АКАДЕМИЈА ТЕХНИЧКО-УМЕТНИЧКИХ СТРУКОВНИХ СТУДИЈА БЕОГРАД

ОДСЕК ВИСОКА ШКОЛА ЕЛЕКТРОТЕХНИКЕ И РАЧУНАРСТВА

**Вуковић Игор**

**Пројектовање и имплементација информационог система на примеру пописа становништва**

**- завршни рад -**



Београд, месец 20\_\_.

Кандидат: **Вуковић Игор**

Број индекса: **НРТ-36/19**

Студијски програм: **Нове рачунарске технологије**

Тема: **Пројектовање и имплементација информационог система на примеру пописа становништва**

Основни задаци:

**1. Анализа пословних процеса**

**2. Пројектовање базе података**

**3. Практична реализација информационог система**

Ментор:

Београд, месец 20\_\_ годинe.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Мр Милош Пејановић, предавач ВИШЕР

Резиме:

Овај рад описује процес развоја програма за **попис становништва**. Програм се састоји од базе која је рађена у ***SQL Server Management Studio Management Studio* 19 (*SSMS*)** и апликативног дела програма рађеног у ***Visual Studio* 2017 (*VS*17).** Апликативни део садржи део за пријаву пописивача, попис становништва и преглед статистике пописаног становништва. За потребе рада су коришћени ***C#***, ***SQL*** и ***LINQ***

**Кључне речи:** Попис становништва, SQL Server Management Studio Management Studio 19, Visual Studio 2017, SQL, LINQ

ABSTRACT:

This assignment describes the process of developing a census program. The program consists of a base that was made in ***SQL Server Management Studio Management Studio* 19** and the application part of the program made in ***Visual Studio* 2017**. The application part contains a part for registration of enumerators, population census and an overview of statistics of the enumerated population. ***C#***, ***SQL*** and ***LINQ*** were used for the purpose of the work

**Key words**: Попис становништва, SQL Server Management Studio Management Studio 19, Visual Studio 2017, SQL, LINQ

САДРЖАЈ:

[1. Увод 1](#_Toc154932870)

[2. Анализа пословних процеса 2](#_Toc154932871)

[2.1. Замена старих технологија рада 2](#_Toc154932872)

[2.2. Извршавање у реалном времену 2](#_Toc154932873)

[2.3. Попис 2](#_Toc154932874)

[3. Пројектовање базе података 8](#_Toc154932877)

[3.1. Ток административне поделе 8](#_Toc154932878)

[3.2. Ток података питања 8](#_Toc154932879)

[4. Практична реализација информационог система 11](#_Toc154932880)

[4.1. Пријављивање 11](#_Toc154932881)

[4.2. Почетна 12](#_Toc154932882)

[4.3. Администрација 13](#_Toc154932885)

[4.4. Додај пописивача 14](#_Toc154932888)

[4.5. Статистика 16](#_Toc154932897)

[4.5.1. Становништво према националној припадности и полу 16](#_Toc154932907)

[4.5.2. Становнишво према старости и полу 17](#_Toc154932908)

[4.5.3. Просечна старост становнишва према полу 18](#_Toc154932909)

[4.5.4. Станови према површини 20](#_Toc154932910)

[4.5.5. Грејање у настањеним становима без централног грејања 22](#_Toc154932911)

[4.6. Попис 23](#_Toc154932912)

[4.6.1. Подаци о особи 23](#_Toc154932913)

[4.6.2. Миграција 25](#_Toc154932914)

[5. Индекс појмова 31](#_Toc154932915)

[6. Литература 32](#_Toc154932916)

[7. Изјава о академској честитости 33](#_Toc154932917)

# Увод

У овом раду ћемо проћи радне кроз радне задатке:

* Анализа пословних процеса
* Пројектовање базе података
* Практична реализација информационог система

Самој изради базе података и информационог система се морало пажљиво приступити због осетљивости пројекта јер се обрађују и анализирају лични подаци становништва који не смеју да се нађу у јавности осим кроз бројеве у статистици. Ако се ти подаци нађу у јавности, могу бити злоупотребљени и нашкодити човеку чији су се подаци нашли у јавности.

# Анализа пословних процеса

У зависности од разних околнисти и догађаја како у свету, тако и у самој држави, попис становништва се организује једном у интервалу од 5 до 10 година. Популациона комисија Уједињених нација је предложила да се пописи становништва организују крајем деценије (2000, 2010, 2020) или почетком деценија (2001, 2011, 2021) због могућности упоређивања података између држава, али и компликованог процеса пописа становништва и израде статистике која може да траје изузетно дуго. Због тога сам процес треба упростити и убрзати што је више могуће.

Да би се процес убрзао и упростио, потребно је:

* Заменити старе технологије рада
* Извршавати цео процес у реалном времену

## Замена старих технологија рада

Технологија је достигла такав ниво да се из године у годину представља нова технологија која се до пре 10 година није могла замислити, или је била у домену научне фантастике, а сад се већ за годину дана замени нечим новим.

Зашто би се користили папир и оловка када данас имамо лаптопове, мобилне телефоне и таблете који су као мини рачунар. Интернет се такође развио, данас су брзине веће за неколико стонина пута него пре 10 година, па сваку информацију или новост можемо сазнати за неколико секунди.

## Извршавање у реалном времену

Анализа прикупљених података некада је трајала месецима код нас, код других и годинама јер је за то потребан велики број људи. Људи нису машине, некад им није дан, некад имају неки проблем, породични или су само гладни, па постоји и степен грешке приликом обраде података. Зато саму анализу треба препустити машинама јер иако је човек писао тај програм, програм је прошао неколико тестирања и ревизија па је самим тим степен грешке смањен скоро на минимум.

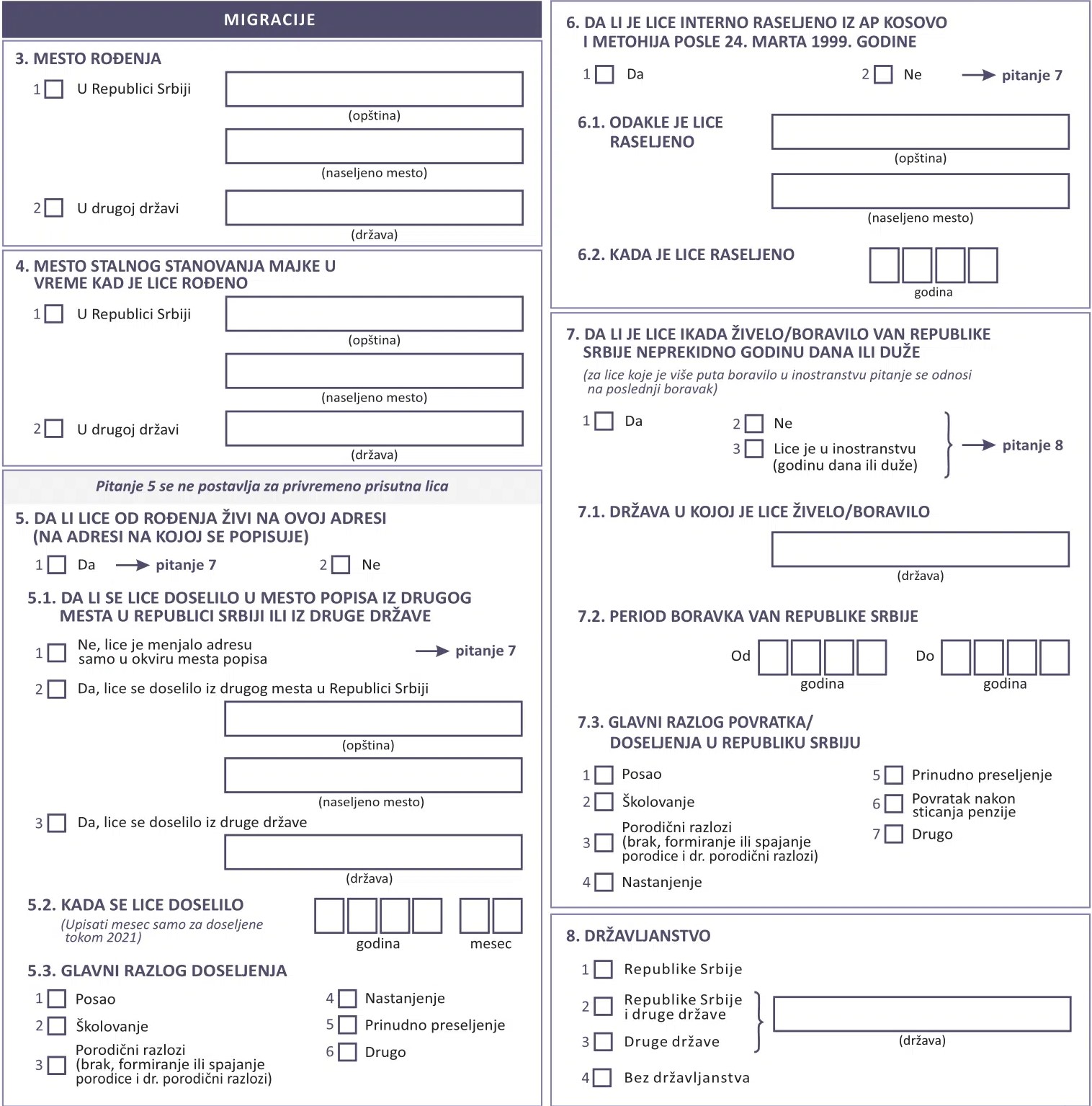
## Попис

За сам попис становништва је потребан пописивач, лаптоп са *Windows* оперативним системом (*Windows* 10 или новији) са програмом за попис и интернет на који ће бити повезан лаптоп.

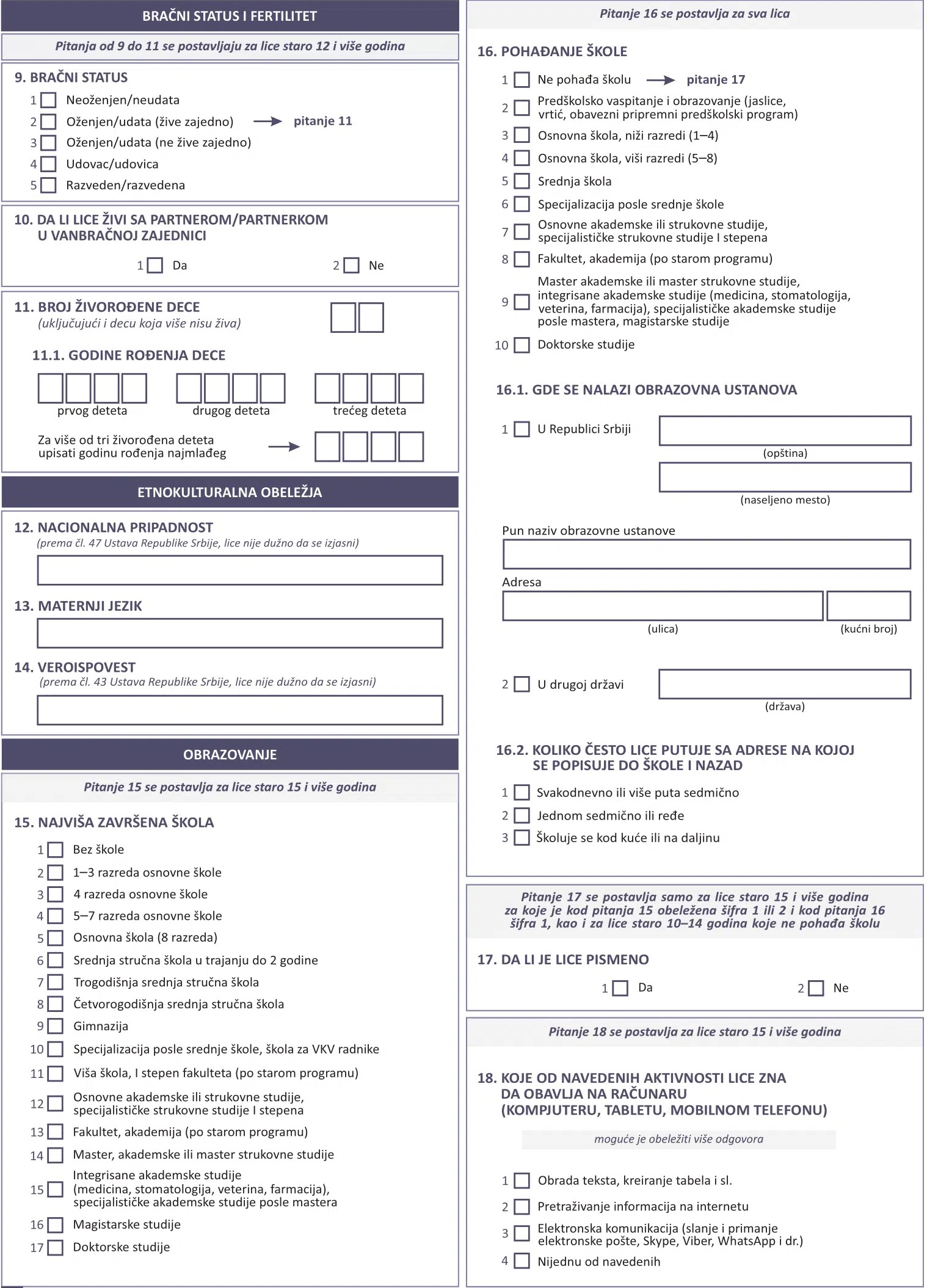
Да би становник постао пописивач, он се мора јавити на конкурс, бити пописан и добити креденцијале са којима ће се пријави у програм и вршити попис. Пописивач добије одређени реон и мора да иде од објекта до објекта да попише становнике.

Становник је дужан да прими пописивача, пописивачу да личне исправе и истинито одговори на сва постављена питања која се налазе на списку за попис. Питања се састоје из два дела:

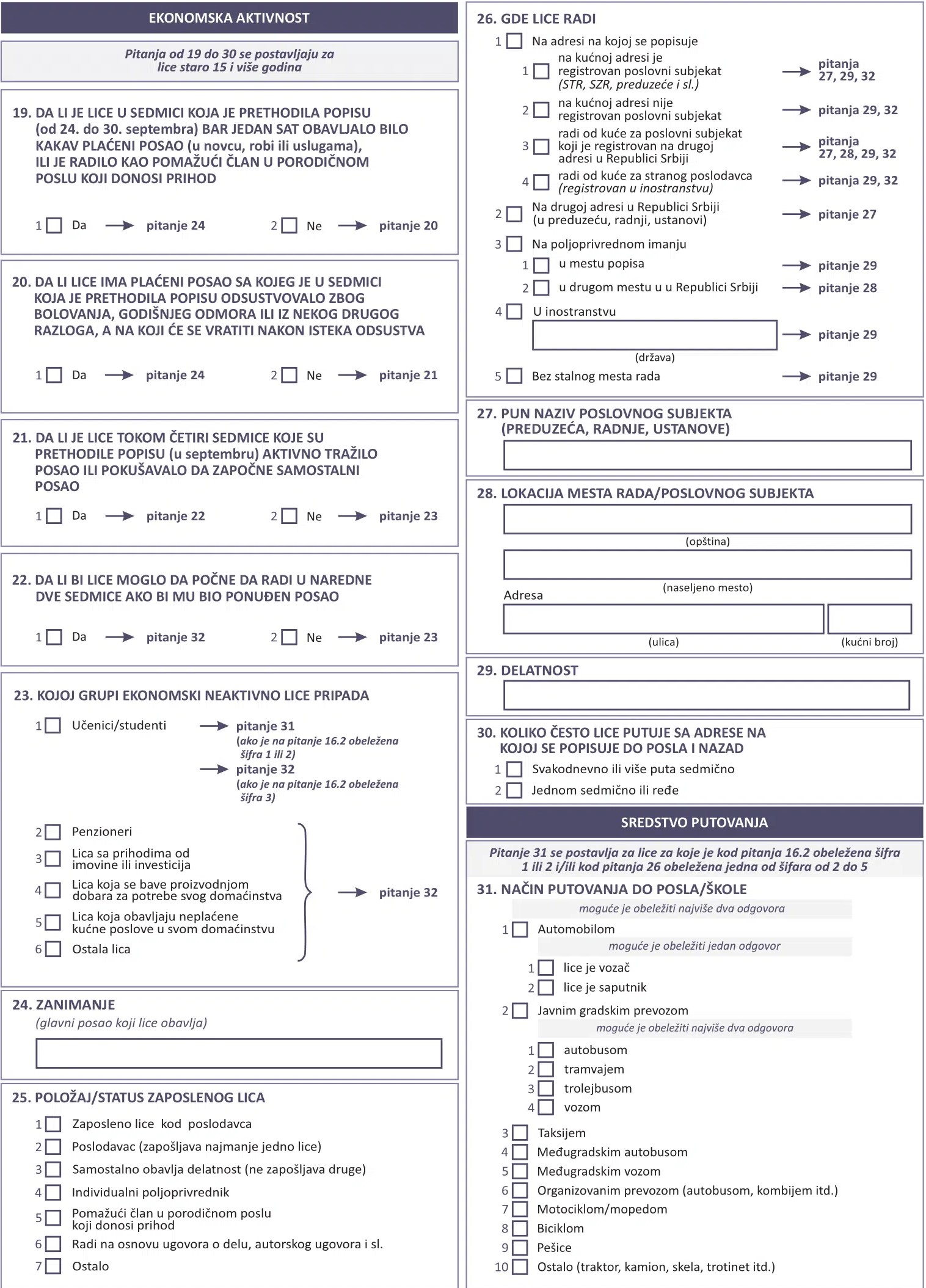
* Упитник за становништво
* Упитник за домаћинство и стан



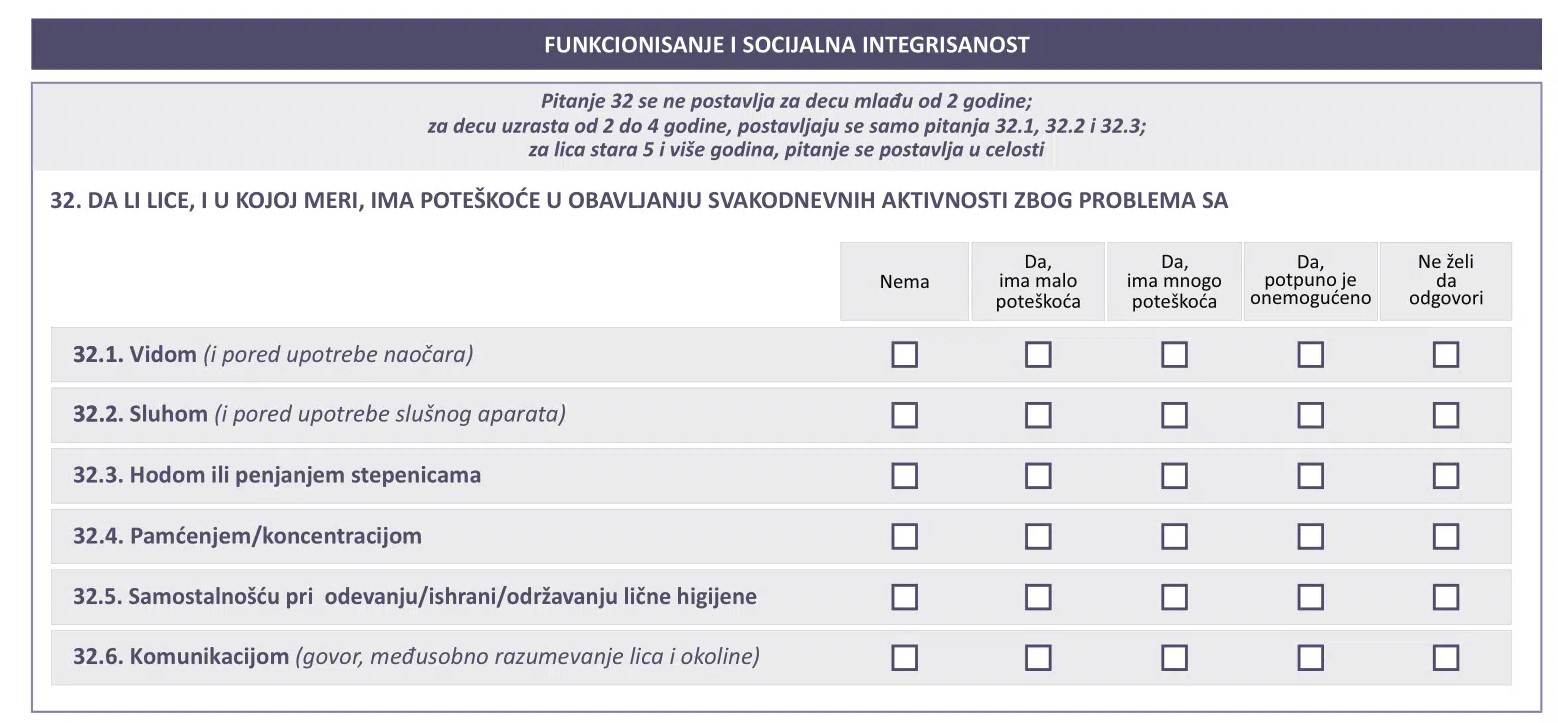
*Слика 2.1 – Питања из одељка „Миграције“*



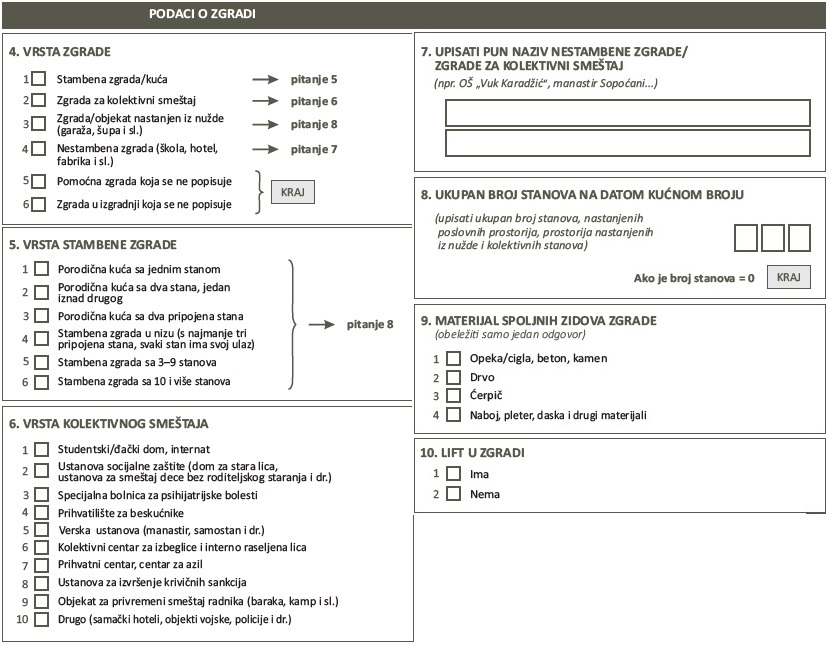
*Слика 2.2 – Питања из одељка „Брачни статус и фертилитет“, „Етнокултурна обележја“ и „Образовање“*



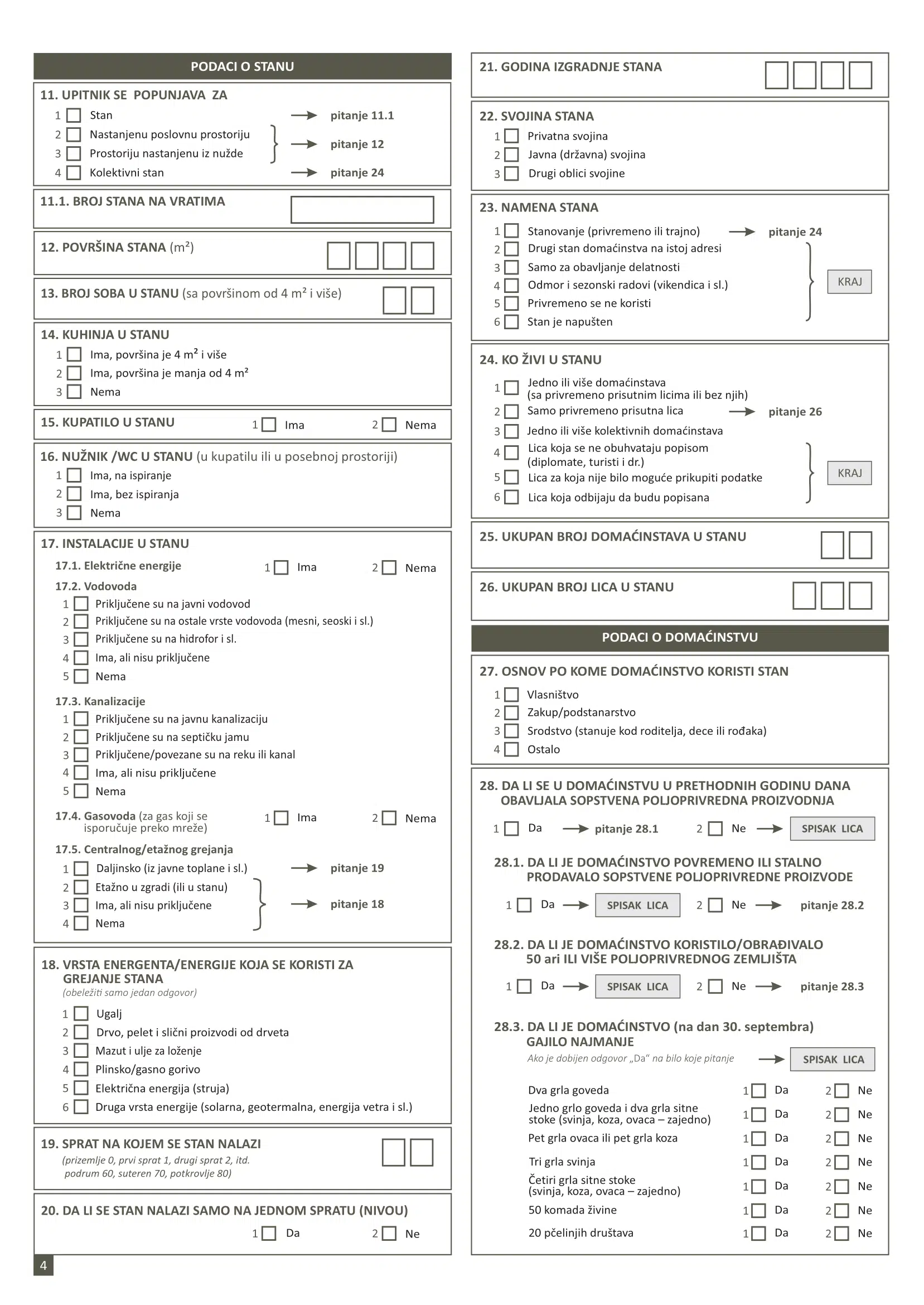
*Слика 2.3 – Питања из одељка „Економска активност“, и „Средство путовања“*



*Слика 2.4 – Питања из одељка „Функционисање и социјална интегрисаност“*



*Слика 2.5 – Питања из одељка „Подаци о згради“*

*Слика 2.6 – Питања из одељка „Подаци о stanu“ и „Подаци о домаћинству“*

# Пројектовање базе података

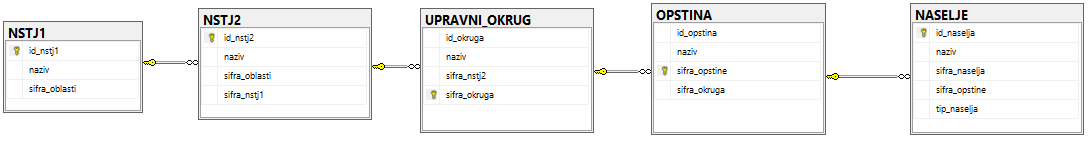
База податак је само срце целог система јер се сви подаци морају негде сачувати и обрадити. База је прављена у ***SSMS*** и писана помоћу ***SQL*** програмског језика, а подаци се прослеђују из апликативног дела програма.

Кључеви су бирани по слободном нахођењу и у зависности од величине самих табела. Табеле (ентитети) су пројектоване из питања и логички.

За ову базу постоје два тока података:

* Ток административне поделе
* Ток података питања

## Ток административне поделе



*Слика 3.1.1 – Дијаграм тока података везана за питања*

Дијаграм слике 3.1.1 је направљен на основу државне поделе и званичних података. Свака веза између табела је један према више. Све ове табеле су попуњене званичним подацима Републике Србије, што значи да подаци нису измишљени, него су реални и садрже све податке које садрже и званична документа (називи насеља, тип насеља, ком округу припада општина).

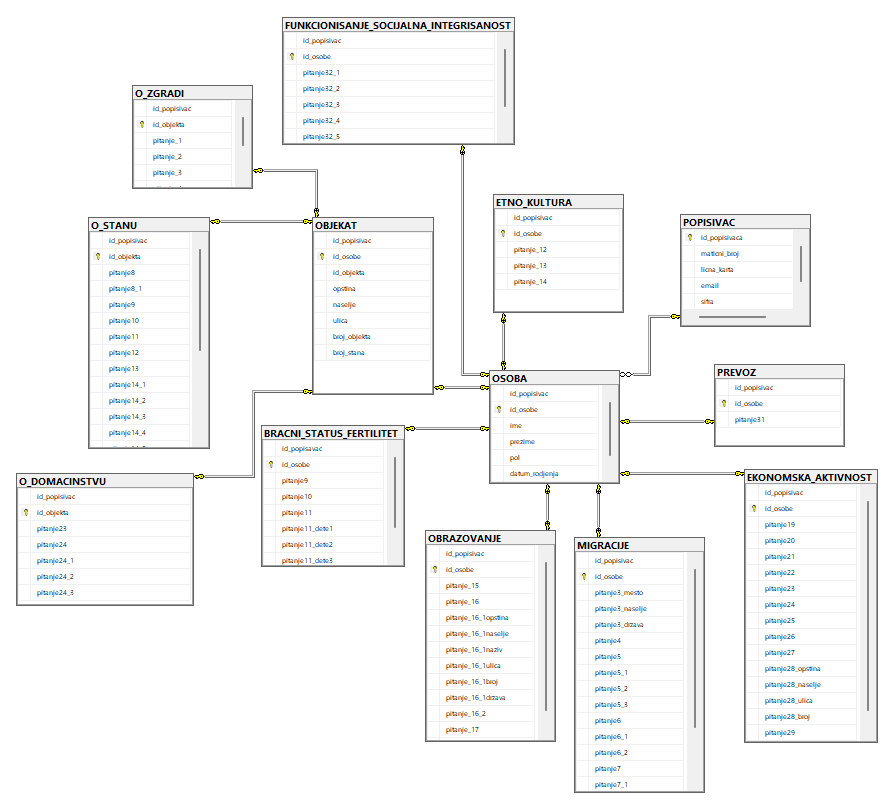
У овом случају, свака NSTJ1 територија (Номенклатура статистичких територијалних јединица) мора да има једну или више NSTJ2 територија, а NSTJ2 територија мора да има само једну NSTJ1 територију. NSTJ2 мора да има један или више управни округ, а управни округ мора да има само једну NSTJ2 територију. Управни округ мора да садржи једну или више општина, а општина мора да садржи само један управни округ. Општина мора да садржи једно или више насеља, а насеље мора да садржи само једну општину. Овде постоје примарни кључеви (sifra\_opstine, sifra\_okruga, id\_naselja), али и секундарни кључеви (sifra\_okruga у табели Општина је секундарни, а у табели Управни\_округ је примарни због везе један према више и sifra\_opstine u табели Општина је примарни, а у табели Насеље је секундарни кључ).

## Ток података питања

Ток података везана за питања је направљен на основу питања која се постављају особи која се пописује. Обзиром да се попис врши над становништвом и објектима за становање, имамо два основне табеле (ентитета):

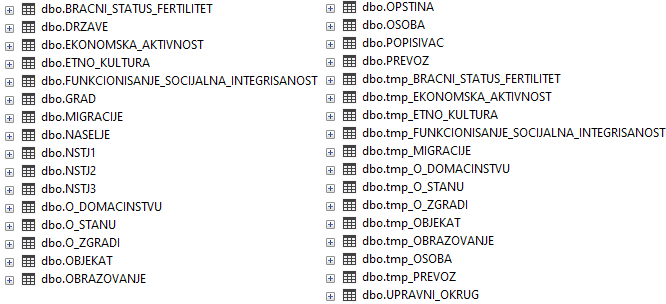
* OSOBA
* OBJEKAT

Свако питање припада неком одељку (миграција, етнокултура, образовање) и на основу одељака су направљене табеле. Све табеле су везане у везу звезда са табелама OSOBA или OBJEKAT и садрже примарни кључ id\_osobe или id\_objekta без постојања секундарних кључева. Свака табела је у вези један према један, то значи да табеле нису међусобно зависнe, али је свака табела зависна за табелу OSOBA или OBJEKAT, у зависности од самог питања, односно одељка ком питање припада.



*Слика 3.2.1 – Дијаграм тога података везана за питања*

Изузетак је веза табеле OSOBA са табелом POPISIVAC. У табели OSOBA постоји секундарни кључ id\_popisivac који је примарни кључ у табели POPISIVAC и ту је веза један према више, што значи да један пописивач може да попише једну или више особа, а једна особа мора да има само једног пописивача.



*Слика 3.1 – Све табеле које чине базу*

Свака табела везана за одељак са питањима има своју копију и подаци из упитника се прво уписују у привремену табелу (нпр. tmp\_osoba), а потом када особа да одговор на сва постављена питања, одговори који су везани за пописивача који је пописао особу, уписују се у сталне табеле. Ово све је урађено да би се спречиле грешке (прекид интернет везе, особа није више у могућности да заврши попис па ће се опет пописати итд).

У самој бази постоје и процедуре које су везане за брисање и додавање редова у табеле због лакше манипулације самим подацима из апликативног дела програма.

CREATE PROCEDURE [dbo].[brisanje\_tmp\_osoba]( @id\_popisivaca int)

as begin

delete from tmp\_OSOBA

where id\_popisivac=@id\_popisivaca  
end

На примеру процедуре brisanje\_tmp\_osoba се види да за прослеђен id\_popisivaca који је типа *integer*, у табели tmp\_OSOBA се проналази id\_popisivac који има вредност прослеђене вредности и брише задату вредност (ако постоји). Ова процедура је направљена да се табела не би препунила погрешним или старим подацима и све процедуре за брисање се позивају када се почиње нови попис.

create procedure [dbo].[upis\_reda\_osoba](@id\_popisivaca int)

as begin

insert into OSOBA

select \* from tmp\_OSOBA

where id\_popisivac=@id\_popisivaca

end

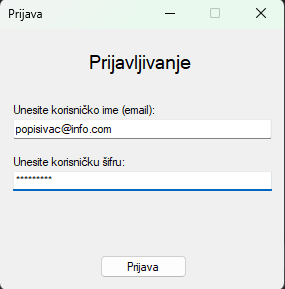
Процедури upis\_reda\_osoba се прослеђује такође id\_popisivaca који је типа integer. Процедура служи да би се после датих свих одговора која су постављена особи, упише све податке из табеле tmp\_OSOBA у табелу OSOBA за одговарајућег пописивача. Ове процедуре се понављају за сваку табелу.

# Практична реализација информационог система

Апликативни део пројекта је рађен у програму *Visual Studio 17* у програмском језику *C#*. *Microsoft* је произвођач и *MMSM* и *VS17*, самим тим су програми прилагођени један другог. Програм се састоји од *Windows Forms* и свака форма је везана за један одељак са питањима.

## Пријављивање

Када се програм покрене, отвара се прозор „Prijava“ у којем треба унети *e-mail* адресу (корисничко име) и шифру који корисник бира приликом аплицирања за пописивача.



*Слика 4.1.1 – Пријављивање пописивача*

Након што корисник унесе своје корисничко имеи лозинку, програм проверава у бази да ли корисничко име постоји, а затим ако постоји, да ли је и унета шифра иста као и она што се налази у бази за унето корисничко име.

var korIme =

(

from p in bd.POPISIVAC

where p.email == txtKorIme.Text

select p.email

).FirstOrDefault();

var sifra =

(

from p in bd.POPISIVAC

where p.email == txtKorIme.Text

select p.sifra

).FirstOrDefault();

if (sifra!=null)  
 psw = Dekodiranje(sifra);

Ова два упита су писана помоћу *LINQ* сета технологија *C#* програмског језика. У првом упиту се испитује да ли корисничко име које је унето у форми, постоји у табели POPISIVAC. У другом упиту се тражи шифра за задато корисничко име и ако постоји, позива се функција која декодира шифру. Шифра се декодира *Base64* врстом кодирања (декодирања).

if (korIme==txtKorIme.Text && psw == txtSifra.Text)

{

id\_popisivaca =

(

from p in bd.POPISIVAC

where p.email == txtKorIme.Text

select p.id\_popisivaca

).FirstOrDefault();

frmPocetna pocetna = new frmPocetna(id\_popisivaca);

this.Hide();

pocetna.ShowDialog();

this.Close();

}

else

{

MessageBox.Show("Niste uneli dobro korisničko ime ili šifru!!!");

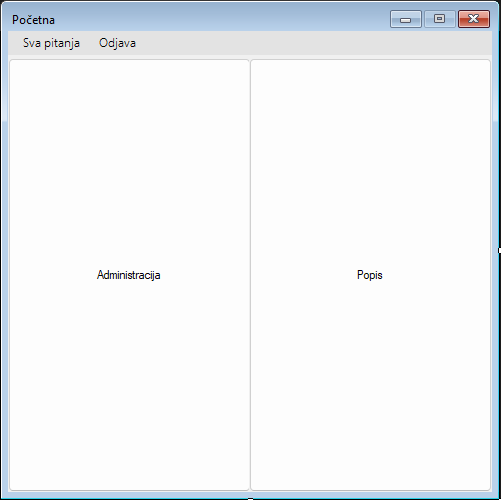
}

Ако се корисничко име из базе слаже са унетим корисничким именом у форми, и ако се унета шифра слаже са шифром из базе, позива се упит који враћа id\_popisivaca и отвара се нова форма. Ако се корисничко име и шифра не слажу, враћа се порука „Niste uneli dobro korisničko ime ili šifru!!!“.

## Почетна

Након успешне пријаве, отвара се прозор за избор опција:

* Administracija
* Popis



*Слика 4.2.1 – Почетна форма*

Осим избора претходне две опције, постоји и могућност да се прегледају сва питања кликом на дугме „*Sva pitanja*“ које отвара *pdf* документ који садржи сва питања. Такође постоји могућност да се пописивач одјави из програма кликом на дугме „Odjava“. Дугме „Odjava“ враћа упитник да пописивач сигурно жели да се одјави, ако је одговор потврдан, враћа корисника на форму за пријаву.

DialogResult result = MessageBox.Show("Da li želite da se odjavite?", "Unos", MessageBoxButtons.YesNo);

if (result == DialogResult.Yes)

{

MessageBoxManager.Unregister();

frmPrijava frm = new frmPrijava();

this.Hide();

frm.ShowDialog();

this.Close();

}

Дугме „Administracija“ води на форму где се врши одабир да ли пописивач жели да дода новог пописивача или преглед статистике.

## Администрација

Дугме администрација води на избор две опције:

* Dodaj popisivača
* Pregled statistike

Дугме „Dodaj popisivača“ води на форму где се врши додавање новог пописивача, а дугме „Pregled statistike“ води на одабир упита за преглед статистике. Такође постоји могућност прегледа питања и дугме „Početna“ које враћа пописивача на почетну страну.

private void svaPitanjaToolStripMenuItem\_Click(object sender, EventArgs e)

{

frmPitanja frm = new frmPitanja();

frm.ShowDialog();

}

private void početnaToolStripMenuItem\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Da li želite da se vratite na početnu stranu?", "Unos", MessageBoxButtons.YesNo);

if (result == DialogResult.Yes)

{

MessageBoxManager.Unregister();

frmPocetna frm = new frmPocetna(id\_popisivaca);

this.Hide();

frm.ShowDialog();

this.Close();

}

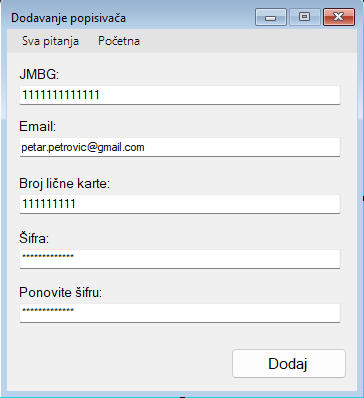
}

Функција *svaPitanjaToolStripMenuItem\_Click* отвара форму са контејнером који садржи *pdf* свих питања. Функција *početnaToolStripMenuItem\_Click* позива упит који пита пописивача да ли жели да се врати на почетну страну. Ако је одговор потврдан, отвара се форма frmPocetna којој се прослеђује id\_popisivaca да би у програму остало записано који пописивач је пријављен. По истом или сличном принципу раде све остале форме.

## Додај пописивача

Одабиром опције „Dodaj popisivača“ се отвара форма која за унос:

* Јединственог матичног броја грађана (ЈМБГ)
* Email адресе будућег пописивача
* Број личне карте будућег пописивача
* Шифре коју уноси будући пописивач
* Поновљену шифру из претходног корака



*Слика 4.4.1 – Форма за додавање пописивача*

Поље за унос јединственог матичног броја грађана је ограничено на 13 карактера, односно мора да има 13 карактера, иначе додавање пописивача неће бити неуспешно. Поље за унос *e-mail* адресе је повезано са функцијом која врши валидирање *e-mail* адресе, односно да ли постоји корисничко име, знак @, и после знака доменски део.

private static bool Validiranje(string email)

{

var validno = true;

try

{

var emailAddress = new MailAddress(email);

}

catch

{

validno = false;

}

return validno;

}

Функцији се прослеђује *e-mail* адреса, а она уз помоћ интегрисане функције покуша да валидира адресу, ако успешно, вратиће *true* вредност за варијаблу validno, а ко није успешно, варијабла validno ће добити вредност *false*. Број личне карте је такође ограничен на минимални и максимално број и износи 9 карактера.

Поље за унос шифре мора да испуни неколико правила:

* Шифра мора да има минимално 8 караткера
* Шифра мора да садржи број
* Шифра мора да садржи специјални карактер

Унос шифре се врши тако што пописивач проследи уређај особи која жели да постане пописивач, особа унесе шифру која одговара правилима, затим исту шифру унесе у друго поље и враћа уређај пописивачу. У приказу шифре, сваки караткер се замењује карактером \* тако да пописивач не може да види шифру, тако се смањује могућност злоупотребе. Шифра се кодира *Base64* врстом кодирања као мера заштите да ако база за подацима заврши у јавности, да не буде тако лако декодирати. Накнадном провером уз помоћ јединственог матичног броја грађана се проверава да ли је особа пописана и да ли је већ пописивач. Ако је особа пописана и није пописивач, додаје се у базу у табелу POPISIVAC, а ко је неки од два услова другачији, враћа одговарајућу поруку.

{

var tmp =

from n in bd.OSOBA

where n.maticni\_broj == txtJMBG.Text

select n.id\_osobe;

var tmp1 =

from n in bd.POPISIVAC

where n.maticni\_broj == txtJMBG.Text

select n.id\_popisivaca;

int postojiOsoba = tmp.Count();

int postojiPopisivac = tmp1.Count();

if ( postojiOsoba == 0)

{

MessageBox.Show("Prvo treba uraditi popis za zadatu osobu.");

}

else if (postojiPopisivac > 0)

{

MessageBox.Show("Postoji popisivač sa zadatim matičnim brojem.");

}

else

{

var tmp\_id =

(

from n in bd.POPISIVAC

select n.id\_popisivaca

).LastOrDefault();

red.id\_popisivaca = tmp\_id + 1;

red.maticni\_broj = txtJMBG.Text;

red.licna\_karta = txtLK.Text;

red.email = txtEmail.Text;

red.sifra = Kodiranje(txtSifra.Text);

bd.POPISIVAC.AddPOPISIVACRow(red);

popisivacAdapter.Update(bd.POPISIVAC);

MessageBoxManager.Unregister();

frmPocetna frm = new frmPocetna(id\_popisivaca);

this.Hide();

frm.ShowDialog();

this.Close();

}

}

У претходном делу кода се врше провере и ако је све успешно, пописивач добија свој id и сви подаци се уписују у табелу POPISIVAC.

## Статистика

Дугме „Pregled statistike“ отвара форму која садржи наслове за избор статистике за одговарајући наслов и у наредном тексту ће бити описани неки од:

* Становништво према националној припадности и полу
* Становништво према национално припадности
* Становнишво према старости и полу
* Просечна старост становнишва према полу
* Становнишво према вероисповести
* Становнишво према матерњем језику
* Породице према типу
* Станови према површини
* Грејање у настањеним становима без централног грејања

### Становништво према националној припадности и полу

Притиском на одређени наслов (у овом случају ћемо узети „*Становништво према националној припадности и полу*“), прослеђује се његова варијабла „stanovnistvo1\_1“ у форму frmUpit. У тој форми се налази контејнер који за одређени упит над табелама у бази, враћа одговарајуће податке.

case "stanovnistvo1\_1":

dt = new DataTable();

dt.Columns.Add(" ", typeof(string));

dt.Columns.Add("Ukupno", typeof(string));

dt.Columns.Add("Ukupno muškaraca", typeof(string));

dt.Columns.Add("Ukupno žena", typeof(string));

dt.Columns.Add("Ukupno Srba", typeof(string));

dt.Columns.Add("Ukupno Srba (muškaraca)", typeof(string));

dt.Columns.Add("Ukupno Srba (žena)", typeof(string));

dt.Columns.Add("Ukupno - ostale nacije", typeof(string));

dt.Columns.Add("Ukupno - ostale nacije (muškaraca)", ));

dt.Columns.Add("Ukupno - ostale nacije (žena)", typeof(string));

pripadnost\_pol(dt);

dgv1.DataSource = dt;

break;

За потребе упита се користи switch case исказ јер програм много брже проналази који део кода му треба и изврши га. Први део кода се користи да би се направила табела са одговарајућим колонама и позива се функција pripadnost\_pol која садржи упите и којој се прослеђује направљена табела. Други пример кода је део великог упита који се извршава над неколико табела и са неколико услова. У овом случају услови су назив округа, да је одговор на дванаесто питање Србин/Српкиња и да је особа мушког пола. Варијабла ukupno\_srba\_muskaraca после извршеног упита добија целобројну вредност, јер упит враћа број особа које испуњавају горе наведене услове.

ukupno\_srba\_muskaraca =

(

from o in bd.OSOBA

join ek in bd.ETNO\_KULTURA on o.id\_osobe equals sobe

join ob in bd.OBJEKAT on o.id\_osobe equals ob.id\_osobe

join op in bd.OPSTINA on ob.opstina equals op.naziv

join uo in bd.UPRAVNI\_OKRUG on op.sifra\_okruga equals

where uo.naziv == red["naziv"].ToString() && ek.pitanje\_12 =="Srbin/Srpkinja" && o.pol == "Muško"

select o.id\_osobe).Count();

Табела 4.5.1.1 – Пример становништво према националној припадности и полу

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Ukupno | Ukupno muškaraca | Ukupno žena | Ukupno Srba |
| Republika Srbija | 2 | 2 | 0 | 2 |
| Grad Beograd | 0 | 0 | 0 | 0 |
| Severnobački upravni okrug | 0 | 0 | 0 | 0 |
| Srednjobanatski upravni okrug | 0 | 0 | 0 | 0 |
| Severnobanatski upravni okrug | 0 | 0 | 0 | 0 |
| Južnobanatski upravni okrug | 0 | 0 | 0 | 0 |
| Zapadnobački upravni okrug | 0 | 0 | 0 | 0 |
| Južnobački upravni okrug | 0 | 0 | 0 | 0 |
| Sremski upravni okrug | 2 | 2 | 0 | 2 |

У табели 4.5.1 је дат део приказа извршених упита и састоји се из поделе према окрузима у Србији, а највећа подела је сама држава, па затим по управним окрузима.

### Становнишво према старости и полу

За наслов „*Становнишво према старости и полу*“ се прослеђује варијабла „stanovnistvo2\_1“.

case "stanovnistvo2\_1":

dt = new DataTable();

dt.Columns.Add(" ", typeof(string));

dt.Columns.Add(" ", typeof(string));

dt.Columns.Add("Ukupno", typeof(string));

dgv1.DataSource = dt;

nacionalna\_pripadnost(dt);

break;

За ову табелу су потребне само три колоне и позива се фунцкија nacionalna\_pripadnost којој се прослеђује направљена табела. У тој фунцкији се извршавају упити који пребројавају особе са одређени опсег година, прво све особе, па затим по половима.

case 2:

lista.Add("15–29 godina");

tmp =

(

from ek in bd.ETNO\_KULTURA

join o in bd.OSOBA on ek.id\_osobe equals o.id\_osobe

where DateTime.Today.Year - o.datum\_rodjenja.Year > 14 && DateTime.Today.Year - o.datum\_rodjenja.Year < 30 && o.pol == pol

select ek.id\_osobe

).Count();

lista.Add(tmp.ToString());

dt.Rows.Add(lista.ToArray());

break;

Табела 4.5.2.1 – Пример табеле „Становнишво према старости и полу“

|  |  |  |
| --- | --- | --- |
|  |  | Ukupno |
| Ukupno | Ukupno | 2 |
| Ukupno | 0–14 godina | 0 |
| Ukupno | 15–29 godina | 1 |
| Ukupno | 30–49 godina | 0 |
| Ukupno | 50–64 godina | 1 |
| Ukupno | 65–84 godina | 0 |
| Ukupno | 85 i više godina | 0 |
| Muško | Ukupno | 2 |
| Muško | 0–14 godina | 0 |
| Muško | 15–29 godina | 1 |
| Muško | 30–49 godina | 0 |
| Muško | 50–64 godina | 1 |
| Muško | 65–84 godina | 0 |
| Muško | 85 i više godina | 0 |

Из табеле 4.5.2.1 се може прочитати да су тренутно пописане две особе, да је једна особа старости између 15 и 29 година, а да је друга особа старости између 50 и 64 година.

### Просечна старост становнишва према полу

За „*Просечна старост становнишва према полу*“ се прослеђује варијабла stanovnistvo2\_1 и овај случај је специфичан јер се узима просек година.

case "stanovnistvo2\_2":

dt = new DataTable();

dt.Columns.Add(" ", typeof(string));

dt.Columns.Add("Ukupno", typeof(string));

dt.Columns.Add("Muško", typeof(string));

dt.Columns.Add("Žensko", typeof(string));

lista = new List<string>();

double tmp;

tmp =

(

from o in bd.OSOBA

select DateTime.Today.Year - o.datum\_rodjenja.Year

).DefaultIfEmpty().Average();

lista.Add("Republika Srbija");

lista.Add(tmp.ToString());

tmp =

(

from o in bd.OSOBA

where o.pol == "Muško"

select DateTime.Today.Year - o.datum\_rodjenja.Year

).DefaultIfEmpty().Average();

lista.Add(tmp.ToString());

tmp =

(

from o in bd.OSOBA

where o.pol == "Žensko"

select DateTime.Today.Year - o.datum\_rodjenja.Year

).DefaultIfEmpty().Average();

lista.Add(tmp.ToString());

dt.Rows.Add(lista.ToArray());

prosek\_godina(dt);

dgv1.DataSource = dt;

break;

Дати код служи да се направи табела са 4 колоне и израчуна просек година за све особе и за сваки пол посебно тако што се од садашње године одузме година рођења особе. Затим се позива функција prosek\_godina којој се прослеђује направљена табела. Функција је направљена да би код био прегледнији.

private void prosek\_godina(DataTable dt)

{

List<string> lista = new List<string>();

double tmp;

foreach (DataRow red in bd.NSTJ1.Rows)

{

foreach (DataColumn dc in bd.NSTJ1.Columns)

{

if (dc.ColumnName == "naziv")

{

lista = new List<string>();

lista.Add(red["naziv"].ToString());

tmp =

(

from o in bd.OSOBA

join ob in bd.OBJEKAT on o.id\_osobe equals ob.id\_osobe

join op in bd.OPSTINA on ob.opstina equals op.naziv

join uo in bd.UPRAVNI\_OKRUG on op.sifra\_okruga equals uo.sifra\_okruga

join ns2 in bd.NSTJ2 on uo.sifra\_nstj2 equals ns2.sifra\_oblasti

join ns1 in bd.NSTJ1 on ns2.sifra\_nstj1 equals ns1.sifra\_oblasti

where ns1.naziv == red["naziv"].ToString()

select DateTime.Today.Year - o.datum\_rodjenja.Year

).DefaultIfEmpty().Average();

lista.Add(tmp.ToString());

tmp =

(

from o in bd.OSOBA

join ob in bd.OBJEKAT on o.id\_osobe equals ob.id\_osobe

join op in bd.OPSTINA on ob.opstina equals op.naziv

join uo in bd.UPRAVNI\_OKRUG on op.sifra\_okruga equals uo.sifra\_okruga

join ns2 in bd.NSTJ2 on uo.sifra\_nstj2 equals ns2.sifra\_oblasti

join ns1 in bd.NSTJ1 on ns2.sifra\_nstj1 equals ns1.sifra\_oblasti

where ns1.naziv == red["naziv"].ToString() && o.pol == "Muško"

select DateTime.Today.Year - o.datum\_rodjenja.Year

).DefaultIfEmpty().Average();

lista.Add(tmp.ToString());

tmp =

(

from o in bd.OSOBA

join ob in bd.OBJEKAT on o.id\_osobe equals ob.id\_osobe

join op in bd.OPSTINA on ob.opstina equals op.naziv

join uo in bd.UPRAVNI\_OKRUG on op.sifra\_okruga equals uo.sifra\_okruga

join ns2 in bd.NSTJ2 on uo.sifra\_nstj2 equals ns2.sifra\_oblasti

join ns1 in bd.NSTJ1 on ns2.sifra\_nstj1 equals ns1.sifra\_oblasti

where ns1.naziv == red["naziv"].ToString() && o.pol == "Žensko"

select DateTime.Today.Year - o.datum\_rodjenja.Year

).DefaultIfEmpty().Average();

lista.Add(tmp.ToString());

dt.Rows.Add(lista.ToArray());

}

}

}

}

Функција prosek\_godina садржи две *for* које пролазе кроз редове и колоне NSTJ1 табеле и резултате три упита смешта у листу. У ова три упита имамо један до два услова, један услов се понавља кроз сва три упита, а то је да питање да ли је ћелија једнака називу из табеле и у два упита постоји услов да ли је пол мушки или женски.

Табела 4.5.3.1 - Просечна старост становнишва према полу

|  |  |  |  |
| --- | --- | --- | --- |
|  | Ukupno | Muško | Žensko |
| Republika Srbija | 38 | 38 | 0 |
| Srbija-sever | 38 | 38 | 0 |
| Srbija-jug | 0 | 0 | 0 |

У табели 4.5.3.1 се види да укупан просек година и просек година за мушкарце исти јер су пописане само две особе. То је тачно јер су пописане две особе старости 23 и 53 година.

### Станови према површини

За „*Станови према површини*“ се прослеђује варијабла stanovi\_1 и садржи највећу функцију у целом програму јер се пролази кроз изузетно велик број табели и података. case "stanovi\_1":

dt = new DataTable();

dt.Columns.Add("Teritorija", typeof(string));

dt.Columns.Add("Tip naselja", typeof(string));

dt.Columns.Add("Ukupno", typeof(string));

dt.Columns.Add("do 30m²", typeof(string));

dt.Columns.Add("31m² - 40m²", typeof(string));

dt.Columns.Add("41m² - 50m", typeof(string));

dt.Columns.Add("51m² - 60m²", typeof(string));

dt.Columns.Add("61m² - 80m²", typeof(string));

dt.Columns.Add("81m² - 100m²", typeof(string));

dt.Columns.Add("101m² - 120m²", typeof(string));

dt.Columns.Add("121m² - 150m²", typeof(string));

dt.Columns.Add("151m² i više", typeof(string));

dgv1.DataSource = dt;

povrsina(dt)break;

Ова табела садржи 12 колона са распонима површине стана и позива функцију povrsina(dt) којој се прослеђује направљена табела.

tmp =

(

from os in bd.O\_STANU

join ob in bd.OBJEKAT on os.id\_objekta equals ob.id\_objekta

join op in bd.OPSTINA on ob.opstina equals op.naziv

join n in bd.NASELJE on ob.naselje equals n.naziv

join uo in bd.UPRAVNI\_OKRUG on op.sifra\_okruga equals uo.sifra\_okruga

where int.Parse(os.pitanje9) > 120 && int.Parse(os.pitanje9) < 151 && n.tip\_naselja == tip && red["naziv"].ToString() == uo.naziv

select os.pitanje9

).Count();

Овај упит је део функције и упит саджри неколико табела и неколико услова. Одговор на питање 9 треба да буде веће од 120 и мање од 151 квадратних метара, да тип насеља буде као из *for* петље (градско или остали типови насеља) и да назив из ћелије буде једнак називу управног округа.

Табела 4.5.4.1 – Примет табеле „Станови према површини“

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Teritorija | Tip naselja | Ukupno | do 30m² | 31m² - 40m² | 41m² - 50m² | 51m² - 60m² | 61m² - 80m² | 81m²  - 100m² | 101m² - 120m² | 121m² - 150m² | 151m² i  više |
| Republika Srbija | Ukupno | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Republika Srbija | Gradska naselja | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Republika Srbija | Ostala | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Grad Beograd | Ukupno | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Grad Beograd | Gradska naselja | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Grad Beograd | Ostala | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Sremski upravni okrug | Ukupno | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Sremski upravni okrug | Gradska naselja | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Sremski upravni okrug | Ostala | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

У табели 4.5.4.1 можемо видети да је пописан само један објекат у Сремском управном округу квадратуре између 41m2 и 50m2 и то је тачан податак јер је пописан стан квадратуре 49m2.

### Грејање у настањеним становима без централног грејања

case "stanovi\_2":

dt = new DataTable();

dt.Columns.Add("Teritorija", typeof(string));

dt.Columns.Add("Tip naselja", typeof(string));

dt.Columns.Add("Ukupno", typeof(string));

dt.Columns.Add("Ugalj", typeof(string));

dt.Columns.Add("Drvo i sl.", typeof(string));

dt.Columns.Add("Mazut i ulje za loženje", typeof(string));

dt.Columns.Add("Plinsko/gasno gorivo", typeof(string));

dt.Columns.Add("Električna energija", typeof(string));

dt.Columns.Add("Druga vrsta energije", typeof(string));

dgv1.DataSource = dt;

grejanje(dt);

break;

За наслов „*Грејање у настањеним становима без централног грејања*“ се прослеђује варијабла stanovi\_2 и прави се табела са 9 колона. Затим се позива функција grejanje и прослеђује направљена табела.

tmp =

(

from os in bd.O\_STANU

join ob in bd.OBJEKAT on os.id\_objekta equals ob.id\_objekta

join op in bd.OPSTINA on ob.opstina equals op.naziv

join n in bd.NASELJE on ob.naselje equals n.naziv

join uo in bd.UPRAVNI\_OKRUG on op.sifra\_okruga equals uo.sifra\_okruga

join nstj2 in bd.NSTJ2 on uo.sifra\_nstj2 equals nstj2.sifra\_oblasti

where os.pitanje15 == grejanje && n.tip\_naselja == tip && nstj2.naziv == red["naziv"].ToString()

select os.id\_objekta

).Count();

У овом упиту се пролази кроз неколико табела да би се упит извршио над NSTJ2 табелом уз услов да одговор на питање 15 буде једнак типу грејања из *for* петље која се односи на грејање, да је тип насеља једнак оном из *for* петље која се односи на насеља и да је назив области једна податку из ћелије.

Табела 4.5.5.1 – Пример табеле „Грејање у настањеним становима без централног грејања“

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Teritorija | Tip naselja | Ukupno | Ugalj | Drvo i sl. | Mazut i ulje za loženje | Plinsko/gasno gorivo | Električna energija | Druga vrsta energije |
| Beogradski region | Ukupno | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Beogradski region | Gradska naselja | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Beogradski region | Ostala | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Region Vojvodine | Ukupno | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| Region Vojvodine | Gradska naselja | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Region Vojvodine | Ostala | 1 | 0 | 0 | 0 | 1 | 0 | 0 |

У табели 4.5.5.1 можемо видети да је пописан један објекат у региону Војводине и да се тај стан греје на плин или гас и не припада градском насељу. Исказ је тачан јер је стан пописан у Новим Бановцима који припадају региону Војводине и стан користи гас као топлотни извор.

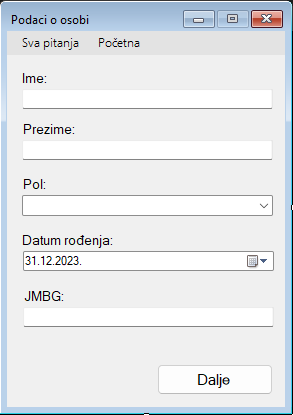
## Попис

Кликом на дугме „Popis“ са почетног екрана се приступа формама које садрже питања. Свака форма је посебан одељак (нпр. миграције, етнокултура, подаци о стану и др.) и за сваки одељак су везане две одговарајуће табеле. За одељак „Миграције“ су повезане табеле MIGRACIJE и tmp\_MIGRACIJE.

### Подаци о особи

У одељку „Подаци о особи“ се уносе опште информације везане за особу која се пописује као и то су:

* Име
* Презиме
* Пол
* Датум рођења
* Јединствени матични број грађана



*Слика 4.6.1.1 – Подаци о особи*

Наслов сваког поља јасно указује који податак треба унети у одређено поље. За датум рођења је као почетни датум постављен датум 1.1.1850. а крајњи се ажурира свакодневно и узима се датум коришћења програма. Јединствени матични број грађана је ограничен на 13 карактера и обзиром да је тај податак јединствен за сваку особу, програм помоћу њега проверава да ли је особа са унетим матичним бројем већ пописана. Кликом на дугме „Dalje“ програм проверава претходни податак и да ли је свако поље попуњено.

var tmp = from n in bd.OSOBA

where n.maticni\_broj == txtJMBG.Text

select n.maticni\_broj;

int tmp\_maticni = tmp.Count();

if (tmp\_maticni > 0)

{

MessageBox.Show("Zadati matični broj već postoji.");

}  
else if (txtJMBG.Text.Length != 13 || !txtJMBG.Text.All(char.IsDigit))

{

MessageBox.Show("Potrebno je uneti pravilno JMBG.");

}

else

{

DialogResult result = MessageBox.Show("Da li ste sigurni?", "Unos", MessageBoxButtons.YesNo);

if (result == DialogResult.Yes)

{

BazaDiplomskog.tmp\_OSOBARow red = bd.tmp\_OSOBA.Newtmp\_OSOBARow();

int id = tmp\_id + 1;

red.id\_popisivac = id\_popisivaca;

red.id\_osobe = id;

red.ime = txtIme.Text;

red.prezime = txtPrezime.Text;

red.pol = cmbPol.SelectedItem.ToString();

red.datum\_rodjenja = dtpDatumRodjenja.Value;

red.maticni\_broj = txtJMBG.Text;

bd.tmp\_OSOBA.Addtmp\_OSOBARow(red);

tmp\_osobaAdapter.Update(bd.tmp\_OSOBA);

MessageBoxManager.Unregister();

frmMigracije forma = new frmMigracije(id, id\_popisivaca);

this.Hide();

forma.ShowDialog();

this.Close();

}

}

У коду је приказана провера услова за јединствени матични број грађана и упис у табелу. Прво се у табели OSOBA проверава да ли постоји јединствени матични број грађана, ако не постоји, проверава да ли је у одговарајућем пољу унет низ од 13 карактера и да ли је сваки карактер број. Ако су испуњени сви услови, појављује се прозор са питањем да ли су то коначни одговори и ако јесу, одговори се уписују у табелу tmp\_OSOBA и отвара се одељак за миграцију. У коду постоји и позив класе *MessageBoxManager* која је преузета са интернета и служи да се модификују одговори за класу *MessageBox*. Одговори се дефинишу:

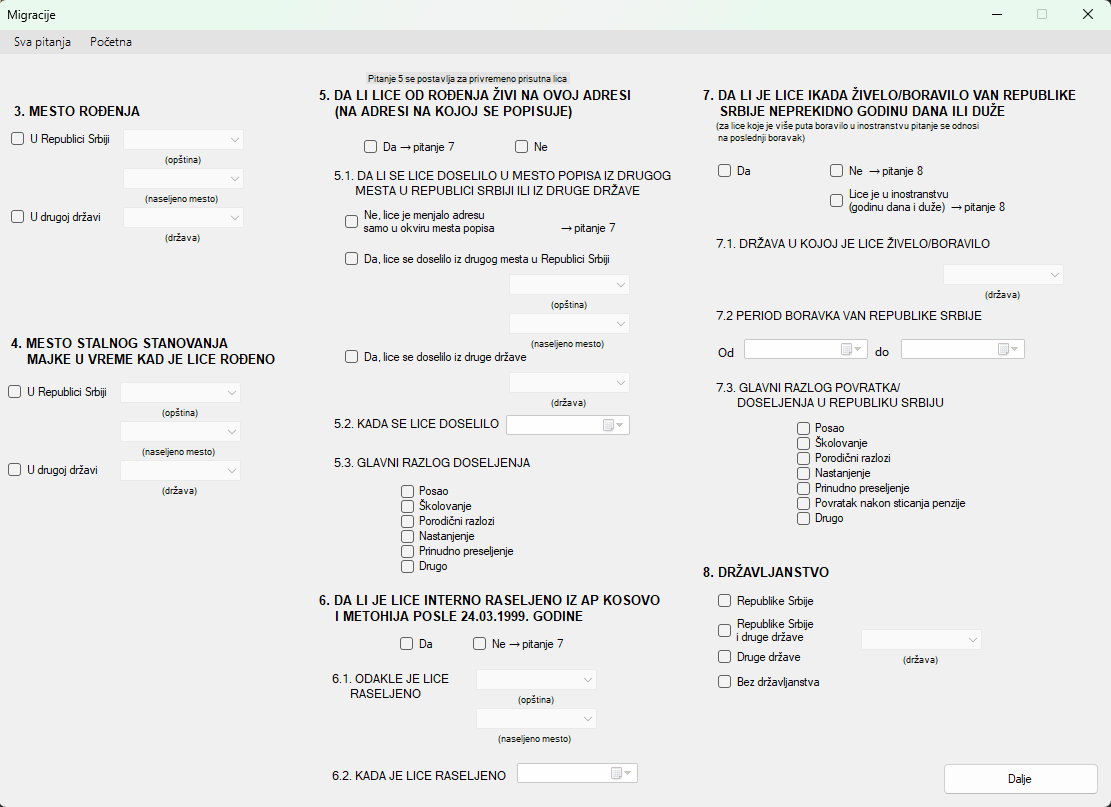
MessageBoxManager.Yes = "Da";

MessageBoxManager.No = "Ne";

Свака форма садржи класу *MessageBoxManager* и на почетку се дефинише изразом *MessageBoxManager.Register(),* а на крају се мора обрисати дефиниција изразом *MessageBoxManager.Unregister()*.

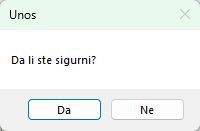
### Миграција

Одељак везан за миграцију садржи 6 питања и 8 потпитања на која, у зависности од одговора на питање за које је везано потпитање, се неће морати одговарати. Питања се односе на место рођења и пребивалишта особе која се пописује.



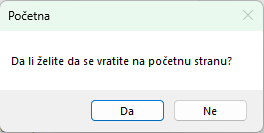
*Слика 4.6.2.1 – форма „Миграције“*

За питање број 3, као и за сва остала слична питања, ако особа одговори да је рођена у Републици Србији, у пољу „opština“ учитавају се све општине Републике Србије (унете у базу) и за сваку општину се у пољу „naseljeno mesto“ учитавају насеља која припадају изабраној општини (сва насељена места се налазе у бази). Ако испитаник одговори да је рођен у другој држави, у пољу „država“ се учитавају све државе (такође из базе). Одговори су међусобно зависни, тако да се не може дати одговор на питање 4 ако нисте одговорили на питање 3. Кликом на дугме „Sva pitanja“, отвара се *pdf* са свим питањима.



*Слика 4.6.2.2 – Упит после датих одговора*

Када особа одговори на сва питања, откључава се дугме „Dalje“, пописивач кликне на то дугме, затим се појави порука са упитом да ли су ти одговори коначни (слика 4.6.2.2), ако јесу, отвара се следећа форма са питањима.

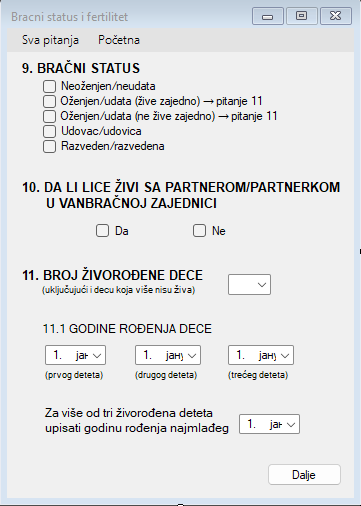


*Слика 4.6.2.3 – Упит за почетну страну*

У сваком тренутку пописник може да прекине попис кликом на дугме „Početna“,затим се појави упит са питањем да ли пописник жели да оде на почетну страну(слика 4.6.2.3), ако је одговор потврдан, отвара се почетна страна.

### Брачни статус и фертилитет

Брачни статус и фертилитет садржи 3 питања и 1 једно потпитање. Питања се односе на брачни статус и породицу особе која се пописује.



*Слика 4.6.3.1 – Форма „Брачни статус и фертилитер“*

Сва питања су међусобно зависна, у случају да особа која се пописује одговори на питање 11 да има једно дете, у питању 11.1 ће бити дуступно само поље за упис године рођења првог детета. Исто важи и за одговоре за двоје и троје деце, а ако особа има више деце, у питању 11.1 ће бити доступно само упис године рођења најмлађег детета. Кликом на дугме „Dalje“ појави се порука са упитом да ли су ти одговори коначни, ако јесу, отвара се следећа форма са питањима. У било ком тренутку пописивач може да оде на почетну страну или да отвори *pdf*  са свим питањима.

if ((clb9.SelectedIndex == 1 || clb9.SelectedIndex == 2) && cmb11.Text == string.Empty)

{

MessageBox.Show("Nedostaje odgovor za pitanje 11!!!");

}

else if (!cb10Da.Checked && !cb10Ne.Checked && clb9.SelectedIndex != 1 && clb9.SelectedIndex != 2)

{

MessageBox.Show("Nedostaje odgovor za pitanje 10!!!");

}

else if (cmb11.Text == string.Empty)

{

MessageBox.Show("Nedostaje odgovor za pitanje 11!!!");

}

else if (cmb11.SelectedIndex < 0)

{

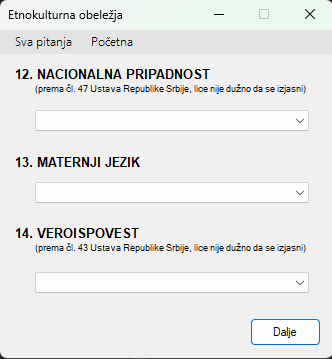
MessageBox.Show("Nedostaje odgovor za pitanje 11.1!!!");

}

Осим што су питања међусобно зависна, за овај одељак постоји и провера да ли су дати одговори на сва питања и да ли су сви услови испуњени.

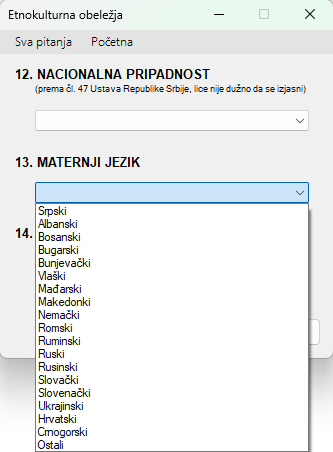
### Етнокултурна обележја

Одељак за етнокултурна обележја се односи на питања која се тичу саме националности и вере особе која се пописује. Одељак садржи 3 питања и за сва 3 питања су дефинисани одговори у падајућем менију. На једно питање је обавезно одговорити, а на остала два, према уставу Републике Србије, особа није дужна да се изјасни.



*Слика 4.6.4.1 – Форма „Етнокултурна обележја“*

Одговори су дефинисани у подешавањима самих падајућих менија (*ComboBox*), односно опција *Items* садржи колекцију одговора.



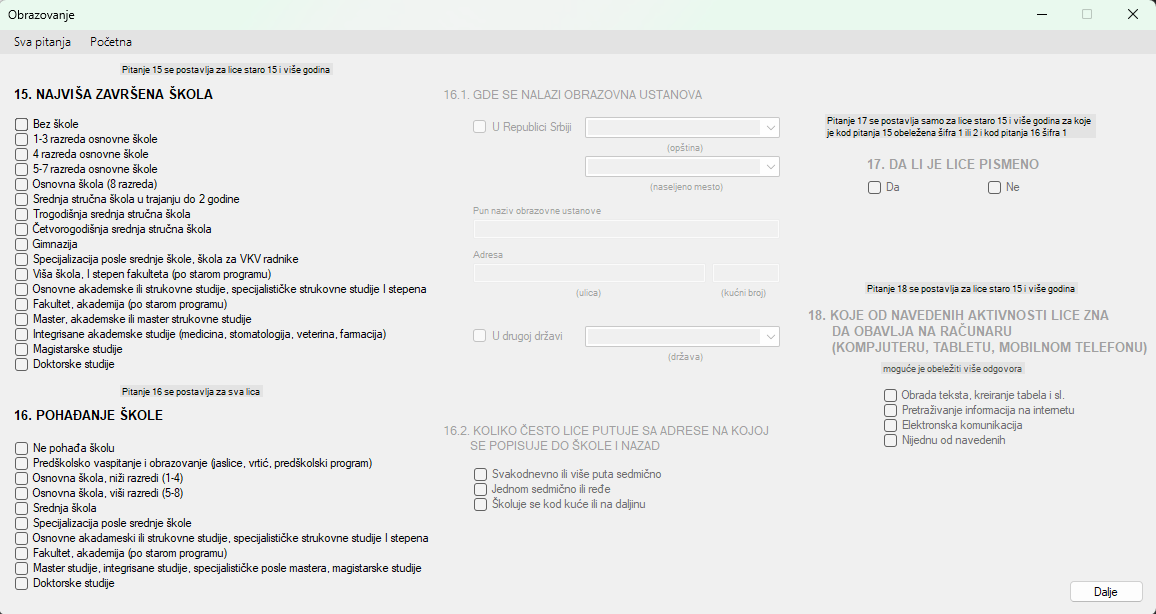
*Слика 4.6.4.2 – Питање „Матерњи језик“ у форми Етнокултурна обележја*

На слици 4.6.4.2 се виде сви понуђени одговори на питање „Матерњи језик“. Одговор који се обележи форма прослеђује у базу изразом:

red.pitanje\_13 = cmb13.Text;

### Образовање

Одељак за образовање се односи на највишу школу или школу коју тренутно похађа особа, али и на само образовање особе. Одељак садржи 4 питања и 2 потпитања. У зависности од одговора на одређена питања, нека питања се неће постављати.



*Слика 4.6.5.1 – Форма „Образовање“*

На слици 4.6.5.1 су сва питања са одговорима. Прва два питања су откључана због услова и у зависности од њих зависи да ли ће се постављати питања 16.1 и 16.2. На питање 18 је могуће да ти више одговора. Обзиром да је питање 18 комплексније, одговори на ово питање се додају у листу и притиском на дугме „Dalje“, одговори се уписују у *string* тако што се између сваког одговора дода зарез и размак, па се тај стринг уписује у базу.

if (clb18.Enabled)

{

foreach (string s in pitanje18)

if (pitanje18.Count != 1)

tmp += s + ", ";

else

tmp = s;

red.pitanje\_18 = tmp;

}

Због честе потребе за брисањем датих одговора на питања која су типа „*CheckedListBox*“, односно питања где се штиклирају одговори, уведене су функције „*Uncheck*“ и „*UncheckAll*“.

private void Uncheck(CheckedListBox clb, ItemCheckEventArgs e)

{

for (int i = 0; i < clb.Items.Count; ++i)

if (i != e.Index) clb.SetItemChecked(i, false);

}

private void UncheckAll(CheckedListBox clb)

{

clb.ClearSelected();

for (int i = 0; i < clb.Items.Count; ++i)

clb.SetItemChecked(i, false);

if (clb.Name!= "clb16")

{

clb.Enabled = false;

}

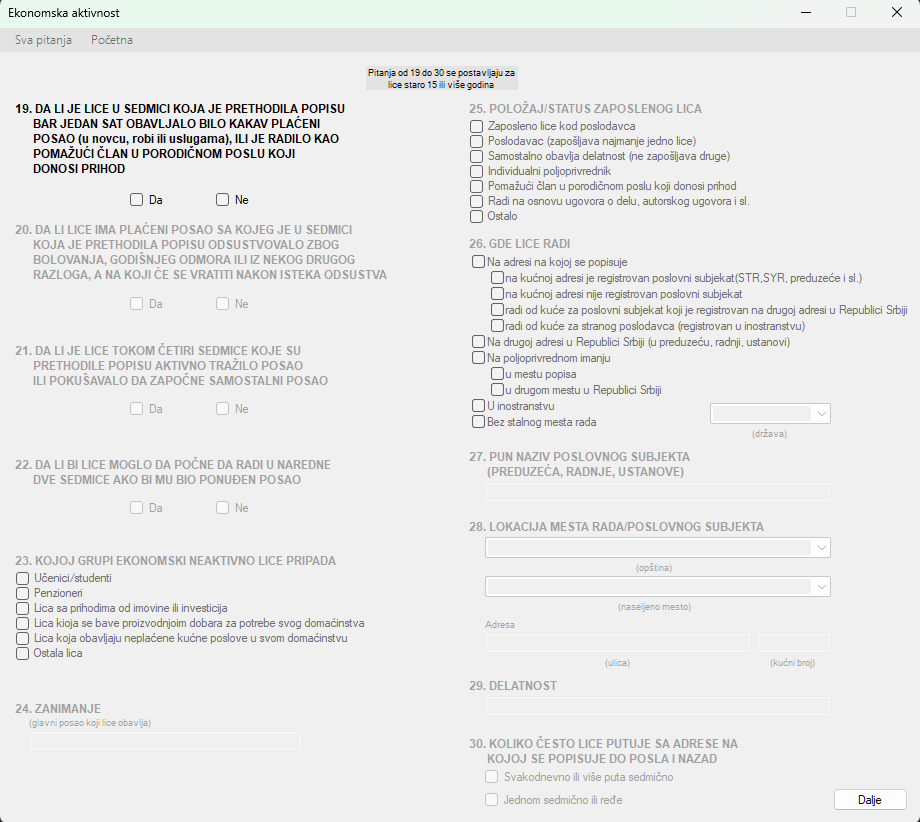
}

Функцији „*Uncheck*“ се прослеђује прослеђују „*CheckedListBox clb*“, односно питање на које се даје одговор, у овом случају су то питања где се одговори штиклирају, и прослеђују се подаци тог питања „*ItemCheckEventArgs e*“. Ова функција служи да би у исто време могао да постоји само један штиклиран одговор, а остали одговори се бришу

Функција „*UncheckAll*“ служи да би се сви одговори обрисали, на пример приликом мењања неког претходног одговора. Прослеђују јој се као и у претходној функцији „*CheckedListBox clb*“, односно питање на које се даје одговор. За сваки одговор који постоји у питању се поставља параметар *SetItemChecked* на вредност *false* и ако питање у форми није „clb16“, питање се гаси, односно на њега се више не може дати одговор док се не испуне неки други услови. Као и у свакој претходној форми, и овде је могуће погледати сва питања кликом на дугме „Sva pitanja“ или прекинути попис кликом на дугме „Početna“.

### Економска активност

Одељак економске активности се односи на послове које особа обавља, односно чиме се бави, да ли је студент или запослено лице и о дета љима самих послова. Састоји се из 11 питања која су међусобно зависна, али се



*Слика 4.6.6.1 – Форма „Економска активност“*

На слици 4.6.6.1 се налазе сва питања, прво питање је откључано и у зависности од њега се откључавају сва друга питања. На врху форме стоји обавештење да се питања постављају само за особе које имају 15 или више година. На основу тога, дугме „Dalje“ је откључано и може се одмах ићи на следећи одељак. Овде се први пут користи елемент „*TreeView*“ у питању 26 да би одговори били груписани. У случају да одговор има своју групу, одговор се у базу запише тако што се наведе име групе, поставе две тачке, па се запише одговор на питање.

if (tv26.Nodes[i].Nodes[j].Checked)

{

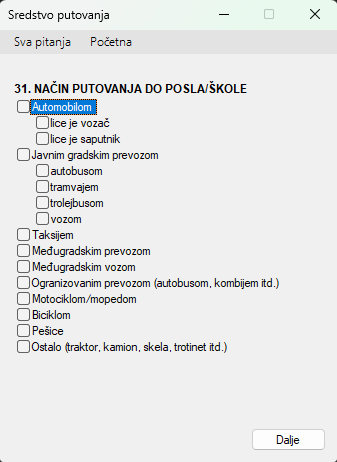
odgovor26 += ": " + tv26.Nodes[i].Nodes[j].Text;

}

Постоји могућност прегледа свих питања и престанак пописа избором из менија кликом на дугме „Sva pitanja“ или кликом на дугме „Početna“.

### Средство путовања

Средство путовања је једноставна форма, садржи једно питање које се односи на начин путовања до школе или до посла. Поред питања, постоје још опције за преглед свих питања притиском на дугме „Sva pitanja“ и опција за прекид пописа кликом на дугме „Početna“ (слика 4.6.7.1.).

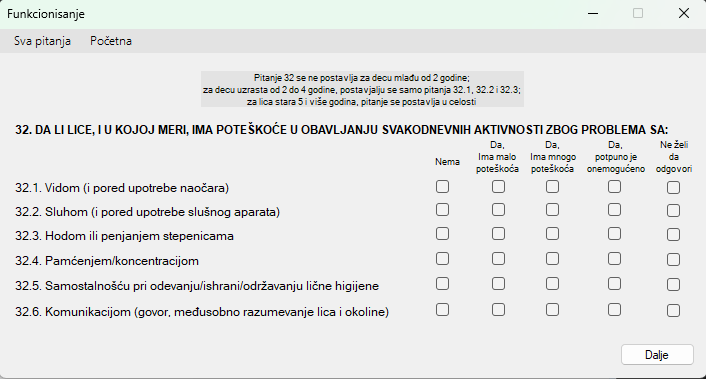


*Слика 4.6.7.1 – форма „Средство путовања“*

### Функционисање

Форма Функционисање се односи на функционалну способност особе која се пописује. Форма се састоји од 6 питања и свако питање има могућност озбора од пет одговора. За ову форму постоји неколико услова:

1. Питање 32 се не поставља за децу млађу од 2 године
2. За децу узраста од 2 до 4 године, постављају се само питања 32.1, 32.2 и 32.3
3. За лица стара 5 и више година, питање се поставља у целости

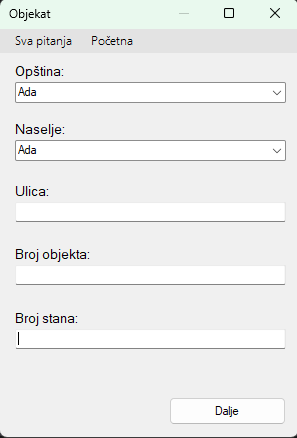


*Слика 4.6.8.1 – Форма „Функционисање“*

Ако је особа старија од 5 година, дужна је да одговори на свако питање, али постоји одговор да не жели да се изјасни. Обзиром на услове, дугме „Dalje“ је стално активно и може се притиснути кад год. Као и у свакој форми, притиском на дугме се појави упит да ли су то коначни одговори особе која се пописује, ако јесу, отвара се следећа форма. Постоји могућност прегледа свих питања притиском на дугме „Sva pitanja“ и могућност прекида пописа притиском на дугме „Početna“.

### Објекат

Форма објекат је, осим што је намењена да се евидентира место становања особе која се пописује, намењена је и да се провери да ли је објекат у ком станује особа већ пописан.



*Слика 4.6.9.1 – Форма „Објекат“*

Сва подаци осим „Broj stana“ су обавезни. Притиском на дугме „Dalje“, програм проверава да ли су обавезна поља испуњена. Ако су сва поља испуњена, појављује се прозор са упитом да ли су ти одговори коначни, ако јесу, програм шаље упит бази, односно табелу „Оbjekat” пита да ли постоји већ објекат који садржи податке из питања. Ако постоји, попис се завршава и програм отвара почетну форму, а ако не постоји, отвара се форма Подаци о згради.

var postoji =

(

from o in bd.OBJEKAT

where o.opstina == cmbOpstina.Text

&& o.naselje == cmbNaselje.Text

&& o.ulica == txtUlica.Text

&& o.broj\_objekta == txtBrojObjekta.Text

&& o.broj\_stana == txtBrojStana.Text

select o

).Count();

У случај да објекат постоји у табели, позива се функција „upisTabela“ којој се ништа не прослеђује, а садржи процедуре које су интегрисане у базу.

private void upisTabela()

{

upis.upis\_reda\_osoba(id\_popisivaca);

upis.upis\_reda\_objekat(id\_popisivaca);

upis.upis\_reda\_bracni\_status(id\_popisivaca);

upis.upis\_reda\_eko\_akt(id\_popisivaca);

upis.upis\_reda\_etno(id\_popisivaca);

upis.upis\_reda\_migracije(id\_popisivaca);

upis.upis\_reda\_obrazovanje(id\_popisivaca);

upis.upis\_reda\_prevoz(id\_popisivaca);

upis.upis\_reda\_soc\_int(id\_popisivaca);

}

Свакој процедури се прослеђује id\_popisivaca и свака процедура је везана за одређену табелу. Процедура upis\_reda\_osoba је везана за табелу OSOBA, док је процедура upis\_reda\_migracije везана за табелу MIGRACIJE итд.

create procedure [dbo].[upis\_reda\_osoba](@id\_popisivaca int)

as begin

insert into OSOBA

select \* from tmp\_OSOBA

where id\_popisivac=@id\_popisivaca

end

create procedure [dbo].[upis\_reda\_migracije](@id\_popisivaca int)

as begin

insert into MIGRACIJE

select \* from tmp\_MIGRACIJE

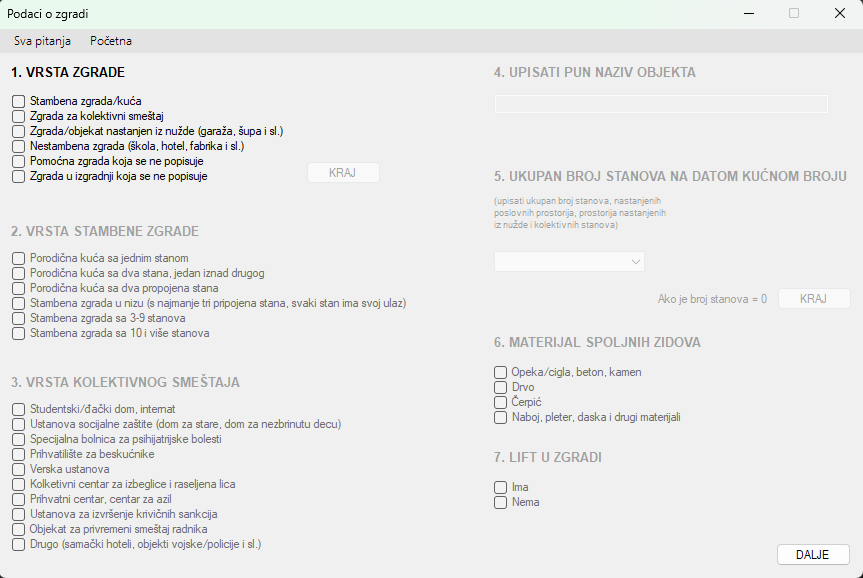
where id\_popisivac=@id\_popisivaca

end

Притиском на дугме „Sva pitanja“, отвара се фајл са списком свих питања за попис, а притиском на дугме „Početna“, појави се искачући прозор са питањем да ли попис жели да се прекине. Ако је одговор потврдан, отвара се почетна страна.

### Подаци о згради

Форма Подаци о згради садржи 7 питања и у зависности од одговора на постављена питања, попис може одмах да се заврши или да се пређе на следећа питања. Ако се на питање број 1 одговори петим или шестим понуђеним одговором, откључава се дугме крај или ако је одговор „нула“ на пето постављено питање, попис се завршава, подаци се уписују у базу, односно табелу, и програм отвара почетну страну. Дугме „Dalje“ је закључано све док се не стигне до последњег питања.



*Слика 4.6.10.1 – Форма „Подаци о згради“*

У случају да се попис заврши, опет се позива функција „upisTabela“, али је у односу на претходну форму додата процедура „upis\_reda\_zgrada“. А ако попис није завршен, притиском на дугме даље, појављује се искачући прозор који пита да ли је особа задовољна одговорима. Ако је задовољна, отвара се форма Подаци о стану. Обзиром да је за особу завршен попис и да се пописује њено место становања, сада се за сваку форму прослеђују id\_objekta и id\_popisivaca.

private void btnDalje\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Da li ste sigurni?", "Unos", MessageBoxButtons.YesNo);

if (result == DialogResult.Yes)

{

BazaDiplomskog.tmp\_O\_ZGRADIRow red = bd.tmp\_O\_ZGRADI.Newtmp\_O\_ZGRADIRow();

red.id\_objekta = id\_objekta;

red.id\_popisivac = id\_popisivaca;

red.pitanje\_1 = clb1.SelectedItem.ToString();

if(clb2.SelectedItem!=null)

red.pitanje\_2 = clb2.SelectedItem.ToString();

if (clb3.SelectedItem != null)

red.pitanje\_3 = clb3.SelectedItem.ToString();

if (txt4.Text != string.Empty)

red.pitanje\_4 = txt4.Text;

if (cmb5.SelectedItem != null)

red.pitanje\_5 = cmb5.SelectedItem.ToString();

if (clb6.SelectedItem != null)

red.pitanje\_6 = clb6.SelectedItem.ToString();

if (clb7.SelectedItem != null)

red.pitanje\_7 = clb7.SelectedItem.ToString();

bd.tmp\_O\_ZGRADI.Addtmp\_O\_ZGRADIRow(red);

tmp\_o\_zgradiAdapter.Update(bd.tmp\_O\_ZGRADI);

MessageBoxManager.Unregister();

frmOStanu forma = new frmOStanu(id\_objekta, id\_popisivaca);

this.Hide();

forma.ShowDialog();

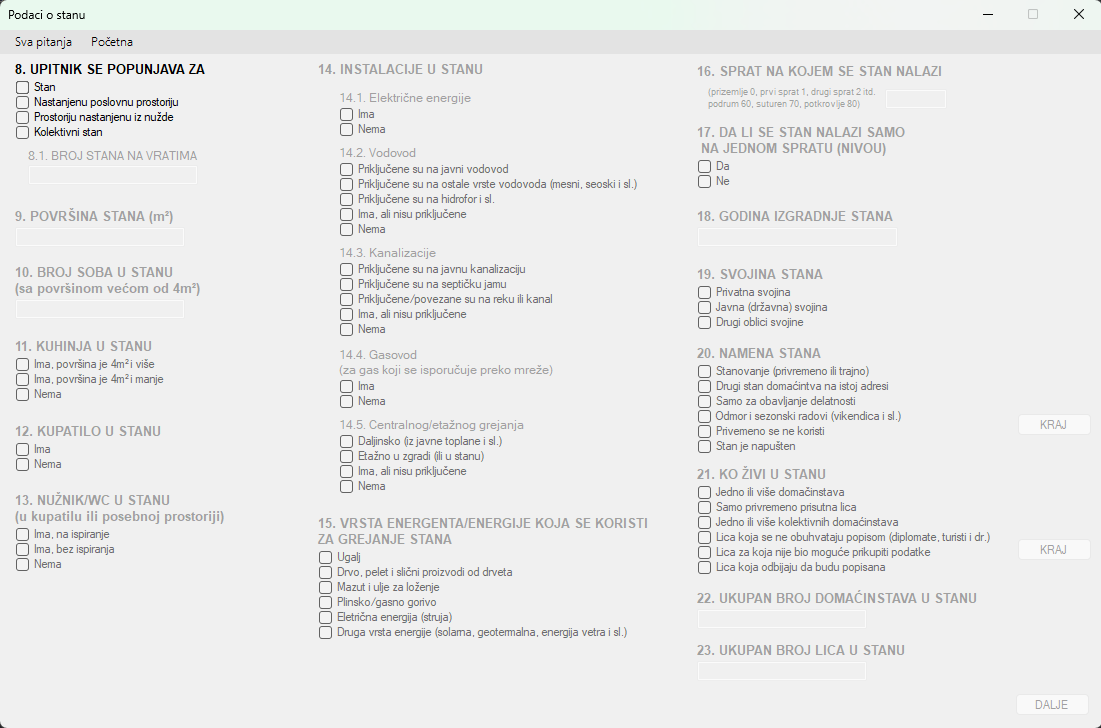
this.Close();

}

}

### Подаци о стану

Форма Подаци о стану је највећа форма у програму и садржи детаље самог стана, као што су квадратура, грејање и број просторија. Такође садржи питања и особама које станују где и особа која се пописује. У зависности од одговора на питања 20 и 21, попис може да се заврши или пређе на следеће питање или форму. У почетку је активно само прво питање и одговор на њега отвара друга питања.

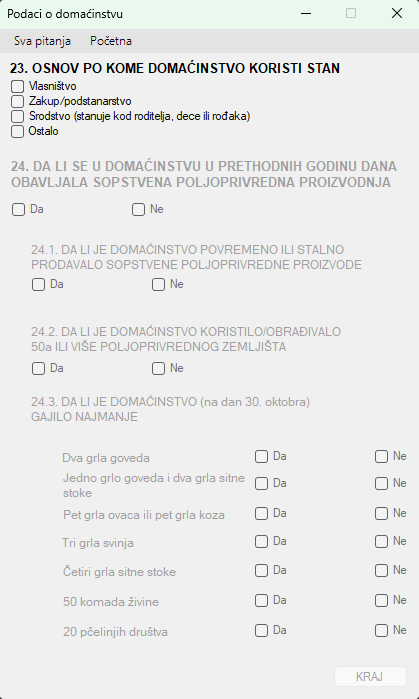


*Слика 4.6.11.1 – Форма „Подаци о стану“*

Дугме „Dalje“ је закључано и одговором на одређена питања се откључава. Када се притисне дугме, појави се искачући прозор. Кликом на дугме „Da“ на прозору, отвара се форма „Podaci o domaćinstvu“. А ако се попис заврши раније, односно притиском на дугме „Kraj“ које се откључава одређеним одговорима на питање 20 и 21, позива се функција „upisTabela“. Као и у претходном случају, у односу на претходну форму, у функцију је додата процедура „upis\_reda\_stan“ која уписује све претходне и ову форму у базу. Кликом на дугме „Sva“ pitanja могу се видети сва питања, односно дугме „Početna“ води на почетну страну.

### Подаци о домаћинству

Подаци о домаћинству је последња форма у попису. Односи се на намену стана и да ли се породица бави пољопривредом. Садржи два питања и 3 потпитања и у зависности од одговора на друго питање, зависи да ли ће се постављати и наредна питања или се завршава попис.



*Слика 4.6.12.1 – Форма „Подаци о домаћинству“*

Одређеним одговором се откључава дугме „Kraj“, а затим кликом на њега се појављује искачући прозор. Потврдим одговором на питање из прозора, позива се функција „upisTabela“ и сви подаци, односно сва питања се уписују у базу и отвара се почетна страна програма.

private void upisTabela()

{

upis.upis\_reda\_osoba(id\_popisivaca);

upis.upis\_reda\_objekat(id\_popisivaca);

upis.upis\_reda\_bracni\_status(id\_popisivaca);

upis.upis\_reda\_eko\_akt(id\_popisivaca);

upis.upis\_reda\_etno(id\_popisivaca);

upis.upis\_reda\_migracije(id\_popisivaca);

upis.upis\_reda\_obrazovanje(id\_popisivaca);

upis.upis\_reda\_prevoz(id\_popisivaca);

upis.upis\_reda\_soc\_int(id\_popisivaca);

upis.upis\_reda\_zgrada(id\_popisivaca);

upis.upis\_reda\_domacinstvo(id\_popisivaca);

upis.upis\_reda\_stan(id\_popisivaca);

}

# Закључак

Програм за попис становништва је направљен да би се убрзао и што више упростио попис становништва чији цео процес, од самог пописа, до обраде података траје изузетно дуго. Програм је прављен у *Visual Studio 17* програму помоћу *C#* програмског језика упареног са *SQL* програмским језиком са којим је прављена база у ***SQL Server Management Studio Management Studio* 19** програму. На овај избор сам се одлучио јер сам имао сличне пројекте и сматрао сам да овом комбинацом може да се направи квалитетан програм. Овај пројекат је до сада највећи изазов за мене, уложено је неколико месеци рада и константним радом и тестирањем су откриване грешке и бољи начини за писање самог програма, односно лакши и бржи начин да програм ради. Велики труд је уложен, али сматрам да овај програм још може да се усавршава тако да сам рад буде још лакши и бржи.

# Индекс појмова

# Литература

[1] Kevin Driedger, *Encode and decode a base64 string* <https://stackoverflow.com/questions/11743160/how-do-i-encode-and-decode-a-base64-string>

[2] Alex C. Duma, *Localizing System MessageBox,* <https://www.codeproject.com/Articles/18399/Localizing-System-MessageBox>  
преузето: септембар 2023.

[3] Републички завод за статистику, *СТАТ база, одељак „Попис становништва, домаћинстава и станова“,* [*https://data.stat.gov.rs/*](https://data.stat.gov.rs/)

[4] Републички завод за статистику, *„Методолошки материјали“*, [*https://popis2022.stat.gov.rs/sr-cyrl/4-metodoloskimaterijali*](https://popis2022.stat.gov.rs/sr-cyrl/4-metodoloskimaterijali)

# Изјава о академској честитости

**ИЗЈАВА О АКАДЕМСКОЈ ЧЕСТИТОСТИ**

|  |  |
| --- | --- |
| **Студент (име, име једног родитеља и презиме):** |  |
| **Број индекса:** |  |

Под пуном моралном, материјалном, дисциплинском и кривичном одговорношћу изјављујем да је завршни рад, под насловом:

1. резултат сопственог истраживачког рада;
2. да овaj рад, ни у целини, нити у деловима, нисам пријављиво/ла на другим високошколским установама;
3. да нисам повредио/ла ауторска права, нити злоупотребио/ла интелектуалну својину других лица;
4. да сам рад и мишљења других аутора које сам користио/ла у овом раду назначио/ла или цитирао/ла у складу са Упутством;
5. да су сви радови и мишљења других аутора наведени у списку литературе/референци који је саставни део овог рада, пописани у складу са Упутством;
6. да сам свестан/свесна да је плагијат коришћење туђих радова у било ком облику (као цитата, прафраза, слика, табела, дијаграма, дизајна, планова, фотографија, филма, музике, формула, вебсајтова, компјутерских програма и сл.) без навођења аутора или представљање туђих ауторских дела као мојих, кажњиво по закону (Закон о ауторском и сродним правима), као и других закона и одговарајућих аката Високе школе електротехнике и рачунарства струковних студија у Београду;
7. да је електронска верзија овог рада идентична штампаном примерку овог рада и да пристајем на његово објављивање под условима прописаним актима Високе школе електротехнике и рачунарства струковних студија у Београду;
8. да сам свестан/свесна последица уколико се докаже да је овај рад плагијат.

У Београду, \_\_. \_\_. 201\_. године

Својеручни потпис студента

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_