|  |  |  |
| --- | --- | --- |
| ZnakUniverziteta | Univerzitet u Novom Sadu  Tehnički fakultet  »Mihajlo Pupin«  Zrenjanin | LOGOpupin |

***SEMINARSKI RAD***

*Predmet: Razvoj softvera otvorenog koda*

Tema: Avio karte

Predmetni nastavnik: Doc. dr Zoltan Kazi Studenti: Dadić Nikola SI 22/18

Radosavljević Jovan SI 4/18

Vlaović Milica SI 30/18

Zrenjanin, 2021. godina

**Sadržaj**

[1. Uvod 3](#_Toc74592851)

[2. Specifikacija zahteva korisnika 3](#_Toc74592852)

[2.1 Lista zahteva za funkcijama softvera 3](#_Toc74592853)

[2.1.1 Zahtev 1. – evidentiranje unetih karata i ostalih tabela aplikacije 4](#_Toc74592854)

[2.1.2 Zahtev 2. – pregled i pretraga unetih karata i ostalih tabela aplikacije 4](#_Toc74592855)

[2.1.3 Zahtev 3. – izmena karata i ostalih tabela aplikacije 5](#_Toc74592856)

[2.1.4 Zahtev 4. – brisanje karata i ostalih tabela aplikacije 5](#_Toc74592857)

[2.1.5 Zahtev 5. – prikaz unetih karata i ostalih tabela aplikacije 6](#_Toc74592858)

[2.1.6 Zahtev 6. – izveštaj za štampanje podataka tabela 6](#_Toc74592859)

[3. Faze razvoja softvera 7](#_Toc74592860)

[3.1 Projektne ideje 7](#_Toc74592861)

[3.2.1 Faze razvoja softvera 8](#_Toc74592862)

[3.2.2 Sloj baze podataka: Konceptualni model baze podataka (BP, DB) 8](#_Toc74592863)

[3.2.3 Fizički model baze podataka (Physical data model, RDM/RM) 9](#_Toc74592864)

[3.2.4 Objektno orjentisani model (Class diagram): 9](#_Toc74592865)

[3.2.5 Dijagram slučajeva korišćenja – softverske funkcije 10](#_Toc74592866)

[3.2.6 Koncept izgleda osnovnog ekrana softvera 11](#_Toc74592867)

[4. Prikaz stranice 12](#_Toc74592868)

[5. Prikaz realizacije i imeplementacije 17](#_Toc74592869)

[5.1 Prikaz izvornog koda 28](#_Toc74592870)

[6. Rad sa GitHub sistemom 58](#_Toc74592871)

[7. Korišćeni alati i softveri 61](#_Toc74592872)

[8. Literatura 62](#_Toc74592873)

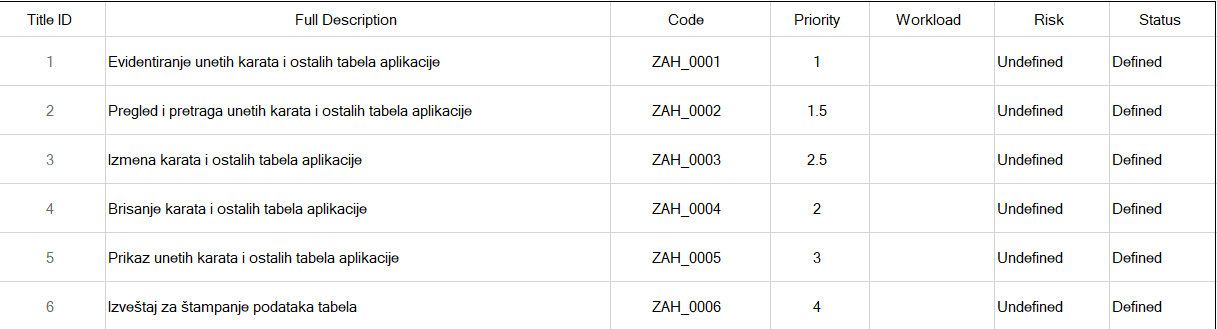
# **1. Uvod**

Seminarski rad iz predmeta „Razvoj softvera otvorenog koda“ bavi se temom izrade Web aplikacije za evidenciju avio karata. Aplikacija je zamišljena tako da tabelarno prikazuje, tj. da softver obuhvata sledeće:

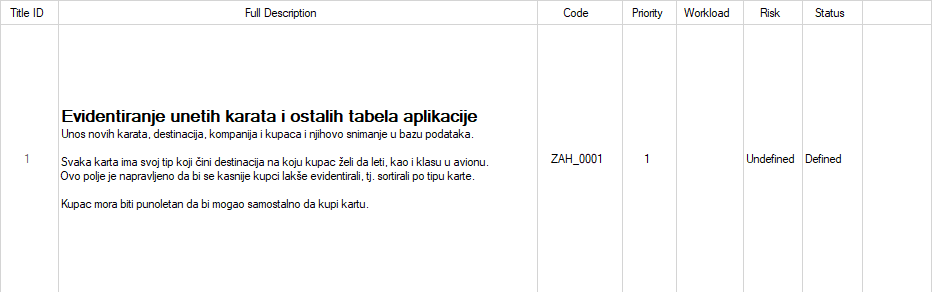
* unos i brisanje avio karata,
* unos i brisanje destinacija za letenje,
* unos i brisanje kompanija,
* unos i brisanje kupaca,
* kao i odabir neke od klasa letova (ekonomska, biznis ili prva klasa).

# **2. Specifikacija zahteva korisnika**

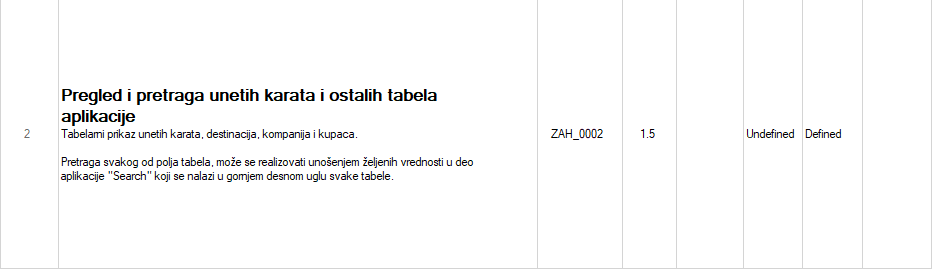
## **2.1 Lista zahteva za funkcijama softvera**



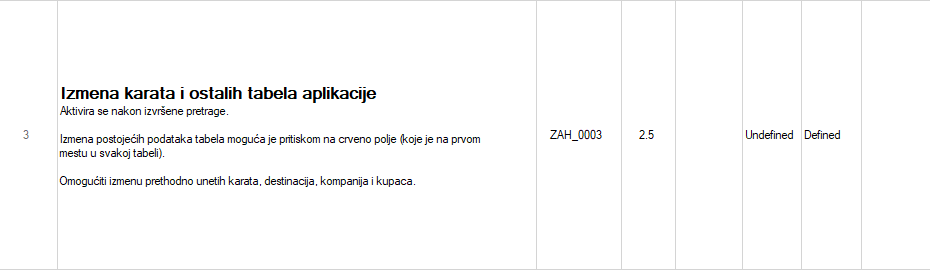
### **2.1.1 Zahtev 1. – evidentiranje unetih karata i ostalih tabela aplikacije**



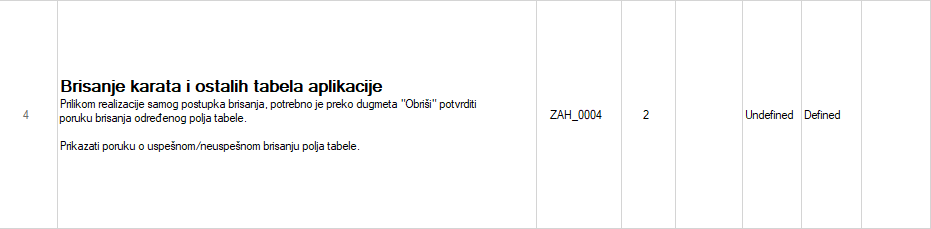
### **2.1.2 Zahtev 2. – pregled i pretraga unetih karata i ostalih tabela aplikacije**



### **2.1.3 Zahtev 3. – izmena karata i ostalih tabela aplikacije**



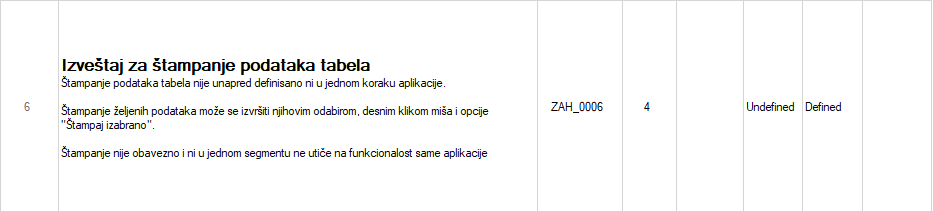
### **2.1.4 Zahtev 4. – brisanje karata i ostalih tabela aplikacije**



### **2.1.5 Zahtev 5. – prikaz unetih karata i ostalih tabela aplikacije**



### **2.1.6 Zahtev 6. – izveštaj za štampanje podataka tabela**



# **3. Faze razvoja softvera**

## **3.1 Projektne ideje**

* Osnovna funkcija softvera: Evidencija avio karata.
* Funkcije softvera: Unos novih karata, destinacija, kompanija i kupaca. Izmena i brisanje unetih polja u tabelama. Tabelarni prikaz svih unetih (naknadno izmenjenih) polja svake od tabela.
* Korisnici softvera:

1. Administrator – jedini koji ima pristup svim funkcijama softvera.
2. Korisnici koji nisu administratori – ima pristup tabelarnom prikazu podataka.

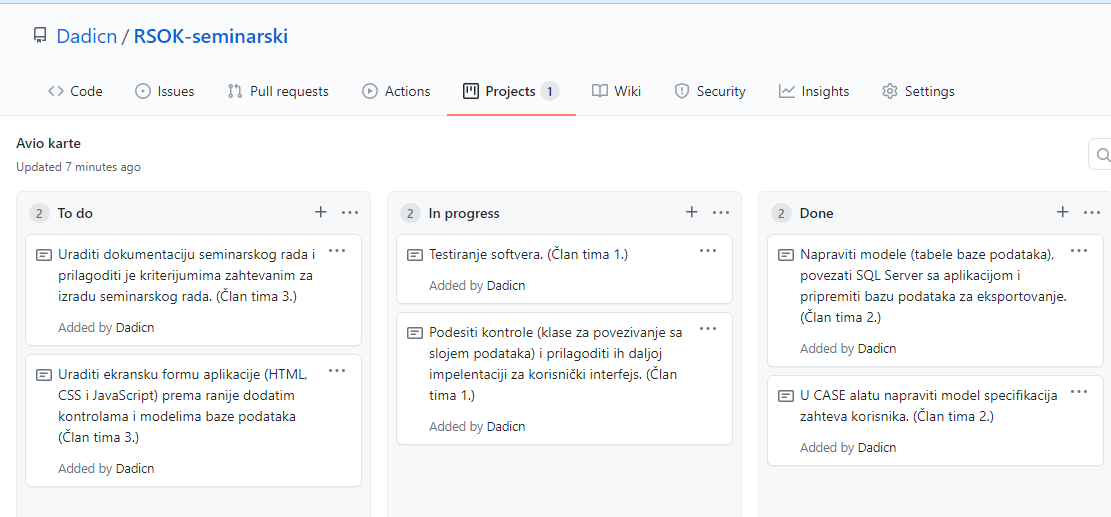
* Vrsta (tip) softvera: Web aplikacija.
* Softverska arhitektura: troslojna arhitektura softvera

1. Sloj podataka: Relaciona baza podataka, MySQL (dizajn: CASE alat).
2. Srednji aplikacioni sloj (backend): logika aplikacije, klase za povezivanje sa slojem podataka i KI (UI) C# programski jezik, alat: Visual Studio 2019.
3. Prezentacioni sloj (KI, UI, frontend): HTML, CSS, JS, C#, Bootstrap radni okvir (framework), alat: Visual Studio 2019.

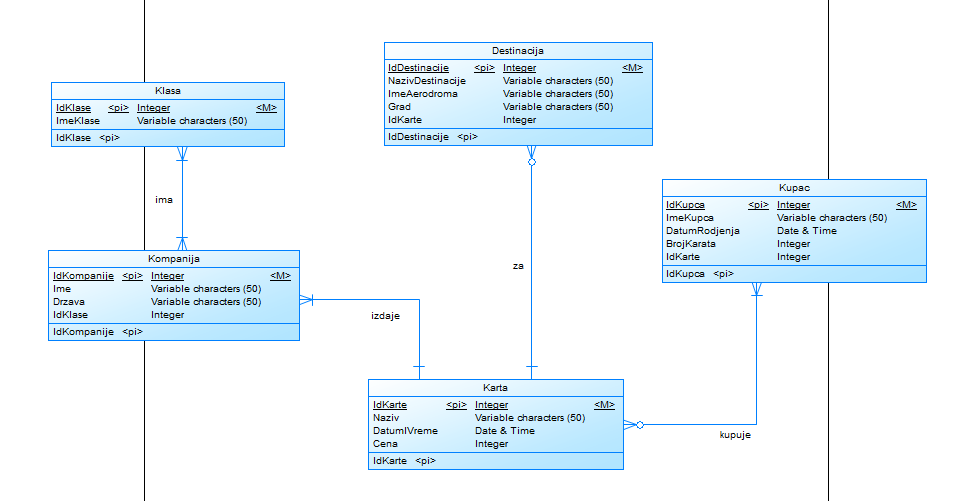
* Server: MySQL server (lokal)
* OS: MS Windows 10.
* Dokumentacija: MS Word.
* Testiranje: funckionalno testiranje.
* CVS sistem: GitHub.

**3.2 Dizajn razvoja softvera**

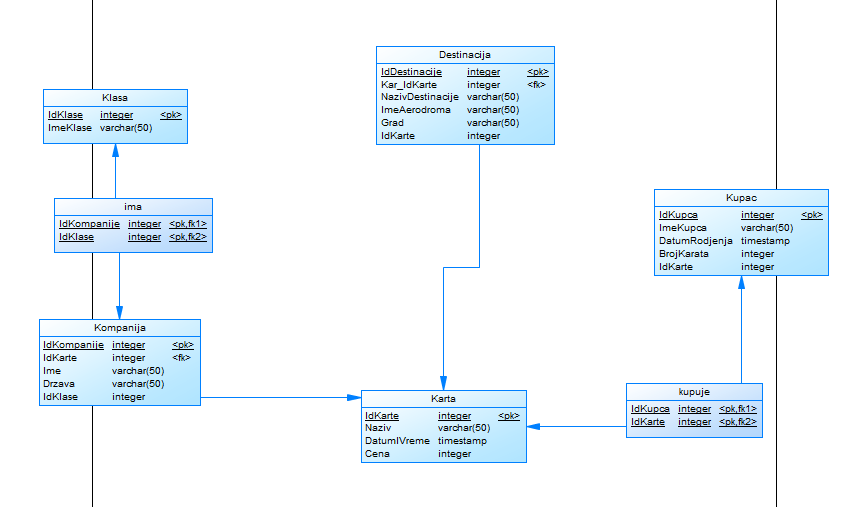
### **3.2.1 Faze razvoja softvera**



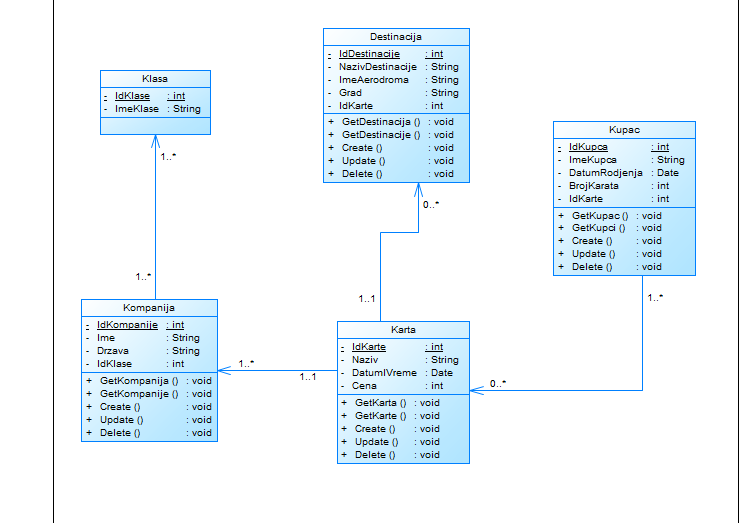
### **3.2.2 Sloj baze podataka: Konceptualni model baze podataka (BP, DB)**



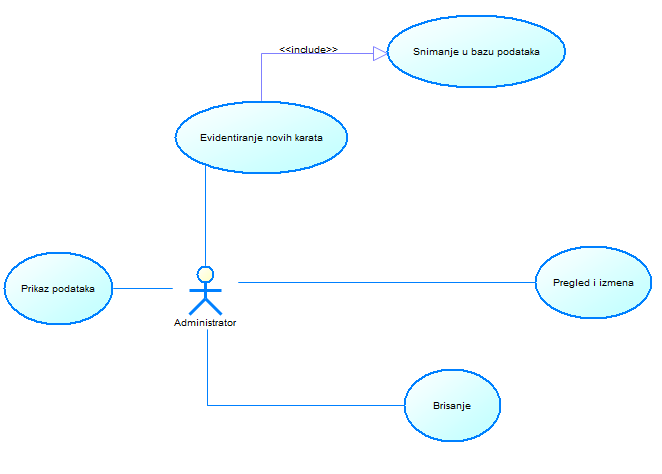
### **3.2.3 Fizički model baze podataka (Physical data model, RDM/RM)**



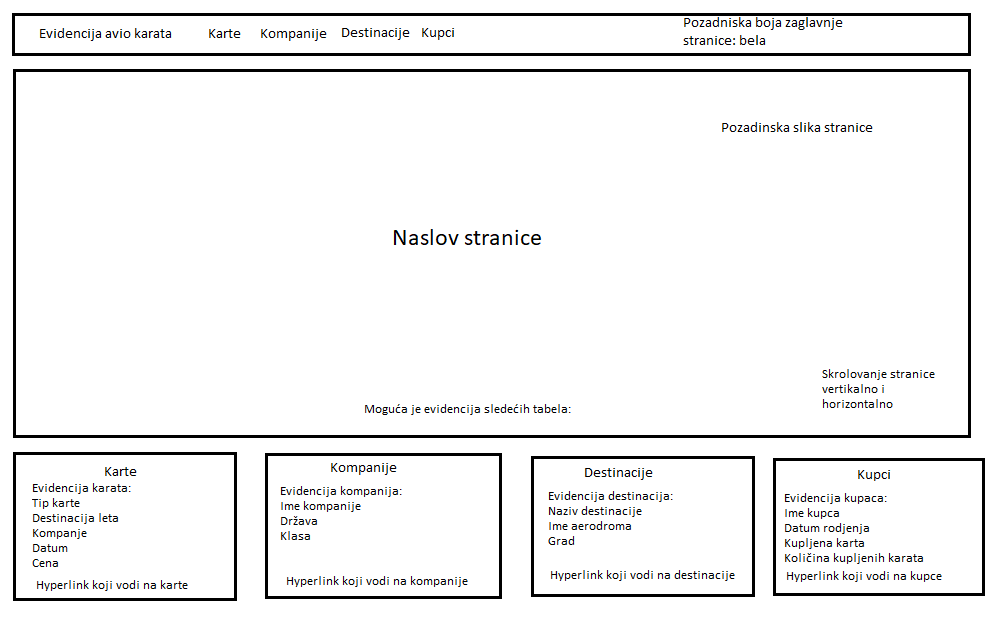
### **3.2.4 Objektno orjentisani model (Class diagram):**



### **3.2.5 Dijagram slučajeva korišćenja – softverske funkcije**



### **3.2.6 Koncept izgleda osnovnog ekrana softvera**



# **4. Prikaz stranice**

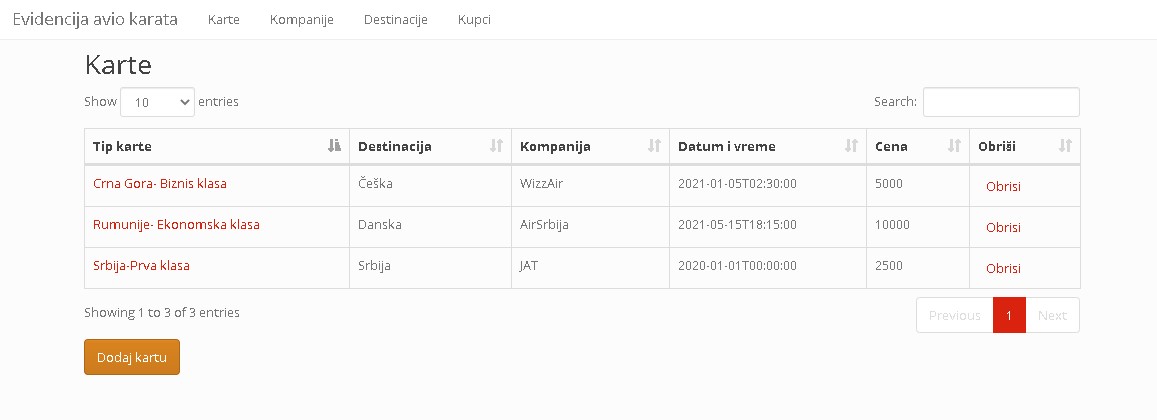
Kreiranje objekata baze podataka vrši se preko migracija u Visual Studiu 2019. i na taj način se i celokupna baza podataka pod nazivom „Evidencija avio karata“ i kreira.

Pokretanje aplikacije vrši se iz Vusial Studia 2019., ili ukucavanjem sledećeg u URL web browser-a:

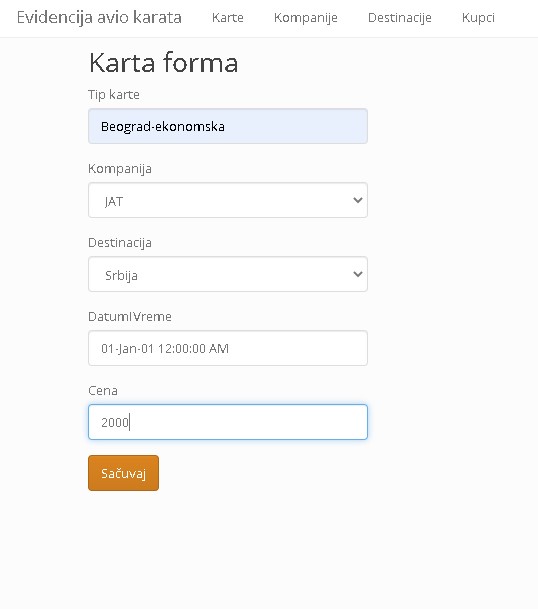
[*http://localhost:44324*](http://localhost:44324)



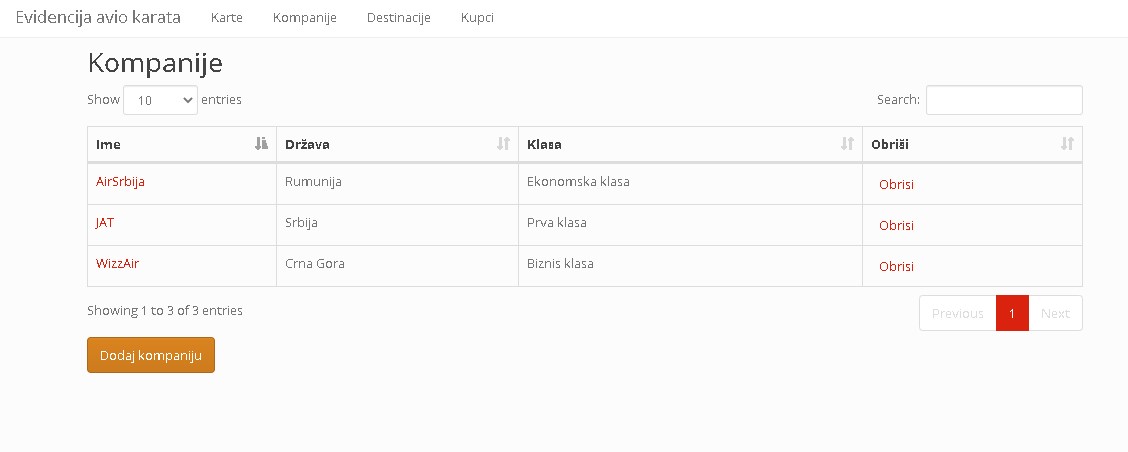
*Slika 1. – izgled početne stranice aplikacije*



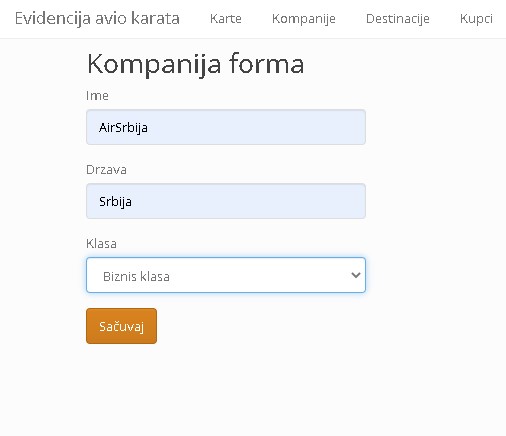
*Slika 2. – prikaz stranice „Karte“*



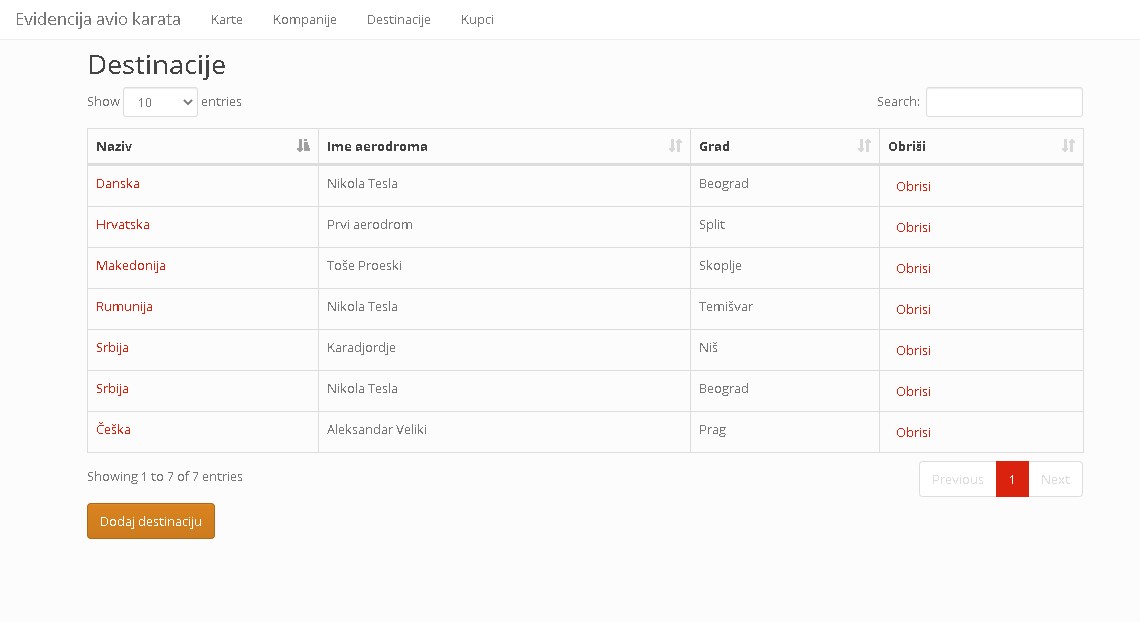
*Slika 3. – prikaz stranice za dodavanje karte*



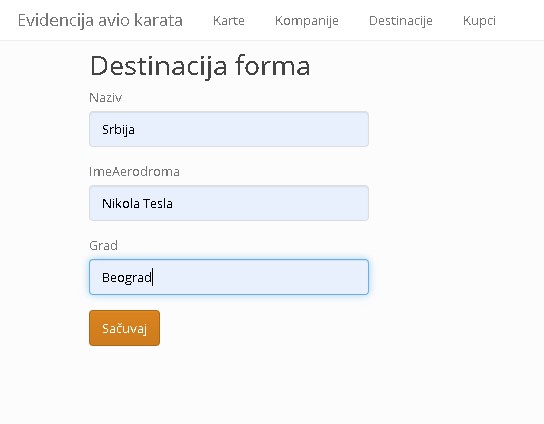
*Slika 3. – prikaz stranice „Kompanije“*



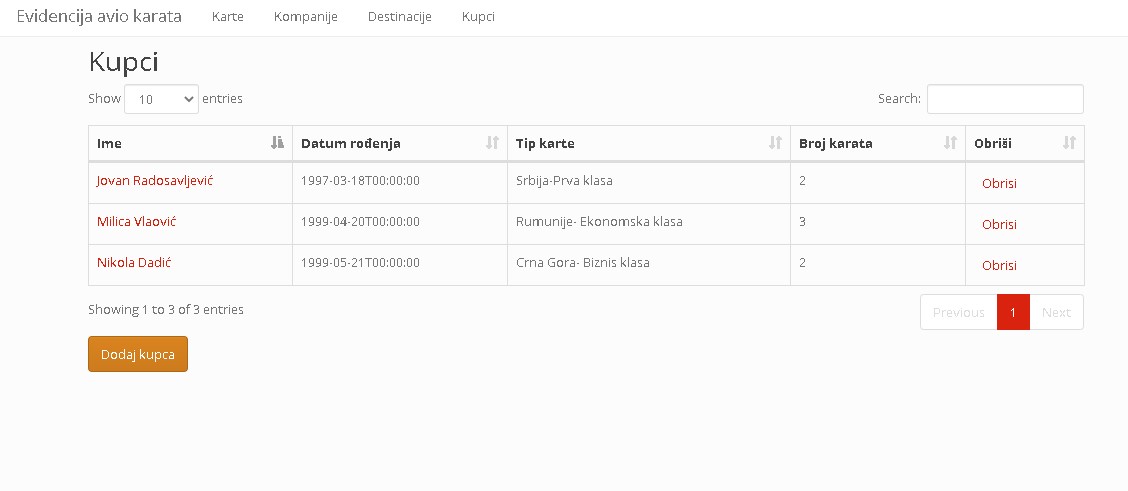
*Slika 4. – prikaz stranice za dodavanje kompanije*



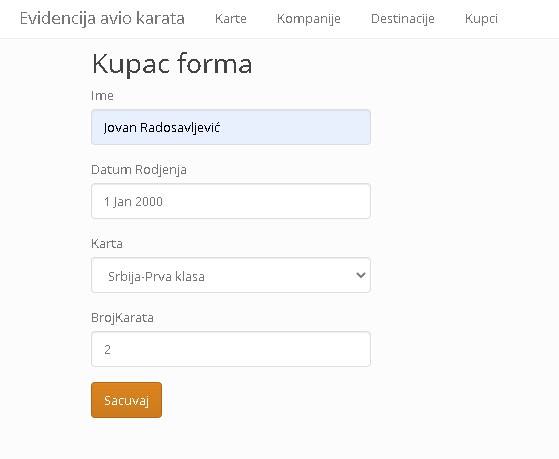
*Slika 5. – prikaz stranice „Destinacije“*



*Slika 6. – prikaz stranice za dodavanje destinacije*



*Slika 7. – prikaz stranice „Kupci“*



*Slika 8. – prikaz stranice za dodavanje kupca*

# **5. Prikaz realizacije i imeplementacije**

Kreiranje baze podataka izvršeno je preko migracija u Visual Studiu 2019. i na taj način je ona dodata na lokalni SQL server. Naziv baze je: *DefaultConnection*

SQL skript za kreiranje baze podataka:

USE [master]

GO

/\*\* Object: Database [DefaultConnection] Script Date: 14-Jun-21 5:55:35 PM \*\*/

CREATE DATABASE [DefaultConnection]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N'DefaultConnection', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS\MSSQL\DATA\DefaultConnection.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB )

LOG ON

( NAME = N'DefaultConnection\_log', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS\MSSQL\DATA\DefaultConnection\_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB )

WITH CATALOG\_COLLATION = DATABASE\_DEFAULT

GO

ALTER DATABASE [DefaultConnection] SET COMPATIBILITY\_LEVEL = 150

GO

IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))

begin

EXEC [DefaultConnection].[dbo].[sp\_fulltext\_database] @action = 'enable'

end

GO

ALTER DATABASE [DefaultConnection] SET ANSI\_NULL\_DEFAULT OFF

GO

ALTER DATABASE [DefaultConnection] SET ANSI\_NULLS OFF

GO

ALTER DATABASE [DefaultConnection] SET ANSI\_PADDING OFF

GO

ALTER DATABASE [DefaultConnection] SET ANSI\_WARNINGS OFF

GO

ALTER DATABASE [DefaultConnection] SET ARITHABORT OFF

GO

ALTER DATABASE [DefaultConnection] SET AUTO\_CLOSE ON

GO

ALTER DATABASE [DefaultConnection] SET AUTO\_SHRINK OFF

GO

ALTER DATABASE [DefaultConnection] SET AUTO\_UPDATE\_STATISTICS ON

GO

ALTER DATABASE [DefaultConnection] SET CURSOR\_CLOSE\_ON\_COMMIT OFF

GO

ALTER DATABASE [DefaultConnection] SET CURSOR\_DEFAULT GLOBAL

GO

ALTER DATABASE [DefaultConnection] SET CONCAT\_NULL\_YIELDS\_NULL OFF

GO

ALTER DATABASE [DefaultConnection] SET NUMERIC\_ROUNDABORT OFF

GO

ALTER DATABASE [DefaultConnection] SET QUOTED\_IDENTIFIER OFF

GO

ALTER DATABASE [DefaultConnection] SET RECURSIVE\_TRIGGERS OFF

GO

ALTER DATABASE [DefaultConnection] SET ENABLE\_BROKER

GO

ALTER DATABASE [DefaultConnection] SET AUTO\_UPDATE\_STATISTICS\_ASYNC OFF

GO

ALTER DATABASE [DefaultConnection] SET DATE\_CORRELATION\_OPTIMIZATION OFF

GO

ALTER DATABASE [DefaultConnection] SET TRUSTWORTHY OFF

GO

ALTER DATABASE [DefaultConnection] SET ALLOW\_SNAPSHOT\_ISOLATION OFF

GO

ALTER DATABASE [DefaultConnection] SET PARAMETERIZATION SIMPLE

GO

ALTER DATABASE [DefaultConnection] SET READ\_COMMITTED\_SNAPSHOT ON

GO

ALTER DATABASE [DefaultConnection] SET HONOR\_BROKER\_PRIORITY OFF

GO

ALTER DATABASE [DefaultConnection] SET RECOVERY SIMPLE

GO

ALTER DATABASE [DefaultConnection] SET MULTI\_USER

GO

ALTER DATABASE [DefaultConnection] SET PAGE\_VERIFY CHECKSUM

GO

ALTER DATABASE [DefaultConnection] SET DB\_CHAINING OFF

GO

ALTER DATABASE [DefaultConnection] SET FILESTREAM( NON\_TRANSACTED\_ACCESS = OFF )

GO

ALTER DATABASE [DefaultConnection] SET TARGET\_RECOVERY\_TIME = 60 SECONDS

GO

ALTER DATABASE [DefaultConnection] SET DELAYED\_DURABILITY = DISABLED

GO

ALTER DATABASE [DefaultConnection] SET ACCELERATED\_DATABASE\_RECOVERY = OFF

GO

ALTER DATABASE [DefaultConnection] SET QUERY\_STORE = OFF

GO

USE [DefaultConnection]

GO

/\*\* Object: Table [dbo].[\_\_MigrationHistory] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[\_\_MigrationHistory](

[MigrationId] [nvarchar](150) NOT NULL,

[ContextKey] [nvarchar](300) NOT NULL,

[Model] [varbinary](max) NOT NULL,

[ProductVersion] [nvarchar](32) NOT NULL,

CONSTRAINT [PK\_dbo.\_\_MigrationHistory] PRIMARY KEY CLUSTERED

(

[MigrationId] ASC,

[ContextKey] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\* Object: Table [dbo].[AspNetRoles] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetRoles](

[Id] [nvarchar](128) NOT NULL,

[Name] [nvarchar](256) NOT NULL,

CONSTRAINT [PK\_dbo.AspNetRoles] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\* Object: Table [dbo].[AspNetUserClaims] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUserClaims](

[Id] [int] IDENTITY(1,1) NOT NULL,

[UserId] [nvarchar](128) NOT NULL,

[ClaimType] [nvarchar](max) NULL,

[ClaimValue] [nvarchar](max) NULL,

CONSTRAINT [PK\_dbo.AspNetUserClaims] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\* Object: Table [dbo].[AspNetUserLogins] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUserLogins](

[LoginProvider] [nvarchar](128) NOT NULL,

[ProviderKey] [nvarchar](128) NOT NULL,

[UserId] [nvarchar](128) NOT NULL,

CONSTRAINT [PK\_dbo.AspNetUserLogins] PRIMARY KEY CLUSTERED

(

[LoginProvider] ASC,

[ProviderKey] ASC,

[UserId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\* Object: Table [dbo].[AspNetUserRoles] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUserRoles](

[UserId] [nvarchar](128) NOT NULL,

[RoleId] [nvarchar](128) NOT NULL,

CONSTRAINT [PK\_dbo.AspNetUserRoles] PRIMARY KEY CLUSTERED

(

[UserId] ASC,

[RoleId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\* Object: Table [dbo].[AspNetUsers] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUsers](

[Id] [nvarchar](128) NOT NULL,

[Email] [nvarchar](256) NULL,

[EmailConfirmed] [bit] NOT NULL,

[PasswordHash] [nvarchar](max) NULL,

[SecurityStamp] [nvarchar](max) NULL,

[PhoneNumber] [nvarchar](max) NULL,

[PhoneNumberConfirmed] [bit] NOT NULL,

[TwoFactorEnabled] [bit] NOT NULL,

[LockoutEndDateUtc] [datetime] NULL,

[LockoutEnabled] [bit] NOT NULL,

[AccessFailedCount] [int] NOT NULL,

[UserName] [nvarchar](256) NOT NULL,

CONSTRAINT [PK\_dbo.AspNetUsers] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\* Object: Table [dbo].[Destinacija] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Destinacija](

[Id] [int] IDENTITY(1,1) NOT NULL,

[Naziv] [nvarchar](max) NOT NULL,

[ImeAerodroma] [nvarchar](max) NOT NULL,

[Grad] [nvarchar](max) NOT NULL,

CONSTRAINT [PK\_dbo.Destinacija] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\* Object: Table [dbo].[Karta] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Karta](

[Id] [int] IDENTITY(1,1) NOT NULL,

[tipKarte] [nvarchar](max) NOT NULL,

[DestinacijaId] [int] NOT NULL,

[KompanijaId] [int] NOT NULL,

[DatumIVreme] [datetime] NOT NULL,

[Cena] [int] NOT NULL,

CONSTRAINT [PK\_dbo.Karta] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\* Object: Table [dbo].[Klasa] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Klasa](

[Id] [tinyint] NOT NULL,

[Naziv] [nvarchar](max) NULL,

CONSTRAINT [PK\_dbo.Klasa] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\* Object: Table [dbo].[Kompanija] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Kompanija](

[Id] [int] IDENTITY(1,1) NOT NULL,

[Ime] [nvarchar](max) NOT NULL,

[Drzava] [nvarchar](max) NOT NULL,

[KlasaId] [tinyint] NOT NULL,

CONSTRAINT [PK\_dbo.Kompanija] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\* Object: Table [dbo].[Kupci] Script Date: 14-Jun-21 5:55:35 PM \*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Kupci](

[Id] [int] IDENTITY(1,1) NOT NULL,

[Ime] [nvarchar](max) NOT NULL,

[DatumRodjenja] [datetime] NOT NULL,

[KartaId] [int] NOT NULL,

[BrojKarata] [int] NOT NULL,

CONSTRAINT [PK\_dbo.Kupci] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

SET ANSI\_PADDING ON

GO

/\*\* Object: Index [RoleNameIndex] Script Date: 14-Jun-21 5:55:36 PM \*\*/

CREATE UNIQUE NONCLUSTERED INDEX [RoleNameIndex] ON [dbo].[AspNetRoles]

(

[Name] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, IGNORE\_DUP\_KEY = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

GO

SET ANSI\_PADDING ON

GO

/\*\* Object: Index [IX\_UserId] Script Date: 14-Jun-21 5:55:36 PM \*\*/

CREATE NONCLUSTERED INDEX [IX\_UserId] ON [dbo].[AspNetUserClaims]

(

[UserId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

GO

SET ANSI\_PADDING ON

GO

/\*\* Object: Index [IX\_UserId] Script Date: 14-Jun-21 5:55:36 PM \*\*/

CREATE NONCLUSTERED INDEX [IX\_UserId] ON [dbo].[AspNetUserLogins]

(

[UserId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

GO

SET ANSI\_PADDING ON

GO

/\*\* Object: Index [IX\_RoleId] Script Date: 14-Jun-21 5:55:36 PM \*\*/

CREATE NONCLUSTERED INDEX [IX\_RoleId] ON [dbo].[AspNetUserRoles]

(

[RoleId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

GO

SET ANSI\_PADDING ON

GO

/\*\* Object: Index [IX\_UserId] Script Date: 14-Jun-21 5:55:36 PM \*\*/

CREATE NONCLUSTERED INDEX [IX\_UserId] ON [dbo].[AspNetUserRoles]

(

[UserId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

GO

SET ANSI\_PADDING ON

GO

/\*\* Object: Index [UserNameIndex] Script Date: 14-Jun-21 5:55:36 PM \*\*/

CREATE UNIQUE NONCLUSTERED INDEX [UserNameIndex] ON [dbo].[AspNetUsers]

(

[UserName] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, IGNORE\_DUP\_KEY = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

GO

/\*\* Object: Index [IX\_DestinacijaId] Script Date: 14-Jun-21 5:55:36 PM \*\*/

CREATE NONCLUSTERED INDEX [IX\_DestinacijaId] ON [dbo].[Karta]

(

[DestinacijaId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

GO

/\*\* Object: Index [IX\_KompanijaId] Script Date: 14-Jun-21 5:55:36 PM \*\*/

CREATE NONCLUSTERED INDEX [IX\_KompanijaId] ON [dbo].[Karta]

(

[KompanijaId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

GO

/\*\* Object: Index [IX\_KlasaId] Script Date: 14-Jun-21 5:55:36 PM \*\*/

CREATE NONCLUSTERED INDEX [IX\_KlasaId] ON [dbo].[Kompanija]

(

[KlasaId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

GO

/\*\* Object: Index [IX\_KartaId] Script Date: 14-Jun-21 5:55:36 PM \*\*/

CREATE NONCLUSTERED INDEX [IX\_KartaId] ON [dbo].[Kupci]

(

[KartaId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

GO

ALTER TABLE [dbo].[AspNetUserClaims] WITH CHECK ADD CONSTRAINT [FK\_dbo.AspNetUserClaims\_dbo.AspNetUsers\_UserId] FOREIGN KEY([UserId])

REFERENCES [dbo].[AspNetUsers] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserClaims] CHECK CONSTRAINT [FK\_dbo.AspNetUserClaims\_dbo.AspNetUsers\_UserId]

GO

ALTER TABLE [dbo].[AspNetUserLogins] WITH CHECK ADD CONSTRAINT [FK\_dbo.AspNetUserLogins\_dbo.AspNetUsers\_UserId] FOREIGN KEY([UserId])

REFERENCES [dbo].[AspNetUsers] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserLogins] CHECK CONSTRAINT [FK\_dbo.AspNetUserLogins\_dbo.AspNetUsers\_UserId]

GO

ALTER TABLE [dbo].[AspNetUserRoles] WITH CHECK ADD CONSTRAINT [FK\_dbo.AspNetUserRoles\_dbo.AspNetRoles\_RoleId] FOREIGN KEY([RoleId])

REFERENCES [dbo].[AspNetRoles] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserRoles] CHECK CONSTRAINT [FK\_dbo.AspNetUserRoles\_dbo.AspNetRoles\_RoleId]

GO

ALTER TABLE [dbo].[AspNetUserRoles] WITH CHECK ADD CONSTRAINT [FK\_dbo.AspNetUserRoles\_dbo.AspNetUsers\_UserId] FOREIGN KEY([UserId])

REFERENCES [dbo].[AspNetUsers] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserRoles] CHECK CONSTRAINT [FK\_dbo.AspNetUserRoles\_dbo.AspNetUsers\_UserId]

GO

ALTER TABLE [dbo].[Karta] WITH CHECK ADD CONSTRAINT [FK\_dbo.Karta\_dbo.Destinacija\_DestinacijaId] FOREIGN KEY([DestinacijaId])

REFERENCES [dbo].[Destinacija] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Karta] CHECK CONSTRAINT [FK\_dbo.Karta\_dbo.Destinacija\_DestinacijaId]

GO

ALTER TABLE [dbo].[Karta] WITH CHECK ADD CONSTRAINT [FK\_dbo.Karta\_dbo.Kompanija\_KompanijaId] FOREIGN KEY([KompanijaId])

REFERENCES [dbo].[Kompanija] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Karta] CHECK CONSTRAINT [FK\_dbo.Karta\_dbo.Kompanija\_KompanijaId]

GO

ALTER TABLE [dbo].[Kompanija] WITH CHECK ADD CONSTRAINT [FK\_dbo.Kompanija\_dbo.Klasa\_KlasaId] FOREIGN KEY([KlasaId])

REFERENCES [dbo].[Klasa] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Kompanija] CHECK CONSTRAINT [FK\_dbo.Kompanija\_dbo.Klasa\_KlasaId]

GO

ALTER TABLE [dbo].[Kupci] WITH CHECK ADD CONSTRAINT [FK\_dbo.Kupci\_dbo.Karta\_KartaId] FOREIGN KEY([KartaId])

REFERENCES [dbo].[Karta] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Kupci] CHECK CONSTRAINT [FK\_dbo.Kupci\_dbo.Karta\_KartaId]

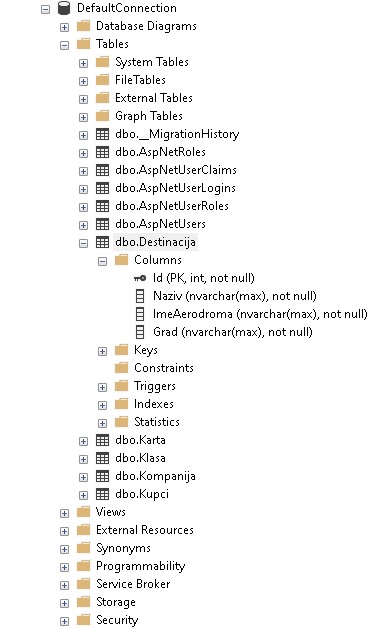
GO

USE [master]

GO

ALTER DATABASE [DefaultConnection] SET READ\_WRITE

GO



*Slika 9. – baza podataka u MYSQL Managment Studiu*

## **5.1 Prikaz izvornog koda**

Prvi segment u izradi aplikacije bio je kreiranje *Modela* tj. tabela baze podataka. Kreirane su sledeće tabele:

* Karta
* Kompanija
* Destinacija
* Kupac
* Klasa

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Web;

namespace EvidencijaAvioKarata.Models

{

[Table("Karta")]

public class Karta

{

public int Id { get; set; }

[Required]

[Display(Name = "Tip karte")]

public string tipKarte { get; set; }

public Destinacija Destinacija { get; set; }

[Required]

[Display(Name = "Destinacija")]

public int DestinacijaId { get; set; }

public Kompanija Kompanija { get; set; }

[Required]

[Display(Name = "Kompanija")]

public int KompanijaId { get; set; }

[Required]

public DateTime DatumIVreme { get; set; }

[Required]

public int Cena { get; set; }

}

}

Listing 1. – kreiranje tabele *Karta*

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Web;

namespace EvidencijaAvioKarata.Models

{

[Table("Kompanija")]

public class Kompanija

{

public int Id { get; set; }

[Required]

public string Ime { get; set; }

[Required]

public string Drzava { get; set; }

public Klasa Klasa { get; set; }

[Required]

[Display(Name = "Klasa")]

public byte KlasaId { get; set; }

}

}

Listing 2. – kreiranje tabele *Kompanija*

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Web;

namespace EvidencijaAvioKarata.Models

{

[Table("Destinacija")]

public class Destinacija

{

public int Id { get; set; }

[Required]

public string Naziv { get; set; }

[Required]

public string ImeAerodroma { get; set; }

[Required]

public string Grad { get; set; }

}

}

Listing 3. – kreiranje tabele *Destinacija*

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Web;

namespace EvidencijaAvioKarata.Models

{

[Table("Kupci")]

public class Kupac

{

public int Id { get; set; }

[Required]

public string Ime { get; set; }

[Required]

[PoslovnoPravilo]

[Display(Name = "Datum Rodjenja")]

public DateTime DatumRodjenja { get; set; }

public Karta Karta { get; set; }

[Required]

[Display(Name = "Karta")]

public int KartaId { get; set; }

[Required]

[Range(1, 5)]

public int BrojKarata { get; set; }

public int Godine = 18;

}

}

Listing 4. – kreiranje tabele *Kupac*

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Web;

namespace EvidencijaAvioKarata.Models

{

[Table("Klasa")]

public class Klasa

{

public byte Id { get; set; }

public string Naziv { get; set; }

}

}

Listing 5. – kreiranje tabele *Klasa*

Kreiranje uslova za samostalnu kupovinu karte (kupac mora da bude punoletan)

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

namespace EvidencijaAvioKarata.Models

{

public class PoslovnoPravilo : ValidationAttribute

{

protected override ValidationResult IsValid(object value, ValidationContext validationContext)

{

var kupac = (Kupac)validationContext.ObjectInstance;

if ((DateTime.Now.Year - kupac.DatumRodjenja.Year) < kupac.Godine)

return new ValidationResult("Kupac nije punoletan.");

else

return ValidationResult.Success;

}

}

}

Listing 6. – kreiranje klase *PoslovnoPravilo*

Nakon kreiranja tabela baze podataka, izvršene su migracije koje su kreirale bazu podataka.

namespace EvidencijaAvioKarata.Migrations

{

using System;

using System.Data.Entity.Migrations;

public partial class InitialCreate : DbMigration

{

public override void Up()

{

CreateTable(

"dbo.Destinacija",

c => new

{

Id = c.Int(nullable: false, identity: true),

Naziv = c.String(nullable: false),

ImeAerodroma = c.String(nullable: false),

Grad = c.String(nullable: false),

})

.PrimaryKey(t => t.Id);

CreateTable(

"dbo.Karta",

c => new

{

Id = c.Int(nullable: false, identity: true),

tipKarte = c.String(nullable: false),

DestinacijaId = c.Int(nullable: false),

KompanijaId = c.Int(nullable: false),

DatumIVreme = c.DateTime(nullable: false),

Cena = c.Int(nullable: false),

})

.PrimaryKey(t => t.Id)

.ForeignKey("dbo.Destinacija", t => t.DestinacijaId, cascadeDelete: true)

.ForeignKey("dbo.Kompanija", t => t.KompanijaId, cascadeDelete: true)

.Index(t => t.DestinacijaId)

.Index(t => t.KompanijaId);

CreateTable(

"dbo.Kompanija",

c => new

{

Id = c.Int(nullable: false, identity: true),

Ime = c.String(nullable: false),

Drzava = c.String(nullable: false),

KlasaId = c.Byte(nullable: false),

})

.PrimaryKey(t => t.Id)

.ForeignKey("dbo.Klasa", t => t.KlasaId, cascadeDelete: true)

.Index(t => t.KlasaId);

CreateTable(

"dbo.Klasa",

c => new

{

Id = c.Byte(nullable: false),

Naziv = c.String(),

})

.PrimaryKey(t => t.Id);

CreateTable(

"dbo.Kupci",

c => new

{

Id = c.Int(nullable: false, identity: true),

Ime = c.String(nullable: false),

DatumRodjenja = c.DateTime(nullable: false),

KartaId = c.Int(nullable: false),

BrojKarata = c.Int(nullable: false),

})

.PrimaryKey(t => t.Id)

.ForeignKey("dbo.Karta", t => t.KartaId, cascadeDelete: true)

.Index(t => t.KartaId);

CreateTable(

"dbo.AspNetRoles",

c => new

{

Id = c.String(nullable: false, maxLength: 128),

Name = c.String(nullable: false, maxLength: 256),

})

.PrimaryKey(t => t.Id)

.Index(t => t.Name, unique: true, name: "RoleNameIndex");

CreateTable(

"dbo.AspNetUserRoles",

c => new

{

UserId = c.String(nullable: false, maxLength: 128),

RoleId = c.String(nullable: false, maxLength: 128),

})

.PrimaryKey(t => new { t.UserId, t.RoleId })

.ForeignKey("dbo.AspNetRoles", t => t.RoleId, cascadeDelete: true)

.ForeignKey("dbo.AspNetUsers", t => t.UserId, cascadeDelete: true)

.Index(t => t.UserId)

.Index(t => t.RoleId);

CreateTable(

"dbo.AspNetUsers",

c => new

{

Id = c.String(nullable: false, maxLength: 128),

Email = c.String(maxLength: 256),

EmailConfirmed = c.Boolean(nullable: false),

PasswordHash = c.String(),

SecurityStamp = c.String(),

PhoneNumber = c.String(),

PhoneNumberConfirmed = c.Boolean(nullable: false),

TwoFactorEnabled = c.Boolean(nullable: false),

LockoutEndDateUtc = c.DateTime(),

LockoutEnabled = c.Boolean(nullable: false),

AccessFailedCount = c.Int(nullable: false),

UserName = c.String(nullable: false, maxLength: 256),

})

.PrimaryKey(t => t.Id)

.Index(t => t.UserName, unique: true, name: "UserNameIndex");

CreateTable(

"dbo.AspNetUserClaims",

c => new

{

Id = c.Int(nullable: false, identity: true),

UserId = c.String(nullable: false, maxLength: 128),

ClaimType = c.String(),

ClaimValue = c.String(),

})

.PrimaryKey(t => t.Id)

.ForeignKey("dbo.AspNetUsers", t => t.UserId, cascadeDelete: true)

.Index(t => t.UserId);

CreateTable(

"dbo.AspNetUserLogins",

c => new

{

LoginProvider = c.String(nullable: false, maxLength: 128),

ProviderKey = c.String(nullable: false, maxLength: 128),

UserId = c.String(nullable: false, maxLength: 128),

})

.PrimaryKey(t => new { t.LoginProvider, t.ProviderKey, t.UserId })

.ForeignKey("dbo.AspNetUsers", t => t.UserId, cascadeDelete: true)

.Index(t => t.UserId);

}

public override void Down()

{

DropForeignKey("dbo.AspNetUserRoles", "UserId", "dbo.AspNetUsers");

DropForeignKey("dbo.AspNetUserLogins", "UserId", "dbo.AspNetUsers");

DropForeignKey("dbo.AspNetUserClaims", "UserId", "dbo.AspNetUsers");

DropForeignKey("dbo.AspNetUserRoles", "RoleId", "dbo.AspNetRoles");

DropForeignKey("dbo.Kupci", "KartaId", "dbo.Karta");

DropForeignKey("dbo.Karta", "KompanijaId", "dbo.Kompanija");

DropForeignKey("dbo.Kompanija", "KlasaId", "dbo.Klasa");

DropForeignKey("dbo.Karta", "DestinacijaId", "dbo.Destinacija");

DropIndex("dbo.AspNetUserLogins", new[] { "UserId" });

DropIndex("dbo.AspNetUserClaims", new[] { "UserId" });

DropIndex("dbo.AspNetUsers", "UserNameIndex");

DropIndex("dbo.AspNetUserRoles", new[] { "RoleId" });

DropIndex("dbo.AspNetUserRoles", new[] { "UserId" });

DropIndex("dbo.AspNetRoles", "RoleNameIndex");

DropIndex("dbo.Kupci", new[] { "KartaId" });

DropIndex("dbo.Kompanija", new[] { "KlasaId" });

DropIndex("dbo.Karta", new[] { "KompanijaId" });

DropIndex("dbo.Karta", new[] { "DestinacijaId" });

DropTable("dbo.AspNetUserLogins");

DropTable("dbo.AspNetUserClaims");

DropTable("dbo.AspNetUsers");

DropTable("dbo.AspNetUserRoles");

DropTable("dbo.AspNetRoles");

DropTable("dbo.Kupci");

DropTable("dbo.Klasa");

DropTable("dbo.Kompanija");

DropTable("dbo.Karta");

DropTable("dbo.Destinacija");

}

}

}

*Listing 7. – kreirana migracija “InitialCreate”*

namespace EvidencijaAvioKarata.Migrations

{

using System;

using System.Data.Entity.Migrations;

public partial class PopunjavanjeKlase : DbMigration

{

public override void Up()

{

Sql("INSERT INTO Klasa (Id, Naziv) VALUES (1, 'Prva klasa')");

Sql("INSERT INTO Klasa (Id, Naziv) VALUES (2, 'Biznis klasa')");

Sql("INSERT INTO Klasa (Id, Naziv) VALUES (3, 'Ekonomska klasa')");

}

public override void Down()

{

}

}

}

*Listing 8. – kreirana migracija “PopunjavanjeKlasa”*

Dodati su Controller-i koji služe za uspostavljanje logike sloja baze podataka i korisničkog interfejsa i koji predstavljaju backend segment razvoja aplikacije.

using EvidencijaAvioKarata.Models;

using EvidencijaAvioKarata.ViewModels;

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace EvidencijaAvioKarata.Controllers

{

public class KarteController : Controller

{

private ApplicationDbContext \_context;

//KONSTRUKTOR - INICIJALIZACIJA CONTEKSTA

public KarteController()

{

\_context = new ApplicationDbContext();

}

protected override void Dispose(bool disposing)

{

\_context.Dispose();

}

//POCETNA FUNKCIJA - INDEX

public ActionResult Index()

{

var karta = \_context.Karte.Include(t => t.Destinacija).Include(t => t.Kompanija).ToList();

return View(karta);

}

public ActionResult DodajKartu()

{

var karta = new KartaForma

{

Karta = new Karta(),

Destinacije = \_context.Destinacije.ToList(),

Kompanije = \_context.Kompanije.ToList()

};

return View("KartaForma", karta);

}

public ActionResult KartaForma(int id)

{

var kartaUBazi = \_context.Karte.SingleOrDefault(p => p.Id == id);

var viewModel = new KartaForma

{

Karta = kartaUBazi,

Destinacije = \_context.Destinacije.ToList(),

Kompanije = \_context.Kompanije.ToList()

};

return View("KartaForma", viewModel);

}

[HttpPost]

public ActionResult Sacuvaj(Karta karta)

{

if (!ModelState.IsValid)

{

var viewModel = new KartaForma

{

Karta = karta,

Destinacije = \_context.Destinacije.ToList(),

Kompanije = \_context.Kompanije.ToList()

};

return View("KartaForma", viewModel);

}

//DODAVANJE NOVOG ZAPISA

if (karta.Id == 0)

{

\_context.Karte.Add(karta);

\_context.SaveChanges();

return RedirectToAction("Index");

}

//AZURIRANJE

else

{

var kartaUBazi = \_context.Karte.SingleOrDefault(p => p.Id == karta.Id);

if (kartaUBazi == null)

return HttpNotFound();

kartaUBazi.Id = karta.Id;

kartaUBazi.tipKarte = karta.tipKarte;

kartaUBazi.KompanijaId = karta.KompanijaId;

kartaUBazi.DestinacijaId = karta.DestinacijaId;

kartaUBazi.DatumIVreme = karta.DatumIVreme;

kartaUBazi.Cena = karta.Cena;

\_context.SaveChanges();

return RedirectToAction("Index");

}

}

}

}

Listing 9. – kreiranje controllor-a *Karta*

using EvidencijaAvioKarata.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace EvidencijaAvioKarata.Controllers

{

public class DestinacijeController : Controller

{

//Instanca DBContext-a(Preko njega pristupamo bazi)

private ApplicationDbContext \_context;

//Inicijalizacija DBContext-a

public DestinacijeController()

{

\_context = new ApplicationDbContext();

}

//Izbacivanje resursa iz memorije(Na kraju svake akcije, dolazi do dispose-a)

protected override void Dispose(bool disposing)

{

\_context.Dispose();

}

//Akcija za vracanje svih destinacija

public ActionResult Index()

{

var destinacije = \_context.Destinacije.ToList();

return View(destinacije);

}

//Akcija za usmeravanje na formu za destinaciju

public ActionResult DodajDestinaciju()

{

var destinacija = new Destinacija();

return View("DestinacijaForma", destinacija);

}

//Akcija za usmeravanje za izmenu neke destinacije

public ActionResult DestinacijaForma(int id)

{

//Trazi destinaciju sa poslatim Id-om u bazi.

var destinacijaUBazi = \_context.Destinacije.SingleOrDefault(p => p.Id == id);

//View model sa podacima te pronadjene destinacije

var viewModel = new Destinacija

{

Naziv = destinacijaUBazi.Naziv,

ImeAerodroma = destinacijaUBazi.ImeAerodroma,

Grad = destinacijaUBazi.Grad,

};

//Vraca mesto u formu

return View("DestinacijaForma", viewModel);

}

//Akcija za sacuvavanje novog zapisa ili izmenu zapisa

[HttpPost]

public ActionResult Sacuvaj(Destinacija destinacija)

{

//Provera da li su sva polja u formi popunjena

if (!ModelState.IsValid)

{

var viewModel = new Destinacija();

return View("DestinacijaForma", viewModel);

}

//Ako je id destinacije jednak 0, to znaci da je u pitanju dodavanje nove destinacije, posto joj je id 0, znaci da je nema u bazi.

if (destinacija.Id == 0)

{

\_context.Destinacije.Add(destinacija);

\_context.SaveChanges();

return RedirectToAction("Index");

}

//Ako je u pitanu id koji nije 0, znaci da se radi o izmeni postojeceg zapisa

else

{

//Nalazi zapis sa id-em za izmenu

var destinacijaUBazi = \_context.Destinacije.SingleOrDefault(p => p.Id == destinacija.Id);

//Provera ako uopste taj zapis uopste postoji u bazi

if (destinacijaUBazi == null)

return HttpNotFound();

//Menja postojece podatke sa poslatim podacima iz forme

destinacijaUBazi.Naziv = destinacija.Naziv;

destinacijaUBazi.ImeAerodroma = destinacija.ImeAerodroma;

destinacijaUBazi.Grad = destinacija.Grad;

\_context.SaveChanges();

return RedirectToAction("Index");

}

}

}

}

Listing 10. – kreiranje controllor-a *Destinacija*

using EvidencijaAvioKarata.Models;

using EvidencijaAvioKarata.ViewModels;

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace EvidencijaAvioKarata.Controllers

{

public class KompanijeController : Controller

{

private ApplicationDbContext \_context;

public KompanijeController()

{

\_context = new ApplicationDbContext();

}

protected override void Dispose(bool disposing)

{

\_context.Dispose();

}

// GET: Kompanija

public ActionResult Index()

{

var kompanija = \_context.Kompanije.Include(p => p.Klasa).ToList();

return View(kompanija);

}

public ActionResult DodajKompaniju()

{

//Pravi novi objekat kompanija i salje ga na view

var kompanija = new KompanijaForma

{

Kompanija = new Kompanija(),

Klase = \_context.Klase.ToList()

};

return View("KompanijaForma", kompanija);

}

//Salje na formu za izmenu

public ActionResult KompanijaForma(int id)

{

var kompanijaUBazi = \_context.Kompanije.SingleOrDefault(p => p.Id == id);

var viewModel = new KompanijaForma

{

Kompanija = kompanijaUBazi,

Klase = \_context.Klase.ToList()

};

return View("KompanijaForma", viewModel);

}

[HttpPost]

public ActionResult Sacuvaj(Kompanija kompanija)

{

if (!ModelState.IsValid)

{

var viewModel = new KompanijaForma

{

Kompanija = kompanija,

Klase = \_context.Klase.ToList()

};

return View("KompanijaForma", viewModel);

}

//Dodavanje nove kompanije u bazu

if (kompanija.Id == 0)

{

\_context.Kompanije.Add(kompanija);

\_context.SaveChanges();

return RedirectToAction("Index");

}

//Izmena postojece kompanije iz baze

else

{

var kompanijaUBazi = \_context.Kompanije.SingleOrDefault(p => p.Id == kompanija.Id);

if (kompanijaUBazi == null)

return HttpNotFound();

kompanijaUBazi.Ime = kompanija.Ime;

kompanijaUBazi.Drzava = kompanija.Drzava;

kompanijaUBazi.KlasaId = kompanija.KlasaId;

\_context.SaveChanges();

return RedirectToAction("Index");

}

}

}

}

Listing 11. – kreiranje controllor-a *Kompanija*

using EvidencijaAvioKarata.Models;

using EvidencijaAvioKarata.ViewModels;

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace EvidencijaAvioKarata.Controllers

{

public class KupciController : Controller

{

// GET: Kupci

private ApplicationDbContext \_context;

public KupciController()

{

\_context = new ApplicationDbContext();

}

protected override void Dispose(bool disposing)

{

\_context.Dispose();

}

// GET: Kompanija

public ActionResult Index()

{

var kupci = \_context.Kupci.Include(c => c.Karta).ToList();

return View(kupci);

}

public ActionResult DodajKupca()

{

var kupac = new KupacForma

{

Kupac = new Kupac

{

DatumRodjenja = new DateTime(2000, 1, 1)

},

Karte = \_context.Karte.ToList(),

};

return View("KupacForma", kupac);

}

public ActionResult KupacForma(int id)

{

var kupacUBazi = \_context.Kupci.SingleOrDefault(c => c.Id == id);

var viewModel = new KupacForma

{

Kupac = kupacUBazi,

Karte = \_context.Karte.ToList(),

};

return View("KupacForma", viewModel);

}

[HttpPost]

public ActionResult Sacuvaj(Kupac kupac)

{

if (!ModelState.IsValid)

{

var viewModel = new KupacForma

{

Kupac = new Kupac(),

Karte = \_context.Karte.ToList(),

};

return View("KupacForma", viewModel);

}

if (kupac.Id == 0)

{

\_context.Kupci.Add(kupac);

\_context.SaveChanges();

return RedirectToAction("Index");

}

else

{

var kupacUBazi = \_context.Kupci.SingleOrDefault(p => p.Id == kupac.Id);

if (kupacUBazi == null)

return HttpNotFound();

kupacUBazi.Ime = kupac.Ime;

kupacUBazi.DatumRodjenja = kupac.DatumRodjenja;

kupacUBazi.KartaId = kupac.KartaId;

kupacUBazi.BrojKarata = kupac.BrojKarata;

\_context.SaveChanges();

return RedirectToAction("Index");

}

}

}

}

Listing 12. – kreiranje controllor-a *Kupci*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace EvidencijaAvioKarata.Controllers

{

public class HomeController : Controller

{

public ActionResult Index()

{

return View();

}

public ActionResult About()

{

ViewBag.Message = "Your application description page.";

return View();

}

public ActionResult Contact()

{

ViewBag.Message = "Your contact page.";

return View();

}

}

}

Listing 13. – Automatska klasa koju je kreirao program kao primer

U okviru Controller-a kreirana je fascikla API, koja će obrađivati dolazne HTTP zahteve i slati odgovor nazad korisniku.

using EvidencijaAvioKarata.Models;

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace EvidencijaAvioKarata.Controllers.Api

{

public class KarteController : ApiController

{

private ApplicationDbContext \_context;

public KarteController()

{

\_context = new ApplicationDbContext();

}

// GET: /api/tickets

[HttpGet]

public IEnumerable<Karta> DajSveKarte()

{

var karteUBazi = \_context.Karte.

Include(t => t.Kompanija).

Include(t => t.Destinacija).ToList();

return karteUBazi;

}

[HttpDelete]

public IHttpActionResult Obrisi(int id)

{

var kartaUBazi = \_context.Karte.SingleOrDefault(p => p.Id == id);

if (kartaUBazi == null)

return NotFound();

\_context.Karte.Remove(kartaUBazi);

\_context.SaveChanges();

return Ok();

}

}

}

Listing 14. – kreiranje API controllor-a *Karte*

using EvidencijaAvioKarata.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace EvidencijaAvioKarata.Controllers.Api

{

public class DestinacijeController : ApiController

{

private ApplicationDbContext \_context;

public DestinacijeController()

{

\_context = new ApplicationDbContext();

}

//Web servis koji vraca listu destinacija

[HttpGet]

public IEnumerable<Destinacija> DajSveDestinacije()

{

var destinacijeUBazi = \_context.Destinacije.ToList();

return destinacijeUBazi;

}

//Web servis koji brise destinaciju u bazi

[HttpDelete]

public IHttpActionResult Obrisi(int id)

{

//Destinacija u bazi sa trazenim id-em

var destinacijaUBazi = \_context.Destinacije.SingleOrDefault(p => p.Id == id);

//Ako ne postoji vraca gresku 404

if (destinacijaUBazi == null)

return NotFound();

//Brise je iz baze ako je prethodno prosla proveru

\_context.Destinacije.Remove(destinacijaUBazi);

\_context.SaveChanges();

return Ok();

}

}

}

Listing 15. – kreiranje API controllor-a *Destinacije*

using EvidencijaAvioKarata.Models;

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace EvidencijaAvioKarata.Controllers.Api

{

public class KompanijeController : ApiController

{

private ApplicationDbContext \_context;

public KompanijeController()

{

\_context = new ApplicationDbContext();

}

[HttpGet]

public IEnumerable<Kompanija> DajSveKompanije()

{

var kompanijeUBazi = \_context.Kompanije.Include(p => p.Klasa).ToList();

return kompanijeUBazi;

}

[HttpDelete]

public IHttpActionResult Obrisi(int id)

{

var kompanijaUBazi = \_context.Kompanije.SingleOrDefault(p => p.Id == id);

if (kompanijaUBazi == null)

return NotFound();

\_context.Kompanije.Remove(kompanijaUBazi);

\_context.SaveChanges();

return Ok();

}

}

}

Listing 16. – kreiranje API controllor-a *Kompanije*

using EvidencijaAvioKarata.Models;

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace EvidencijaAvioKarata.Controllers.Api

{

public class KupciController : ApiController

{

private ApplicationDbContext \_context;

public KupciController()

{

\_context = new ApplicationDbContext();

}

// GET: /api/kupci

[HttpGet]

public IEnumerable<Kupac> DajSveKupce()

{

var kupciUBazi = \_context.Kupci.Include(c => c.Karta).ToList();

return kupciUBazi;

}

[HttpDelete]

public IHttpActionResult Obrisi(int id)

{

var kupacUBazi = \_context.Kupci.SingleOrDefault(p => p.Id == id);

if (kupacUBazi == null)

return NotFound();

\_context.Kupci.Remove(kupacUBazi);

\_context.SaveChanges();

return Ok();

}

}

}

Listing 17. – kreiranje API controllor-a *Kupci*

Kreiranje View-ova omugući će generisanje HTML-a pregledačima koji posećuju web stranicu. Dodati su sledeći view-ovi:

* Destinacija: Index i DestinacijaForma
* Karta: Index i KartaForma
* Kompanija: Index i KompanijaForma
* Kupci: Index i KupacForma
* Home: Index, About i Contact

@model IEnumerable<EvidencijaAvioKarata.Models.Destinacija>

@{

ViewBag.Title = "Destinacije";

}

<div class="custom-container">

<h2>Destinacije</h2>

<table id="destinacije" class="table table-bordered">

<thead>

<tr>

<th>Naziv</th>

<th>Ime aerodroma</th>

<th>Grad</th>

<th>Obriši</th>

</tr>

</thead>

<tbody>

</tbody>

</table>

@Html.ActionLink("Dodaj destinaciju", "DodajDestinaciju", "Destinacije", new { @class = "btn btn-danger" })

</div>

@section scripts

{

<script>

$(document).ready(function () {

var table = $("#destinacije").DataTable({

//Ajax sluzi za pozivanje API-a

ajax: {

url: "/api/destinacije/",

method: "GET",

dataSrc: ""

},

//Vraca sve kolone u tabeli sa vrednostima

columns: [

{

data: "naziv",

render: function (data, type, destinacija) {

return "<a href='/destinacije/destinacijaForma/" + destinacija.id + "'>" + destinacija.naziv + "</a>";

}

},

{

data: "imeAerodroma"

},

{

data: "grad"

},

{

data: "id",

render: function (data) {

return "<button class='btn-link js-delete' data-destinacija-id=" + data + ">Obrisi</button>";

}

}

]

});

//Kreiranje dugmeta za brisanje - daje mu klasu .js-delete, i na klik brise odredjeno mesto

$("#destinacije").on("click", ".js-delete", function () {

var button = $(this);

//AlertBox za potvrdu brisanja

bootbox.confirm("Da li ste sigurni da zelite da obrišete ovu destinaciju?", function (result) {

if (result) {

$.ajax({

url: "/api/destinacije/" + button.attr("data-destinacija-id"),

method: "DELETE",

success: function () {

table.row(button.parents("tr")).remove().draw();

}

})

}

})

})

});

</script>

}

Listing 18. – kreiranje View-a *Index za Destinaciju*

@model EvidencijaAvioKarata.Models.Destinacija

@{

ViewBag.Title = "DestinacijaForma";

}

<div class="custom-container">

<h2>Destinacija forma</h2>

@using (Html.BeginForm("Sacuvaj", "Destinacije"))

{

<div class="form-group">

@Html.LabelFor(p => p.Naziv)

@Html.TextBoxFor(p => p.Naziv, new { @class = "form-control" })

@Html.ValidationMessageFor(p => p.Naziv)

</div>

<div class="form-group">

@Html.LabelFor(p => p.ImeAerodroma)

@Html.TextBoxFor(p => p.ImeAerodroma, new { @class = "form-control" })

@Html.ValidationMessageFor(p => p.ImeAerodroma)

</div>

<div class="form-group">

@Html.LabelFor(p => p.Grad)

@Html.TextBoxFor(p => p.Grad, new { @class = "form-control" })

@Html.ValidationMessageFor(p => p.Grad)

</div>

@Html.HiddenFor(p => p.Id)

<button class="btn btn-danger" type="submit">Sačuvaj</button>

}

</div>

Listing 19. – kreiranje View-a *DestinacijaForma*

@model IEnumerable<EvidencijaAvioKarata.Models.Karta>

@{

ViewBag.Title = "Karte";

}

<div class="custom-container">

<h2>Karte</h2>

<table id="karte" class="table table-bordered">

<thead>

<tr>

<th>Tip karte</th>

<th>Destinacija</th>

<th>Kompanija</th>

<th>Datum i vreme</th>

<th>Cena</th>

<th>Obriši</th>

</tr>

</thead>

<tbody>

</tbody>

</table>

@Html.ActionLink("Dodaj kartu", "DodajKartu", "Karte", new { @class = "btn btn-danger" })

</div>

@section scripts

{

<script>

$(document).ready(function () {

var table = $("#karte").DataTable({

ajax: {

url: "/api/karte/",

method: "GET",

dataSrc: ""

},

columns: [

{

data: "tipKarte",

render: function (data, type, karta) {

return "<a href='/karte/kartaForma" + karta.Id + "'>" + karta.tipKarte + "</a>";

}

},

{

data: "destinacija.naziv"

},

{

data: "kompanija.ime"

},

{

data: "datumIVreme"

},

{

data: "cena"

},

{

data: "id",

render: function (data) {

return "<button class='btn-link js-delete' data-karta-id=" + data + ">Obrisi</button>";

}

}

]

});

$("#karte").on("click", ".js-delete", function () {

var button = $(this);

bootbox.confirm("Da li ste sigurni da želite da obrišete ovu kartu?", function (result) {

if (result) {

$.ajax({

url: "/api/karte/" + button.attr("data-karta-id"),

method: "DELETE",

success: function () {

table.row(button.parents("tr")).remove().draw();

}

})

}

})

})

});

</script>

}

Listing 20. – kreiranje View-a *Index za Kartu*

@model EvidencijaAvioKarata.ViewModels.KartaForma

@{

ViewBag.Title = "KartaForma";

}

<div class="custom-container">

<h2>Karta forma</h2>

@using (Html.BeginForm("Sacuvaj", "Karte"))

{

<div class="form-group">

@Html.LabelFor(m => m.Karta.tipKarte)

@Html.TextBoxFor(p => p.Karta.tipKarte, new { @class = "form-control" })

@Html.ValidationMessageFor(m => m.Karta.tipKarte)

</div>

<div class="form-group">

@Html.LabelFor(m => m.Karta.KompanijaId)

@Html.DropDownListFor(m => m.Karta.KompanijaId, new SelectList(Model.Kompanije, "Id", "Ime"), "Izaberite kompaniju", new { @class = "form-control" })

@Html.ValidationMessageFor(m => m.Karta.KompanijaId)

</div>

<div class="form-group">

@Html.LabelFor(m => m.Karta.DestinacijaId)

@Html.DropDownListFor(m => m.Karta.DestinacijaId, new SelectList(Model.Destinacije, "Id", "Naziv"), "Izaberite destinaciju", new { @class = "form-control" })

@Html.ValidationMessageFor(m => m.Karta.DestinacijaId)

</div>

<div class="form-group">

@Html.LabelFor(p => p.Karta.DatumIVreme)

@Html.TextBoxFor(p => p.Karta.DatumIVreme, new { @class = "form-control" })

@Html.ValidationMessageFor(p => p.Karta.DatumIVreme)

</div>

<div class="form-group">

@Html.LabelFor(p => p.Karta.Cena)

@Html.TextBoxFor(p => p.Karta.Cena, new { @class = "form-control" })

@Html.ValidationMessageFor(p => p.Karta.Cena)

</div>

@Html.HiddenFor(p => p.Karta.Id)

<button class="btn btn-danger" type="submit">Sačuvaj</button>

}

</div>

Listing 21. – kreiranje View-a *KartaForma*

@model IEnumerable<EvidencijaAvioKarata.Models.Kompanija>

@{

ViewBag.Title = "Kompanije";

}

<div class="custom-container">

<h2>Kompanije</h2>

<table id="kompanije" class="table table-bordered">

<thead>

<tr>

<th>Ime</th>

<th>Država</th>

<th>Klasa</th>

<th>Obriši</th>

</tr>

</thead>

<tbody>

</tbody>

</table>

@Html.ActionLink("Dodaj kompaniju", "DodajKompaniju", "Kompanije", new { @class = "btn btn-danger" })

</div>

@section scripts

{

<script>

$(document).ready(function () {

var table = $("#kompanije").DataTable({

ajax: {

url: "/api/kompanije/",

method: "GET",

dataSrc: ""

},

columns: [

{

data: "ime",

render: function (data, type, kompanija) {

return "<a href='/kompanije/kompanijaForma/" + kompanija.id + "'>" + kompanija.ime + "</a>";

}

},

{

data: "drzava"

},

{

data: "klasa.naziv"

},

{

data: "id",

render: function (data) {

return "<button class='btn-link js-delete' data-kompanija-id=" + data + ">Obrisi</button>";

}

}

]

});

$("#kompanije").on("click", ".js-delete", function () {

var button = $(this);

bootbox.confirm("Da li ste sigurni da zelite da obrišete ovu kompaniju?", function (result) {

if (result) {

$.ajax({

url: "/api/kompanije/" + button.attr("data-kompanija-id"),

method: "DELETE",

success: function () {

table.row(button.parents("tr")).remove().draw();

}

})

}

})

})

});

</script>

}

Listing 22. – kreiranje View-a *Index za Kompanije*

@model EvidencijaAvioKarata.ViewModels.KompanijaForma

@{

ViewBag.Title = "KompanijaForma";

}

<div class="custom-container">

<h2>Kompanija forma</h2>

@using (Html.BeginForm("Sacuvaj", "Kompanije"))

{

<div class="form-group">

@Html.LabelFor(p => p.Kompanija.Ime)

@Html.TextBoxFor(p => p.Kompanija.Ime, new { @class = "form-control" })

@Html.ValidationMessageFor(p => p.Kompanija.Ime)

</div>

<div class="form-group">

@Html.LabelFor(p => p.Kompanija.Drzava)

@Html.TextBoxFor(p => p.Kompanija.Drzava, new { @class = "form-control" })

@Html.ValidationMessageFor(p => p.Kompanija.Drzava)

</div>

<div class="form-group">

@Html.LabelFor(m => m.Kompanija.KlasaId)

@Html.DropDownListFor(m => m.Kompanija.KlasaId, new SelectList(Model.Klase, "Id", "Naziv"), "Izaberi klasu", new { @class = "form-control" })

@Html.ValidationMessageFor(m => m.Kompanija.KlasaId)

</div>

@Html.HiddenFor(p => p.Kompanija.Id)

<button class="btn btn-danger" type="submit">Sačuvaj</button>

}

</div>

Listing 23. – kreiranje View-a *KompanijaForma*

@model IEnumerable<EvidencijaAvioKarata.Models.Kupac>

@{

ViewBag.Title = "Kupci";

}

<div class="custom-container">

<h2>Kupci</h2>

<table id="kupci" class="table table-bordered">

<thead>

<tr>

<th>Ime</th>

<th>Datum rođenja</th>

<th>Tip karte</th>

<th>Broj karata</th>

<th>Obriši</th>

</tr>

</thead>

<tbody>

</tbody>

</table>

@Html.ActionLink("Dodaj kupca", "DodajKupca", "Kupci", new { @class = "btn btn-danger" })

</div>

@section scripts

{

<script>

$(document).ready(function () {

var table = $("#kupci").DataTable({

ajax: {

url: "/api/kupci/",

method: "GET",

dataSrc: ""

},

columns: [

{

data: "ime",

render: function (data, type, kupac) {

return "<a href='/kupci/kupacForma/" + kupac.id + "'>" + kupac.ime + "</a>";

}

},

{

data: "datumRodjenja"

},

{

data: "karta.tipKarte"

},

{

data: "brojKarata"

},

{

data: "id",

render: function (data) {

return "<button class='btn-link js-delete' data-karta-id=" + data + ">Obrisi</button>";

}

}

]

});

$("#kupci").on("click", ".js-delete", function () {

var button = $(this);

bootbox.confirm("Da li ste sigurni da zelite da obrisete ovog kupca?", function (result) {

if (result) {

$.ajax({

url: "/api/kupci/" + button.attr("data-karta-id"),

method: "DELETE",

success: function () {

table.row(button.parents("tr")).remove().draw();

}

})

}

})

})

});

</script>

}

Listing 24. – kreiranje View-a *Index za Kupce*

@model EvidencijaAvioKarata.ViewModels.KupacForma

@{

ViewBag.Title = "KupacForma";

}

<div class="custom-container">

<h2>Kupac forma</h2>

@using (Html.BeginForm("Sacuvaj", "Kupci"))

{

<div class="form-group">

@Html.LabelFor(c => c.Kupac.Ime)

@Html.TextBoxFor(c => c.Kupac.Ime, new { @class = "form-control" })

@Html.ValidationMessageFor(c => c.Kupac.Ime)

</div>

<div class="form-group">

@Html.LabelFor(c => c.Kupac.DatumRodjenja)

@Html.TextBoxFor(c => c.Kupac.DatumRodjenja, "{0:d MMM yyyy}", new { @class = "form-control" })

@Html.ValidationMessageFor(c => c.Kupac.DatumRodjenja)

</div>

<div class="form-group">

@Html.LabelFor(m => m.Kupac.KartaId)

@Html.DropDownListFor(m => m.Kupac.KartaId, new SelectList(Model.Karte, "Id", "tipKarte"), "Izaberi kartu", new { @class = "form-control" })

@Html.ValidationMessageFor(m => m.Kupac.KartaId)

</div>

<div class="form-group">

@Html.LabelFor(c => c.Kupac.BrojKarata)

@Html.TextBoxFor(c => c.Kupac.BrojKarata, new { @class = "form-control" })

@Html.ValidationMessageFor(c => c.Kupac.BrojKarata)

</div>

@Html.HiddenFor(c => c.Kupac.Id)

<button class="btn btn-danger" type="submit">Sacuvaj</button>

}

</div>

Listing 25. – kreiranje View-a *KupacForma*

@{

ViewBag.Title = "Home Page";

}

<div class="hero" style="background-image: url(/Images/cover.jpg)">

<h1 class="hero-heading">Evidencija avio karata</h1>

</div>

<div>

<h2 class="between-heading">Moguća je evidencija sledećih tabela: </h2>

</div>

<div class="main-columns">

<div class="col-lg-3 single-column">

<h3 class="columns-heading">Karte</h3>

<p class="columns-paragraph">Evidencija karata: </p>

<ul class="columns-list">

<li class="columns-list-item">Tip karte</li>

<li class="columns-list-item">Destinacija leta</li>

<li class="columns-list-item">Kompanija</li>

<li class="columns-list-item">Datum</li>

<li class="columns-list-item">Cena</li>

</ul>

<a href="~/Karte" class="btn btn-primary columns-btn">Karte</a>

</div>

<div class="col-lg-3 single-column">

<h3 class="columns-heading">Kompanije</h3>

<p class="columns-paragraph">Evidencija kompanija: </p>

<ul class="columns-list">

<li class="columns-list-item">Ime kompanije</li>

<li class="columns-list-item">Država</li>

<li class="columns-list-item">Klasa</li>

</ul>

<a href="~/Kompanije" class="btn btn-primary columns-btn">Kompanije</a>

</div>

<div class="col-lg-3 single-column">

<h3 class="columns-heading">Destinacije</h3>

<p class="columns-paragraph">Evidencija destinacija: </p>

<ul class="columns-list">

<li class="columns-list-item">Naziv destinacije</li>

<li class="columns-list-item">Ime aerodroma</li>

<li class="columns-list-item">Grad</li>

</ul>

<a href="~/Destinacije" class="btn btn-primary columns-btn">Destinacije</a>

</div>

<div class="col-lg-3 single-column">

<h3 class="columns-heading">Kupci</h3>

<p class="columns-paragraph">Evidencija kupaca: </p>

<ul class="columns-list">

<li class="columns-list-item">Ime kupca</li>

<li class="columns-list-item">Datum rodjenja</li>

<li class="columns-list-item">Kupljena karta</li>

<li class="columns-list-item">Količina kupljenih karata</li>

</ul>

<a href="~/Kupci" class="btn btn-primary columns-btn">Kupci</a>

</div>

</div>

Listing 26. – kreiranje View-a *Index za Home*

Kreirani su ViewModel-i koji se koriste za oblikovanje više entiteta iz jednog ili više modela u jedan objekat. Kreirani su sledeći ViewModeli:

* KartaForma
* KompanijaForma
* KupacForma

using EvidencijaAvioKarata.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace EvidencijaAvioKarata.ViewModels

{

public class KartaForma

{

public Karta Karta { get; set; }

public IEnumerable<Destinacija> Destinacije { get; set; }

public IEnumerable<Kompanija> Kompanije { get; set; }

}

}

Listing 27. – kreiranje ViewModel-a *za KartaFormu*

using EvidencijaAvioKarata.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace EvidencijaAvioKarata.ViewModels

{

public class KompanijaForma

{

public Kompanija Kompanija { get; set; }

public IEnumerable<Klasa> Klase { get; set; }

}

}

Listing 28. – kreiranje ViewModel-a *za KompanijaFormu*

using EvidencijaAvioKarata.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace EvidencijaAvioKarata.ViewModels

{

public class KupacForma

{

public Kupac Kupac { get; set; }

public IEnumerable<Karta> Karte { get; set; }

}

}

Listing 29. – kreiranje ViewModel-a *za KupacFormu*

# **6. Rad sa GitHub sistemom**

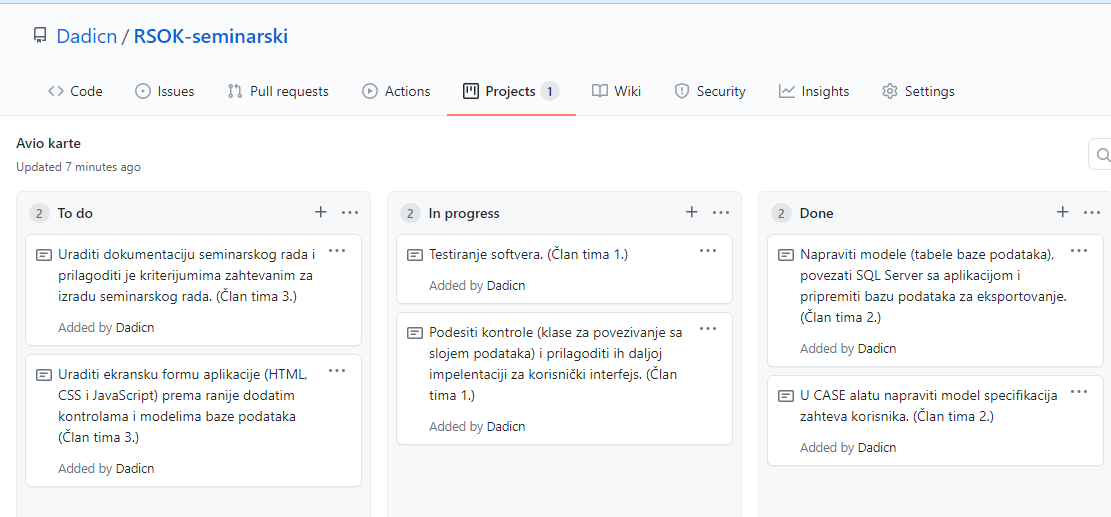
Kreiran je repozitorijum – <https://github.com/Dadicn/RSOK-seminarski>

Učesnici projekta: 1. Dadić Nikola (kreator repozitorijuma) 2. Radosavljević Jovan, 3. Vlaović Milica

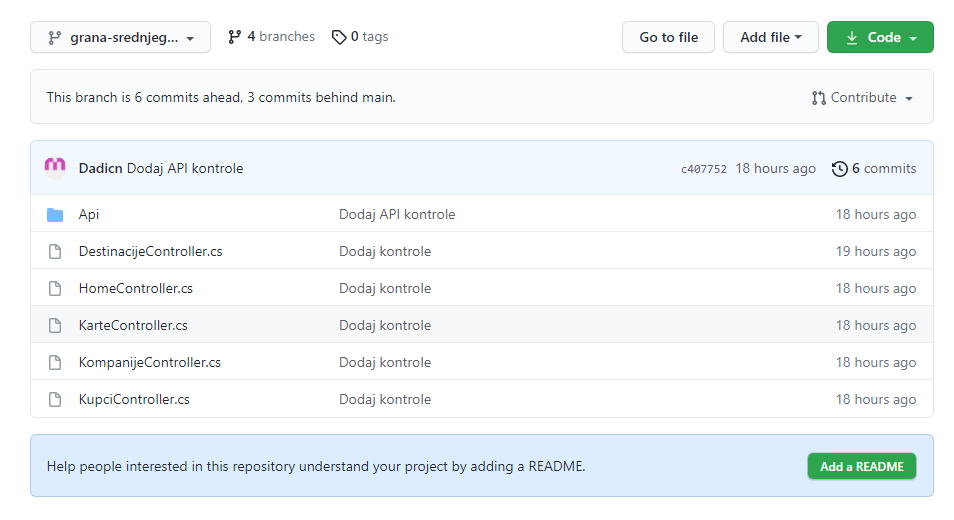
Kreirane grane:

1. grana-baze-podataka
2. grana-srednjeg-sloja
3. grana-korisnickog-interfejsa

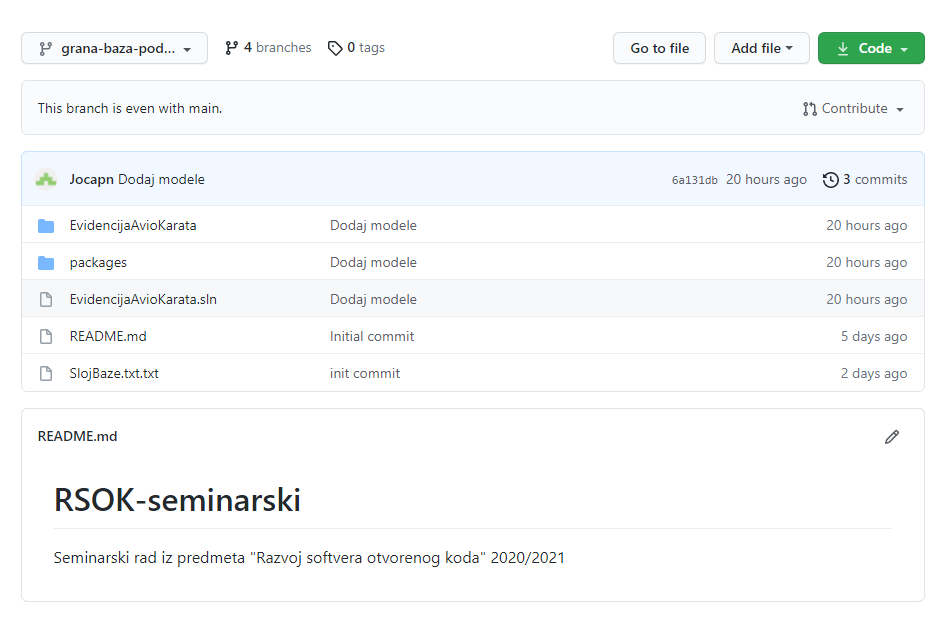
Prikaz projekta u repozitorijumu metodom Kanban kartica:



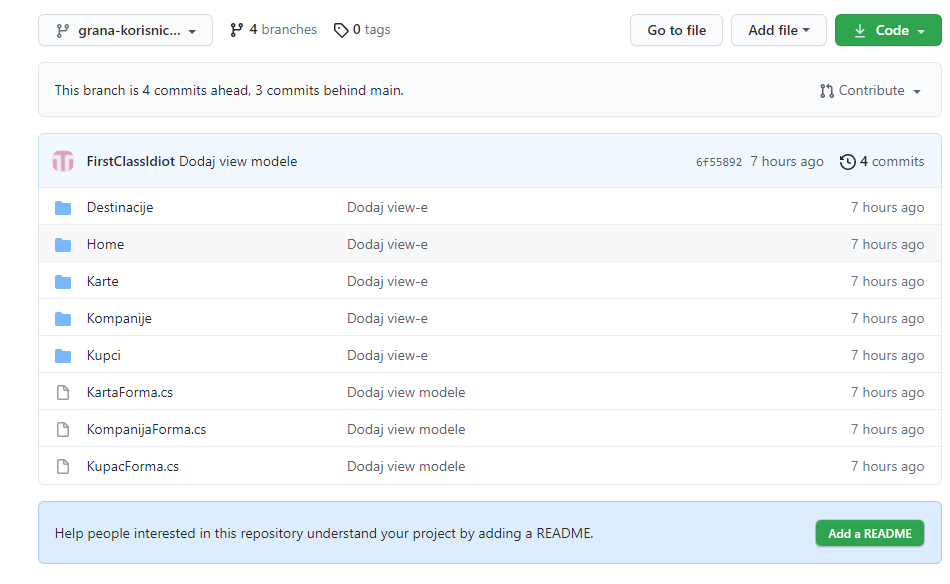
Prikaz broja Commit-a po granama, srednji aplikacioni sloj, ukupno 6 Commit-a:



Prikaz broja Commit-a po granama, sloj baze podataka, ukupno 3 Commit-a:



Prikaz broja Commit-a po granama, korisnički interfejs, ukupno 4 Commit-a:



Spajanje grana je izvršeno nakon završetka postavljanja *Push* svih datoteka na granama opcijom *Pull*.

# **7. Korišćeni alati i softveri**

* MySQL sistem za rukovanje bazama podataka,
* C# programski jezik za srednji aplikacioni sloj,
* HTML, CSS i JavaScript za korisnički interfejs uz korišćenje Bootstrap radnog okvira
* Editor programskog koda: Visual Studio 2019.

# **8. Literatura**

[1] <https://github.com/>

[2] <https://drive.google.com/file/d/11vxk2cLZ-JmJLyqG-0OVtXd0j8yljoEa/view>

[3] <http://www.tfzr.uns.ac.rs/Predmet/razvoj-softvera-otvorenog-koda/seminarski-radovi>