SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

ZAVRŠNI RAD br. 4001

**Web-aplikacija za planiranje putovanja**

Ante Tomić

Zagreb, lipanj 2015.

SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

ZAVRŠNI RAD br. 4001

**Web-aplikacija za planiranje putovanja**

Ante Tomić

Zagreb, lipanj 2015.

Izvornik

Posebno se zahvaljujem mentoru doc. dr. sc. Igoru Mekteroviću na prihvaćanju teme, razumijevanju, strpljenju, idejama i poticajnim sugestijama prilikom izrade ovog završnog rada.

Sadržaj

[1. Uvod 1](#_Toc421042075)

[2. Opis problema i korisnički zahtjevi aplikacije 3](#_Toc421042076)

[3. Opis podatkovnog modela aplikacije 4](#_Toc421042077)

[4. Pregled korištenih tehnologija u izradi aplikacije 5](#_Toc421042078)

[4.1. .NET Framework (radni okvir) 5](#_Toc421042079)

[4.2. ASP.NET radni okvir 7](#_Toc421042080)

[4.2.1. ASP.NET MVC (Model-View-Controller) 7](#_Toc421042081)

[4.3. Entity Framework 8](#_Toc421042082)

[5. Izrada aplikacije i opis rješenja 9](#_Toc421042083)

[6. Zaključak 9](#_Toc421042084)

[7. Literatura i reference 10](#_Toc421042085)

[8. Sažetak 11](#_Toc421042086)

[9. Summary 11](#_Toc421042087)

[10. Ključne riječi 11](#_Toc421042088)

# Uvod

Pojam aplikacije možemo interpretirati na više načina, ali je nama najvažnija računalna interpretacija pojma. Ona nam kaže da je aplikacija program ili softver koji je stvoren da radi određeni posao. Postoje mnoge vrste aplikacija te su one podijeljene na obrazovne, simulacijske, zabavne, simulacijske i mnoge druge vrste. Nas najviše zanimaju web-aplikacije bilo kakve vrste koje su zapravo računalne aplikacije kojima se pristupa preko internetskih preglednika pomoću Interneta ili bilo koje druge mreže.

Razvoj web-aplikacija počinje razvojem samog Weba, a nastavlja se tijekom 90-ih godina prošlog stoljeća i traje i do današnjeg dana. One koriste web-dokumente napisane u standardnim formatima poput HTML-a ili JavaScripta koji su podržani od strane većine internetskih preglednika. Razvoj i korištenje tehnologija poput Flasha, Silverlighta, JavaScripta i mnogih drugih omogućuje korištenje posebnih metoda i značajki pojedine tehnologije koje možda inače ne bi bile dostupne.

Web-aplikacije su sve popularnije zbog sve veće prisutnosti internetskih preglednika u različitim platformama što olakšava pristup bilo kojoj web-aplikaciji koja je dostupna na izabranoj mreži. Takve aplikacije također koriste internetski preglednik kao klijent koji se povremeno naziva tanki klijent jer internetski preglednik sadrži samo sloj prezentacijske logike. Glavni razlog popularnosti web-aplikacija je mogućnost korištenja bez distribucije i instalacije bilo kakvih dodatnih aplikacija na mogućim klijentskim računalima što omogućuje podršku na više platformi.

Svaka se aplikacija sastoji od servisa koji su udruženi u rješenje te se definiraju kao dio aplikacijske logike koji uključuje metode za implementaciju operacije, funkcije ili transformacije. Servisi su pridruženi akcijama su ih je moguće pristupiti preko mreže pomoću objavljenog sučelja koje sadrži specifikaciju sučelja. Korisniku nije važna implementacija servisa, nego mu je važnija sposobnost servisa da obavi traženu radnju. Jedan se servis obično sastoji od korisničkih, poslovnih, podatkovnih i sistemskih servisa od kojih svaki dio pruža određenu funkcionalnost.

Servisi su organizirani u rješenje pomoću odabrane aplikacijske arhitekture. Aplikacijska se arhitektura sastoji od definicija, pravila i veza koje formiraju strukturu aplikacije, ali ne sadrži implementacijske detalje. Neke od najkorištenijih aplikacijskih arhitektura su: klijent/server, višeslojna, nepamteća, pričuvna i višeslojna klijent-pričuvna-nepamteća-pričuvna-server arhitektura. Aplikacije su obično podijeljene u logičke cjeline nazvane slojevi od kojih je svakom sloju pridružena određena uloga.

Najkorištenija arhitektura, koja se koristila u izradi ovog završnog zadatka, je višeslojna arhitektura. Ona je napredna verzija klijent/server arhitekture koja se sastoji od hijerarhijskih slojeva u koje su spremljeni odgovarajući razni servisi. Najčešće se koristi troslojna arhitektura koja se sastoji od prezentacijskog, poslovnog i podatkovnog sloja. Slojevi enkapsuliraju servise i štite jedan servis od drugog uz pružanje pojednostavljenog kompleta sučelja za dijeljena sredstva. Prednosti ove arhitekture su poboljšana skalabilnost sustava, visoka sigurnost, mogućnost uravnoteživanja servisa između dostupnih resursa i raznih slojeva te mogućnost implementacije sigurnost na definiranim granicama uz minimalni utjecaj na komunikacije između servisa unutar sistema, dok su neki od nedostataka povećanje općih troškova, kompleksnosti i latentnosti.

Cilj je ovog završnog rada bila izrada web-aplikacije za planiranje putovanja koja bi pokrila sve bitne dijelove aplikacije poput objave sadržaja, spajanja sadržaja u stranicu, pregleda objavljenih sadržaja i stranica, dijeljenja i mnogih drugih dijelova. S obzirom na širinu ovog područja, neki dijelovi nisu toliko opširno uzeti u obzir, ali je aplikacija napravljena što realističnije i detaljnije te je ostavljena mogućnost za buduće nadogradnje sustava.

# Opis problema i korisnički zahtjevi aplikacije

Aplikacija za upravljanje putovanjima je web-aplikacija koja korisnicima omogućuje izradu virtualnih kolaža ili fascikala u obliku stranica u kojima bi se nalazili sadržaji koje je korisnik izdvojio ili odabrao. Cilj takvih stranica je isplanirati putovanje slaganjem raznih članaka i drugih vrsta sadržaja u jednu cjelinu podijeljeno po lokacijama čime bi imao sve zanimljive sadržaje sklopljene na jedno mjesto (stranicu). Nakon što je stranica stvorena, korisnik ju može zadržati za sebe, podijeliti sa svim korisnicima i posjetiteljima ili odabranom grupom korisnika.

Implementirana aplikacija bi trebala gostu (neregistriranom korisniku) pružiti mogućnost prijave, registracije i pregleda najposjećenijih javnih stranica koje su napravili registrirani korisnici. Nakon registracije ili prijave gost postaje registrirani korisnik te dobiva sve mogućnosti koje imaju registrirani korisnici. Gosti ne mogu vidjeti dodatne sadržaje jer im to onemogućuje aplikacija te se moraju prijaviti ili registrirati.

Registrirani korisnik može pregledavati sve dostupne sadržaje, stranice za koje ima dozvolu pregleda, vlastite grupe te korisničke podatke koje može i uređivati. Također može dodavati, uređivati i uklanjati grupe kojima može pridružiti korisnike ili iste ukloniti iz grupe, lokacije koje uključuju vrstu lokacije, ime lokacije i državu, sadržaje koje drugi korisnici mogu stavljati na svoje stranice i stranice za koje može namjestiti postavke vidljivosti. Stranice koje korisnik može vidjeti su vlastite (autorske), javne, stranice na kojima sudjeluje kao dodatni sudionik ili stranice za koje mu je omogućen pregled pomoću grupe, a stranicama se još mogu pridružiti dodatni sudionici stranice koji će onda imati sve dozvole za dodavanje sadržaja i uređivanje stranice kao i autor odabrane stranice i tagovi koji se mogu koristiti za pretraživanje po pridruženom tagu.

# Opis podatkovnog modela aplikacije

# Pregled korištenih tehnologija u izradi aplikacije

## .NET Framework (radni okvir)

.NET radni okvir je Microsoftov softverski radni okvir koji se uglavnom izvodi na Microsoft Windows platformama. Sastoji se od velike biblioteke klasa *Framework Class Library (FCL)* koja se koristi unutar nekoliko programskih jezika te pruža interoperabilnost jezika zahvaljujući čemu svaki programski jezik može koristiti kod pisan u drugom jeziku. FCL pruža korisničko sučelje, pristup podatcima, mogućnost povezivanja s bazama podataka, kriptografiju, razvoj web-aplikacija, numeričke komunikaciju i mrežnu komunikaciju. Programeri razvijaju aplikaciju kombiniranjem vlastitog koda s .NET radnim okvirom i drugim bibliotekama. Najčešće se koristi *Visual Studio*, Microsoftova integrirana razvojna okolina za .NET aplikacije.



Slika 4.1. – Model .NET Frameworka

CLR (*Common Language Runtime*) je komponenta virtualne mašine u .NET razvojnom okviru koja upravlja kodom i izvršavanjem .NET aplikacija. Kod svih programa u .NET-u je tzv. upravljani kod te se sad kod u jednom trenu prevodi u stogovni međujezik CIL (*Common Intermediate Language*). CLR provodi JIT (*just-in-time*) prevođenje gdje se program prvo prevodi u stogovni međujezik te se prilikom prvog izvršavanja programa CIL kod pretvori u strojni kod platforme što se naziva JIT prevođenje. Prilikom svakog sljedećeg izvršavanje aplikacije se uzima strojni kod iz pričuvne (*cache*) memorije čime se postiže ubrzanje rada programa. Od mogućnosti CLR pruža i oslobađanje resursa putem korištenja sakupljača smeća koji briše objekte koji se više ne koriste, višejezično okruženje koje koristi CTS (*Common Type System*) gdje se nalaze tipovi podataka koji su zajednički svim jezicima u .NET-u, strukturirano hvatanje iznimki i pakiranje CIL koda u asemblije, najmanje cjelina izvršnog koda u .NET-u koje su zapravo komponente koje opisuju same sebe.

BCL (*Base Class Library*) je podskup FCL-a i odnosi se na cijelu klasu razreda koji dolaze uz .NET razvojni okvir. Neki od najpoznatijih biblioteka iz BCL-a su: Windows forme, ADO.NET, ASP.NET, *Language Integrated Query* (LINQ), *Windows Presentation Foundation* (WPF), *Windows Communication Foundation* (WCF) i *Workflow Fountation* (WF). Razredi su grupirani u prostore imena (*namespace*) poput *System*, *System.Collections*, *System.Data*, *System.IO*, *System.Text*, *System.Threading* i mnogih drugih.

## ASP.NET radni okvir

ASP.NET radni okvir je aplikacijski radni okvir otvorenog koda koji radi na serverskoj strani te je izrađen za web-razvoj radi izrade dinamičkih web-stranica. Izrađen je od strane Microsofta s ciljem izrade dinamičkih web-stranica, web-aplikacija i web-servisa. On je danas u procesu reimplementacije u moderni i modularni internetski radni okvir MVC 6 koji bi se sastojao od radnih okvira ASP.NET MVC, ASP.NET Web API i ASP.NET Web Pages (Razor).

ASP.NET web-stranice, koje se službeno zovu web-forme (*Web Forms*), glavni su dio razvoja aplikacije u ASP.NET-u.

### ASP.NET MVC (Model-View-Controller)

Slika 4.2. – Model-View-Controller koncept. Puna crta prikazuje direktnu povezanost, dok isprekidana prikazuje indirektnu povezanost kod promatrača.



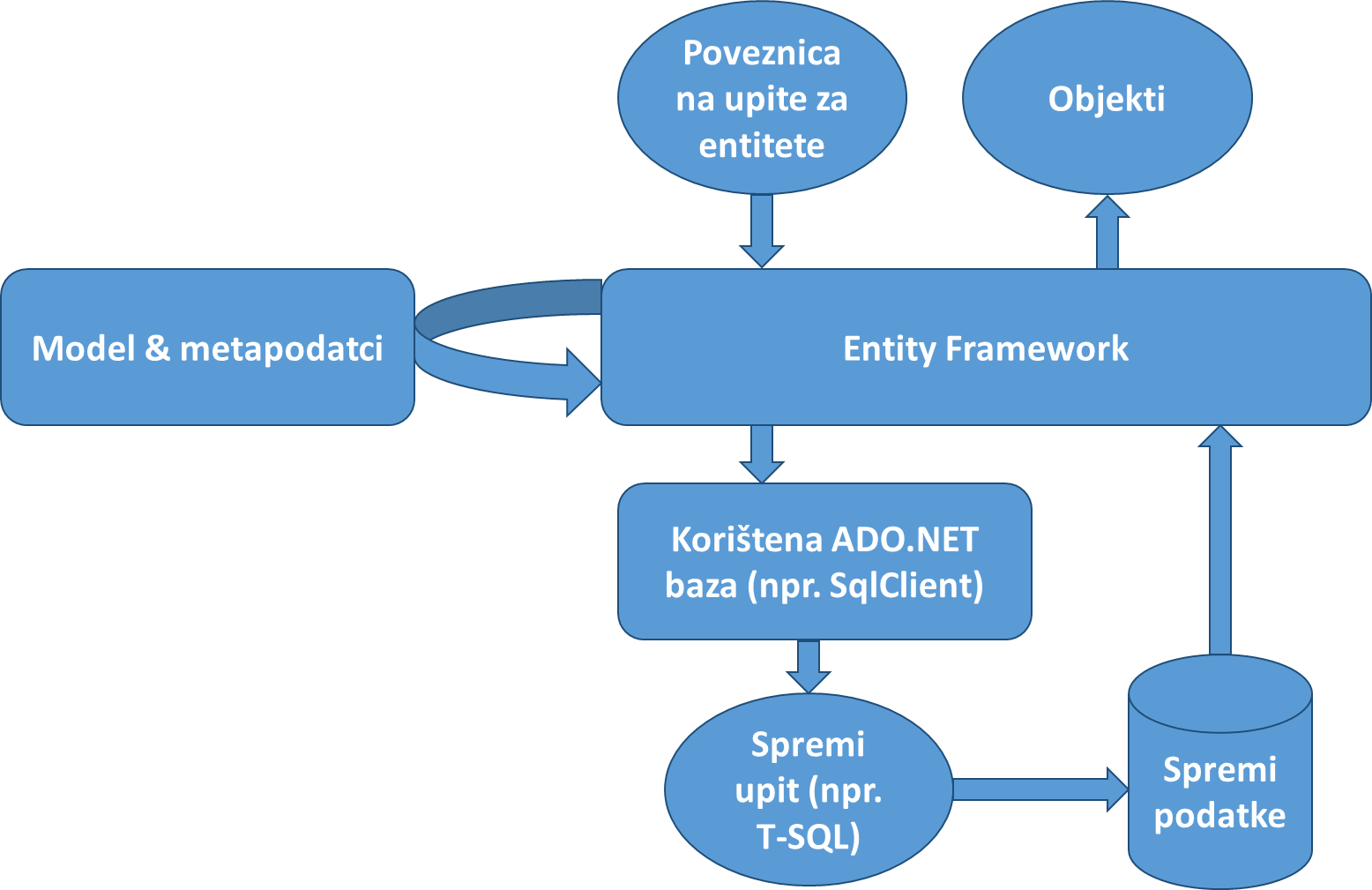
ASP.NET MVC je aplikacijski radni okvir otvorenog koda koji je temeljen na *Model-View-Controller* softverskoj arhitekturi. MVC je uzorak za razvoj aplikacija koje su dobro dizajnirane, koje se mogu testirati i koje se lako održavaju, a sastoji se od tri cjeline: modela (*Model*) koji sadrži klase kojima se modeliraju podatci i ostvaruje validacija, pogleda (*View*) koji prikazuje prethodno modelirane podatke i upravitelja (*Controller*) koji upravlja korisničkim zahtjevima tako da hvata zahtjeve preglednika, dohvaća podatke iz modela i prosljeđuje ih za prikaz u pogled. MVC uzorak nam pomaže u stvaranju aplikacija koje odvajaju različite dijelove aplikacije uz pružanje barem nekakve veze između stvorenih elemenata.

Neke od glavnih značajki ASP.NET MVC-a su: razdvajanje aplikacijskih zadataka (slojeva), mogućnost ispitivanja, ispitivanje čitavog sustava (funkcijsko ispitivanje i ispitivanje performansi), proširivost, podrška za ASP.NET usmjeravanje, podrška za korištenje označavanja u postojećim ASP.NET stranicama, korisničkim kontrolama i stranicama zaglavlja (*master pages*) te podrška za dostupne ASP.NET značajke.

## Entity Framework

Entity Framework (u daljnjem tekstu: EF) je ADO.NET radni okvir za objektno-relacijsko preslikavanje koji je izvorno bio dio .NET radnog okvira, dok je trenutno paket otvorenog koda (*Open Source*) čiji je izvorni kod dostupan javnosti na uvid, korištenje, izmjene i daljnji razvoj ili izmjenu funkcionalnosti odabrane aplikacije. Arhitekti i programeri podatkovno-orijentiranih aplikacijama su se mučili s ostvarivanjem dva različita cilja jer moraju modelirati entitete, veze i logiku poslovnih problema koje rješavaju te također moraju raditi s bazama podataka koje koriste za spremanje i dohvaćanje podataka. Podatci mogu biti spremljeni na više mjesta ili baza podataka od kojih svaka može imati svoj vlastiti protokol.

EF omogućuje programerima da rade s podatcima u obliku objekata i svojstava karakterističnih samo za određenu domenu bez brige o osnovnim entitetima i atributima gdje su dohvaćeni podatci spremljeni. Mogu raditi na većoj razini apstrakcije gdje mogu razviti aplikacije s manje koda temeljene na podatcima dohvaćenim pomoću EF-a.



Slika 4.3. – Entity Framework i procesiranje rezultata

## Javascript i jQuery

# Izrada aplikacije i opis rješenja

# Zaključak

# Literatura i reference

SREDITI KASNIJE

http://en.wikipedia.org/wiki/Web\_application

K. Harding, Analyzing requirements and defining Microsoft .net solution architectures. Microsoft Press, 2003.

http://en.wikipedia.org/wiki/.NET\_Framework

http://en.wikipedia.org/wiki/ASP.NET

https://msdn.microsoft.com/en-us/library/dd381412%28v=vs.108%29.aspx

https://msdn.microsoft.com/en-us/library/vstudio/bb399567%28v=vs.100%29.aspx

# Sažetak

# Summary

# Ključne riječi