Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tim Rijavec

Razvoj ogrodja in sistema za upravljanje spletnih vsebin v Pythonu

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: doc. dr. Mira Trebar

Ljubljana 2014

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Z razvojem tehnologij se pojavljajo sistemi, ki vse večjemu številu uporabnikov omogočajo samostojno vstavljanje podatkov, urejanje in upravljanje spletnih vsebin. Najpogosteje se v ta namen uporabljajo različne aplikacije, ki jih poznamo pod skupnim imenom Content Manegement System (CMS). Kandidat naj v diplomski nalogi pregleda in analizira najbolj razširjene in popularne. Na osnovi ugotovljenih prednosti in slabosti naj razvije samostojno rešitev s primernim ogrodjem in aplikacijo za upravljanje spletnih vsebin z uporabo programskega jezika Python in drugih tehnologij. Delovanje in uporabnost sistema naj predstavi z izdelavo spletnega portala FRI.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Tim Rijavec, z vpisno številko **63050211**, sem avtor diplomskega dela z naslovom:

Razvoj ogrodja in sistema za upravljanje spletnih vsebin v Pythonu

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Mire Trebar,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 11. maja 2014 Podpis avtorja:

Zahvaljujem se mentorici doc. dr. Miri Trebar za dobro voljo, podporo in potrpežljivost.

Največja zahvala gre Tjaši, ki me je v zadnjih letih spodbujala in mi dajala moči za dokončanje študija.

Na koncu se zahvaljujem staršem, prijateljem in pravzaprav čisto vsem, ki so kakorkoli pripomogli k uspešnemu dokončanju moje študijske poti.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
CMS	Content Management System	sistem za urejanje vsebin
SEO	Search Engine Optimization	optimizacija spletnih strani za iskalnike
URL	Uniform Resource Locator	spletni naslov
HTTP	Hypertext Transfer Protocol	protokol za prenos hiperteksta
MVC	Model View Controller	model, pogled in upravljalec
MVP	Model View Presenter	model, pogled in predstavnik
MVVM	Model View ViewModel	model, pogled, model pogleda
API	Application Programming Interfaces	programski vmesnik
IDE	Integrated Development Environment	integrirano razvojno okolje
DOM	Document Object Model	objektni model dokumenta
SOAP	Simple Object Access Protocol	protokol za izmenjavo informacij
CSS	Cascading Style Sheets	kaskadna slogovna predloga
SQL	Structured Query Language	strukturirani povpraševalni jezik
IP	Internet Protocol	spletni protokol
IDE	Integrated development environment	interaktivno razvojno okolje
DRY	Don't repeat yourself	ne ponavljaj se

Povzetek

Diplomsko delo obravnava načrtovanje in razvoj programskega ogrodja in sistema za upravljanje spletnih vsebin. Analizirali smo opisane pomanjkljivosti obstoječih rešitev in se na osnovi izvedenih testov odločili, da bomo za razvoj našega sistema uporabili ogrodje Django. Pri izdelavi našega ogrodja smo uporabili nekaj obstoječih funkcionalnosti, zasnovali in vključili dodatne module tako, da ga lahko z dodajanjem novih vtičnikov poljubno razširimo in enostavno prilagodimo uporabnikovim zahtevam. Izdelano ogrodje smo nadgradili s sistemom za upravljanje vsebin, ki ga poznamo pod imenom CMS (Content Management System). Ogrodje in CMS sta izdelana v programskem jeziku Python in spletnih tehnologijah, kot so HTML, CSS, JavaScript. Aplikacija predstavlja okolje za urejanje podatkov in omogoča oblikovanje izgleda spletnega mesta. Uporabnost in funkcionalnosti so bile preverjene in predstavljene v praktični rešitvi za urejanje spletnih vsebin Fakultete za računalništvo in informatiko.

Ključne besede: splet, ogrodje, upravljanje vsebin, Python.

Abstract

This Thesis addresses the planning and development of web framework and content management system. An analysis of the existing solutions described deficiencies was made and we came to a conclusion, based on these tests, that we will be using Django as the base framework for developing our system. To create our framework, we used some of the existing functionalities and we developed and included additional modules that allow us to extend and easily adapt to users requirements by simply adding new plugins. The framework was upgraded with a system for content managing, also known as CMS (Content Management System). Both the framework and the CMS are written in Python and web technologies such as HTML CSS, Javascript. The application serves as a data editing environment and enables us to create and edit the appearance of a website. Usability and functionality were tested and presented with a practical solution for editing web content of Faculty of Computer and Information Science website.

Keywords: www, framework, content management, Python.

Poglavje 1

Uvod

V zadnjih nekaj letih se je potreba po hitro in dobro narejenih spletnih straneh zelo povečala. Zaradi tega so se številna podjetja in posamezniki usmerili v razvoj različnih sistemov, ki bi kar v največji meri zadovoljili potrebe uporabnikov. Posledica tega je številna izbira različnih sistemov. Nekateri so neprimerni zaradi izbranih tehnologij, drugi prezahtevni za uporabo, pri nekaterih manjkajo funkcionalnosti, ki jih uporabniki potrebujejo. Izbira primernega sistema je težka, orodje, ki bi ugodilo popolnoma vsem zahtevam uporabnikov in bilo enostavno za uporabo, pa neobstoječa dobrina.

Ogrodje (ang. Framework) in sistem za upravljanje spletnih vsebin (CMS - content management system), ki sta nastala v okviru diplomskega dela, sta dober približek orodju, ki bi združeval prav to; enostavno in hitro izdelavo ter urejanje spletne aplikacije. Z njim želimo poenostaviti in pospešiti pisanje spletnih aplikacij ter prihraniti čas in denar pri njihovem razvoju.

Na začetku diplomskega dela je predstavljena ideja ogrodja za razvoj spletnih aplikacij in sistema za urejanje spletnih vsebin CMS. Sledi kratek opis uporabljenih tehnologij in orodij. Jedro dela predstavlja opis izdelave ogrodja in CMS-ja. Opisana je struktura sistema, ki smo ga razvili in analiza obstoječih sistemov. Na koncu je predstavljena praktična rešitev za urejanje spletnih vsebin Fakultete za računalništvo in informatiko, ki smo jo izdelali s pomočjo ogrodja in CMS-ja.

Poglavje 2

Spletne vsebine

V zadnjem desetletju je ogromno podjetij začelo izkoriščati splet kot kanal za komunikacijo in izmenjavo informacij. Med takšne informacije se štejejo tudi oglasi, s pomočjo katerih lahko podjetja tudi posredno služijo. Če te aplikacije podrobneje raziščemo, ugotovimo, da nekatere izmed njih zbirajo podatke o uporabnikih. Med temi podatki so največkrat zgodovina brskanja, zgodovina iskanja, IP (Internet Protocol) naslov uporabnika, elektronski naslov, vprašanja in odgovori, oddani preko podpornih obrazcev, seznam izdelkov v nakupovalnih košaricah, ipd.

Zato je splet odličen prodajni kanal za posameznike ter velika in majhna podjetja. Podjetje comScore zbira podatke o tem, kaj ljudje počnejo v digitalnem svetu. Po njihovih meritvah, narejenih na osnovi statistike prodaje preko spleta, so na ameriškem trgu uporabniki v tretjem četrtletju leta 2013 naredili za več kot 47 milijard dolarjev nakupov [5]. Številka je precej visoka, podobno pa je pri trgovskem velikanu Wal-Mart, ki na dan proda izdelke v vrednosti 36 milijonov dolarjev [6].

Splet je okolje, ki uporabniku omogoča uporabo številnih brezplačnih spletnih aplikacij. V ta namen so mu na voljo spletni brskalniki, ki omogočajo interakcijo s spletnimi aplikacijami.

Spletna aplikacija je kateri koli program, ki ga s pomočjo brskalnika uporabljamo preko svetovnega spleta ali pa z njim komunicirajo druge aplika-

cije. Mednje sodijo socialna omrežja (Facebook, MySpace, hi5.com), galerije slik (Flickr, Shutterfly, Picasaweb), spletni koledarji (Google koledar, Yahoo! koledar, Live koledar), virtualne klepetalnice (Meebo, omgpop), spletne trgovine (Amazon, eBay, enaA, bolha.com), spletni imeniki (Yellow.com, TIS, AnyWho), blogi (Blogger, Tumblr, WordPress), spletne novice (Delo.si, Slovenskenovice.si, 24ur.com, The Guardian), spletni slovarji (Dictionary.com, islovar.org), spletne pošte (Gmail, HotMail, Yahoo Mail), zemljevidi (Google Maps, Bing maps, Najdi.si zemljevid), iskalniki (Google, Bing, Yahoo, Najdi.si), pisarniška orodja (Google Docs, Zoho), ipd. Dandanes so vse prej kot skupek statičnih vsebin, torej besedil in slik. Uporabniku omogočajo personalizacijo storitev, dinamično vsebino, interakcijo z drugimi uporabniki preko spletne pošte, forumov, socialnih omrežij, idr. Ob vsaki zahtevi pošljejo zahtevo na strežnik, ki iz podatkovne baze s shranjenimi podatki uporabniku prikaže želeno informacijo.

Spletna stran je dokument z nadbesedilom, s katerim delimo z uporabniki različne podatke. Lahko jo namenimo osebni uporabi, predstavitvi podjetja ali izdelka ipd. Povprečen uporabnik se hitro znajde pri uporabi spletnih strani brez kakršnih koli predhodnih informacij o tem, kako delujejo.

Pri izdelavi spletnih strani uporabljamo različne načine. Lahko jih izdelamo od začetka do konca s pomočjo programskih jezikov, kot so: Python, PHP, C#, si pomagamo z ogrodji [3], ki nam omogočajo hitrejšo in lažjo izdelavo, ali pa uporabimo predpripravljene spletne strani in z njihovim urejevalnikom uredimo vsebino ter obliko strani.

Nekatera ogrodja so zahtevnejša za uporabo, spet druga ne ustrezajo našim potrebam. Pri izboru moramo paziti, da izpolnijo čim več zahtev, ki so pomembne za našo spletno stran. V nasprotnem primeru se lahko čas izdelave občutno podaljša. Pogoste pomankljivosti ogrodij so: jedro ogrodja ne prenese obremenitve, sredi razvoja ugotovimo nepovezljivost ogrodja z drugimi sistemi, nekompatibilnost strežniške tehnologije s končno aplikacijo, ipd. Pri izbiri CMS-ja ni nič drugače. Paziti moramo, da lahko z njim urejamo želeno vsebino.

2.1 Ogrodje za razvoj spletnih vsebin

Ogrodje je univerzalna programska oprema, zaključena arhitektura, ki uporabniku ponuja skupek orodij za lažji razvoj programov. Ta orodja so lahko algoritmi za kompresijo, rezanje slik, avtomatsko generiranje obrazcev za urejanje podatkov shranjenih v podatkovni bazi, generiranje spletnih naslovov za optimizacijo spletnih strani (SEO - search engine optimization), ipd. Vsebuje vmesnike (APIs - application programming interfaces), ki razvijalcu ponujajo dostop do funkcionalnosti ogrodja in s tem omogočijo razvoj aplikacije.

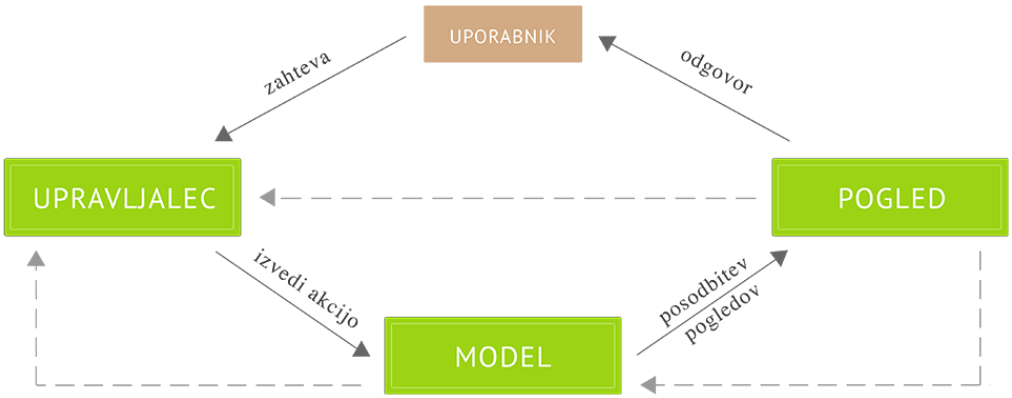
Ideja ogrodja je, da razvijalec ne narekuje toka strani, temveč ga narekuje ogrodje samo. Poleg osnovnih funkcij, lahko ogrodje razvijalcu ponuja možnost razširitve in s tem dodajanja novih, bolj specifičnih operacij za aplikacijo.

V začetni fazi spletnega razvoja so se uporabljale statične HTML (hypertext markup language) datoteke, med saboj povezane s hiperpovezavami (ang. hypertext). Te datoteke so bile naložene na spletnih strežnikih. Ob vsaki spremembi podatkov smo morali te datoteke posodobiti in jih ponovno naložiti na strežnike. Zaradi hitre rasti interneta so se začeli razvijati novi programski jeziki (PHP, ASP.NET, D [9], Go [10]), nekateri pa so se samo prilagodili (Java, Python). Najbolj priljubljen je postal programski jezik Java, zaradi zgodnje integracije z brskalnikom Netscape Navigator.

Ogrodja uporabljajo arhitekturne vzorce. To so splošno priznane in uporabljene rešitve za reševanje programskih arhitekturnih problemov. Med najbolj poznanimi vzorci so MVC (model, pogled, upravljalec), MVP (model, pogled, predstavnik), MVVM (model, pogled, model pogleda).

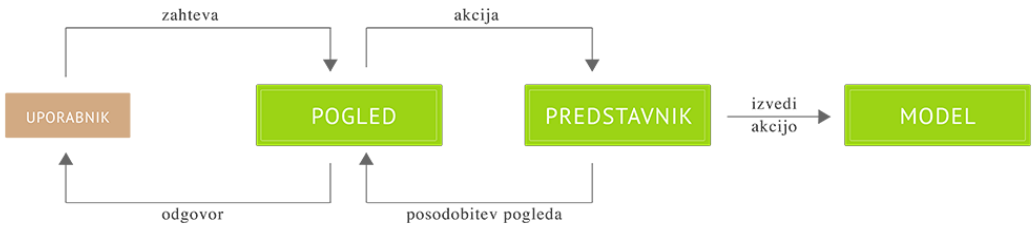
Vzorec MVC se uporablja pri razvoju uporabniških vmesnikov. Razdeljen je na tri dele, ki so med seboj povezani. Na eni strani imamo interni predstavitveni del; model, ki skrbi za pravilno predstavitev podatkov. Na drugi strani je pogled, ki uporabniku podatke pravilno prikaže. To so lahko kateri koli podatki, grafi, seznami, besedila, idr. Upravljalec vsebuje pravila in povezovalno logiko, ki podatke pridobljene od modela, ob vsaki zahtevi

posreduje pogledu. Ta jih nato pravilno upodobi. Strukturo vzorca MVC prikazuje slika 2.1.



Slika 2.1: MVC vzorec.

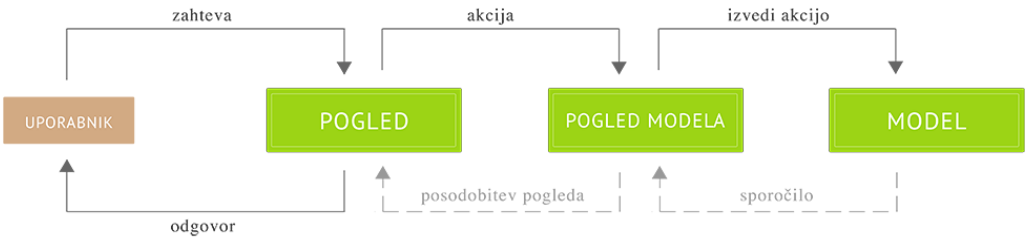
Vzorec MVP je predelava MVC in se prav tako uporablja pri razvoju uporabniških vmesnikov. Razlika med njima je v osrednjem delu, kjer predstavnik, ki vsebuje vso predstavitveno logiko modela, zamenja vlogo upravljalca. Sprejem zahtev pa prevzame pogled. Model predstavlja samo domenski model (entitete, atribute in povezave). Struktura vzorca MVP je prikazana na sliki 2.2.



Slika 2.2: MVP vzorec.

MVVM vzorec je nadgradnja predstavitvenega modela PM s strani podjetja Microsoft [12], ki ga je predstavil Martin Fowler [11]. Prednost tega

vzorca je, da strogo loči poslovno in predstavitveno logiko aplikacije od uporabniškega vmesnika. Struktura vzorca MVVM je prikazana na sliki 2.3.



Slika 2.3: MVVM vzorec.

Pri razvoju našega ogrodja smo uporabili vzorec MVC predvsem zato, ker le tega uporablja tudi ogrodje Django, ki je spisano v programskem jeziku Python [4].

2.2 Sistem za urejanje spletnih vsebin

Pogosto je urejanje spletnih strani zahteven in časovno potraten proces. Kljub temu, da obstaja veliko število CMS sistemov, katerih glavni namen je, da nam poenostavijo urejanje naših strani, je težko izbrati primernega. Pri izbiri moramo upoštevati njihovo združljivost z našimi željami in zmožnost izpolnjevanja naših pričakovanj oziroma potreb pri urejanju spletne strani. Sistem za upravljanje vsebin je sistematičen proces ustvarjanja, urejanja in predstavitve vsebine.

Tak sistem je v prvi vrsti spletna aplikacija, ki nam ponuja orodja za takojšnje urejanje spletnih strani in med tem beleži dane spremembe. Omogoča nam shranjevanje slikovnih, glasbenih, video ter drugih datotek in podatkov, ki jih shrani v eno izmed podatkovnih baz: MS SQL, MySQL, SQLite, PostgreSQL, ipd. [13].

Idealen CMS bi uporabniku omogočal urejanje vseh podatkov, urejanje izgleda strani in analizo zbranih podatkov, vendar takšnega sistema zaradi

zahtevnosti izdelave in prilagoditve raznolikosti uporabnikov in želja ni. Obstajajo samo dobri približki. Nekateri sistemi so namenjeni točno določenim uporabam: 'blogger' je sistem za objavljanje prispevkov, 'google analytics' je sistem za pregled zbranih podatkov o obiskovalcih strani, 'flickr' in 'picasa' sta sistema za urejanje in objavo galerije slik.

Glavni problemi obstoječih sistemov so zapletenost uporabe, težave pri združevanju z obstoječo programsko opremo, cenovna neugodnost, pomanjkanje tehnične dokumentacije, varnost, ipd. V praksi bi si več kot polovica uporabnikov želela enostavnejše sisteme in možnosti personalizacije uporabniškega vmesnika.

Pri razvoju CMS-ja moramo zgoraj naštete probleme prepoznati in jih poizkusiti odpraviti. S tem lahko pohitrimo izdelavo in urejanje spletnih strani ter tako izboljšamo uporabniško izkušnjo. Nekatere lastnosti, ki jih lahko CMS vsebuje so: urejanje podatkov, iskalnik podatkov, možnost komentiranja, sistem za ocenjevanje, urejanje uporabniškega vmesnika, nadzor nad meniji, uporabo podatkovne baze za shranjevanje podatkov, registracija in prijava uporabnikov, urejanje dovoljenj za ogled vsebine, sistem za predpomnjenje podatkov, SEO povezave, modularnost, izmenjava informacij med uporabniki, večjezičnost, različne statistike podatkov, pregled nad dogodki, forum, blog, vremenska postaja, dinamično generiranje strani, idr.

Med pomembnejše lastnosti, ki jih mora vsebovati dober CMS štejemo uvoz in izvoz podatkov, identifikacijo uporabnikov, nastavljanje vlog in pravic uporabnikov, zgodovino sprememb, modularnost, ipd.

Na trgu obstajata dve glavni različici takih sistemov. Odprtokodni sistemi so brezplačni, se konstantno posodabljajo, preizkušeni s strani spletne skupnosti in za njih obstaja veliko oblikovnih predlog. Na drugi strani imamo plačljive sisteme, ki so namenjeni in prilagojeni strankam ter pogosto vsebujejo veliko napak zaradi nepopolnih testov.

Poglavje 3

Orodja in tehnologije

Orodja za razvoj spletnih aplikacij razvijalcem omogočajo lažje in hitrejše pisanje, testiranje, razhroščevanje in analiziranje kode. Nekatera so združena v interaktivno razvojno okolje IDE (Integrated development environment): Adobe Flash Builder, Eclipse, MonoDevelop, Code::Blocks, Microsoft Visual Studio, LispWorks, NetBeanse, Aptana Studio, Delphi, Komodo IDE, Zend Studio, Xcode, PyCharm, idr. Druga so samostojna in jih uporabljamo po potrebi: Firebug, YSlow, HttpWatch, ColorZilla, idr. Namenjena so tako razvijalcem kot tudi drugim uporabnikom, ki jih uporabljajo za namen razvoja, testiranja, ipd. Zaradi večjega interesa razvijalcev, so nekatera od teh orodij že vključena v brskalnike same.

Za razvoj potrebujemo tudi spletne tehnologije, ki povezujejo označevalni jezik HTML (HyperText Markup Language), spletne protokole, HTTP (Hypertext Transfer Protocol), SOAP (Simple Object Access Protocol), idr. in programske jezike (PHP, Pythom, C#, Ruby, idr.).

3.1 Orodja

Adobe Flash Builder je razvojno okolje za izdelavo iger in aplikacij z uporabo jezika ActionScript [15]. Vsebuje orodja za testiranje, spremljanje prometa, testne enote, ipd.

Eclipse IDE je program, sestavljen iz osrednjega delovnega prostora in veliko vtičnikov [16]. Večinoma je napisan v jeziku Java, uporabljamo pa ga lahko za razvoj aplikacij v jezikih Java, JavaScript, PHP, Python, Perl, R, Ruby, Natural, Scala, Groovy, idr.

MonoDevelop je program, neodvisen od operacijskega sistema, ki je primarno namenjen razvoju aplikacij v jeziku C# [17]. Razvijalcem omogoča hitro izdelavo namiznih in ASP.NET spletnih aplikacij.

Code::Blocks je program za izdelavo aplikacij v jezikih C, C++ in Fortran, ki zadovoljuje najzahtevnejše potrebe uporabnikov [18]. Zaradi modularne zasnove lahko po lastni potrebi dodamo vtičnike, ki jih potrebujemo.

Microsoft Visual Studio vsebuje veliko število orodij in storitev, ki nam pomagajo ustvariti aplikacije tako za Microsoftove, kot tudi za druge sisteme [19].

LispWorks je idealen za izdelavo zapletenih LISP programov [20].

NetBeans je odprtokoden IDE, idealen za hiter in enostaven razvoj namiznih, mobilnih in spletnih aplikacij v programskih jezikih Java, PHP, C, C++, idr [21].

Aptana Studio je orodje, ki izkorišča prilagodljivost programa Eclipse in je usmerjeno v izdelavo spletnih aplikacij [22].

Delphi se uporablja za razvoj Windows, Mac, Android in iOS aplikacij [23].

Komodo IDE je profesionalno orodje za razvoj aplikacij v praktično katerem koli programskem jeziku (Python, PHP, Ruby, Perl, HTML, Javascript, idr) [24].

Zend Studio ponuja orodje za razvoj PHP aplikacij, reševanje problemov in timsko delo [25].

Xcode vsebuje urejevalnik, instrumente in simulator za izdelavo OSX in iOS aplikacij [27].

PyCharm je napredno orodje za profesionalno izdelavo aplikacij v jeziku Python [26] Obstajata brezplačna in plačljiva različica. Podpira obstoječa ogrodja kot so Django, Google App Engine, idr. Z lahkoto se ga prilagodi

tudi drugim. Uporabili smo ga pri razvoju našega ogrodja in CMS-ja. Orodje je razvijalcu prijazno, saj podpira veliko programskih jezikov in tehnologij kot so Python, JavaScript, HTML, idr. Med drugimi lahko v njem opišemo jezik, ki ga v osnovi ne podpira. Za pojmom 'podpira' se skrivajo načini predstavitve programske kode v urejevalniku, samodejno zaključevanje kode, ipd. Kot večina visoko zmogljivih orodij tudi PyCharm vsebuje razhroščevalnik, s katerim si pomagamo pri samem razvoju.

3.2 Tehnologije

Pri izdelavi ogrodja smo uporabili programski jezik Python, za izdelavo CMS-ja pa smo uporabili dinamični JavaScript, označevalni jezik HTML in jezik za opisovanje oblike CSS (cascading style sheets).

3.2.1 Python

Python uvrščamo med enostavnejše programske jezike [7]. Primeren je za začetnike, kot tudi za profesionalne programerje. Zanj lahko rečemo, da je visokonivojski skriptni in objektno orientiran programski jezik. Ima značilnosti skriptnih jezikov (uporabnik sam izvede kodo, hitro se ga naučimo, izvorna koda se prevede v binarno predstavitev izvršnega progama) kot tudi značilnosti tradicionalnih programskih jezikov.

Zaradi velikega števila že napisanih modulov, je lahko programiranje v Pythonu zelo hitro. Obsežne programe, napisane v drugih jezikih, lahko tako v Pythonu napišemo v nekaj vrsticah. Obstoječi moduli so napisani v Pythonu, ali pa v drugih jezikih: C, C++, idr. Uporaba obojih je popolnoma enaka, v naš program jih vključimo z ukazom 'import'. Ker je Python objektno orientiran jezik, so funkcije, moduli in spremenljivke objekti. Pomankljivost Pythona je hitrost, ki se ne more primerjati s hitrejšimi: Javo, C, ipd. Kljub temu pa ga odlikuje dobro povezovanje z jezikom C, v katerem lahko napišemo zahtevnejše operacije in s tem občutno pohitrimo program. V Pythonu so napisane mnoge večje spletne aplikacije, med njimi tudi Yo-

uTube, ki poleg tega uporablja še C in JavaScript kot glavna programska jezika.

3.2.2 JavaScript

JavaScript je skriptni, objektno orientiran, dinamičen programski jezik [28]. Najbolj razširjen je v brskalnikih. Zaradi njegove popularnosti pa je čedalje bolj uporabljen tudi na strežniških sistemih v aplikacijah, kot sta node.js in Apache CouchDB. JavaScript uporabljamo v brskalnikih za upravljanje vsebine spletne strani in asinhrone komunikacije s strežnikom. Čeprav njegovo ime močno spominja na programski jezik Java, sta si semantično različna.

3.2.3 HTML in CSS

HTML je označevalni jezik, ki z zaporedjem elementov, sestavljenih iz začetne in končne značke, ter vsebine, ki se nahaja med njima, brskalnikom omogoča prikazovanje spletne strani [29]. Začetno značko predstavimo z <IME_ZNAČKE>, končno značko pa z </IME_ZNAČKE>. Imena značk so lahko napisana z malimi ali velikimi črkami. Za pisanje HTML dokumentov lahko uporabimo kateri koli urejevalnik besedila, npr. beležnico.

Deklaracija tipa dokumenta DOCTYPE (Document Type Declaration) je navodilo, ki povezuje dokument z definicijo tipa dokumenta DTD [30].

Strukturo dokumenta si lahko predstavljamo kot drevo, ki ima lahko poljubno število elementov. Pravila, ki definirajo vsebino elementov, predstavijo dokument kot zaključeno celoto.

Z uporabo slogovnih predlog lahko HTML elementom nastavimo oblikovne lastnosti, kot so: barva, velikost, pozicija, odmike ipd. V HTML dokumente lahko CSS vstavimo kot povezavo na zunanjo datoteko, zapišemo določila v 'style' značko ali v 'style' atribut elementa.

3.2.4 SQLite

SQLite je podatkovna baza, ki ne potrebuje strežnika in konfiguracijskih datotek [8]. Za razliko od večine drugih SQL podatkovnih baz, SQLite nima ločenih procesov (to je program, ki izvaja naloge), temveč za branje in pisanje uporablja kar datoteko. Celotna podatkovna baza, skupaj s tabelami, indeksi, prožilci in pogledi je tako shranjena v eni sami datoteki. To jo naredi izredno enostavno za prenašanje med aplikacijami in strežniki. SQLite koda je prosto dostopna in se jo lahko uporablja v kakršne koli namene. Uporabili smo jo torej tudi za namen predstavitve našega sistema.

Poglavje 4

Razvoj sistema

Kvantiteta različnih sistemov za urejanje in izdelavo spletnih aplikacij se iz dneva v dan povečuje, to pa še ne pomeni, da se povečuje tudi kvaliteta. Med najbolj popularnimi ogrodji so ASP.NET, Ruby on Rails, Django, CodeIgniter, Symfony in Spring. Najprej smo pripravili tabelo funkcionalnosti 4.1, ki jo posamezno ogrodje ponuja. Nato smo se lotili testiranja odzivnosti ogrodij. Testirali smo hitrost branja in pisanja v podatkovno bazo, rezultate pa predstavili v tabeli 4.2.

Ime	ASP.NET	Ruby-Rails	Django	CodeIgniter	Symfony	Spring
Jezik	ASP.NET	Ruby	Python	PHP	PHP	Java
i18	Da	Da	Da	Da	Da	Da
MVC	Da	Da	Da	Da	Da	Da
ORM	Da	Da	Da	Ne	Da	Da
Ajax	Da	Da	Da	Da	Da	Da
Cache	Da	Da	Da	Da	Da	Da
Obrazci	Da	Da	Da	Da	Da	Da
Varnost	Da	Da	Da	Da	Da	Da
Predloge	Da	Da	Da	Da	Da	Da

Tabela 4.1: Primerjava funkcionalnosti.

Teste smo naredili z orodjem 'ab' (Apache HTTP server benchmarking

tool). To je orodje za merjenje odzivnosti spletnih strani v številu odgovorov na sekundo. Vsak test je razdeljen na dva dela: a in b. Pri delu a predstavlja boljši rezultat večja številka, pri delu b pa manjša številka. Številka je povprečje stotih testiranj.

1. Test 1 – Testirali smo število dobljenih odgovorov na sekundo v formatu JSON.
2. Test 2 – Testirali smo hitrost branja iz podatkovne baze z eno poizvedbo.
3. Test 3 – Testirali smo hitrost branja iz podatkovne baze z desetimi hkratnimi poizvedami.
4. Test 4 – Testirali smo hitrost pisanja v podatkovno bazo.

Test št.	ASP.NET	Ruby-Rails	Django	CodeIgniter	Symfony	Spring
Test 1a	28,000	4,600	9,200	4,300	400	35,500
Test 1b	9 ms	2 ms	29 ms	61 ms	300 ms	7 ms
Test 2a	11,000	2,900	4,100	3,900	370	20,500
Test 2b	20 ms	8 ms	60 ms	70 ms	470 ms	17 ms
Test 3a	3,200	60	510	2,000	160	2,400
Test 3b	80 ms	40 ms	500 ms	130 ms	50 ms	100 ms
Test 4a	1,800	5	240	10	5	240
Test 4b	140 ms	90 ms	1000 ms	770 ms	1800 ms	3300 ms

Tabela 4.2: Testiranje ogrodij.

Če primerjamo ogrodja glede na funkcionalnost, med njimi praktično ni razlike. Po testiranju pa rezultati seveda prikažejo drugačno situacijo. Ogrodji Spring in ASP.NET izstopata iz povprečja, toda izdelava programov v jezikih Java in ASP.NET, ki jih uporabljata ogrodji, je zamudnejša. Med ostalimi se je najslabše odrezalo ogrodje Symfony, pri katerem smo, glede na ostala ogrodja, dobili podpovprečne rezultate pri testiranju hitrosti. Za

razvoj našega ogrodja smo kot osnovo izbrali Django, ki je bil boljši med preostalimi. Razvijalci ga predstavljajo kot elegantno orodje za perfekcioniste, ki želijo izdelati kvalitetne aplikacije. Ker je naša želja razviti sistem, ki deluje kot celota, s katero lahko razvijamo spletne aplikacije in hkrati urejamo vsebino, smo se odločili za razvoj lastnega ogrodja in sistema za urejanje vsebine. Z Razvoj smo razdelili na: pripravo načrta ogrodja, izdelavo ogrodja, analizo že obstoječih sistemov CMS, prikazano v tabeli 4.3, pripravo načrta in izdelavo CMS. Pri analizi obstoječih sistemov smo opazili, da veliko dodatnih funkcionalnosti dobim z namestitvijo vtičnikov. Zaradi tega smo se odločili za izdelavo ogrodja in CMS-ja, ki je v celoti sestavljen iz vtičnikov in na ta način popolnoma prilagodnljiv.

Ime	Joomla!	Drupal	WordPress	TYPO3	Plone
Jezik	PHP	PHP	PHP	PHP	Python
Preverjanje uporabnika	Da	Da	Da	Da	Da
Zgodovina prijave	Da	Da	Vtičnik	Da	Da
Modularnost z vtičniki	Da	Da	Da	Da	Da
Urejanje slik	Da	Vtičnik	Vtičnik	Da	Da
WYSIWYG urejevalnik	Da	Vtičnik	Da	Da	Da
Razširljivost uporabnika	Da	Da	Vtičnik	Vtičnik	Da
Predpomnenje	Da	Da	Vtičnik	Da	Da
Večjezičnost	Vtičnik	Da	Vtičnik	Da	Da
Statična vsebina	Da	Da	Da	Da	Da
Urejanje vsebine	Da	Da	Da	Da	Da
Statistika obiska	Da	Da	Vtičnik	Vtičnik	Vtičnik
UTF-8 podpora	Da	Da	Da	Da	Da
Blog	Da	Da	Da	Vtičnik	Da
Galerija	Vtičnik	Vtičnik	Da	Vtičnik	Da
SEO generator	Da	Da	Da	Da	Da
Urejanje pravicami	Da	Da	Da	Da	Da
Sistem za predloge	Da	Da	Ne	Da	Da

Tabela 4.3: Analiza obstoječih sistemov CMS.

4.1 Načrtovanje ogrodja

Načrt za izdelavo je ključnega pomena za uspešno izdelavo projekta. Naš cilj je bil narediti čim manj zahteven sistem, zato smo tudi mi v ospredje postavili načrtovanje.

Ker smo si za osnovo izbrali obstoječa ogrodja, smo pripravili seznam primernih funkcionalnosti, ki smo jih uporabili in seznam dodatnih modulov, ki bi jih razvili.

Seznam obstoječih funkcionalnosti v ogrodju Django:

- vstopna točka aplikacije,
- vmesnik za povezovanje s podatkovno bazo in pošiljanje poizvedb,
- modul za izdelavo podatkovnih modelov,
- modul za sprejemanje zahtev,
- modul za nastavitve,
- vmesnik za upodabljanje predlog,
- vmesnik za branje SEO povezav.

Seznam razvitih modulov:

- modul za dinamično usmerjanje zahteve,
- modul za avtomatsko generacijo SEO povezav,
- modul za vtičnike,
- modul za branje gradnikov strani,
- modul za nalaganje in upravljanje datotek,
- modul za urejanje slik,
- modul za dodajanje in preverjanje pravic,
- modul za večjezičnost,

- modul za asinhrone klice,
- modul za izdelavo obrazcev,
- sistem za avtomatsko povezovanje podatkovnega modela s SEO povezavami,
- modul za predloge,
- modul za prikaz celostne strani.

Vse uporabljene in razvite module smo definirali kot jedro ogrodja. Zaradi znanega principa DRY (Don't repeat yourself) smo funkcije, ki jih uporablja več modulov, združili v splošne pomočnike, pomočnike za upravljanje z datotekami, pomočnike za sporočila, pomočnike za nadzor nad modeli in pomočnike za generiranje odgovorov. Osnoven podatkovni model za delovanje ogrodja je sestavljen iz tabel: Directory (seznam imenikov, v katerem so shranjene datoteke), File (seznam naloženih datotek), Group (skupine uporabnikov), User (uporabniki), Page (seznam tipskih strani), Seo (seznam SEO povezav), Menu (seznam menijev), MenuItem (seznam elementov v menijih), Language (seznam jezikov za večjezičnost), Phrase (seznam fraz, ki jih je potrebno prevesti) in Translation (seznam prevodov).

4.1.1 Modul za dinamično usmerjanje zahteve

Modul za dinamično usmerjanje zahteve je shranjen v datoteki 'urls.py'. Zahtevo po vsebini glavni usmerjevalnik razdeli na tri dele: stran, gradnik in statično vsebino. Gradnik je osnoven del strani in je lahko: gumb, seznam shranjenih podatkov, slika, meni, idr. Stran je sestavljena iz enega ali več gradnikov. Za razliko od njih, je stran vedno povezana z instanco objekta tipa 'Page'. Ta objekt ima shranjeno konfiguracijo zahtevane strani: povezavo do predhodne strani, ime, status, postavitev, pravice, gradnike in dodatke, kot sta CSS in JavaScript. Statično vsebino predstavljajo slike, datoteke za oblikovanje stila in druge.

4.1.2 Modul za avtomatsko generacijo SEO povezav

Vsaka vsebina je povezana z vsaj enim objektom in vsak objekt je predstavljen s spletno povezavo. Ta povezava se imenuje SEO povezava. Ob ustvarjanju novega objekta, se nekatere lastnosti uporabijo za generiranje ključa, ki je edinstven vsakemu objektu istega tipa. S pomočjo tega se generira SEO povezava za ta objekt.

4.1.3 Modul za vtičnike

Osnovna aplikacija je zgrajene iz jedra in drugih vtičnikov. Vsak vtičnik ima ime in predstavlja eno zaključeno skupino, ki je neodvisna od drugih delov aplikacije. Takšna skupina je lahko blog, galerija, sistem za pošiljanje spletnih novic, nakupovalni modul, ipd. Vtičnik je sestavljen iz podatkovnega modela (predstavljen v datoteki 'model'), namestitvene datoteke (shranjene v datoteki 'install'), gradnikov (shranjeni v modulu 'blocks', kjer vsaka datoteka predstavlja neko skupino gradnikov), predlog (shranjene v modulu 'templates', kjer so predloge razvrščene v druge module), pravic (shranjene v modulu 'permission', kjer vsaka datoteka predstavlja neko skupino pravic) in drugih datotek, ki so lahko pomočniki, statične datoteke, ipd. Sistem je narejen modularno in preko teh vtičnikov ponuja možnost modularnga nadgrajevanja aplikacije, brez dodatnih posegov.

4.1.4 Modul za branje gradnikov strani

Gradnik je najmanjši element, ki predstavlja neko vsebino na strani. Vsak gradnik je povezan z eno metodo, ki je shranjena v eni izmed skupin gradnikov znotraj modula 'blocks'. Ime gradnika je enolično in vsebuje pot do gradnika ter ime metode, ki ga predstavlja. Ob zahtevi po gradniku, sistem poišče njegovo metodo in od nje zahteva vsebino. Vsak gradnik je dostopen programsko, s klicem po imenu ali preko določene spletne povezave, ki se začne z '/blocks' in nadaljuje z imenom gradnika.

4.1.5 Modul za nalaganje in upravljanje datotek

Sistem uporabniku omogoča nalaganje datotek na strežnik. V konfiguraciji aplikacije nastavimo pot, kamor se te datoteke shranjujejo. Vsaka naložena datoteka je lahko prosto dostopna vsem, ki poznajo njeno povezavo oziroma zaprta za lastnika, skupino, itd.

4.1.6 Modul za urejanje slik

Objekt lahko vsebuje polje, ki ima za predstavitev nastavljen modul za urejanje slik. Temu polju lahko določimo tudi dodatne lastnosti za samodejno obrezovanje. Vsaka lastnost vsebuje ime, predpono obrezane slike, širino in višino podano z enoto 'px' in logično vrednost 'crop', ki sistemu dovoli, da samodejno obreže sliko.

4.1.7 Modul za dodajanje in preverjanje pravic

Vsakemu delu aplikacije lahko nastavimo pravice za dostop. Na ta način omejujemo dostop do zaklenjenih delov aplikacije. Vsaka pravica je povezana z metodo, ki se pokliče ob zahtevi te pravice. Vsaka metoda lahko preveri, ali je uporabnik prijavljen oziroma je uporabnik v določeni skupini in ima tako dostop do določene vsebine.

4.1.8 Modul za večjezičnost

Večjezičnost je danes že nuja. Zato smo se odločili, da tudi našemu sistemu dodamo možnost večjezičnosti. Ta del sistema je sicer še v povojih in podpira samo prevajanje fraz, ki jih določi razvijalec. Ta ima tako možnost, da poda seznam jezikov, v katerih bo spletna stran na voljo in v seznam prevodov doda fraze in njihove prevode.

4.1.9 Modul za asinhrone klice

Dinamičnost je pomembna za dobro uporabniško izkušnjo. Pri izdelavi spletne strani se lahko odločimo, da bomo nekatere gradnike strani osveževali brez ponovnega nalaganja celotne strani. To lahko dosežemo z uporabo asinhronih ('ajax') klicev. Kot smo že omenili v podpoglavju 4.1.4, je vsak gradnik dostopen preko SEO povezave, ki jo uporabimo za asinhrono osveževanje vsebine gradnika.

4.1.10 Modul za izdelavo obrazcev

Obrazce lahko sestavimo na dva načina. Prvi način je ročno pisanje in nastavljanje elementov (polj), drugi pa uporaba modula. Modul nam omogoča hitro izdelavo obrazcev, kot so prijavni obrazci, registracijski obrazci, obrazci za naročilo izdelkov, obrazi za kontakt, ipd. Izdelane obrazce lahko prikažemo s pomočjo gradnikov na naši spletni strani, ali pa jih posredujemo drugim, ki jih vključijo na svojo spletno stran. Vsak element je predstavljen z metodo in skupino. Vsaka skupina je shranjena v modulu 'form/elements'. Ker so elementi za obrazce narejeni modularno, jih lahko tako kot vtičnike, poljubno dodajamo. Vsak vtičnik lahko definira svoje elemente, ki jih definira v prej omenjenem modulu. Sistem elemente iz vseh aktivnih vtičnikov avtomatsko zazna in omogoči njihovo uporabo.

Osnovni elementi so: oznaka, html značka, polje za vnos kratkega besedila, polje za vnos gesla, skrito polje, polja za izbiranje ene izmed več vrednosti, polja za izbiro več vrednosti, polje za izbiranje ene izmed več vrednosti iz seznama, polje za izbiranje več vrednosti iz seznama, polje za vnos daljšega besedila, gumb, gumb za konec vnosa in gumb za nastavitev začetnega stanja.

Dodatni elementi, ki uporabljajo knjižnico 'Bootstrap', so: 'bootstrap' polje za izbiranje ene izmed več vrednosti s seznama, 'bootstrap' polje za izbiranje več vrednosti s seznama, polje za vklop ali izklop, polje za vnos datuma, polje za vnos datuma in ure.

Poleg teh dveh skupin jedro vsebuje še polje za urejanje datotek, polje za urejanje slike, polje za elektronski naslov, polje za izbiro modela s seznama, polje za izbiro tipske strani s seznama, polje za urejanje vtičnikov na tipski strani, polje za urejanje predloge tipske strani in polje za izbiro nadrejenega objekta s seznama.

4.1.11 Sistem za avtomatsko povezovanje podatkovnega modela s SEO povezavami

Za hranjenje in predstavitev podatkov potrebujemo postopke, s katerimi opredelimo objekte in pravilno razporedimo njihove atribute. Za to potrebujemo podatkovni model. Kot smo že prej omenili, je vsak objekt povezan s 'Seo' objektom in tako dostopen preko enolične spletne povezave. Ob vsaki zahtevi se iz te povezave prebere ime modela in oznaka, ki predstavlja objekt. Sistem s pomočjo danih podatkov naloži instanco tega objekta, iz katere lahko kasneje preberemo zahtevane podatke.

4.1.12 Modul za predloge

Predloge so HTML datoteke, ki skupaj s konfiguracijsko datoteko definirajo skupine označbenih mest, kamor se naložijo gradniki zahtevane strani. Vsaka tipska stran je povezana z eno predlogo. Primer enostavne predloge z dvema označbenima mestoma '{{ page_title }}' in '{{ main_content }}' je prikazan na sliki 4.1. V konfiguracijski datoteki, ki je prikazana na sliki 4.2, je definirana skupina 'Content', ki vsebuje en element, kamor bo sistem naložil gradnike, označbeno mesto 'main_content'. Predloge so shranjene v vtičnikih kot mape znotraj modula 'templates/layout' in vsebujejo dve datoteki: HTML datoteko (index.html) in konfiguracijsko datoteko (__init__.py).

```
<!DOCTYPE html><html>
        <head>
                <title>{{ page_title }}</title>
        </head>
        <body>{{ content }}</body>
</html>
```

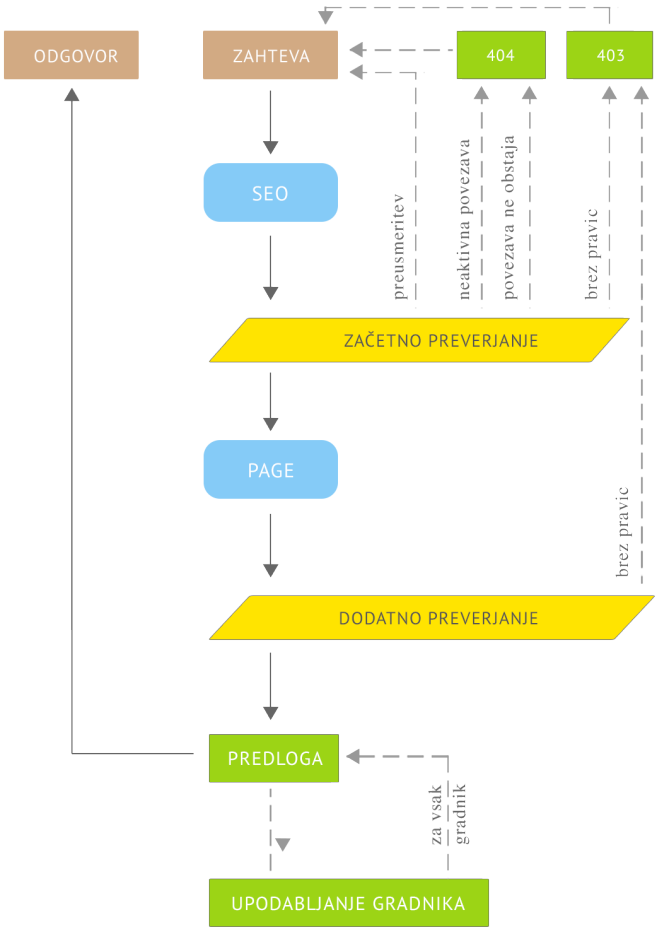
Slika 4.1: Primer enostavnega predloge.

```
name='Predloga'
config=[dict(
        name='Content',
        elements=['main_content']
)]
```

Slika 4.2: Primer konfiguracijske datoteke.

4.1.13 Modul za prikaz celostne strani

Ob vsakem klicu celostne strani se naloži modul za njen prikaz. Najprej se iz zahteve prebere naslov strani in naloži instanca 'Seo' objekta. Če ta objekt ne obstaja oziroma je neaktiven, se uporabnika preusmeri na stran s kodo 404, ki pomeni, da zahtevana stran ne obstaja. Če ima objekt aktivno preusmeritev, se ta izvrši in uporabnika preusmeri na drugo stran. Za tem se preveri pravice za ogled strani. Uporabnika, ki nima vseh pravic, se preusmeri na stran s kodo 403, ki pomeni, da za ogled zahtevane strani nima pravic.



Slika 4.3: Postopek prikaza celostne strani.

V nadaljevanju se naloži instanca 'Page' objekta, ki ima shranjen tip predloge, seznam gradnikov, seznam statičnih datotek (CSS in JavaScript) in seznam zahtevanih pravic za ogled. Najprej se preveri pravice. Če uporabnik nima vseh pravic za ogled strani, se ga preusmeri na stran 403. Nato se vsak gradnik upodobi in postavi na dodeljeno označbeno mesto v predlogi. Na koncu se pravilno porazdeli statične datoteke in vrne upodobljeno stran za odgovor na zahtevo. Celoten postopek prikazuje slika 4.3.

4.2 Načrtovanje sistema za urejanje vsebin

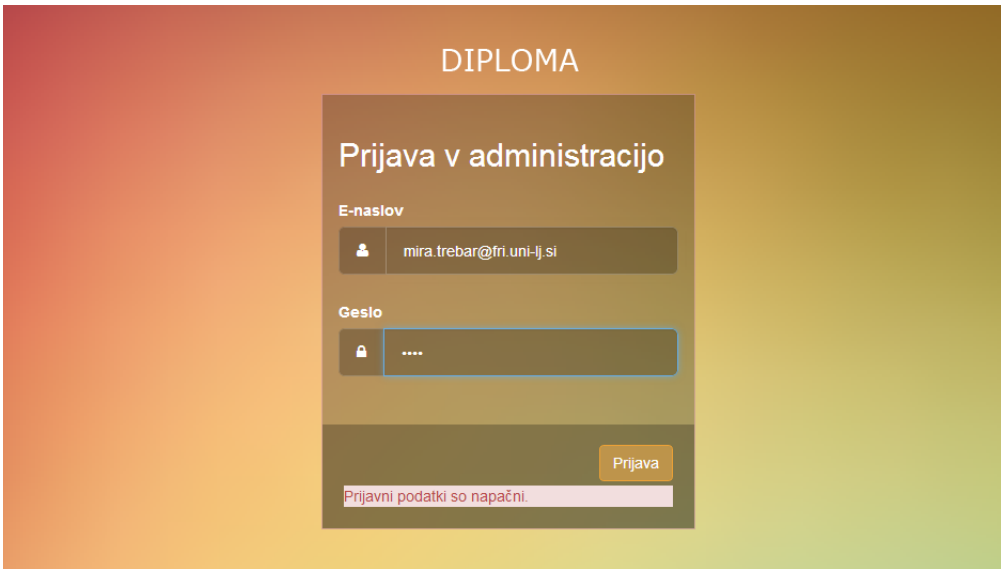
Sistem za urejanje vsebine smo izdelali s pomočjo našega ogrodja. Za seznam funkcionalnosti smo si pomagali z analizo obstoječih CMS sistemov predstavljeno v tabeli 4.3. Uporabnike sistema smo razdelili na tri skupine: uporabnike, urednike in administratorje. Uporabniki lahko urejajo svoje podatke in vsebine, s katerimi so povezani. Uredniki lahko dodatno urejajo druge določene vsebine. Administratorji imajo polni dostop z vsemi pravicami.

Seznam funkcionalnosti, ki smo si jih izbrali za izdelavo CMS-ja:

- obrazec za prijavo uporabnika v sistem,
- glava strani,
- menijski seznam objektov,
- nadzorna plošča,
- modul za seznam objektov izbranega tipa,
- modul za dodajanje novega objekta izbranega tipa,
- modul za urejanje izbranega objekta,
- večjezičnost uporabniškega vmesnika

4.2.1 Obrazec za prijavo uporabnika v sistem

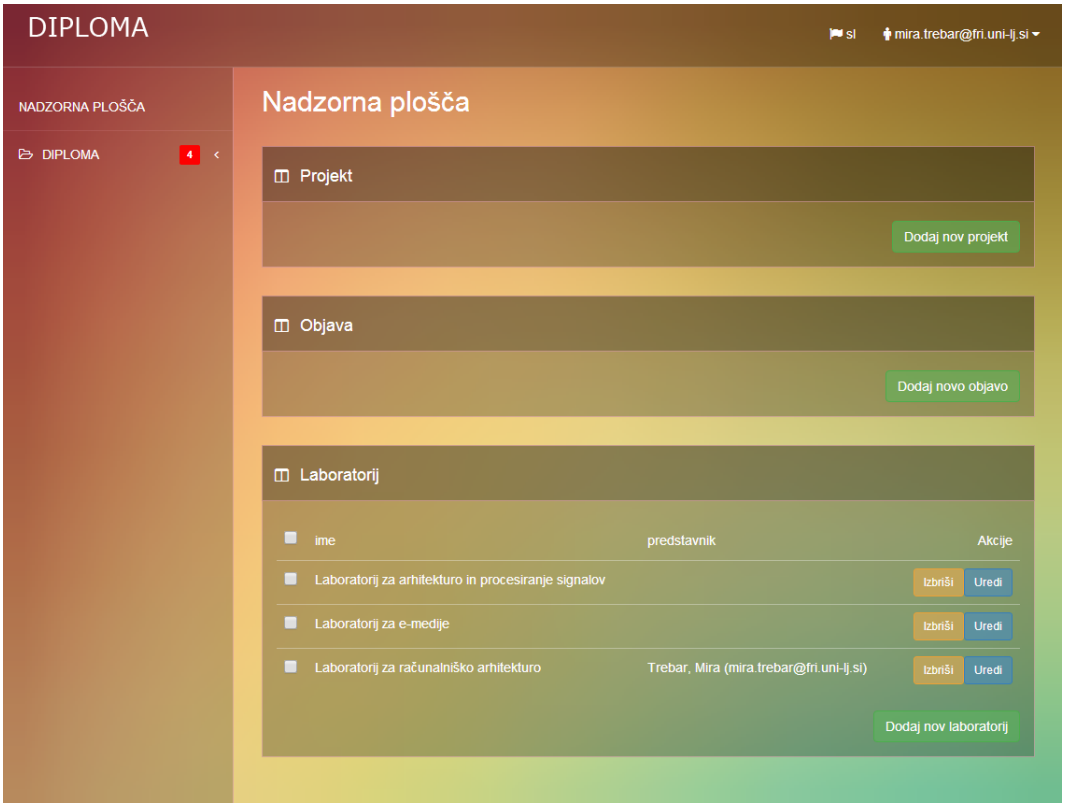
Na vstopni strani je prikazan obrazec, preko katerega se uporabnik prijavi v CMS. Obrazec je sestavljen iz polja za vnos uporabniškega imena in gesla ter prijavnega gumba. Če je uporabnik vnesel napačne podatke, ga sistem po kliku na prijavni gumb opozori s sporočilom, da so prijavni podatki napačni, kar prikazuje slika 4.4, sicer uporabnika prijavi in preusmeri na prvo stran administracije.



Slika 4.4: Prijavni obrazec v sistem za urejanje vsebine.

4.2.2 Glava strani

Vsaka stran, z izjemo prijavne, ima na vrhu vrstico, ki ima na levi strani prikazano izbrano ime ali logotip strani, na desni pa možnost izbire jezika in odjave uporabnika. Vrstica je prikazana na sliki 4.5.



Slika 4.5: Nadzorna plošča administracije.

4.2.3 Menijski seznam objektov

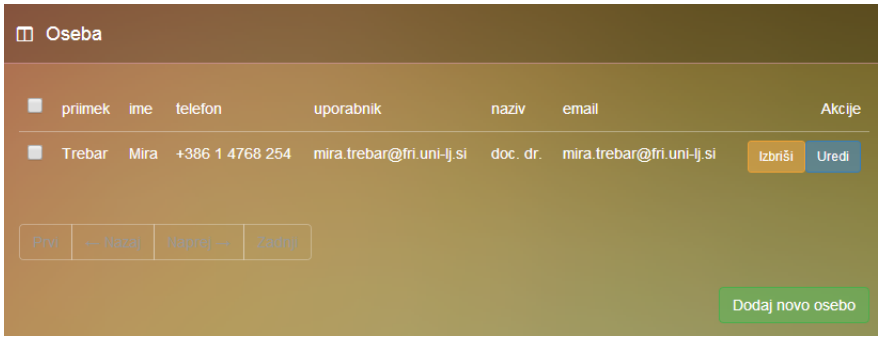
Meni se nahaja na levem delu strani. Element 'nadzorna plošča' je vedno viden, uporabnika pa pripelje na prvo stran administracije. Pod njim je seznam skupin objektov, ki jih lahko uporabnik pregleduje in ureja. Vsak menijski element je povezan s tipsko stranjo in pravicami, ki so potrebne za dostop do njih. Meni, ki ga vidi uporabnik, to je oseba z najmanj pravicami, je prikazan na sliki 4.5.

4.2.4 Nadzorna plošča

Nadzorna plošča je del glavne strani. Na njej so prikazani moduli za seznam objektov izbranega tipa. V naši predstavitvi smo kot tipe objektov izbrali naslednje: projekt, objavo in laboratorij. Nadzorna plošča je prikazana na sliki 4.5.

4.2.5 Modul za seznam objektov izbranega tipa

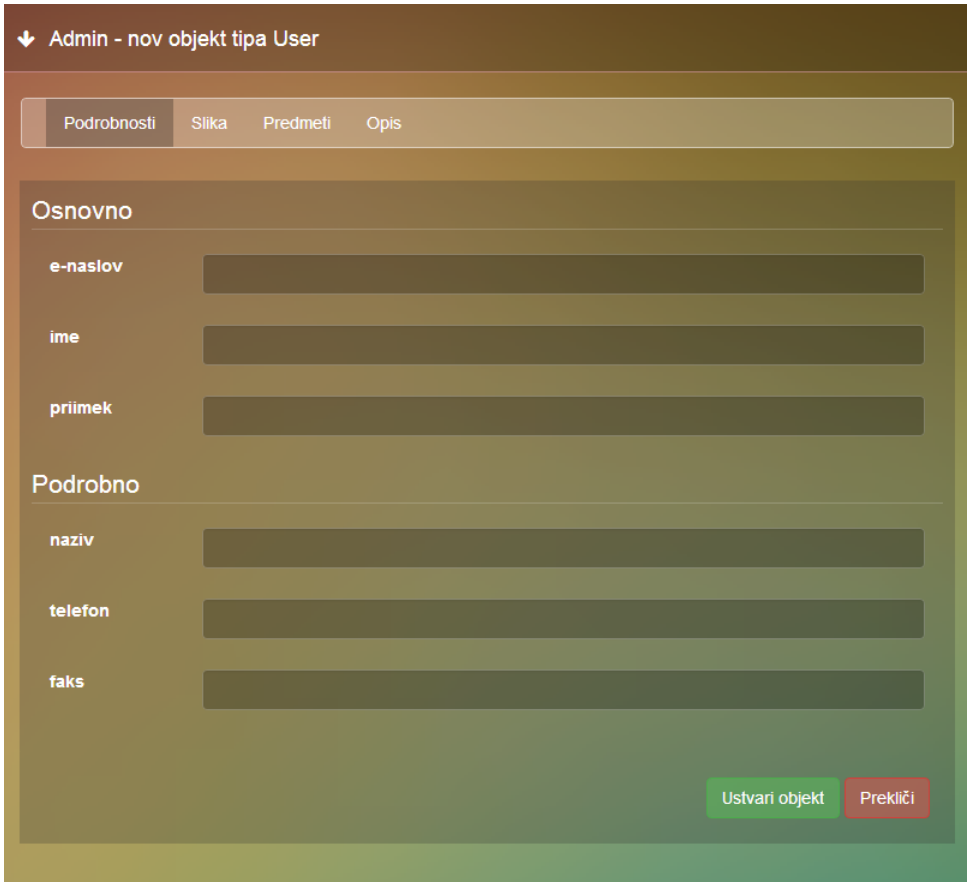
Modulu za seznam objektov izbranega tipa definiramo vhodne parametre. Ti parametri so: tip objekta, število prikazanih objektov na strani, trenutna prikazana stran in logična vrednost za izklop navigacije. Vsak tip objekta ima definirana polja. Objekt tipa 'User' ima definirana polja: email (elektronski naslov, ki se uporablja za identifikacijo), password (geslo, ki se uporabi pri avtentikaciji uporabnika), name (ime), surname (priimek), groups (seznam skupin, v katerih je uporabnik), permissions (seznam dodatnih pravic, ki so mu dodeljene) in last_login (datum zadnje uspešne prijave). Vsako polje ima definiran podatek o tem, ali se prikaže v seznamu objektov ali ne. Vsaka vrstica v seznamu predstavlja en objekt. Na desni strani sta dva gumba, s katerima lahko brišemo (Izbriši) ali uredimo (Uredi) ta objekt. V nogi modula je prikazana navigacija z označeno trenutno stranjo. Če je nastavljen parameter za skritje navigacije, se ta ne prikaže. Modul je prikazan na sliki 4.6.



Slika 4.6: Seznam objektov tipa.

4.2.6 Modul za dodajanje novega objekta izbranega tipa

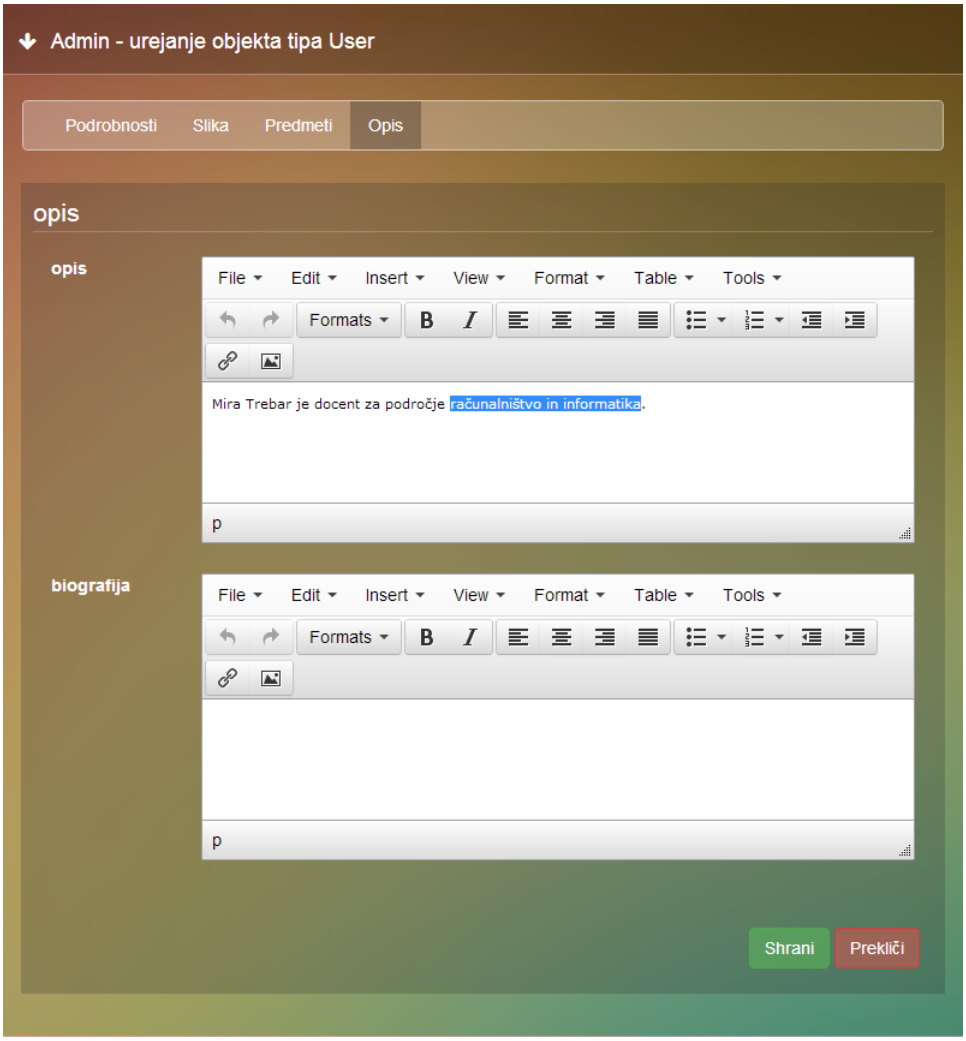
Modul za dodajanje novega objekta izbranega tipa prikaže generiran obrazec, v katerega vnesemo podatke o novem objektu in ga shranimo. Modul za vhodni parameter sprejme tip objekta. Vsakemu polju objekta lahko nastavimo tip vnosnega polja, ki se bo prikazal v obrazcu. Tipi vnosnega polja so predstavljeni v razdelku 4.1.10. Polja so lahko razdeljena tudi v skupine. Obrazec modula je prikazan na sliki 4.7.



Slika 4.7: Obrazec za dodajanje novega objekta.

4.2.7 Modul za urejanje izbranega objekta

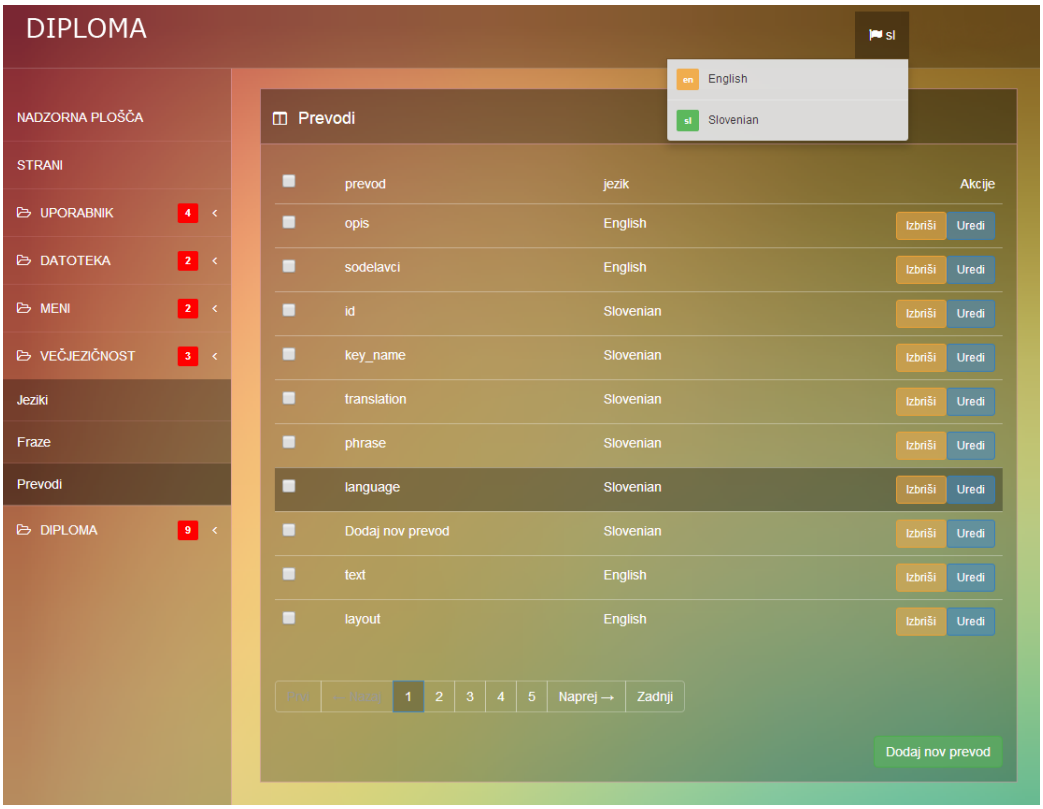
Ta modul je skoraj identičen modulu za dodajanje novega objekta. Kot dodaten parameter sprejme še podatek o objektu, ki ga želimo urediti. Na sliki 4.8 je prikazan modul, ki v izbranih poljih prikaže podatke o urejanem objektu. Ob kliku na gumb Shrani se ti podatki prenesejo na strežnik, kjer se posodobijo.



Slika 4.8: Obrazec za urejanje objekta.

4.2.8 Večjezičnost uporabniškega vmesnika

Sistem za urejanje vsebine v povezavi z modulom za večjezičnost, ki je predstavljen v razdelku 4.1.8, omogoča uporabniku s pravicami za urejanje jezikov in prevodov prevajanje uporabniškega vmesnika. Vsaka prikazana fraza je del vmesnika in je prevedljiva. Če prikazan jezik ni izbran s strani uporabnika, se uporabi angleščina kot privzeti jezik. Modul je prikazan na sliki 4.9.



Slika 4.9: Obrazec za prevajanje vmesnika.

Poglavje 5

Urejanje spletnega mesta

Za predstavitev našega ogrodja in CMS-ja smo izdelali okrnjeno različico spletne strani Fakultete za računalništvo in informatiko. Izdelali smo vtičnik za ogrodje, ki smo ga poimenovali 'Diploma'. Vtičniku smo dodali osnovne sestavne dele: gradnike, predloge, model in statične datoteke. Gradniki, ki smo jih izdelali, so: 'extra' (zadnja dodana novica na glavni strani), 'index' (seznam štirih zadnjih dodanih novic, razen prve), osebje (seznam oseb), novica (podrobnosti novice), objava (podrobnosti objave), projekt (podrobnosti projekta), laboratorij (podrobnosti laboratorija), predmet (podrobnosti predmeta) in oseba (podrobnosti osebe). Metoda gradnika oseba je prikazana na sliki 5.1 in predloga, ki jo uporabi za upodobitev osebe, je prikazana na 5.2. Vse gradnike smo shranili v skupino 'main'.

```
# -*- coding: utf-8 -*-
#! /usr/bin/python
#
#  Diploma index
#

from core.helper.common_help import get_template
from apps.diploma.models import Oseba, Objava, Laboratorij, Projekt
from django.db.models import Q

def oseba(request, block):
    template = get_template('diploma/oseba.html')

    object_id = request.wepo.seo.content
    oseba = Oseba.objects.filter(id=object_id)
    if oseba and len(oseba) > 0:
        oseba = oseba[0]

    # load objave
    objave = Objava.objects.filter(avtor=oseba).distinct()

    # find laboratories
    query = Q(predstavnik=oseba)
    query |= Q(sodelavci=oseba)
    laboratoriji = Laboratorij.objects.filter(query).distinct()

    # load projects
    query = Q(lastnik=oseba)
    query |= Q(sodelavci=oseba)
    projekti = Projekt.objects.filter(query).distinct()

    context = {
        'static': MEDIA_URL,
        'oseba': oseba,
        'objave': objave,
        'laboratoriji': laboratoriji,
        'projekti': projekti
    }

    return template.render(context)
```

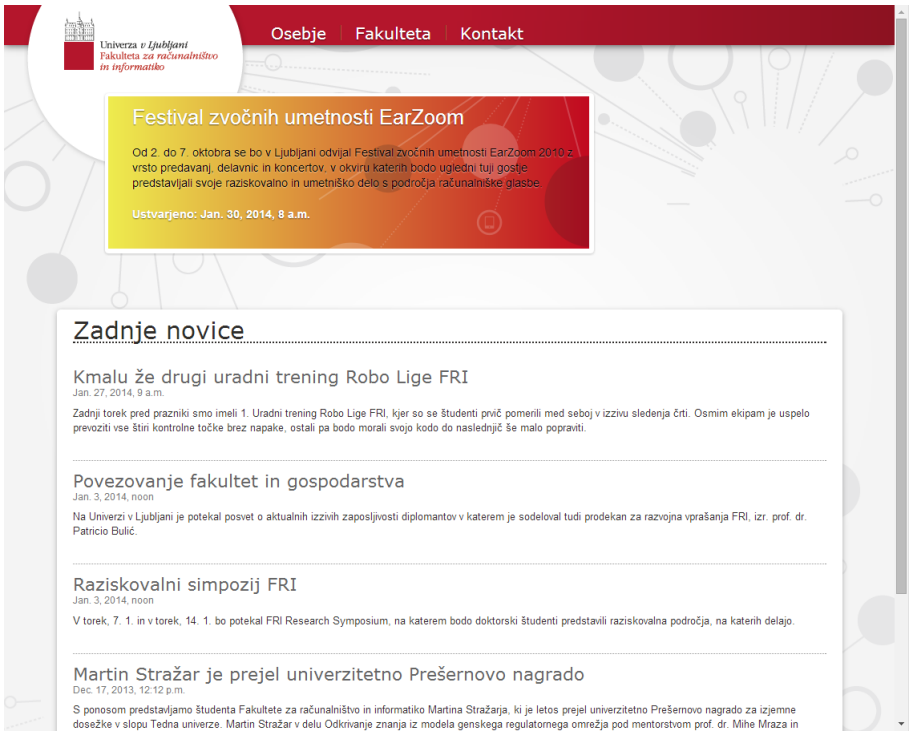
Slika 5.1: Gradnik za prikaz podrobnosti osebe.

```
{% autoescape off %}
{% load core_filters %}

<h1 class="page-title">{{ oseba.naziv }} {{ oseba.name }} {{ oseba.surname }}</h1>

<div class="l-labeledList_cf">
   {% if oseba.image %}
   <div class="l-floatRight_imgFrame"><img src="{{_oseba.image|image:"osebevelika"_}}"></div>
   {% endif %}
   <div><strong>Ime:            </strong> <span>{{ oseba.name }}</span></div>
   <div><strong>Priimek:        </strong> <span>{{ oseba.surname }}</span></div>
   <div><strong>e-Naslov:       </strong> <span>{{ oseba.email }}</span></div>
   <div class="separator"></div>
   <div><span>{{ oseba.opis }}</span>  </div>
</div>

<div class="separator"></div>

<section>
   <h2 class="page-sub-title">Predmeti</h2>

   <div class="content">
      {% for predmet in oseba.predmeti_nosilec.all %}
         <div><a href="{{_predmet.get_seo_}}"><strong> {{ predmet }}</strong>, nosilec</a>
</div>
      {% endfor %}
   </div>

   <div class="content">
      {% for predmet in oseba.predmeti_asistent.all %}
         <div><a href="{{_predmet.get_seo_}}"><strong> {{ predmet }} </strong></a>
</div>
      {% endfor %}
   </div>
</section>

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

{% if oseba.biografija %}
<section>
   <h2 class="page-sub-title">Biografija</h2>

   <div class="content">
      <div>{{ oseba.biografija }}</div>
   </div>
</section>
{% endif %}

{% endautoescape %}
```

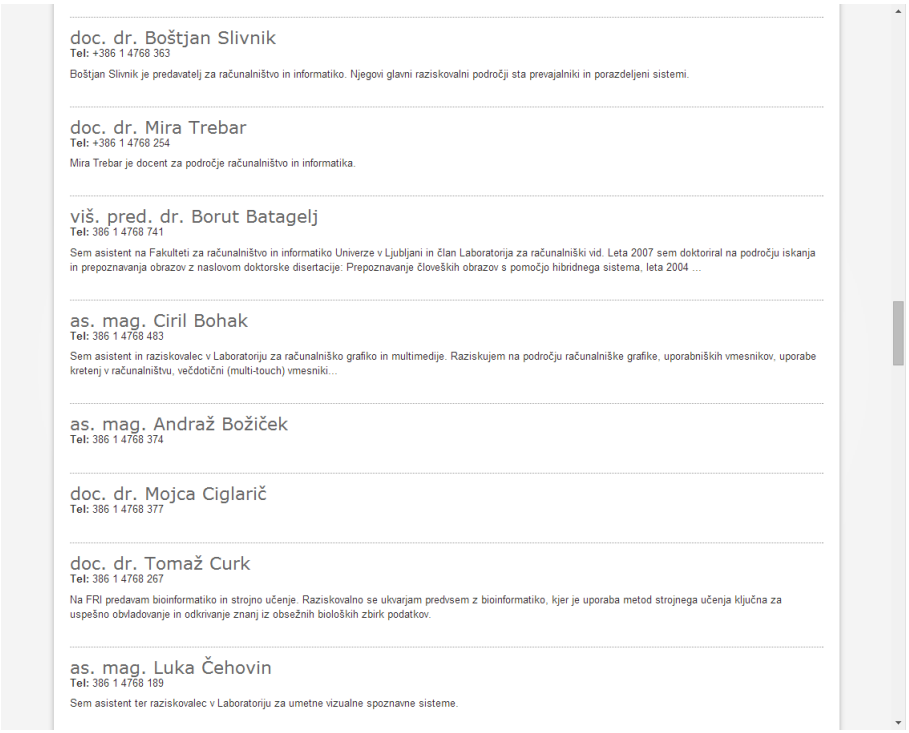
Slika 5.2: Del predloge za prikaz podrobnosti osebe.

5.1 Predstavitev podstrani

Spletno stran smo razdelili na osem tipskih podstrani: vstopno stran, seznam oseb, podrobnosti osebe, podrobnosti laboratorija, podrobnosti predmeta, podrobnosti projekta, podrobnosti objave in podrobnosti novice. Vsaka podstran je razdeljena na dva dela: glavo in osrednji del. Glava strani je razdeljena na logotip, ki je postavljen levo zgoraj in navigacije, ki se nahajajo desno od logotipa. Navigacija vsebuje povezave do seznama oseb, stare strani Fakultete za računalništvo in informatiko in povezavo do kontakta. V osrednjem delu je prikazana vsebina zahtevane strani.
Vstopna stran je prikazana na sliki 5.3. Za njeno izdelavo smo uporabili gradnika 'extra' in 'index'. Seznam oseb je prikazan na sliki 5.4. Uporabili smo gradnik 'osebje'.

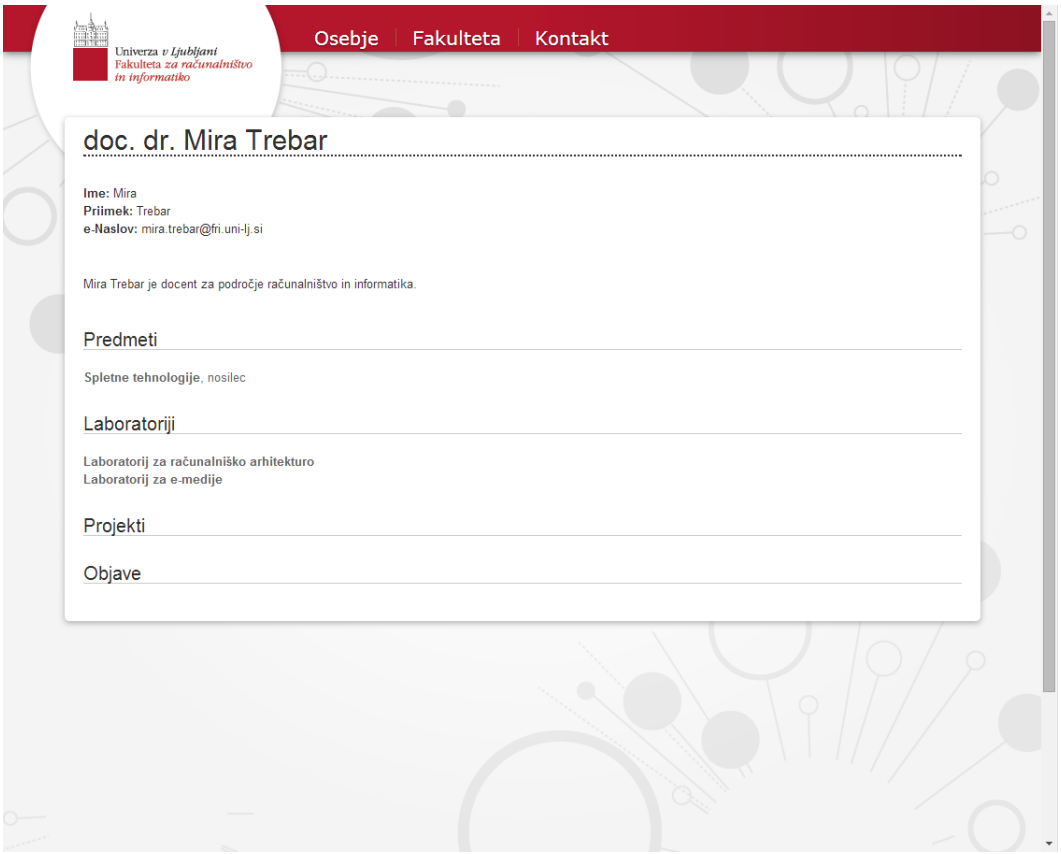


Slika 5.3: Vstopna stran Fakultete za računalništvo in informatiko.



Slika 5.4: Seznam oseb.

Na podstrani za prikaz podrobnosti osebe smo uporabili gradnik 'oseba', ki je prikazan na sliki 5.5. Ostale podstrani so si med seboj zelo podobne in se razlikujejo samo v vsebini. Na podstrani podrobnosti laboratorija smo uporabili gradnik 'laboratorij', na podstrani predmeta gradnik 'predmet', na podstrani projekta gradnik 'projekt', na podstrani objave gradnik 'objava' in na podstrani novice gradnik 'novica'.



Slika 5.5: Seznam oseb.

5.2 Namestitev končnega produkta

Produkt lahko namestimo na kateri koli strežnik, ki podpira programski jezik Python. Postopek namestitve je za vse sisteme enak. Za uporabo sistema moramo namestiti nekatere programe.

Namestitev programov, ki so potrebni za delovanje:

- spletni strežnik uWSGI, ki sprejema zahteve na določenih vratih in jih posreduje aplikaciji [31],
- program memcached, ki se uporablja za predpomnenje podatkov [32],

- knjižnica GraphicsMagick, ki se uporablja za procesiranje slik [33].

Namestitev produkta:

- kopiranje izvorne kode aplikacije na strežnik,
- zagon spletnega strežnika uWSGI, ki kaže na konfiguracijsko datoteko uwsgi.py, shranjeno v korenskem imeniku aplikacije.

Poglavje 6

Sklep in ugotovitve

V diplomskem delu smo razvili ogrodje za izdelavo spletnih aplikacij in sistem za upravljanje spletnih vsebin. Oba sistema smo uporabili pri izdelavi predstavitvene strani Fakultete za računalništvo in informatiko. Z ogrodjem smo postavili stabilno osnovo, ki je modularno razširljiva. Sistem za upravljanje spletnih vsebin predstavlja enega izmed vtičnikov. Pisanje vtičnikov je zelo enostavno in intuitivno. Za osnovo si lahko uporabnik izbere enega izmed že obstoječih vtičnikov in ga prilagodi svojim potrebam. Izmed vseh lastnosti ogrodja bi kot glavni dve prednosti lahko izpostavili modul za vtičnike, ki samodejno prepozna novo dodane in modul za izdelavo obrazcev, s pomočjo katerega lahko hitro in enostavno izdelamo različne obrazce.

Zaradi razvitih modulov, je izdelava spletne strani zelo enostavna. Z uporabo modula za dinamično usmerjanje zahteve nam ni potrebno pisati pravil, ki bi zahteve pravilno preusmerile. Modul za avtomatsko generiranje SEO povezav pripomore k uvrstitev spletne strani na višje mesto v iskalnikih.

Ker so vtičniki zaključena celota in neodvisni drug od drugega, je tako ogrodje idealno za razvoj aplikacij, ki jih razvija večje število razvijalcev. Izdelana rešitev ima še nekatere pomankljivosti. Nobenega merila nimamo, kako dobro bi se odrezala pri zelo velikih aplikacijah. Izdelava takšnega projekta bi seveda vzela veliko časa. V praksi bi to pomenilo tri do štiri mesece dela. Druga pomankljivost ogrodja je pomanjkanje podpore za večjezično

vsebino. Modul za večjezičnost podpira prevajanje statičnih vsebin, ne pa tudi dinamičnih. V prihodnje želimo odpraviti vse slabosti našega ogrodja in nadgraditi CMS sistem. Dodati želimo večjezičnost dinamičnih vsebin, modul za zgodovino sprememb, idr. Nadobudnemu bralcu prepuščamo implementacijo novih modulov. Seveda je glavni in osnovni cilj našega sistema, da ponudi uporabniku čim večje število vtičnikov, ki bi mu izdelavo aplikacije zelo poenostavila. Na tem področju se trudimo z iskanjem novih razvijalcev, ki bi bili pripravljeni nekaj svojega prostega časa žrtvovati za razvoj potrebnih modulov.

Literatura

[1] A. Martelli, D. Ascher, "Python Cookbook", *O'Reilly Media*, 2002.

[2] C. Newman, "SQLite", *Sams* 2004.

[3] K, Cwalina, B. Abrams, "Framework Design Guidelines: Conventions, Idioms, and Patterns for Reusable .NET Libraries", *Addison-Wesley Professional* 2005.

[4] A. Holovaty, J. Kaplan-Moss, "The Django Book", *Apress* 2007.

[5] comScore `https://www.comscore.com/Insights/Press_Releases/2013/11/comScore_Reports_47_5_Billion_Dollar_in_Q3_2013_Desktop_Based_US_Retail_ECommerce_Spending_Up_13_Percent_vs_Year_Ago` *comScore Reports $47.5 Billion in Q3 2013 Desktop-Based U.S. Retail E-Commerce Spending, Up 13 Percent vs. Year Ago*, 04.24.2014

[6] [2014] Statistika podjetja Wal-Mart. Dostopno na: `http://www.statisticbrain.com/wal-mart-company-statistics/`.

[7] [2014] Python. Dostopno na: `https://www.python.org/`

[8] [2014] SQLite. Dostopno na: `http://sqlite.org/`, 04.24.2014

[9] [2014] D programming language. Dostopno na: `http://dlang.org/`

[10] [2014] Go Programming Language. Dostopno na: `http://golang.org/`

[11] [2014] Martin Fowler. Dostopno na: http://en.wikipedia.org/wiki/Martin_Fowler

[12] [2014] Microsoft Corporation. Dostopno na: http://www.microsoft.com/

[13] [2014] Podatkovne baze. Dostopno na: http://en.wikipedia.org/wiki/Database

[14] [2014] GNU General Public Licence. Dostopno na: https://www.gnu.org/copyleft/gpl.html

[15] [2014] Flash Builder. Dostopno na: http://www.adobe.com/products/flash-builder.html

[16] [2014] Eclipse. Dostopno na: https://www.eclipse.org/

[17] [2014] MonoDevelop. Dostopno na: http://monodevelop.com/

[18] [2014] Code::Blocks. Dostopno na: http://www.codeblocks.org/

[19] [2014] Microsoft Visual Studio. Dostopno na: http:/%/www.visualstudio.com/

[20] [2014] LispWorks. Dostopno na: http://www.lispworks.com/

[21] [2014] NetBeans. Dostopno na: https://netbeans.org/

[22] [2014] Aptana Studio. Dostopno na: http://aptana.com/

[23] [2014] Delphi. Dostopno na: http://www.embarcadero.com/products/delphi

[24] [2014] Komodo IDE. Dostopno na: http://komodoide.com/

[25] [2014] Zend Studio. Dostopno na: http://www.zend.com/en/products/studio/

[26] [2014] PyCharm. Dostopno na: http://www.jetbrains.com/pycharm/

[27] [2014] Xcode. Dostopno na: `https://developer.apple.com/xcode/`

[28] [2014] JavaScript. Dostopno na: `https://developer.mozilla.org/en/docs/Web/JavaScript`

[29] [2014] HTML. Dostopno na: `http://www.w3.org/`

[30] [2014] DTD - Document Type Definition. Dostopno na: `http://www.w3schools.com/DTD/`

[31] [2014] uWSGI. Dostopno na: `http://projects.unbit.it/uwsgi/`

[32] [2014] Memcached. Dostopno na: `http://memcached.org/`

[33] [2014] GraphicsMagick. Dostopno na: `http://www.graphicsmagick.org/`