



УНИВЕРЗИТЕТ У НОВОМ САДУ  
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У  
НОВОМ САДУ

---



Божидар Марић

# **Софтверски пакет за подршку рада запослених у школским установама**

ДИПЛОМСКИ РАД

- Основне академске студије -

Нови Сад, 2020





УНИВЕРЗИТЕТ У НОВОМ САДУ • ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА  
21000 НОВИ САД, Трг Доситеја Обрадовића 6

## КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, <b>РБР</b> :	
Идентификациони број, <b>ИБР</b> :	
Тип документације, <b>ТД</b> :	Монографска документација
Тип записа, <b>ТЗ</b> :	Текстуални штампани материјал
Врста рада, <b>ВР</b> :	Дипломски (Bachelor) рад
Аутор, <b>АУ</b> :	Божидар Марић
Ментор, <b>МН</b> :	Др Милан Челиковић, доцент
Наслов рада, <b>НР</b> :	Софтверски пакет за подршку рада запослених у образовним институцијама
Језик публикације, <b>ЈП</b> :	Српски / ћирилица
Језик извода, <b>ЈИ</b> :	Српски
Земља публиковања, <b>ЗП</b> :	Република Србија
Уже географско подручје, <b>УГП</b> :	Војводина
Година, <b>ГО</b> :	2020
Издавач, <b>ИЗ</b> :	Ауторски репринт
Место и адреса, <b>МА</b> :	Нови Сад; трг Доситеја Обрадовића 6
Физички опис рада, <b>ФО</b> : (поглавља/страна/ цитата/табела/слика/графика/прилога)	5/69/0/90/46/0/0
Научна област, <b>НО</b> :	Електротехника и рачунарство
Научна дисциплина, <b>НД</b> :	Примењене софтверско инжењерство
Предметна одредница/Кључне речи, <b>ПО</b> :	Базе података 2
<b>УДК</b>	
Чува се, <b>ЧУ</b> :	У библиотеци Факултета техничких наука, Нови Сад
Важна напомена, <b>ВН</b> :	
Извод, <b>ИЗ</b> :	У овом раду презентован је пројекат за реализацију дела информационог система једне школе. Циљ изабраног дела информационог система је да поједностави процес евиденције рада запосленима у школи. Описани информациони систем омогућава учитељима и наставницима да организују предавања, часове, контролне, домаће, евидентирају присуство ученика и оцене радове истих. На основу података унетих у базу система, даље је могуће прегледати испланиране часове за одређени датум, као и послати информације везане за успех ученика њиховим родитељима путем мејла. Модел шеме базе података пројектована је помоћу пакета Entity Framework. Пројекат је затим реализован на платформи Visual Studio 2019.
Датум прихватања теме, <b>ДП</b> :	
Датум одбране, <b>ДО</b> :	09.10.2020
Чланови комисије, <b>КО</b> :	Председник: Др Иван Луковић, редовни професор
	Члан: Др Владимир Иванчевић, доцент
	Члан, ментор: Др Милан Челиковић, доцент
	Потпис ментора






## KEY WORDS DOCUMENTATION

Accession number, <b>ANO</b> :			
Identification number, <b>INO</b> :			
Document type, <b>DT</b> :	Monographic publication		
Type of record, <b>TR</b> :	Textual printed material		
Contents code, <b>CC</b> :	Bachelor Thesis		
Author, <b>AU</b> :	Božidar Marić		
Mentor, <b>MN</b> :	Millan Čeliković, Associate Professor, Ph.D		
Title, <b>TI</b> :	A software tool for school System Support Service		
Language of text, <b>LT</b> :	Serbian		
Language of abstract, <b>LA</b> :	Serbian		
Country of publication, <b>CP</b> :	Republic of Serbia		
Locality of publication, <b>LP</b> :	Vojvodina		
Publication year, <b>PY</b> :	2020		
Publisher, <b>PB</b> :	Author's reprint		
Publication place, <b>PP</b> :	Novi Sad, Dositeja Obradovica sq. 6		
Physical description, <b>PD</b> : (chapters/pages/ref./tables/pictures/graphs/appendixes)	5/69/0/90/46/0/0		
Scientific field, <b>SF</b> :	Electrical and computer engineering		
Scientific discipline, <b>SD</b> :	Applied software engineering		
Subject/Key words, <b>S/KW</b> :	Databases 2		
<b>UC</b>			
Holding data, <b>HD</b> :	The Library of Faculty of Technical Sciences, Novi Sad, Serbia		
Note, <b>N</b> :			
Abstract, <b>AB</b> :	<p>This thesis presents the specification and implementation of a section of an information system for a school. The aim of the selected section of the information system is to simplify the process of recording employees work. The described information system allows for teachers to organise lectures, lessons, tests, homeworks, record student attendance and student grades. Based on the information present in system's database it is further possible to review lessons planned for a certain date and to send informations pertaining to student's achievements to their parents via email. The database schematic was created using the Entity Framework package. The project was afterwards implemented using the Visual Studio 2019 platform.</p>		
Accepted by the Scientific Board on, <b>ASB</b> :			
Defended on, <b>DE</b> :	09.10.2020.		
Defended Board, <b>DB</b> :	President:	Ivan Luković, Full Professor, Ph. D	
	Member:	Vladimir Ivančević, Associate Professor, Ph. D	Menthor's sign
	Member, Mentor:	Milan Čeliković, Associate Professor, Ph. D	



	УНИВЕРЗИТЕТ У НОВОМ САДУ • ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА 21000 НОВИ САД, Трг Доситеја Обрадовића 6	Датум: 09.10.2020.
	<b>ЗАДАТАК ЗА ДИПЛОМСКИ РАД</b>	Лист/Листова:

(Податке уноси предметни наставник - ментор)

Студијски програм:	<b>Примењено софтверско инжењерство</b>
Руководилац стидијског програма:	<b>Др Драган Поповић</b>

Студент:	<b>Божидар Марић</b>	Број индекса:	<b>ПР 32/2016</b>
Област:	<b>Базе података 2</b>		
Ментор:	<b>др Милан Челиковић</b>		

НА ОСНОВУ ПОДНЕТЕ ПРИЈАВЕ, ПРИЛОЖЕНЕ ДОКУМЕНТАЦИЈЕ И ОДРЕДБИ СТАТУТА ФАКУЛТЕТА ИЗДАЈЕ СЕ ЗАДАТАК ЗА ДИПЛОМСКИ РАД, СА СЛЕДЕЋИМ ЕЛЕМЕНТИМА:

- проблем – тема рада;
- начин решавања проблема и начин практичне провере резултата рада, ако је таква провера неопходна;

### НАСЛОВ ДИПЛОМСКОГ РАДА:

**Софтверски пакет за подршку рада запослених у школским установама**

### ТЕКСТ ЗАДАТКА:

- Проучити аспекте практичне примене изабраних алата за пројектовање и имплементацију шеме базе података и алата за пројектовање и имплементацију апликација информационих система.
- Испројектовати сегмент концептуалне и имплементационе шеме базе података, потребан за развој дела информационог система прихватилишта за животиње, а намењен за проналажење животиње са карактеристикама дефинисаним од стране потенцијалног усвојитеља.
- Имплементирати испројектовани сегмент шеме базе података на изабраном систему за управљање базама података.
- Имплементирати апликативно софтверско решење за део система који покрива функције за подршку процеса анализе упитника и проналажење животиње са карактеристикама дефинисаним од стране потенцијалног усвојитеља.

Руководилац студијског програма:	Ментор рада:

Примерак за: ☐ - Студента; ☐ - Ментора





## Садржај

1. Увод.....	1
2. Опис реалног система .....	3
2.1 Типови корисника и њихове карактеристике.....	3
2.2 Функционални захтеви .....	3
2.2.1 Основне корисничке функционалности.....	4
2.2.2 Администраторске функционалности .....	4
2.2.3 Функционалности запослених .....	5
2.3 Перспектива система.....	5
3. Опис шеме базе података.....	7
3.1 Концептуална шема базе података.....	7
3.1.1 Појмови коришћени при моделовању концептуалних шема база података .....	7
3.2 Модел шеме базе података.....	10
3.2.1 Део модела шеме базе података за смештање података о корисницима .....	10
3.2.2 Део модела шеме базе података за смештање података о запосленима.....	10
3.2.3 Део модела шеме базе података за смештање података о ученицима .....	11
3.2.4 Део модела шеме базе података за смештање података о предметима.....	12
3.2.5 Део модела шеме базе података за смештање информација о предавањима и часовима .....	12
3.2.6 Део модела шеме базе података за смештање података о контролним тачкама.....	14
3.3 База података генерисана из модела шема базе података.....	15
3.3.1 Изглед база података генерисаних за потребе информационог система .....	16
3.4 Опис ентитета и асоцијација у моделима .....	18
3.5 Улога базе података.....	24
4. Опис апликативног решења .....	25
4.1 Софтверска архитектура .....	25
4.2 Коришћени пакети.....	25
4.3 Администраторске функционалности.....	26

4.3.1 Креирање корисничких информација.....	27
4.4 Функционалности запослених .....	29
4.4.1 Креирање новог часа .....	29
4.4.1 Слање информација о ученику .....	31
5. Закључак.....	33
Речник појмова .....	35
Скраћенице .....	37
Литература .....	39
Биографија .....	41

## 1. Увод

Образовне установе данас и даље користе принципе и начин евидентирања обављеног посла који је коришћен деценијама пре. Напредак технологија отвара врата за побољшање у свим сферама живота, па и у начину обављања посла запосленим у школским установама. Тренутне методе евидентирања обављеног посла у једној школској установи захтевају документовање на различитим папирним формама и у различитим дневницима. Увођењем информационог система у школске установе који би заменио преопширну папирологију представљао би значајно унапређење и олакшицу запосленима у школским установама.

У овом дипломском раду описан је информациони систем једне школске установе. Примарни циљ описаног информационог система је да пружи подршку евиденције рада запосленима у школи. Софтверска подршка за наведени процес омогућава запосленима да информације неопходне за рад у образовној установи имају увек на располагању на једном месту. Без софтверске подршке запослени би били принуђени да са собом носе неопходне информације на различитим формуларима у папирној форми што непотребно компликује рад запослених. Узимајући у обзир количину документације укључене у процес, постоји вероватноћа да ће доћи до губљења или занемаривања дела података, или грешке приликом обраде.

Такође, родитељи ученика, често немају доступне све потребне информације о успеху свог детета. Због тога, они су принуђени да долазе лично у школу да траже те информације, што може представљати проблем родитељима који су на послу у време рада школе.

Информациони систем доприноси прегледности и доступности података и убрзава и олакшава рад запослених при планирању часова, контролних тачака, евиденцији присуства ученика на часу и сл., при чему се истовремено повећава степен информисаности родитеља о академским успесима своје деце. Смањење папирологије, запосленима омогућава да се више посвете раду са својим ученицима, што је примарни део њиховог посла.

Поред Увода, Закључка, Речника појмова, Литературе и Биографије, овај дипломски рад садржи три поглавља:

- **Поглавље 2. Опис реалног система** – у овом поглављу описани су типови корисника, функционални захтеви и перспектива система,
- **Поглавље 3. Опис шема базе података** – у овом поглављу представљени је модел базе података и дат опис свих ентитета и релација присутних у систему и
- **Поглавље 4. Опис апликативног решења** – у овом поглављу приказано је програмско решење за функционалне захтеве описане у поглављу 2.



## 2. Опис реалног система

У овом поглављу описани су типови корисника, њихове карактеристике и надлежности у информационом систему основне школе. Потом су наведени функционални захтеви које систем треба да реализује, као и перспективе таквог система.

### 2.1 Типови корисника и њихове карактеристике

У информационом систему основне школе присутни су следећи типови корисника:

1. Учитељ:
  - задужен за једно одељење,
  - организује предавања, часове, домаће и контролне задатке за своје одељење,
  - евидентира присуство ученика на часовима,
  - оцењује радове ученика и
  - шаље извештај о тренутним успесима својих ученика њиховим родитељима
2. Наставник:
  - предаје један предмет,
  - организује предавања, часове, домаће и контролне задатке из свог предмета, а у оквиру одељења којима предаје,
  - оцењује радове ученика и
  - шаље извештај о тренутним успесима ученика у одељењима којима предаје на његовом предмету.
3. Администратор система:
  - уноси информације о тренутним запосленима у школској установи,
  - ажурира информације о одељењима школе,
  - ажурира информације о ученицима који похађају школу,
  - ажурира информације о учионицама које у школи постоје и
  - ажурира информације о предметима и њиховим областима који се у школи предају.

Карактеристике типова корисника информационог система су:

1. *Учитељи* – предају више предмета једном одељењу. У систему је неопходно да постоји одељење којем је учитељ додељен, као и предмети за разред који одговара разреду одељења којем учитељ предаје. У супротном учитељ не може ефикасно да користи информациони систем.
2. *Наставници* – предају један предмет једном или више одељења. У систему је неопходно да постоји предмет који наставник држи и барем једно одељење којем наставник предаје. Иначе наставник не може да користи систем на планирани начин.
3. *Администратор система* – задужен за иницијализацију базе података неопходним информацијама које би запосленима омогућиле што ефикасније коришћење информационог система. Неопходне информације обухватају креирање корисничких налога за запослене, евидентирање учионица, ученика који похађају школу, одељења која постоје у школи, предмета који се предају и сл.

### 2.2 Функционални захтеви

У оквиру поглавља наведени су и описани функционални захтеви које информациони систем треба да испуни. Ради лакшег разумевања, захтеви су груписани према типовима корисника за које су везани и то у три групе: основне корисничке, администраторске и функционалности запослених.

### 2.2.1 Основне корисничке функционалности

Основне корисничке функционалности обухватају групу захтева везане за управљање корисничким налозима у оквиру информационог система једне школске установе. У наведене функционалности спадају:

1. **Регистрација запослених** – обавља се од стране администратора система. Приликом регистрације новог запосленог администратор уноси основне информације о запосленом и додељује му улогу у оквиру система у складу са којом запослени може да позива друге функционалности система.
2. **Пријава на систем** – сви претходно регистровани корисници система су у могућности да позову ову функционалност. Након успешне пријаве корисник добија приступ систему у складу са својом улогом.
3. **Ажурирање информација о личности** – запослени који имају направљен кориснички налог су у могућности да позову ову функционалност. Изузев администратора, податке о личности могу видети само они корисници на које се ти подаци односе.
4. **Ажурирање лозинке** – сви запослени могу да промене своју лозинку, при чему се од њих тражи да је промене при првој пријави на систем.

### 2.2.2 Администраторске функционалности

Функционалности описане у оквиру овог поглавља намењене су да подрже рад запослених у школској установи и представљају основу за даљу употребу информационог система од стране запосленог. Њих може позвати искључиво администратор система и у њих спадају:

1. **Ажурирање корисничких налога** – администратор може направити нови кориснички налог и обрисати постојећи. По креирању корисничког налога, одговорност одржавања информација о личности у ажурном стању прелази на запосленог за ког је тај кориснички налог креиран, те је онемогућено да администратор мења једном унете податке. У случају прекида радног односа запосленог, он се мора уклонити из система од стране одговорног лица, те је ова функционалност везана за самог администратора и само је он може позвати.
2. **Ажурирање учионица** – запосленима је потребно обезбедити валидне информације о постојећим учионицама у школској установи ради планирања часова. Како не би требало да се те информације мењају често и од стране било ког корисника система ова функционалност је везана само за администратора информационог система.
3. **Ажурирање информација о предметима** – запосленима је такође неопходно обезбедити информације о предметима који се предају у школској установи. Креирање, мењање и брисање ових информација одговорност је искључиво администратора система.
4. **Ажурирање информација о ученицима** – у систему је неопходно евидентирати све ученике који похађају школску установу како би запослени имали информације о ученицима којима предају. Унос, модификација и брисање ових информација мора обављати одговорно лице, те је ово још једна од функционалности администратора система.
5. **Ажурирање информација о одељењима** – како се часови одржавају у групама ученика и обзиром да запосленима морамо омогућити информације о групама ученика којима држе часове, одговорност је администратора система да те информације унесе у систем. Унос подразумева креирање, модификацију и брисање одељења, али и додавање постојећих ученика у одељења, као и додељивање одељења запосленима, ако тим одељењима предају.

### 2.2.3 Функционалности запослених

У овом поглављу описане су функционалности које запослени који имају кориснички налог могу да позову уз освит на разлике, ако их има, између позива од стране учитеља и од стране наставника. Обухвата следеће функционалности:

1. **Креирање часова** – запосленима је омогућено евидентирање планираних часова, при чему запослени бира предмет и његову област за коју планира час, као и одељење и учионицу у којој ће час бити одржан. Дозвољено је брисање и модификација часова.
2. **Евидентирање присуства ученика** – запосленима је омогућено да евидентирају присутне ученике на часовима које су унели у систем.
3. **Креирање домаћих задатака и контролних** – у циљу евидентирања рада ученика на неком предмету, запосленима је дозвољено да креирају, мењају и бришу домаће и контролне задатке из предмета које предају, при чему су ти задаци повезани са је одељењем којем су намењени.
4. **Креирање предавања** – ради лакшег планирања области које ће запослени да прелази у неком дану, омогућено је креирање, мењање и брисање предавања. Предавања, за разлику од часова, нису везана за учионицу нити одељење јер она представљају подсетник запосленом да за одређени дан жели да испредаје одређену област.
5. **Оцењивање ученичких радова** – запосленом је такође омогућено да оцењује домаће и контролне задатке које је је претходно задао одређеном одељењу.
6. **Преглед распореда часова** – ради додатног олакшања при планирању часова, запосленима је омогућен преглед постојећих часова за неки задати датум.
7. **Информисање родитеља о успеху ученика** – запосленима је такође омогућено да пошаљу тренутне информације о постигнутом успеху ученика на мејл адресу родитеља тог ученика.

### 2.3 Перспектива система

Примарна намена информационог система, описаног у овом дипломском раду, је да олакша рад запосленима у образовним институцијама. Увођењем информационог система у образовне установе, запосленима је омогућено да потребне информације обрађују електронским путем на једноставнији начин, тиме смањујући време које би се у супротном трошило на документацију где би се те информације иначе чувале. Такође, аутоматским слањем извештаја о тренутном успеху ученика његовим родитељима избегава се потреба за родитељским састанцима или позивима. Заобилажењем информисања сваког родитеља појединачно о академским постигнућима његовог детета, оставља се више простора запосленима у школама да се посвете свом истинском задатку, што је едукација ученика. У светлу тренутне глобалне ситуације, требало би напоменути да је ова функционалност система од посебног значаја, јер се помоћу ње смањује потреба за личним сусретом родитеља и запосленог.

У наставку рада описан је информациони систем развијен према потребама дефинисаним у реалним системом. Апликативно решење је урађено као десктоп апликација са базом података у коју се смештају информације. У наредном поглављу детаљно је објашњена база података коришћена у оквиру информационог система.





### 3. Опис шеме базе података

У овом поглављу биће описана концептуална шема базе података коришћена за развој информационог система, а затим и модели шема базе података коришћене за генерисање базе. Такође, биће анализирана и сама база података генерисана на основу модела шема база података употребом функционалности из *Entity Framework* пакета. На крају, дати су описи ентитета и асоцијација у оквиру модела креираног помоћу *Entity Framework Designer* алата.

#### 3.1 Концептуална шема базе података

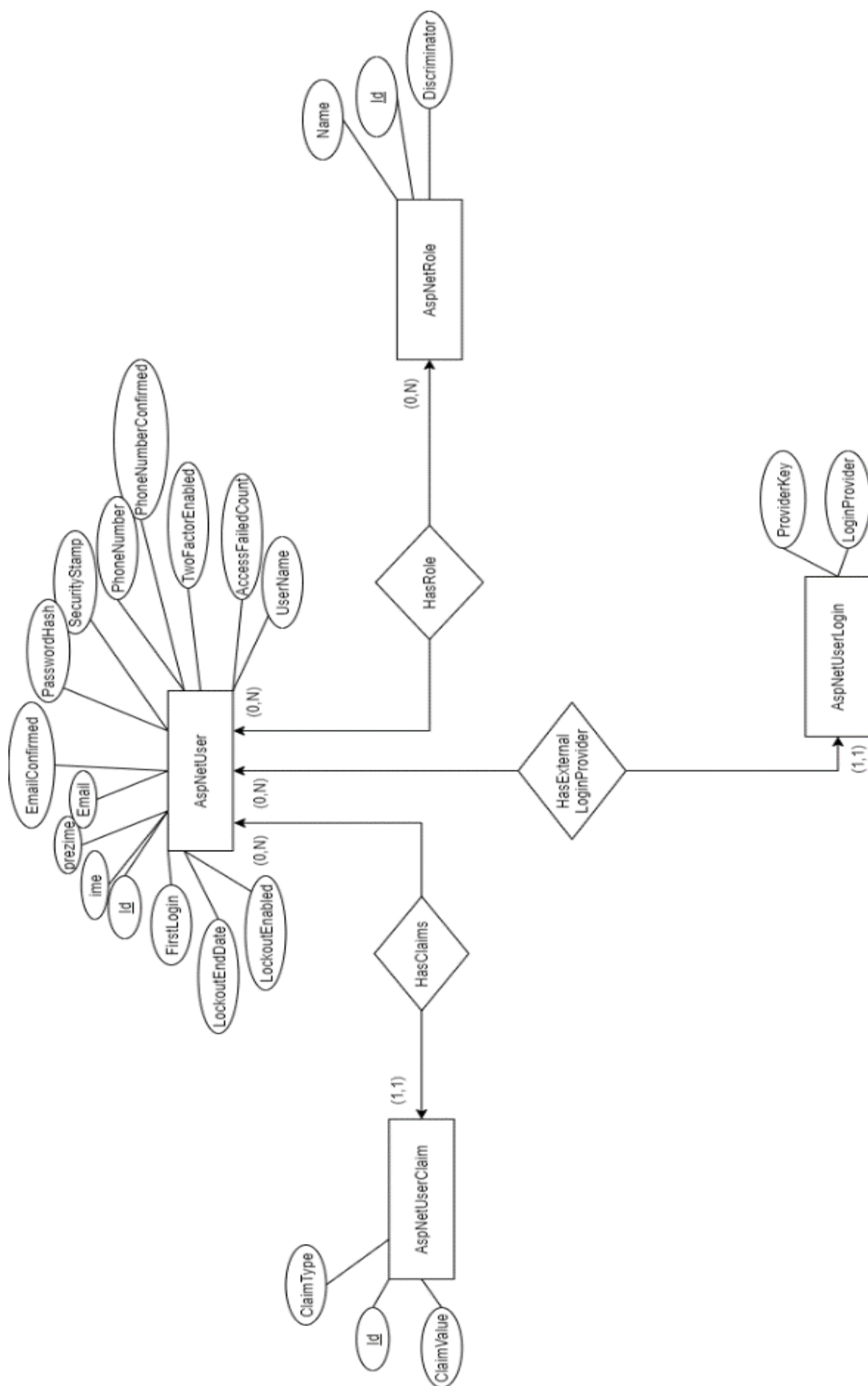
Концептуални модели шема база података засновани су на анализи информација које су неопходне за што једноставнији рад запослених у образовним институцијама. Модели су презентовани у форми ЕР дијаграма и налази се на сликама испод (слика 3.1 и слика 3.2). Како су све информације са тог модела пренете у модел шеме базе података направљен помоћу *Entity Framework* [1] пакета, детаљи типова ентитета и типова повезника биће објашњени у следећем поглављу, док ће се овде размотрити појмови коришћени у моделу и начин на који су ти појмови преведени тако да их *Entity Framework* пакет разуме.

##### 3.1.1 Појмови коришћени при моделовању концептуалних шема база података

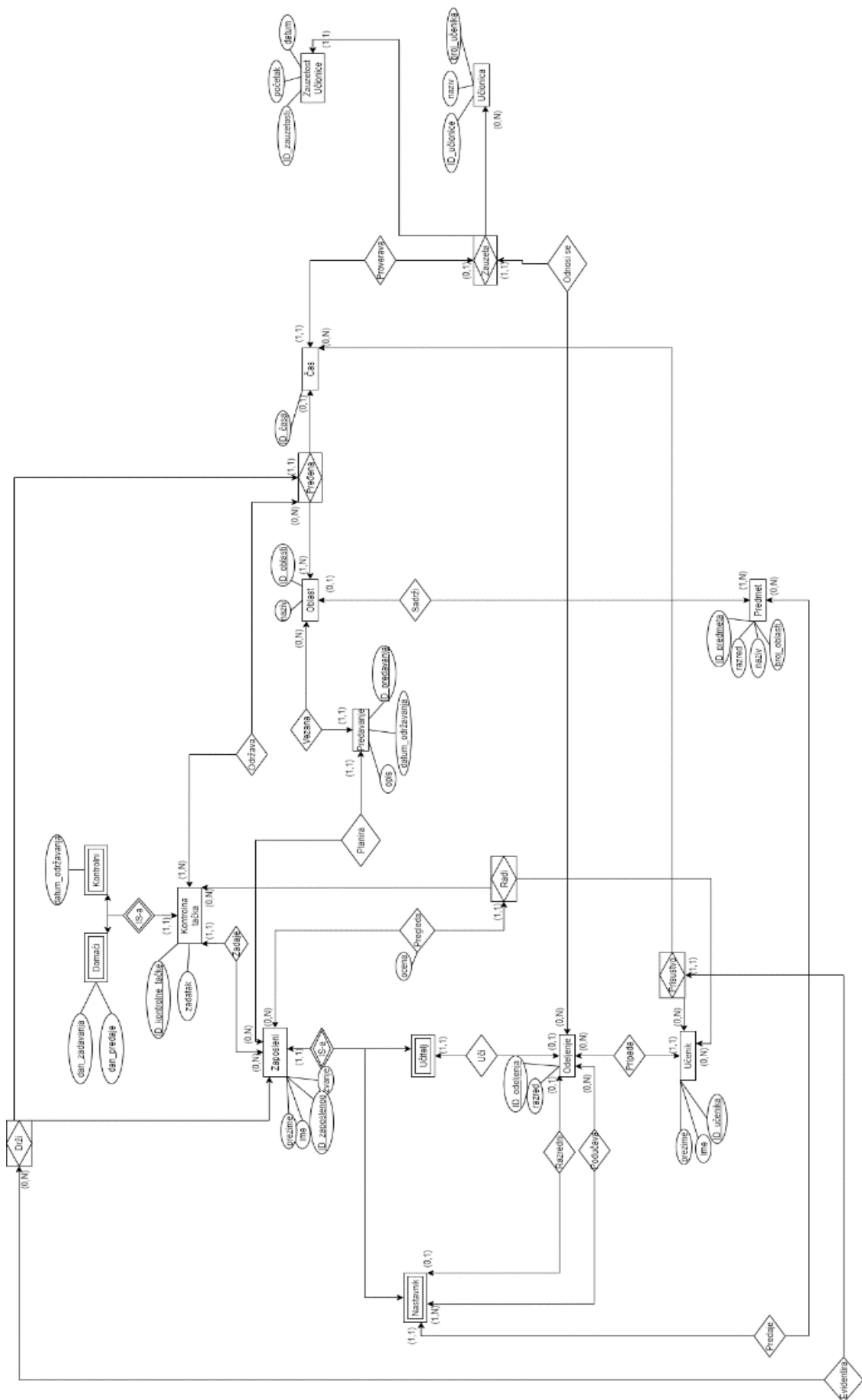
Основни концепти за моделовању у ЕР дијаграмима су типови ентитета са њиховим обележјима и типови повезника. Типови ентитета представљају ресурс пословања у реалном систему. Обележја типова ентитета су особине које тај ресурс поседује. Тип повезника представља однос у којем се ресурси реалног система налазе. При моделовању концептуалне шеме базе података, поред наведених појмова коришћени су још и герунд и ИС-А хијерархија. Герунд представља тип повезника који се може посматрати као тип ентитета у релацији са неким другим типом ентитета. ИС-А хијерархија је концепт који нам омогућава да моделујемо суперкласу и поткласе, односно наслеђивање [2].

При моделовању шеме базе података употребом *Entity Framework Designer* алата, типови ентитета представљају се преко концепта *Entity*, обележја се додају сваком *Entity*-у као *ScalarProperty*, док се типови повезника моделују помоћу *Association* концепта. У случајевима када имамо тип повезника са неким обележјем, то обележје се преноси у тип ентитета који има максимални кардиналитет 1. Тип повезника који са обе стране има максимални кардиналитет „више“ се у случајевима, ако има неко додатно обележје или представља тип ентитета у релацији са другим типом ентитета (прецизније ако је у питању герунд) моделује посебним *Entity*-ем. Герунд, осим у претходно описаном случају, прати иста правила као и обичан тип повезника при моделовању у *Entity Framework Designer*-у. ИС-А хијерархија се моделује при креирању *Entity*-а који су поткласе у хијерархији, тако што се из падајућег менија означеног као *Base type* изабере који од већ постојећих ентитета представља суперкласу.

Уколико се модел базе података прави употребом *Code First* приступа, у оквиру класе која представља тип ентитета, обележја наводимо преко пропертија одговарајућег типа, ИС-А хијерархију дефинишемо сходно правилима наслеђивања програмског језика који користимо, док се типови повезника реализују пропертијима класног типа који означавају тип ентитета са којим је класа у релацији.



Слика 3.1 – Концептуална шема базе података за смештање информација о корисницима



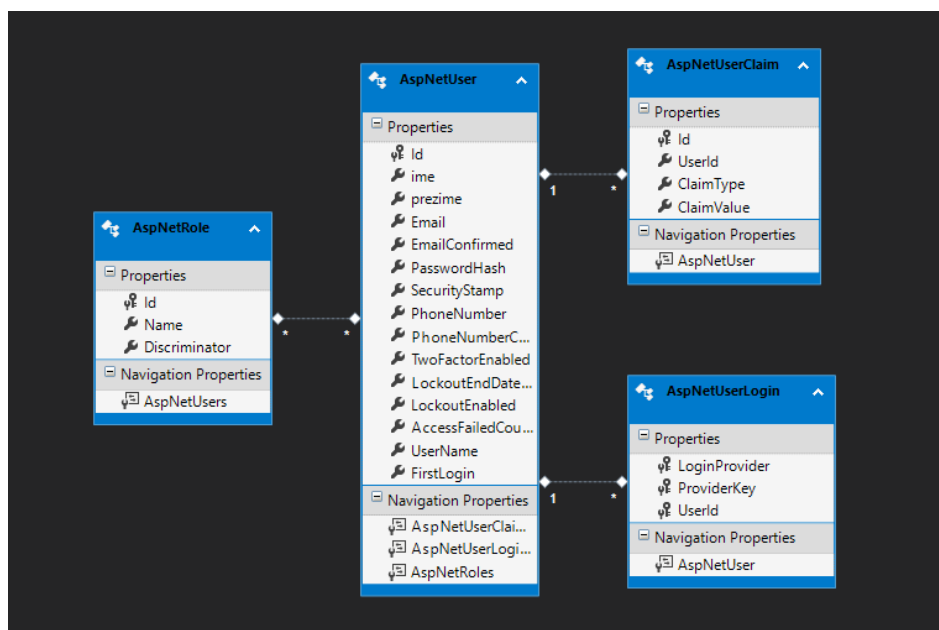
Слика 3.2 – Концептуална шема базе података за смештање информација које су потребне за рад запослених

## 3.2 Модел шеме базе података

Модели шема база података засноване су на концептуалној шеми базе података према правилима која су описана у претходном поглављу, уз посебан обзир на могућности за даља проширења. У циљу веће прегледности модел је представљен кроз више делова.

### 3.2.1 Део модела шеме базе података за смештање података о корисницима

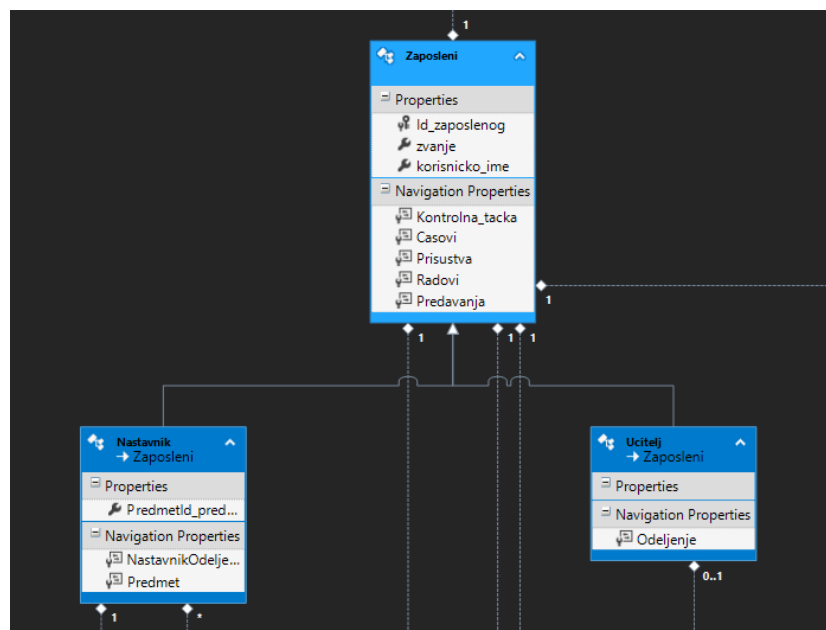
Корисничке информације представљају личне информације везане за запослене у школи, те су оне смештене у посебну базу ради додатне заштите. За креирање нове базе искоришћене су класе добијене у пакету *Microsoft.AspNet.Identity*. Изглед модела шеме базе података са овим информацијама дат је на слици испод (слика 3.3). Заједничке особине корисника представљене су типом ентитета *AspNetUser*. За корисника се везује тачно једна улога у оквиру система која је представљена типом ентитета *AspNetRole*. Типови ентитета *AspNetUserClaim* и *AspNetUserLogin* у тренутној имплементацији нису коришћени, али могу бити искоришћени за даљи развој апликације. *AspNetUserClaim* може се користити уместо улога за одређивање шта све корисник има могућност да уради у оквиру нашег информационог система, док се у *AspNetUserLogin* чувају информације о начинима на који корисник може да се пријави на систем коришћењем сервиса који нису део нашег система. Ова два типа ентитета би нам могла бити од користи када бисмо даље развијали информациони систем у веб апликацију, о чему је било речи у поглављу 2.3 *Перспектива система*.



Слика 3.3 – Део дијаграма модела шеме базе података за смештање података о корисницима

### 3.2.2 Део модела шеме базе података за смештање података о запосленима

Овај део модела шеме базе података обухвата типове ентитета и повезника потребних за чување података о запосленима (слика 3.4). Типови ентитета приказани на слици испод представљају збир информација које су неопходне за даљи рад запосленог, односно помоћу ових ентитета и њихових особина могуће је повезати запосленог са свим информацијама које су му потребне за рад на систему, као и приказати информације о раду који је у оквиру система запослени већ обавио.

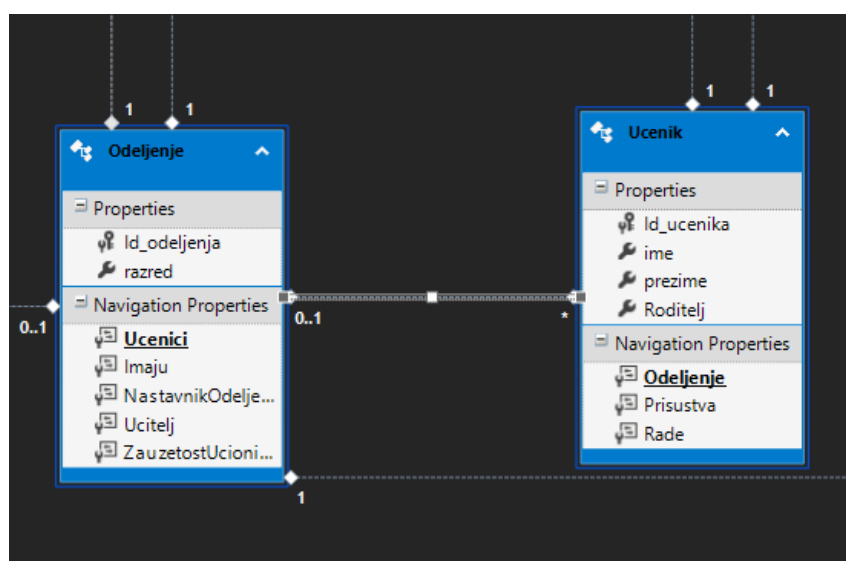


Слика 3.4 – Део дијаграма модела шеме базе података за смештање података о запосленима

Разлика између *Properties* и *NavigationProperties* је у томе што се ови други аутоматски генеришу из асоцијација између типова ентитета, тако на пример *Predmet* представља тип ентитета Предмет са којим је тип ентитета Наставник повезан. Такође, треба напоменути да је *korisnicko\_ime* особина искоришћена да повеже одређеног запосленог са његовим корисничким информацијама које се налазе у другој бази. Повезивање запосленог са његовим корисничким информацијама одрађено је на апликативном нивоу и о томе ће бити више речи у неком од следећих поглавља.

### 3.2.3 Део модела шеме базе података за смештање података о ученицима

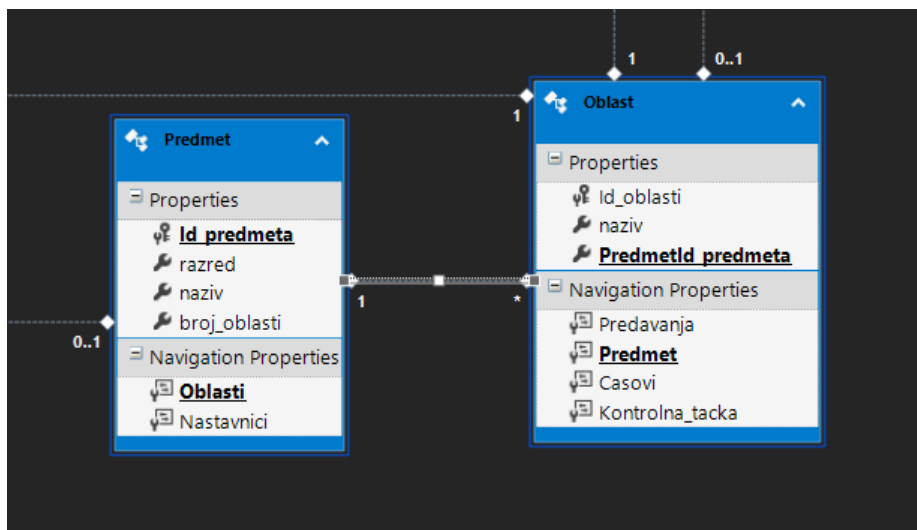
Подаци о ученицима који похађају школску установу као и о припадности једног ученика неком одељењу приказани су на слици испод (слика 3.5). На слици је подебљана асоцијација која представља припадност ученика једном одељењу, при чему су навигационе особине креиране на основу те асоцијације на слици приказане подебљаним словима.



Слика 3.5 – Део дијаграма модела шеме базе података за смештање података о ученицима и одељењима

### 3.2.4 Део модела шеме базе података за смештање података о предметима

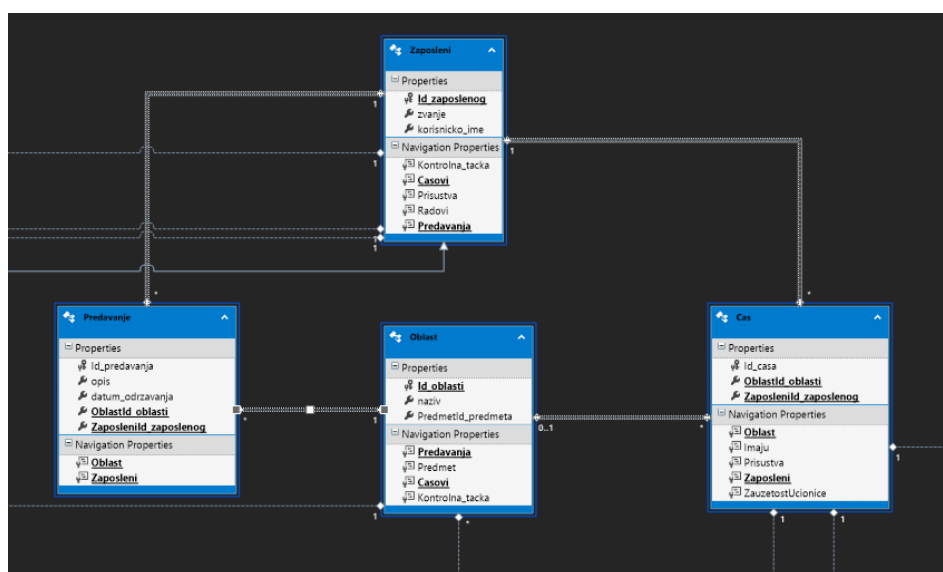
За рад запослених у образовној институцији неопходне су информације о предметима који се предају као и о областима које ти предмети садрже. На слици 3.6 приказани су типови ентитета искоришћени за складиштење тих информација. Веза између типа ентитета Предмет и типа ентитета Област, као и изгенерисане навигационе особине, су подебљане на слици. Такође, како предмет садржи више области дошло је до преноса страног кључа из Предмета у Област, ради бољег повезивања табела у бази и ефикаснијег приступа информацијама из апликације, што се може и приметити на слици.



Слика 3.6 – Део дијаграма модела шеме базе података за смештање информација о предмету и његовим областима

### 3.2.5 Део модела шеме базе података за смештање информација о предавањима и часовима

Једна од основних функционалности која мора бити омогућена запосленима образовне установе је да испланирају предавања и евидентирају часове. На слици испод (слика 3.7) можемо видети како изгледају и на који начин су повезани типови ентитета Предавање и Часови.



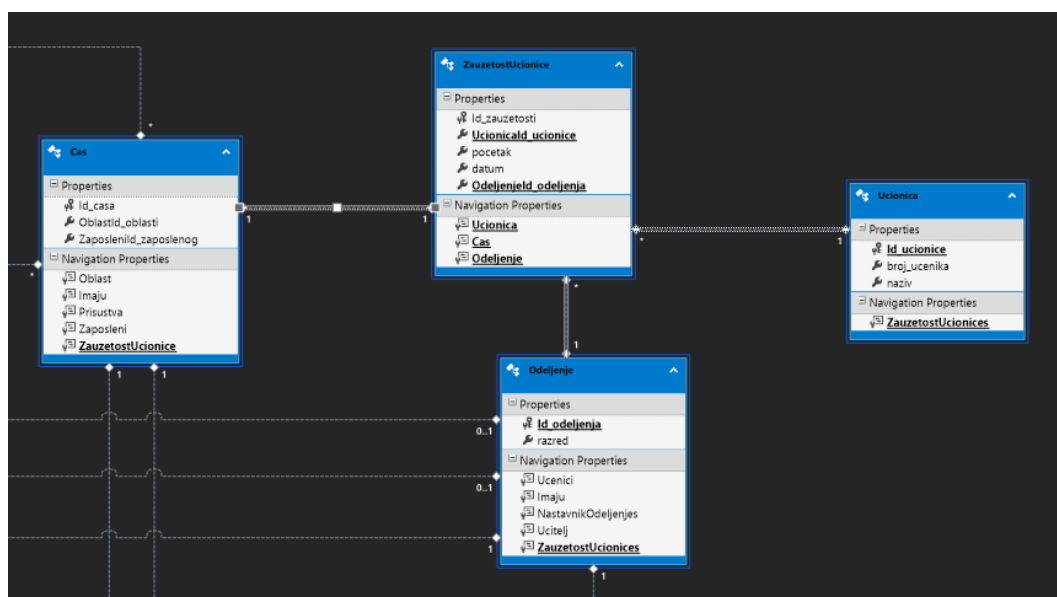
Слика 3.7 – Део дијаграма модела шеме базе података за смештање информација о предавањима и часовима

Као што се на слици може видети, Област је повезана и са Часом и са Предавањем, а како једна област може бити везана за више Часова и више Предавања, тако је дошло до преноса страног кључа из Области у Предавање, односно Час. Такође, један запослени може да држи више часова и да испланира више предавања, те је страни кључ пренесен из Запосленог у Час и Предавање.

Приликом евидентирања часова од интереса је и учионица у којој ће се час одржавати, као и одељење којем се час одржава. На слици испод (слика 3.8) приказани су типови ентитета Час, Учионица, Заузетост Учионице и Одељење, као и повезаност између њих.

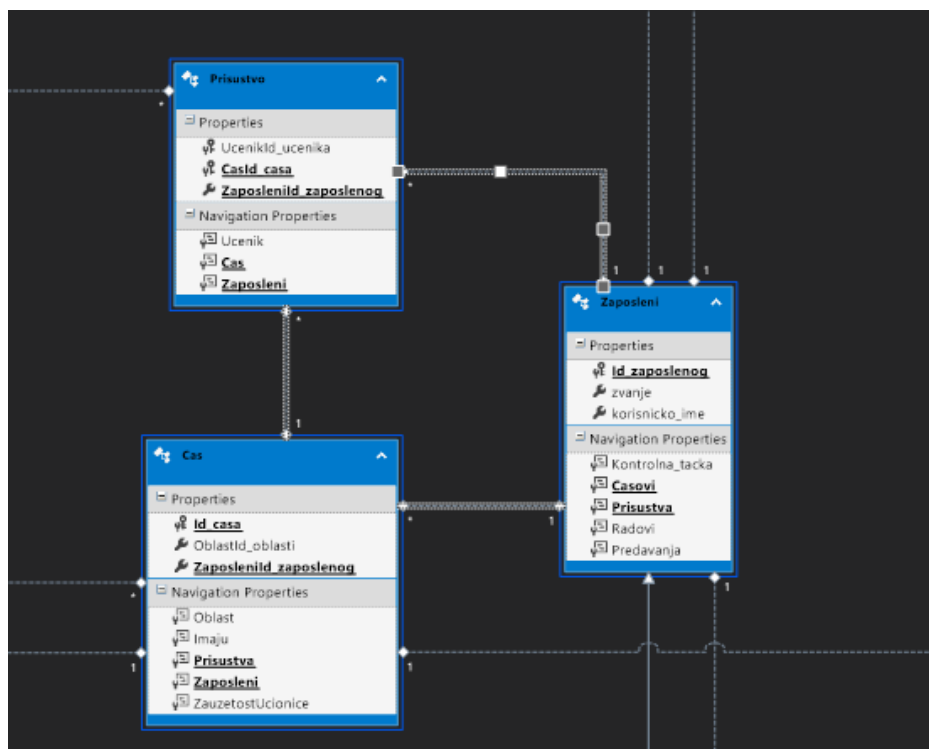
Учионица представља тип ентитета који садржи информације о учионицама које у образовној установи постоје, док је Заузетост Учионице тип ентитета у којој се бележе информације о датуму и времену када је учионица заузета, како не би дошло до двоструког резервисања учионица. Тип ентитета Одељење се у овом случају користи да се у оквиру Заузетости Учионице забележи које Одељење је заузето у одређено време.

Увођењем Заузетости Учионице као посебан тип ентитета омогућили смо да се све информације о заузетостима како учионице, тако и одељења налазе на једном месту. На основу повезаности између Часа и Заузетости Учионице, можемо проверити и заузетост Запосленог, јер као што се види на слици 3.7 у оквиру Часа је садржана информација о Запосленом који је евидентирао тај Час.



Слика 3.8 – Део дијаграма модела шеме базе података за смештање информација о заузетости учионица и одељења

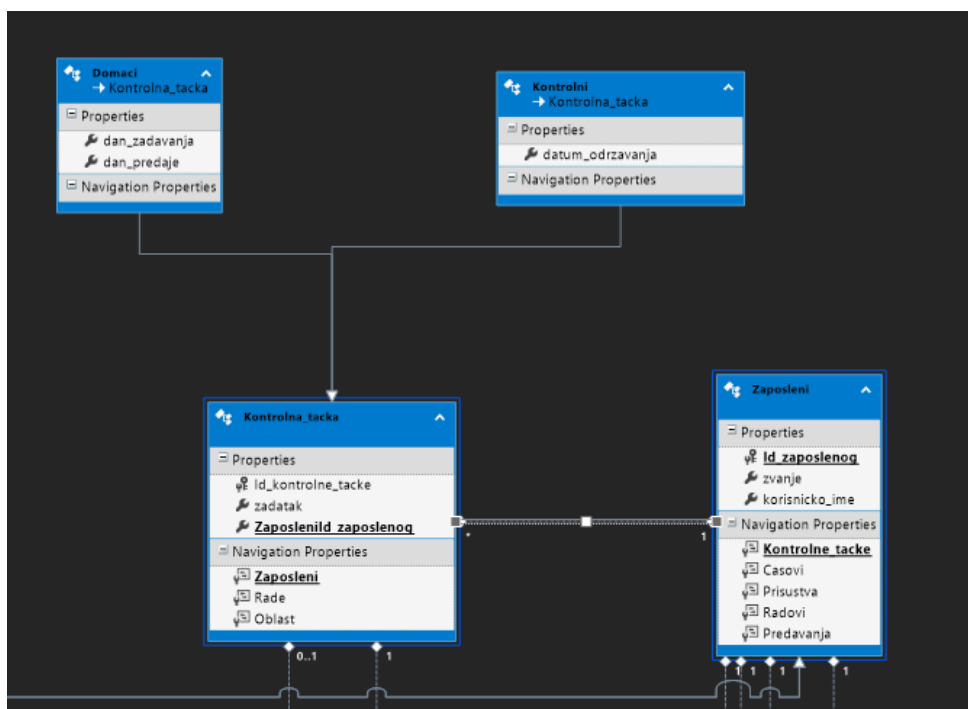
Иако се час планира између осталог и према одељењу којем ће бити држан, то не мора да значи да су сви ученици тог одељења били на одређеном часу. С обзиром на то да је битна информација да ли је ученик био на часу, направљен је посебан тип ентитета Присуство који ће садржати информацију о томе који је Ученик био на ком Часу. На слици испод (слика 3.9) можемо видети поменуте типове ентитета и везе између њих.



Слика 3.9 – Део дијаграма модела шеме базе података за смештање информација о присутности ученика на часу

### 3.2.6 Део модела шеме базе података за смештање података о контролним тачкама

Друга веома битна функционалност која треба бити омогућена запосленима је да евидентирају контролне и домаће задатке које планирају, као и да оцене радове ученика. На слици 3.10 приказани су типови ентитета Контролна тачка, Контролни и Домаћи као и начин на који смо повезали те типове са типом ентитета Запослени.

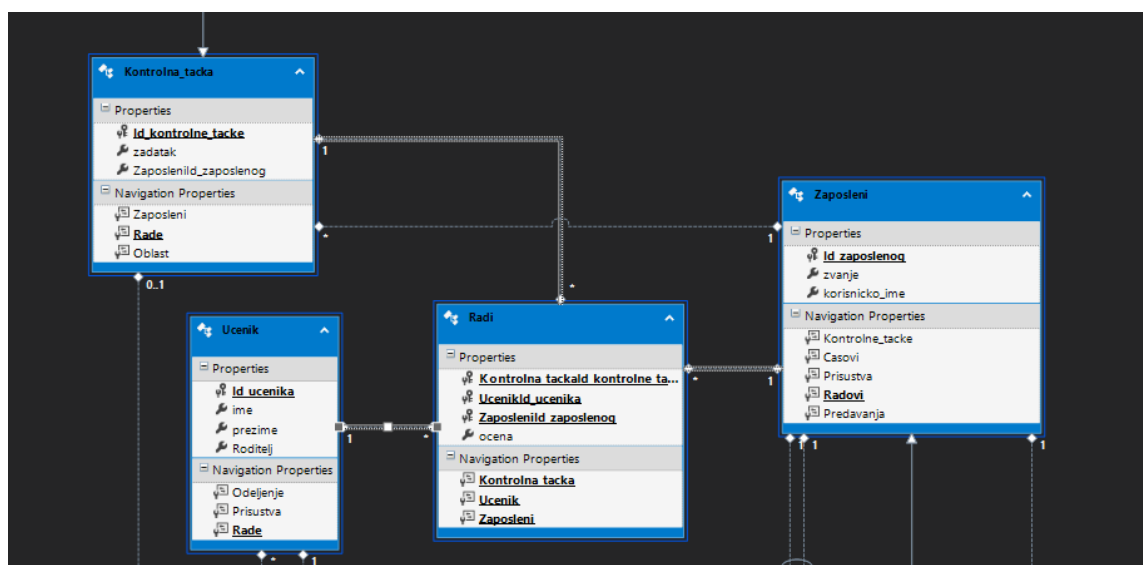


Слика 3.10 – Део дијаграма модела шеме базе података за смештање података о контролним и домаћим задацима



На слици можемо приметити да смо повезали тип ентитета Контролна тачка са типом ентитета Запослени, узимајући у обзир да је Контролна тачка родитељски тип ентитета у односу на Контролни и Домаћи, у навигационој особини Контролне\_тачке типа ентитета Запослени имаћемо информације и о Домаћима и о Контролним у складу са правилима наслеђивања.

Ради праћења постигнутог успеха ученика у школи, неопходна је евиденција његових оцена које је остварио на контролним, односно домаћим задацима. Ради бележења ових информација направљен је нови тип ентитета Ради. Овај тип ентитета и његову повезаност са Учеником, Запосленим и Контролном тачком види се на слици 3.11.



Слика 3.11 – Део дијаграма модела шеме базе података за смештање података о контролним и домаћим задацима које ученик ради

На слици се такође види да смо Ради повезали само са Контролном тачком, али као што је већ речено, у складу са правилима наслеђивања, ово нам је довољно да у навигациону особину Контролна\_тачка сместимо било Контролни, било Домаћи. Помоћу особине *ocena* у типу ентитета Ради бележимо коју је оцену ученик добио на задатом контролном, то јест домаћем.

### 3.3 База података генерисана из модела шема базе података

Модели шема података описани у претходном поглављу представљају основу за моделовање базе података која је искоришћена за генерисање саме базе података помоћу интегрисане програмске подршке у оквиру Microsoft Visual Studi-a 2019, а преко пакета Entity Framework 6.4.4. Поменути пакет омогућава да се на основу дефинисаног модела изгенерише база података, али и класе у C# које представљају типове ентитета.

На основу типова ентитета изгенерисане су табеле у бази и класе које представљају те ентитете. Асоцијације између типова ентитета су у оквиру класе претворене у пропертије, док су у бази креирани страни кључеви према правилима о преносу страног кључа.

За потребе информационог система при моделовању базе података коришћен је и Code First и Model First приступ. Разлике између ова два приступа су у томе што се у Code First приступу моделују класе и на основу њих креира база, док се у Model First приступу, као што назив и сугерише, прво креира модел базе података и онда, на основу њега, сама база података. Ефективно, базе приликом генерисања прате иста правила, тако на пример асоцијације у моделу,

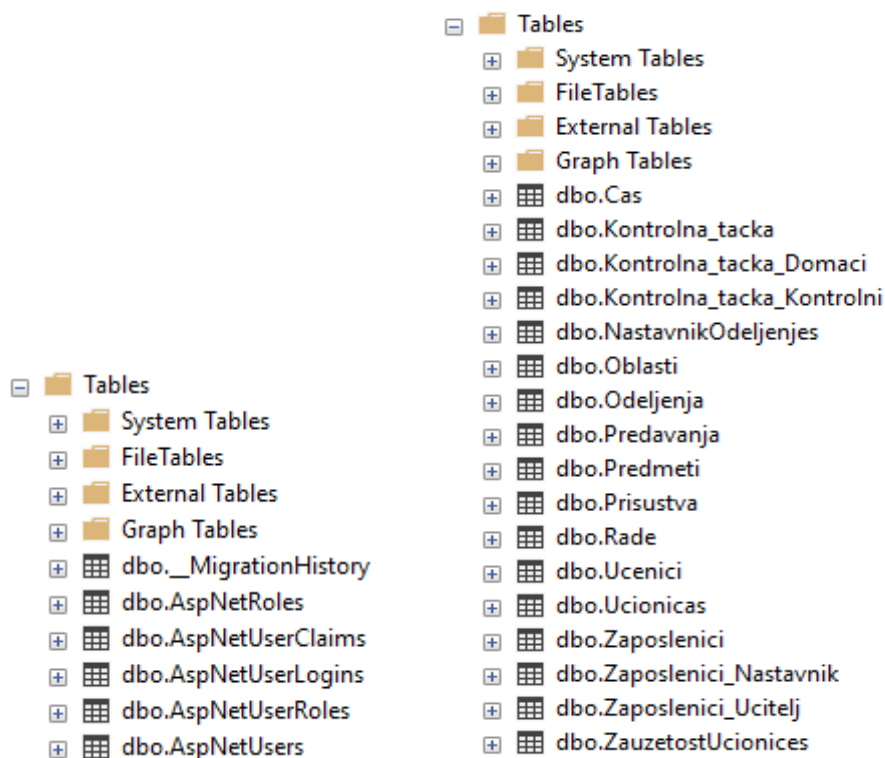
као што је и речено, у бази простају страни кључеви, а у класама пропертији. С друге стране пропертији у класама, при генерисању базе на основу њих, се такође претварају у стране кључеве сходно правилима о преносу страног кључа.

Разлог за коришћење оба приступа моделовању базе података, огледа се у могућностима које класе имплементирани у пакету Microsoft.AspNet.Identity поседују. Наиме, класе у оквиру овог пакета се најчешће користе за чување корисничких информација, што је и разлог зашто су управо ове класе искоришћене за моделовање базе података о корисницима. Сам пакет нуди напредне функционалности, као што су чување лозинке у кодираном облику и додељивање улога корисницима у систему. Ове особине поменутог пакета биће детаљније објашњене када буде описивано апликативно решење информационог система.

### 3.3.1 Изглед база података генерисаних за потребе информационог система

У оквиру овог дела поглавља биће представљен изглед самих база података које су генерисане према претходно наведеним правилима. Како су за потребе информационог система направљене две базе, једна за потребе складиштења корисничких информација и друга за смештање информација о самој школској установи, обе од њих биће презентоване примерима.

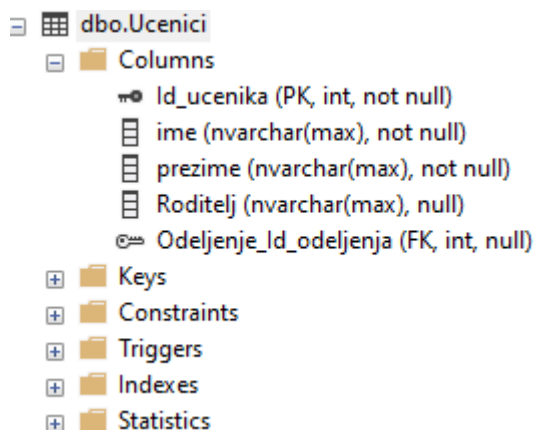
За потребе хостовања базе података коришћен је *Microsoft SQL Express* сервер, а као СУБП коришћен је *Microsoft SQL Server Management Studio*. СУБП је употребљен за креирање база података и за потребе прегледа начина на који су подаци смештани у базу података. На основу модела шеме базе података који су претходно описани креиране су табеле у базама (слика 3.12), док је платформа Microsoft Visual Studio 2019 омогућила рад са подацима у бази.



Слика 3.12 – Изглед табела у базама података коришћене у информационом систему, у оквиру *Microsoft SQL Server Management Studio-a*

На слици 3.12 види се једина разлика приликом креирања база података помоћу Code First и Model First приступа. Та разлика огледа се у додатној табели која није изгенерисана на основу нашег модела, а под називом `_MigrationHistory`. Додатна табела служи са чување метаподатака о миграцијама које су у оквиру развојног окружења направљене.

Као што је већ речено, генерисање базе података прати иста правила без обзира на приступ који користимо, стога ће изглед начина на који се информације у оквиру једне табеле смештају бити приказан на најрепрезентативнијој табели у информационом систему, а то је табела `Ucenici`. На слици испод (слика 3.13) види се табела са информацијама вазаним за њу које су организоване у потфолдере.



Слика 3.13 –Изглед табеле `Ucenici`, која је одабрана да репрезентује начин на који СУБП складишти информације о табелама у оквиру базе

У оквиру фолдера `Columns` садржане су информације о колонама које табела садржи. Као што се на слици 3.13 види примарни кључ табеле `Id_ucenika` визуелно је означен сликом обојеног кључа, док је страни кључ према табели `Odeljenja` означен необојеним кључем. Обележја `ime`, `prezime` и `Roditelj` представљени су сликом која симболизује колону у табели. Треба напоменути, да је разлог постојања страног кључа у оквиру ове табеле моделована асоцијација између ентитета `Ucenik` и `Odeljenje` у оквиру Entity Framework Designer-a. Такође, назив страног кључа је аутоматски генерисан на основу правила Entity Framework Designer-a.

	<code>Id_ucenika</code>	<code>ime</code>	<code>prezime</code>	<code>Roditelj</code>	<code>Odeljenje_Id_o...</code>
▶	8	Mirko	Mirkovic	bokimarin97@g...	7
	9	Nikola	Nikolic	bokimarin97@g...	7
*	NULL	NULL	NULL	NULL	NULL

Слика 3.14 –Изглед података смештених у табелу `Ucenici`

На слици изнад (слика 3.14) види се начин на који СУБП уписује податке у оквиру табеле `Ucenici`. Значење података које смештамо у базу описани су у следећем поглављу.

### 3.4 Опис ентитета и асоцијација у моделима

У овом поглављу представљен је табеларни приказ свих ентитета у базама података који су коришћени у информационом систему и описи асоцијација које ентитет има ка другим ентитетима.

#### *Ентитет `AspNetRoles`*

Ентитет `AspNetRoles` моделује могуће улоге корисника. Свака улога дефинисана помоћу торке у оквиру табеле означава посебан тип корисника у оквиру информационог система.

Табела 3.1 – Обележја у ентитету `AspNetRoles`

Назив обележја	Тип података	Nullable	Опис обележја
Id	String	Не	Примарни кључ
Name	String	Не	Назив улоге
Discriminator	String	Не	Опис улоге

#### *Ентитет `AspNetUser`*

Ентитет `AspNetUser` моделује корисника информационог система. Ентитет се налази у оквиру пакета `Microsoft.AspNet.Identity`, и проширен је обележјима `ime`, `prezime` и `FirstLogin`. Подаци из табеле настале од овог ентитета користе се за бележење корисничких информација.

Табела 3.2 – Обележја у ентитету `AspNetUser`

Назив обележја	Тип података	Nullable	Опис обележја
Id	String	Не	Примарни кључ
ime	String	Да	Име корисника
prezime	String	Да	Презиме корисника
Email	String	Да	Е-мејл корисника
EmailConfirmed	Boolean	Не	Е-мејл је потврђен
PasswordHash	Hash Strng	Да	Лозинка
SecurityStamp	Hash String	Да	Рандом вредност која се мења кад год се лозинка промени
PhoneNumber	String	Да	Телефон запосленог
PhoneNumberConfirmed	Boolean	Не	Телефон је потврђен
TwoFactorEnabled	Boolean	Не	Двострука аутентификација је омогућена
LockoutEndDateUtc	Date Time	Да	Датум до ког корисник нема приступ систему
LockoutEnabled	Boolean	Не	Закључавање корисника ван система је омогућено
AccessFailedCount	Integer	Не	Број неуспелих покушаја пријаве на систем
UserName	String	Не	Корисничко име
FirstLogin	Boolean	Не	Означава прву пријаву на систем.

AspNetUser ентитет има асоцијацију са *AspNetRoles* ентитетом која представља тип корисника у информационом систему.

### Ентитет Cas

Ентитет *Cas* моделује школски час који запослени заказује помоћу информационог система.

Табела 3.3 – Обележја у ентитету Cas

Назив обележја	Тип података	Nullable	Опис обележја
Id_casa	Integer	He	Примарни кључ
OblastId_oblasti	Integer	Да	Страни кључ
ZaposlenId_zaposlenog	Integer	He	Страни кључ
ZauzetostUcionice_Id_zauzetosti	Integer	He	Страни кључ

Као што се може и приметити на основу присуства страних кључева у табели изнад, Ентитет *Cas* има асоцијације према ентитетима *Oblast*, *Zaposleni* и *ZauzetostUcionice*.

### Ентитет Kontrolna\_tacka

Ентитет *Kontrolna\_tacka* представља контролне тачке које запослени може да испланира. У оквиру система је направљена да буде надтип ентитетима *Kontrolni* и *Domaci*.

Табела 3.4 – Обележја у ентитету Kontrolna\_tacka

Назив обележја	Тип података	Nullable	Опис обележја
Id_kontrolne_tacke	Integer	He	Примарни кључ
zadatak	String	He	Опис контролне тачке
ZaposlenId_zaposlenog	Integer	He	Страни кључ
OblastId_oblasti	Integer	He	Страни кључ

Ентитет *Kontrolna\_tacka* има асоцијације према ентитетима *Zaposleni* и *Oblast*.

### Ентитет Domaci

Ентитет *Domaci* представља домаћи задатак који запослени може да евидентира. Његов надтип је ентитет *Kontrolna\_tacka* и од тог ентитета се наслеђује примарни кључ.

Табела 3.5 – Обележја у ентитету Domaci

Назив обележја	Тип података	Nullable	Опис обележја
Id_kontrolne_tacke	Integer	He	Примарни кључ
dan_zadavanja	Date Time	He	Датум када се домаћи задаје
dan_predaje	Date Time	He	Датум када се домаћи предаје на оцењивање

**Ентитет Kontrolni**

Ентитет *Kontrolni* моделује контролни задатак који запослени може да дефинише.

Табела 3.6 – Обележја у ентитету Kontrolni

Назив обележја	Тип података	Nullable	Опис обележја
Id_kontrolne_tacke	Integer	Ne	Примарни кључ
datum_odrzavanja	Date Time	Ne	Датум када се контролни одржава

**Ентитет NastavnikOdeljenje**

Ентитет представља везу између ентитета *Nastavnik* и *Odeljenje*. Овај ентитет је моделован како би се у њему бележиле информације о наставницима који предају одређеним одељењима.

Табела 3.7 – Обележја у ентитету NastavnikOdeljenje

Назив обележја	Тип података	Nullable	Опис обележја
NastavnikId_zaposlenog	Integer	Ne	Примарни кључ
Odeljenjeld_odeljenja	Integer	Ne	Примарни кључ
Razredni	Boolean	Ne	Наставник је одељењу разредни

Ентитет има асоцијације са ентитетима *Nastavnik* и *Odeljenje*, с тим што су пренети страни кључеви искоришћени за прављење комбинованог примарног кључа

**Ентитет Predmet**

Ентитет *Predmet* моделује предмете који се предају у школи.

Табела 3.8 – Обележја у ентитету Predmet

Назив обележја	Тип података	Nullable	Опис обележја
Id_predmeta	Integer	Ne	Примарни кључ
razred	Short Integer	Ne	Разред којем је предмет намењен
naziv	String	Ne	Назив предмета
broj_oblasti	Short Integer	Ne	Број области предмета

**Ентитет Oblast**

Ентитет *Oblast* представља област једног предмета.

Табела 3.9 – Обележја у ентитету Oblast

Назив обележја	Тип података	Nullable	Опис обележја
Id_oblasti	Integer	Ne	Примарни кључ
naziv	String	Ne	Назив области
PredmetId_predmeta	Integer	Ne	Страни кључ

Овај ентитет има једну асоцијацију и то према ентитету *Predmet* за који је везан.

### Ентитет *Ucenik*

Ентитет садржи обележја која описују ученике који похађају школску установу.

Табела 3.10 – Обележја у ентитету *Ucenik*

Назив обележја	Тип података	Nullable	Опис обележја
Id_ucenika	Integer	Не	Примарни кључ
ime	String	Не	Име ученика
prezime	String	Не	Презиме ученика
Roditelj	String	Да	Е-мејл родитеља ученика
Odeljenjeld_odeljenja	Integer	Да	Страни кључ

Ентитет има асоцијацију према ентитету *Odeljenje* која симболизује припадност одељењу.

### Ентитет *Odeljenje*

Ентитет моделује одељења школске установе.

Табела 3.11 – Обележја у ентитету *Odeljenje*

Назив обележја	Тип података	Nullable	Опис обележја
Id_odeljenja	Integer	Не	Примарни кључ
razred	Short Integer	Не	Разред одељења
UciteljId_zaposlenog	Integer	Да	Страни кључ

*Odeljenje* има асоцијацију према ентитету *Ucitelj*.

### Ентитет *Predavanje*

Ентитет представља предавање које један запослени може да евидентира у оквиру информационог система.

Табела 3.12 – Обележја у ентитету *Predavanje*

Назив обележја	Тип података	Nullable	Опис обележја
Id_predavanja	Integer	Не	Примарни кључ
opis	String	Не	Кратак опис планираног предавања
datum_odrzavanja	Date Time	Не	Датум када се предавање одржава
OblastId_oblasti	Integer	Не	Страни кључ
ZaposleniId_zaposlenog	Integer	Не	Страни кључ

*Predavanje* има асоцијације према *Zaposlenom* који га планира и *Oblasti* за коју је везан.

**Ентитет *Prisustvo***

Ентитет служи за бележење да је ученик присуствовао одређеном часу.

Табела 3.13 – Обележја у ентитету *Prisustvo*

Назив обележја	Тип података	Nullable	Опис обележја
UcenikId_ucenika	Integer	He	Примарни кључ
CasId_casa	Integer	He	Примарни кључ
ZaposleniId_zaposlenog	Integer	He	Страни кључ

*Prisustvo* има асоцијације према ентитетима *Ucenik*, *Cas* и *Zaposleni*, при чему су пренесени страни кључеви из часа и ученика искоришћени за формирање примарног кључа ове табеле.

**Ентитет *Radi***

Ентитет садржи обележја која повезују одређену контролну тачку коју је ученик радио и у којој се садржи информација о оцени коју је на њој добио.

Табела 3.14 – Обележја у ентитету *Radi*

Назив обележја	Тип података	Nullable	Опис обележја
Kontrolna_tackaId_kontrolne_tacke	Integer	He	Примарни кључ
UcenikId_ucenika	Integer	He	Примарни кључ
ZaposleniId_zaposlenog	Integer	He	Примарни кључ
ocena	Short Integer	He	Оцена коју је ученик добио на контролној тачци

Ентитет има асоцијације према ентитетима *Kontrolna\_tacka*, *Ucenik* и *Zaposleni*, при чему су пренесени страни кључеви из ових асоцијација искоришћени за формирање примарног кључа.

**Ентитет *Ucionica***

Ентитет представља учионицу која у школској установи постоји.

Табела 3.15 – Обележја у ентитету *Ucionica*

Назив обележја	Тип података	Nullable	Опис обележја
Id_ucionice	Integer	He	Примарни кључ
broj_ucenika	Integer	He	Број ученика које учионица смешта
naziv	String	He	Назив учионице

**Ентитет *Zaposleni***

Ентитет моделује запосленог у школској установи и служи да помоћу њега и његових подтипова имамо повезане информације о забележеном раду запосленог у оквиру информационог система.



Табела 3.16 – Обележја у ентитету *Zaposleni*

Назив обележја	Тип података	Nullable	Опис обележја
Id_zaposlenog	Integer	He	Примарни кључ
zvanje	String	He	Звање које је запослени стекао
korisnicko_ime	String	He	Корисничко име запосленог у систему

**Ентитет *Nastavnik***

Ентитет је подтип ентитета *Zaposleni* од ког наслеђује примарни кључ и повезује *Nastavnika* са *Predmetom* који предаје преко асоцијације између та два ентитета.

Табела 3.17 – Обележја ентитета *Nastavnik*

Назив обележја	Тип података	Nullable	Опис обележја
Id_zaposlenog	Integer	He	Примарни кључ
PredmetId_predmeta	Integer	Да	Страни кључ

**Ентитет *Ucitelj***

Ентитет моделује учитеља у школској установи и представља подтип ентитета *Zaposleni* од ког наслеђује примарни кључ.

Табела 3.18 – Обележја у ентитету *Ucitelj*

Назив обележја	Тип података	Nullable	Опис обележја
Id_zaposlenog	Integer	He	Примарни кључ

**Ентитет *ZauzetostUcionice***

Ентитет служи за бележење информација о заузетости учионице.

Табела 3.19 – Обележја у ентитету *ZauzetostUcionice*

Назив обележја	Тип података	Nullable	Опис обележја
Id_zauzetosti	Integer	He	Примарни кључ
UcionicId_ucionice	Integer	He	Страни кључ
OdeljenjId_odeljenja	Integer	He	Страни кључ
datum	Date Time	He	Датум када је учионица заузета
pocetak	Time	He	Почетак часа од 45 минута када је учионица заузета

*ZauzetostUcionice* има асоцијације са ентитетима *Odeljenje* и *Ucionica* што се у табели види присуством страних кључева.

### 3.5 Улога базе података

База података представља основу за развијање одговарајуће програмске подршке запосленима. Као што се види из моделованих ентитета у бази ће бити складиштене све информације које су запосленима у школској установи од значаја. Из начина на који се повезују ентитети примећује се комплексност модела базе података што истовремено наглашава њихову релевантност.

У следећем поглављу описано је апликативно решење развијено за потребе информационог система, а на основу моделоване базе података. Заједно са базом, апликација представља софтверски пакет намењен једној школској установи. Тим пакетом, као што је речено, олакшава се рад запослених у оквиру образовне институције.

## 4. Опис апликативног решења

У овом поглављу биће описано апликативно решење уз кратак освит на архитектуру софтвера и пакете коришћене при реализацији информационог система. При презентовању апликативног решења биће објашњене сложеније функционалности подељене према типу корисника.

Апликативно решење је реализовано као рачунарска апликација, где је кориснички интерфејс имплементиран коришћењем WPF графичког подсистема, водећи се MVVM програмским обрасцем.

### 4.1 Софтверска архитектура

При развијању информационог система у апликативном решењу је коришћена сервисно оријентисана архитектура која се заснива на подели имплементираних функционалности на групе према одређеним особинама, а у циљу повећања скалабилности и поновне употребљивости самог програмског кода[3].

Функционалности развијене за потребе информационог система су подељене на четири логички независна пројекта у оквиру *solution*-а посматране апликације. Од пројеката у апликацији постоје:

- *AuthTesting* – управља корисничким информацијама како на апликативном новоу тако и на нивоу базе података
- *OsnovnaSkola* – садржи шему модела базе података на основу које је изгенерисана база података нужна за рад запослених у школи, као и класе за приступ подацима у бази
- *OsnovnaSkolaPL* – у оквиру овог пројекта издвојена је пословна логика апликације
- *OsnovnaSkolaUI* – дефинише кориснички графички интерфејс

Размена информација између претходно описаних пројеката реализована је употребом WCF канала и то на следећи начин: пројекат *OsnovnaSkolaUI* представља клијента у односу на пројекат *OsnovnaSkolaPL*, који репрезентује сервер и где је дефинисана пословна логика која користи елементе за приступ бази дефинисане у оквиру пројекта *OsnovnaSkola*; такође је отворена нова конекција између *OsnovnaSkolaPL*, који представља, у овом случају, клијента у односу на пројекат *AuthTesting*, који има улогу сервера и приступа бази са корисничким информацијама.

Ради реализовања сервисно оријентисане архитектуре, односно у циљу постизања жељене одвојености програмске логике на различите пројекте у којима ће се обрађивати специфични захтеви и давати приступ само одређеним подацима од значаја, употребљен је управо WCF који обезбеђује захтеване функционалности без великих измена у начину моделовања и руковања класама. WCF комуникација је развијена управо за потребе комуникације између сервиса у сервисно оријентисаним апликацијама [4]. Она такође пружа могућност сигурног протока информација кроз мрежу коришћењем добро знаних стандарда као што је на пример SSL протокол.

### 4.2 Коришћени пакети

Microsoft Visual Studio развојно окружење подржава механизам преко ког програмери могу развијати, делити и искористити постојеће имплементације које су скупљене у „пакете“ компајлираног кода. Тај механизам се назива *NuGet*, и он дефинише како су пакети креирани, хостовани и употребљавани у оквирима *.NetFramework*-а [5].

За потребе апликативног решења додата су и искоришћена два *NuGet* пакета под називима: „*Microsoft.AspNet.Identity*“ верзија 2.2.3 и „*Microsoft.EntityFrameworkCore*“ верзија 6.4.4.

Функционалности које пакет *EntityFramework* пружа огледају се у раду са базом. У претходним поглављима више пута је поменут *Entity Framework Designer*, алат искоришћен за креирање дијаграма модела шеме базе података о запосленима. Овај алат се налази у оквиру поменуте верзије *EntityFramework* пакета. Пакет такође омогућује писање класа за рад са базом, као и креирање базе података *Code First* приступом.

Из пакета *AspNet.Identity* употребљене су класе *IdentityUser* и *IdentityRole*, за моделовање информација о корисницима и њиховим улогама у систему респективно. *IdentityUser* класа садржи већину информација које су неопходне за моделовање корисника у оквиру неког информационог система, због чега је употребљена као родитељска класа у односу на класу *ApplicationUser* преко које смо моделовали корисника, где су њој додати пропертији: *ime*, *prezime* и *FirstLogin*.

Оно што је од великог интереса за развој апликације је начин на који се лозинка складишти у бази, а што је функционалност *AspNet.Identity* пакета. Наиме, пакет нам за приступ бази пружа већ имплементирану класу под називом *UserManager*, а која при упису корисничких информација лозинку смешта у енкриптованом облику добијеном *hash*-ирањем унете лозинке. Такође, помоћу те класе је могуће верификовати корисника преко методе *CheckPassword*, која од аргумената прима објекат типа *ApplicationUser* (којег проналазимо на основу унетог корисничког имена) и *string* променљиву која репрезентује лозинку коју корисник уноси. Сама метода *hash*-ира прослеђену лозинку и пореди је са вредношћу која већ постоји у бази, ако се унета и постојећа лозинка не поклапају метода враћа *false*.

При упису нових корисничких информација у базу лозинка се *hash*-ира аутоматским позивом тзв. „*Key Derivation Function*“ функције која је названа *Rfc2898DeriveBytes(String, Int32, Int32)* [6] и која је уграђена у сам *.NET Framework*. Иста функција се позива приликом позива *CheckPassword* методе *UserManager* класе, с тим што се у оквиру те функције врши и провера да ли се лозинка прослеђена *CheckPassword* методи после *hash*-ирања поклапа са оном вредношћу која стоји у бази у колони *PasswordHash* за одговарајућег корисника.

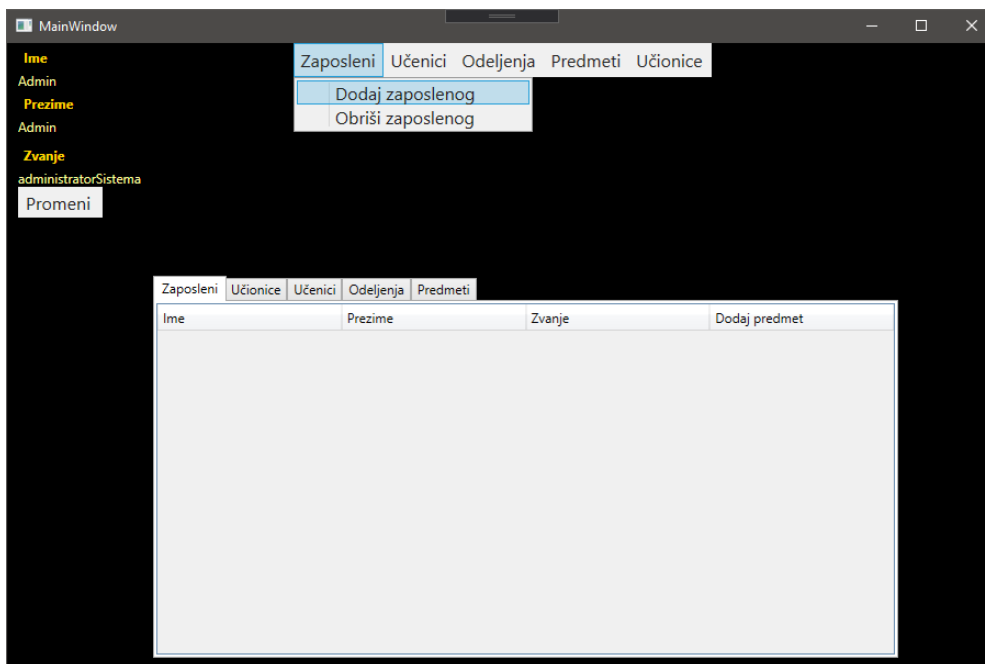
Као што се може приметити, једном унета лозинка се из базе никада неће повући у свом читком облику, већ ће се употребљавати само њена *hash*-ирана вредност. Ове и бројне друге могућности које *AspNet.Identity* пакет пружа, на пример: двофактроска аутентификација, пријава на систем преко других сервиса, забрана кориснику приступ систему и сл. су разлог зашто је пакет искоришћен при моделовању базе корисничких информација. Требало би напоменути да наведене функционалности нису имплементирани у оквиру постојећег апликативног решења али да представљају одличну основу за потенцијална проширења. Такође, постоје и новије верзије пакета са додатним проширењима, али оне не могу да се користе у оквиру *.NetFramework-a*.

### 4.3 Администраторске функционалности

Основни задатак администратора је да систем иницијализује подацима који су запосленима неопходни за ефикасан рад. Изузев креирања корисничких налога, у његове функционалности спадају и креирање, измена и по потреби брисање информација о учioniцама, ученицима, одељењима и предметима који се предају у школи. Због значаја и сложености детаљно је објашњен процес креирања корисничких информација, док је рад са осталим информацијама, иако битан, занемарен јер се своди на унос података преко графичког корисничког интерфејса и њихово додавање у базу података.

### 4.3.1 Креирање корисничких информација

Под креирањем корисничких информација подразумева се унос основних информације о запосленом и додељивање улоге у оквиру система у складу са којом запослени може да позива функционалности система.

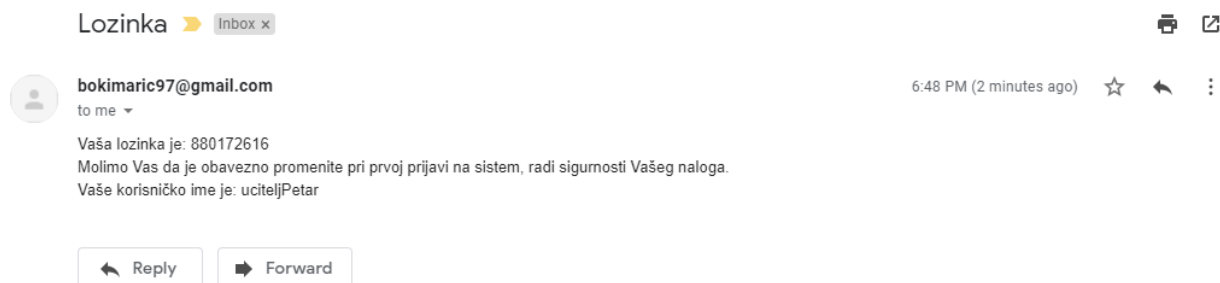


Слика 4.1 – Изглед графичког интерфејса намењеног администратору система

На слици 4.1 се види изглед прозора који је намењен администраторима за приказ опција за извршавање одређених функционалности. Кликот на дугме *Dodaj zaposlenog* из падајућег менија *Zaposleni* отвара се нови прозор преко којег се уносе основне корисничке информације везане за запосленог којем се прави кориснички налог. Изглед прозора дат је на слици испод (слика 4.2).

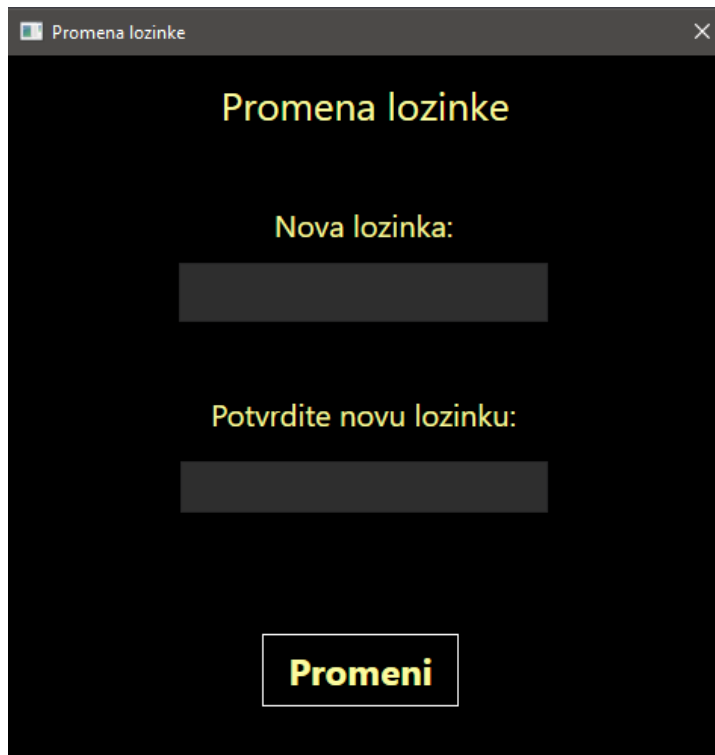
Слика 4.2 – Изглед прозора за унос корисничких информација о запосленом

Кликом на дугме *Dodaj* креира се кориснички налог са унетим информацијама. Лозинка намењена новокреираном запосленом се аутоматски генерише и представља деветоцифрени број. Уколико је све прошло без грешке и нове корисничке информације су забележене у бази, на мејл запосленог стиже порука о лозинки и креираном корисничком имену. Изглед мејла дат је на слици испод (слика 4.3). У циљу симулације ове функционалности искоришћен је мејл аутора рада.



Слика 4.3 – Изглед мејла који садржи лозинку и корисничко име запосленог у оквиру система

Аутоматским генерисањем шифре од стране информационог система и потом слањем креираних креденцијала сваком од додатих запослених, олакшавамо посао администратора система и спречавамо потенцијалне злоупотребе. У циљу повећања безбедности корисничких података, при првој пријави на систем од корисника се захтева промена шифре, тако што се отвара прозор за њену промену (слика 4.4).



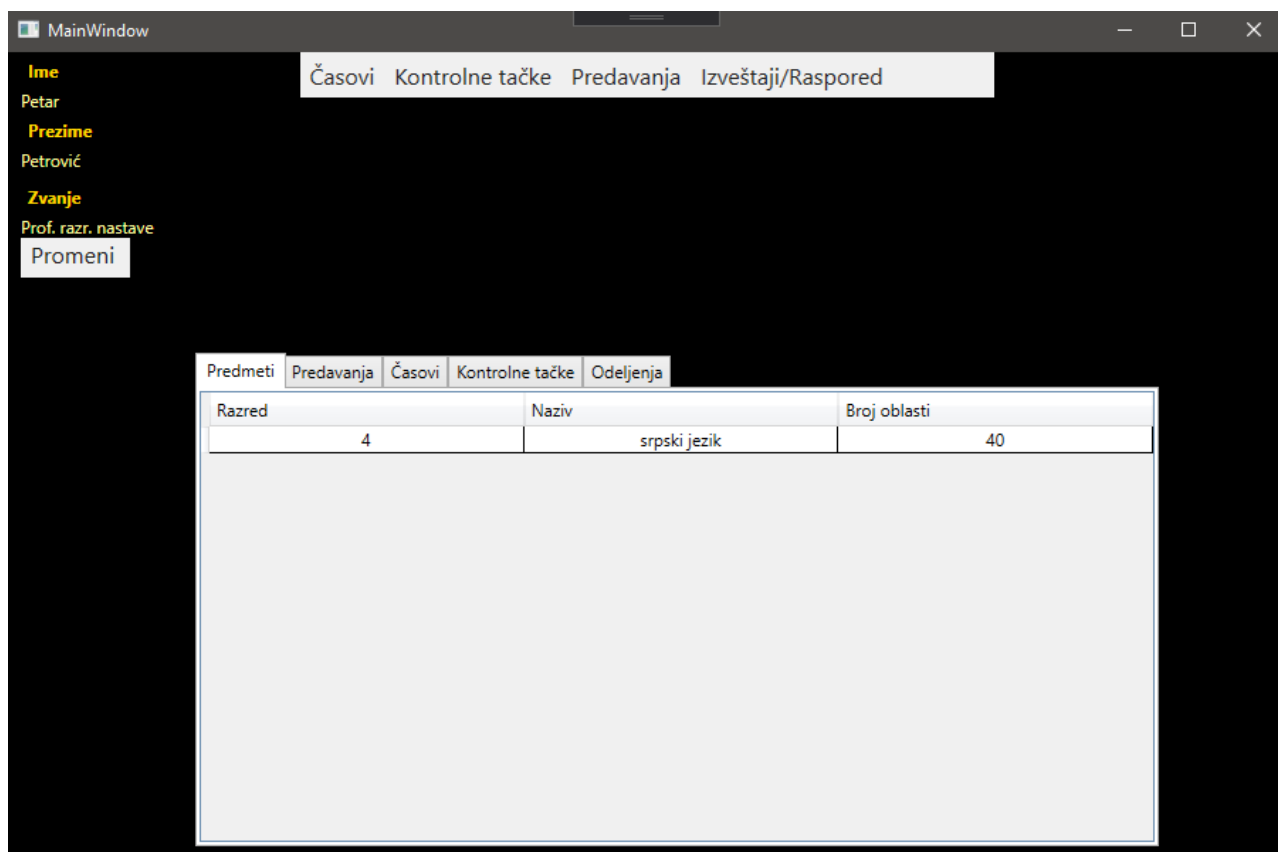
Слика 4.4 – Изглед прозора за промену лозинке

Као што се на слици види потребно је унети шифру и потврдити, што је мера која обезбеђује да корисник није погрешно унео лозинку.

По уносу корисничких информација, уколико су унети подаци валидирани успешно, они се преко WCF канала шаљу на даљу обраду у пројекат *OsnovnaSkolaPL*. Ти подаци се преко другог WCF канала прослеђују у *AuthTesting* пројекат где се прво покушава направити кориснички налог, уколико из неког разлога креирање корисничког налога није могуће, што ће се десити уколико унето корисничко име већ постоји у систему, нови налог се не креира и грешка се пријављује администратору. Ако је пак, кориснички налог успешно креиран, онда се креира и ред у табели *Zaposleni* који повезује корисника са свим информацијама у систему које су везане за његов налог.

#### 4.4 Функционалности запослених

Информациони систем подржава све функционалности неопходне за бележење рада запослених у оквиру школске установе. Већина њих се своди на просто додавање информација од значаја у базу података и неће бити даље разматране. Функционалности које су у наставку поглавља описане издвојене су или због своје сложености или због тога што представљају искорак у начину рада запослених, а те функционалности су: додавање часова и слање информација о ученику његовом родитељу. На слици испод (слика 4.5) приказан је прозор који се отвара по успешној пријави на систем од стране запосленог.



Слика 4.5 – Изглед графичког интерфејса који је намењен запосленом

##### 4.4.1 Креирање новог часа

На слици 4.5 видимо мени у оквиру ког стоји дугме под називом *Časovi*, кликом на њега отвара се падајући мени у оквиру ког имамо опцију да додамо час или да евидентирамо присуства ученика на неком одређеном часу. Ако се одабере опција додавања новог часа отвара се прозор приказан на слици 4.6, под условом да је претходно изабран један од понуђених предмета, у супротном се приказује грешка кориснику у којем пише да се прво мора одабрати предмет.

Izaberite učionicu		Izaberite oblast	
Naziv	Broj učenika	Naziv	Id predmeta
A1	30	padezi	2
A2	25		
Svečana sala	200		
Sala za fizičko	100		

**Napravi čas**

Слика 4.6 – Прозор за одабир учионице и области предмета

Прозор са слике изнад је први корак при одабиру информација од значаја за креирање новог часа. Помоћу њега, као што се на слици и види, бирамо учионицу у којој корисник жели да држи час и област изабраног предмета на коју се планирани час односи. По одабиру учионице и области отвара се нови прозор приказан на слици 4.7, уколико учионица или област није одабрана пријавиће се грешка кориснику у којој се наводи да је неопходно одабрати учионицу, односно час.

**ČAS**

Datum održavanja:

06-Oct-20 15

Početak:

17:30

Razred	Razredni
4	Petar

**Dodaj**

Слика 4.7 – Прозор за одабир одељења којем је час намењен и датума и времена одржавања часа



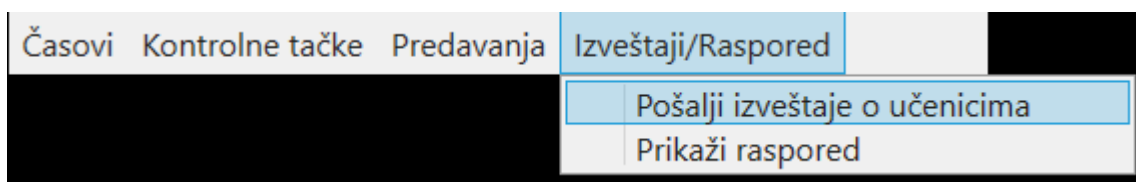
У оквиру овог прозора од корисника се тражи да унесе датум и време када је час планиран, као и да одабере одељење којем је час намењен. Уколико се изостави неки од ових података одговарајућа информација о грешци ће кориснику бити пријављена, у супротном информације о новом часу се уписују у базу података, под условом да је могуће креирати час под условима дефинисаним преко описаних прозора. Наиме, пре него што се нов час дода у базу, информације пролазе кроз процес верификације који обухвата провере да ли је за унето време и датум учионица, запослени или одељење заузето. Уколико јесте, нови час се неће креирати и кориснику ће се пријавити информација да је запослени (или учионица или одељење) у то време заузет.

За сваки час се, уколико је прошао све валидације, при додавању у базу прави посебан ред у оквиру табеле *ZauzetostUcionice* која, као што је описано у поглављу 3.4 *Опис ентитета и асоцијација у моделима*, садржи све информације о заузетости учионице укључујући и одељење. Помоћу поменуте табеле олакшана је провера информација о заузетости учионица и одељења, јер приликом провера приступамо само једној табели која садржи све неопходне информације, док се на основу повезаности часа (који садржи и информацију о запосленом који је евидентирао час) и заузетости такође проверава и заузетост запосленог који прави нови час.

Уобичајан начин бележења података о часовима који се држе у школама подразумева прављење распореда часова у којем се бележе информације о заузетости учионице и одељења у оквиру једног дана и који важи целе школске године. Планирање распореда часова и заузетости учионица на овај начин креира проблем у евентуалним случајевима када учионица није доступна, односно када се не може користити не зато што је заузета, већ јер се десила нека хаварија те је одржавање часова у њој постало немогуће, што се заобилази уколико се заузетости у базу уносе на горе описан начин.

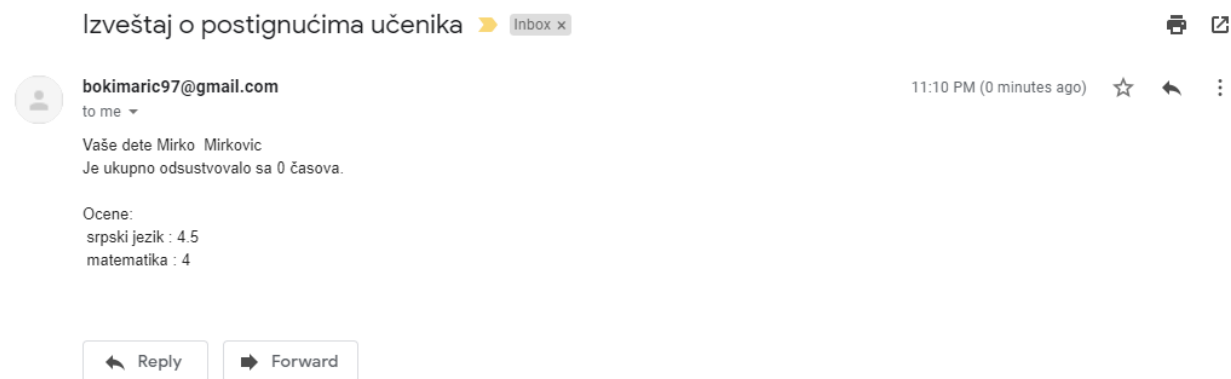
#### 4.4.1 Слање информација о ученику

Тренутни начин информисања родитеља о постигнутом успеху њиховог детета своди се на две опције, а то су родитељски састанак или телефонски позив. Ова два начина обухватају проналазак контакт информација родитеља и прикупљање информација о ученику, при чему ти подаци често нису забележени на једном месту. Апликативно решење информационог система који је предмет овог рада пружа могућност аутоматског генерисања извештаја о тренутно постигнутом успеху сваког ученика понаособ и слање тог извештаја путем е-мејла његовом родитељу. Овим се заобилази дуготрајан процес прикупљања информација о ученицима као и организовање позива или састанака са њиховим родитељима. Ова могућност нуди се свим учитељима и оним наставницима који су разредне старешине неком одељењу.



Слика 4.8 – Дугме за слање извештаја о ученицима

На слици 4.8 приказано је дугме у оквиру падајућег менија *Izveštaji/Raspored* где се кликом на њега покреће функционалност генерисања извештаја о ученицима и њихово слање на е-мејл адресе родитеља. При генерисању извештаја прикупљају се информације из две табеле: *Radi* и *Prisustvo*. На основу њих се рачуна просек оцена по предметима и број часова са којих је ученик одсуствовао, те се шаље е-мејл порука родитељу која има облик приказан на слици 4.9.



Слика 4.9 – Изглед извештаја о успеху ученика који се шаље родитељу

## 5. Закључак

Софтверски пакет за подршку рада запослених у школским установама, описан у овом дипломском раду олакшава рад запосленима у образовним установама, што је и његова примарна улога.

Увођењем информационог система корисницима је омогућено да потребне податке уносе и обрађују електронским путем на једноставнији начин, чиме се смањује време које се иначе троши на документацију где би се ти подаци чували. Смањујући време неопходно за манипулисање подацима записаним на бројним папирима, омогућава се запосленима у школским установама да се више посвете едукацији својих ученика, што је и њихов главни задатак. Аутоматским слањем извештаја о постигнутом успеху ученика избегава се потреба за организовањем родитељских састанака или телефонских позива, током којих би се информације о ученицима предочавале сваком родитељу појединачно. Смештањем релевантних информација у једну базу података информационог система олакшава се претрага датих података и омогућава ефикаснија организација рада.

Такође, увођењем информационог система у школске установе, ствара се могућност за статистичку анализу података, на основу које би се могао унапредити рад запослених са својим ученицима. Ово би представљало једно од могућих проширења система које би требало размотрити.

Уколико се узму у обзир предности које пружају веб апликације као потенцијално проширење описаног информационог система, могу се уочити додатне олакшице у раду запослених, јер би у том случају њима било омогућено да свој рад планирају са било ког места, не само са рачунара на којем би се информациони систем налазио. Посматрајући веб апликације као потенцијално унапређење и проширивањем базе података корисничким информацијама везаним за родитеље ученика, било би дозвољено да информације о ученицима буду доступне родитељима у било ком тренутку и са било које локације.

Следеће могуће проширење вредно разматрања, било би да запослени сами креирају своје корисничке налоге на систему, при чему би на администратора система спала одговорност валидације унетих података. На овај начин био би смањен мукотрпан посао креирања налога за сваког запосленог појединачно, који тренутно администратор мора обављати и истовремено би била смањена могућност грешке коју сваки људски фактор са собом уноси у систем.

Обзиром на могућности које су нам пакети искоришћени за потребе информационог система пружили, а знајући да постоје њихове новије верзије са додатним функционалностима требало би узети у обзир и могућност пребацивања постојећег апликативног решења на платформу која подржава рад са новијим верзијама поменутих пакета.

Раздвајањем функционалности на логичке целине, тј. имплементацијом сервисно оријентисане архитектуре, апликативном решењу дата је основа за даље унапређење на микросервисну архитектуру, која би погодовала креирању веб апликација. Микросервисна архитектура представља унапређење сервисно оријентисане архитектуре, где се апликација структурира као колекција лабаво повезаних сервиса [7]. Тренутно решење подразумева постојање једног сервиса у ком је садржана целокупна пословна логика апликације. У микросервисној архитектури, прво би он био издељен на сервисе који би били одговорни за посебне, логички повезане целине, нпр. сервис за рад са корисничким информацијама, сервис за рад са часовима, контролним тачкама и сл. На исти начин могла би се поделити и база и кориснички интерфејс.



## Речник појмова

**Code First** – приступ моделовању базе података, где се прво класе конфигуришу и на основу њих генерише база

**Model First** – приступ моделовању базе података, где се прво помоћу неког алата конфигурише дијаграм базе података и онда, у складу са њим, сама база података

**Метаподаци** – „Подаци о подацима“; подаци који описују карактеристике неких других података

**Миграција** – у овом контексту, начин на који пакет Entity Framework базу података синхронизује са текућим моделом базе података, при коришћењу Code First приступа моделовању базе података



## Скраћенице

**СУБП** – систем за управљање базама података

**WPF** – *Windows Presentation Foundation*, графички подсистем за развијање корисничког интерфејса у Windows базираним апликацијама

**MVVM** – *Model–View–ViewModel* је структурни програмски образац по којем се одваја графички кориснички интерфејс (View) од модела бизнис логике (Model), док се за размену информација између њих користи посебан модел (ViewModel)

**WCF** – *Windows Communication Foundation*, сервисно оријентисани модел размене порука који омогућава програмима размену порука преко рачунарске мреже или локално на асинхрон начин

**SSL** – *Secure Sockets Layer*, криптографски протокол за безбедну комуникацију преко мреже





## Литература

- [1] Документација Entity Framework пакета, .NetFramework – <https://docs.microsoft.com/en-us/ef/ef6/what-is-new/>
- [2] Презентација на предмету Базе података 1 на ПСИ – [http://www.acs.uns.ac.rs/sites/default/files/4\\_BP\\_ER\\_Model\\_Osnove\\_1.pdf](http://www.acs.uns.ac.rs/sites/default/files/4_BP_ER_Model_Osnove_1.pdf)
- [3] "Service-oriented architecture explained." – Hashimi, Sayed, *ONDot Net.com*, August 18 (2003)
- [4] Документација WCF, .NetFramework – <https://docs.microsoft.com/en-us/dotnet/framework/wcf/whats-wcf>
- [5] Документација NuGet, .NetFramework – <https://docs.microsoft.com/en-us/nuget/what-is-nuget>
- [6] Објашњење .NetFramework-ове уграђене *hash* функције, StackOverflow – <https://stackoverflow.com/questions/20621950/asp-net-identitys-default-password-hasher-how-does-it-work-and-is-it-secure>
- [7] Микросервисна архитектура, Wikipedia – <https://sr.wikipedia.org/wiki/Mikroservisi>



## Биографија

Божидар Марић рођен је 12. децембра 1997. године у Липничком Шору. Завршио је основну школу „Свети Сава“ у Липничком Шору, а затим природни смер гимназије „Вук Караџић“ у Лозници. Школске 2016/2017. године уписује Факултет техничких наука у Новом Саду, на студијском програму Примењено софтверско инжењерство. Положио је све испите прописане планом и програмом.