Izrada web aplikacije u programskom jeziku C#

Kratofil, Dominik

Undergraduate thesis / Završni rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:999545

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni preddiplomski studij računarstva

IZRADA WEB APLIKACIJE U PROGRAMSKOM JEZIKU C#

Završni rad

Dominik Kratofil

Osijek, 2017.

SADRŽAJ

1. UVOD 1

1.1 Zadatak završnog rada 1

2. RAZVOJ WEB APLIKACIJE S NAGLASKOM NA KORIŠTENE ALATE 2

2.1 Opis postupka razvoja aplikacije 2

2.2 Mogućnosti i prednosti Microsoft okoline 3

2.3 Programski jezik C# 4

2.4 Neke od prednosti i nedostataka C# 6

2.4.1 Prednosti 6

2.4.2 Nedostaci 7

3. OPIS KORIŠTENIH TEHNOLOGIJA 8

3.1 ASP.NET MVC 8

3.1.1 Modeli 8

3.1.2 Pregled 8

3.1.3 Kontroleri 8

3.2 Entity Framework 9

3.3 CSS 10

4. PROGRAMSKO RJEŠENJE WEB APLIKACIJE 12

4.1 Prikaz programskih komponenti 12

4.2 Primjer koda za pojedine dijelove aplikacije 14

5. UPOTREBA I TESTIRANJE APLIKACIJE 18

5.1. Način upotrebe aplikacije 18

5.2. Testiranje aplikacije 20

6. ZAKLJUČAK 25

LITERATURA 26

SAŽETAK 27

DEVELOPMENT OF A C# WEB APPLICATION 28

ŽIVOTOPIS 29

PRILOZI 30

1. UVOD

U ovom završnom radu zadatak je izrada web aplikacije u programskom jeziku C# koja će služiti za rezerviranje stolova u ugostiteljskim objektima. Završni rad će se baviti konceptualnim i stvarnim rješenjem predviđenog zadatka. Korisniku je potrebno pružiti grafički prikaz stolova – njihov razmještaj i lokaciju. Aplikacija treba raditi s bazom podataka te iz nje dohvatiti podatke o stolovima i uvjetima za rezervaciju, a spremiti u nju odabranu rezervaciju. Zajedno s rezervacijom korisniku je potrebno omogućiti unos kontakt podataka te pružiti informacije o načinu rezerviranja, kako bi korisničko iskustvo bilo jednostavno i učinkovito, na obostrano zadovoljstvo – ugostitelja i gosta.

Također, za funkcionalnost aplikacije bitna je mogućnost biranja datuma za pojedine događaje određenog ugostiteljskog objekta prema pripadajućim datumima. Cilj ove aplikacije je korisnicima olakšati postupak rezerviranja stolova te pružanje više informacija o samom ugostiteljskom objektu, dok ugostiteljima olakšava planiranje prostora, organizaciju klijenata i organizaciju zaposlenika. Navedeni cilj postići će se koristeći aplikacijski okvir ASP.NET MVC, Entity Framework kao Object Relational Mapper te SQL Server kao RDBMS.

U završnom radu nakon uvoda slijedi opis postupka razvoja aplikacije sa naglaskom na korištene alate u kojima su dodatno opisani mogućnosti i prednosti Microsoft okoline, programski jezik C# te njegove prednosti i nedostatci. Nakon toga slijedi detaljan opis korištenih tehnologija u kojem će se dodatno opisati ASP.NET MVC uz njegove tri glavne komponente - modele, preglede i kontrolore, zatim je dodatno opisan Entity Framework i CSS te njihov rad. Nakon toga slijedi programsko rješenje web aplikacije koje se sastoji od prikaza programskih komponenti i i primjera koda za pojedine dijelove aplikacije. Nastavno na programsko rješenje web aplikacije slijedi upotreba i testiranje aplikacije sadržana od načina upotrebe aplikacije i testiranja aplikacije.

1.1 Zadatak završnog rada

U završnom radu potrebno je savladati i opisati metodologiju izrade web aplikacije u C#, kao i ostale potrebne programske tehnologije i pristupe. U prvom redu, u radu će se koristiti aplikacijski okvir ASP.NET MVC, Entity Framework kao Object Relational Mapper te SQL Server kao RDBMS. Korištenjem navedenih tehnologija i pristupa, programski je potrebno ostvariti web aplikaciju i bazu podataka za odgovarajuću primjenu, a cjelovito rješenje prikladno ispitati i nalizirati. (sumentor: Denis Sušac, Mono d.o.o., Osijek).

2. RAZVOJ WEB APLIKACIJE S NAGLASKOM NA KORIŠTENE ALATE

Kako je već u uvodu spomenuto, u izradi završnog rada u Microsoft Visual Studiu 2017 korišteni su aplikacijski okviri ASP.NET MVC, Entity Framework kao Object Relational Mapper te SQL Server kao RDBMS. Uz navedeno korišten je i Google Chrome kako bi se moglo obaviti testiranje aplikacije.

2.1 Opis postupka razvoja aplikacije

Razvoj web aplikacije podrazumijeva sve potrebne postupke i alate za realizaciju traženog programskog rješenja. Navedeno se obično sastoji iz pisanja koda u nekom od razvojnih okruženja, kreiranja potrebne baze podataka, potrebnih modela podataka te testiranja same web aplikacije. U ovome završnom radu, postupak razvoj aplikacije se sastojao iz sljedećeg: ponajprije kreiranje projekta u Microsoft Visual Studiu kao odabranom razvojnom okruženju, stvaranje potrebnog modela podataka u istom korištenjem C# koda, uporaba Entity Frameworka te kreiranje baze podataka na osnovu stvorenog modela podataka, kreiranje kontrolera za definiranje načina rada s podacima i pogleda za prikaz istih te konačno testiranje same aplikacije.

Kreiranje projekta podrazumijeva stvaranje novog programskog rješenja u razvojnom okruženju Microsoft Visual Studio kao ASP.NET MVC web aplikacije. Korišteno razvojno okruženje nudi određen broj već gotovih datoteka istog projekta (kao što su npr. početna stranica, stranica o kontaku, aplikaciji i sl.).

Pri daljnjoj realizaciji aplikacije, korišten je tzv. *Code-first* pristup – ponajprije se C# kodom stvaraju klase odnosno modeli podataka na osnovu kojeg će se uz pomoć *Object Relational Mapper*-a stvoriti baza podataka. Suprotan postupak navedenom bio bi *Database-first* pristup kod koga se ponajprije kreira baza podataka (npr. u SQL Serveru) te se na osnovi te baze podataka uz pomoć *Object Relational Mapper*-a stvaraju potrebni modeli podataka i klase.

Dakle, nakon stvorenih klasa (modela) uz pomoć Entity Frameworka, automatski je kreirana i potrebna lokalna baza podataka. Ista je za potrebe testiranja ispunjena podatcima. Pri samom povezivanju aplikacije i baze podataka, podaci o njihovoj vezi se spremaju u posebnu datoteku projekta za kasnija pozivanja.

Sljedeći korak je kreiranje kontrolera – pisanjem C# koda stvaraju se dijelovi rješenja (datoteke) koje sadrže i opisuju potrebne akcije za upravljanje podatcima (dodavanje unosa, izmjena, brisanje, prikaz detalja i sl.) kao i prikaz istih. Razvojno okruženje, pri tome, automatski generira i potrebne prikaze (poglede) ovisno o odabranom kontroleru. Način rada modela, pogleda i kontrolera, odnosno princip rada MVC aplikacije opisan je kasnije u radu.

Nakon kreiranih svih potrebnih datoteka, aplikacija je realizirana te je posljedni korak u postupku razvoja upravo testiranje same aplikacije. Uz unesene vrijednosti u bazu podataka, testiraju se sve moguće funkcionalnosti koje bi aplikacija trebala obavljati, traže se moguće greške i izuzeci koje je potrebno korigirati i ispraviti. Nakon obavljenog testiranja, aplikacija je uspješno stvorena te spremna za korištenje.

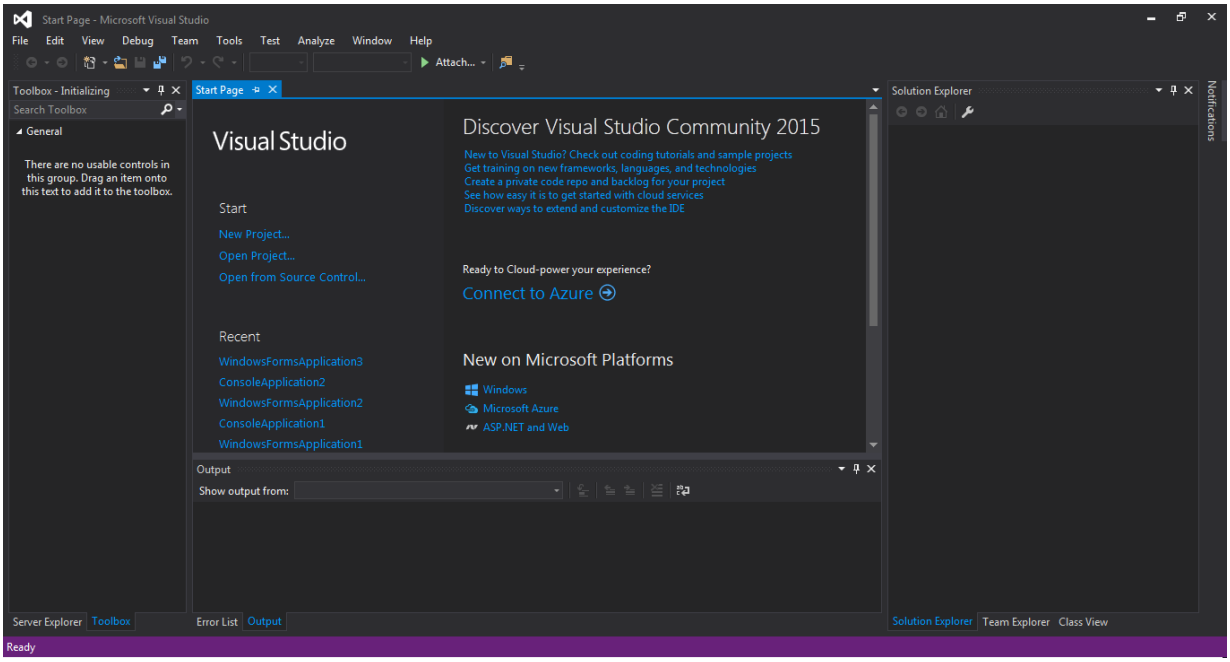
2.2 Mogućnosti i prednosti Microsoft okoline

Može se reći kako Microsoft nudi mnoge alate i tehnologije koji služe korisnicima u izradama aplikacija, ali odabirom Micrsoft tehnologija na početku nemamo potrebno znanje za izradu funkcionirajuće aplikacije. U posljednje vrijeme Microsoft se puno više trudi izdavati vodiče za korištenje njihovih proizvoda i razvijati svoje alate u skladu sa potrebama i sugestijama mnogih programera. Microsoft redovito izdaje vodiče u obliku videa koji detaljno opisuju sve mogućnosti, izmjenu sadržaja te nove mogućnosti u novijim inačicama alata koje izdaju.

Microsoft je razvio Visual Studio kao proizvod idealan za razvojne programere kako bi im omogućio izrađivanje mnogih vrsta aplikacija. Prema [1], Microsoft Visual Studio koristi se za izradu *desktop* aplikacija, web aplikacija ili mobilnih aplikacija. To je ujedno i jedan od najraširanijih alata za izradu aplikacija na Windows operacijskim sustavima. Prednost Visual Studia je što je dosta fleksibilan i omogućuje korisnicima jednostavan razvoj mobilnih aplikacija, ali on zahtjeva puno više vremena za savladavanje svih njegovih mogućnosti. Postoje tri glavne inačice Visual Studo integriranog razvonog okruženja:Visual Studio, Visual Studio Code i Visual Studio Online. Kako je navedeno u [2], također postoji više inačica programa: Visual Studio Community (besplatna inačica dostupna za korištenje za akademske svrhe ili individualni razvoj), Visual Studio Professional (plaćena inačica s raznim korisnim alatima za razvoj namijenjena manjim timovima ili pojedinačnim developerima) te Visual Studio Enterprise (plaćena inačica za poduzeća i timove bilo koje veličine s naprednim alatima za razvoj i najkompleksnijih rješenja). Cilj svake inačice Visual Studia je pružiti bogato razvojno okruženje

svim programerima na globalnoj razini na bilo kojoj platformi. Visual Studio nudi bogat izbor razvojnih jezika. Trenutno programeri mogu razvijati aplikacije u jezicima Visual Basic, C #, PHP, Obijektni-C, JavaScript i Visual C ++. Aplikacijsko sučelje za programiranje na kojem se temelji razvoj tvrtke Microsoft zove se .NET Framework i pruža podršku za interoperabilnost jezika.

Microsoft .NET okvir je programski okvir koji pruža veliku zbirku kodiranih sklopova koje programeri mogu koristiti za lakše razvijanje programa. On nadograđuje operacijski sustav kako bi olakšao rad programerima koji razvijaju aplikacije u dotičnoj tehnologiji.Najvažniji dio te infrastrukture je CLR, to je programski dio u kojem se pisani kod izvršava, on prevodi program u jezik razumljiv kompjuter i na taj način se .Net okvir može prenositi i na druge operacijske sustave osim Microsoftovih, odnosno svaki program koji je napisan u jezicima koje podržava Microsoft. Neki primjeri od tih jezika su C++,C#,VB.Net,Jscript,J#. Pojednostavljeno, .NET je razvojni okvir s kojim imamo mogućnosti novog sučelja za programiranje aplikacija te objedinjuje klasična sučelja operacijskog sustava Winows s mnogim tehnologijama koje je izdao



Microsoft.

Sl 2.2. *Microsoft Visual Studio sučelje*

2.3 Programski jezik C#

Općenito govoreći, C# je objektno orijentirani programski jezik opće namjene, kao što je objašnjeno u [3]. Andres Hejlsberg - zajedno sa svojim timom 1999. godine napravio je

programski jezik C#. Početni naziv za C# bio je „Cool“, što je značilo C-like Object Oriented Language, ali - nakon samo godinu dana naziv se promjenio u C#. C# sintaksa je zapravo dosta slična sintaksi programskog jezika C++. Važno je napomenuti da je C# modernizirana inačica C++. Izvorno je napravljen programski jezik C, koji je imao široku primjenu u programiranju. Nakon njega C++ je postao jezik za izgradnju cijelovitih aplikacija za Windows operacijski sustav. C++ se koristio za pisanje infrastrukture i aplikacija nižih razina, dok su programeri u programskom jeziku Visual Basic razvijali poslovne aplikacije. C# koristi prednosti .NET okvira, što znači da imate pristup brojnim aplikacijskim okvirima, baš kao i Visual Basic razvojni programeri godinama. Dodane su nove vrste podataka, kao što je decimalna vrsta podataka za obavljanje financijskih kalkulacija što dodatno olakšava dio posla. C# je siguran jezik, što znači nekoliko stvari. Na primjer, ne može se upotrebljavati neinicijalizirane varijable. U C++ lako je proglasiti varijablu, a zatim provjeriti njegovu vrijednost; Onda bi se prikazalo ono što je bilo u memorijskoj adresi koja se daje tim varijablama, što bi moglo onemogućiti cijelu aplikaciju za njezino ispravno korištenje. C# - ov compiler će vas obavijestiti ako pokušate upotrijebiti varijablu prije nego što ga inicijalizirate na neku ispravnu vrijednost. Važno je objektna orjentiranost kod C# jezika, za razliku od C++ jezika, daleko više istaknuta. Čak i jednostavne vrste podataka mogu se tretirati kao objekti, što znači da i tip podatka int ima povezane metode. Na primjer, možete koristiti metodu *ToString* da biste dobili vrijednost niza za int, kao što je prikazano u nastavku (Programski kod 2.1):

```
Int brojač = 14;
Console.Write (brojač.ToString ());
```

Programski kod 2.1.

Osim toga, doslovni nizovi mogu se tretirati kao objekti i podržavati razne metode, kao što su Trim, ToUpper, ToLower i mnogi drugi, kao što je prikazano ovdje:

```
Console.Write ("hello, world" .ToUpper ());
```

Programski kod 2.2.

Na C++ se gleda kao na iznimno moćan jezik, ali je problem što se uglavnom ne smatra laganim za naučiti i koristiti. Stvar kod C# je da pokušava pojednostaviti sintaksu da bude dosljedniji i logičniji dok istovremena uklanja kompliciranije značajke C++ jezika. Na primjer, C# uklanja pokazivače. Kao siguran jezik, C# ne dopušta direktnu manipulaciju memorijom,

tako da pokazivači više nisu potrebne u C#. Stoga su neke od najvećih promjena uvedenih u C# jeziku rukovanje memorijom (engl. memory management i prikupljanje smeća (engl. garbage collection).

C# uklanja probleme za upravljanje memorijom od strane programera pomoću sheme za sakupljanje smeća .NET-a. Stavke koje se više ne spominju označene su za sakupljanje smeća, a Okvir može po potrebi vratiti ovu memoriju.C# također podržava uvođenje XML komentara. To nije samo još jedan način za dodavanje komentara u kod, XML komentari se mogu zapravo pretvoriti u vlastitu dokumentaciju. Komentari se postavljaju u XML format i mogu se upotrijebiti prema potrebi za dokumentiranje koda. Ova dokumentacija može uključivati primjer koda, parametre i reference na druge teme. Napokon ima smisla da programer dokumentira svoj kod, jer ti komentari mogu postati dokumentacija neovisno o izvornom kodu.C# više nije namjenjen samo za korisnike Microsofta. Microsoft je izdao C# ECMA-u - i objavio ga je u osnovnoj inačici. Osim toga, daljnji Mono projekti nastoje napraviti otvorenu i prilagodljivu inačicu .NET okvira za druge operacijske sustave, čak i za Linux. Sponzoriran od strane Microsofta, Mono je otvorena i prilagodljiva implementacija Microsoftovog .NET okvira na temelju ECMA standarda za C # i Common Language Runtime.

2.4 Neke od prednosti i nedostataka C#

2.4.1 Prednosti

Tipovi podataka koje koristi C# identični su tipovima korištenim u C i C++ jezicima. U idućoj tablici (Tab. 2.1.) možemo vidjeti neke od najčešće korištenih tipova podataka, njihovu vrstu te veličinu memoriju koju podatak zauzima.

Tab 2.1. *Najčešći tipovi podataka*

Tip podatka	Vrsta podatka	Memorija u bitovima
char	znak	16
int	cijeli broj	32
short	cijeli broj	16
long	cijeli broj	64
float	cijeli broj jednostruke preciznosti	32
double	cijeli broj dvostruke preciznosti	64
bool	logički podatak	8

string	slijed znakova	/
decimal	decimalni broj	128

- C# posuđuje koncepte Jave i C++, usvajajući samo dobre dijelove tih jezika i uklanjajući prekomjerno zbunjujuće i pogrešne značajke koje su glavni izvori grešaka u kodu.

- Zahvaljujući .NET Frameworku omogućena je uporaba više jezika unutar istog riješenja, kao i korištenje biblioteka pisanih u drugim alatima sa podrškom za .NET.

- C# implementira suvremeni kencept objektno orijentiranog programiranja koji omogućuje razvojnom programeru da izrađuje sigurne aplikacije koje pružaju korisniku sljedeću razinu iskustva.

- C# se može koristiti za pisanje širokog raspona aplikacija, od jednostavnih desktop widgeta do razvoja raznih web usluga, programiranje sigurnosnih sustava, pa čak i robota.

- C# se koristi za web programiranje i sa svakodnevnim novim informacijama donosi raspodijeljenu razmjenu informacija, tako da sve što korisnik treba je računalo i preglednik.

2.4.2 Nedostaci

Glavni nedostatak ovog jezika bilo bi vrijeme potrebno za savladavanje i učenje .NET-a kao jezičnog koncepta. Jedna stvar koja otežava učenje su nove značajke koje dolaze svake godine.

3. OPIS KORIŠTENIH TEHNOLOGIJA

3.1 ASP.NET MVC

ASP.NET je zapravo Microsoft Web Framework, odnosno radni okvir korišten za izrađivanje web aplikacija, stanica i usluga (engl. services).

Uzorak Model-View-Controller (MVC), prema [4], odvaja aplikaciju na tri glavne komponente: model, pregled i kontroler (sl. 3.1.). ASP.NET MVC radni okvir pruža alternative s obzirom na ASP.NET Web Forms za izradu web aplikacija. ASP.NET MVC okvir je lagani, provjereno istestirani okvir koji je integriran s postojećim ASP.NET značajkama. Okvir MVC definiran je u sklopu System.Web.Mvc. MVC je standardni uzorak dizajna s kojim su upoznati mnogi razvojni programeri. Na prvu MVC može djelovati jako jednostavno, nakon što pročitate definiciju i pogledate par crteža, ali sama primjena je ipak kompliciranija.

Okvir MVC uključuje komponente koje slijede u nastavku teksta:

3.1.1 Modeli

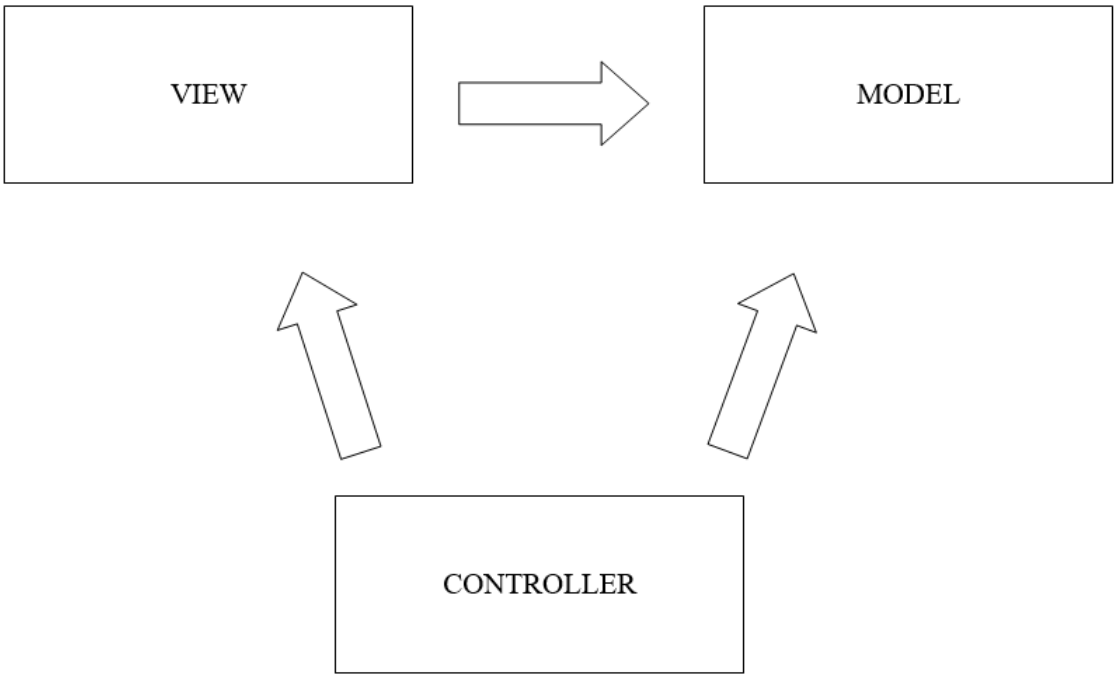
Modeli su objekti kao dijelovi aplikacije koji implementiraju logiku domene podataka aplikacije. Često objekti modela preuzimaju i pohranjuju stanje modela u bazi podataka. Na primjer, objekt proizvoda može preuzeti podatke iz baze podataka, raditi na njemu, a zatim ažurirane podatke vratiti u tablicu Proizvodi u bazi podataka sustava SQL Server.

3.1.2 Pregled

Pregled je zapravo komponenta koje prikazuju korisničko sučelje (engl. user interface – UI) aplikacije. Obično se ovo korisničko sučelje stvara iz podataka o modelu. Primjer bi bio prikaz uređivanja tablice proizvoda koji prikazuje tekstualne okvire, padajuće popise i potvrdne okvire (check boxes) na temelju trenutačnog stanja objekta proizvoda.

3.1.3 Kontroleri

Kontroleri su komponente kojima je zadatak sjediniti pregled zajedno sa modelom. U MVC aplikaciji, pregled samo prikazuje informacije. Kontrolor obrađuje i odgovara korisničkom unosu i interakciji. Na primjer, kontroler upravlja vrijednostima upita i prosljeđuje te vrijednosti na model, koji zauzvrat može koristiti ove vrijednosti za daljnje upite povezane s bazom podataka.



Sl. 3.1. *Princip rada MVC aplikacije*

3.2 Entity Framework

Entity Framework je okvir objektno-relacijskog mapiranja (ORM) otvorenog koda za ADO.NET. Bio je dio .NET okvira, ali od Entity Framework 6 inačice odvojen je od .NET okvira, kao što je opisano u [5].

Entity Framework je skup tehnologija u ADO.NET-u koji podržavaju razvoj programskih aplikacija orijentiranih na podatke. Arhitekti i programeri aplikacija orijentiranih na podatke obično su se borili s potrebom da se postignu dva različita cilja. Oni moraju modelirati entitete, odnose i logiku poslovnih problema koje rješavaju, dok također moraju raditi sa bazama podataka koje se koriste za pohranu i dohvaćanje podataka. Entity Framework omogućuje razvojnim programerima rad s podacima u obliku objekata i svojstava specifičnih za određenu domenu, kao što su korisnici i adrese kupaca, bez potrebe da se brinu o tablicama i stupcima baze podataka u kojima se ti podaci pohranjuju. Budući da je Entity Framework sastavni dio .NET okvira, Entity Framework aplikacije mogu se izvoditi na bilo kojem računalu na kojem je instaliran .NET Framework (počevši od inačice 3.5 SP1).

3.3 CSS

Tehnologija kaskadnih listi oblikovanja (eng. *Cascading Style Sheets,* CSS), kako je navedeno u [6], stilski je jezik korišten za opis prezentacije dokumenta napisanog pomoću markup (HTML) jezika. Danas se uvelike koristi pri dizajnu web stranica i aplikacija – CSS uređuje raspored i izgled stranice odnosno definira na koji način se prikazuju HTML elementi (sadržaj).

Sintaksa CSS koda sastoji se - prvenstveno iz selektora i deklaracijskog bloka. Selektor određuje dio markupa na koji se zadani stil (oblikovanje) primjenjuje. Tako npr. selektor mogu činiti elementi specificiranog *id* (jedinstven element) ili *class* (više od jednog elementa) svojstva ili npr. svi elementi istog tipa (npr. svi *h1* elementi – elementi te razine zaglavlja). Deklaracijski blok podrazumijeva sve deklaracije unutar vitičastih zagrada kojima se definira oblikovanje elemenata definiranih selektorom. Deklaracija se vrši na način "*svojstvo : vrijednost*". Svake dvije deklaracije istog deklaracijskog bloka međusobno su odvojene s ";".

U HTML kod CSS se može ugraditi ugrubo na tri načina: unutar samih HTML oznaka (*tags*), unutar HTML zaglavlja ili kao posebna datoteka. Unutar samih HTML oznaka, CSS se uključuje uz pomoć ključne riječi *style* čime se definira oblikovanje za taj element. Primjer:

```
<h style="color: red; margin-top: 30px">Tekst</h>
```

Programski kod 3.1.

Unutar HTML zaglavlja, CSS definiramo uz pomoć <style> oznaka unutar koji se upisuje CSS kod. Primjer:

```
<style type="text/css">
h1 { color: blue;
margin-bottom:10px;
}
</style>
```

Programski kod 3.2.

Kao posebna datoteka, cijeli CSS kod se piše odvojeno u zasebnu datoteku (*.css* ekstenzije) koja se zatim uvozi u potrebni HTML kod. Primjer:

```
<link rel="stylesheet" href="naziv.css" type="text/css">
```

Programski kod 3.3.

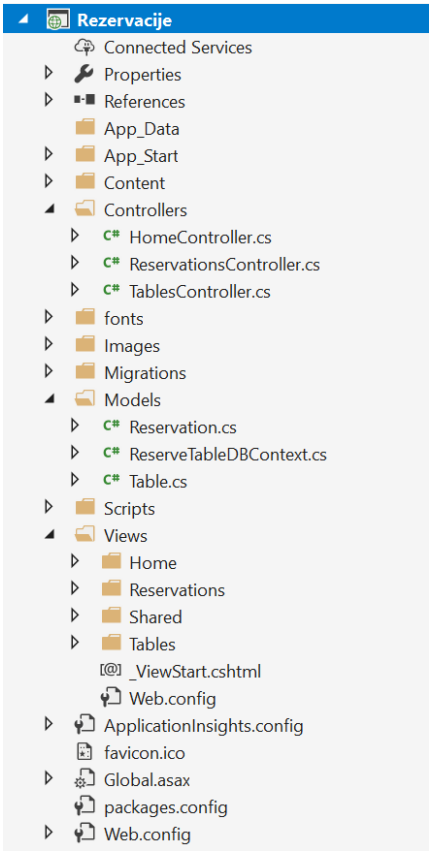
U izradi ove aplikacije korišten je ponajviše CSS unutar samih HTML oznaka za pojedine elemente, ali i prethodno definirani CSS kod iz dostupne *bootstrap.css* datoteke.

4. PROGRAMSKO RJEŠENJE WEB APLIKACIJE

Web aplikacija za rezerviranje stolova realizirana je kao programsko rješenje – ASP.NET aplikacija koja se sastoji iz više programskih komponenti i funkcionalnih dijelova aplikacije. Pri samoj izradi nisu korištene neke od suvremenih naprednih tehnologija (*layering, dependency injection, unit testing*), već se težilo jednom jednostavnijem pristupu kako bi se upoznale i prikazale osnovne mogućnosti ASP.NET MVC-a, princip rada i način funkcioniranja.

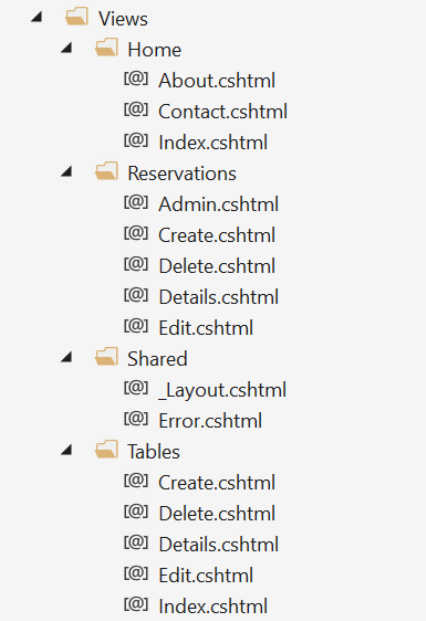
4.1 Prikaz programskih komponenti

Sama aplikacija (projekt) sastoji se od više datoteka (Sl. 4.1.). Među njima bitno je izdvojiti: mapu *App_Data* koja sadrži korištenu bazu podataka (podatke o stolovima i rezervacije), mapu *Controllers* koja sadrži korištene kontrolere (*HomeController, TablesController te ReservationsController*), mapu *Models* koja sadrži korištene modele podataka odnosno klase (*Table, Reservation* te *ReserveTableDBContext* kao model baze podataka), mapu *Views* koja sadrži korištene poglede organizirane kao pogledi početne stranice (automatski generirani od strane razvojnog okruženja, *Home*), dijeljeni pogledi (za *Layout* koji je zajednički većini pogleda), pogledi vezani za rezervacije (*Reservations*) te za stolove (*Tables*). Uz to bitno je spomenuti datoteku *Web.confing* u kojoj se spremaju između ostalog i informacije o *Entity Frameworku* te samoj vezi s bazom podataka.



Sl. 4.1. *Komponente web aplikacije za rezervacije stolova*

Mape kao što su *Content* i *Scripts* sadrže potrebne dodatne datoteke koje definiraju izgled i ponašanje određenih elemenata odnosno dijelova aplikacije. Tako se u *Content* nalaze biblioteke za CSS dizajn HTML elemenata u web aplikaciji (između ostalog *bootstrap* biblioteka) a u *Scripts* gotove *JavaScript* funkcije koje se mogu pozivati. Kao što se može vidjeti na slici 4.2, pogledi su grupirani u više mapa ovisno o modelu odnosno funkciji.



Sl. 4.2. *Datoteke pogleda*

Na osnovu *HomeControllera* stvoreni su pogledi u ***Home*** mapi (*About* – za detalje o aplikaciji, *Contact* – za prikaz informacija o kontaktu, *Indeks* – početna stranica). Na temelju *TablesControllera* stvoreni su ***Tables*** pogledi kojima pristupa samo ugostitelj budući da se odnose na rad sa stolovima (*Create* – unos novog stola u bazu, *Delete* – brisanje postojećeg stola iz baze, *Details* – prikaz detalja o pojedinom stolu, *Edit* – uređivanje podataka o pojedinom stolu te *Index* – početna stranica o stolovima, tj. prikaz baze stolova). *ReservationsController* generirao je poglede ***Reservations*** potrebne za rezerviranje stola – odabira stola, izmjenu ili potvrdu rezervacije (*Create* – unos nove rezervacije i odabira željenog stola, *Delete* – brisanje rezervacije, dostupno vlasniku aplikacije, tj. ugostitelju, *Details* – prikaz detalja o rezervaciji, cijenama, uvjetima, stolu i sl., *Edit* – izmjene podataka o rezervaciji te *Admin* – prikaz svih rezervacija dostupan vlasniku, tj. ugostitelju). Svi navedeni pogledi koriste zajednički *Layout* iz *Shared* grupe pogleda kojim je definiran osnovni izgled pogleda (pozadina, izbornik i sl.), a pisani su u tzv. *Razor* jeziku (kombinacija C# i HTML jezika).

4.2 Primjer koda za pojedine dijelove aplikacije

Kao što je prethodno spomenuto (potpoglavlje 2.1), prvi korak u realizaciji programskog rješenja je pisanje potrebnih modela podataka (*Code-first* pristup). Najprije su stvorene klase *Reservation.cs* i *Table.cs* koje predstavljaju model podataka za rezervacije odnosno stolove. Sadržaji klasa prikazani su slikama 4.3. i 4.4., a pisane su u programskom jeziku – C#.

```
public class Table
{
    3 references | 0 exceptions
    public int ID { get; set; }
    7 references | 0 exceptions
    public int TableNo { get; set; }
    1 reference | 0 exceptions
    public string Type { get; set; }
    1 reference | 0 exceptions
    public string TableClass { get; set; }
    1 reference | 0 exceptions
    public string Requirement { get; set; }
    2 references | 0 exceptions
    public int Capacity { get; set; }
    10 references | 0 exceptions
    public bool Available { get; set; }
    1 reference | 0 exceptions
    public virtual Reservation Reservation { get; set; }
}
```

Sl. 4.3. *Model podataka za stolove*

```
public class Reservation
{
    [Key, ForeignKey("Table")]
    12 references | 0 exceptions
    public int ID { get; set; }
    7 references | 0 exceptions
    public virtual Table Table { get; set; }
    1 reference | 0 exceptions
    public string Ime { get; set; }
    1 reference | 0 exceptions
    public string Prezime { get; set; }
    1 reference | 0 exceptions
    public string BrojMob { get; set; }
    1 reference | 0 exceptions
    public string Email { get; set; }
    [DataType(DataType.Date)]
    [DisplayFormat(DataFormatString = "{0:dd.MM.yyyy}", ApplyFormatInEditMode = true)]
    6 references | 0 exceptions
    public DateTime Datum { get; set; }

    2 references | 0 exceptions
    public int BrojOsoba { get; set; }

}
```

Sl. 4.4. *Model podataka za rezervacije*

U klasi *Table* primjećuju se podatci za stol koji predstavljaju stupce u tablici baze podataka (broj stola, vrsta, razred, uvjet za rezervaciju, kapacitet, dostupnost) te strani ključ *Reservation* koji upućuje na rezervaciju koja pripada tom stolu. Analogno, i u klasi *Reservation* postoje potrebni podatci o rezervaciji (ime, prezime, broj mobitela, e-mail, broj osoba, datum za rezervirati) te strani ključ *Table* koji predstavlja broj stola na koji se rezervacija odnosi te ujedno i primarni ključ cijele tablice rezervacija.

Na osnovu modela podataka te unesenih podataka o vezi (*ConnectionStrings*) u *Web.config* datoteci, stvaraju se spominjani kontroleri *TablesController* i *ReservationsController* koji sadrže akcije za svaki pojedini pogled kojem pripadaju. Tako je npr. na slici 4.5 prikazan dio

kontrolera za rezervacije namjenjen stvaranju odnosno unosu nove rezervacije, a slika 4.6 prikazuje dio koda za odgovarajući pogled u kojem se isto omogućuje (prikazan je dio za odabir stola i unos imena).

```
// GET: Reservations/Create
0 references | 0 requests | 0 exceptions
public ActionResult Create()
{

    var Dostupni = (from t in db.Tables where t.Available == true select t);
    ViewBag.ID = new SelectList(Dostupni, "ID", "TableNo");
    return View();
}

// POST: Reservations/Create
// To protect from overposting attacks, please enable the specific properties you want to bind to, for
// more details see https://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
0 references | 0 requests | 0 exceptions
public ActionResult Create([Bind(Include = "ID,Ime,Prezime,BrojMob,Email,Datum,BrojOsoba")] Reservation reservation)
{
    if (ModelState.IsValid)
    {
        db.Reservations.Add(reservation);
        var query = (from t in db.Tables where t.TableNo == reservation.ID select t);
        foreach (Table t in query) t.Available = false;
        db.SaveChanges();
        return RedirectToAction("Details/" + reservation.ID.ToString());
    }

    var Dostupni = (from t in db.Tables where t.Available == true select t);
    ViewBag.ID = new SelectList(Dostupni, "ID", "TableNo", reservation.ID);

    return View(reservation);

}
```

Sl. 4.4. *Dio koda za unos nove rezervacije (kontroler)*

```
@model Rezervacije.Models.Reservation
@{
    ViewBag.Title = "Rezervacija stola";
}
<h2>Rezervacija stola</h2>
<p>1. Pogledaj stolove &raquo; 2. Odaberi jedan od slobodnih stolova &raquo; 3. Ostavi svoje podatke</p>

@using (Html.BeginForm())
{  @Html.AntiForgeryToken()
    <div class="col">
    <hr />
        <div class="row">
            <div class="col-md-6">
                <div class="form-horizontal">
                    @Html.ValidationSummary(true, "", new { @class = "text-danger" })
                    <div class="form-group">
                        @Html.LabelFor(model => model.ID, "Broj stola", htmlAttributes: new { @class = "control-label col-md-2" })
                        <div class="col-md-10">
                            @Html.DropDownList("ID", null, "--Odaberi--", htmlAttributes: new { @class = "form-control" })
                            @Html.ValidationMessageFor(model => model.ID, "", new { @class = "text-danger" })
                        </div>
                    </div>
                    <div class="form-group">
                        @Html.LabelFor(model => model.Ime, "Ime", htmlAttributes: new { @class = "control-label col-md-2" })
                        <div class="col-md-10">
                            @Html.EditorFor(model => model.Ime, new { htmlAttributes = new { @id = "ime", @class = "form-control" } })
                            @Html.ValidationMessageFor(model => model.Ime, "", new { @class = "text-danger" })
                        </div>
                    </div>
```

Sl. 4.5. *Dio koda za unos nove rezervacije (pogled)*

Kao što se može vidjeti, pri samom učitavanju stranice (pogleda) korisniku se nudi da na osnovu slike (grafičkog prikaza stolova) odabere željeni stol iz ponuđene liste (*DropDownList)* slobodnih (dostupnih) stolova. Po unosu ostalih parametara odnosno podataka o rezervaciji, klikom na gumb za prikaz rezervacije, podaci se šalju kontroleru koji dohvaća podatke iz baze te ih sprema u istu (dodaje novu rezervaciju stola).

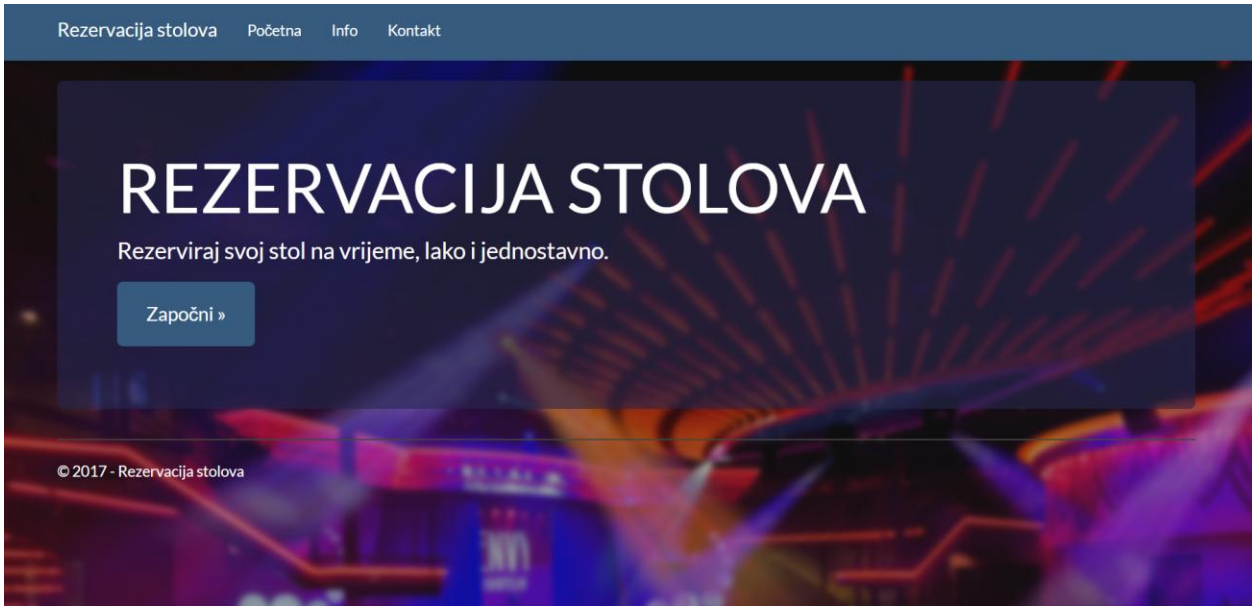
U kontroleru se (akcija *Create*) ponajprije upitom iz baze podataka dohvaćaju stolovi koji nisu rezervirani (*available* svojstvo je postavljeno u *true*). Korisnikovog odabira stola sprema se u obliku varijable *reservation.ID* u *ViewBag*-u. Na klik gumba, prije no što se za odabrani stol unese rezervaciju u bazu rezervacija, svojstvo *available* odabranog stola se postavlja u *false* čime se osigurava da je rezervirani stol sada zauzet. Istovremeno, pogled sadrži potrebne HTML elemente koji omogućuju unos parametara rezervacije – spomenuti *DropDownList* za odabir stola, *TextBox*-ove odnosno polja (*Html.EditorFor*) za unos ostalih parametara (ime, prezime, broj mobitela i sl.) te pripadne *Label*-e za opis pojedinih parametara koje je potrebno unijeti. Na početku samog pogleda naglašeno je koji će model podataka koristiti upotrebom "*@model Rezervacije.Models.Reservation*" dijela koda.

Na sličnom principu funkcioniraju i odnosi kontrolera (akcija kontrolera) i odgovarajućih pogleda za prikaz detalja (*Details*), brisanje (*Delete*) ili izmjenu (*Edit*) rezervacije.

5. UPOTREBA I TESTIRANJE APLIKACIJE

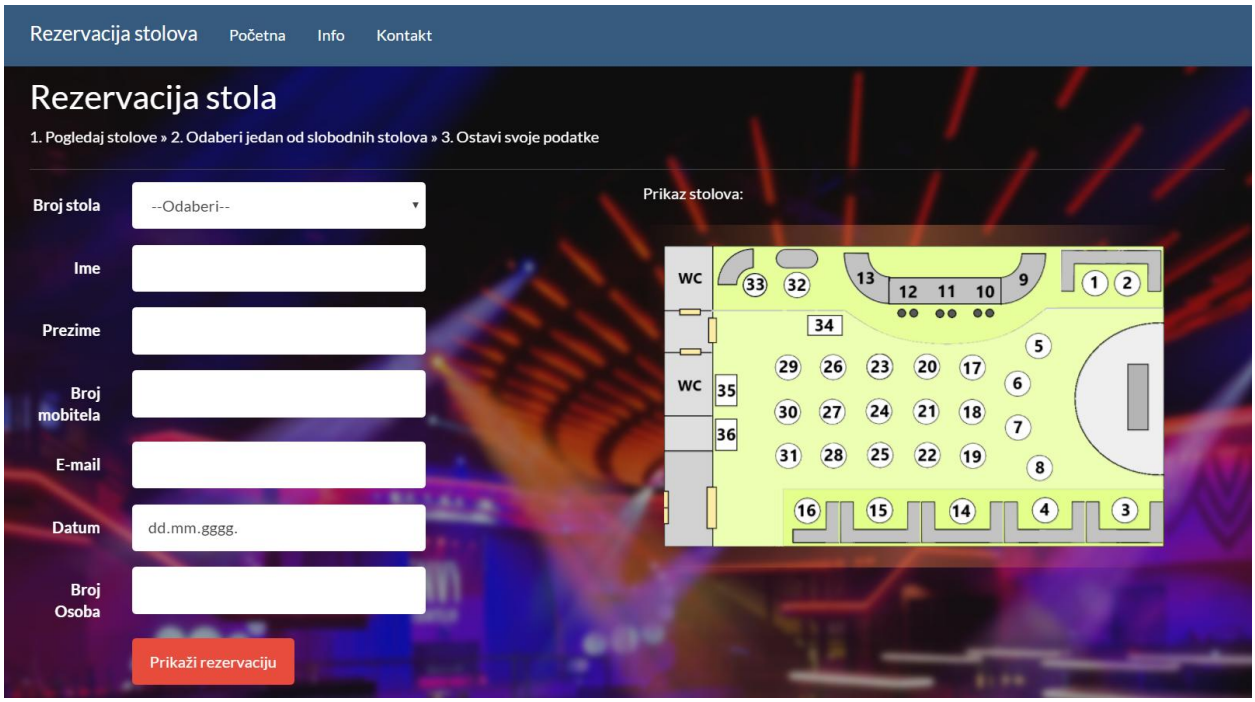
5.1. Način upotrebe aplikacije

Aplikacija za rezervacije stolova pri pokretanju automatski otvara početnu stranicu (Sl. 5.1). Korisniku je u svakom trenutku na raspolaganju izbornička traka gdje može otići na rezerviranje stolova, vratiti se na početnu stranicu, otići na stranicu o kontaktu ili o samoj aplikaciji.



Sl. 5.1. *Početna stranica*

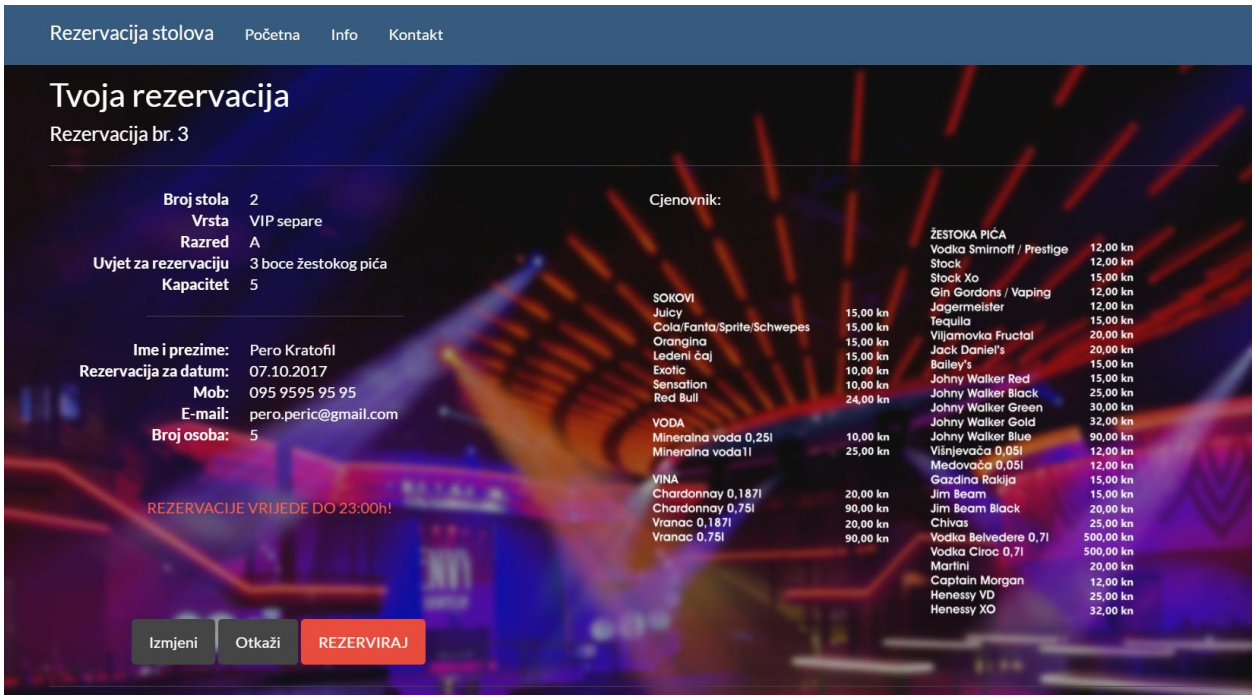
Klikom na gumb „Započni“ korisnik odlazi na stranicu za rezerviranje stola (pogled *Create*). Tu mu se prikazuju potrebna polja za unos nove rezervacije stola te slika rasporeda stolova (Sl. 5.2). Potrebno je kliknuti na padajući izbornik stolova, pri čemu će aplikacija prikazati slobodne stolove koje korisnik može izabrati. Zatim korisnik unosi ime, prezime, broj mobitela, e-mail, željeni datum i broj osoba za stolom. U slučaju da neki od podataka nedostaje, korisnik će biti obavješten popratnom porukom, a klikom na gumb „Prikaži rezervaciju“ korisnik odlazi na sljedeću stranicu.



Sl. 5.2. *Unos nove rezervacije*

Sljedeća stranica (Sl. 5.3.) predstavlja prikaz detalja o rezervaciji, ali i samom rezerviranom stolu (uvjet rezervacije i slično). U slučaju da korisniku neodgovara pojedini dio rezervacije, može ga izmjeniti klikom na gumb „Izmjeni“ pri čemu odlazi na stranicu za izmjenjivanje, gdje može promjeniti bilo koji od početnih parametara rezervacije. Ako želi otkazati rezervacije, to može učiniti klikom na gumb „Otkaži“ pri čemu odlazi na stranicu za rezerviranje stola, gdje sada iznova može rezervirati stol ili napustiti aplikaciju. Ako se slaže s rezervacijom i uvjetima, klikom na „Rezerviraj“ korisnik potvrđuje rezervaciju i uspješno je rezervirao svoj stol.

Navedeno predstavlja dio aplikacije namijenjen korisnikovoj strani. U aplikaciji je omogućeno da vlasnik vrši sve operacije sa stolovima, kao i uvid u bazu stolova – dodavanjem linka u adresu: /*Tables* (npr. */Tables/Create*, ili */Tables/Details/3*) te da ima uvid u bazu rezervacija (*/Reservations/Admin*), gdje iste može uređivati ili brisati.



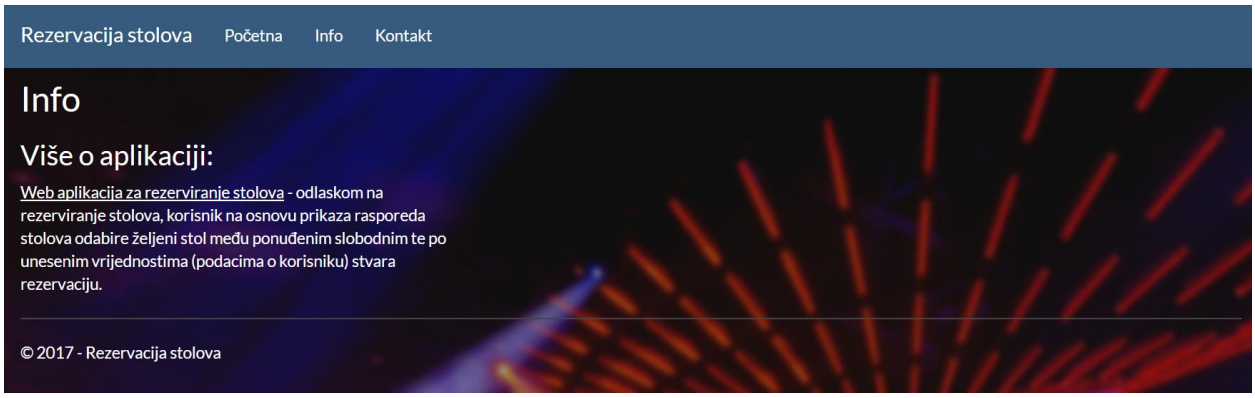
Sl. 5.3. *Prikaz detalja rezervacije*

5.2. Testiranje aplikacije

Testiranje aplikacije podrazumijeva testno korištenje aplikacije u svrhu ispitivanja osnovnih funkcionalnosti koje ona mora zadovoljavati. U ovom slučaju to su pristup stranicama (stvaranje rezervacije, kontakt, početna, o aplikaciji), rezerviranje stola, uređivanje rezervacije, otkazivanje rezervacije te potvrda rezervacije, s korisnikove strance. S vlasnikove strane, testira se i prikaz baze stolova i baze rezervacija.

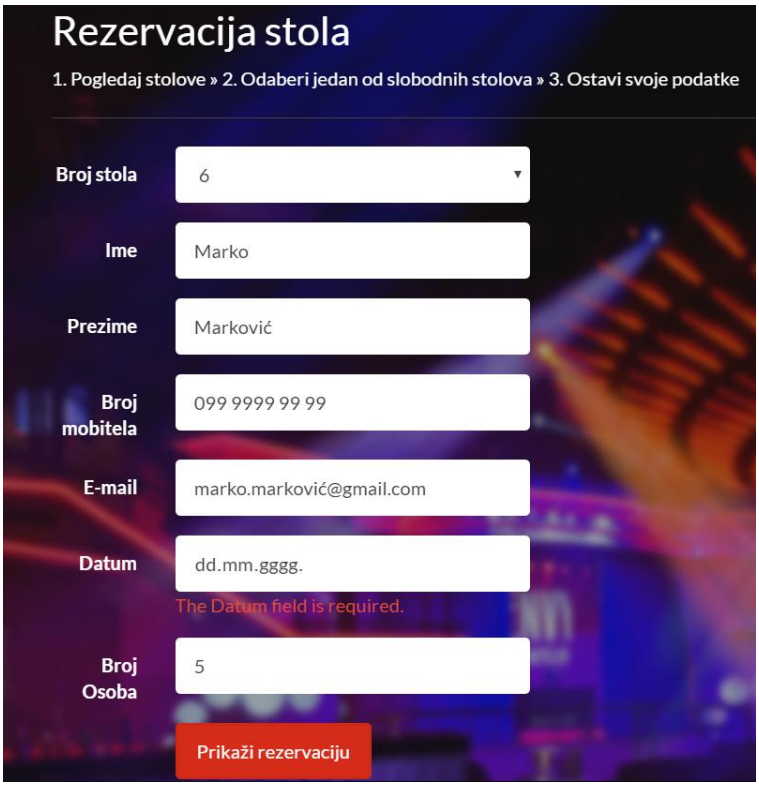
Klikom na svaki od gumba u glavnoj izborničkoj traci, korisnik može pristupiti spomenutim stranicama (rezerviranje, početna, o aplikaciji te kontakt). Primjer stranice o aplikaciji dan je na slici 5.4.

Klikom na gumb „Započni“ početne stranice ili „Rezervacija stola“ izborničke trake, korisnik uspješno pristupa stranici za rezervaciju stola. Unose se probne vrijednosti, pri čemu se da primjetiti da se već zauzeti stolovi ne prikazuju u listi mogućih odabira slobodnih stolova padajućeg izbornika.



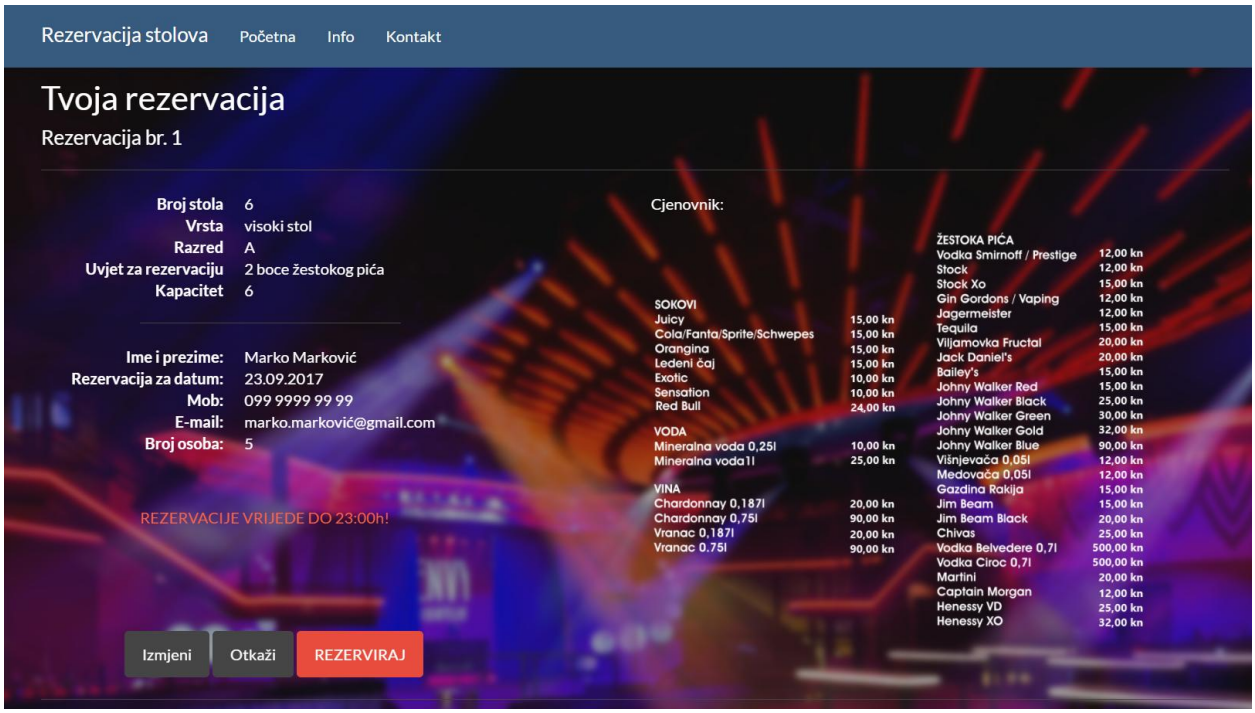
Sl. 5.4. *Pristup stranici o aplikaciji preko izborničke trake*

U slučaju da neki od parametara nije unesen ispisuje se napomena o tome prilikom klika na gumb „Prikaži rezervaciju“ (Sl. 5.5).

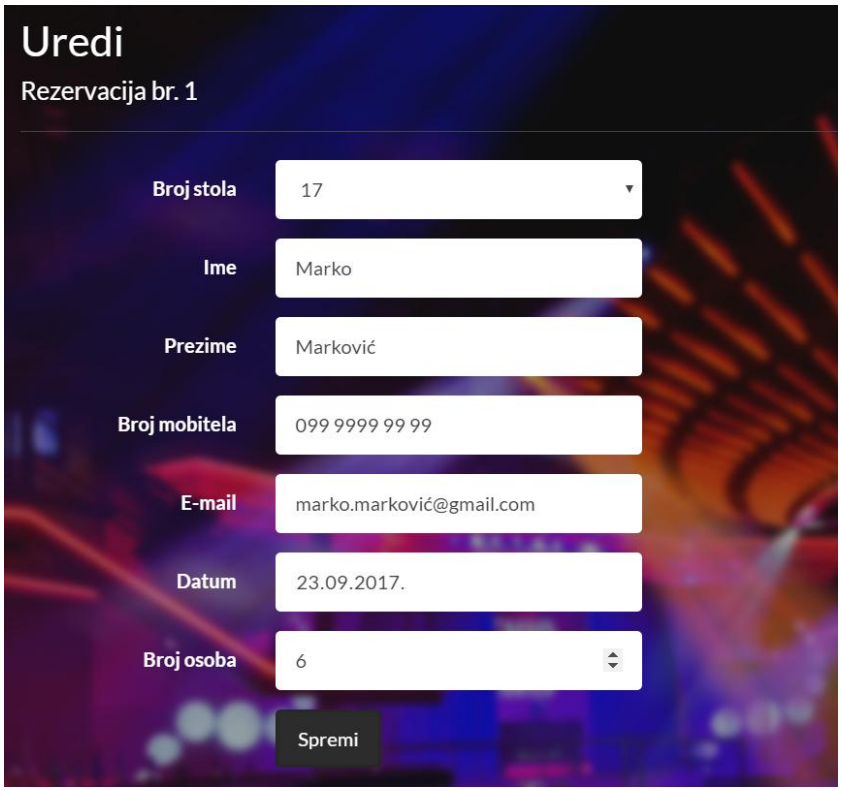


Sl. 5.5. *Prikaz napomene o neispunjenom polju*

Ako su uneseni svi potrebni parametri klikom na gumb dolazi se uspješno do detalja o rezervaciji (Sl. 5.6). Prikazani su podaci o zauzetom stolu, uvjet rezervacije, kapacitet, vrsta i razred stola, kao i do kada vrijede rezervacije. S desne strane dan je cjenik kako bi korisnik okvirno znao cijenu za rezervaciju, na osnovu uvjeta. Pri dnu su ponuđene opcije za izmjenu, otkazivanje ili potvrdu rezervacije. Testirat će se svaka o mogućih opcija.



Sl. 5.6. *Prikaz detalja o unesenoj rezervaciji stola*



Sl. 5.7. *Prikaz za izmjenu podataka o rezervaciji*

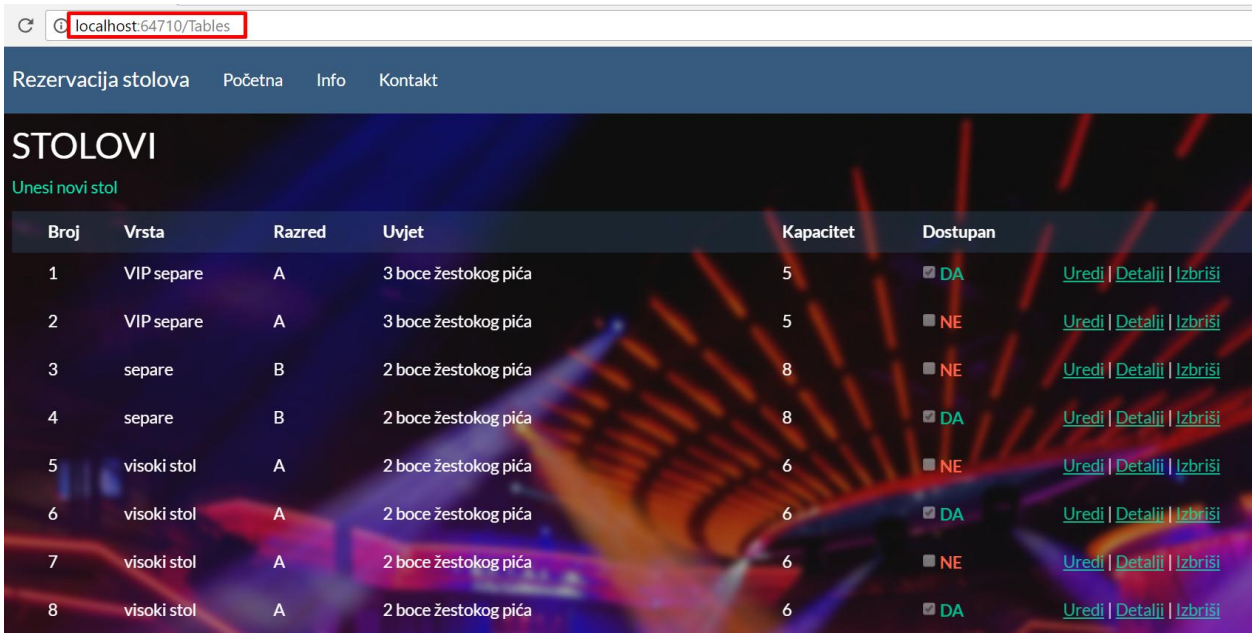
Odlaskom na izmjenu, za potrebe testiranja mijenja se broj stola u broj 17 te broj osoba u 6 osoba. Spremanjem izmjena uočava se da su se detalji rezervacije adekvatno promjenili (Sl. 5.8).



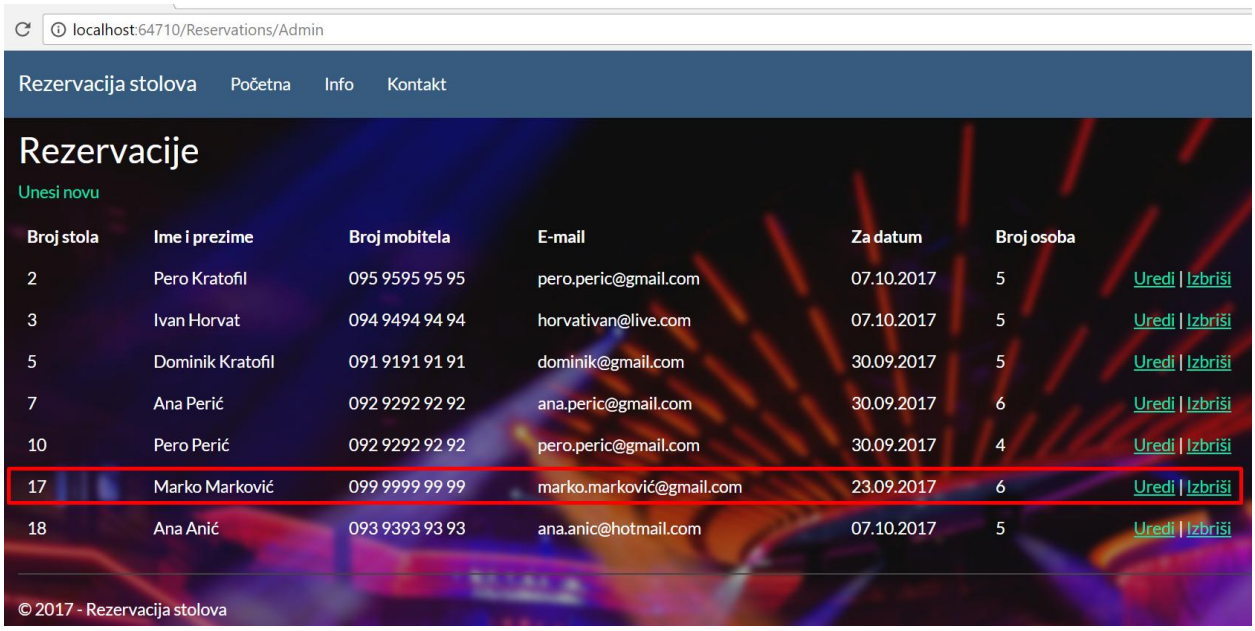
Sl. 5.8. *Izmjenjena rezervacija*

Otkazivanjem rezervacije (klikom na gumb) vraća se na stranicu za kreiranje rezervacije (ispočetka), a postojeća rezervacija se briše i ne ostaje u bazi. Potvrđivanjem rezervacije, rezervacija se sprema i korisnik se vraća na početnu stranicu aplikacije. Testna rezervacija je spremljena i potvrđena te se sada nalazi u bazi rezervacija.

Glede vlasnikove strance, prethodno spomenutim načinima (5.1), moguće je pristupiti bazi stolova (Sl. 5.9) i bazi rezervacija (5.10), gdje se može, između ostalog uočiti i skoro napravljena rezervacija stola broj 17.



Sl. 5.9. *Baza testova i način pristupa (adresa)*



Sl. 5.10. *Prikaz rezervacija i dodana testna rezervacija*

Time se uspješno potvrđuju osnovne funkcionalnosti aplikacije koja odgovara zahtjevima.

6. ZAKLJUČAK

Cilj ovog zavrsnog rada bio je izrada web aplikacije u programskom jeziku C# koja služi za rezerviranje stolova u ugostiteljskim objektima. Bilo je potrebno korisniku pružiti grafički prikaz stolova, kako slobodnih tako i rezerviranih, te njihov razmještaj i lokaciju. Jedan od uvjeta koji je ispunjen je također da aplikacija bude učinkovita, brza i jednostavna za korištenje.

Aplikacija je realizirana u programskom okruženju Microsoft Visual Studia uz korištenje znanja i tehnologija koja zahtjevaju aplikacijski okvir ASP.NET MVC te Entity Framework kao Object Relational Mapper, HTML-a i sličnih potrebnih tehnologija i programskih jezika. Korištenjem navedenih tehnologija i pristupa programski je ostvarena web aplikacija i baza podataka za odgovarajuću primjenu.

Neke od nadogradnji vezanih uz aplikaciju mogle bi biti različito rješavanje problema zauzetih stolova, aplikacija bi mogla imati napredniji grafički pristup uz mogućnost da korisnik odabire stol direktno sa slike.

LITERATURA

[1] Microsoft Visual Studio: Wikipedia, 2017., dostupno na: https://en.wikipedia.org/wiki/Microsoft_Visual_Studio [13. rujna 2017.]

[2] Visual C# Language: Microsoft, 2017., dostupno na: https://msdn.microsoft.com/en-us/library/aa287558(v=vs.71).aspx

[3] Visual C# Language: Microsoft, 2017., dostupno na: https://msdn.microsoft.com/en-us/library/aa287558(v=vs.71).aspx [13. rujna 2017.]

[4] ASP.NET MVC Overview: Microsoft, 2017., dostupno na: https://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx [19. rujna 2017.]

[5] Entity Framework: Wikipedia, 2017., dostupno na: https://en.wikipedia.org/wiki/Entity_Framework [19. rujna 2017.]

[6] CSS: Wikipedia, 2017., dostupno na: https://hr.wikipedia.org/wiki/CSS [23. rujna 2017.]

SAŽETAK

Zadatak ovog završnog rada bio je izrada web aplikacije u programskom jeziku C# koja će služiti za rezerviranje stolova u ugostiteljskim objektima. Bilo je bitno pružiti olakšano rješenje za komplikacije s kojima se svakodnevni ugostitelji te korisnici ugostiteljskih objekata susreću. Aplikacija omogućuuje jednostavan uvid u rezervirane stolove i podatke koje korisnik rezerviranog stola unosi, također moguća je i potvrda rezervacije putem unesenog broja mobitela što rješava problem ostavljanja praznih stolova. U teorijskom dijelu rada sadržan je opis korištenih tehnologija sa naglaskom na korištene alate, opis postupka razvoja aplikacije, mogućnosti i prednosti Microsoft okoline te model aplikacije, zatim je opisano programsko rješenje web aplikacije koje se sastoji od primjera koda za pojedine dijelove zaključno sa uputama o korištenju aplikacije. U realizaciji aplikacije neke od tehnologija koje je potrebno spomenuti su aplikacijski okvir ASP.NET MVC, C# te HTML.

Ključne riječi: ASP.NET, C#, Microsoft Visual Studio, rezervacija, Web aplikacija.

DEVELOPMENT OF A C# WEB APPLICATION

The task of this final work was to create a web application in the C# programming language that will serve as a table reservation in catering facilities. It was important to provide an easy solution to the complications with which everyday caterers and users of catering facilities meet. The application allows for easy access to the reserved tables and information provided by the user of the booked table, and it is also possible to confirm the reservation by entering the number of mobile phones, which solves the problem of leaving desks empty. The theoretical part of this paper contains a description of the technology used, focusing on the tools used, the description of the application development process, the capabilities and advantages of the Microsoft environment and the application model, and then describing the web application software solution that consists of code examples for particular parts in application. In implementing the application some of the technologies that need to be mentioned are the ASP.NET MVC application framework, C # and HTML.

Keywords: ASP.NET, C#, Microsoft Visual Studio, reservation, Web application.

ŽIVOTOPIS

Dominik Kratofil rođen je 17.11.1995. godine u Osijeku, Hrvatska. Godine 2002. započinje osnovnoškolsko obrazovanje u OŠ Ivana Gorana Kovačića u Đakovu. Nakon završene osnovne škole 2010. godine upisuje Srednju strukovnu školu Braće Radić u Đakovu, računalni tehničar za strojarstvo. Nakon završene srednje škole 2014. godine upisuje Preddiplomski sveučilištni studij računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku, koji trenutno pohađa. Sebe opisuje kao mladu i ambicioznu osobu, komunikativnu, te snalažljivu u ostvarivanju svojih ciljeva. Od vještina posebno se ističu dizajniranje i crtanje pomoću CAD alata koje su stečene srednjoškolskim obrazovanjem.

_____

Potpis studenta

PRILOZI

CD:

- Elektronička inačica rada (.docx i .pdf format)
- Projekt web aplikacije u Visual Studiu