Izrada demonstracijskog modela sustava za online rezervaciju dvorana

Mustić, Filip

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI

Filip Mustić

IZRADA DEMONSTRACIJSKOG MODELA SUSTAVA ZA ONLINE REZERVACIJU DVORANA

ZAVRŠNI RAD

Zagreb, kolovoz 2020

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI

ZAVRŠNI RAD

IZRADA DEMONSTRACIJSKOG MODELA SUSTAVA ZA ONLINE REZERVACIJU DVORANA

DEVELOPMENT OF A DEMONSTRATION SYSTEM FOR CLASSROOM RESERVATIONS

Mentor: Tomislav Erdelić, mag. ing. el. techn. inf.

Student: Filip Mustić

JMBAG: 0135241268

Zagreb, kolovoz 2020

Sažetak

Zadatak završnog rada je izraditi demonstracijski sustav za rezervaciju dvorana koji koristi relacijsku bazu podataka i web aplikaciju napravljenu u Visual Studio programskom alatu. Program integrira više programskih jezika koji su potrebni za programiranje ovakvog sustava. Za izradu web aplikacije, kao osnova sustava, potrebno je izraditi relacijsku bazu podataka za pohranu podataka o dvoranama, zaposlenicima i vremenskim zauzećima. Potrebno je i izraditi web aplikaciju s korisničkim sučeljem da se pojednostavi rad korisnika sa sustavom za rezervaciju. Sučelje je direktno povezano s relacijskom bazom podataka koja koristi SQL upite za dohvat željenih podataka. Za izradu web aplikacije korišten je većinom C# programski jezik, ali su potrebni i programski jezici HTML, CSS i JavaScript za izradu pregleda koji se prikazuju korisniku u pregledniku. Relacijska baza podataka sprema sve informacije koje će sustav koristiti za potrebe pregleda rezervacija, te se brine o pravilnom unosu informacija u sustav. Web aplikacija prikazuje informacije iz baze podataka, tako da korisnik može imati jednostavan uvid u informacije koje su potrebne za rezervaciju, te same rezervacije.

KLJUČNE RIJEČI: MVC, C#, SQL, baza podataka, web aplikacija, rezervacije

Summary

The aim of this paper is to create a demonstration system for classroom reservations that uses relation database and web application made in Visual Studio programming tool. Implemented program integrates multiple programming languages that are required for creating this type of a system. As a requirement for creating web application it is necessary to create relation database that will store the data about classrooms, employees and time occupancy. It is necessary to create web application with user interface to simplify user actions within the reservation system. User interface is directly connected to the relation database that uses SQL queries to get the data. Main programming language for creating web application is C#, but it is necessary to use also HTML, CSS and JavaScript to create view that users will use in browser. Relation database stores all needed data for checking reservations and it will check for any wrong input of information into the system. Web application creates view for users to see all information that are required for making reservations.

KEY WORDS: MVC, C#, SQL, database, web application, reservations

Sadržaj

1. Uvod

Kod velikih i manjih organizacija uvijek postoji potreba za korištenjem organizacijskih prostora: dvorana, soba i sl. S većim brojem prostora javlja se problem kako pratiti iskorištenost i evidenciju takvih prostora. Prema tome korištenjem sustava za rezervaciju prostora može se jako dobro pratiti kako je taj prostor iskorišten. U velikim organizacija teško se može dogovoriti vrijeme korištenja određenih dvorana ili prostora za sastanke i zbog toga je sustav za rezervaciju dvorana ili prostorija gotovo nužan.

Trenutno većina organizacija koriste sustave za rezervacije koji omogućuju svim zaposlenicima uvid u prostorije za sastanke, te na takav način bilo koji zaposlenik koji ima pristup sustavu može zatražiti rezervaciju za takve sobe. Jedan od popularnih sustava je Microsoft Outlook sustav koji ima ugrađenu opciju za rezervacije. Takav sustav pruža vrlo jednostavan način gdje zaposlenici mogu vidjeti vrijeme u koje je određeni prostor zauzet, te odabrati drugu prostoriju ili odabrati vrijeme u kojem željena prostorija nije zauzeta. Uz trenutno dostupna rješenja moguće je i vrlo jednostavno napraviti takav sustav koji će se brinuti za rezervacije i paziti da zaposlenici mogu jednostavno rezervirati prostorije koje su im potrebne.

Za spremanje podataka organizacije može se koristiti relacijska baza podataka koristi SQL jezik za upite, odnosno dohvat podataka. Relacijska baza podataka se sastoji od tablica i atributa (stupaca) koji služe za jednostavno digitalno vođenje rezervacija prostorija.

Web aplikacija koja služi za jednostavniji pregled, unos, ažuriranja i brisanja informacija o rezervacijama se može napraviti točno po potrebama organizacije, te se za potrebe demonstracijskog sustava rezervacije dvorana u ovome radu koristio MVC (eng. Model-View-Controller) model izrade aplikacije. Kako bi se olakšao rad korisnika sa sustavom za rezervaciju izrađeno je web grafičko sučelje.

Rad se dijeli na 6 poglavlja:

1. Uvod
2. Opis i pohrana podataka
3. Izrada relacijske baze podataka
4. Opis i analiza MVC programskog alata
5. Izrada web sučelja za rezervaciju i pregled korištenja dvorana
6. Zaključak

2. Opis i pohrana prikupljenih podataka

Podaci pohranjeni u bazi podataka se odnose na bazu podataka koja služi za spremanje podataka o nastavnicima koji koriste sustav, dvoranama koje se rezerviraju, nastalim rezervacijama, te spremanju podataka rezervacije. Na početku rada baza podataka neće imati puno podataka, ali što se duže sustav koristi to će biti veća količina podataka tako da se preporučuje da se baza podataka održava, te da ukoliko postoje nepotrebni podaci poput starih rezervacije da se brišu za što efikasniji i brži rad sustava.

Podaci u o nastavnicima se odnose na opće podatke nastavnika (ime, prezime, e-mail, odjel), također ako je potrebno podaci se mogu proširiti tko je voditelj odjela, ali za osnovne potrebe rezervacija to zasada nije bilo potrebno. Zbog toga razloga potrebno je bilježiti podatke o rezervaciji dvorana koja sprema podatke o nastavniku koji rezervira dvoranu preko identifikacijskog broja nastavnika, dvorane preko identifikacijskog broja dvorane, te rezervacije s identifikacijskim brojem rezervacije.

Podaci o dvoranama se sastoje od naziva visokog učilišta, oznake dvorane, tip dvorane, kapacitet za predavanja, kapacitet za ispit i je li javna dvorana. Podaci o rezervacijama se sastoje od imena rezervacije, početka i kraja rezervacije.

Relacijske baze podataka koriste identifikatore koje nazivamo primarnim i stranim ključevima. Pomoću primarnog i stranog ključa može se točno dohvatiti informacija koja je potrebna bez mogućnosti drugih podataka zbog jedinstvenog identifikatora. Baza podataka pri kreiranju odabire vrijednost primarnog ključa i sama se brine da se ni u kojem slučaju ne može unijeti isti primarni ključ, to jest da svaki objekt ima atribut primarnog ključa koji je jedinstven. Relacijske baze podataka moraju imati jedinstven primarni ključ, te će se kasnije u radu spomenuti točan niz pravila koja relacijska baza podataka mora imati.

Uz podatke web aplikacije koji će se spremati u bazu podataka, također se spremaju i korisnički podaci za prijavljivanje u aplikaciju tako da se može pristupiti rezerviranju dvorana. Podaci nisu kriptirani i mogu se provjeravati u bazi podataka jer su spremljeni u tablici. . Uz korisničke podatke za prijavu također postoji još dodatni podatak za ovlasti, odnosno da li je korisnik administrator ili običan korisnik, te će i na temelju uloge imati drugačije ovlasti.

Za prijavljivanje u sustav potrebno je pohraniti korisničke podatke koji se koriste pri autentifikaciji korisnika i takvi podaci moraju se sigurno pohraniti u sustav, te uvijek moraju biti zaštićeni od neautoriziranog pristupa takvim podacima. Prema tome mora se dobro paziti na zaštitu podataka u bazi i dodatno mora se paziti da nije moguće kroz aplikaciju promijeniti podatke pohranjene u bazi podataka.

Baza podataka kako je već navedeno sprema veliku količinu podataka i za funkcioniranje web aplikacije potrebni su podaci s kojima može raditi. Bez pohranjenih podataka web aplikacija ne bi mogla raditi jer ne bi ni bilo moguće prikazati podatke poput nastavnika i dvorana. Za potrebe demonstracijskog sustava za rezervaciju dvorana koriste se podaci s Fakulteta prometnih znanosti koji sadrže sve podatke o

nastavnicima i dvoranama. Baza podataka osim općih podataka pohranjuje i sve podatke o rezervacijama koji se sastoje od imena rezervacije, početka i kraja rezervacija. Količina takvih podataka ovisi o tome koliko dvorana se koristi i koliko rezervacija postoji dok se podaci o nastavnicima i dvoranama rijetko mijenjaju.

Baza podataka koja sprema podatke o nastavnicima u tablicu nastavnik sastoji se od podataka svih nastavnika s Fakulteta prometnih znanosti kojih sveukupno ima 228, te svaki od njih ima svoj jedinstveni identifikator. Osnovni podaci koji se pohranjuju su ime i prezime nastavnika koje služi za identificiranje nastavnika i upotrebu ukoliko je potrebno pronaći nastavnika po imenu i/ili prezimenu. Uz imena i prezimena nastavnika također se pohranjuje i odjel u kojem rade nastavnici. Primjer podataka o nastavnicima se može vidjeti na slici 1.

	A	B	C	D	E
55	51	Francetić	Ivana	Zavod za aeronautiku	
56	52	Franjković	Davor	Zavod za aeronautiku	
57	53	Fratrović	Tomislav	Katedra za primjenjenu matematiku i statistiku	
58	54	Gerek	Suzana	Studentska služba	
59	55	Gerhardinger	David	Zavod za aeronautiku	
60	56	Gold	Hrvoje	Zavod za inteligentne transportne sustave	
61	57	Golubić	Jasna	Zavod za prometno planiranje	
62	58	Grebličkl	Marijana	Katedra za primjenjenu matematiku i statistiku	
63	59	Gregurić	Martin	Zavod za inteligentne transportne sustave	
64	60	Grgurević	Ivan	Zavod za informacijsko-komunikacijski promet	
65	61	Hajdinjak	Ivana	Zavod za aeronautiku	
66	62	Haramina	Hrvoje	Zavod za željeznički promet	
67	63	Horvat	Rajko	Zavod za cestovni promet	
68	64	Hozjan	Dubravka	Zavod za cestovni promet	
69	65	Hrkać	Ivana	Zavod za prometno planiranje	
70	66	Hunjak	Diana	Katedra za primjenjenu matematiku i statistiku	
71	67	Husnjak	Siniša	Zavod za informacijsko-komunikacijski promet	
72	68	Ivanjko	Edouard	Zavod za inteligentne transportne sustave	
73	69	Ivandić	Irena	Ured za projekte i transfer tehnologije	
74	70	Ivezić	Darko	Tehnička služba	
75	71	Ivošević	Jurica	Zavod za aeronautiku	
76	72	Jakara	Martina	Zavod za transportnu logistiku	
77	73	Jakovljević	Marijan	Zavod za prometno planiranje	
78	74	Jelušić	Niko	Katedra za opće programske sadržaje	
79	75	Jovančević	Romina	Tajništvo	
80	76	Jovanović	Bojan	Zavod za prometno planiranje	
81	77	Jović	Ivan	Zavod za informacijsko-komunikacijski promet	
82	78	Jurak	Julijan	Zavod za gradski promet	
83	79	Jurčević	Marinko	Zavod za prometno planiranje	
84	80	Juričić	Biljana	Zavod za aeronautiku	
85	81	Jurić	Ivo	Zavod za cestovni promet	
86	82	Kačan	Marin	Zavod za prometno planiranje	

Slika 1. Prikaz podataka nastavnika

Podaci o dvoranama su također preuzeti sa Fakulteta Prometnih Znanosti koji služe da se vide pojedine karakteristike dvorane, te da se može odabrati dvorana po potrebama nastavnika. Sastoji se od 38 dvorana ukupno dostupno za rezervaciju. Podaci o dvoranama sastoje se od imena visokog učilišta koje služi da ukoliko se koristi dvorana drugog učilišta da se može jasno vidjeti. Oznaka dvorane služi za

identificiranje dvorane, te da se može jednostavnije prepoznati koja se dvorana rezervira. Tip dvorane ovisi o tome kako je dvorana napravljena, te se svrstava u kategoriju amfiteatar, predavaonica, računalna učionica, višenamjenska dvorana, prostorija za laboratorijske vježbe, vijećnica ili online predavanje. Kapacitet za predavanje služi da se prikaže broj sjedećih mjesta i da se poveća efikasnost prilikom rezerviranja dvorane, odnosno da kapacitet odgovara broju studenta koji sluša predmet, te da se ne rezerviraju velike dvorane za mali broj studenta. Kapacitet za ispit omogućuje da kada nastavnik rezervira dvoranu za ispit može po broju prijavljenih ispita rezervirati odgovarajuću dvoranu. Javna dvorana označava koristi li se dvorana u javne potrebe ili ukoliko je navedeno da nije javna za druge potrebe. Prikaz podataka dvorane može se vidjeti na slici 2.

	Naziv visokog učilišta	Oznaka dvorane	Tip dvorane	Kapacitet za predavanja	Kapacitet za ispit	Javna dvorana
2	Fakultet prometnih znanosti	SD5	A	185	185	D
3	Fakultet prometnih znanosti	VD2	P	65	65	D
4	Fakultet prometnih znanosti	VD1	P	50	50	D
5	Fakultet prometnih znanosti	SD1	P	200	200	D
6	Fakultet prometnih znanosti	SD2	P	170	170	D
7	Fakultet prometnih znanosti	SD3	P	70	70	D
8	Fakultet prometnih znanosti	SD4	P	50	50	D
9	Fakultet prometnih znanosti	SD5	P	70	70	D
10	Fakultet prometnih znanosti	KMD	P	22	22	D
11	Fakultet prometnih znanosti	KVD	P	58	58	D
12	Fakultet prometnih znanosti	B70D1	A	48	48	D
13	Fakultet prometnih znanosti	B70D2	R	20	20	D
14	Fakultet prometnih znanosti	B70D3	A	48	48	D
15	Fakultet prometnih znanosti	B70D4	P	15	15	D
16	Fakultet prometnih znanosti	B70D5	R	12	12	D
17	Fakultet prometnih znanosti	B71D1	P	96	96	D
18	Fakultet prometnih znanosti	B71D2	P	56	56	D
19	Fakultet prometnih znanosti	B71D3	P	50	50	D
20	Fakultet prometnih znanosti	B71D4	P	96	96	D
21	Fakultet prometnih znanosti	B71D5	P	96	96	D
22	Fakultet prometnih znanosti	B71D6	P	96	96	D
23	Fakultet prometnih znanosti	B71D7	P	20	20	D
24	Fakultet prometnih znanosti	B71D8	R	20	20	D
25	Fakultet prometnih znanosti	B71D9	R	20	22	D

Slika 2. Prikaz podataka dvorana

Osim preuzetih podataka koriste se i informacije o korisničkim podacima za prijavljivanje u sustav, oni su povezani sa podacima nastavnika preko identifikatora nastavnika koji se sprema pod korisnički račun za prijavljivanje. Pohranjuje se jedinstveni identifikator korisničkog računa, uloga korisničkog računa (admin, obični korisnik), korisnički podaci za prijavljivanje u sustav s korisničkim imenom i lozinkom.

Pohranjuju se i podaci o rezervacijama u tablicu event koja služi za spremanje imena rezervacije koje služi za definiranje razloga rezervacije ukoliko je potrebno navesti razlog, početka i kraja rezervacije za prikaz unutar kalendara.

3. Izrada relacijske baze podataka

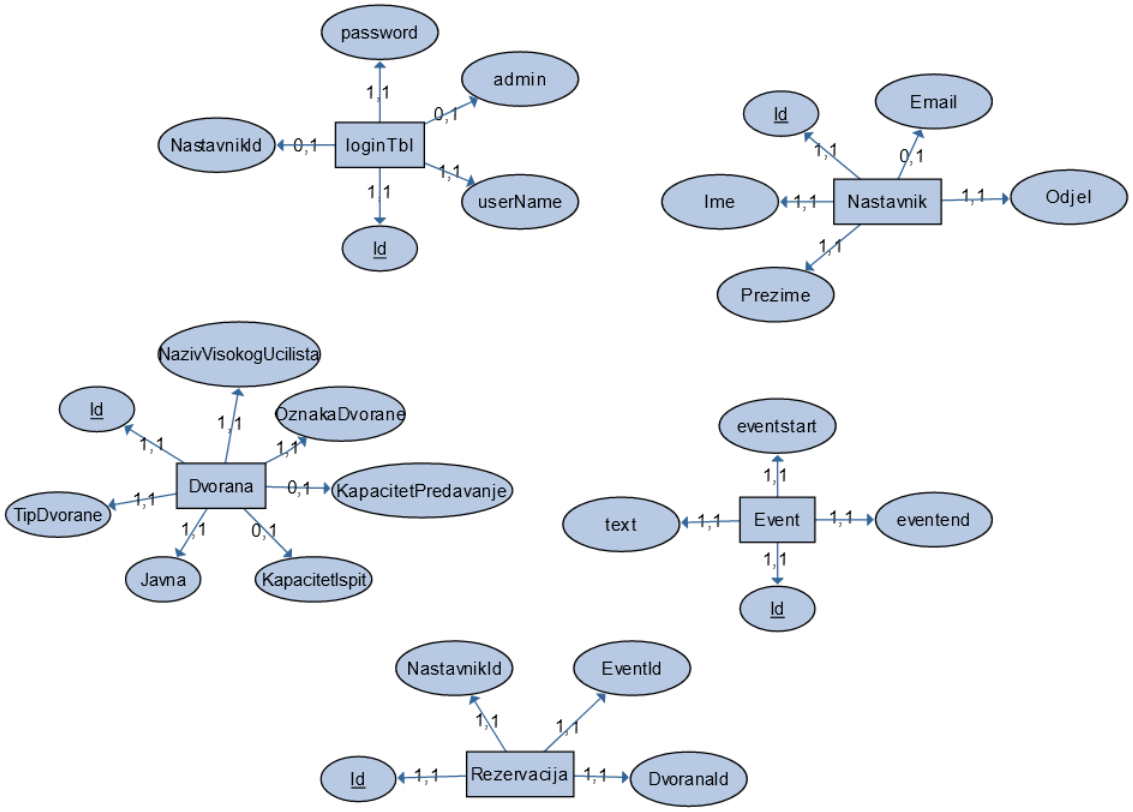
Relacijska baza podataka je napravljena u SQL Server Management Studio programskom alatu, te prema tome radi se o Microsoft (MS) bazi podataka. Prije same izrade relacijske baze podataka potrebno je napraviti ER (Entity-relationship) model da se može jasno definirati kako će baza podataka biti konstruirana. Vrlo je bitno dobro definirati korisničke potrebe, a i potrebe za rad aplikacije. Ukoliko je za rad aplikacije potrebna baza podataka, veliki problem nastaje ukoliko je baza podataka puštena u rad, a potrebne su dodatne izmjene.

3.1. Entity-relationship model

ER model na jednostavan način pomoću geometrijskih tijela prikazuje/definira karakteristike baze podataka, te prikazuje na koji način će funkcionirati baza podataka i u kojem su odnosu same tablice u bazi podataka. Pri izradi ER modela prvo se odrede dijagrami entiteta koji služe da se vide entiteti u modelu i koje atribute će imati svaki od entiteta.

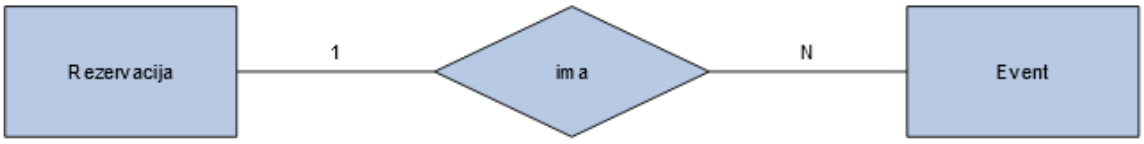
Entiteti su objekti ili događaji o kojima se spremaju informacije, te se sastoje od atributa. Atributi su obilježja entiteta pomoću kojih definiramo informacije koje se spremaju u bazu podataka, te su potrebni za definiranje informacija prilikom njihova prikaza. Atributima se bilježi kakav će podatak biti i jasno se definira što podatak označava, npr. podatak o entitetu koji označava broj sjedećih mjesta u dvorani bez atributa koji označava broj sjedećih mjesta ne bi bio razumljiv.

Atribut koji se označava kao identifikator služi za jedinstveno označavanje jednog elementa entiteta tako da se uvijek može jasno i jedinstveno identificirati na koji se elementu entitetu informacija odnosi. Kardinalitet atributa označuje koliko vrijednosti pojedini atribut može imati za opis jednog elementa entiteta. Na slici 3 može se vidjeti dijagram entiteta relacijske baze podataka za potrebe web aplikacije za rezervaciju dvorana.



Slika 3. Dijagram entiteta baze podataka

Nakon izrade dijagrama entiteta prelazi se na odnose između tablica koje su prikazane ER dijagramom. On prikazuje međusobne odnose između entiteta. Kao što je prikazano na slici 4 tablica rezervacija i tablica event su u 1-N odnosu što znači da za svaku rezervaciju ima N objekata eventa, a svaki objekt event ima 1 rezervaciju.



Slika 4. ER dijagrama baze podataka

Najčešće veze koje postoje su jednostavne veze i one se sastoje od:

- 1:1 – sastoji se od odnosa koji označava da jedan element entitet sadrži samo jedan element drugog entiteta s kojim je u vezi,
- 1:N – sastoji se od odnosa koji označava da jedan entitet može biti povezan s N elementa drugog entiteta s kojim je u vezi, dok element drugog entiteta može biti povezan samo s jednim elementom,

- M:N – sastoji se od odnosa koji označava da svaki element entiteta može biti povezan s N elementa drugog entiteta i obrnuto

3.2. Pretvorba dijagrama entiteta i ER dijagrama u relacijsku shemu

Napravljeni ER model služi za prikaz što je potrebno napraviti u bazi podataka da ona može funkcionirati po korisničkim potrebama. Pomoću ER modela prikazuje se struktura baze podataka koja služi da se može pristupiti samoj izradi baze podataka u programskom alatu koji je namijenjen za izradu baze podataka. Veze koje se mogu vidjeti u ER dijagramu su jako važne pri izradi baze podataka.

Nakon napravljenog ER modela može se vidjeti koji atributi su potrebni određenoj tablici, te kardinalitet pojedinog atributa i odnosi između entiteta koji su zapravo tablice u bazi podataka.

Na osnovu ER dijagrama se pomoću transformacijskih pravila kreira relacijska baza podataka. Svaki od entiteta je zasebna tablica koja ima isti naziv i svaki od atributa će biti jedan stupac tablice. Veze između entiteta koje se mogu vidjeti u ER modelu služe kao vodič kako treba izgledati struktura baze podataka.

Transformacijska pravila se sastoje od, [1]:

- 1. transformacijsko pravilo – Dijagram entiteta ER modela preslikava se u relacijski model na sljedeći način:
  - Svaki entitet postaje jedna tablica – ime tablice je isto
  - Svaki atribut postaje jedan stupac tablice
  - Svaka tablica ima jedinstven primarni ključ
- 2. transformacijsko pravilo – Prikaz veze 1:1 ostvaruje se na tri načina ovisno o članstvu u vezi:
  - 1. način – ako su članstva E1 i E2 obavezna
    - Entiteti E1 i E2 spajaju se u jednu tablicu sa unijom njihovih atributa
  - 2. način – ako je članstvo samo jednog od entiteta obavezno
    - U entitetu koji je obavezan dodaje se strani ključ entiteta koji nije obavezan
  - 3. način – ako su oba članstva za E1 i E2 neobavezna
    - Radi se još jedna tablica koja sadrži primarne ključeve od obje tablice
- 3. transformacijsko pravilo – pretvaranje binarnih veza 1:N
  - Ako su E1 i E2 u vezi 1:N onda E1 ne utječe na vezu dok se E2 proširuje dodatnim atributom (stranim ključem) koji je primarni ključ u E1
  - Ako je članstvo E1 neobavezno strani ključ E2 se postavi da može postati NULL tip
- 4. transformacijsko pravilo – pretvaranje binarnih veza M:N
  - Za transformaciju veze M:N uvijek se uvodi nova tablica koja se sastoji od primarnih ključeva entiteta E1 i E2 koji zajedno čine primarni ključ nove tablice

- 5. transformacijsko pravilo – pretvaranje involuirane veze
  - Radi se način da se prepozna veza (1:1, 1:N i M:N) entiteta sa samim sobom zatim se radi pretvaranje kao kod jednostavnih veza
- 6. transformacijsko pravilo – pretvaranje podskup veze
  - Pretvaranje podskup veze se radi na način da svi podskupovi u vezi kreiraju u posebne tablice i sadrže samo svoje specifične atribute te strani ključ na nadskup tablicu
- 7. transformacijsko pravilo – pretvaranje ternarne veze
  - Ternarna veza se u relacijski model transformira tako što se svaki od entiteta prikazuje posebnom tablicu, za povezivanje se uvodi nova tablica koja sadrži primarne ključeve od sva tri entiteta i primarni ključ nove tablice mogu biti ta tri strana ključa ili zasebni primarni ključ [1].

Po navedenim transformacijskim pravilima može se jasno vidjeti koji su potrebni koraci za izradu baze podataka iz ER modela. Kod izrade baze podataka mora se pridržavati navedenih pravila za što jednostavnije i efikasnije funkcioniranje baze podataka.

Tablica rezervacija preko tri ključa povezuje sve potrebne podatke za identifikaciju nastavnika koji rezervira, te dvorane koja je rezervirana i podatke o rezervaciji poput koliko je određeno vrijeme rezervacije. Ako je nužno može se i točno navesti razlog rezervacije preko stranog ključa event tablice. Sustav za rezervaciju dvorana dopušta jednostavan unos informacija u bazu podataka koja prikuplja, sprema i održava integritet informacije.

Web aplikacija za rezervaciju dvorana pri zahtjevu dohvaća informacije o rezervacijama iz baze podataka, te se pomoću jednostavnog web sučelja prikazuje korisniku. U slučaju unosa podataka koji su krivo uneseni baza podataka neće unijeti takve podatke, te će odbiti unos.

Rezervacija u bazi podataka se može vrlo jednostavno mijenjati putem web sučelja koje prikazuje rezervacije, te se može promijeniti vrijeme trajanja, datum i kreirati nova rezervacija koja sprema nove informacije o rezervacijama u bazu podataka, te se postojeći podaci ažuriraju.

Pri rezerviranju dvorane u web aplikaciji unosi se naziv rezervacije, te se može unijeti razlog rezervacije u vrlo jednostavnom unosu atributa teksta rezervacije. U bazi podataka označen je obavezan unos svih atributa tablice event koja služi za spremanje podataka o rezervaciji koja se kreira i pomoću koje se ažuriraju podaci početka i kraja rezervacije, te može se vidjeti na slici 5.

Results | Messages

	id	text	eventstart	eventend
1	1	Rezervacija Dvorane 1 - za predavanje iz Algorita...	2020-06-29 11:00:00.000	2020-06-29 15:30:00.000
2	2	Rezervacija Dvorane 2 - predavanje iz ViS	2020-07-01 11:00:00.000	2020-07-01 15:00:00.000
3	3	Rezervacija Dvorane 1 - ispitni rokovi	2020-07-02 09:30:00.000	2020-07-02 14:00:00.000
4	4	Rezervacija Dvorane 1 - Prezentacija o ERASMUS	2020-07-02 14:00:00.000	2020-07-02 17:00:00.000

Slika 5. Prikaz podataka tablice event

Web aplikacija dopušta pomicanje rezervacije jednostavnim pomicanjem rezervacije u web sučelju koji sprema novi datum i vrijeme rezervacije, te se u relacijskoj bazi podataka ažuriraju atributi datum početka rezervacije i datum kraja rezervacije preko identifikacijskog broja rezervacije.

Preko posebnog web sučelja može se također pristupiti nastavnicima koji rezerviraju dvoranu pomoću tablice nastavnik koja sprema podatke o nastavnicima i omogućuje jednostavan unos novog nastavnika.

Prikaz u bazi podataka može se vidjeti na slici 6 koji prikazuje jednostavnu formu potrebnih podataka pri kreiranju novog nastavnika. Kao i u prethodnoj tablici korišteni su samo obavezni atributi, ali po potrebama korisnika može se proširiti na dodatne atribute.

Id	Ime	Prezime	Email	Odjel
1	Tomislav	Erdelić	terdelić@fpz.unizg.com	Zavod za inteligente transportne sustave
3	Marko	Matulin	mmatulin@fpz.unizg.com	Zavod za informacijsko-komunikacijski promet

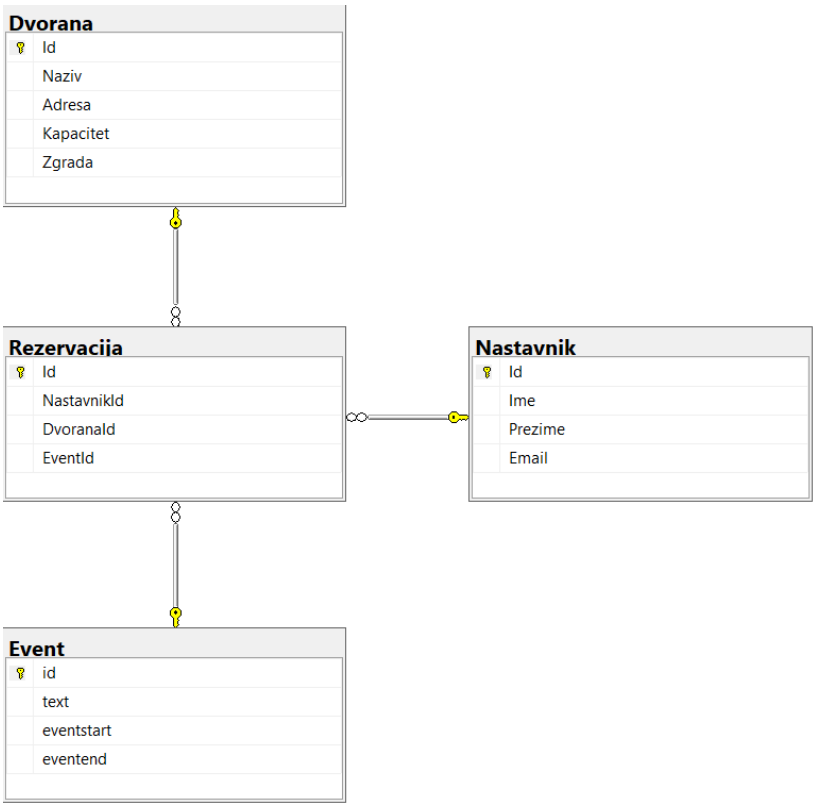
Slika 6. Prikaz podataka nastavnika u bazi podataka

Tablica nastavnik se sastoji od atributa nastavnik identifikator, odjel, ime i prezime nastavnika i email. Ako je potrebno vidjeti tko je napravio rezervaciju s tim podacima može se lagano kontaktirati nastavnik. Pruža također uvid u nastavnike koji koriste web aplikaciju za rezervaciju dvorana.

Tablica dvorana služi za opće podatke o dvoranama koje su dostupne za rezervaciju poput dvorana identifikatora, naziv visokog učilišta, oznake dvorane, broj sjedala za predavanja u dvorani, broj sjedala za ispit, kategorija dvorane za javnu ili privatnu upotrebu . Takvi atributi jednostavno pružaju uvid u podatke koji mogu biti bitni pri samom rezerviranju dvorane.

Tablice u relacijskoj bazi podataka imaju međusobne odnose koji služe da relacijska baza podataka služi kao jedan cjeloviti sustav u kojem se može naći povezanost između samih tablica.

Nije potrebno da je svaka tablica povezana sa svakom, ali povezanost više tablica može se jako dobro iskoristiti, kao što je već prije navedeno, da se prikaže cjelovita informacija iz baze podataka koja je potrebna. U izrađenoj relacijskoj bazi podataka za potrebe web aplikacije rezervacije dvorana dobro služi povezanost tablica nastavnika, rezervacija i dvorana jer u kombinaciji tih tablica može se koristiti još jedna kojom možemo vidjeti tko je napravio rezervaciju kao što je prikazano na slici 7.



Slika 7. Prikaz povezanosti tablica pomoću primarnih i stranih ključeva

3.3. Tipovi podataka baze podataka

Pri izradi relacijske baze podataka točno se određuje format atributa koji se unosi u sustav. Postoji više vrsta formata koji se koriste pri unosu informacije u sustav kako ne bi došlo do pogrešnog unosa i tako narušio rad relacijske baze podataka. Najčešće vrste podataka koji se mogu koristiti pri izradi relacijske baze podataka, [2]:

- int (Integer) – cijeli broj u rasponu od -2147483648 do 2147483647,
- nvarchar (national character varying) – dopušta varijabilni unicode unos karaktera,
- date time – datum i vrijeme u formatu yyyy-mm-dd hh:mm:ss,
- varchar (character varying) – varijabilna dužina kod unosa (slovo, broj, specijalni znakovi) koja će spremiti karaktere koji su upisani,
- char (character) – fiksni broj dopuštenih karaktera (slovo, broj, specijalni znakovi) za unos,
- nchar (national character) – kao i kod char tipa podatka samo što dopušta unicode unos,
- money – pohranjuje se unos brojeva sa 4 decimale u rasponu od -922,337,203,685,477.5808 do 922,337,203,685,477.5807 [2].

Kod odabira tipa podataka poput nvarchar mora se dodatno navesti do koliko znakova će baza podataka dopustiti unos za pojedini atribut, te ukoliko ima više od dozvoljenog broja znakova unos neće biti u potpunosti spremljen jer će se višak znakova odrezati.

Za potrebe rezervacijskog sustava najčešće korišteni formati su integer i national varying character, ali također može se koristiti format za vrijeme početka i kraja rezervacije date time. Date time format za spremanje datuma i vremena rezervacije služi za vrlo jednostavno korištenje baze podataka za pretraživanje i spremanje takve informacije bez potrebe za bilokakvo dodatno dodavanje kompleksnosti relacijske baze podataka. Na slici 8 može se vidjeti odabir tipova podataka u tablici event koja služi za web sučelje.

Napravljena baza podataka koristi veliki broj atributa koji svaki ima zadani svoj tip podataka. Tablica event koristi sljedeće tipove podataka koji su obavezni ili neobavezni za unos:

- Id – int (cijeli broj) obavezan,
- text – nvarchar (tekstualni unos) obavezan,
- eventstart – datetime (datum i vrijeme) obavezan,
- eventend – datetime (datum i vrijeme) obavezan.

Tablica nastavnik koristi sljedeće tipove podataka:

- Id – int (cijeli broj) obavezan,
- Ime – nvarchar (tekstualni unos) obavezan,
- Prezime – nvarchar (tekstualni unos) obavezan,
- Email – nvarchar (tekstualni unos) neobavezan,
- Odjel – nvarchar (tekstualni unos) obavezan.

Tablica dvorana koristi sljedeće tipove podataka:

- Id – int (cijeli broj) obavezan,
- NazivVisokogUcilista – nvarchar (tekstualni unos) obavezan,
- OznakaDvorane – nvarchar (tekstualni unos) obavezan,
- KapacitetPredavanje – int (cijeli broj) neobavezan,
- KapacitetIspit – int (cijeli broj) neobavezan,
- Javna – char (znakovni unos) obavezan,
- TipDvorane – char (znakovni unos) obavezan.

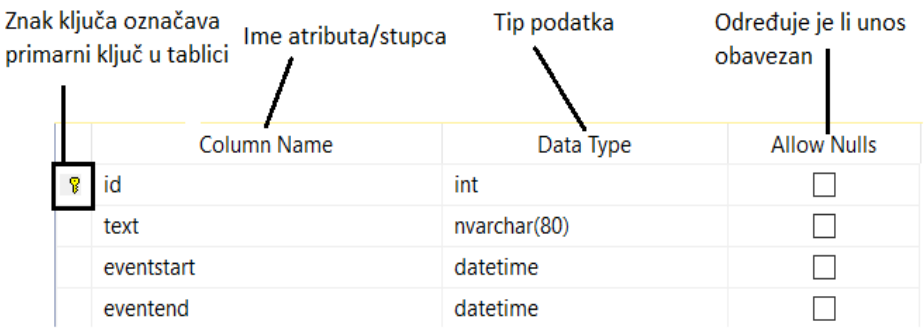
Tablica rezervacija koristi sljedeće tipove podataka:

- Id – int (cijeli broj) obavezan,
- NastavnikId – int (cijeli broj) obavezan,
- EventId – int (cijeli broj) obavezan,
- DvoranaId – int (cijeli broj) obavezan.

Tablica loginTbl koristi sljedeće tipove podataka:

- Id – int (cijeli broj) obavezan,
- userName – nvarchar (tekstualni unos) obavezan,

- password – nvarchar (tekstualni unos) obavezan,
- admin – char (znakovni unos) neobavezan,
- NastavnikId – int (cijeli broj) neobavezan.



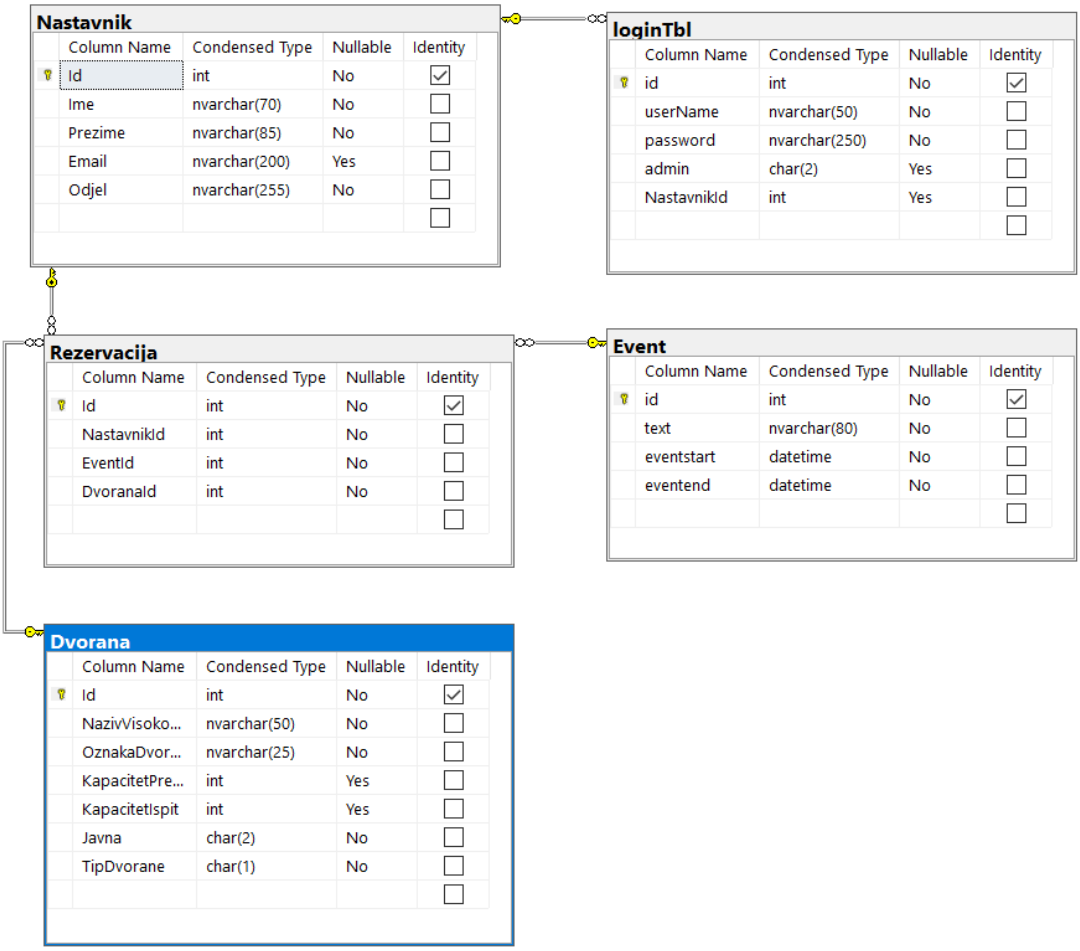
Slika 8. Prikaz tipova podataka tablice event

Izrada relacijske baze podataka može biti preko sučelja za izradu tablice, ali može se i korištenjem SQL naredbi kreirati baza podataka, te i brisati ukoliko je to potrebno. Preko kôdiranja tablice može se vidjeti cjelokupna struktura tablice što pruža dobar uvid u strukturu i funkcioniranje baze podataka kao što se može vidjeti na slici 9.

```
CREATE DATABASE ZavrsniRad01
CREATE TABLE Dvorana
(Id int NOT NULL PRIMARY KEY IDENTITY (1,1),
NazivVisokogUcilista nvarchar(50) NOT NULL,
OznakaDvorane nvarchar(25) NOT NULL,
KapacitetPredavanje int NULL,
KapacitetIspit int NULL,
Javna char(2) NOT NULL,
TipDvorane char(1) NOT NULL)
CREATE TABLE Event
(id int NOT NULL PRIMARY KEY IDENTITY(1,1),
text nvarchar(80) NOT NULL,
eventstart datetime NOT NULL,
eventend datetime NOT NULL)
CREATE TABLE Nastavnik
(Id int NOT NULL PRIMARY KEY IDENTITY(1,1),
Ime nvarchar(70) NOT NULL,
Prezime nvarchar(85) NOT NULL,
Email nvarchar(200) NULL,
Odjel nvarchar(255) NOT NULL)
CREATE TABLE loginTbl
(Id int NOT NULL PRIMARY KEY IDENTITY(1,1),
userName nvarchar(50) NOT NULL,
password nvarchar(250) NOT NULL,
admin char(2) NULL,
NastavnikId int NULL FOREIGN KEY REFERENCES Nastavnik(Id))
CREATE TABLE Rezervacija
(Id int NOT NULL PRIMARY KEY IDENTITY(1,1),
NastavnikId int NOT NULL FOREIGN KEY REFERENCES Nastavnik(Id),
EventId int NOT NULL FOREIGN KEY REFERENCES Event(id),
DvoranaId int NOT NULL FOREIGN KEY REFERENCES Dvorana(Id))
```

Slika 9. SQL kôd za kreiranje baze podataka

Nakon kreiranja tablica po kôdu iz prethodne slike dobiva se dijagrama baze podataka prikazan na slici 10.



Slika 10. Prikaz kreirane baze podataka iz SQL kôda

3.4. Korištenje relacijske baze podataka

Relacijske baze podataka dominantne su na tržištu jer pružaju visoku efikasnost u području spremanja velikih količina potrebnih informacija o svijetu oko nas. Mogu se koristiti za sve današnje potrebe modernog svijeta poput bankarskih sustava, sustava za praćenje inventara, sustavi za praćenje transakcija pojedine organizacije i sl. Mogu se upotrijebiti na razne probleme uz dovoljno znanja za izradu i korištenje relacijske baze podataka.

Podacima u MS bazi podataka može se upravljati naredbama poput SELECT – za dohvat podataka, DELETE – za brisanje podataka, INSERT – dodavanje novih podataka i ALTER, UPDATE - mijenjanje postojećih podataka.

Procesiranje SQL upita odvija se u slijedećim koracima, [3]:

- Provjera sintakse

- Provjerava se da li je upit ispravan; ako da parsira ga se u logičko stablo, gdje svaka grana stabla predstavlja logičku operaciju koju upit treba izvršiti,

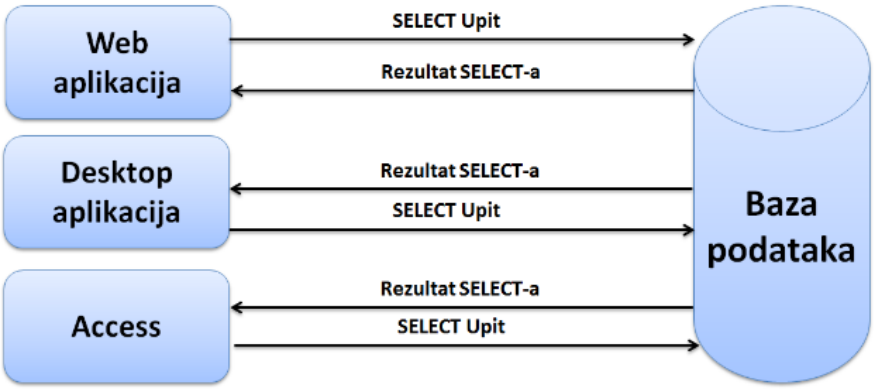
❖ Provjera objekata koji se referenciraju
- Provjerava se da li struktura relacijske sheme odgovara upitu,

❖ Optimizator upita
- Optimizator upita generira moguće planove izvršavanja upita, provjerava da li već postoje upiti u cache memoriji i odabire onaj koji je najpovoljniji,

❖ Plan izvršavanja
- Plan izvršavanja se sprema u cache memoriju i izvršava se na execution engine-u,

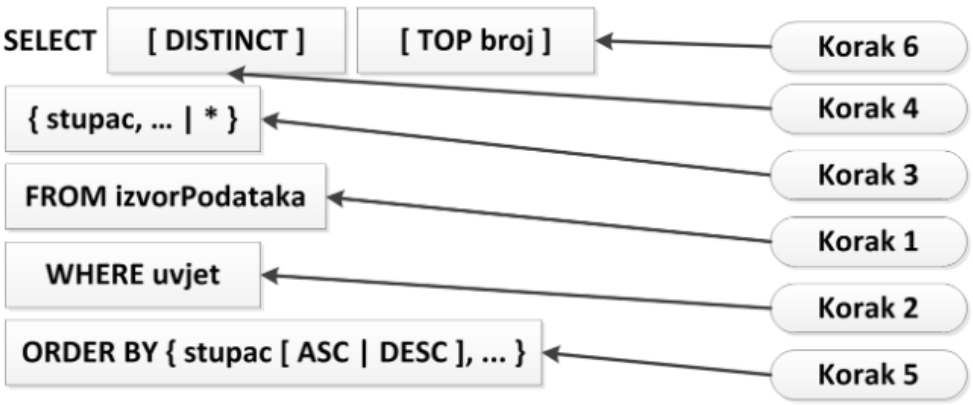
❖ Rezultat upita [3].

Za dohvaćanje već spremljenih podataka iz tablica baze podataka koristi se naredba SELECT. SELECT naredba vraća skup podataka u tabličnom obliku. Primjer kako SELECT naredba dohvaća podatke se može vidjeti na slici 11.



Slika 11. Prikaz dohvaćanja SELECT naredbom, Izvor: [3]

SELECT naredba je kompleksna i ima mnogo opcija. Osnovni izgled SELECT naredbe prikazan je na slici 12.



Slika 12. Prikaz osnovnih opcija SELECT naredbe, Izvor: [3]

Ostale naredbe poput DELETE, ALTER i UPDATE se mogu koristiti na sličan način kao i SELECT naredba gdje je najbitnije dobro definirati gdje se nalazi podatak koji se želi promijeniti i uvjete koji omogućuju da se dohvati podatak koji se želi promijeniti.

SELECT naredba za dohvat podataka iz baze podataka pomoću kojih se prikazuju rezervacije za dvorane, nastavnici koji koriste sustav ili dvorane koje se koriste za rezervacije su definirane i web aplikacija će sama napraviti naredbu ovisno o podacima koji se prikazuju. Primjer SELECT naredbe za dohvat rezervacija:

```
SELECT * FROM Event
```

Bez obzira iz kojeg programa sustav šalje naredbu kao na primjer iz web aplikacije ili SQL server manager-a baza podataka će uvijek odgovoriti isto . Ukoliko je potrebno na primjeru slike 9. može se kreirati komanda koja sadrži uvjete da se prikaže točna informacija koja se traži od baze podataka, te ukoliko se traži nastavnik iz tablice nastavnik može se napraviti pretraga po imenu traženog nastavnika kao što je prikazano sljedećom naredbom:

```
SELECT * FROM Nastavnik
WHERE Ime = 'Ime_nastavnika'
ORDER BY Ime DESC
```

UPDATE naredba se koristi za ažuriranje podataka u bazi podataka, koja služi ukoliko korisnik želi promijeniti podatke o rezervacijama, nastavnicima i dvoranama. Na primjer ako nastavnik promjeni e-mail, sustav će sam kreirati naredbu za promjenu u bazi podataka, te korisnik samo treba promjeniti informaciju pomoću web aplikacije. Primjer UPDATE naredbe:

```
UPDATE Nastavnik
SET Email = 'primjerbr1@provider.com'
WHERE Ime = 'Tomislav'
```

ALTER ALTER naredba se koristi ukoliko se želi dodati stupac, promijeniti tip podatka stupca ili obrisati stupac. Primjer ALTER naredbe:

```
ALTER TABLE Nastavnik
ADD Adresa nvarchar(255)
```

DELETE naredba se koristi za brisanje informacija iz baze podataka. Ukoliko korisnik želi obrisati rezervaciju, nastavnika ili dvoranu može se koristiti opciju delete. Primjer DELETE naredbe:

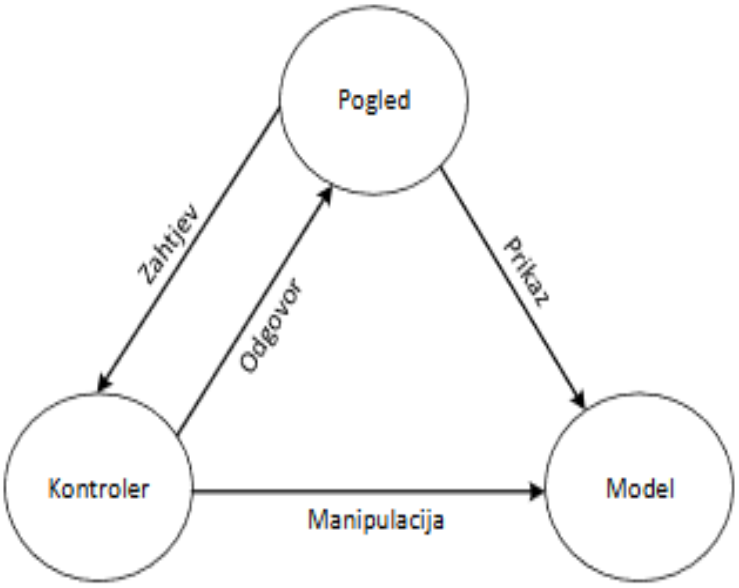
```
DELETE FROM Nastavnik
WHERE Ime = 'Tomislav'
```

4. Opis i analiza MVC programskog alata

Za izradu MVC (eng. Model-View-Controller) aplikacije korišten je Visual Studio koji pruža opcije za izradu MVC aplikacija zbog mogućnosti korištenja više različitih programskih jezika u jednom programskom alatu.

MVC je kratica od Model, View and Controller, odnosno MVC aplikacija sadrži tri komponente, [4]:

- Model predstavlja oblik podataka i poslovnu logiku te sadrži podatke aplikacije. Objekti modela dohvaćaju i spremaju stanje modela u bazu podataka.
- Pogled (View) je korisničko sučelje koje omogućuje pregledavanje, ažuriranje i brisanje podataka.
- Kontroler (Controller) obrađuje korisnički zahtjev koji je najčešće došao iz pogleda (View) u kojem je korisnik generiralo URL zahtjev. Kao odgovor kontroler vraća pogled koji sadrži podatke modela [4].



Slika 13. MVC arhitektura, Izvor: [4]

Svatko može skinuti besplatnu verziju Visual Studio alata koja je namijenjena za studente koji uče programirati i za edukaciju studenta za korištenje programskih alata.

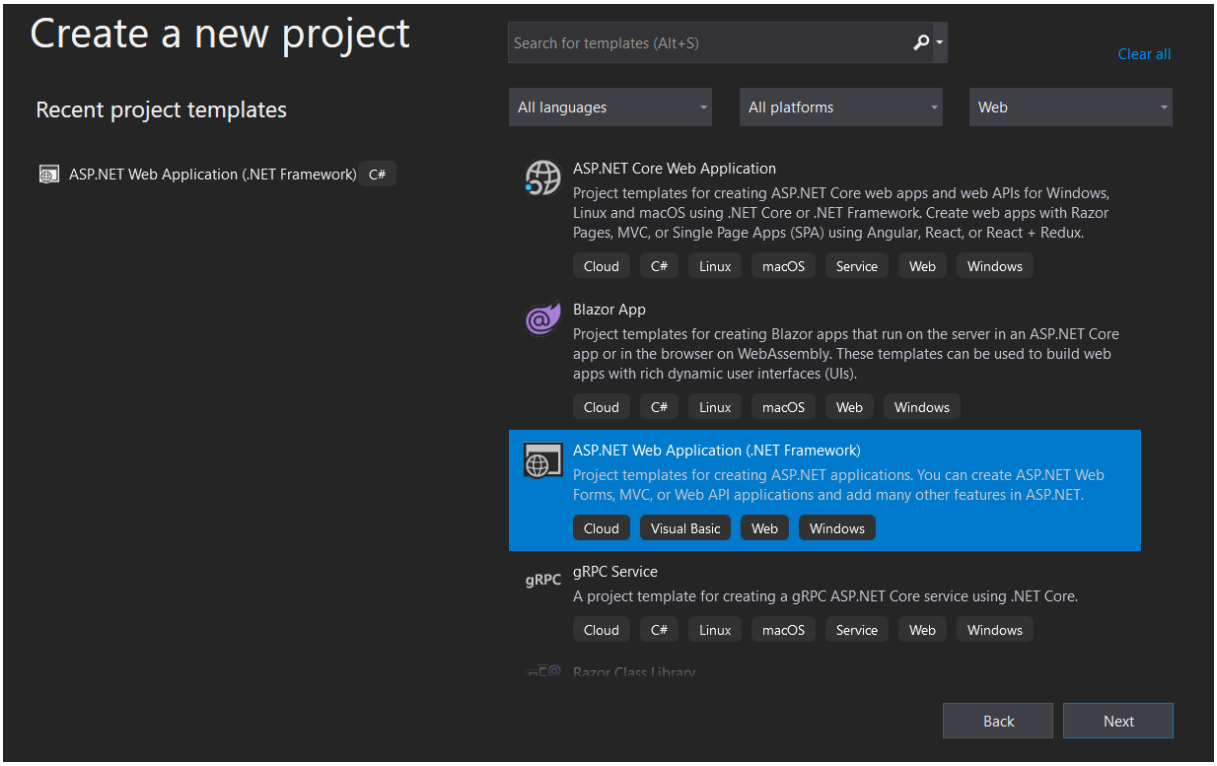
Visual Studio moćan je alat koji pruža velik broj usluga za izradu web aplikacija, te se može kreirati i baza podataka iz samog programskog alata. Visual Studio omogućuje i uvid u samu potrošnju memorije i procesora pri paljenju aplikacije što može služiti ukoliko se želi napraviti memorijski efikasna aplikacija.

4.1. Predlošci za izradu aplikacije

Unaprijed napravljeni predlošci MVC aplikacije ubrzavaju izradu konačnog izgleda aplikacije. Za aplikaciju rezervacije dvorana odabran je predložak koji kreira potrebne direktorije za modele, poglede i kontrolere kod kojih se također mogu kreirati novi pogled ili kontroler po predlošku.

Visual studio također ima već ugrađen alat za pokretanje aplikacije koji se može dodatno izmijeniti po potrebi, npr. odabir preglednika u kojem se testira web aplikacija tako da se može vidjeti kako će u pojedinom pregledniku raditi web aplikacija. Za MVC web aplikacije može se odabrati velik broj scenarija za pokretanje testnih okruženja u kojima se web aplikacija može koristiti.

Odabran predložak je ASP.NET Core Web app koji pruža predefinirano okružje za izradu takve aplikacije. Kreira foldere u koje se mogu stavljati modeli, pogledi i kontroleri koji se kreiraju. Pri izradi programa jako je bitno imati dobar pregled u različite dijelove aplikacije. U direktoriju Models mogu se vidjeti sve klase koje sustav koristi u web aplikaciji kreiranoj na bazi odabranog predloška. Osim direktorija koji omogućuju lakše snalaženje kreiraju se i ostali dijelovi za efikasniju izradu poput konfiguracije pri pokretanju web aplikacije i ruta koje se koriste. Odabrani predložak i ostali dostupni predlošci mogu se vidjeti na slici 14.



Slika 14. Predlošci za izradu aplikacija

4.2. Povezivanje web aplikacije s bazom podataka i NuGet Paketi

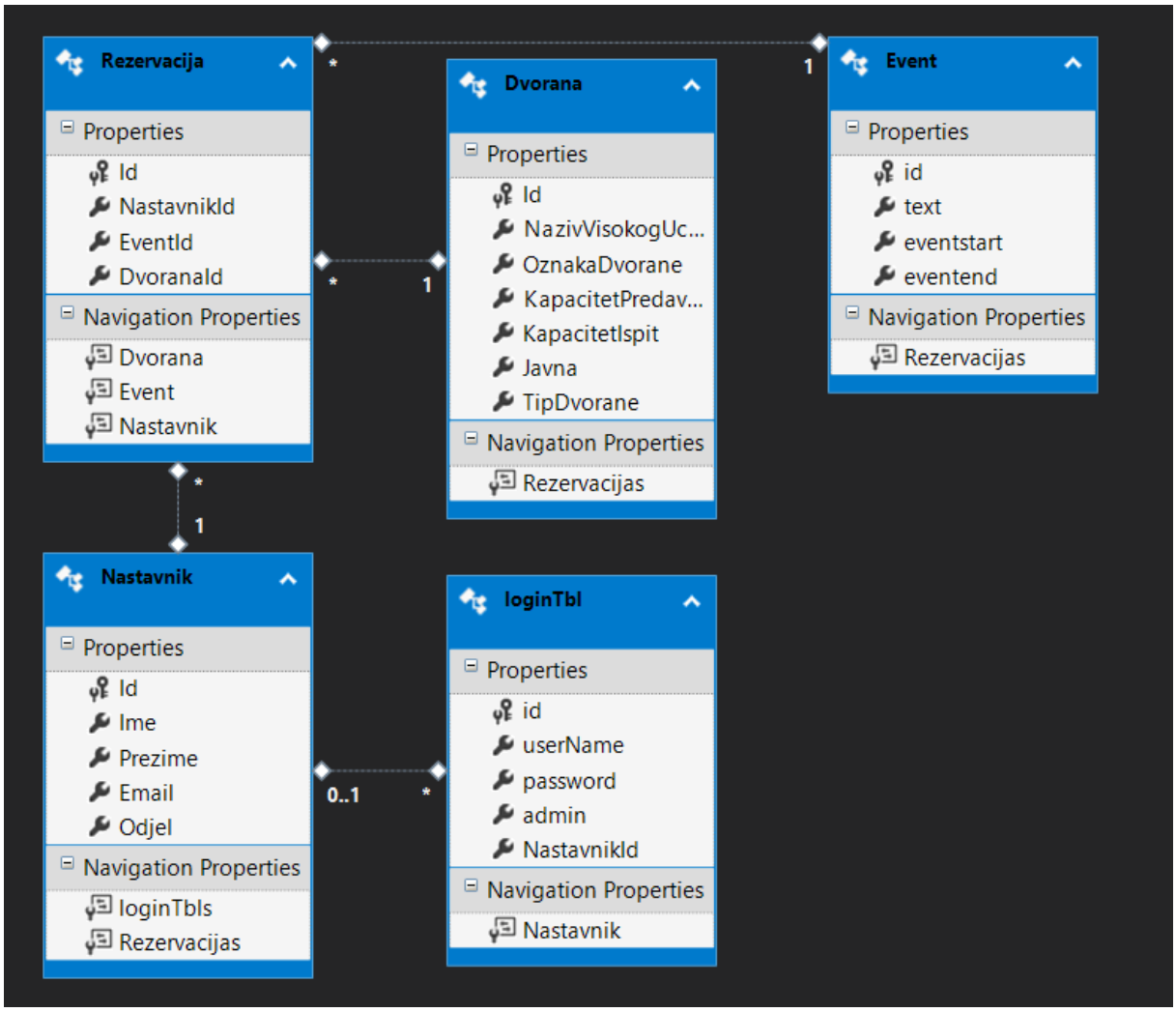
Povezivanje web aplikacije preko Visual Studio programskog alata se može jednostavno napraviti preko korištenja komanda ugrađenih u visual studio programski alat, ali i također preko kreiranja modela pomoću entity framework-a.

Za sve dodatne alate pruža se veliki broj dodataka nazvani NuGet paketi koji pružaju dodatnu fleksibilnost pri izradi aplikacija. Moguće je instalirati koliko god je potrebno paketa koji se mogu koristiti za razne situacije od provjere kôdiranja do paketa koji pružaju uvid u kvalitetu kôda, a sve s ciljem ubrzanja izrade aplikacije.

Web aplikacija mora se povezati sa serverom ukoliko se želi koristiti baza podataka za pohranu i dohvaćanje podataka koje sustav koristi. Postoji više odabira koji se koriste pri spajanju s bazom podataka jer može postajati više različitih načina spremanja baza podataka.

Jedan od korištenih NuGet paketa za izradu demonstracijskog sustava za rezervaciju dvorana je spomenuti NuGet paket entity framework koji znatno olakšava povezivanje i komunikaciju između demonstracijskog sustava za rezervaciju dvorana i relacijske baze podataka. Pomoću entity framework-a može se vrlo jednostavno povezati baza podataka sa sučeljem koje služi kao vodič za kreaciju klase koja služi da se razmjenjuju podaci između baze podataka i aplikacije. Kreirana veza se može koristiti bilo gdje u programu pod korisnički definiranim imenom klase.

Entity framework je alat koji omogućuje jednostavnu izradu konekcije između web aplikacije i baze podataka koja kreira instancu konekcije, čijim se pozivanjem mogu dohvaćati, brisati i ažurirati informacije u bazi podataka. Mogu se koristiti različiti definirani načini izrade konekcije između web aplikacije i baze podataka. Odabran predložak za potrebe web aplikacije je ASP.NET Core Web app. Entity framework kreira konekciju i potom kreira model na primjeru tablica iz baze podataka koja je odabrana. Model koji je napravljen je zapravo klasa unutar programa, te je identična tablici u bazi podataka. Takve klase mogu se kasnije i ažurirati jer one nisu ništa drukčije od samostalno izrađenih modela u programu. Model izrađen pomoću entity frameworka može se vidjeti na slici 15.



Slika 15. Model izrađen pomoću entity framework-a

Potrebni modeli su izrađeni prema strukturi baze podataka tako da je moguće razmjenjivati podatke iz baze podataka u aplikaciju, te iz aplikacije slati podatke u bazu podataka i ažurirati podatke. Većinu osnovnih operacija Visual Studio može kreirati bez potrebe za dodatnim kôdiranjem što pruža vrlo brz i efikasan način za izradu novih aplikacija. Služi i kao jako moćan alat za učenje kôdiranja aplikacije na bazi MVC modela jer se Visual Studio sam brine za rad aplikacije.

Način spremanja baze podataka može biti kao datoteka, lokalni server računala ili preko online servera baze podataka. Povezivanje ukoliko se ne radi preko entity framework-a može se napraviti pozivanje komande za SQL konekciju u kojoj se definira lokacija baze podataka na računalu.

Mogu se unositi i SQL naredbe preko kojih se mogu prikazati spremljeni podaci u bazi podataka. Ukoliko se radi povezivanje preko entity framework-a postoji odabir u kojem se uvode sve spremljene procedure u bazi podataka za pozivanje naredbe u programskom alatu Visual Studio.

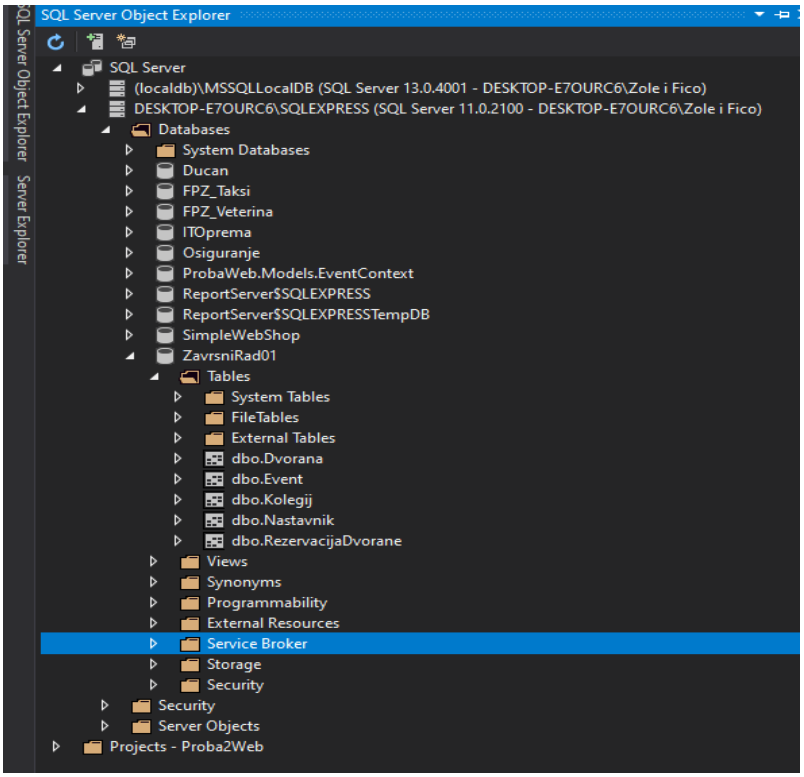
Dodavanjem preko add funkcije može se pokrenuti sučelje u kojem se preko SQL servera pristupa bazama podataka spremljenim na serveru. Pomoću SQL servera može se točno odabrati baza podataka i potrebni dijelovi koji se koriste za izradu modela koji služi za izmjenu podataka između aplikacije i baze podataka. Tako se jednostavno može povezati i uvijek imati spremljena konekcija.

Za korištenje baze podataka koristi se connection string koji se napravi unutar Web.config datoteke u web aplikaciji. Označava se sa <connectionString> i u njoj se definira ime konekcije, lokacija baze podataka, zaštita baze podataka i ime baze podataka. Primjer connection string-a se može vidjeti na slici 16.

```
data source=DESKTOP-LF99JDI;initial catalog=ZavrsniRad01;integrated security=True;
```

Slika 16. Prikaz connection string-a za kreiranje konekcije

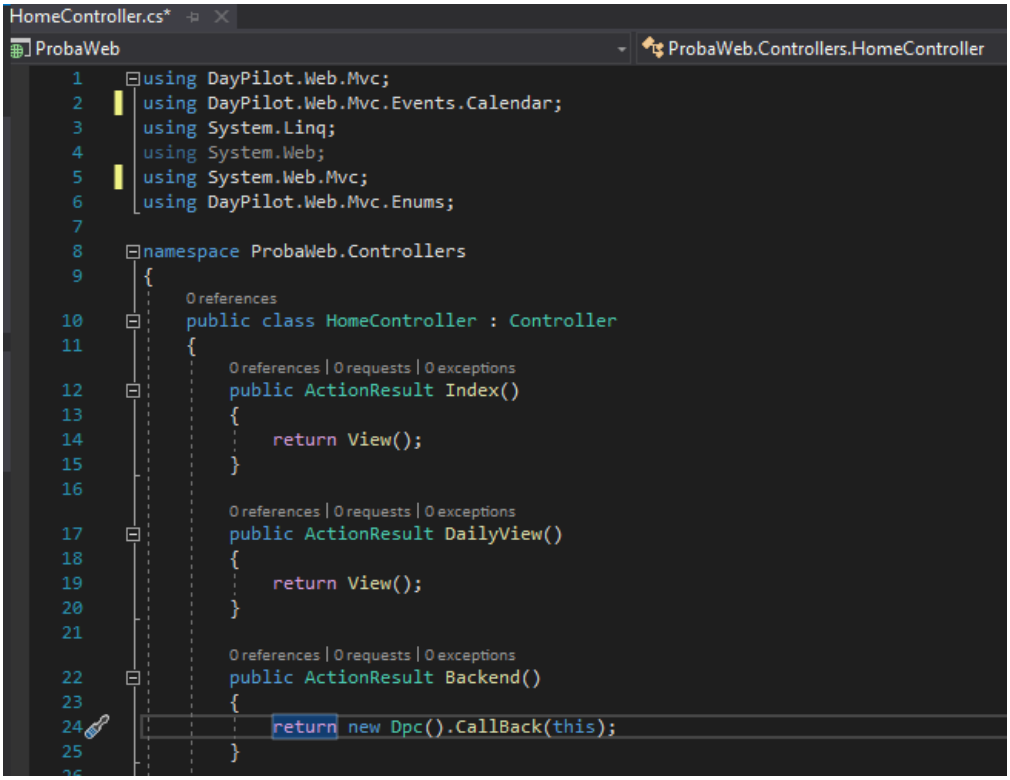
Nakon što je uspostavljena konekcija Visual Studio programskog alata sa serverom baze podataka može vrlo jednostavno pristupiti izrađenim bazama podataka preko SQL object server explorer-a gdje se može vidjeti struktura i podaci spremljeni u odabranoj bazi podataka kao što je prikazano na slici 17.



Slika 17. Prikaz SQL server object explorer

4.3. Izrada kontrolera u programskom alatu Visual Studio

Kontroleri su vrlo važan dio web aplikacija jer u programiranju web aplikacije oni odlučuju kako će program reagirati na određene događaje prilikom korištenja web aplikacije. Svaki kontroler je smješten pod određeni naziv koji služi za pozivanje metoda koje su određene u kontroleru. To služi da ukoliko postoji više kontrolora, preko naziva kontrolera moguće je jednostavno prebacivanje u pogledima iz jednog u drugi. Web aplikacije pozivaju pregled iz kontrolera po imenima koja su definirana žutom bojom na slici 15. Zelenom bojom su označene klase i radnje koje se događaju prilikom korištenja web aplikacije. Često korišteni kôd koji nema veliku važnost prikazan je plavom bojom. Ukoliko se aplikacija pokreće na *localhost*, da bi se prikazao sadržaj *Home* kontrolera potrebno je u pregledniku navigirati do *localhost/Home*, prilikom čega se predefinirano prikazuje pogled (stranica) *Index*. Metode kontrolera koje vraćaju pogled odgovaraju različitim stranicama koje taj kontroler može prikazati. Primjerice metoda *Index* vraća pogled koji najčešće odgovara početnoj stranici *Home* kontrolera u kojoj se prikazuju svi podaci, a navedena stranica se nalazi na putanji *localhost/Home/Index*. Prikaz kôda kontrolera može se vidjeti na slici 18.



Slika 18. Prikaz kontrolera

Može se definirati kako će program reagirati na komande korisnika kao na primjer događaj gdje korisnik klikom odvuče rezervaciju i time promjeni početak i kraj rezervacije. Pri tom događaju kontroler odlučuje što će se dogoditi u web aplikaciji. Na slici 19 može se jasno vidjeti kako je kôdirana metoda za navedeni događaj.

```
protected override void OnEventMove(EventMoveArgs e)
{
    var db = new ZavrsniRad01Entities1();
    int id = int.Parse(e.Id);
    var toBeResized = (from ev in db.Event where ev.id == id select ev).First();
    toBeResized.eventstart = e.NewStart;
    toBeResized.eventend = e.NewEnd;
    db.SaveChanges();
    Update();
}
```

Slika 19. Prikaz primjera metode za događaj prebacivanja rezervacije

Kao što je prikazano na slici vidi se da ukoliko korisnik odvuče rezervaciju da se kreira varijabla nazvana db koja je tip podataka klase napravljene prilikom povezivanja baze podataka sa Visual Studiom na modelu tablice u bazi podataka. Integer identifikator koristi za jedinstveno identificiranje koja se rezervacija ažurira u web aplikaciji. Kreirana varijabla toBeResized služi da se u nju spremi rezervacija iz baze koja će imati ažurirano vrijeme početka i kraja rezervacije, te dodatno da joj se ažuriraju vrijeme početka i kraja. Nakon toga se spremaju promjene i ažuriraju podaci u bazi podataka.

4.4. Izrada pogleda u programskom alatu Visual Studio

Pogled u programskom alatu visual Studio je prikaz podataka koje program koristi. Informacije su prikazane kako je određeno pri kreaciji web aplikacije, te se može koristiti gotov predložak u programskom alatu ili se kôdira zasebno. Koristi kombinaciju HTML, CSS i C# programskih jezika i pomoću tih programskih jezika može se kreirati jedinstven prikaz informacija. Korisniku se pregledom može pružiti jedinstven uvid u informacije kao na primjer informacije o dvoranama iz baze podataka koje bi ukoliko se ne koristi pregled bile ne čitljive za krajnjeg korisnika. Pogled koristi uglavnom HTML programski jezik, ali s može koristiti i C# dodavanjem @ znaka prije C# naredbe. Primjer kôda za izradu pogleda može se vidjeti na slici 20.

Create.cshtml*

```
@model ProbaWeb.Event

@{
    ViewBag.Title = "Create";
}

<h2>New Reservation</h2>

@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        <h4>Reservation</h4>
        <hr />
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
    <div class="form-group">

        <div class="col-md-10">
            @Html.DisplayName("Reservation name:")
            @Html.EditorFor(model => model.text, new { htmlAttributes = new { @class = "form-control" } })
            @Html.ValidationMessageFor(model => model.text, "", new { @class = "text-danger" })
        </div>
    </div>

        <div class="form-group">

            <div class="col-md-10">
                @Html.DisplayName("Start Date for Reservation:")
                @Html.EditorFor(model => model.eventstart, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.eventstart, "", new { @class = "text-danger" })
            </div>
        </div>
```

Slika 20. Prikaz kôda za izradu pogleda

5. Izrada web sučelja za rezervaciju i pregled korištenja dvorana

Pri izradi aplikacije prvo se mora uzeti u obzir namjena aplikacije, te se na bazi toga odlučiti koji je najbolji dizajn aplikacije za potrebe korisnika. Za demonstracijski sustav rezervacije dvorana odabrana je web aplikacija na MVC dizajnu jer pruža jednostavni prikaz podataka pohranjenih u bazi podataka preko web preglednika koji ima grafičko sučelje i može efikasno dohvaćati podatke za prikaz korisniku. Takvi programi su kôdirani u ASP.NET Core web aplikacijama. Dostupan odabrani predložak je ASP.NET Core Web Application koji može koristiti cloud infrastrukturu, C# programski jezik, Linux operativni sustav, macOS operativni sustav, web i Windows operativni sustav.

Nakon što je učitan predložak i implementirano povezivanje aplikacije s bazom podataka, potrebno je implementirati kontroler. Prilikom izrade potrebno je uzeti u obzir sve radnje koje će korisnik poduzimati prilikom korištenja web aplikacije, te na bazi toga se svaka radnja kôdira i to određuje kako će sustav svaku radnju obraditi.

Kontrolerom se određuje kako će sustav kreirati pregled koji se prikazuje korisniku i koje će informacije biti prikazane korisniku. Može se i odrediti hoće li varijabla koju sustav koristi biti dostupna korisniku za promjenu.

Na slici 21. može se vidjeti home kontroler koji određuje što će se prikazati korisniku nakon prijavljivanja u sustav i definira što će se dogoditi ukoliko je pokrenut određeni događaj od strane korisnika.

```
public class HomeController : Controller
{
    0 references
    public ActionResult Index()
    {
        return View();
    }


    0 references
    public ActionResult Backend()
    {
        return new Dp().CallBack(this);

    }

    0 references
    public ActionResult Create()
    {
        return View();
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    0 references
    public ActionResult Create([Bind(Include = "id,text,eventstart,eventend")] Event @event)
    {
        ZavrsniRad01Entities2 db = new ZavrsniRad01Entities2();
        if (ModelState.IsValid)
        {
            db.Events.Add(@event);
            db.SaveChanges();
            return RedirectToAction("Index");
        }

        return View(@event);
    }
```

Slika 21. Prikaz home kontrolera

Kontroler prikazan na slici 21. naziva se home kontroler i koristi se prilikom otvaranja web sučelja ili ukoliko je prikazan drugi pregled može se otvoriti pritiskom na naziv web aplikacije i potom na home. Odabrano ime pregleda je *Index* koje označava glavni prikaz u home kontroleru. Sva imena pregleda i stranica koje se prikazuju moguće je definirati u web aplikaciji. Home kontroler sadrži još naredbu poput create koja služi da se kreira nova rezervacija u kojoj se određuje ime rezervacije, početak i kraj rezervacije. Kreiranje se prikazuje u pogledu koji je kreiran za izradu nove rezervacije. Nakon napravljene nove rezervacije poziva se http post metoda istog naziva create koja koristi klasu za konekciju i jednostavnu if petlju s uvjetom da je upis točan koja će ako je uvjet istinit spremiti novu rezervaciju u tablicu Event i spremiti promjene, te preusmjeriti korisnika na glavni prikaz.

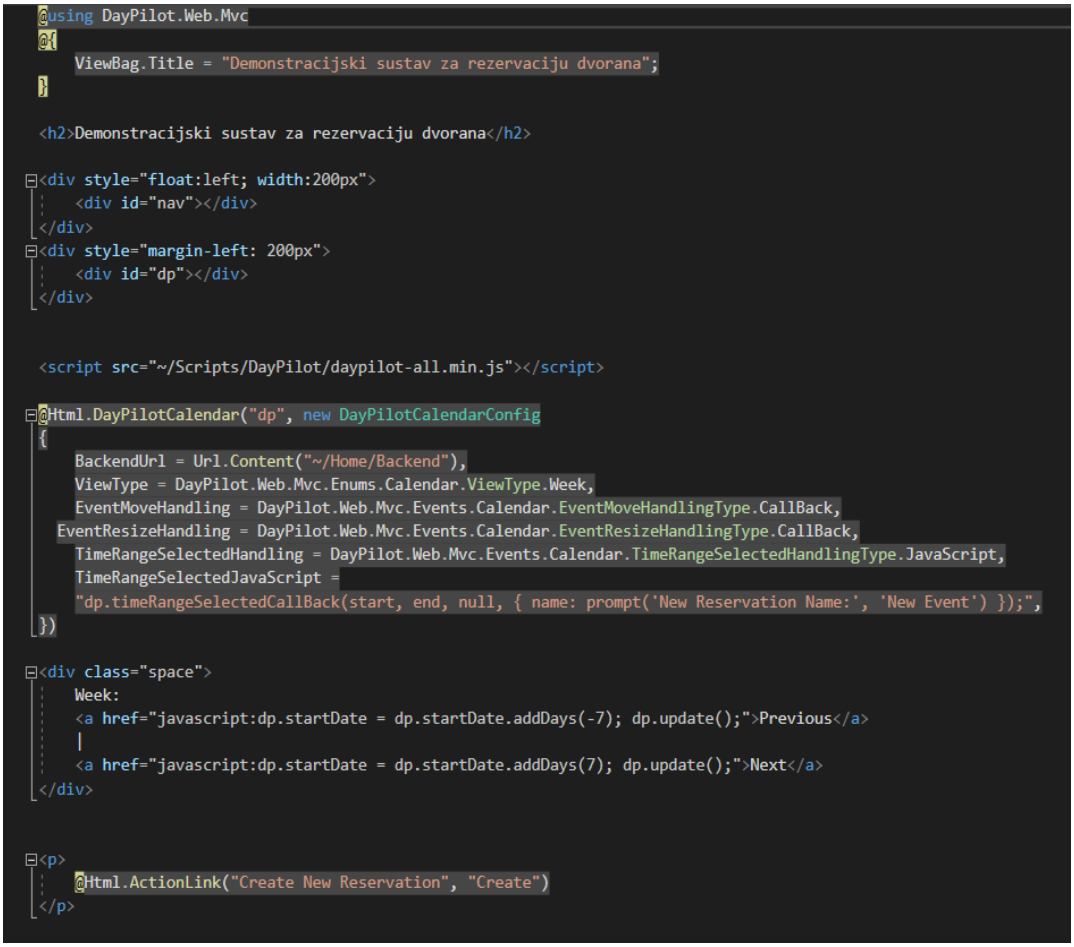
Uz navedene metode unutar home kontrolera nalazi se i klasa day pilot calendar koja sadrži metode za inicijalizaciju kalendara (OnInit), kod produživanja ili skraćivanja trajanja rezervacije (OnEventResize) i pri pomicanju pritiskom miša na rezervaciju koja se želi promijeniti (OnEventMove).

Pri inicijalizaciji kalendara se komandom update dohvate potrebne informacije o rezervacijama za prikaz glavnog kalendara. Kod promjene trajanja rezervacije prvo se pronalazi identifikator rezervacije pomoću kojeg se može promijeniti informacija u bazi podataka. Definira se varijabla (toBeResized) koja služi za spremanje novog početka i kraja rezervacije, te se nakon toga spremaju promjene u bazi podataka. Kod pomicanja rezervacije na drugi datum koristi se metoda OnEventMove koja radi na isti način kao i metoda OnEventResize. Primjer kôda klase day pilot calendar unutar home kontrolera može se vidjeti na slici 22.

```
1 reference
public class Dp : DayPilotCalendar
{
    ZavrsniRad01Entities2 db = new ZavrsniRad01Entities2();


    0 references
    protected override void OnInit(InitArgs e)
    {
        Update(CallBackUpdateType.Full);
    }


    0 references
    protected override void OnEventResize(EventResizeArgs e)
    {
        var db = new ZavrsniRad01Entities2();
        int id = int.Parse(e.Id);
        var toBeResized = (from ev in db.Events where ev.id == id select ev).First();
        toBeResized.eventstart = e.NewStart;
        toBeResized.eventend = e.NewEnd;
        db.SaveChanges();
        Update();
    }
    0 references
    protected override void OnEventMove(EventMoveArgs e)
    {
        var db = new ZavrsniRad01Entities2();
        int id = int.Parse(e.Id);
        var toBeResized = (from ev in db.Events where ev.id == id select ev).First();
        toBeResized.eventstart = e.NewStart;
        toBeResized.eventend = e.NewEnd;
        db.SaveChanges();
        Update();
    }
```

Slika 22. Prikaz klase day pilot calendar u home kontroleru

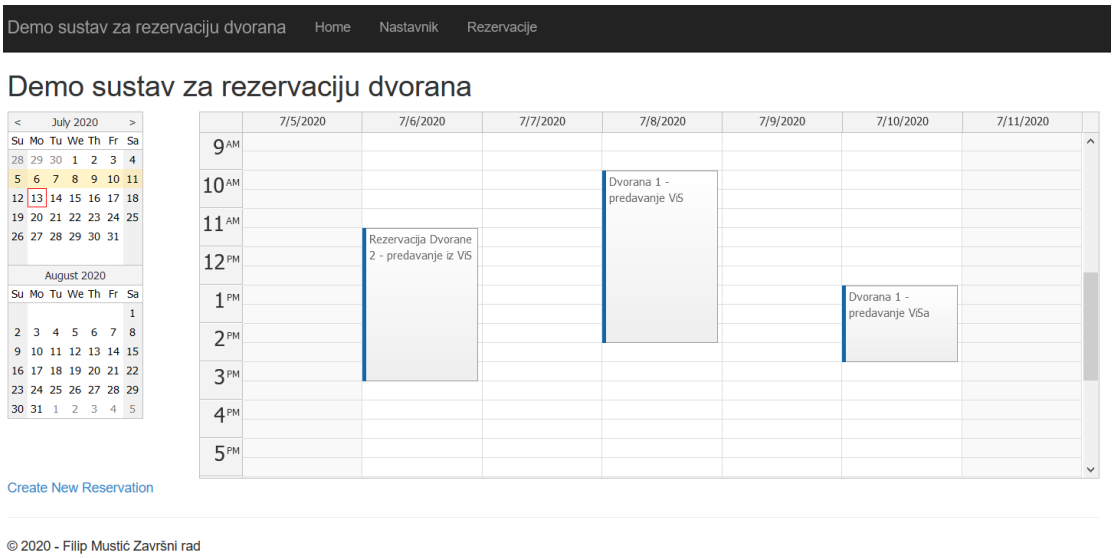
Nakon izrade kontrolera potrebno je izraditi i pogled koji će se prikazati korisniku kada pristupi web aplikaciji na određenoj web adresi. Kod pregleda gdje se može vidjeti osnovni dio pogleda index. Najprije se definira klasa koju pogled koristi, te naziv pogleda. Nakon toga unutar se definira prikaz navigator kalendara i glavnog kalendara. Još se i definira lokacija skripte koju će pogled koristiti. Korištenje html helper-a se koristi unutar C# kôda koji služi za definiranje kod kojeg događaja će se koja metoda koristiti iz home kontrolera i na koji način se definira metoda. Također može se vidjeti kôd koji se koristi da se korisniku omogući jednostavan prijelaz u prošli ili sljedeći tjedan za prikaz na glavnom kalendaru. Može se i vidjeti komanda koja kreira link koji vodi korisnika na pregled za kreiranje rezervacije Opisan kôd može se vidjeti na slici 23.



```
@using DayPilot.Web.Mvc
@{
    ViewBag.Title = "Demonstracijski sustav za rezervaciju dvorana";
}

<h2>Demonstracijski sustav za rezervaciju dvorana</h2>

<div style="float:left; width:200px">
    <div id="nav"></div>
</div>
<div style="margin-left: 200px">
    <div id="dp"></div>
</div>


<script src="~/Scripts/DayPilot/daypilot-all.min.js"></script>

@Html.DayPilotCalendar("dp", new DayPilotCalendarConfig
{
    BackendUrl = Url.Content("~/Home/Backend"),
    ViewType = DayPilot.Web.Mvc.Enums.Calendar.ViewType.Week,
    EventMoveHandling = DayPilot.Web.Mvc.Events.Calendar.EventMoveHandlingType.CallBack,
  EventResizeHandling = DayPilot.Web.Mvc.Events.Calendar.EventResizeHandlingType.CallBack,
    TimeRangeSelectedHandling = DayPilot.Web.Mvc.Events.Calendar.TimeRangeSelectedHandlingType.JavaScript,
    TimeRangeSelectedJavaScript =
    "dp.timeRangeSelectedCallBack(start, end, null, { name: prompt('New Reservation Name:', 'New Event') });",
})

<div class="space">
    Week:
    <a href="javascript:dp.startDate = dp.startDate.addDays(-7); dp.update();">Previous</a>
    |
    <a href="javascript:dp.startDate = dp.startDate.addDays(7); dp.update();">Next</a>
</div>


<p>
    @Html.ActionLink("Create New Reservation", "Create")
</p>
```

Slika 23. Prikaz pogleda Index

Rezultat pogleda koji se generira na temelju opisanog kôda može se vidjeti na slici 24.



Slika 24. Prikaz indeks stranice

Pregled koji je izrađen za rezerviranje dvorana poziva se na instaliranu klasu koju Visual Studio koristi preko dodane reference koja sadrži klasu day pilot calendar. Day pilot calendar sadrži definirane komande koje određuju što se događa kod određenog događaja pri korištenju web aplikacije i zbog toga može se vrlo jednostavno napraviti osnovne funkcije kalendara. Osim toga koristi se i html naredba za skriptu koja služi web aplikaciji da generira dodatni prikaz mjeseca i dana u mjesecu. On služi da korisnik ima pregled u trenutni i sljedeći mjesec. Također označen je i tjedan koji se trenutno prikazuje i za koji se mogu vidjeti rezervacije.

Pritiskom miša na označeni tjedan pomiče prikaz na odabrani tjedan, te omogućuje da se vide rezervacije prije aktualnog tjedna ili da se unaprijed mogu napraviti rezervacije. Skripta funkcionira na način da kada se želi promijeniti pregled tjedna na drugi tjedan onda se u klasi napravljenoj za day pilot calendar nazvanoj dp pomoću komande dp.startDate mijenja aktivni tjedan prikaza.

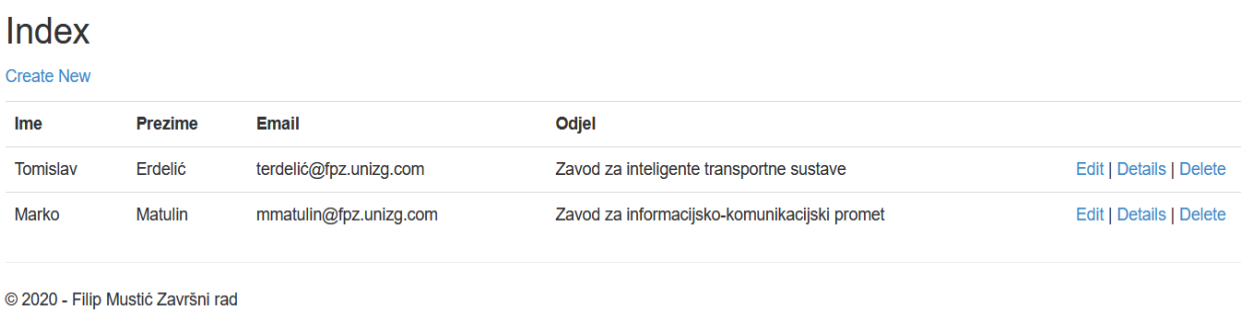
Definirana je varijabla navigator koja služi da se napravi prikaz trenutnog i prošlog mjeseca pored glavnog kalendara rezervacija. On kako je već navedeno služi da se može mijenjati tjedan koji se prikazuje na glavnom kalendaru. Pomoću navigatora može se definirati koliko će se prikazati mjeseci i koje vremensko razdoblje se može odabrati za prikaz. Klikom miša može se na navigator kalendaru odabrati tjedan koji se želi prikazati te se tada odabrani tjedan sprema u klasu scheduler-a novi početak tjedna i dane koji će se prikazati.

Uz to još se definira varijabla scheduler koja služi za označavanje prikaza po tjednu i označavanja tjedna koji će se prikazati. Definirana klasa sprema početni dan koji će se prikazati u tjednom prikazu, te koliko dana će se prikazati. Još se može mijenjati veličina pojedinog dana u glavnom kalendaru. JavaScript kôd može se vidjeti na slici 25.

```
<script type="text/javascript">
    var nav = new DayPilot.Navigator("nav");
    nav.cssClassPrefix = "navigator_8";
    nav.showMonths = 2;
    nav.selectMode = "week";
    nav.onTimeRangeSelected = function (args) {
        dp.startDate = args.start;
        dp.days = args.days;
        // load event to dp.events.list from the server
        dp.update();
    };

    nav.init();

    var dp = new DayPilot.Scheduler("dp");

    // view
    dp.startDate = nav.selectionStart;
    dp.cellGroupBy = "Week";
    dp.days = DayPilot.Date.daysDiff(nav.selectionStart, nav.selectionEnd);
    dp.cellDuration = 1440; // one day

    dp.init();

</script>
```

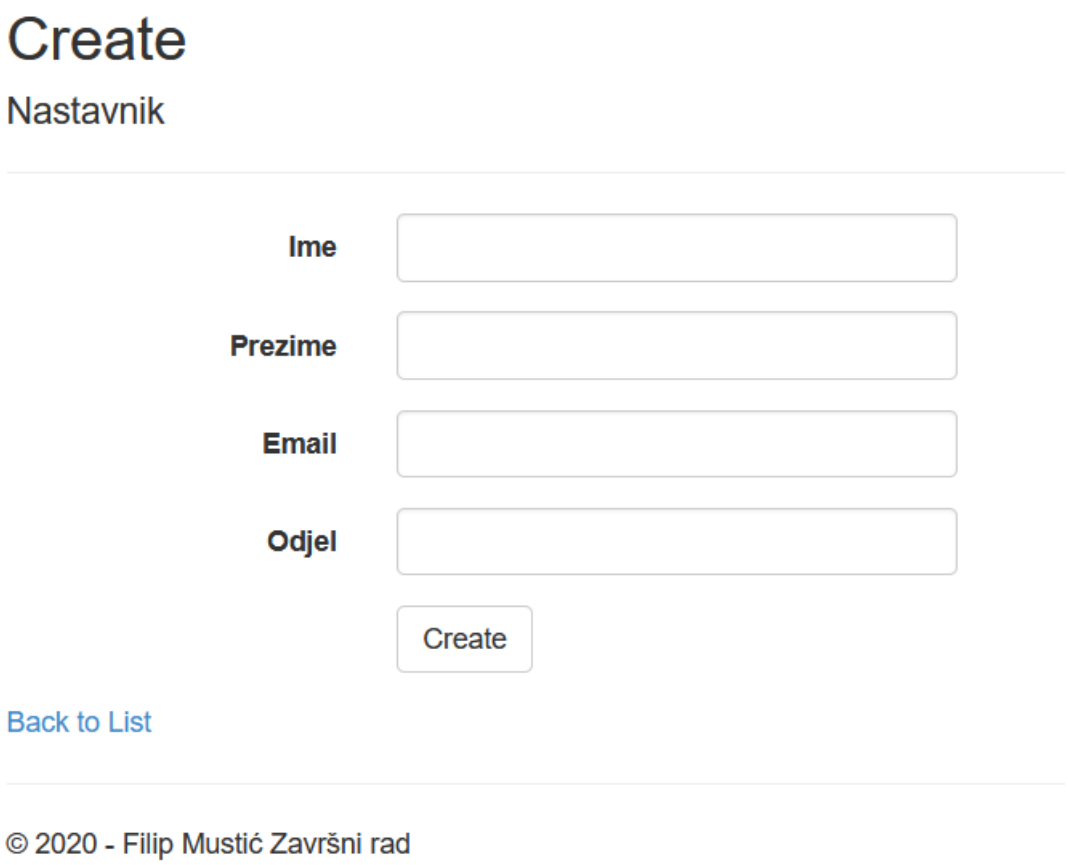
Slika 25. JavaScript kôd za promjenu prikazanog tjedna

Web aplikacija ima više pogleda koji se mogu koristiti. Ostali pogledi u web aplikaciji su generirani automatski na bazi klasa koje su prikazane poput pregleda koji prikazuje listu nastavnika i rezervacija. Pregledu nastavnika, kao i rezervacija može se pristupiti iz menija koji je dostupan na vrhu web aplikacije. U pregledu nastavnik mogu se vidjeti podaci o svakom nastavniku, te se mogu kreirati novi nastavnici, brisati nastavnici ili se postojećim nastavnicima mogu ažurirati podaci. Prikaz nastavnika u web sučelju preko web browsera Mozilla Firefox može se vidjeti na slici 26.



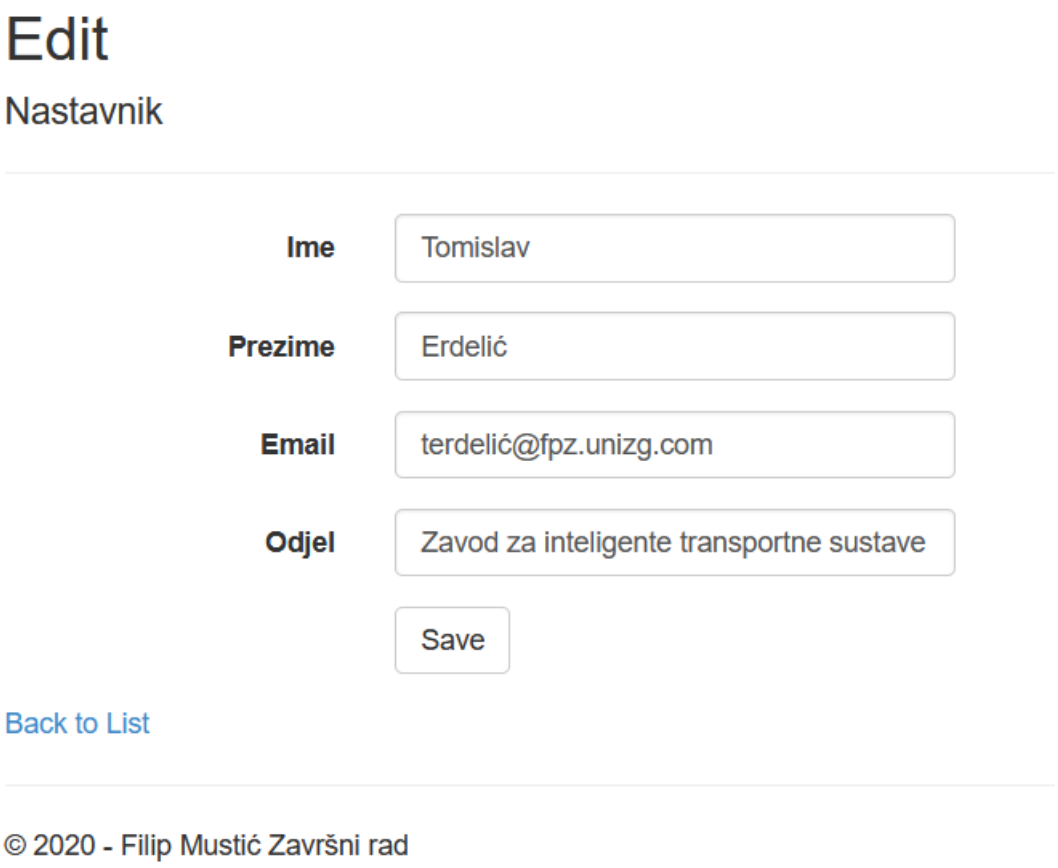
Slika 26. Prikaz pogleda nastavnik

Pri kreiranju novog nastavnika odabire se create new opcija koja omogućuje unos informacija o novom nastavniku. Sustav sam pazi na točan unos informacija jer se informacije nakon pritiska na create šalju u bazu podataka. Prikaz pogleda za dodavanje novog nastavnika može se vidjeti na slici 27.



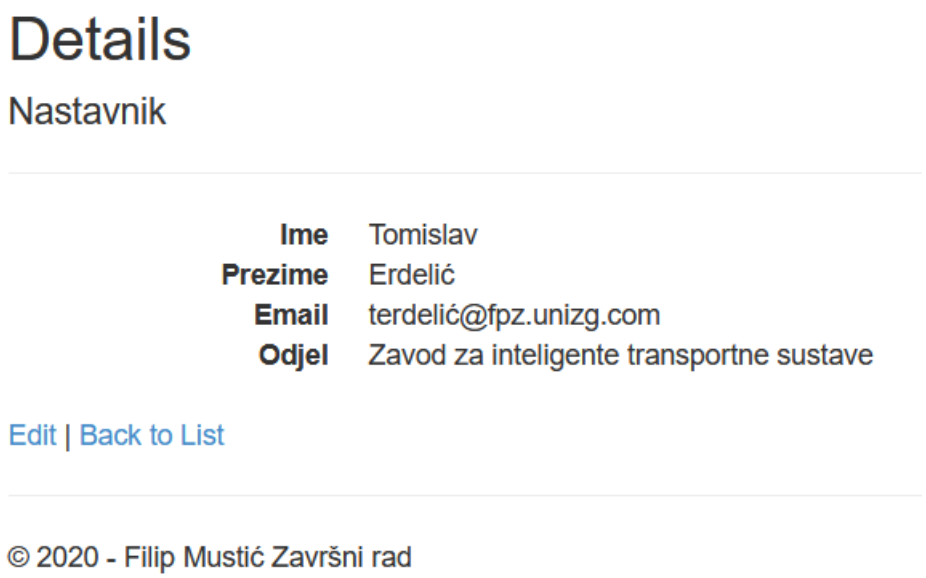
Slika 27. Prikaz pogleda za dodavanje novog nastavnika

Pri odabiru opcije edit otvara se pogled za ažuriranje informacija o nastavniku, te se mogu promijeniti atributi ime, prezime i email. Nakon promjene svih željenih informacija spremaju se pritiskom miša na save. Primjer pogleda edit može se vidjeti na slici 28.



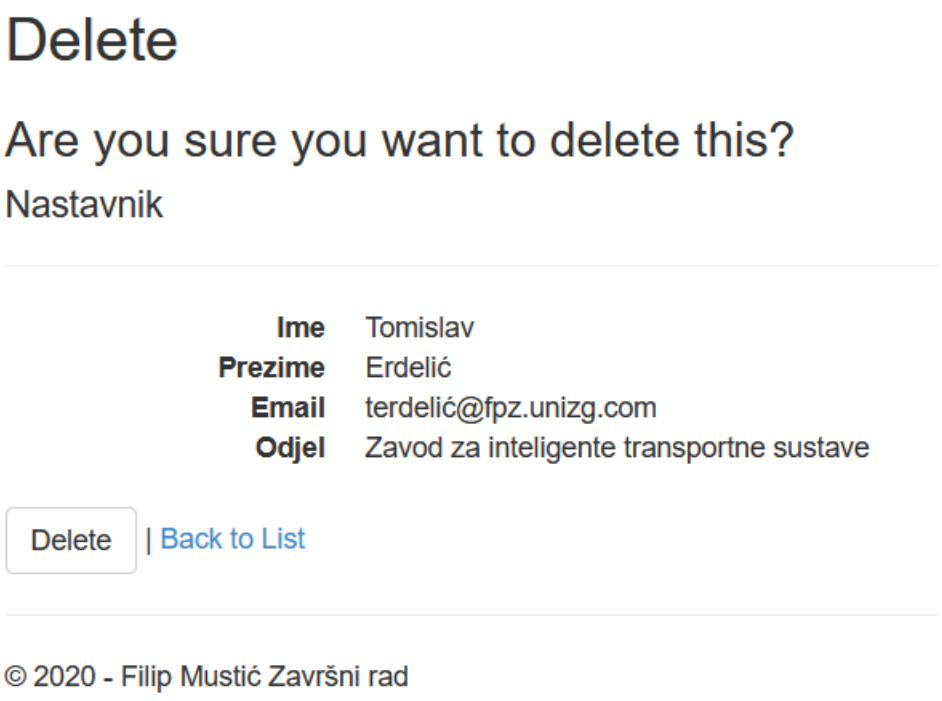
Slika 28. Prikaz pogleda edit za promjenu podataka nastavnika

Pri odabiru opcije details otvara se pogled koji prikazuje sve informacije o odabranom nastavniku osim njegovog identifikatora koji se ne prikazuje. On služi ukoliko je potrebna specifična informacija o nastavniku. Primjer se može vidjeti na slici 29.



Slika 29. Prikaz detalja o nastavniku

Pri odabiru opcije delete otvara se pogled koji dodatno provjerava želi li se odabrani nastavnik izbrisati sa popisa kao što se može vidjeti na slici 30. Iz pogleda se može izaći bez brisanja pritiskom na poveznicu *Back to list*.



Slika 30. Prikaz pogleda za brisanje nastavnika

Pregled rezervacija i dvorana služi ukoliko je potrebno vidjeti prikaz svih rezervacija ili dvorana u posebnom sučelju, ali također se mogu kreirati nove, ažurirati i brisati postojeće rezervacije, kao i dvorane u korisničkom sučelju koje prikazuje i funkcionira vrlo slično prethodnom primjeru nastavnika.

Prije korištenja web sučelja korisnici prvo se prijavljuju u sustav pomoću izrađenog sučelja za prijavu. Koristi se jednostavna prijava pomoću korisničkog imena i lozinke. Prijava je izrađena tako da uspoređuje unesene korisničke podatke za prijavljivanje i uspoređuje ih sa podacima u bazi podataka tablice loginTbl. Prikaz korisničkog sučelja za prijavljivanje može se vidjeti na slici 31.

Slika 31. Prikaz korisničkog sučelja za prijavu

6. Zaključak

Cilj rada je napraviti relacijsku bazu podataka koja sprema sve potrebne podatke za rezervaciju dvorana. Kod velikog broja korisnika i dvorana vrlo je teško pratiti zauzetost pojedine dvorane. Relacijska baza podataka koja sprema sve potrebne podatke odličan je izbor kod kojeg se može vrlo jednostavno kreirati odnos između pojedinih objekata i dohvatiti svaki potreban podatak o rezervaciji.

Uz relacijsku bazu podataka potrebno je izraditi web sučelje za prikaz rezerviranih dvorana. Kontroleri su kôdirani u C# programskom jeziku dok je pregled i izgled web sučelja kôdiran u HTML i CSS programskim jezicima. Kod događaja pri korištenju aplikacije koriste se .js (JavaScript) datoteke koje sadrže naredbe koje izvršava web aplikacija.

Web aplikacija napravljena je prema MVC predlošku namijenjenom za prikaz podataka u web pregledniku koje odgovara potrebama prikaza, unosa, ažuriranja i brisanja podataka iz relacijske baze. Za autorizaciju i autentifikaciju korisnika koristi se stranica za prijavu u sustav tako da se web aplikacija zaštiti od neovlaštenog pristupa informacijama koje sustav koristi. Prijavljivanje u sustav je omogućeno pomoću baze podataka u kojoj se nalaze korisničke informacije za prijavljivanje u sustav.

Web aplikacija koristi više kontrolera kojima je određeno što se događa pri korištenju aplikacije. Glavni prikaz se sastoji od glavnog kalendara na kojemu se mogu pregledati rezervacije, uređivati napravljene rezervacije, te unositi nove rezervacije u posebnom pogledu. Pomoću glavnog izbornika korisnik može otvarati druge poglede koji koriste različite kontrolere tako da su sve informacije dostupne korisniku. Ostali pregledi se koriste za prikaz informacija o rezervacijama, dvoranama i nastavnicima s ciljem da se omogući jedinstven i jednostavan pristup potrebnim informacijama za rezervaciju.

Pomoću korištenja ove web aplikacije organizaciji se omogućuje jednostavno upravljanje rezervacijama dvorane bez fizičkog pristupa dvoranama, te se spremaju sve potrebne informacije u bazu podataka.

Popis Literature

[1] Autorizirana predavanja iz kolegija Baze podataka: https://www.weboteka.net/fpz/Baze podataka/Predavanja/A05_auditorne.pdf (pristupljeno 2.8.2020.)

[2] W3Schools.com članak o tipovima podataka baze podataka https://www.w3schools.com/sql/sql_datatypes.asp (pristupljeno 7.8.2020)

[3] Autorizirana predavanja iz kolegija Baze podataka: https://moodle.srce.hr/2019-2020/pluginfile.php/2828420/mod_resource/content/4/7.tjedan.pdf (pristupljeno 12.7.2020.)

[4] Autorizirane laboratorijske vježbe iz kolegija Napredne baze podataka: https://moodle.srce.hr/2019-2020/pluginfile.php/3301523/mod_resource/content/2/Vje%C5%BEba13.pdf (pristupljeno 6.8.2020.)

[5] Autorizirana predavanja iz kolegija Baze podataka: https://moodle.srce.hr /2019-2020/pluginfile.php/2828395/mod_resource/content/6/3.tjedan.pdf (pristupljeno 12.7.2020)

[6] Autorizirana predavanja iz kolegija Baze podataka: https://moodle.srce.hr/2019-2020/pluginfile.php/2828400/mod_resource/content/6/4.tjedan.pdf (pristupljeno 13.7.2020)

Popis slika