ob tem spremenimo tudi barvo izbranega gumba. Če meni nima dokumenta oz. ima vpisan url naslov, preusmerimo povezavo na ta naslov. Koda za prikaz spletne strani se nahaja v prilogi b.



Slika 23: Prikaz končnega rezultata (izgled narejene spletne strani)

4. IZVEDBA IN PRIMERJAVA MODULOV V ASP.NET IN PHP RAZLIČICI

V skladu z zahtevami je uredniški vmesnik narejen tako v ASP.NET kot v PHP različici. Tu ne gre za nadgradnjo, saj jezika nista kompatibilna. Gre za dve ločeni aplikaciji, ki služita za primerjavo omenjenih programskih jezikov oz. tehnologij. Koda je zaradi večje preglednosti priložena v prilogi. Pred nalaganjem vsake spletne strani se preveri, če je uporabnik, ki zahteva želeno spletno stran, ustrezno prijavljen (če je njegov primarni ključ v seji). V nasprotnem primeru ga preusmerimo na prijavno stran.

4.1. Izdelava drevesa

Najzanimivejša uporabljena kontrola v ASP.NET – u je `TreeView`. Realizacija te kontrole, ki prikaže drevesno strukturo podatkov, je bila v PHP-ju časovno zelo zahtevna.

1. Razlaga kode ASP.NET različice drevesa (priloga c):

- `TreeView()`
  Za prikaz drevesa smo uporabili kontrolo TreeView(). Poznamo njene javne funkcije in atribute, izvedba funkcionalnosti pa nam ni poznana.

- `BindTreeView()`
  Funkcija kreira očete drevesa. Tabelo z vozlišči najprej filtrira, nato pa še sortira. Tako ostanejo tisti vozli, kjer je `id_meni_prej` enak `null` oz. ga ni, sortirani po zaporedni številki. Za vsakega očeta se nato kliče funkcija `BindSubGroups`.

- `BindSubGroups(TreeNode parentNode)`
  To je rekurzivna funkcija, ki generira sinove podanega očeta. Najprej se tabela z vsemi vozli filtrira in sortira, tako da dobimo vse vozle, ki pripadajo danemu očetu. V atributu `parentNode` je shranjena instanca očeta, kateremu dodajamo sinove. Za vsakega sina kličemo rekurzivno to isto funkcijo, da mu potem kot očetu dodajamo sinove.

2. Razlaga kode PHP različice drevesa (priloga d):

- Za izris drevesa na zaslon smo funkcijo malce preuredili, osnova pa ostaja ista.

- `BindNode($dt)`
  Funkcija prejme kot parameter tabelo z vsemi vozli. Podobna je ASP.NET funkciji `BindTreeView()` le da tu vozel izrišemo. Tabela se najprej filtrira, nato pa sortira, tako da ostanejo vsi vozli, ki so na nivoju nič, po vrstnem redu. Sledi izris vozla z echo ukazom in klic funkcije `BindChildNode()`.

- `BindChildNode($dt, $id_meni, $nivo)`
  Podan argument je tabela vozlov `($dt)`, ki se tu spet filtrira in sortira. Naslednji argument funkcije je ključ očeta `($id_meni)`. Tabelo filtriramo s tem ključem tako, da ostanejo samo otroci podanega očeta. Sledi izris zamikov glede na argument `$nivo`, nato pa v zanki izris sinov. Za vsakega sina se rekurzivno spet kliče tale funkcija, kjer nastopa kot oče.

4.2. Nov meni

Ob kliku na gumb Nov meni se odpre vnosna stran za vnos menija (slika 19). Za koledar je bila v ASP.NET-u uporabljena kontrola `Calendar`, v kateri lahko izbereš datum. V PHP različici pa je potrebno datum ročno vnesti.

1. Razlaga kode ASP.NET različice izdelave novega menija (priloga e):

- `FillDdlMeni(idParent)`
  Je funkcija, ki ob prvem nalaganju strani napolni padajoči meni z vsemi vozli, ki jih imamo na razpolago (to so tisti katerim so dodeljene pravice). Prednastavljen vozel je tisti, ki je bil izbran pred klikom na `Nov Meni (idParent)`. Funkcija je skupna tako za izdelavo novega, kot za urejanje predhodno narejenega menija. Pri izdelavi novega menija funkcija vrne podatke o očetu, v primeru urejanja pa podatke o meniju, ki se ureja.

- `FillDdlJezik()`
  Podobno kot prejšnja funkcija tudi ta napolni padajoči meni za izbiro jezika za katerega hočemo izdelati spletno stran.

- Za pravilno prikazovanje kontrole `Calendar` v `div` odseku, je bilo treba omogočiti vidljivost koledarja s pomočjo JavaScript kode ob nalaganju strani. Problem, ki ni bil rešljiv samo z `.aspx` kodo, je nastal, ker se je, ob kliku na naslednji mesec, koledar zaprl in je bil tako potreben dodaten klik, da se spet odpre. Za določanje vidljivosti smo tako uporabili dogodka: `calOd_SelectionChanged(object sender, EventArgs e)` in `calOd_VisibleMonthChanged(object sender,`

`MonthChangedEventArgs e)`. Tako smo vidljivost s pomočjo javascript kode določali s tema dogodkoma.

- `btnSave_Click(object sender, EventArgs e)`
  Dogodek se proži po vnosu potrebnih podatkov (glej Izdelava in urejanje menijev) ob kliku na gumb shrani. Tu se najprej preveri, če je vneseno obvezno polje naslov, nato pa se preberejo ostali podatki in vpišejo v bazo. V nasprotnem primeru pa dobimo opozorilo, da nismo izpolnili obveznega polja.

2. Razlaga kode PHP različice izdelave novega menija (priloga f):

- `FillMeni()`
  Podobno kot v ASP.NET različici se ob nalaganju strani najprej napolni padajoči meni z vozli, nad katerimi imamo pravice. Ker sta vnos novega menija in urejanje menija dve ločeni datoteki, se njihova koda tu ne prepleta.

- `FillJezik()`
  Funkcija napolni padajoči meni `jezik` z vsemi jeziki, ki so v prisotni bazi.

- Ob kliku na gumb `Shrani`, se zgodi metoda POST, ki nas s pomočjo HTML povezave preusmeri na skripto `ShraniMeni.php`. Tu se najprej preveri vnos obveznega polja, naslov in v primeru, da je le to izpolnjeno, se podatki shranijo v bazo.

4.3. Urejanje dokumenta

Po izbiri dokumenta v seznamu lahko s klikom na `Spremeni dokument`, uredimo izbrani dokument.

1. Razlaga kode ASP.NET različice urejanja dokumenta (priloga g):

- `ibEditDokument_Click(object sender,ImageClickEventArgs e)`
  Dogodek se sproži ob kliku na `Spremeni dokument`. Tu se najprej preveri, če je izbran dokument za urejanje. Če ni izbran, vrne opozorilo, v nasprotnem primeru se shrani primarni ključ dokumenta v sejo in s pomočjo HTML povezave preusmerimo na `wfDokument.aspx` spletno stran.

- `wfDokument.aspx`
  Tu se prebere ključ dokumenta in morebitni ključ menija, na katerega je vezan. Sledi poizvedba na bazo ter prikaz podatkov tega dokumenta. Stran ima vnosno polje za naslov, možnost potrditve objave dokumenta in povezavo na stran, kjer lahko dodamo sliko na strežnik. Na spodnjem delu strani je dodano tekstovno polje, katero se s pomočjo JavaScript-a prikaže kot CKEditor. Tu lahko spreminjamo vsebino dokumenta. Želene spremembe shranimo s klikom na gumb `Shrani`, ali pa prekličemo urejanje z gumbom Prekliči. V obeh primerih se izvede preusmeritev na `wfUrednik.aspx` (slika 15).

2. Razlaga kode PHP različice urejanja dokumenta (priloga h):

- Po kliku na gumb "Uredi dokument" s pomočjo javascript kode in HTML povezave prikažemo spletno stran `UrediDokument.php`. Parameter `id_dokument` (primarni ključ dokumenta) prenesemo s pomočjo `GET` metode.

- `UrediDokument.php`
  Pred nalaganjem strani pregledamo, če se je ključ dokumenta uspešno prenesel. Po branju ključa izvedemo povpraševanje na bazi, ki nam vrne zahtevan dokument. Podatke dokumenta prikažemo v vnosnih poljih, katere je možno urejati. Po kliku na gumb `Shrani` s pomočjo `POST` metode prenesemo podatke v skripto `ShraniDokument.php` kjer podatke preberemo in jih nato shranimo v bazo. Po uspešnem shranjevanju preusmerimo povezavo na začetno stran. V nasprotnem primeru se med preusmerjanjem prikaže napaka.

5. PRIMERJAVA TEHNOLOGIJ

Primer prve vrstice v datoteki .aspx:

```
<%@ Page Language="C#" AutoEventWireup="false"
CodeBehind="wfUrednik.aspx.cs" Inherits="Urednik.wfUrednik"
MasterPageFile="~/mpUrednik.Master" %>
```

Prva vrstica nam pove, da uporabljamo jezik C#, da imamo vklopljene dogodke, da se koda, ki se izvaja v ozadju, nahaja v razredu (datoteki) `wfUrednik.aspx.cs`, da stran deduje iz strani `Urednik.wfUrednik`. Navedena je tudi glavna stran (master page), kar nam pove, da je to vsebovana stran (content page).

Primer enostavne PHP skripte:

```
<html>
<body>

<?php
echo "Hello World";
?>

</body>
</html>
```

PHP skriptni blok se začne z oznako `<?php` konča z `?>`. Na zgornjem primeru PHP skripta pošlje brskalniku besedilo `Hello World`, kateri ga izpiše.

ASP.NET nam z dogodki (ang. events) lahko prihrani ogromno dela s pisanjem javascript kode. Slabost tega je, da se vsa koda ASP.NET izvaja na strežniku in tako, ko ne potrebujemo podatkov iz baze, po nepotrebnem obremenjujemo strežnik.

PHP ni dogodkovno usmerjen, njegova objektna usmerjenost pa nam omogoča zgraditi strukturo, ki simulira dogodke kot so v ASP.NET tehnologiji.

5.1. Shranjevanje lastnosti kontrol

ASP.NET ob vsakem post dogodku shrani lastnosti kontrol v skrito polje `__VIEWSTATE`. Tako nam omogoči ob ponovni osvežitvi strani (postback dogodku) osvežitev samo želenih kontrol in vrednosti, izris ostalih kontrol z vrednostmi ostane enak. To lastnost lahko vklopimo ali izklopimo tako na nivoju celotne strani kot za vsako kontrolo posebej.

PHP nima podobne funkcionalnosti oz. jo mora programer sam razviti.

5.2. Kontrole

PHP razvojna orodja ne vsebujeno kontrol. Za izdelavo osnovne forme Uredniškega vmesnika v PHP različici, smo tako uporabili HTML frameset tehnologijo, s katero lahko prikažemo več spletnih strani v oknu brskalnika. V ASP.NET tehnologiji pa smo osnovno formo izdelali s pomočjo glavne spletne strani (ang. Master Page).

Glavna spletna stran je v svojem bistvu kontrola, ki nam omogoča, da na enem mestu definiramo izgled in obnašanje spletne strani, ki je skupno več spletnim stranem. Tako glavna spletna stran prinaša svoje prednosti v povezavi z vsebovanimi stranmi (ang. content pages). Na prvi pogled se uporaba glavne spletne strani zdi kot uporaba frameset tehnologije, vendar pa je bistvena razlika, ki ju loči ta, da se vsakokrat pri osvežitvi katerekoli vsebovane strani še enkrat naloži celotna spletna stran tj. glavna stran s podstranmi.

Vsakokratno osveževanje glavne strani je slabost glavne strani. To naredi uporabo glavne strani primerno za bolj statične strani, kjer se ob spremembi podstrani glavna stran ne spreminja.

5.3. Objektno programiranje

Objektno lahko programiramo tako ASP.NET kot v PHP-ju, vendar je uporaba OOP različna. Ne ločita ju samo sintaksa, ampak tudi osnovni koncepti npr. prekrivanja (ang. overloading). PHP ne ponuja "klasičnega" prekrivanja, kot smo ga vajeni z istim imenom metode in različnim številom in tipom argumentov. Namesto tega ponuja magične metode (ang. magic methods), ki se prožijo, ko kličemo nedostopne funkcije. Te metode služijo kot nekakšen ovoj (ang. wrapper) prekrite funkcije oz. za preusmeritev na drugo funkcijo. Sama prekrita funkcija ima tako drugo ime in različne parametre, njen klic pa izvedemo v magičnih metodah, ko ugotovimo, da je prišlo do klica te prekrite metode. Tako je doseženo, da lahko kličemo isto funkcijo z različnimi parametri, čeprav je funkcija s tem imenom lahko samo ena, klice z drugimi parametri pa preusmerimo na ostale funkcije.

5.4. Namestitev

Z namestitvijo obeh tehnologij in orodij ni bilo težav.

5.5. Sintaksa

Sintaksa je pri ASP.NET-u lažja za pisanje, kar nam omogoča tudi hitrejše kodiranje. Pri PHP-ju vidim problem predvsem v pretirani uporabi posebnih znakov. Prednost in hkrati slabost PHP-ja je, da pri deklaraciji spremenljivke ni potrebno definirati tipa. Tako je lahko razvoj hitrejši, pri obširnejših aplikacijah z uporabo objektno usmerjenega programiranja pa lahko pride do problemov, ker ne vemo kaj spremenljivka hrani. Tudi .NET tehnologija nam omogoča, da deklariramo spremenljivko kot `"var"`. Kar pomeni, da tip ni določen. V to spremenljivko lahko shranimo kakršenkoli tip ali objekt. Vendar pa `"var"` spremenljivka vedno generira tipizirano referenco (ang. strongly typed reference). Ob prevajanju kode, se tip spremenljivke določi glede na prvo inicializacijo spremenljivke.

Poglejmo si nekaj primerov sintakse:

ASP.NET deklaracija spremenljivk:

```
int stevilo = 1;
```

PHP deklaracija spremenljivk:

```
$stevilo = 1;
```

ASP.NET klicanje metode objekta, ter statične metode

```
objekt.metoda(parametri);
Razred.metoda(parametri);
```

PHP klicanje metode objekta, ter statične metode

```
$objekt->metoda(parametri);
Razred::metoda(parametri);
```

5.6. Prevajanje (ang. compilation)

PHP

Ko je obravnavana zahteva PHP strani, se HTML in vsebovana PHP skripta prevedeta v Zend Optokodo. Optokoda so nižje nivojska binarna navodila, ki se uporabljajo pri serviranju PHP strani. Po prevajanju Zend motor zažene optokodo, generira se HTML in servira odjemalcu.

Za izboljšanje performans PHP skript se uporablja predpomnjenje (ang. caching).

ASP.NET

Ko se zahteva `.aspx` stran, se zahteva pošlje ASP.NET-u za obdelavo (procesiranje). Ob prvem zahtevku strani ta prevede stran v MSIL (ang. Microsoft intermediate language). MSIL kodo nato predela CLR (ang. Common language runtime) v strojno kodo in zahteva je pognana z uporabo te prevedene kode. Naslednje zahteve so servirane iz iste strojne kode s predpostavko, da stran ni bila spremenjena.

5.7. Razhroščevanje

Visual Studio in Visual Web Developer Express imata že vgrajeno orodje za razhroščevanje (ang. debugging). Zaradi predhodnega programiranja v Visual Studiu nam je tudi uporaba tega enostavna in intuitivna. Razhroščevalnik PHP- ja je potrebno dodatno namestiti.

Čas namenjen za izbiri ustreznega razvojnega okolja, namestitvi in učenju uporabe sem raje (zaradi ne prevelike obsežnosti aplikacije) namenil razhroščevanju na star način z uporabo »`echo`« stavkov. Tako sem izpisoval vrednosti spremenljivk in pozicijo v skripti. Problem so mi delale sintaktične napake, katere je bilo včasih težko odkriti, saj mi je pri prikazovanju PHP strani vedno prikazalo isto sporočilo.

5.8. Avtomatsko generiranje kode

Za razvoj aplikacije v PHP-ju smo uporabili Notepad++, ki pomaga pri kodiranju z avtomatskim dokončanjem klica funkcije s klikom na `Ctrl + presledek`. Podobno ponujata tudi Visual Studio in Visual Web Developer Expess, kjer nam intelisense (Microsoftovo orodje za avtomatsko dokončanje) ponuja precej obširnejše možnosti, ki se lahko primerjajo z ostalimi IDE (Integrated Development Environment) orodji za razvoj PHP strani.

Pri pisanju kode v ASP.NET-u se avtomatsko generira cel kup kode. To je lahko prednost, ker nam pohitri programiranje, lahko je tudi slabost, ker se lahko generirajo nepotrebni deli kode in je tako koda bolj zahtevna.

5.9. Dokumentacija

Tako za ASP.NET kot za PHP je na internetu objavljeno ogromno primerov uporabe. Microsoftova uradna spletna stran z dokumentacijo pa se mi zdi veliko bolj izpopolnjena kot PHP-jeva, zato ker ponuja poleg obsežne dokumentacije, primerov uporabe, forumov in

blogov tudi video predavanja. Večino rešitev za probleme v PHP-ju sem dobil na forumih in na PHP uradni strani. Za kodiranje v ASP.NET-u pa sem se poslužil video predavanj. Tako sem se izognil večini problemov ali pa zanje dobil rešitve. Na splošno je dokumentacija za ASP.NET bolj izpopolnjena in lažje dojemljiva za navigacijo. Ta mi je pri PHP ju povzročala sitnosti.

5.10. Ugotovitve

V tem zadnjem delu diplomske naloge bomo podali ugotovitve. Primerjali bomo obe tehnologiji in ocenili orodja, ki smo jih uporabili pri izdelavi. To bomo prikazali z oceno njihovih prednosti in slabosti.

5.10.1. Ocena uporabljenih orodij

Večina uporabljenih tehnologij in orodij, ki so nam precej olajšala razvoj aplikacije, se ni razlikovala med ASP.NET in PHP izvedbo.

V obeh primerih smo uporabili:

- za podatkovno bazo in upravljanje s podatki : Microsoft SQL Server 2008 Express,
- za modeliranje: Rational Rose,
- za pregledovanje dogodkov na Microsoft SQL serverju: AnjLab SQLProfiler,
- za omogočanje izdelave spletne strani CKEditor,
- za grafiko: Photoimpact 12,
- spletni strežnik IIS.

Različno pri obeh verzijah je bilo razvojno okolje. V primeru ASP.NET je bil to Visual Studio 2008, v primeru PHP različice pa smo uporabili Notepad++. Uporabljena orodja so se izkazala za pravilno in učinkovito izbiro. Nadomestil bi le orodje Notepad ++ z Eclipse PDT ali enim izmed programov za izdelavo spletnih aplikacij (Dreamweaver, FrontPage).

5.10.2. Prednosti in slabosti tehnologij

Tako s PHP kot z ASP.NET tehnologijo lahko naredimo obširnejše spletne aplikacije. Če se odločamo med eno in drugo tehnologijo je dobro vedeti predvsem namen uporabe, obseg ter določiti infrastrukturo aplikacije.

PHP prednosti:

- deluje lahko na večini operacijskih sistemov,
- lažje razumljiv za začetnike,
- brezplačna uporaba,
- veliko literature in pomoči na internetu,
- podpira večina spletnih strežnikov in
- veliko razvojnih orodij.

PHP slabosti:

- skromen opis napak, če se le te pojavijo,
- slaba podpora za objektno usmerjeno programiranje,
- zbirke podatkov (kot je npr. DataSet),
- izbira pravega razvojnega orodja in
- razvojna orodja ne vsebujejo kontrol.

ASP.NET prednosti:

- brezplačna uporaba,
- veliko literature in dokumentacije ter pomoči na internetu,
- dobro razvojno okolje s podporo več programskih jezikov,
- enostavno razhroščevanje in dober opis napak,
- veliko zbirk podatkov,
- dober sistem za avtomatsko dokončanje kode (ang. IntelliSense),
- dobra podpora za objektno usmerjeno programiranje,
- shranjevanje lastnosti kontrol in
- paleta kontrol.

ASP.NET slabosti:

- deluje samo na Microsoftovih operacijskih sistemih,
- dogodkovni razvoj je za začetnike težje razumljiv in
- podpirajo ga le Microsoftovi strežniki in Apache.

6. ZAKLJUČEK

Pri razvoju spletne aplikacije ni bilo bistvenih težav.

V diplomski nalogi so predstavljene tehnologije, orodja in smernice, ki jih programer potrebuje pri razvoju spletnih aplikacij.

Cilj diplomskega dela je bil, poleg primerjave obeh programskih jezikov oziroma tehnologij, razviti tudi konkretno aplikacijo. Spletna aplikacija za upravljanje s spletnimi vsebinami (WCMS) je namenjena vsem, ki se želijo predstaviti na spletu. Pri tem je izdelava in urejanje strani prepuščena uporabnikom samim.

Programiranje z ASP.NET tehnologijo se je izkazalo za bistveno lažje. Temu je botrovalo enostavno razhroščevanje ter dober opis napak, Microsoftova podpora za morebitna vprašanja ter dobra dokumentacija s primeri uporabe na Microsoftovi uradni strani, kjer lahko najdemo koristne informacije za tehnologijo ASP.NET.

Za razvoj aplikacije Urednik smo uporabili predvsem Microsoftove tehnologije. Velja poudariti, da v kolikor bi se odločili za razvijanje v PHP-ju, bi najverjetneje izbrali različni podatkovni ter spletni strežnik. PHP se večinoma uporablja v integraciji s spletnim strežnikom Apache in podatkovnim strežnikom MySQL. Privrženci odprtokodnih rešitev pa bi uporabili tudi različen operacijski sistem.

Čeprav smo se trudili objektivno oceniti tehnologiji, ocena mogoče ni prava, saj imamo mnogo več izkušenj s programiranjem v jeziku C#. Pridobljene izkušnje nam bodo pomagale tudi pri nadaljnji karierni odločitvi.

V nadaljevanju so podane določene ideje za izboljšavo spletne aplikacije. Nekatere so bile predvidene za realizacijo, vendar je zmanjkalo časa, do drugih pa smo prišli med samim razvojem. Predlagane izboljšave:

- možnost izdelave leve in desne pasice,
- izboljšave na aplikaciji za prikaz, kot so izbira in postavitev menija (navaden in/ali drevesni) ter postavitev pasic,
- potrditev registracije preko elektronske pošte,
- dodajanje novih jezikov,
- urejanje uporabniškega profila in
- avtomatsko dodeljevanje pravic uporabniku. Ker se z dodeljevanjem pravic porabi preveč časa, bi bilo smiselno omejiti uporabnika na eno skupino. Tako bi lahko ob kreiranju novega menija avtomatsko dodali skupini vse pravice, ki jih ima na predhodnem meniju.

PRILOGE

a) Primer fizičnega podatkovnega modela za tabelo `sys_meni`:

```
USE [UREDNIK]
GO

/****** Object:  Table [dbo].[sys_meni]    Script Date: 01/22/2010 10:24:01 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[sys_meni](
	[id_meni] [int] IDENTITY(1,1) NOT NULL,
	[id_meni_prej] [int] NULL,
	[id_jezik] [int] NOT NULL,
	[zap_st] [int] NULL,
	[naslov] [varchar](100) NOT NULL,
	[lcUserId] [int] NULL,
	[lcDatum] [datetime] NULL,
	[ts] [timestamp] NULL,
	[datum_od] [datetime] NULL,
	[datum_do] [datetime] NULL,
	[url] [varchar](200) NULL,
 CONSTRAINT [PK_sys_meni] PRIMARY KEY CLUSTERED
(
	[id_meni] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]

GO

SET ANSI_PADDING OFF
GO

ALTER TABLE [dbo].[sys_meni]  WITH CHECK ADD  CONSTRAINT [FK_sys_jezik_sys_meni] FOREIGN
KEY([id_jezik])
REFERENCES [dbo].[sys_jezik] ([id_jezik])
GO

ALTER TABLE [dbo].[sys_meni] CHECK CONSTRAINT [FK_sys_jezik_sys_meni]
GO
```

b) .aspx.cs koda za prikaz spletne strani:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Drawing;

namespace myCms
{
    public partial class _Default : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            CreateHeader();
```

```
    CreateMeni();
    CreateBody();
}

void _btnMeni_Click(object sender, EventArgs e)
{
    SetButtonsColor(sender);
    SetPageBody(sender);
}

private void CreateHeader()
{
    DataTable _dtDokument = utils.DbExecute("sys_dokument_select", "naslov", "Glava");
    if (_dtDokument.Rows.Count == 1)
    {
        lblHeader.Text = Convert.ToString(_dtDokument.Rows[0]["dokument"]);
    }
}

private void CreateMeni()
{
    Button _btnMeni = new Button();
    _btnMeni.ID = "0";
    _btnMeni.Click += new EventHandler(_btnMeni_Click);

    _btnMeni.Text = "Domov";
    _btnMeni.BackColor = System.Drawing.Color.FromArgb(00, 99, 33);
    _btnMeni.ForeColor = Color.White;
    _btnMeni.Width = 100;
    Panel1.Controls.Add(_btnMeni);

    DataTable _dtMeni = utils.DbExecute("sys_meni_Select4CMS", null, null);
    string _sort = "zap_st ASC";
    DataRow[] _drArray = _dtMeni.Select(null, _sort);



    foreach (DataRow _dr in _drArray)
    {
        _btnMeni = new Button();
        _btnMeni.ID = Convert.ToString(_dr["id_meni"]);
        _btnMeni.Click += new EventHandler(_btnMeni_Click);

        _btnMeni.Text = Convert.ToString(_dr["naslov"]);
        _btnMeni.BackColor = System.Drawing.Color.FromArgb(00, 99, 33);
        _btnMeni.ForeColor = Color.White;
        _btnMeni.Width = 100;
        Panel1.Controls.Add(_btnMeni);
    }
}

private void CreateBody()
{
    DataTable _dtDokument = utils.DbExecute("sys_dokument_select", "naslov", "Telo");
    if (_dtDokument.Rows.Count == 1)
    {
        lblPage.Text = Convert.ToString(_dtDokument.Rows[0]["dokument"]);
    }
}

private void SetButtonsColor(object sender)
{
    foreach (Control _control in Panel1.Controls)
    {
        if (_control is Button)
        {
            ((Button)_control).BackColor = System.Drawing.Color.FromArgb(00, 99, 33);
        }
    }
    ((Button)sender).BackColor = System.Drawing.Color.Black;
}

private void SetPageBody(object sender)
{
    string _id = ((Button)sender).ID;
    if (_id == "0")
    {
```

```
                DataTable _dtDokument = utils.DbExecute("sys_dokument_select", "naslov",
"Telo");
                if (_dtDokument.Rows.Count == 1)
                {
                    lblPage.Text = Convert.ToString(_dtDokument.Rows[0]["dokument"]);
                }
            }
            else
            {
                DataTable _dtDok = utils.DbExecute("sys_meni_Select4CMS", "id_meni", _id);

                if (_dtDok.Rows.Count == 1)
                {
                    if (_dtDok.Rows[0]["url"] != DBNull.Value && _dtDok.Rows[0]["url"] != null
&& _dtDok.Rows[0]["url"] != "")
                    {
                        Response.Redirect(Convert.ToString(_dtDok.Rows[0]["url"]));
                    }
                    else
                    {
                        lblPage.Text = Convert.ToString(_dtDok.Rows[0]["Dokument"]);
                    }
                }
            }
        }
    }
}
```

c) ASP.NET C# različica drevesa

```
private void BindTreeView()
{
    string _filter = "id_meni_prej is null";
    string _sort = "zap_st ASC";
    if (pDtMeni.Rows.Count > 0)
    {
        DataRow[] _drArray = pDtMeni.Select(_filter, _sort);
        foreach (DataRow _dr in _drArray)
        {
            string _text = _dr["naslov"].ToString();
            int _value = Convert.ToInt32(_dr["id_meni"]);
            TreeNode _tn = new TreeNode(_text, _value.ToString());
            BindSubGroups(_tn);
            this.tvMeni.Nodes.Add(_tn);
            this.tvMeni.DataBind();
        }
    }
}

private void BindSubGroups(TreeNode ParentNode)
{
    string _filter = "id_meni_prej = " + ParentNode.Value;
    string _sort = "zap_st ASC";
    DataRow[] _drArray = pDtMeni.Select(_filter, _sort);
    foreach (DataRow _dr in _drArray)
    {
        string _text = _dr["naslov"].ToString();
        string _value = Convert.ToString(_dr["id_meni"]);
        TreeNode _tn = new TreeNode(_text, _value);
        BindSubGroups(_tn);
        ParentNode.ChildNodes.Add(_tn);
        this.tvMeni.DataBind();
    }
}
```

d) PHP različica drevesa

```
function CreateTree()
{
	$lib = new Sql();
	$conn=mssql_connect($lib->GetServer(),$lib->GetUserName(),$lib->GetPassword());
	if ($conn)
	{
		mssql_select_db($lib->GetDatabase(),$conn)or die( "Unable to select database");

		$sp=mssql_init("sys_meni_SelectWithPermissions",$conn);
		$idUporabniki = $_SESSION['IdUporabnika'];
		mssql_bind($sp,"@id_uporabniki",&$idUporabniki,SQLINT4,FALSE);
		$result=mssql_execute($sp);
		mssql_close($conn);
		$array = CreateArray($result);
		BindNode($array);
	}
	else
	{
		print("Can't connect to database");
	}
}

function BindNode($dt)
{
	$nResult = filterByValue($dt, 'id_meni_prej', NULL);
	$nResult = order_array_num($nResult,'zap_st');

	foreach(array_keys($nResult) as $key)
	{
		echo "     ";
		$meni = $nResult[$key]["id_meni"];
		$naslov = $nResult[$key]["naslov"];
		echo"<a    href='../UrednikPHP/Drevo.php?meni=".$meni."'    target='Middle'><img
src='../UrednikPHP/Icons/folder_16x16.png'>$naslov</a><br>";
		BindChildNode($dt, $meni, 1);
	}
}

function BindChildNode($dt, $id_meni, $nivo)
{
	$nResult = filterByValue($dt, 'id_meni_prej', $id_meni);
	$nResult = order_array_num($nResult,'zap_st');

	foreach(array_keys($nResult) as $key)
	{
		for($i = 0; $i <= $nivo; ++$i)
		{
			echo "     ";
		}
		$meni = $nResult[$key]["id_meni"];
		$naslov = $nResult[$key]["naslov"];
		echo"<a    href='../UrednikPHP/Drevo.php?meni=".$meni."'    target='Middle'><img
src='../UrednikPHP/Icons/folder_16x16.png'>$naslov</a><br>";
		$nivo = BindChildNode($dt, $meni, ++$nivo);
	}
	--$nivo;
	return $nivo;
}
```

e) ASP.NET C# različica izdelave novega menija

```
public partial class wfMeni : System.Web.UI.Page
{
    static string action = string.Empty;

    protected void Page_Load(object sender, EventArgs e)
    {
        if (Convert.ToString(Session["IdUporabnika"]) == string.Empty)
        {
            Response.Redirect("wfLogIn.aspx", true);
        }
        else if (!IsPostBack)
        {
            UrlParameterPasser _urlWrapper = new UrlParameterPasser();
            action = Convert.ToString(_urlWrapper["Action"]);
            string _idParent = Convert.ToString(_urlWrapper["idParent"]);
            Meni _meni = FillDdlMeni(_idParent);
            FillDdlJezik();
            if (action == "Edit")
            {
                if (_meni.Naslov != string.Empty)
                {
                    txtNaslov.Text = _meni.Naslov;
                }
                if (_meni.ZapSt != string.Empty)
                {
                    txtZapSt.Text = _meni.ZapSt;
                }
                if (_meni.DatumDo != null && _meni.DatumDo != Convert.ToDateTime("1.1.0001"))
                    {
                        tbDatumDo.Text = _meni.DatumDo.ToShortDateString();
                    }
                    if (_meni.DatumOd != null && _meni.DatumOd !=
Convert.ToDateTime("1.1.0001"))
                {
                    tbDatumOd.Text = _meni.DatumOd.ToShortDateString();
                }
                if (_meni.Url != string.Empty)
                {
                    tbUrl.Text = _meni.Url;
                }
                if (_meni.Jezik != string.Empty)
                {
                    ddlJezik.SelectedValue = _meni.Jezik;
                }
            }
    }

    private void FillDdlJezik()
    {
        Meni _meni = new Meni();

        DataTable _dt = Lib.utils.DbExecute("sys_jezik_Select", null, null);
        ddlJezik.DataSource = _dt;

        ddlJezik.DataValueField = "id_jezik";
        ddlJezik.DataTextField = "opis";
        ddlJezik.DataBind();
    }

    private Meni FillDdlMeni(string idParent)
    {
        Meni _meni = new Meni();
        string _idMeni = Convert.ToString(Session["idMeni"]);
        string _idUporabnik = Convert.ToString(Session["IdUporabnika"]);
        DataTable _dt = Lib.utils.DbExecute("sys_meni_SelectWithPermissions", "id_uporabniki",
_idUporabnik);

         if (action == "Edit")
         {
             if (_idMeni != string.Empty)
             {
                 for (int _i = 0; _i < _dt.Rows.Count; _i++)
```

```
                {
                    if (Convert.ToInt32(_dt.Rows[_i]["id_meni"]) == Convert.ToInt32(_idMeni))
                    {
                        _meni.Naslov = Convert.ToString(_dt.Rows[_i]["naslov"]);
                        _meni.ZapSt = Convert.ToString(_dt.Rows[_i]["zap_st"]);
                        _meni.Url = Convert.ToString(_dt.Rows[_i]["url"]);
                        _meni.Jezik = Convert.ToString(_dt.Rows[_i]["id_jezik"]);
                        if (_dt.Rows[_i]["datum_od"] != DBNull.Value)
                        {
                            _meni.DatumOd = Convert.ToDateTime(_dt.Rows[_i]["datum_od"]);
                        }
                        if (_dt.Rows[_i]["datum_do"] != DBNull.Value)
                        {
                            _meni.DatumDo = Convert.ToDateTime(_dt.Rows[_i]["datum_do"]);
                        }

                        _dt.AcceptChanges();
                        _dt.Rows.RemoveAt(_i);
                    }
                }
            }
        }
        ddlMeni.DataSource = _dt;


        if (idParent != string.Empty)
            ddlMeni.SelectedValue = idParent;

        ddlMeni.DataValueField = "id_meni";
        ddlMeni.DataTextField = "naslov";
        ddlMeni.DataBind();
        ddlMeni.Items.Insert(0, new ListItem("", ""));
        return _meni;
    }

    protected void btnSave_Click(object sender, EventArgs e)
    {
        Lib.DBParameters _par = new Urednik.Lib.DBParameters();
        if (txtNaslov.Text != string.Empty)
        {
            _par.Add("naslov", txtNaslov.Text, System.Data.SqlDbType.VarChar);
        }
        else
        {
            lblError.Text = "Naslov menija je obvezno polje";
            return;
        }

        _par.Add("lcUserId", Convert.ToInt32(Session["IdUporabnika"]), SqlDbType.Int);
        _par.Add("@lcDatum", DateTime.Now, SqlDbType.DateTime);

        string _url = Convert.ToString(tbUrl.Text);
        _par.Add("@url", _url, SqlDbType.VarChar);

        if (tbDatumOd.Text != string.Empty)
        {
            DateTime _datumOd = Convert.ToDateTime(tbDatumOd.Text);
            _par.Add("@datum_od", _datumOd, SqlDbType.DateTime);
        }

        if (tbDatumDo.Text != string.Empty)
        {
            DateTime _datumDo = Convert.ToDateTime(tbDatumDo.Text);
            _par.Add("@datum_do", _datumDo, SqlDbType.DateTime);
        }

        int _idJezik = Convert.ToInt32(ddlJezik.SelectedValue);
        _par.Add("id_jezik", _idJezik, System.Data.SqlDbType.Int);

        if (ddlMeni.SelectedValue != string.Empty)
        {
            int _idMeniPrej = Convert.ToInt32(ddlMeni.SelectedValue);
            _par.Add("id_meni_prej", _idMeniPrej, System.Data.SqlDbType.Int);
        }
        else
        {
            _par.Add("id_meni_prej", DBNull.Value, System.Data.SqlDbType.Int);
```

```
        }
        if(txtZapSt.Text != string.Empty)
            _par.Add("zap_st", Convert.ToInt32(txtZapSt.Text), System.Data.SqlDbType.Int);
        else
            _par.Add("zap_st", DBNull.Value, System.Data.SqlDbType.Int);

        if (action == "Insert")
        {
            Lib.utils.DbExecute("sys_meni_Insert", _par, this);
        }
        else if (action == "Edit")
        {
            _par.Add("id_meni", Convert.ToInt32(Session["idMeni"]),
System.Data.SqlDbType.Int);
            Lib.utils.DbExecute("sys_meni_Update", _par, this);
        }

        Session.Remove("idDokument");
        Session.Remove("idMeni");
        Session.Remove("tvValuePath");
        Response.Redirect("wfUrednik.aspx");
    }

    protected void btnCancel_Click(object sender, EventArgs e)
    {
        Session.Remove("idDokument");
        Session.Remove("idMeni");
        Session.Remove("tvValuePath");
        Response.Redirect("wfUrednik.aspx");
    }

    protected void calOd_SelectionChanged(object sender, EventArgs e)
    {
        tbDatumOd.Text = calOd.SelectedDate.ToShortDateString();
        HtmlGenericControl masterPageBodyTag =
(HtmlGenericControl)Page.Master.Master.FindControl("bod");

        string _visibility = "none";
        masterPageBodyTag.Attributes.Add("onLoad", "SetDivCalOdVisibility('" + _visibility +
"');");
    }

     protected void calOd_VisibleMonthChanged(object sender, MonthChangedEventArgs e)
     {
        string _visibility = "block";
        HtmlGenericControl masterPageBodyTag =
(HtmlGenericControl)Page.Master.Master.FindControl("bod");
        masterPageBodyTag.Attributes.Add("onLoad", "SetDivCalOdVisibility('" + _visibility +
"');");
    }


    protected void calDo_SelectionChanged(object sender, EventArgs e)
    {
        tbDatumDo.Text = calDo.SelectedDate.ToShortDateString();
        HtmlGenericControl masterPageBodyTag =
(HtmlGenericControl)Page.Master.Master.FindControl("bod");

        string _visibility = "none";
        masterPageBodyTag.Attributes.Add("onLoad", "SetDivCalDoVisibility('" + _visibility +
"');");
    }

    protected void calDo_VisibleMonthChanged(object sender, MonthChangedEventArgs e)
    {
        string _visibility = "block";
        HtmlGenericControl masterPageBodyTag =
(HtmlGenericControl)Page.Master.Master.FindControl("bod");
        masterPageBodyTag.Attributes.Add("onLoad", "SetDivCalDoVisibility('" + _visibility +
"');");

    }
}
```

f) PHP različica izdelave novega menija

```
//Ime datoteke: NovMeni.php

function FillMeni()
{
	$lib = new Sql();
	$conn=mssql_connect($lib->GetServer(),$lib->GetUserName(),$lib->GetPassword());
	if ($conn)
	{
		mssql_select_db($lib->GetDatabase(),$conn)or die( "Unable to select database");
		$sp=mssql_init("sys_meni_SelectWithPermissions",$conn);
		$idUporabnik = $_SESSION['IdUporabnika'];
		mssql_bind($sp,"@id_uporabniki",&$idUporabnik,SQLVARCHAR,FALSE);

		$result=mssql_execute($sp);
		$rows = mssql_num_rows($result);

		echo "<select name='predhodniMeni' size='1'>";

		if($rows>0)
		{
			while ($row = mssql_fetch_array($result, MSSQL_BOTH))
			{
				$idmeni = $row["id_meni"];
				$naslov = $row["naslov"];
				echo"<option value='".$idmeni."'>".$naslov."</option>";
			}
		}
			echo "</select>";
	}
	else
	{
		print("Can't connect to database");
	}
}

function FillJezik()
{
	$lib = new Sql();
	$conn=mssql_connect($lib->GetServer(),$lib->GetUserName(),$lib->GetPassword());
	if ($conn)
	{
		mssql_select_db($lib->GetDatabase(),$conn)or die( "Unable to select database");
		$sp=mssql_init("sys_jezik_Select",$conn);

		$result=mssql_execute($sp);
		$rows = mssql_num_rows($result);

		echo "<select name='jezik' size='1'>";

		if($rows>0)
		{
			while ($row = mssql_fetch_array($result, MSSQL_BOTH))
			{
				$idjezik = $row["id_jezik"];
				$naslov = $row["opis"];
				echo"<option value='".$idjezik."'>".$naslov."</option>";
			}
		}
			echo "</select>";
	}
	else
```

```
	{
		print("Can't connect to database");
	}
}

//Ime datoteke: ShraniMeni.php

<?php
include("Lib.php");
 if(  $_POST["predhodniMeni"]  ||  $_POST["naslov"]||  $_POST["zaporednaSt"]||  $_POST["url"]||
$_POST["datumOd"]|| $_POST["datumDo"] || $_POST["jezik"])
{

	$predhodniMeni = $_POST["predhodniMeni"];
	echo $predhodniMeni;
	$naslov = $_POST["naslov"];
	$zaporednaSt = $_POST["zaporednaSt"];
	$url = $_POST["url"];
	$datumOd = $_POST["datumOd"];
	$datumDo = $_POST["datumDo"];
	$jezik = $_POST["jezik"];

	$lib = new Sql();
	$conn=mssql_connect($lib->GetServer(),$lib->GetUserName(),$lib->GetPassword());
	if ($conn)
	{
		mssql_select_db($lib->GetDatabase(),$conn)or die( "Unable to select database");
		if($naslov =="")
		{
			echo  "<table  align  =  \"center\"  border  =\"0\"><tr>  <td  align  =
\"center\">Naslov je obvezno polje!</td></tr>";
		}
		else
		{
			$sp=mssql_init("sys_meni_Insert",$conn);
			mssql_bind($sp,"@id_meni_prej",&$predhodniMeni,SQLINT4,FALSE);
			mssql_bind($sp,"@naslov",&$naslov,SQLVARCHAR,FALSE);
			mssql_bind($sp,"@zap_st",&$zaporednaSt,SQLINT4,FALSE);
			mssql_bind($sp,"@url",&$url,SQLVARCHAR,FALSE);
			mssql_bind($sp,"@datum_od",&$datumOd,SQLVARCHAR,FALSE);
			mssql_bind($sp,"@datum_do",&$datumDo,SQLVARCHAR,FALSE);
			mssql_bind($sp,"@id_jezik",&$jezik,SQLVARCHAR,FALSE);

			$result=mssql_execute($sp);

			echo                    "<meta                    http-equiv='Refresh'
content='0;url=../UrednikPHP/Drevo.php?meni=0'>";
			mssql_close($conn);
		}
	}
	else
	{
		print("Can't connect to database");
	}
}

?>
```

g) ASP.NET C# različica urejanja dokumenta

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
using System.Text;
using System.Collections.Specialized;
using System.Data;
using Urednik.Lib;

namespace Urednik
{
    public partial class wfDocument : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (Convert.ToString(Session["IdUporabnika"]) == string.Empty)
            {
                Response.Redirect("wfLogIn.aspx", true);
            }
            else if (!IsPostBack)
            {
                string _idDok = Convert.ToString(Session["idDokument"]);
                string _idMeni = Convert.ToString(Session["idMeni"]);
                if (_idDok != string.Empty)
                {
                    DataTable _dtDokument = Lib.utils.DbExecute("sys_dokument_Select",
"id_dokument", _idDok);
                    if(_dtDokument.Rows.Count == 1)
                    {
                        tbNaslovStrani.Text = Convert.ToString(_dtDokument.Rows[0]["naslov"]);
                        cbObjavi.Checked = Convert.ToBoolean(_dtDokument.Rows[0]["objavi"]);
                        string _fckEditorText =
Convert.ToString(_dtDokument.Rows[0]["dokument"]);
                        _fckEditorText = ClearString(_fckEditorText);
                        string _function = "SetDocument('" + _fckEditorText + "');";
                        Page.ClientScript.RegisterStartupScript(this.GetType(), "", _function,
true);
                    }
                }

            }
        }


        protected void btnShrani_Click(object sender, EventArgs e)
        {
            string _fckEditorText = Request.Form["fckEditor"];
            string _editorText = ClearString(_fckEditorText);
            int _idUporabnika = Convert.ToInt32(Session["IdUporabnika"]);
            string _idMeni = string.Empty;
            string _idDok = Convert.ToString(Session["idDokument"]);

            DataTable _dtDokument = Lib.utils.DbExecute("sys_dokument_Select", "id_dokument",
_idDok);
            if (_dtDokument.Rows.Count == 1)
            {
                _idMeni = Convert.ToString(_dtDokument.Rows[0]["id_meni"]);
            }

            Lib.DBParameters _param = new DBParameters();
            _param.Add("dokument", _editorText , SqlDbType.VarChar);
            _param.Add("naslov", tbNaslovStrani.Text, SqlDbType.VarChar);
            _param.Add("objavi", cbObjavi.Checked, SqlDbType.Bit);
            _param.Add("lcUserId", _idUporabnika, SqlDbType.Int);
            if(_idMeni != string.Empty)
                _param.Add("id_meni", _idMeni, SqlDbType.Int);
```

```
            if (_idDok != string.Empty)
            {
                _param.Add("id_dokument", _idDok, SqlDbType.Int);
                Lib.utils.DbExecute("sys_dokument_Update", _param, this);
            }
            else
            {
                _param.Add("id_uporabniki", _idUporabnika, SqlDbType.Int);
                Lib.utils.DbExecute("sys_dokument_Insert", _param, this);
            }

            Session.Remove("idDokument");
            Response.Redirect("wfUrednik.aspx");
        }

        private string ClearString(string str)
        {
            string _str = str;

            ///r/t/n
            int _index = Int32.MinValue;
            do
            {
                _index = FindIndex(_str);
                if (_index != -1)
                {
                    _str = _str.Remove(_index, 1);
                }
            }
            while (_index != -1);

            return _str;
        }

        private int FindIndex(string str)
        {
            int _index = str.IndexOf("\r");
            if (_index == -1)
            {
                _index = str.IndexOf("\n");
            }
            if (_index == -1)
            {
                _index = str.IndexOf("\t");
            }
            return _index;
        }

        protected void btnPreklici_Click(object sender, EventArgs e)
        {
            Session.Remove("idMeni");
            Session.Remove("tvValuePath");
            Response.Redirect("wfUrednik.aspx");
        }

    }
}
```

h) PHP različica urejanja dokumenta

```
//Ime datoteke: UrediDokument.php

<html>
<head>
<META HTTP-EQUIV="Cache-Control" CONTENT="no-cache">
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
<META HTTP-EQUIV="Expires" CONTENT="0">
<?php include("Lib.php"); ?>
       <title>Uredi dokument</title>
       <script type="text/javascript" src="../UrednikPHP/ckeditor/ckeditor.js"></script>
```

```
</head>
<?php

if($_SESSION['IdUporabnika'] == "")
{
echo "Za dostop do podatkov je potrebna registracija.";
echo "<meta http-equiv='Refresh'  content='0;url=../UrednikPHP/Login.php'>";
}
$idDok = $_GET["dok"];
if($idDok != "")
{
	$lib = new Sql();
	$conn=mssql_connect($lib->GetServer(),$lib->GetUserName(),$lib->GetPassword());
	$naslov="";
	$url ="";
	$datumOd="";
	$datumDo="";
	$jezik="";
	$objavi="";
	$dokument="";
	if ($conn)
	{
		mssql_select_db($lib->GetDatabase(),$conn)or die( "Unable to select database");
		$sp=mssql_init("sys_dokument_Select",$conn);

		mssql_bind($sp,"@id_dokument",&$idDok,SQLINT4,FALSE);

		$result=mssql_execute($sp);
		mssql_close($conn);
		while ($row = mssql_fetch_array($result, MSSQL_BOTH))
		{
			$naslov = $row["naslov"];
			$url = $row["url"];
			$datumOd = $row["datum_od"];
			$datumDo = $row["datum_do"];
			$jezik = $row["jezik"];
			$objavi = $row["objavi"];
			$dokument =$row["dokument"];
		}
	}
	else
	{
		print("Can't connect to database");
	}
?>
<body>
<table style="width: 950px; " align="center" border="0">
<form action="ShraniDokument.php" method="post">
	<tr>
		<td width="100px">
			Naslov strani:
		</td>
		<td>
			<?php  echo"<input name=\"Naslov\" type=\"text\" style=\"width: 600px;\"
value=\"".$naslov."\"/>"; ?>
		</td>
	</tr>




	<tr>
		<td width="100px">
			Objavi
```

```
		</td>
		<td style="align: left;" align ="left">
			<?php
				if($objavi == 1)
				{
					echo"<input	name=\"Objavi\"	type=\"checkbox\"
style=\"width: 600px;\" checked/>";
				}
				else
				{
					echo"<input	name=\"Objavi\"	type=\"checkbox\"
style=\"width: 600px;\" />";
				}
				echo	"<input	type=\"hidden\"	id=\"idDok\"	name=\"idDok\"
value=\"".$idDok."\"/>"
			?>
		</td>
	</tr>

	<tr>
		<td width="100px">
			<a href="DodajSliko.php">Dodaj sliko</a>
		</td>
		<td>
		</td>
	</tr>
	<tr >
	<td colspan="2">

	<div id="alerts">
		<noscript>
			<p>
				<strong>CKEditor	requires	JavaScript	to	run</strong>.	In	a
browser with no JavaScript
				support,	like	yours,	you	should	still	see	the	contents	(HTML
data) and you should
				be able to edit it normally, without a rich editor interface.
			</p>
		</noscript>
	</div>
			<p>
				<?php		echo"<textarea	cols=\"120\"	id=\"fckEditor\"
name=\"fckEditor\" rows=\"50\">".$dokument."</textarea>"; ?>
				<script type="text/javascript">
					CKEDITOR.replace( 'fckEditor',
						{
							skin : 'v2'
						});
				</script>
			</p>
			<br>
			<input type="submit" value="Shrani"/>  
			<button
type="button"
ONCLICK="window.location.href='../UrednikPHP/Drevo.php?meni=0'">Prekliči</button>
		</form>

	</td>
	</tr>
</table>
<?php
}
else
{
	echo "<div align=\"center\">Izberi dokument!</div>";
```

```
        echo "<meta http-equiv='Refresh'  content='2;url=../UrednikPHP/Drevo.php?meni=0'>";
}
?>
</body>
</html>
```

```
//Ime datoteke: ShraniDokument.php

<?php session_start();
include("Lib.php");
?>

<html>
 <?php
        $editorTxt = $_POST["fckEditor"];
        $naslov = $_POST["Naslov"];
        $url = $_POST["Url"];
        $datumOd = $_POST["DatumOd"];
        $datumDo = $_POST["DatumDo"];
        //$jezik = $_POST["Jezik"];
        $objavi = $_POST["Objavi"];
        $idDok = $_POST["idDok"];
        $idUporabnik = $_SESSION['IdUporabnika'];

        echo $naslov;
        $lib = new Sql();
        $conn=mssql_connect($lib->GetServer(),$lib->GetUserName(),$lib->GetPassword());
        if ($conn)
        {
                mssql_select_db($lib->GetDatabase(),$conn)or die( "Unable to select database");
                if($idDok == "")
                {
                        $sp=mssql_init("sys_dokument_Insert",$conn);
                }
                else
                {
                        $sp=mssql_init("sys_dokument_Update",$conn);
                        mssql_bind($sp,"@id_dokument",&$idDok,SQLINT4,FALSE);
                }
                mssql_bind($sp,"@dokument",&$editorTxt,SQLVARCHAR,FALSE);
                mssql_bind($sp,"@naslov",&$naslov,SQLVARCHAR,FALSE);
                mssql_bind($sp,"@objavi",&$objavi,SQLBIT,FALSE);
                mssql_bind($sp,"@lcUserId", &$idUporabnik,  SQLINT4, FALSE);

                $result=mssql_execute($sp);
                mssql_close($conn);

                echo                            "<meta                          http-equiv='Refresh'
content='0;url=../UrednikPHP/Drevo.php?meni=0'>";
        }
        else
        {
                print("Can't connect to database");
        }
?>
        <br />

</html>
```

VIRI

[1] Internet Information Services
http://en.wikipedia.org/wiki/Internet_Information_Services

[2] Simon Klemen: Naučite se PHP v 24 urah. Tretja izdaja. Ljubljana: Pasadena, 2004. ISBN 961-6361-62-7

[3] Agilne metodologije razvoja informacijskih sistemov
http://bajecm.fri.uni-lj.si/CRP2001/Clanki/Agilne%20Metodologije%20Razvoja%20IS.pdf

[4] (2007) Marko Bajec: Prosojnice s predavanj 2006/2007 predmeta Razvoj informacijskih sistemov

[5] Microsoft ASP.NET
http://www.asp.net/(S(ywiyuluxr3qb2dfva1z5lgeg))/learn/videos/

[6] Bill Evjen, Scott Hanselman, Farhan Muhammad, Srinivasa Sivakumar, Devin Rader: Professional ASP.NET 2.0. Indianapolis: Wiley Publishing, Inc., 2006. ISBN-13: 978-0-7645-7610-2

[7] Migrating from PHP do ASP.NET
http://msdn.microsoft.com/en-us/library/aa479002.aspx

[8] ASP.NET
http://en.wikipedia.org/wiki/ASP.NET

[9] PHP
http://en.wikipedia.org/wiki/PHP

[10] Žiga Jeločnik: Sistem za spremljanje in analizo medijev, diplomsko delo, FRI, 2009

[11] SQL Server 2008 Express
http://www.microsoft.com/Sqlserver/2008/en/us/express.aspx

[12] CKEditor
http://ckeditor.com/what-is-ckeditor

[13] Photoimpact
http://en.wikipedia.org/wiki/Ulead_PhotoImpact