

УНИВЕРЗИТЕТ У БЕОГРАДУ
ФАКУЛТЕТ ОРГАНИЗАЦИОНИХ НАУКА

ЗАВРШНИ РАД

Развој софтверског система за праћење
пословања рада спа центра употребом .NET
технологија

Ментор:

проф. др Саша Лазаревић

Студент:

Тијана Вукомановић 82/2015

Београд, 2023. године

Попис слика

Слика 1 - Архитектура .NET-а	2
Слика 2 - Код у C#.....	3
Слика 3 - Први део - код у VB.NET	3
Слика 4 - Други део - код у VB.NET-у	4
Слика 5 - Код у CIL	4
Слика 6- Компоненте BCL.....	6
Слика 7 - C# и остали програмски језици	9
Слика 8 - Креирање ПДФ документа.....	11
Слика 9 - Табела и ћелије ПДФ документа	12
Слика 10 - Додавање слике у ПДФ документ	12
Слика 11 - Манипулисање постојећег документа	12
Слика 12 - Креирање нити	13
Слика 13 - Креирање нове нити	13
Слика 14 - Креирање нове нити - објашњење	14
Слика 15 - Животни циклус нити	14
Слика 16- Сокети, протоколи и портови	15
Слика 17 - Клијент - сервер конекција	17
Слика 18 - MVC патерн.....	19
Слика 19 - Дијаграм случајева коришћења – корисник	21
Слика 20 - Дијаграм случајева коришћења - админ	22
Слика 21 - DC Унос новог клијента ОС	34
Слика 22 - DC Уноса новог клијента AC 1	34
Слика 23 - DC Претраживање клијента ОС	36
Слика 24 - DC Претраживање клијента AC 1	36
Слика 25 - DC Претраживање клијента AC 2	37
Слика 26 - DC Измена података о клијенту ОС.....	38
Слика 27 - DC Измена података о клијенту AC1	38
Слика 28 - DC Измена података о клијенту AC2	39
Слика 29 - DC Измена података о клијенту AC3	39
Слика 30 - DC Измена података о клијенту AC4	40
Слика 31 - DC Унос нове услуге ОС.....	41
Слика 32 - DC Унос нове услуге AC1	41
Слика 33 - DC Претраживање услуге ОС	43
Слика 34 - DC Претраживање услуге AC1	43
Слика 35 - DC Измена података о услуги ОС.....	45
Слика 36 - DC Измена података о услуги AC1	45
Слика 37 - DC Измена података о услуги AC2	46
Слика 38 - DC Измена података о услуги AC3	46
Слика 39 - DC Измена података о услуги AC4	47
Слика 40- DC Креирај рачун ОС.....	48
Слика 41 - DC Креирај рачун AC1	49
Слика 42 - DC Креирај рачун AC2	49
Слика 43 - DC Креирај рачун AC3	50

Слика 44 - ДС Сторнирај рачун ОС	51
Слика 45 - ДС Сторнирај рачун АС1.....	51
Слика 46 - ДС Сторнирај рачун АС2.....	52
Слика 47 - ДС Сторнирај рачун АС3.....	52
Слика 48 - ДС Сторнирај рачун АС4.....	53
Слика 49 - ДС Унос новог запосленог ОС.....	54
Слика 50 - ДС Унос новог запосленог АС1	54
Слика 51 - ДС Деактивирање запосленог ОС.....	55
Слика 52 - ДС Деактивирање запосленог АС1	55
Слика 53 - ДС Деактивирање запосленог АС2	56
Слика 54 - ДС Деактивирање запосленог АС3	56
Слика 55 - ДС Деактивирање запосленог АС4	57
Слика 56 - ДС Додавање улоге запосленом ОС.....	58
Слика 57 - ДС Додавање улоге запосленом АС1	58
Слика 58 - ДС Додавање улоге запосленом АС2	59
Слика 59 - ДС Додавање улоге запосленом АС3	59
Слика 60 - ДС Додавање улоге запосленом АС4	60
Слика 61 - Концептуални дијаграм класа.....	67
Слика 62 - КМ структура система.....	74
Слика 63 - КМ Понашање система	75
Слика 64- Тронивојска архитектура.....	76
Слика 65 - Структура корисничког интерфејса.....	77
Слика 66 - Форма за унос новог клијента	77
Слика 67 - Унос новог клијента.....	78
Слика 68 - Клијент је сачуван	78
Слика 69 - Клијент није сачуван	78
Слика 70 - Форма за претрагу клијената.....	79
Слика 71 - Претрага клијената	80
Слика 72 - Претрага је завршена	80
Слика 73 - Избор клијента.....	81
Слика 74 - Приказ података о клијенту	81
Слика 75 - Недовољно карактера за претрагу	82
Слика 76 - Клијенти нису пронађени	82
Слика 77 - Клијент није учитан	82
Слика 78 - Форма за претрагу клијената.....	83
Слика 79 - Приказ клијената добијених претрагом	84
Слика 80 - Приказ података о изабраном клијенту	84
Слика 81 - Приказани подаци о изабраном клијенту.....	85
Слика 82 - Измена клијента.....	85
Слика 83 - Подаци о клијенту су изменењени.....	86
Слика 84 - Неодвољан унос карактера за претрагу.....	86
Слика 85 - Клијенти нису пронађени	86
Слика 86 - Клијент није учитан	86
Слика 87 - Клијент није изменењен.....	87

Слика 88 - Форма за унос нове услуге.....	88
Слика 89 - Унос нове услуге.....	89
Слика 90 - Услуга је сачувана.....	89
Слика 91 - Услуга није сачувана.....	89
Слика 92 - Форма за претрагу услуга	90
Слика 93 - Претраживање услуга	91
Слика 94 - Претрага је завршена	91
Слика 95 - Приказ улига добијених претрагом	91
Слика 96 - Избор услуге	92
Слика 97 - Приказ података о услуги	92
Слика 98 - Услуга није пронађена	93
Слика 99 - Услуга не може да се учита.....	93
Слика 100 - Форма за претрагу услуга.....	94
Слика 101 - Претрага услуга	95
Слика 102 - Претрага је завршена.....	95
Слика 103 - Приказ услуга добијених претрагом	95
Слика 104 - Избор услуге	96
Слика 105 - Приказ података о изабраној услуги.....	96
Слика 106 - Измена услуге	97
Слика 107 - Услуга је изменљена	97
Слика 108 - Недовољан број карактера за претрагу.....	97
Слика 109 - Услуга није пронађена	98
Слика 110 - Услуга није учитана.....	98
Слика 111 - Услуга није изменљена.....	98
Слика 112 - Форма за унос рачуна	99
Слика 113 - Додавање клијента на рачун	100
Слика 114 - Претрага клијената.....	100
Слика 115 - Претрага је завршена.....	101
Слика 116 - Приказ клијената добијених претрагом	101
Слика 117 - Приказ клијента на рачуну.....	102
Слика 118 - Додавање услуга.....	102
Слика 119 - Унос нове услуге на рачун.....	103
Слика 120 - Приказ услуге која је додата на рачун.....	103
Слика 121 - Унос новог рачуна.....	104
Слика 122 - Рачун је сачуван.....	104
Слика 123 - Клијенти нису пронађени	104
Слика 124 - Недовољно карактера за претрагу.....	105
Слика 125 - Рачун није сачуван	105
Слика 126 - Форма за претрагу рачуна	106
Слика 127 - Претрага рачуна.....	107
Слика 128 - Претрага је завршена.....	107
Слика 129 - Учитавање података о рачуну.....	108
Слика 130 - Форма са подацима о рачуну	108
Слика 131 - Сторнирање рачуна	109

Слика 132 - Рачун је сторниран.....	109
Слика 133 - Рачуни нису пронађени.....	110
Слика 134 - Недовољан број карактера за претрагу.....	110
Слика 135 - Рачун није учитан	110
Слика 136 - Рачун није сторниран	110
Слика 137 - Форма за унос новог запосленог	111
Слика 138 - Унос новог запосленог	112
Слика 139 - Запослени је сачуван.....	112
Слика 140 - Запослени није сачуван.....	112
Слика 141 - Форма за претрагу запосленог	113
Слика 142 - Претрага запосленог	114
Слика 143 - Претрага је успешно извршена	114
Слика 144 - Приказ за запослених добијених претрагом.....	115
Слика 145 - Учитавање података о запосленом	115
Слика 146 - Приказ података о изабраном запосленом.....	116
Слика 147 - Деактивирање запосленог	116
Слика 148 - Запослени је деактивиран	117
Слика 149 - запослени нису пронађени	117
Слика 150 - Недовољан број карактера за претрагу.....	117
Слика 151 - Запослени не може да се учита	118
Слика 152 - Запослени је деактивиран	118
Слика 153 - Форма за претрагу запослених.....	119
Слика 154 - Претрага запослених.....	120
Слика 155 - Запослени су пронађени	120
Слика 156 - Приказ запослених.....	120
Слика 157 - Приказ изабраног запосленог.....	121
Слика 158 - Додавање нове улоге запосленом.....	121
Слика 159 - Форма за додавање нове улоге.....	122
Слика 160 - Унос нове улоге.....	122
Слика 161 - Нова улога је сачувана.....	123
Слика 162 - Запослени нису пронађени.....	123
Слика 163 - Недовољан број карактера за претрагу.....	123
Слика 164 - Запослени није учитан	124
Слика 165 - Нова улога није сачувана.....	124
Слика 166 - Дијаграм УГ1.....	126
Слика 167 - Дијаграм УГ2.....	127
Слика 168 - Дијаграм УГ3.....	127
Слика 169 - Дијаграм УГ4.....	128
Слика 170 - Дијаграм УГ5.....	128
Слика 171 - Дијаграм УГ6.....	129
Слика 172-Дијаграм УГ7.....	129
Слика 173 - Дијаграм УГ8.....	130
Слика 174 - Дијаграм УГ9.....	130
Слика 175 - Дијаграм УГ10.....	131

Слика 176- Дијаграм УГ12	131
Слика 177 - Дијаграм УГ13	132
Слика 178 - Дијаграм УГ14	132
Слика 179 - Дијаграм УГ15	133
Слика 180 - Дијаграм УГ16	133
Слика 181 - Дијаграм УГ17	134
Слика 182 - Дијаграм УГ18	134
Слика 183 - Дијаграм УГ19	135
Слика 184 - Дијаграм УГ20	135
Слика 185 - Дијаграм УГ21	136
Слика 186 - Дијаграм УГ22	136
Слика 187 - Дијаграм УГ11	137
Слика 188 - Класе које су одговорне за СО наслеђују класу ОпштаСО	138
Слика 189 - код Опште доменске класе	139
Слика 190 - Брокер класа се повезује са класом ОпштиДоменскиОбјекат	141
Слика 191 - Архитектура софтверског система.....	142
Слика 192 - Пројекти из којих се састоји пројекат Спа центар	143
Слика 193 - Доменске класе	144
Слика 194 – Пројекат Клијент	145
Слика 195 - Пројекат Сесија.....	145
Слика 196 - Пројекта Сервер	146
Слика 197 - Пројекат Системске операције	146
Слика 198- Пројекат за креирање ПДФ докумената.....	147
Слика 199 - скрипта за креирање табела.....	148
Слика 200 - Табела Категорије	148
Слика 201 - Табела Клијент	149
Слика 202 - Табела Запослени	149
Слика 203 - Табела Запослени - Улога	150
Слика 204 - Табела Лојалти	150
Слика 205 - Табела Начини плаћања	151
Слика 206 - Табела Заглавље рачуна	151
Слика 207 - Табела Линије рачуна	152
Слика 208 - Табела Улоге	152
Слика 209 - Табела Услуге	153
Слика 210 - Код СО Креирај клијента	154
Слика 211 - Код СО Врати лојалти типове	154
Слика 212 - Код СО Нађи клијенте	155
Слика 213 - Код СО имплементација различитих типова претраге клијента	155
Слика 214 - Код СО Учитај клијента	156
Слика 215 - Код СО Измени клијента	156
Слика 216 - Код СО Врати категорије	157
Слика 217 - Код СО Креирај услугу	157
Слика 218 - Код СО Нађи услуге	158
Слика 219 - Имплементација могућности претраге услуга по различитим критеријумима.	158

Слика 220 - Код СО Учитај услугу.....	159
Слика 221 - Код СО Измени услугу	159
Слика 222 - Код СО Врати услуге	160
Слика 223 - Код СО Врати начине плаћања	160
Слика 224 - Код СО Креирај рачун	161
Слика 225 - Код СО Нађи рачун	162
Слика 226 - Код СО Учитај рачун	162
Слика 227 - Код СО Сторнирај рачун	163
Слика 228 - Код СО Нађи запослене.....	163
Слика 229 - Код СО Учитај запосленог	164
Слика 230 - Код СО Креирај запосленог.....	164
Слика 231 - Код СО Деактивирај запосленог	165
Слика 232 - Код СО Врати улоге	165
Слика 233 - Код СО Креирај везу улога - запослен.....	166

Попис таблица

Таблица 1 – Ограничења Запослен.....	68
Таблица 2 – Ограничења Улога	68
Таблица 3 – Ограничења ЗапосленУлога	69
Таблица 4 – Ограничења Клијент	69
Таблица 5 – Ограничења Лојалти.....	70
Таблица 6 – Ограничења Услуга	70
Таблица 7- Ограничења Категорија.....	71
Таблица 8 - Ограничења Рачун.....	72
Таблица 9 - Ограничења Ставка рачуна	72
Таблица 10 - Ограничења Начин Плаћања.....	73

Садржај

Попис слика.....	2
Попис таблица	8
1. Увод.....	1
1.1. .NET технологија	1
1.1.1.1. Common Intermediate language (<i>CIL</i>).....	2
1.1.1.2. Base class library	5
1.1.1.3. Common Language Runtime (<i>CLR</i>).....	6
1.2. Програмски језик <i>C#</i>	7
1.2.1.1. Историја <i>C#</i>	7
1.2.1.2. Карактеристике <i>C#</i>	8
1.2.1.3. Сличности и разлике <i>C#</i> и осталих језика.....	8
1.2.1.4. Особине <i>C#</i>	10
1.3. <i>iText7</i> библиотека.....	11
1.4. Нити	13
1.5. Сокети	15
1.5.1.1. Животни циклус TCP сокета	16
1.6. <i>Model view controller (MVC)</i> патерн.....	18
2. Прикупљање захтева.....	20
2.1. Вербални опис модела.....	20
2.2. Случајеви коришћења.....	21
СК1: Случај коришћења – Унос новог клијента	23
СК2: Случај коришћења – Претраживање клијента	24
СК3: Случај коришћења – Измена података о клијенту	25
СК4: Случај коришћења – Унос нове услуге	26
СК5: Случај коришћења – Претраживање услуге.....	27
СК6: Случај коришћења – Измена података о услуги	28
СК7: Случај коришћења – Креирање рачуна	29
СК8: Случај коришћења – Сторнирање рачуна	30
СК9: Случај коришћења – Унос новог запосленог	31
СК10: Случај коришћења – Деактивирање запосленог	32
СК11: Случај коришћења – Додавање улоге запосленом	33
3. Анализа	34

3.1.	Понашање софтверског система – системски дијаграм секвенци.....	34
	ДС1: Унос новог клијента.....	34
	ДС2: Претраживање клијента.....	36
	ДС3: Измена података о клијенту.....	38
	ДС4: Унос нове услуге	41
	ДС5: Претраживање услуге.....	43
	ДС6: Измена података о услуги	45
	ДС7: Креирање рачуна.....	48
	ДС8: Сторнирање рачуна.....	51
	ДС9: Унос новог запосленог.....	54
	ДС10: Деактивирање запосленог.....	55
	ДС11: Додавање улоге запосленом.....	58
3.2.	Понашање софтверског система.....	62
	Дефинисање уговора о системским операцијама.....	62
3.3.	Структура софтверског система.....	67
	Концептуални модел.....	67
	Резултат анализе.....	74
4.	Проектовање	76
4.1.	Проектовање корисничког интерфејса – пројектовање екранских форми	77
	СК1: Случај коришћења – Унос новог клијента	77
	СК2: Случај коришћења – Претраживање клијента	79
	СК3: Случај коришћења – Измена података о клијенту	83
	СК4: Случај коришћења – Унос нове услуге	88
	СК5: Случај коришћења – Претраживање услуге.....	90
	СК6: Случај коришћења – Измена података о услуги	94
	СК7: Случај коришћења – Креирање рачуна	99
	СК8: Случај коришћења – Сторнирање рачуна	106
	СК9: Случај коришћења – Унос новог запосленог	111
	СК10: Случај коришћења – Деактивирање запосленог	113
	СК11: Случај коришћења – Додавање улоге запосленом	119
4.2.	Пројектовање апликационе логике	125
4.2.1.1.	Комуникација са клијентима.....	125
4.2.1.2.	Контролер апликационе логике.....	125

4.3.	Пословна логика	126
4.3.1.1.	Пројектовање понашања софтверског система – системске операције	126
4.3.1.2.	Пројектовање структуре софтверског система – Доменске класе.....	139
4.4.	Пројектовање брокера базе података.....	140
4.5.	Релациони модел.....	141
4.6.	Архитектура софтверског система	142
5.	Имплементација	143
5.1.	Имплементација пројеката из којих се састоји решење <i>SpaCenter</i>	143
5.1.1.1.	Класе SpaIWellness – Domain.....	143
5.1.1.2.	Класе SpaIWellness – Client	144
5.1.1.3.	Класе SpaIWellness – Session.....	145
5.1.1.4.	Класе SpaIWellness – Server.....	146
5.1.1.5.	Класе SpaIWellness – SystemOperations.....	146
5.1.1.6.	Класе SpaIWellness – GeneratingPDF.....	147
5.2.	Имплементација складишта података	148
5.3.	Имплементација системских операција	154
5.3.1.1.	CO <i>KreirajKlijenta</i>	154
5.3.1.2.	CO <i>VratiLoyaltyTipove</i>	154
5.3.1.3.	CO <i>NadjiKlijente</i>	155
5.3.1.4.	CO <i>UcitajKlijenta</i>	156
5.3.1.5.	CO <i>IzmeniKlijenta</i>	156
5.3.1.6.	CO <i>VratiKategorije</i>	157
5.3.1.7.	CO <i>KreirajUslugu</i>	157
5.3.1.8.	CO <i>NadjiUsluge</i>	158
5.3.1.9.	CO <i>UcitajUslugu</i>	159
5.3.1.10.	CO <i>IzmeniUslugu</i>	159
5.3.1.11.	CO <i>VratiUsluge</i>	160
5.3.1.12.	CO <i>VratiNacinePlacanja</i>	160
5.3.1.13.	CO <i>KreirajRacun</i>	161
5.3.1.14.	CO <i>NadjiRacun</i>	162
5.3.1.15.	CO <i>UcitajRacun</i>	162
5.3.1.16.	CO <i>StornirajRacun</i>	163
5.3.1.17.	CO <i>NadjiZaposlene</i>	163

5.3.1.18.	<i>CO UcitajZaposlenog</i>	164
5.3.1.19.	<i>CO KreirajZaposlenog</i>	164
5.3.1.20.	<i>CO DeaktivirajZaposlenog</i>	165
5.3.1.21.	<i>CO VratiUloge</i>	165
5.3.1.22.	<i>CO KreirajVezuUZ</i>	166
5.4.	Имплементација екранских форми	167
5.4.1.1.	Главна екранска форма.....	167
5.4.1.2.	Форма Претрага услуга.....	171
6.	Тестирање	173
7.	Закључак	174
8.	Литература	175

1. Увод

1.1. .NET технологија

Развој .NET технологије започео је крајем 1990-их година, када је Microsoft почeo радити на новој технологије под називом “*Next Generation Windows Services*” (NGWS). Циљ овог пројекта је био развити заједничко окружење за израду, дистрибуцију и извођење апликација на *Windows* платформи. Приликом развоја апликација на *Windows* платформи јављали се бројни изазови као што су: сигурност, једноставност употребе, интероперабилност... Да би одговорио на дате изазове и да би остао конкурентан на тржишту Microsoft започиње развој **.NET Framework-a**. На конференцији 2000. године је објављено јавности да се ради на изградњи таквог решења. Прва бета верзија објављена је крајем 2001. године, а званична верзија .NET Framework-a објављена је 13. фебруара 2002. године.

“Microsoft .NET Framework је софтверска платформа која може бити инсталација на рачунарима које покреће Microsoft Windows оперативни систем. Он укључује велики број готових библиотека кодова за уобичајене проблеме у програмирању.”¹

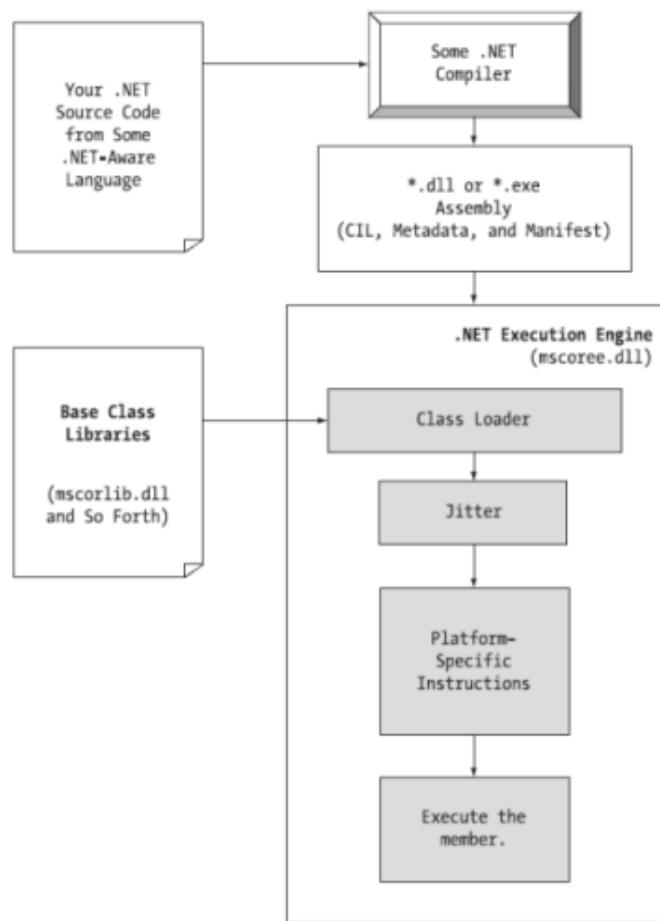
Са напредовањем технологије било је потребно одговорити на нове захтеве тржишта. Како су перформансе система који су користили .NET Framework напредовале, није више било потребно радити све провере које је .NET Framework радио у својим унутрашњим слојевима. Microsoft 2014. године представља .NET CORE као наследника .NET Framework-a. .NET CORE је нови софтверски framework који је бесплатан, отвореног-кода, независан од оперативног система. Може се користити на *Windows*, *Linux* и *macOS* оперативним системима. Последња активна верзија .NET Framework-a је 4.8, зато .NET CORE задржава своје име до своје 3.1 верзије. Како се не би правила конфузија, одлучује се да .NET CORE прескочи верзију 4, пошто постоји иста верзија и .NET Framework-a, и након 3. верзије имплементира се 5. верзија .NET CORE-a. Како не постоји и 5. верзија .NET Framework-a, .NET CORE добија ново име, .NET 5..NET Framework и даље постоји, дугорочно је подржан од стране Microsoft-a, али све нове функционалности имплементирају се на .NET CORE-у односно .NET-у. Постоји неколико карактеристике које чине .NET. Једна од њих је **интероперабилност** која омогућава програмерима да пишу код у различитим програмским језицима а да притом могу да користе исте библиотеке и компоненте. Све што се може написати у једном програмском језику може се написати у било ком програмском језику који подржава .NET технологија. Исто тако кодови могу бити написани са истим перформансама и истом ефикасношћу. Како би се омогућила **сигурност** као следећа карактеристика, постоје уgraђени механизми попут CLR(Common Language Runtime) који проверава исправност кода пре његовог извршавања и тако штити од различитих сигурносних претњи. .NET омогућава програмерима да развију **скалабилне** апликације које се могу прилагодити различитим оптерећенима и броју корисника. Карактеристика која је довела до популарности .NET је и **једноставност**

¹ Преузето 26. марта 2023. са https://sr.wikipedia.org/sr-ec/.NET_Framework

употребе. Постоје многе уgraђене библиотеке и функције које олакшавају развој апликација. Такође, постоје бројна развојна окружења, као што је *Visual Studio*, која помажу програмерима да брзо и једноставно развију апликације. Све ове карактеристике учиниле су .NET јако популарном, моћном и флексибилном технологијом за развој различитих врста апликација, од десктоп и web апликација до мобилних апликација...

1.1.1.1. Common Intermediate language (CIL)

Microsoft Intermediate language или касније назван *Common Intermediate language (CIL)* је један од најзначајнијих коцпата. Представља један унутрашњи слој .NET-а који омогућава да се програмски језици не компонирају директно у извршни код. На следећој слици се може видети архитектура:



Слика 1 - Архитектура .NET-a²

² Слика је преузета из pdf.књиге: Karnataka state open university, department of studies in information technologies, (2014) MSIT-120 Dot Net Technologies

Како што се види на слици, компајлер било ког језика који подржава .NET покреће *CIL*. Код било ког програмског језика ће бити преведен у скуп инструкција које су погодне за даље извршење. У наставку ће бити приказани примери како се код у *C#* и *VB* конвертује у исти скуп инструкција у *CIL*.

У наставку су слике где је приказан код у *C#* и *VB.NET*-у:

```
// Calc.cs
using System;
namespace Calculate or Example
{
    // This class contains the app's entry point.
    public class CalcApp
    {
        static void Main()
        {
            Calc c = new Calc();
            int ans = c.Add(10, 84);
            Console.WriteLine("10 + 84 is {0}.", ans);
            // Wait for user to press the Enter key before shutting down.
            Console.ReadLine();
        }
    }
    // The C# calculator.
    public class Calc
    {
        public int Add(int x, int y)
        {
            return x + y;
        }
    }
}
```

Слика 2 - Код у *C#*

```
' Calc.vb
Imports System
Namespace CalculatorExample
    ' A VB .NET 'Module' is a class that only contains
    ' static members.
    Module CalcApp
        Sub Main()
            Dim ans As Integer
            Dim c As New Calc
            ans = c.Add(10, 84)
            Console.WriteLine("10 + 84 is {0}.", ans)
            Console.ReadLine()
    End Sub
End Module
```

Слика 3 - Први део - код у *VB.NET*

```

    End Sub
End Module
Class Calc
    Public Function Add(ByVal x As Integer, ByVal y As Integer) As Integer
        Return x + y
    End Function
End Class
End Namespace

```

Слика 4 - Други део - код у VB.NET-у

Add() метода представљена у *CIL*:

```

.method public hidebysig instance int32 Add(int32 x, int32 y) cil managed
{
    // Code size 8 (0x8)
    .maxstack 2
    .locals init ([0] int32 CS$1$0000)
    IL_0000: ldarg.1
    IL_0001: ldarg.2
    IL_0002: add
    IL_0003: stloc.0
    IL_0004: br.s      IL_0006
    IL_0006: ldloc.0
    IL_0007: ret
} // end of method Calc::Add

```

Слика 5 - Код у CIL³

Превођење кода у *CIL*, а не одмах у извршни код омогућава језичку интеграцију, као што је приказано на сликама изнад. Сваки компајлер поризводи готово идентичне *CIL* инструкције, тако да сви језици могу да комуницирају. Могуће је да *C#* пројекат користи неку библиотеку од било ког другог језика који подржава *.NET*.

CIL је сличан *Java Byte* коду. *Java* програм се компајлира у *Java Byte* код помоћу *Java* компајлера.

³ Сви наведени примери су преузето из пдф.књиге: Karnataka state open university, department of studies in information technologies, (2014) MSIT-120 Dot Net Technologies

1.1.1.2. *Base class library*

".NET садржи стотине класа, интерфејса, структура, итд. које се користе у тренутку када се покреће *CLR*, пружају низ корисних услуга које помажу програмерима да повећају своју продуктивност. Сви ови типови се заједнички називају *Framework class library*. Ова билиотека класа се може грубо поделити у четири групе: Библиотека основних класа(*Base class library (BCL)*), *ADO.NET* и *XML*, *Windows Forms* и *ASP.NET*."⁴

Base class library у .NET представља основни скуп класа и метода који пружа много корисних функционалности које су неопходне за развој .NET апликација. *BCL* је доступна у свим програмским језицима које .NET подржава. Као таква *BCL* није специјализована за један програмски језик већ је језички неутрална. Сваки језик који покреће *CLR* користи исте библиотеке. Приликом коришћења програмског језика *C#* уколико програмер жели да користи *System* компоненту користиће "using System". Програмер у *VB.NET*-у користиће "Imports System".

BCL се састоји од неколико кључних компоненти:

- *System* – пружа основне класе које се користе у свим .NET апликацијама, као што су: *Object, String, Array...*
- *System.IO* – пружа подршку у раду са датотекама.
- *System.Collections* – омогућава рад са колекцијама.
- *System.Threading* – пружа подршку при креирању брзих и ефикасних апликација са више нити.
- *System.Net*-омогућава да апликације комууницирају путем мреже.

Овде су побројане само неке од основних компоненти, листа компоненти је огромна, на следећој слици се могу видети још неке.

System.Timers	Provides the ability to raise events or take an action within a given timer period.
System.Transactions	Contains methods for the management of transactions
System.Web	Namespace for ASP.NET capabilities such as Web Services and browser communication.
System.Windows.Forms	Namespace containing the interface into the Windows API for the creation of Windows Forms programs.
System.Xml	Provides the methods for reading, writing, searching and changing XML documents and entities.

⁴ Pradeep Tapadiya, (2002) .NET programming a practical guide using C#, стр.15, 16

System.Linq	Interface to LINQ providers and the execution of LINQ queries
System.Linq.Expressions	Namespace which contains delegates and lambda expressions
System.Management	Provides access to system information such as CPU utilization, storage space, etc.
System.Media	Contains methods to play sounds
System.Messaging	Used when message queues are required within an application, superseded by WCF
System.Net	Provides access to network protocols such as SSL, HTTP, SMTP and FTP
System.Reflection	Ability to read, create and invoke class information.
System.Resources	Used when localizing a program in relation to language support on web or form controls
System.Runtime	Contains functionality which allows the management of runtime behavior.
System.Security	Provides hashing and the ability to create custom security systems using policies and permissions.
System.ServiceProcess	Used when a windows service is required
System.Text	Provides the StringBuilder class, plus regular expression capabilities

Слика 6- Компоненте BCL

Све ово олакшава развој апликација јер се многе ствари могу решити помоћу већ постојећих класа и метода.

1.1.1.3. Common Language Runtime (CLR)

Common language runtime је средњи део архитектуре .NET-а који се користи за извршење .NET апликација. *CLR* омогућава да се код написан на било ком програмском језику који подржава .NET изврши на било ком рачунару који има инсталiran .NET.

“*Common Intermediate language (CIL)* код и ресурси, попут битмапа и стрингова, чувају се у склопу (*assembly*) – извршном фајлу који обично има екstenзије *.exe* или *.dll*. *Assembly* садржи и манифест који пружа информације о типу, врсти, верзији и безбедносним захтевима. Када се изврши *C#* програм, *assembly* се учита у *CLR*. Ако су испуњени безбедносни захтеви из манифesta, *CLR* врши компајлирање “тачно-на-време” (*JIT – Just-In-Time*) преводећи *CIL* код у машинске инструкције.⁵ *JIT* компајлира код сваки пут пре употребе како би *CIL* инструкције које су платформски независне превео у одговарајће *CPU* инструкције које одговарају одређеном рачунару на ком се покреће програм. *JIT* омогућава да програмери могу да напишу један програм који ће се ефикасно компајлирати и извршити на машинама са различитим архитектурама.

Поред компајлирања, *CLR* поседује и друге сервисе. Ову су пар најважнијих:

- Аутоматско скупљање података (*automatic garbage collection*):

⁵ Преузето 28.3.2023. са <https://www.radlovacki.com/net-framework-arhitektura/>

омогућава програмерима да не морају да брину о расподели меморије, јер овај механизам аутоматски ослобађа сву неискоришћену меморију. Ово је јако корисно јер се понекад деси да програмери забораве да ослободе претходно коришћену меморију. „.NET се понекад назива и управљив код. Зове се управљив првенствено зато што користи скупљач смећа за управљање меморијом и зато што намеће безбедност типа и меморије.”⁶ Машински компајлиран код се назива неуправљив код.

- Руковање изузецима (*exception handling*):

овај механизам успешно обрађује грешке из различитих програмских језика које .NET подржава. Користи структуриран и контролисан приступ обраде грешака и неочекиваног понашања током извршавања програма. Овај процес обухвата откривање и реаговање на изузетне ситуације које могу настати током извршавања програма, као што су *null* референце, дељење нулом или индекс изван опсега.

- Управљање ресурсима (*resource management*):

односи се на процес управљања ресурсима као што су фајлови, базе података, мрежни ресурси и други објекти који захтевају затварање и ослобађање меморијског простора након коришћења. Овај механизам се ослања на механизам скупљања података (*garbage collection*).

1.2. Програмски језик C#

1.2.1.1. Историја C#

Програмски језик C# је први пут представљен јавности на Конференцији професионалних програмера 2000. када је представљен и .NET Framework. Као такав представљаје саставни део новог Microsoftovog развојног окружења. Идејно је био замишљен као програмски језик C или објектно орјентисан. Одатле је добио и назив C#, где # представља музичку нотацију да сеnota изводи за пола корака више, што у овом случају значи да је C# виши од C. Андрејс Хејсберг је био на челу тима који се бавио завојем овог програмског језика.

C# је развијан у протеклих 20 година са циљем да се прилагоди новим технологијама и новим парадигмама програмирања. Са новим верзијама Microsoft настоји да обезбеди што више флексибилности и ефикасности. Са сваком новом верзијом добија нове функционалности и унапређења.

Најновија верзија C# 11 објављена је у новембру 2022. године. Новине које је C# 11 донела: *generic math* и функције које подржавају исти циљ, нумеричке функције, функције које олакшавају рад са типовима структура, функције као што су локални типови датотека.⁷

⁶ Преузето 28.2.2023 са <https://learn.microsoft.com/en-us/dotnet/core/introduction>

⁷ Преузето са : <https://learn.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-version-history>

1.2.1.2. Карактеристике C#

Представља виши програмски језик јер подржава више парадигми (објектно оријентисану, императивну, декларативну, генеричку). Језик је опште примене и намењен је за израду апликација за .NET платформу. Потпуно је базиран на принципима објекто-оријентисаног програмирања, па тако нуди многе функционалности као што су наслеђивање, полиморфизам, апстракција и енкапсулација. Сви елемененти унутар њега представљају објекат. Најчешће се користи за прављење Windows, Web апликација и Web сервиса. У наставку су побројане неке од карактеристика:

- **Једноставност** – не постоји предпроцесор, сви код је записан на линији, не захтева датотека заглавља
- **Модеран** програмски језик – подржава многе модерне карактеристике управљање меморијом, управљање грешкама, функционалности за откањивање грешака(*debugging*)
- **Доследно понашање** – сви типови се третирају као објекти и програмери могу лако проширити систем типова.
- У потпуности **објектно ојенитисан** програмски језик - нуди многе функционалности као што су наслеђивање, полиморфизам, апстракција и енкапсулација. Сви елемененти унутар њега представљају објекат.
- **Сигурност** – врши се провера границе низа, уколико се употреби неиницијализована променљива приказује се порука о грешци, сви објекти и низови се динамички иницијализују нулом...
- **Компабилан** је са другим програмским језицима – користи *Common Language Specifications(CLS)* и тиме омогућава интеракцију са другим .NET језицима.

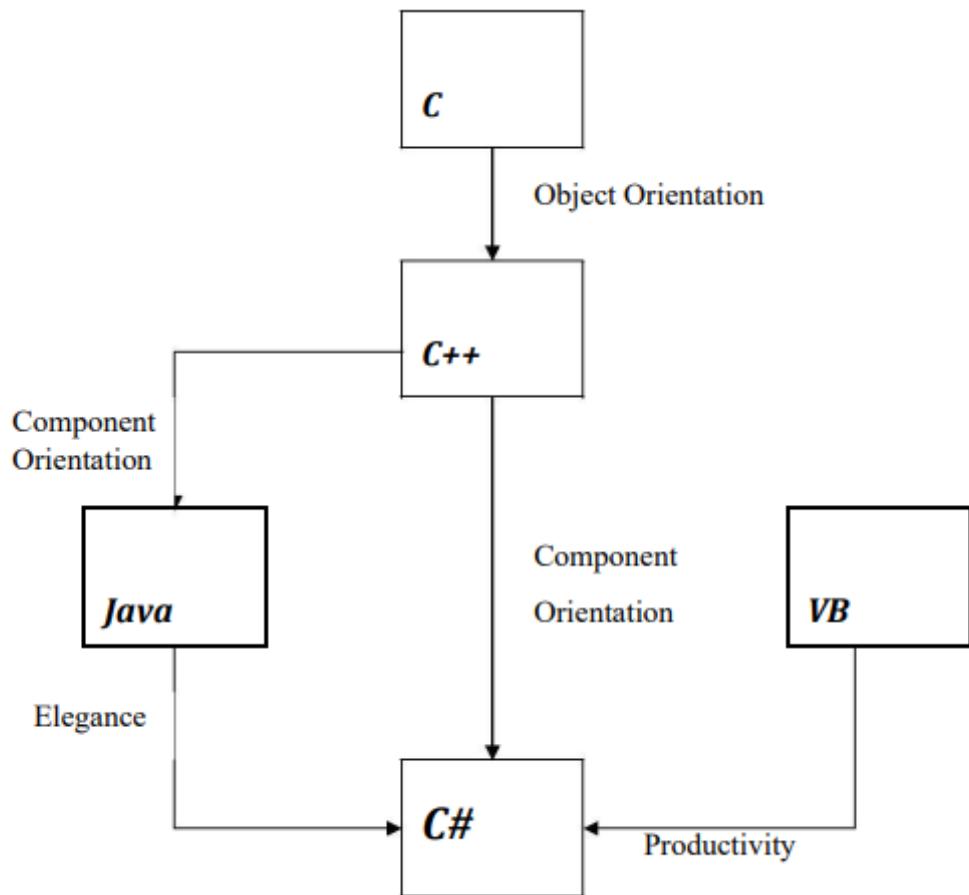
1.2.1.3. Сличности и разлике C# и осталих језика

Настало је као одговор на недостатке постојећих програмских језика, комбинујући њихове добре стране. Представља наследника програмских језика C, C++, а у тренутку када је настало јако је подсећао и на Java.

“Због чињенице да је C# хибрид бројних језика, резултат је производ који је синтаксички чист – ако не и чистији од Java, једноставан је као VB и пружа само отприлике исто толико снаге и флексибилности као C++(без ружних делова).”⁸ C++ представља језик који је најсличнији C#. Једна од разлика је да се C# код компајлира прво у IML па у машински код, док код C++ код се компајлира одмах у машински. Ова разлика потиче из тога што је C# саставни део .NET платформе. Једна од већих промена је да C# коду нису потребне датотеке заглавља. Идентификатори се препознају из изворних датотека и читају из динамичких симбола библиотека. Пошто CRL има механизме за управљање меморијом, употреба

⁸ Преузето из пдф.књиге: Karnataka state open university, department of studies in information technologies, (2014) MSIT-120 Dot Net Technologies, 72.стр.

показивача у *C#* много је мање важна, јер сакупљач смећа (*garbage collection*) аутоматски ради провере. Показивачи се у *C#* могу користити уколико је код означен као небезбедан. Још једна разлика је у томе што су низови у *C#* проверени па није могуће писати после краја низа, за разлику од *C++*. Као наследник *C++* језика, *C#* омогућава лакше и једноставније програмирање. Са друге стране *Java* и *C#* се сматрају језицима новије генерације, па имају пуно сличности, првенствено јер оба потичу од *C++-a*. Код се прво компајлира у *IML* код *C#-a*, а у *Java bytecode* у *Javi*, па онда у машински код. У оба језика постоји механизам за управљање меморијом (*garbage collection*). *C#* садржи примитивне типове од *Java*. Поред енумерација (типови вредности ограничени на дефинисан скуп константних променљивих) које подржавају оба језика, подржава и структуре (кориснички дефинисани типови вредности) за разлику од *Java*. Након 2005. године *Java* и *C#* почињу да се развијају у различитим правцима, тако да је сличности све мање. Што се *Visual Basic-a* тиче, постоје сличности, свакако су језици који се користе на истој .NET платформи, али је *Visual Basic* мање заступљен у пословном свету од *C#*. Следећа слика приказује везу *C#* са другим програмским језицима.



Слика 7 - *C#* и остали програмски језици⁹

⁹ Слика је преузета из pdf књиге: Ms. Poonam Verma, Dot Net Technology Notes(BCA 602) BCA VI Unit I & Unit II

1.2.1.4. Особине C#

У наставку су побројане неке од особина:

- Примитивни типови података

Примитивни типови података представљају пресликање основних типова података које подржава *CLR*, тако да је тип *string* исто што и *System.String*. *System.Object* представља основну класу и из овог типа су изведени сви остали типови.

- Модификатори приступа

Сваком типу и члановима типа приступ се може контролисати помоћу модификатора. Модификатор не мора да се наведе и уколико се не наведе има *internal* модификатор приступа.

- Иницијализовање атрибута класа
- Конструктори типова
- Вредносни и референтни типови

Једноставни типови података (*int, long, double...*) представљају вредносне типове података који се креирају на стеку. Типови који се креирају на *heap*-у се називају референтним типовима података, такви типови су класе. Разлика између референтног и вредносног типа огледа се у следећем: када се променљива референтног типа додели другој, меморијска локација за основни објекат се дели. Док вредносне променљиве држе засебну копију објекта. На следећем примеру ће бити приказано како функционишу референтни типови:

```
//Project ReferenceType

public class Foo
{
    public int x;

    public static void Test(Foo a, Foo b) {

        a.x = 5;
        b.x = 10;
        b = a;
        Console.WriteLine(b.x); // исписаће "5"
        a.x = 20
        Console.WriteLine(b.x); // исписаће "20"
    }
};
```

- Делегати и догађаји

У *C#* делегати су референтни типови података који се могу користити за дефинисање функција или метода који се могу позвати касније у програму. То омогућава флексибилност

и омогућава да се програм прилагоди динамичким захтевима. Делегати су корисни за многе ствари, као што су асинхроно програмирање и друге ситуације у којима је потребно преносити функције као параметре или чувати их као променљиве. Догађаји су специјални типови делегата који омогућавају да се функције позивају аутоматски када се деси неки одређени догађај, као што је клик на дугме, промена текста у пољу или повезивање уређаја.

- Низови

Јако сличан појму низа у *C* и *C++* програмским језицима. Низ садржи елементе истог типа, елементима се може приступати преко индекса. Низ може поседовати и ранг уколико је у питању низ са више димензија.

- Параметри методе – *ref, in, out*

Сваки параметар може бити означен као улаз у методу, као излаз из методе или и улаз и излаз, па тако постоје модификатори *in, out* и *ref*.

- Улазни(*in*) параметар се преноси слично семантици преноса по вредности. Пример: *mojaMetoda(int x, String y)*, параметри се прослеђују на стек, односно *mojaMetoda* добија локалну копију параметара. Било која промена параметара унутар функције се не одражава у простору позиваоца, то значи да се прослеђени параметар не може модификовати методом.
- Параметар референтног(*ref*) типа, показује на меморију у *heap*-у и свака промена параметара унутар функције одражава се на простор позиваоца, што значи да се прослеђени параметра може променити методом.
- Излазни(*out*) параметар се користи како би се означило да се параметар мора модификовати у методи. Овај модификатор не захтева иницијализацију параметра пре него што се проследи методи, већ се иницијализује унутар методе.

1.3. *iText7* библиотека

iText7 је једноставна и ефикасна библиотека која која се успешно носи са изазовима данашњих токова дигиталних података. Користи се за рад са ПДФ документима у *Javi* и *.NET(C#)*. Омогућава генерисање, манипулисање и читање ПДФ докумената.

У следећим примерима ће бити приказана неке основне функционалности коришћења:

- **Креирање ПДФ датотеке:**

```
PdfWriter writer = new PdfWriter(path);
PdfDocument pdf = new PdfDocument(writer);
Document document = new Document(pdf);
```

Слика 8 - Креирање ПДФ документа

- **Додавање табеле у ПДФ датотеку**

Пошто се креира нови објекат класе *Table* могуће је дефинисати ћелије преко класе *Cell*. За сваку ћелији је могуће подесити различите функционалности. У овоме се огледа једноставност употребе *iText* библиотеке. На овај начин је могуће креирати и прилагодити сваку ћелију у табели, тако да структуру ПДФ документа чине ћелије табеле. Свакој ћелији се може додати текст, прилагодити изглед ћелије жељеним потребама, подесити фонт текста и разна друга својства. Следећа слика приказује креирање ћелије.

```
Table table = new Table(colwidth);

Cell cell111 = new Cell(1, 1)
    .SetFontSize(26)
    .SetTextAlignment(TextAlignment.LEFT)
    .SetBold()
    .SetBorder(Border.NO_BORDER)
    .Add(image)
    .Add(new Paragraph("SPA CENTER"));

table.AddCell(cell111);
```

Слика 9 - Табела и ћелије ПДФ документа

- **Додавање слике**

```
ImageData data = ImageDataFactory.Create(imFile);

// Creating an Image object
Image image = new Image(data);
image.SetHeight(50);
image.SetWidth(30);
image.SetFixedPosition(110, 700);
```

Слика 10 - Додавање слике у ПДФ документ

Овако креирана слика се врло лако додаје ћелији. На слици 9 се то може и видети.

- **Манипулисање и читање постојећих ПДФ датаотека**

```
PdfDocument pdfDoc = new PdfDocument(new PdfReader("postojeciPDF.pdf"), new PdfWriter("noviPDF.pdf"));
Document dokument = new Document(pdfDoc);

// dodavanje sadržaja na stranicu
Paragraph pasus = new Paragraph("Zdravo!");
dokument.Add(pasus);

dokument.Close();
```

Слика 11 - Манипулисање постојећег документа

1.4. Нити

C# подржава паралелно извршење кода помоћу извршавања више нити одједном (*multithreading*). Нит представља независну путању извршења која може да се покрене истовремено са другим нитима. Другим речима нит је секвенца извршавања програма. Када се креира нит, она извршава методу која јој је додељена, то значи да се нити извршавају независно, без мешања са другим нитима које се извршавају истовремено.

Нит се креира користећи класу *Thread* која се налази у *System.Threading* именском простору. Једна истанца класе *Thread* представља једну нит то јест једну секвенцу извршавања. Класа *Thread* омогућава да се креирају нити, контролишу и добијају статуси у ком се нити налазе. Да би се нит креирала потребно је да се класи *Thread* проследи метода која треба да се изврши у датој нити.

```
Thread thread = new Thread(Process);
thread.Start();
```

Слика 12 - Креирање нити

C# клијентски програм (*Console, Windows form...*) почиње у једној нити коју аутоматски креира *CLR* и оперативни систем. Ова нит се назива “главна нит”. Могуће је и креирање нових нити.

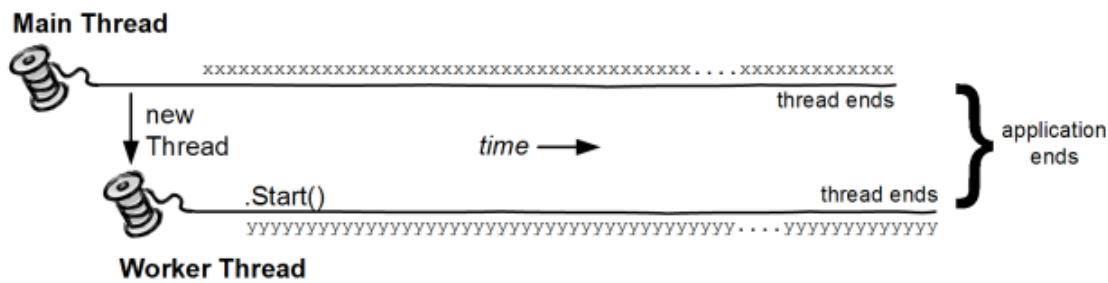
```
class ThreadTest
{
    static void Main()
    {
        Thread t = new Thread (WriteY);           // Kick off a new thread
        t.Start();                                // running WriteY()

        // Simultaneously, do something on the main thread.
        for (int i = 0; i < 1000; i++) Console.Write ("x");
    }

    static void WriteY()
    {
        for (int i = 0; i < 1000; i++) Console.Write ("y");
    }
}
```

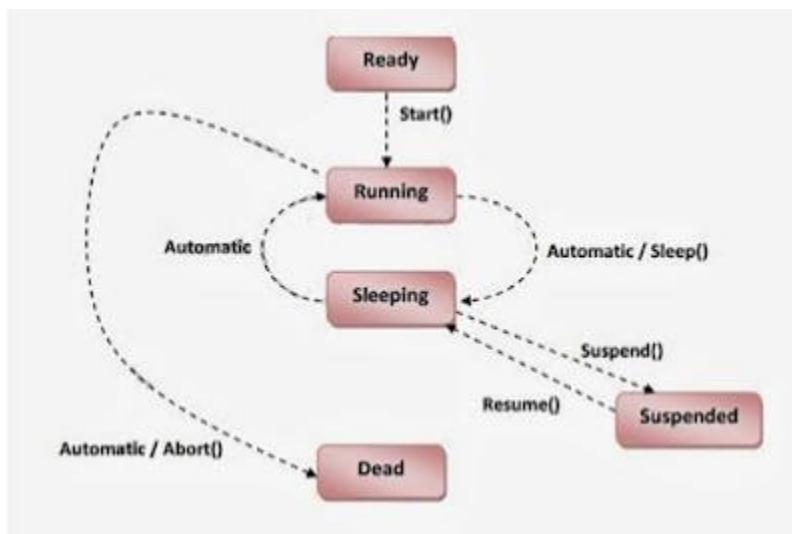
Слика 13 - Креирање нове нити

Главна нит креира нову нит. Нова нит покреће методу која више пута штампа знак “y”. Истовремено главна нит штампа знак “x”. На следећој слици је објашњено.



Слика 14 - Креирање нове нити - објашњење¹⁰

Креирањем објекта *Thread* нит још увек није активна. Нит се активира методом *Start()*. Све док је нит активирана *IsAlive* својство (*property*) враћа вредност *true*. Нит се може и суспендовати, паузирати, наставити или прекинути. Нит се завршава када делегат прослеђен конструктору нити заврши извршавање. Завршену нит није могуће поново покренути, ако се покуша поновно покетање позивањем методе *Start()* јавља се изузетак *ThreadStateException*.



Слика 15 - Животни циклус нити¹¹

Ово су нека од својства нити:

- *Thread.CurrentThread* – враћа инстанцу тренутне нити која се извршава
- *Thread.IsAlive* – уколико је нит активна враћа *true*, уколико нит није покренута или је завршена враћа *false*
- *Thread.Name* – када се нити извршавају истовремено помоћу овог својства могу да се прате. Користи се за добијање или постављање имена нити.

¹⁰ Примери на овој страни су преузети из ПДФ књиге: Jozeph Albahari, *Threading in C#*, последњи пут ажурирано 2011-04-07.

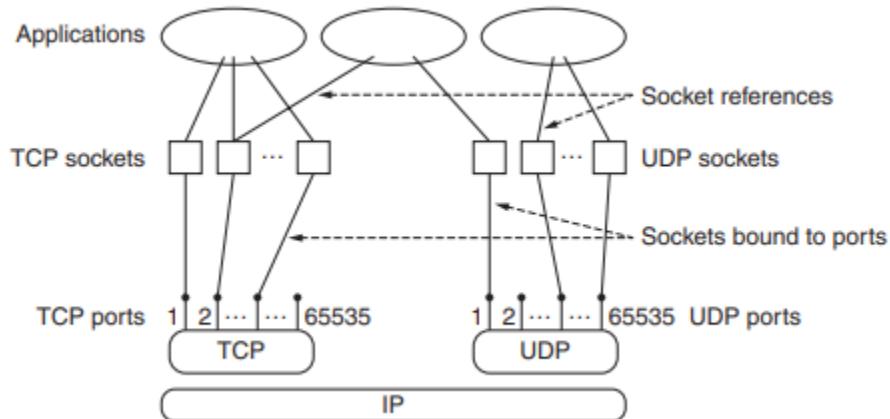
¹¹ Слика је преузета са: <https://www.manuelradovanovic.com/2016/04/rad-sa-nitima-u-c-programskom-jeziku.html>

- *Thread.Priority* - могуће је додати приоритет нити, приоритети су дефинисани помоћу енумерације (*Lowest, Normal, Highest*)
- *Thread.ThreadState* - могуће је добити стање (*Unstarted, Running, Stopped, WithSleepJoin, Suspended, AbortRequested, Aborted, Stopped*) нити
- *Thread.IsBackground* - ако је ово својство постављено на *true*, нит ће се завршити када се главна нит програма заврши, уколико је постављено на *false* нит ће наставити да се извршава и након завршетка главне нити.

1.5. Сокети

Сокети представљају класу помоћу које два или више рачунара могу да комуницирају преко мреже. Коришћењем сокета једна апликација може да се прикључи на мрежу и да шаље и прима податке од друге апликације која је такође на мрежи. То значи да сокети омогућавају да се на једној апликацији читају подаци који су послати преко сокета са апликације која је на другом уређају. Постоје две врсте сокета *TCP* и *UDP* сокети. *TCP* сокети се користе за поуздану комуникацију између рачунара, гарантују да ће подаци бити испоручени без губитака. *UDP* сокети се користе за брзу комуникацију, без потребе да се конекција стално успоставља. Сокети се идентификују помоћу:

- Интернет адресе
- *End-to-end* протокол(*TCP* и *UDP*)
- Броја порта



Слика 16- Сокети, протоколи и портови¹²

TCP протокол представља двосмерни канал. За сваки крај везе се веже инстанца једне до класа *TcpClient* или *TcpListener*. Крајеви канала идентификовани су и *IP* адресом и бројем порта, који се подешавају коришћењем класе *EndPoint* и њене подклasse *IPEndPoint*.

¹² Слика је преузета из пдф књиге: David B. Makofske, Michael J. Donahoo, Kenneth L. Calvert, TCP/IP Socket in C# Particular guide for programmers

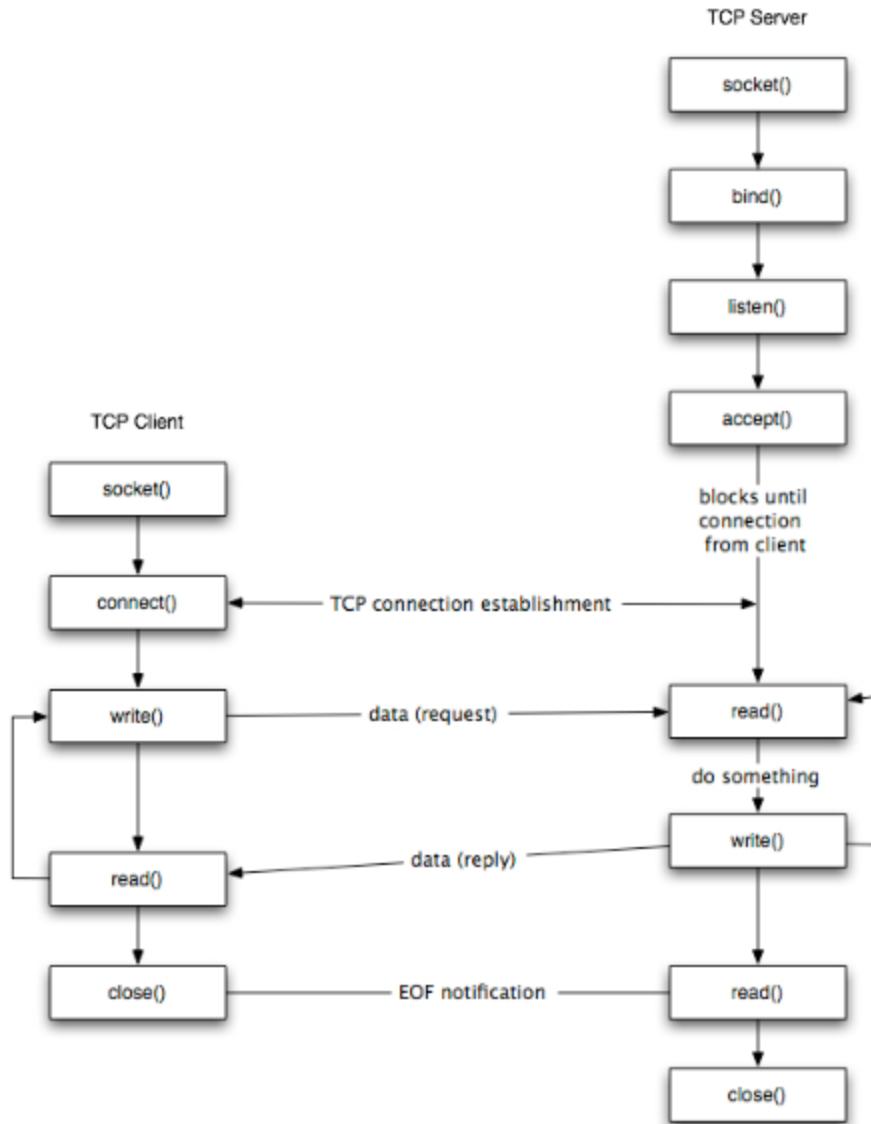
Комуникација се одвија тако што клијентов *TCP* шаље захтев за везу *TCP* серверу. Инстанца *TcpListener*-а ослушају захтеве за *TCP* везу. Када се захтев појави креира се нови сокет као *TcpClient* или *Socket* инстанца.

1.5.1.1. Животни циклус *TCP* сокета

Када клијент позове конструктор са серверском интернет адресом и портом креира се инстанца сокета која је у затвореном стању. Копирају се локална и удаљена адреса и порт у основну структуре сокета, покреће се руковање(*handshake*) за успостављање *TCP* везе. Овај процес се назива “троструко руковање” (*3-way handshake*) јер се састоји од три корака:

- Први корак је да клијент шаље захтев за успостављање конекције до сервера.
- Други корак је да сервер шаље потврду до клијента.
- Трећи корак је да клијент након што прими потврду од сервера, серверу пошаље потврду.

Може се десити да клијент не добије поруку од сервера јер се порука изгуби, или ако сервер не прихвати позив од клијента и проследи поруку одбијања, то доводи до изузетка *SocketException* са кодом грешке 10061 што значи да је конекција одбијена. На серверској страни се креира инстанца *TCPListener/Socket*. Након позива методе *Start()* за *TCPListener* или *Listen()* за *Socket*, сокет прелази у стање “слушање” и постаје спреман да прихвати долазну везу. Сервер може да блокира прихватање позива за конекцију све док се не заврши торслојно руковање. Када захтев за конекцију стигне, креира се нова структура сокета у којој се адреса попуњава на основу података који су пристигли. Након овога сокет прелази у стање “повезивање”(*Connecting*). Тек након што се троструко руковање заврши и стигне трећа порука, сокет прелази у стање “успостављено”(*Established*). Са друге стране клијент може да пошаље податке одмах након друге поруку током троструког руковања. *TCP* има механизам затварања који омогућава апликацијама да прекине везу, а да није потребно бринути о губитку података који су још увек у транзиту. Механизам је такође дизајниран да омогући да се пренос података у сваком смеру прекине независно. Након што се позове метода *Close()*, *TCP* чека на потврду о затварању руковања, која указује да су сви подаци безбедно послати. Када се потврда прими веза је делимично затворена. Потребно је да се затварање позове и у другом правцу. *TCP* може да се затвори на два начина: када се у једној апликацији позове *Close()* пре него што се то деси у другој апликацији или истовремено, тако да се њихове поруке затварања укрштају у мрежи.



Слика 17 - Клијент - сервер конекција¹³

¹³ Слика је преузета са: <https://www.cs.dartmouth.edu/~campbell/cs60/socketprogramming.html>

1.6. Model view controller (MVC) патерн

Model-View-Controller архитектура представља патерн који се често користи у развоју софтвера како би се раздвојила логика апликације од приказа података кориснику. Обично се користи за развој корисничких интерфејса. Захваљујући оваквој архитектури могуће је поново употребити већ постојећи софтверски код, олакшава развој и касније одржавање апликационог софтвера. Компоненте из којих се састоји су независне, уколико се нешто промени у једној нису потребне велике измене у другим компонентама. Постоје 3 главне компоненте:

- Модел

Компонента представља логику апликације и управља подацима. Обухвата динамичку структуру података. Сва пословна логика апликације садржана је у моделу. Обрађује податке и даље их прослеђује до погледа и контролера.

- Поглед(*View*)

Представља кориснички интерфејс за приказ података који се добијају од модела. Задатак ове компоненте је што једноставније приказати податке кориснику. Служи само за приказ података, не врши никакве измене у моделу. Ово је последња слој *MVC* архитектуре.

- Контролер

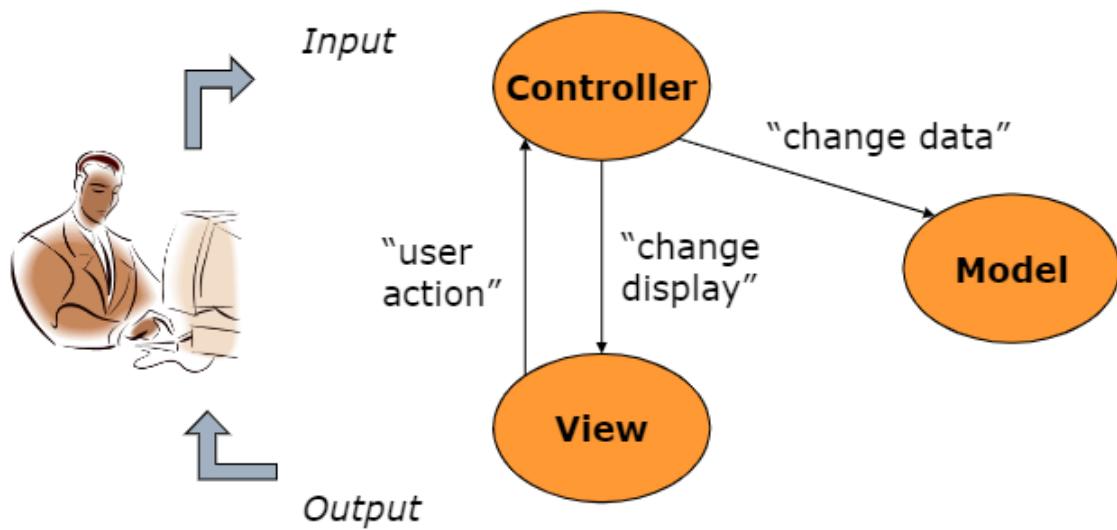
Користи се за управљање интеракцијом корисника са апликацијом. Прима *inpute* од корисника, обрађује их и ажурира модел ако је то потребно. Контролер прослеђује податке погледу како би их приказао кориснику. Може да контролише ток програма, односно понашање саме апликације и управља корисничким захтевима, па тако има способност да утиче на стање модела.

“Модел представља “тело”, приказ података “очи”, а контролер “мозак” пројекта.”¹⁴

Корисник уноси податке и позива операције које треба да се изврше над подацима. Контролер ослушкује и прихвата захтеве за извршење неке операције. Позива операцију која је дефинисана у моделу. Када се изврши операција над објектима може се променити модел података. Када модел промени стање, ново стање се приказује кориснику тако што се преко контролера приказује кориснику.

Модел не зависи од остале две компоненте, док поглед и контролер зависе од модела.

¹⁴ Преузето са: https://sr.wikipedia.org/wiki/MVC_arhitektura



[Слика 18 - MVC патерн¹⁵](#)

Постоје многобројне предности *MVC* патерна: модел се може приказивати на више начина, могућност поновног коришћења кода, појединачни делови се могу лако мењати, тестирали и побољшавати, приказ података одвојен је од логике података... Што се мана тиче, оваква архитектура може бити компелксна за мање апликације, што може да доведе до погоршања њених карактеристика. Уколико постоје честе измене модела, приказ података може остати преплављен захтевима за измену.

¹⁵ Слика је презета са: <https://docplayer.net/13965042-Model-view-controller-mvc-design-pattern.html>

2. Прикупљање захтева

2.1.Вербални опис модела

Потребно је креирати софтверски систем за праћење рада спа и велнес центра. Софтверски систем би омогућио лакше чување и управљање подацима. Евиденције би се водиле о запосленима, клијентима, услугама, лојалти програму, рачунима, начинима плаћања, категоријама, улогама које запослени може да има.

Корисник система ће бити запослени из спа и велнес центра. Постојала би два типа запослених, обичан и запослени који има админ улогу коме би због дате улоге било омогућено коришћење више функционалности него обичном запосленом.

Услуге би се састојале од разних третмана, масажа, неге лица и тела.

Клијенти представљају регистроване кориснике спа и велнес центра. Редовни клијенти могу да остваре могућност на лојалти програм који им омогућава коришћење спа и велнес центра по повољнијим ценама.

У систему ће постојати могућност издавања рачуна за одрађене услуге.

Обичном кориснику би биле омогућене следеће функционалности: унос нових, претрага, измена и деактивирање клијената. Такође потребно је омогућити и унос, претрагу, измену и деактивирање услуга. Што се рачуна тиче, постојала би могућност креирања и сторнирања рачуна, док би подаци о услугама и клијентима били већ расположиви за ову функционалност. Уколико се рачун креира за клијента који је регистрован и који поседује лојалти, износ на рачуну би био умањен за попуст у зависности од типа лојалтија. На крају, систем ће омогућити да се за рачун који је издат креира ПДФ документ користећи неку од библиотека предвиђену за ову функционалност.

Админ кориснику би биле поред функционалности наведених за обичног корисника омогућене и следеће функционалности: унос, измена, деактивирање запослених. Постојала би и могућност додавање нових улога запосленима. Што се лојалти програма тиче админ кориснику би било омогућено да уноси, мења и деактивира лојалти програме.

2.2. Случајеви коришћења

За потребе развијаног софтвера идентификовани су следећи случајеви коришћења који су приказани и на слици 1:

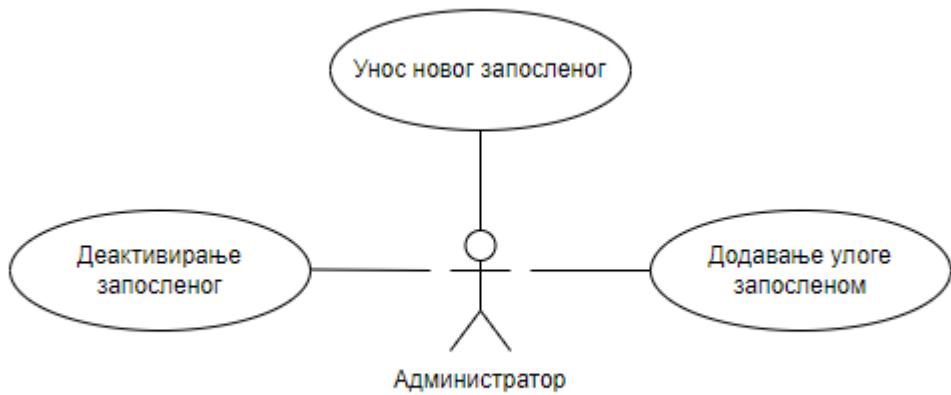
- 1) унос новог клијента
- 2) претраживање клијента
- 3) измена података о клијенту
- 4) унос нове услуге
- 5) претраживање услуге по категорији
- 6) измена података о услуги
- 7) креирање рачуна
- 8) сторнирање рачуна



Слика 19 - Дијаграм случајева коришћења – корисник

Поред случајева коришћења са слике 1, **администратору** ће бити омогућени и следећи случајеви коришћења:

- 1) унос новог запосленог
- 2) деактивирање запосленог
- 3) додавање улоге запосленом



Слика 20 - Дијаграм случајева коришћења - админ

СК1: Случај коришћења – Унос новог клијента

Назив СК

Унос новог клијента

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за унос клијента. Учитана је листа са типовима лојалти програма.

Основни сценарио СК

1. Корисник уноси податке о новом клијенту. (АПУСО)
2. Корисник контролише да ли је коректно унео податке о новом клијенту. (АНСО)
3. Корисник позива систем да запамти податке о клијенту. (АПСО)
4. Систем памти податке о клијенту. (СО)
5. Систем приказује поруку: "Клијент је сачуван!". (ИА)

Алтернативна сценарија

5.1. Уколико систем не може да запамти податке о клијенту он приказује кориснику поруку "Клијент није сачуван. Проверите унете податке!". (ИА)

СК2: Случај коришћења – Претраживање клијента

Назив СК

Претраживање клијента

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са клијентима.

Основни сценарио СК

1. Корисник уноси вредност по којој претражује клијенте. (АПУСО)
2. Корисник позива систем да нађе клијенте по задатој вредности. (АПСО)
3. Систем претражује клијенте по задатој вредности. (СО)
4. Систем приказује кориснику листу клијената и поруку "Претрага је завршена!". (ИА)
5. Корисник бира једног клијента. (АПСО)
6. Систем учитава податке о одабраном клијенту. (СО)
7. Систем приказује кориснику податке о одабраном клијенту. (ИА)

Алтернативна сценарија

4.1. Уколико корисник није унео довољан број карактера за претрагу систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!". (ИА)

4.2. Уколико систем не може да нађе клијенте систем приказује кориснику поруку: "Ниједан клијент није пронађен!" (ИА)

7.1. Уколико систем не може да учита податке о одабраном клијенту, он приказује кориснику поруку: "Систем не може да учита податке о изабраном клијенту!".

СК3: Случај коришћења – Измена података о клијенту

Назив СК

Измена података о клијенту

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са клијентима. Учитана је листа са типовима лојалти програма.

Основни сценарио СК

1. Корисник уноси вредност по којој претражује клијенте. (АПУСО)
2. Корисник позива систем да нађе клијенте по задатој вредности. (АПСО)
3. Систем тражи клијенте по задатој вредности. (СО)
4. Систем приказује кориснику листу клијената и поруку "Претрага је завршена!". (ИА)
5. Корисник бира клијента чије податке жели да измени. (АПУСО)
6. Корисник позива систем да учита податке о одабраном клијенту. (АПСО)
7. Систем учитава податке о одабраном клијенту. (СО)
8. Систем приказује кориснику податке о клијенту. (ИА)
9. Корисник уноси (мења) податке о клијенту. (АПУСО)
10. Корисник контролише да ли је коректно унео податке о клијенту. (АНСО)
11. Корисник позива систем да запамти податке о клијенту. (АПСО)
12. Систем памти податке о клијенту. (СО)
13. Систем приказује кориснику поруку: "Подаци о клијенту су изменjeni!". (ИА)

Алтернативна сценарија

4.1. Уколико корисник није унео довољан број карактера за претрагу систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)

4.2. Уколико систем не може да нађе клијента он приказује кориснику поруку: "Ниједан клијент није пронађен!". Прекида се извршење сценарија. (ИА)

8.1. Уколико систем не може да учита клијента он приказује кориснику поруку: "Систем не може да учита податке о изабраном клијенту!". Прекида се извршење сценарија. (ИА)

13.1. Уколико систем не може да запамти податке о клијенту он приказује кориснику поруку: "Клијент није изменjen. Проверите изменене податке!". (ИА)

СК4: Случај коришћења – Унос нове услуге

Назив СК

Унос нове услуге

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за унос услуге. Учитана је листа са категоријама.

Основни сценарио СК

1. Корисник уноси податке о услуги. (АПУСО)
2. Корисник контролише да ли је коректно унео податке о услуги. (АНСО)
3. Корисник позива систем да запамти податке о услуги. (АПСО)
4. Систем памти податке о услуги. (СО)
5. Систем приказује кориснику поруку: "Услуга је сачуван!". (ИА)

Алтернативна сценарија

- 5.1. Уколико систем не може да запамти податке о услуги он приказује кориснику поруку: "Услуга није сачувана. Проверите унете податке!". (ИА)

СК5: Случај коришћења – Претраживање услуге

Назив СК

Претраживање услуге

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са услугама.

Основни сценарио СК

1. Корисник уноси вредност по којој претражује услуге. (АПУСО)
2. Корисник позива систем да нађе услуге по задатој вредности. (АПСО)
3. Систем претражује услуге по задатој вредности. (СО)
4. Систем приказује кориснику листу услуга и поруку "Претрага је завршена!". (ИА)
5. Корисник бира једну услугу. (АПСО)
6. Систем учитава податке о одабраној услуги. (СО)
7. Систем приказује кориснику податке о одабраној услуги. (ИА)

Алтернативна сценарија

4.1. Уколико систем не може да нађе услуге систем приказује кориснику поруку: "Ниједна услуга није пронађена!" (ИА)

7.1. Уколико систем не може да учита податке о одабраној услуги, он приказује кориснику поруку: "Систем не може да учита податке о одабраној услуги!".

СК6: Случај коришћења – Измена података о услуги

Назив СК

Измена података о услуги

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са услугама. Учитана је листа са категоријама.

Основни сценарио СК

1. Корисник уноси вредност по којој претражује услуге. (АПУСО)
2. Корисник позива систем да нађе услуге по задатој вредности. (АПСО)
3. Систем тражи услуге по задатој вредности. (СО)
4. Систем приказује кориснику листу услуга и поруку "Претрага је завршена!". (ИА)
5. Корисник бира услугу чије податке жели да измени. (АПУСО)
6. Корисник позива систем да учита податке о одабраној услуги. (АПСО)
7. Систем учитава податке о одабраној услуги. (СО)
8. Систем приказује кориснику податке о услуги. (ИА)
9. Корисник уноси (мења) податке о услуги. (АПУСО)
10. Корисник контролише да ли је коректно унео податке о услуги. (АНСО)
11. Корисник позива систем да запамти податке о услуги. (АПСО)
12. Систем памти податке о услуги. (СО)
13. Систем приказује кориснику поруку: "Подаци о услуги су изменjeni!". (ИА)

Алтернативна сценарија

4.1. Уколико корисник није унео довољан број карактера за претрагу систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)

4.2. Уколико систем не може да нађе услугу он приказује кориснику поруку: "Ниједна услуга није пронађена!". Прекида се извршење сценарија. (ИА)

8.1. Уколико систем не може да учита услугу он приказује кориснику поруку: "Систем не може да учита податке о одабраној услуги!". Прекида се извршење сценарија. (ИА)

13.1. Уколико систем не може да запамти податке о услуги он приказује кориснику поруку: "Услуга није изменjena. Проверите изменjene податке!". (ИА)

СК7: Случај коришћења – Креирање рачуна

Назив СК

Креирање рачуна

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са рачунима. Листа Начини плаћања је учитана. Листа Услуге је учитана.

Основни сценарио СК

1. Корисник уноси вредност по којој претражује клијенте. (АПУСО)
2. Корисник позива систем да нађе клијенте по задатој вредности. (АПСО)
3. Систем тражи клијенте по задатој вредности. (СО)
4. Систем приказује кориснику листу клијената и поруку: "Претрага је завршена!". (ИА)
5. Корисник бира клијента ког жели да дода на рачун. (АПУСО)
6. Корисник бира услуге које жели да дода на рачун. (АПУСО)
7. Корисник уноси податке о рачуну. (АПУСО)
8. Корисник контролише да ли је коректно унео све потребне податке. (АНСО)
9. Корисник позива систем да запамти податке о рачуну. (АПСО)
10. Систем памти податке о рачуну. (СО)
11. Систем приказује кориснику поруку: "Рачун је сачуван!". (ИА)

Алтернативна сценарија

4.1. Уколико систем не може да нађе клијенте он приказује кориснику поруку: "Ниједан клијент није пронађен!". Прекида се извршење сценарија. (ИА)

4.2. Уколико корисник није унео довољан број карактера за претрагу клијената систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)

11.1. Уколико систем не може да запамти податке о рачуну он приказује кориснику поруку: "Рачун није сачувана. Проверите унете податке!". (ИА)

СК8: Случај коришћења – Сторнирање рачуна

Назив СК

Сторнирање рачуна

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са рачунима.

Основни сценарио СК

1. Корисник уноси вредност по којој претражује рачуне. (АПУСО)
2. Корисник позива систем да нађе рачуне по задатој вредности. (АПСО)
3. Систем тражи рачуне по задатој вредности. (СО)
4. Систем приказује кориснику листу рачуна и поруку: "Претрага је завршена!". (ИА)
5. Корисник бира рачун који жели да сторнира. (АПУСО)
6. Корисник позива систем да учита податке о одабраном рачуну. (АПСО)
7. Системчитава податке о одабраном рачуну. (СО)
8. Систем приказује кориснику податке о рачуну и исписује поруку: "Тражени рачун је приказан!". (ИА)
9. Корисник позива систем да сторнира рачун. (АПСО)
10. Систем сторнира рачун. (СО)
11. Систем приказује кориснику поруку: "Изабрани рачун је сторниран! ". (ИА)

Алтернативна сценарија

4.1. Уколико систем не може да нађе рачуне он приказује кориснику поруку: "Ниједан рачун није пронађен!". Прекида се извршење сценарија. (ИА)

4.2. Уколико корисник није унео довољан број карактера за претрагу рачуна систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)

8.1. Уколико систем не може да учита рачун, он приказује кориснику поруку: "Систем није учитао рачун!". Прекида се извршење сценарија. (ИА)

11.1. Уколико систем не може да сторнира рачун он приказује кориснику поруку: "Рачун није сторниран!". (ИА)

СК9: Случај коришћења – Унос новог запосленог

Назив СК

Унос новог запосленог

Актори СК

Администратор

Учесници СК

Администратор и систем (програм)

Предуслов: Систем је укључен и администратор је пријављен под својом шифром. Систем приказује форму за унос запосленог.

Основни сценарио СК

1. Администратор уноси податке о запосленом. (АПУСО)
2. Администратор контролише да ли је коректно унео податке о запосленом. (АНСО)
3. Администратор позива систем да запамти податке о запосленом. (АПСО)
4. Систем памти податке о запосленом. (СО)
5. Систем приказује кориснику поруку: "Запослени је сачуван!". (ИА)

Алтернативна сценарија

5.1. Уколико систем не може да запамти податке о запосленом он приказује администратору поруку: "Запослени није сачуван. Проверите унете податке!". (ИА)

СК10: Случај коришћења – Деактивирање запосленог

Назив СК

Деактивирање запосленог

Актори СК

Администратор

Учесници СК

Администратор и систем (програм)

Предуслов: Систем је укључен и администратор је пријављен под својом шифром. Систем приказује форму за рад са запосленима.

Основни сценарио СК

1. Администратор уноси вредност по којој претражује запослене. (АПУСО)
2. Администратор позива систем да нађе запослене по задатој вредности. (АПСО)
3. Систем тражи запослене по задатој вредности. (СО)
4. Систем приказује администратору листу запослених и поруку: "Систем је нашао запослене по задатој вредности!". (ИА)
5. Администратор бира запосленог ког жели да деактивира. (АПУСО)
6. Администратор позива систем да учита податке о одабраном запосленом. (АПСО)
7. Системчитава податке о одабраном запосленом. (СО)
8. Систем приказује администратору податке о запосленом. (ИА)
9. Администратор позива систем да деактивира запосленог. (АПСО)
10. Систем деактивира запосленог. (СО)
11. Систем приказује администратору поруку: "Избрани запослени је деактивиран!". (ИА)

Алтернативна сценарија

4.1. Уколико систем не може да нађе запослене он приказује администратору поруку: "Ниједан запослени није пронађен!". Прекида се извршење сценарија. (ИА)

4.2. Уколико администратор није унео довољан број карактера за претрагу запослених систем приказује администратору поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)

8.1. Уколико систем не може да учита запосленог, он приказује администратору поруку: "Систем није учитао запосленог!". Прекида се извршење сценарија. (ИА)

11.1. Уколико систем не може да деактивира запосленог он приказује администратору поруку: "Запослени није деактивиран!". (ИА)

СК11: Случај коришћења – Додавање улоге запосленом

Назив СК

Додавање улоге запосленом

Актори СК

Администратор

Учесници СК

Администратор и систем (програм)

Предуслов: Систем је укључен и администратор је пријављен под својом шифром. Систем приказује форму за рад са запосленима. Листа Улога је учитана.

Основни сценарио СК

12. Администратор уноси вредност по којој претражује запослене. (АПУСО)
13. Администратор позива систем да нађе запослене по задатој вредности. (АПСО)
14. Систем тражи запослене по задатој вредности. (СО)
15. Систем приказује администратору листу запослених и поруку: “ Претрага је завршена!”. (ИА)
16. Администратор бира запосленог ком жели да додели улогу. (АПУСО)
17. Администратор позива систем да учита податке о одабраном запосленом. (АПСО)
18. Систем учитава податке о одабраном запосленом. (СО)
19. Систем приказује администратору податке о запосленом. (ИА)
20. Администратор бира улогу коју жели да додели запосленом. (АПУСО)
21. Администратор уноси податке о вези улога - запослени. (АПУСО)
22. Администратор контролише да ли је коректно изабрао улогу и унео остале потребне податке. (АНСО)
23. Администратор позива систем да запамти податке о вези улога - запослен. (АПСО)
24. Систем памти податке о вези улога - запослен. (СО)
25. Систем приказује кориснику поруку: “Веза улога – запослен је сачувана!”. (ИА)

Алтернативна сценарија

4.1. Уколико систем не може да нађе запослене он приказује администратору поруку: “Ниједан запослени није пронађен!”. Прекида се извршење сценарија. (ИА)

4.2. Уколико администратор није унео довољан број карактера за претрагу запослених систем приказује администратору поруку: “Нисте унели довољан број карактера за претрагу!” Прекида се извршење сценарија. (ИА)

8.1. Уколико систем не може да учита запосленог, он приказује администратору поруку: “Систем није учитао запосленог!”. Прекида се извршење сценарија. (ИА)

14.1. Уколико систем не може да запамти податке о вези улога - запослен он приказује администратору поруку: “Веза улога - запослен није сачувана. Проверите унете податке!”. (ИА)

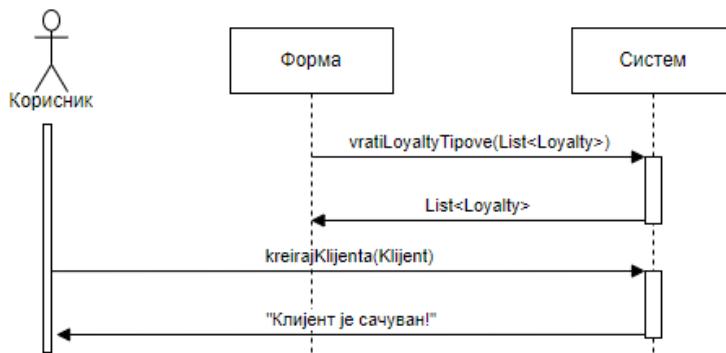
3. Анализа

3.1. Понашање софтверског система – системски дијаграм секвенци

ДС1: Унос новог клијента

Основни сценарио

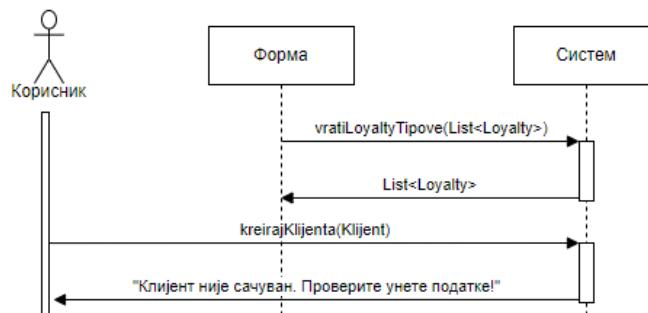
- Форма позива систем да учита податке о типовима лојатија. (АПСО)
- Систем приказује на форми листу лојатија. (ИА)
- Корисник позива систем да запамти податке о клијенту. (АПСО)
- Систем приказује поруку: "Клијент је сачуван!". (ИА)



Слика 21 - ДС Унос новог клијента ОС

Алтернативна сценарија

- 4.1. Уколико систем не може да запамти податке о клијенту он приказује кориснику поруку "Клијент није сачуван. Проверите унете податке!". (ИА)



Слика 22 - ДС Уноса новог клијента АС 1

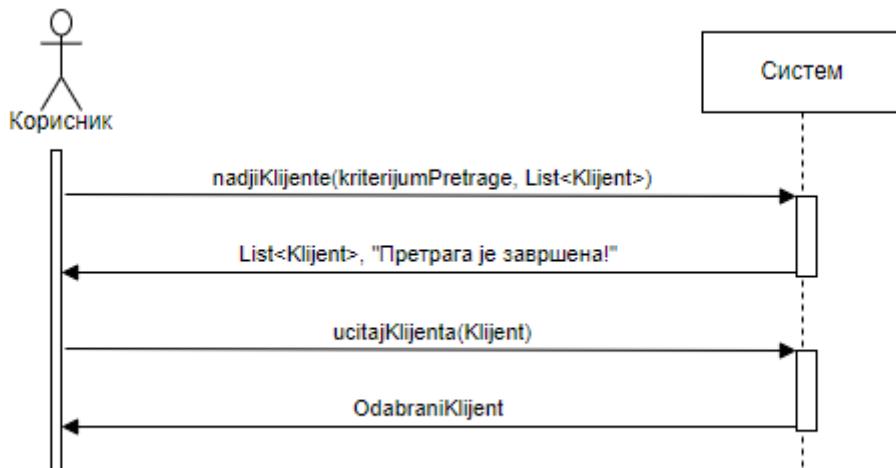
Идентификоване системске операције:

1. signal **KreirajKlijenta**(Klijent)
2. signal **VratiLoyaltyTipove**(List<Loyalty>)

ДС2: Претраживање клијента

Основни сценарио

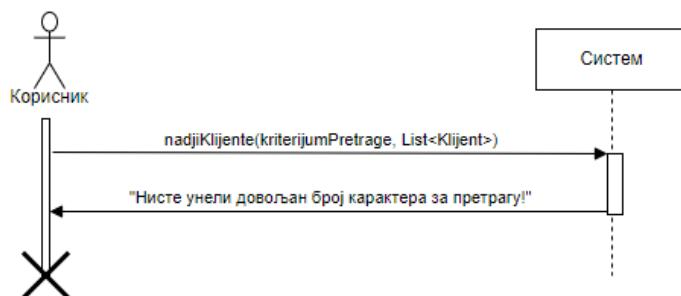
1. Корисник позива систем да нађе клијенте по задатој вредности. (АПСО)
2. Систем приказује кориснику листу клијената и поруку "Претрага је завршена!". (ИА)
3. Корисник бира једног клијента. (АПСО)
4. Систем приказује кориснику податке о одабраном клијенту. (ИА)



Слика 23 - ДС Претраживање клијента ОС

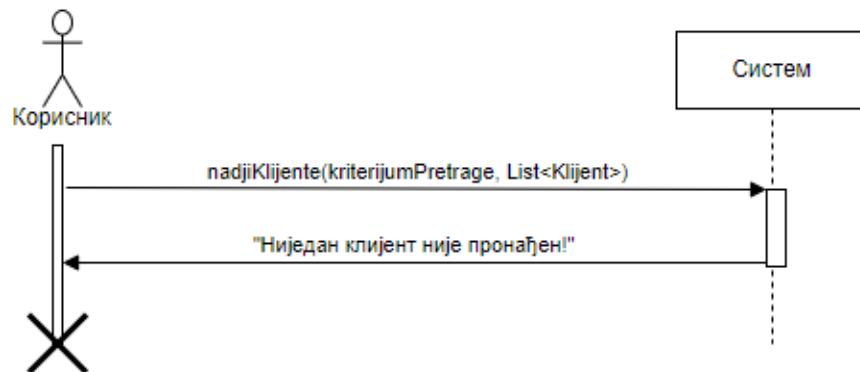
Алтернативна сценарија

- 2.1. Уколико корисник није унео довољан број карактера за претрагу систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!". (ИА)



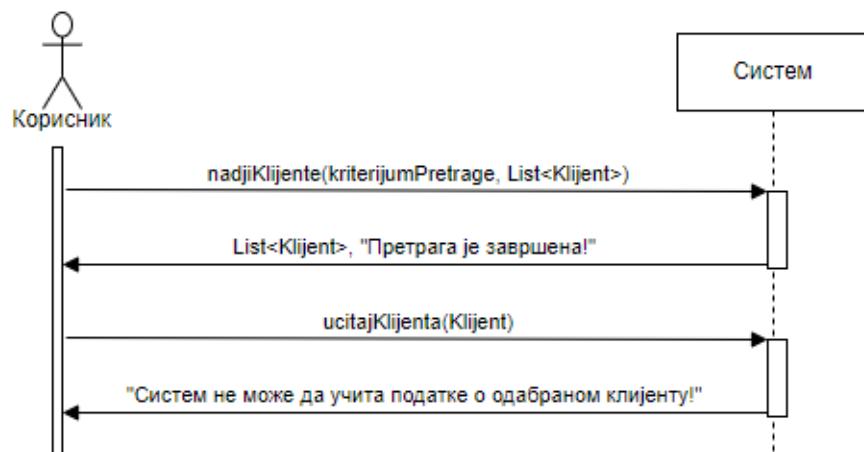
Слика 24 - ДС Претраживање клијента АС 1

2.2. Уколико систем не може да нађе клијенте систем приказује кориснику поруку: "Ниједан клијент није пронађен!" (ИА)



Слика 25 - ДС Претраживање клијента АС 2

4.1. Уколико систем не може да учита податке о одабраном клијенту, он приказује кориснику поруку: "Систем не може да учита податке о изабраном клијенту!".



Слика 9 - ДС Претраживање клијента АС 3

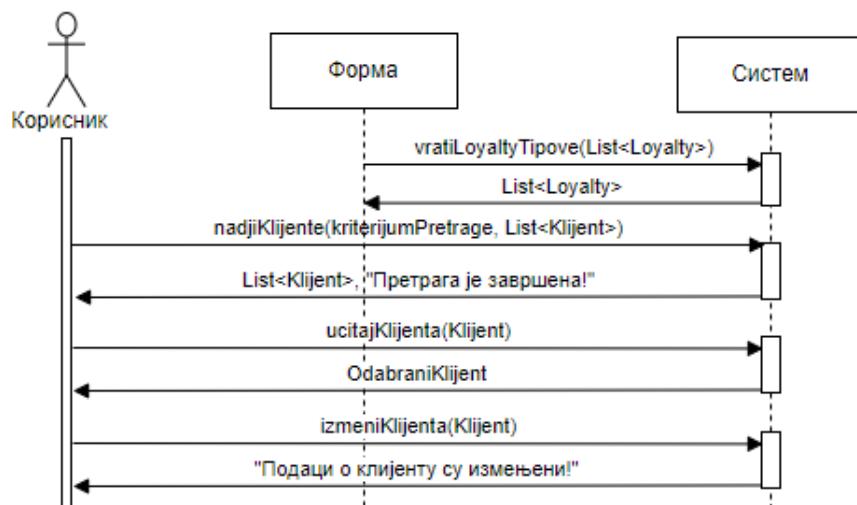
Идентификовани системске операције:

1. signal **NadjiKlijente** (kriterijumPretrage, List<Klijent>)
2. signal **UcitajKlijenta** (Klijent)

ДС3: Измена података о клијенту

Основни сценарио

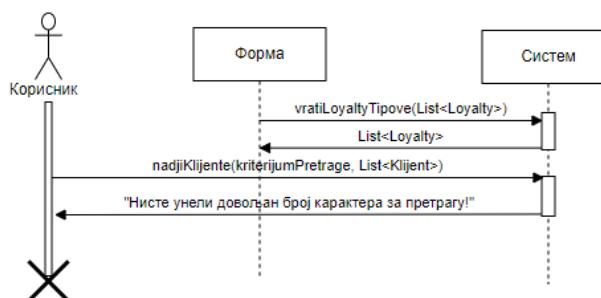
1. Форма позива систем да учита податке о типовима лојатија. (АПСО)
2. Систем приказује на форми листу лојатија. (ИА)
3. Корисник позива систем да нађе клијенте по задатој вредности. (АПСО)
4. Систем приказује кориснику листу клијената и поруку "Претрага је завршена!". (ИА)
5. Корисник позива систем да учита податке о одабраном клијенту. (АПСО)
6. Систем приказује кориснику податке о клијенту. (ИА)
7. Корисник позива систем да запамти податке о клијенту. (АПСО)
8. Систем приказује кориснику поруку: "Подаци о клијенту су изменјени!". (ИА)



Слика 26 - ДС Измена података о клијенту ОС

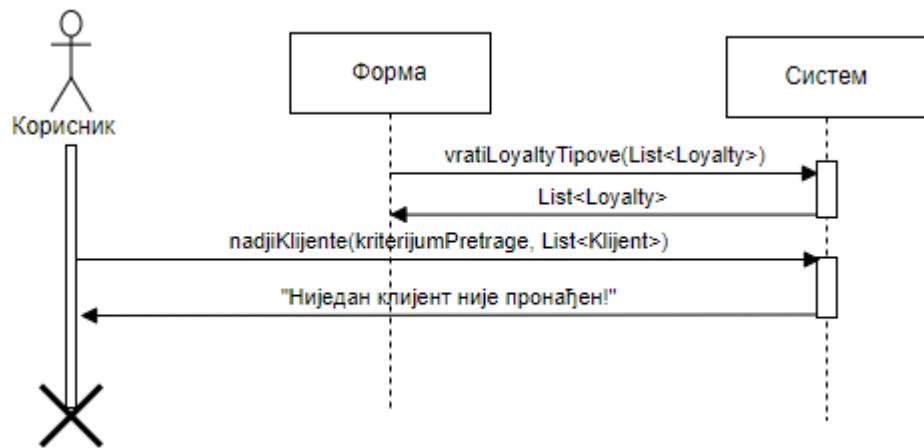
Алтернативна сценарија

- 4.1. Уколико корисник није унео довољан број карактера за претрагу систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



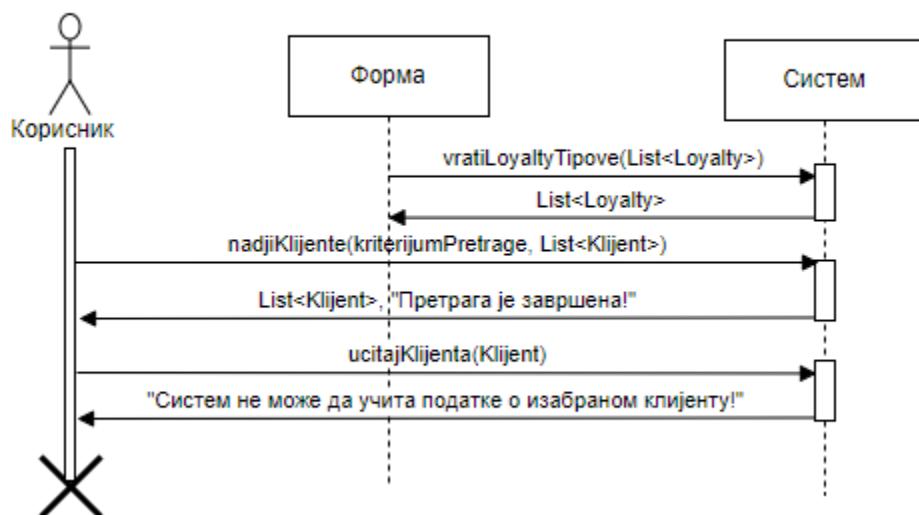
Слика 27 - ДС Измена података о клијенту АС1

4.2. Уколико систем не може да нађе клијента он приказује кориснику поруку: "Ниједан клијент није пронађен!". Прекида се извршење сценарија. (ИА)



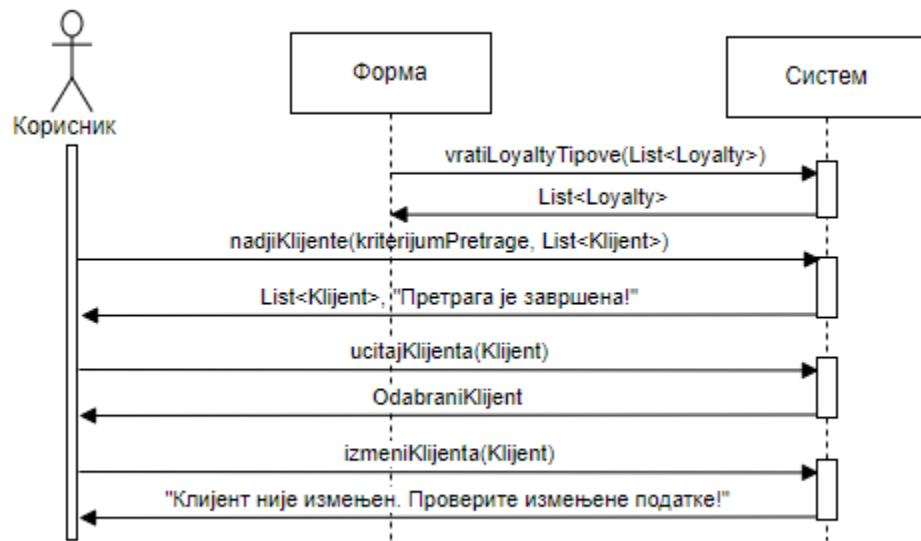
Слика 28 - ДС Измена података о клијенту АС2

6.1. Уколико систем не може да учита клијента он приказује кориснику поруку: "Систем не може да учита податке о изабраном клијенту!". Прекида се извршење сценарија. (ИА)



Слика 29 - ДС Измена података о клијенту АС3

8.1. Уколико систем не може да запамти податке о клијенту он приказује кориснику поруку: "Клијент није изменјен. Проверите изменење податке!". (ИА)



Слика 30 - ДС Измена података о клијенту АС4

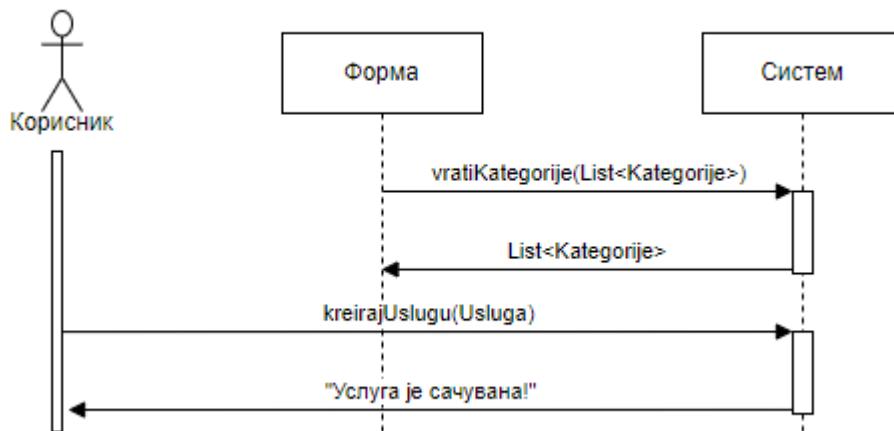
Идентификоване системске операције:

1. signal **VratiloyaltyTipove** (List<Loyalty>)
2. signal **NadjiKlijente** (kriterijumPretrage, List<Klijent>)
3. signal **UcitajKlijenta** (Klijent)
4. signal **IzmeniKlijenta**(Klijent)

ДС4: Унос нове услуге

Основни сценарио

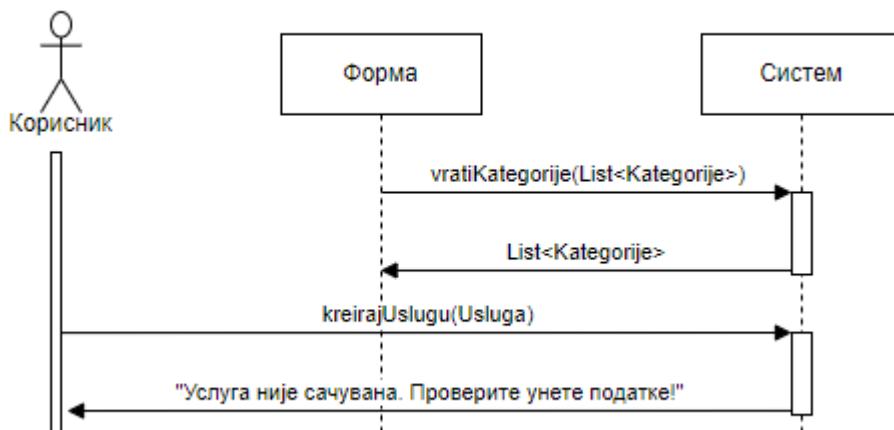
- Форма позива систем да учита податке о категоријама. (АПСО)
- Систем приказује на форми листу категорија. (ИА)
- Корисник позива систем да запамти податке о услуги. (АПСО)
- Систем приказује кориснику поруку: "Услуга је сачувана!".(ИА)



Слика 31 - ДС Унос нове услуге ОС

Алтернативна сценарија

- 4.1. Уколико систем не може да запамти податке о услуги он приказује кориснику поруку: "Услуга није сачувана. Проверите унете податке!". (ИА)



Слика 32 - ДС Унос нове услуге АС1

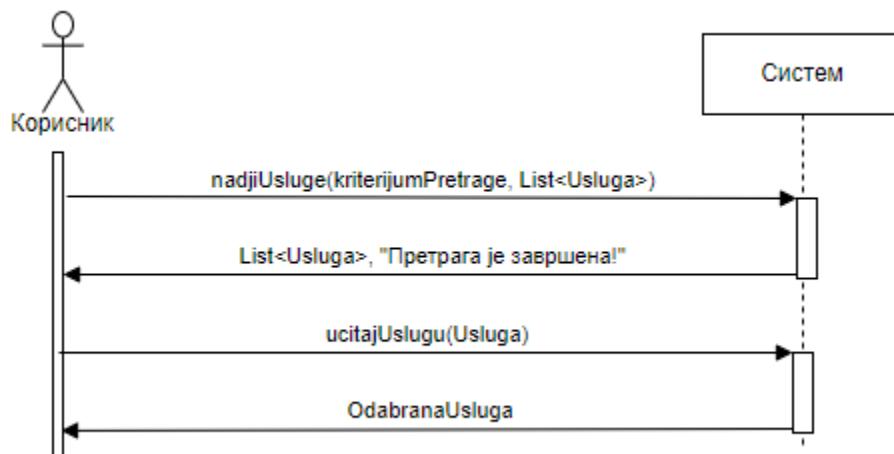
Идентификација системских операција:

1. signal **VratiKategorije** (List<Kategorija>)
2. signal **KreirajUslugu**(Usluga)

ДС5: Претраживање услуге

Основни сценарио

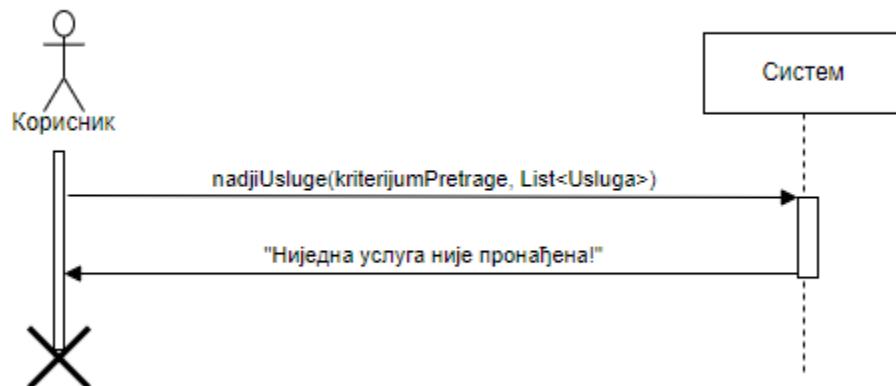
1. Корисник позива систем да нађе услуге по задатој вредности. (АПСО)
2. Систем приказује кориснику листу услуга и поруку: "Претрага је завршена!". (ИА)
3. Корисник бира једну услугу. (АПСО)
4. Систем приказује кориснику податке о одабраној услуги. (ИА)



Слика 33 - ДС Претраживање услуге ОС

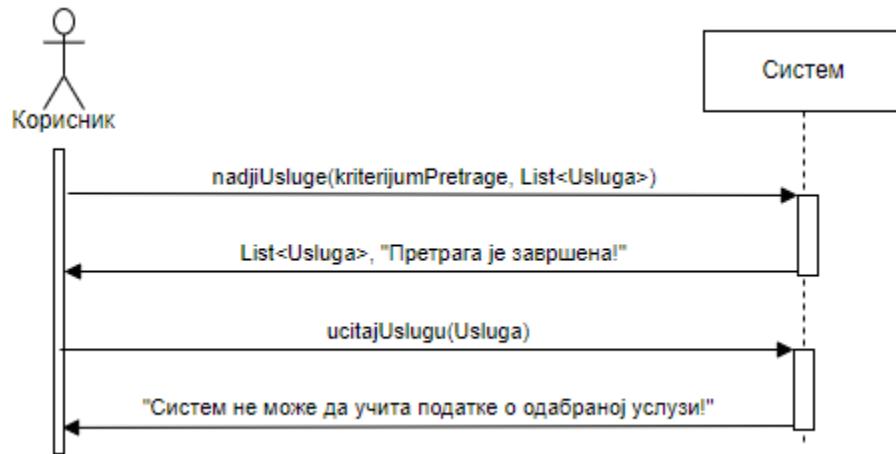
Алтернативна сценарије

- 2.1. Уколико систем не може да нађе услуге систем приказује кориснику поруку: "Ниједна услуга није пронађена!" (ИА)



Слика 34 - ДС Претраживање услуге АС1

4.1. Уколико систем не може да учита податке о одабраној услуги, он приказује кориснику поруку: "Систем не може да учита податке о одабраној услуги!".



Слика 18 - ДС Претраживање услуге АС2

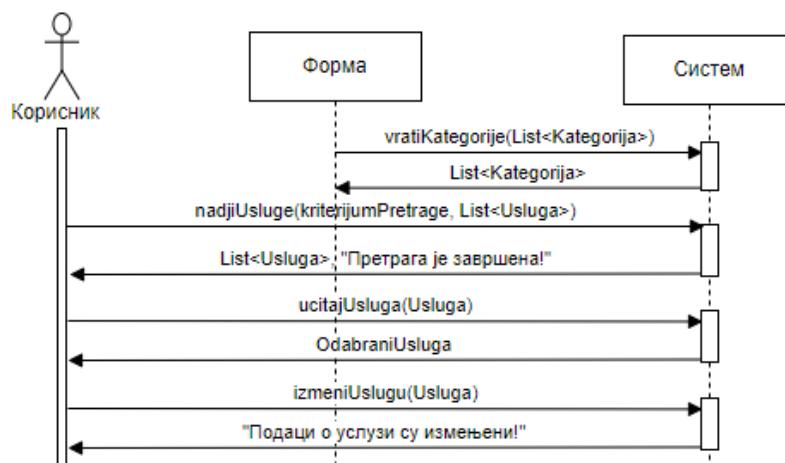
Идентификоване системске операције:

1. signal **NadjiUsluge** (kriterijumPretrage, List<Usluga>)
2. signal **UcitajUslugu** (Usluga)

ДС6: Измена података о услуги

Основни сценарио

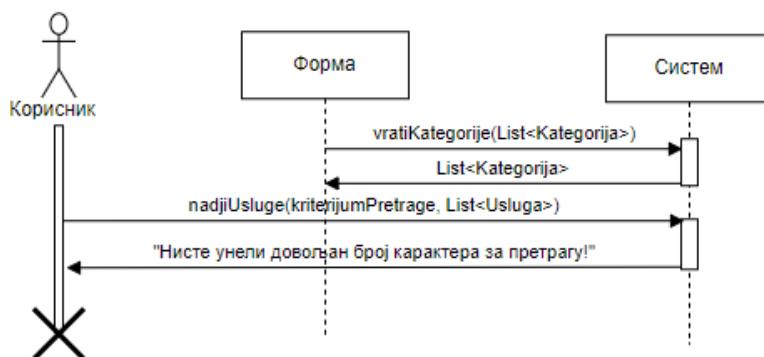
1. Форма позива систем да учита податке о категоријама. (АПСО)
2. Систем приказује на форми листу категорија. (ИА)
3. Корисник позива систем да нађе услуге по задатој вредности. (АПСО)
4. Систем приказује кориснику листу услуга и поруку "Претрага је завршена!". (ИА)
5. Корисник позива систем да учита податке о одабраној услуги. (АПСО)
6. Систем приказује кориснику податке о услуги. (ИА)
7. Корисник позива систем да запамти податке о услуги. (АПСО)
8. Систем приказује кориснику поруку: "Подаци о услуги су изменењени!". (ИА)



Слика 35 - ДС Измена података о услуги ОС

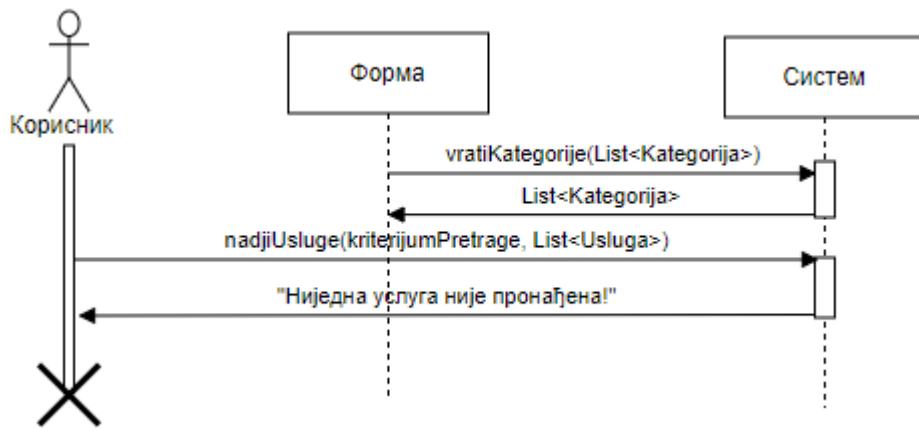
Алтернативна сценарија

- 4.1. Уколико корисник није унео довољан број карактера за претрагу систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



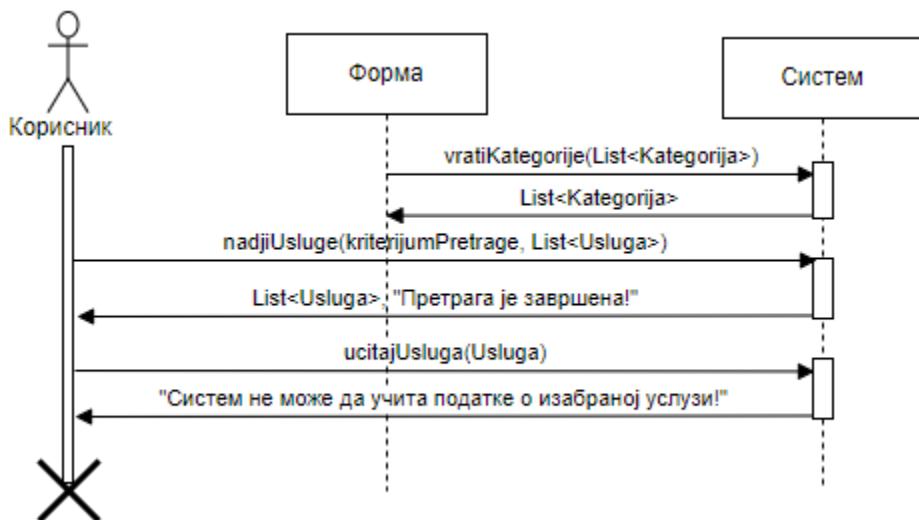
Слика 36 - ДС Измена података о услуги АС1

4.2. Уколико систем не може да нађе услугу он приказује кориснику поруку: "Ниједна услуга није пронађена!". Прекида се извршење сценарија. (ИА)



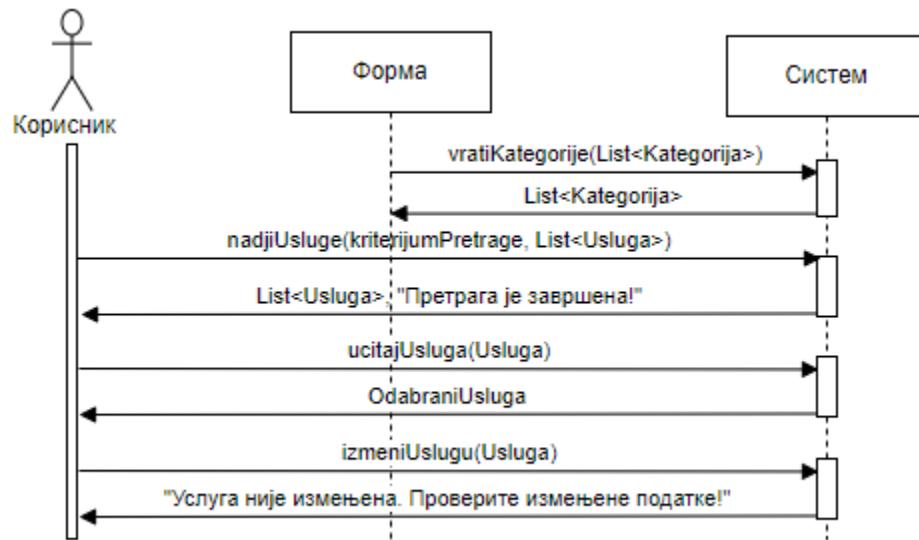
Слика 37 - ДС Измена података о услуги АС2

6.1. Уколико систем не може да учита услугу он приказује кориснику поруку: "Систем не може да учита податке о изабраној услуги!". Прекида се извршење сценарија. (ИА)



Слика 38 - ДС Измена података о услуги АС3

8.1. Уколико систем не може да запамти податке о услуги он приказује кориснику поруку: "Услуга није изменењена. Проверите изменењене податке!". (ИА)



Слика 39 - ДС Измена података о услуги АС4

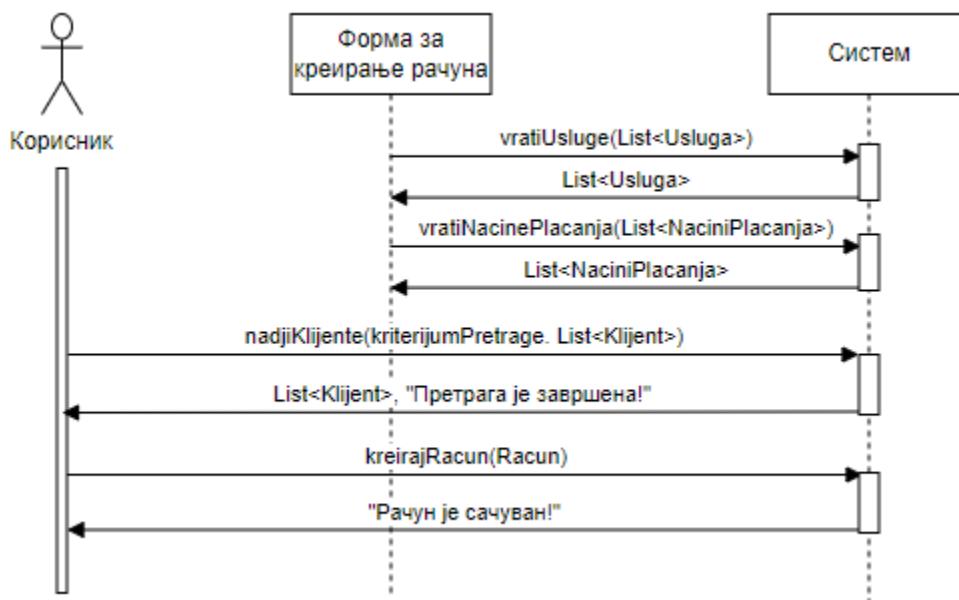
Идентификоване системске операције:

1. signal **VrsitiKategorije (List<Kategorija>)**
2. signal **NadjUsluge (kriterijumPretrage, List<Usluga>)**
3. signal **UcitajUslugu (Usluga)**
4. signal **IzmeniUslugu(Usluga)**

ДС7: Креирање рачуна

Основни сценарио

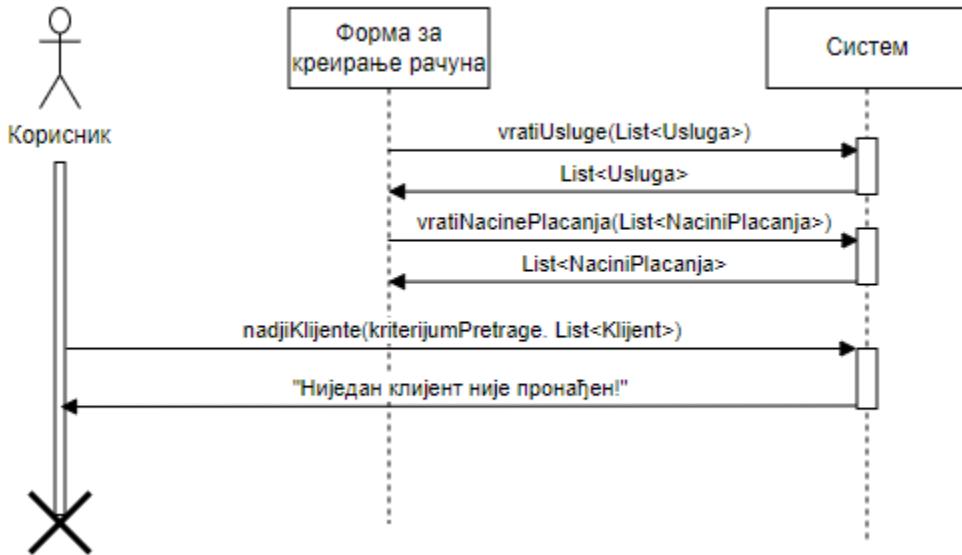
1. Форма позива систем да учита податке о услугама. (АПСО)
2. Систем приказује на форми листу услуга. (ИА)
3. Форма позива систем да учита податке о начинима плаќања. (АПСО)
4. Систем приказује на форми листу начина плаќања. (ИА)
5. Корисник позива систем да нађе клијенте по задатој вредности. (АПСО)
6. Систем приказује кориснику листу клијената и поруку: "Претрага је завршена!". (ИА)
7. Корисник позива систем да запамти податке о рачуну. (АПСО)
8. Систем приказује кориснику поруку: "Рачун је сачуван!". (ИА)



Слика 40- ДС Креирај рачун ОС

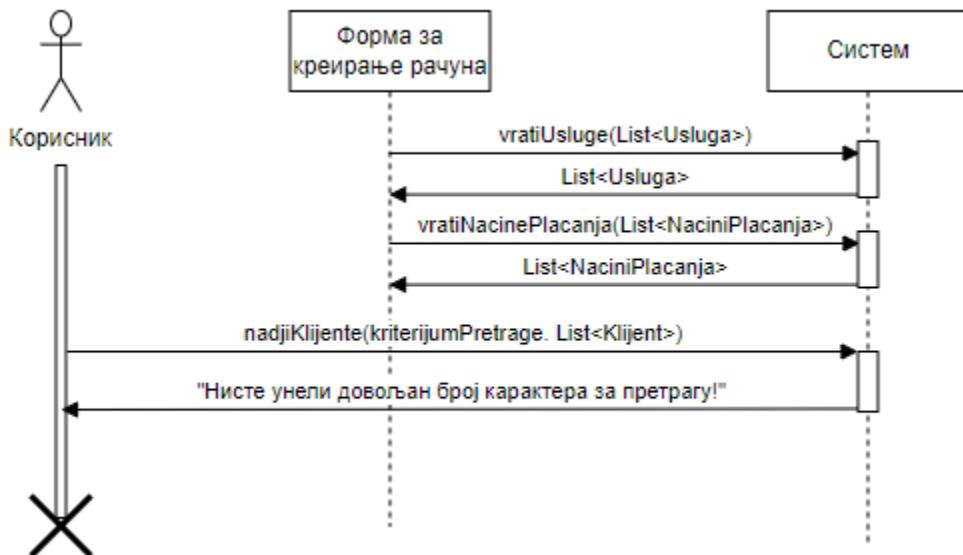
Алтернативна сценарија

- 6.1. Уколико систем не може да нађе клијенте он приказује кориснику поруку: "Ниједан клијент није пронађен!". Прекида се извршење сценарија. (ИА)



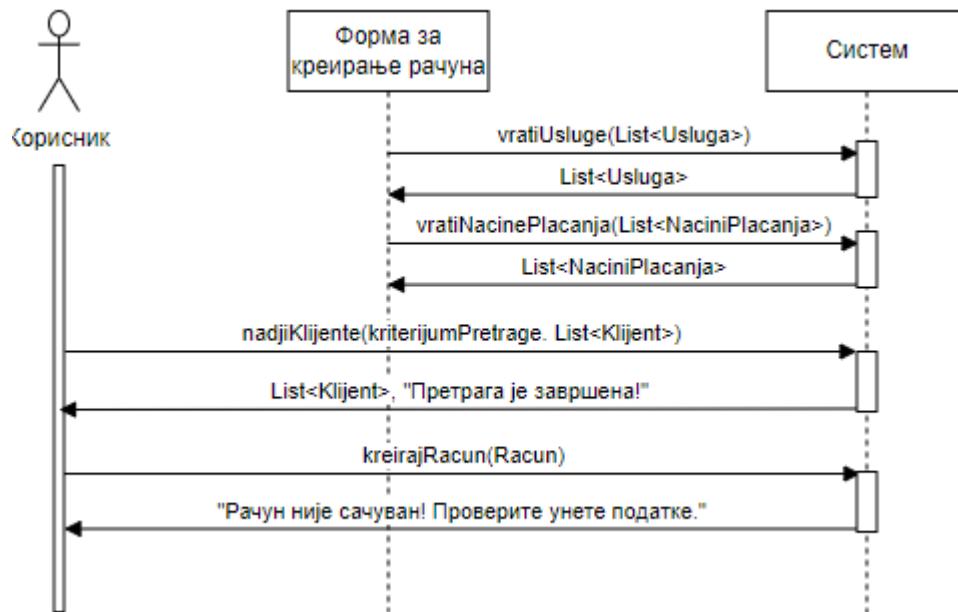
Слика 41 - ДС Креирај рачун АС1

6.2. Уколико корисник није унео довољан број карактера за претрагу клијената систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



Слика 42 - ДС Креирај рачун АС2

8.1. Уколико систем не може да запамти податке о рачуну он приказује корисику поруку: "Рачун није сачувана. Проверите унете податке!". (ИА)



Слика 43 - ДС Креирај рачун АСЗ

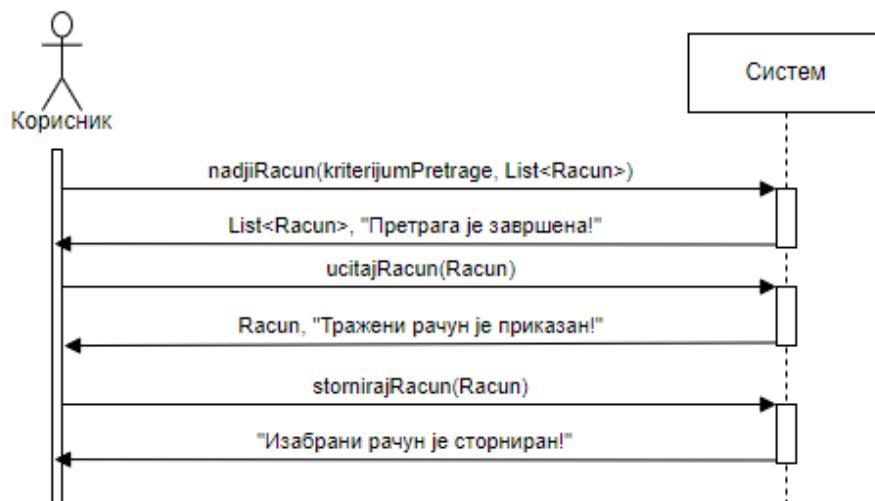
Идентификовање системских операција:

1. signal **VratiUsluge(List<Usluga>)**
2. signal **VratiNacinePlacanja(List<NaciniPlacanja>)**
3. signal **Nadjiklijente(kriterijumPretrage, List<Klijent>)**
4. signal **KreirajRacun(Racun)**

ДС8: Сторнирање рачуна

Основни сценарио

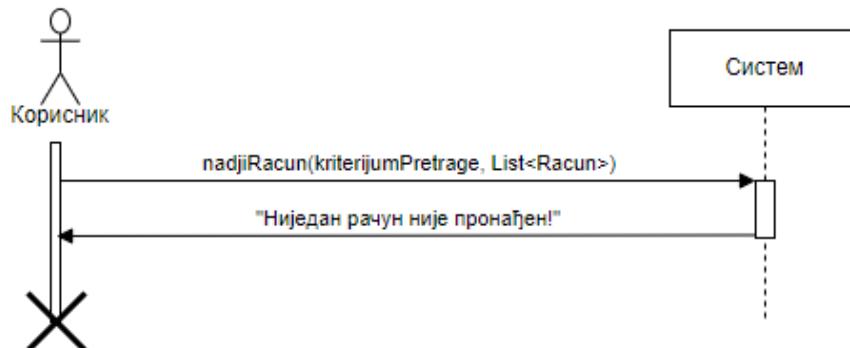
1. Корисник позива систем да нађе рачуне по задатој вредности. (АПСО)
2. Систем приказује кориснику листу рачуна и поруку: "Претрага је завршена!". (ИА)
3. Корисник позива систем да учита податке о одабраном рачуну. (АПСО)
4. Систем приказује кориснику податке о рачуну и исписује поруку: "Тражени рачун је приказан!". (ИА)
5. Корисник позива систем да сторнира рачун. (АПСО)
6. Систем приказује кориснику поруку: "Изабрани рачун је сторниран!". (ИА)



Слика 44 - ДС Сторнирај рачун ОС

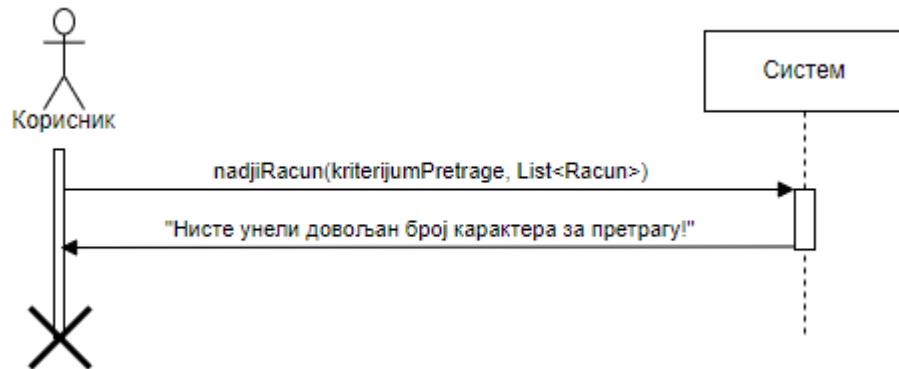
Алтернативна сценарија

- 2.1. Уколико систем не може да нађе рачуне он приказује кориснику поруку: "Ниједан рачун није пронађен!". Прекида се извршење сценарија. (ИА)



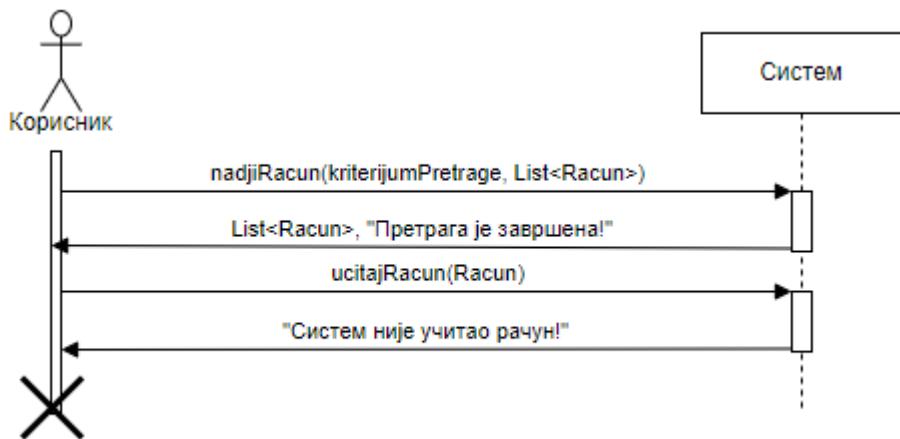
Слика 45 - ДС Сторнирај рачун АС1

2.2. Уколико корисник није унео довољан број карактера за претрагу рачуна систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



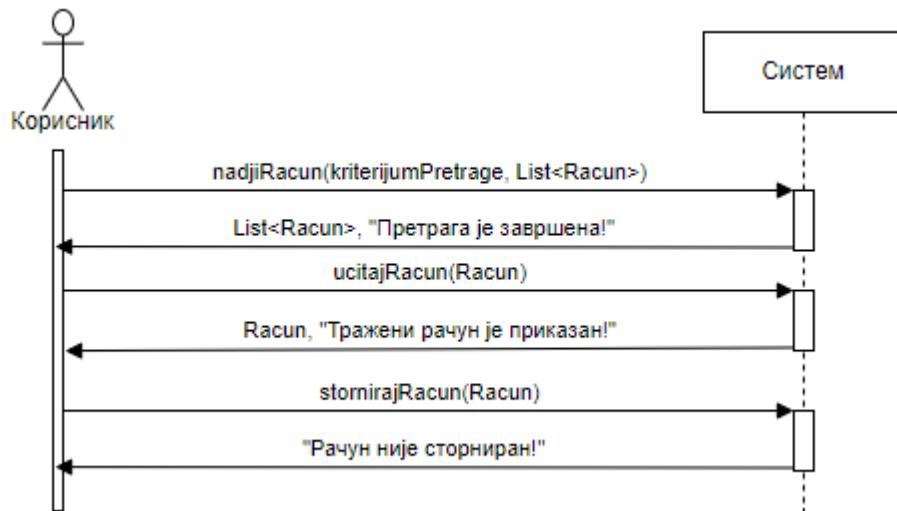
Слика 46 - ДС Сторнирај рачун АС2

4.1. Уколико систем не може да учита рачун, он приказује кориснику поруку: "Систем није учитао рачун!". Прекида се извршење сценарија. (ИА)



Слика 47 - ДС Сторнирај рачун АС3

6.1. Уколико систем не може да сторнира рачун он приказује кориснику поруку: "Рачун није сторниран!". (ИА)



Слика 48 - ДС Сторнирај рачун АС4

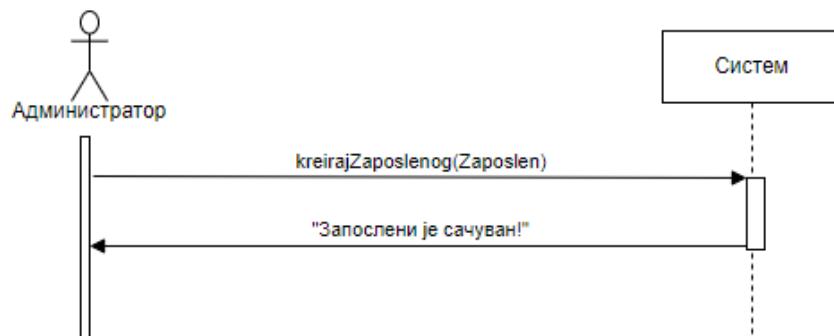
Идентификовање системских операција:

1. signal **NadjRacun**(kriterijumPretrage, List<Racun>)
2. signal **UcitajRacun** (Racun)
3. signal **StornirajRacun** (Racun)

ДС9: Унос новог запосленог

Основни сценарио

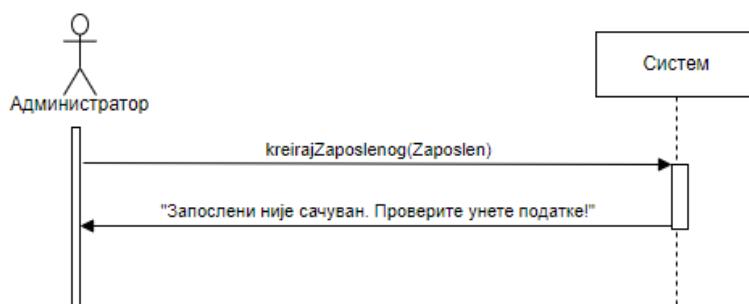
1. Администратор позива систем да запамти податке о запосленом. (АПСО)
2. Систем приказује кориснику поруку: "Запослени је сачуван!". (ИА)



Слика 49 - ДС Унос новог запосленог ОС

Алтернативна сценарија

- 2.1. Уколико систем не може да запамти податке о запосленом он приказује администратору поруку: "Запослени није сачуван. Проверите унете податке!". (ИА)



Слика 50 - ДС Унос новог запосленог АС1

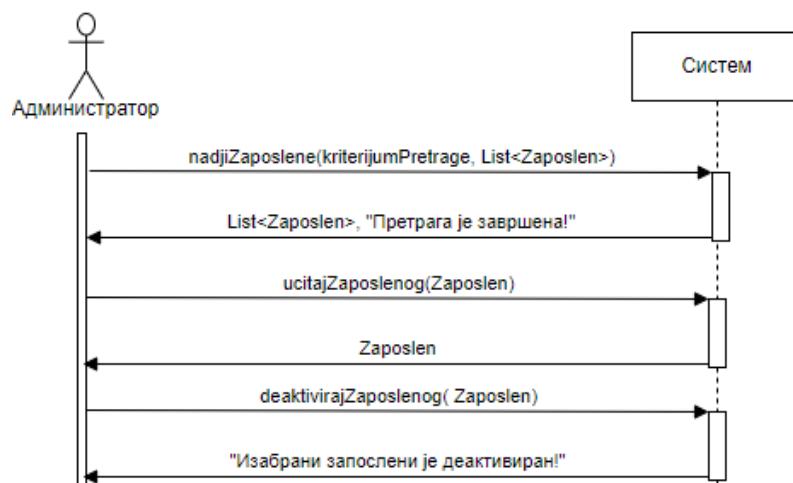
Идентификовање системских операција:

1. signal KreirajZaposlenog (Zaposlen)

ДС10: Деактивирање запосленог

Основни сценарио

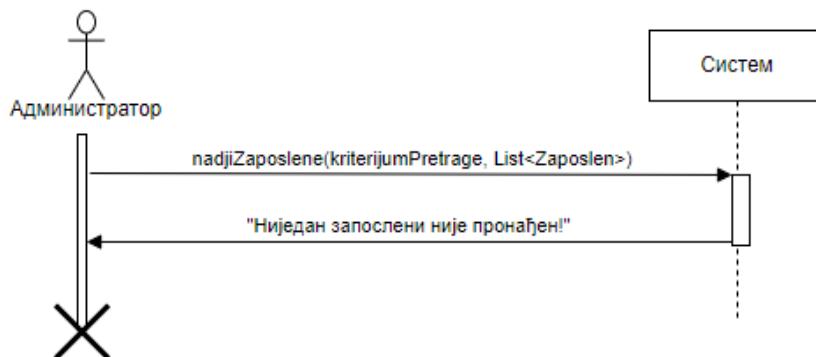
1. Администратор позива систем да нађе запослене по задатој вредности. (АПСО)
2. Систем приказује администратору листу запослених и поруку: "Претрага је завршена!". (ИА)
3. Администратор позива систем да учита податке о одабраном запосленом. (АПСО)
4. Систем приказује администратору податке о изабраном запосленом. (ИА)
5. Администратор позива систем да деактивира запосленог. (АПСО)
6. Систем приказује администратору поруку: "Изабрани запослени је деактивиран!". (ИА)



Слика 51 - ДС Деактивирање запосленог ОС

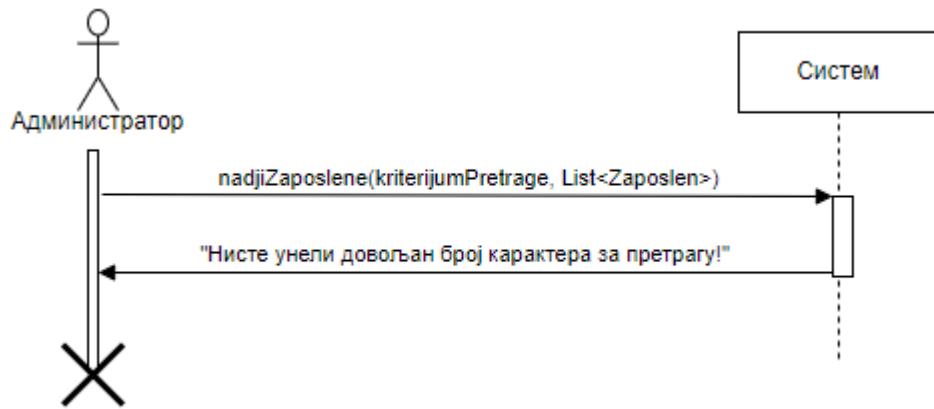
Алтернативна сценарија

- 2.1. Уколико систем не може да нађе запослене он приказује администратору поруку: "Ниједан запослени није пронађен!". Прекида се извршење сценарија. (ИА)



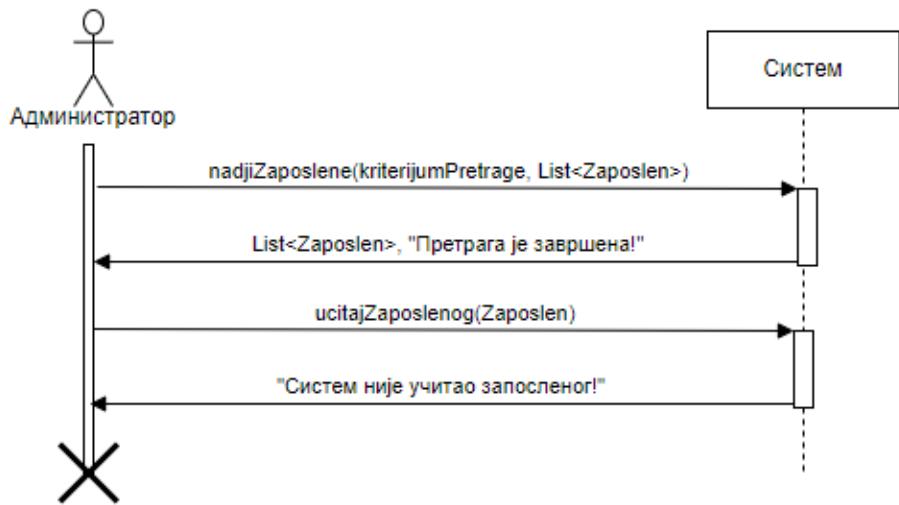
Слика 52 - ДС Деактивирање запосленог АС1

2.2. Уколико администратор није унео довољан број карактера за претрагу запослених систем приказује администратору поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



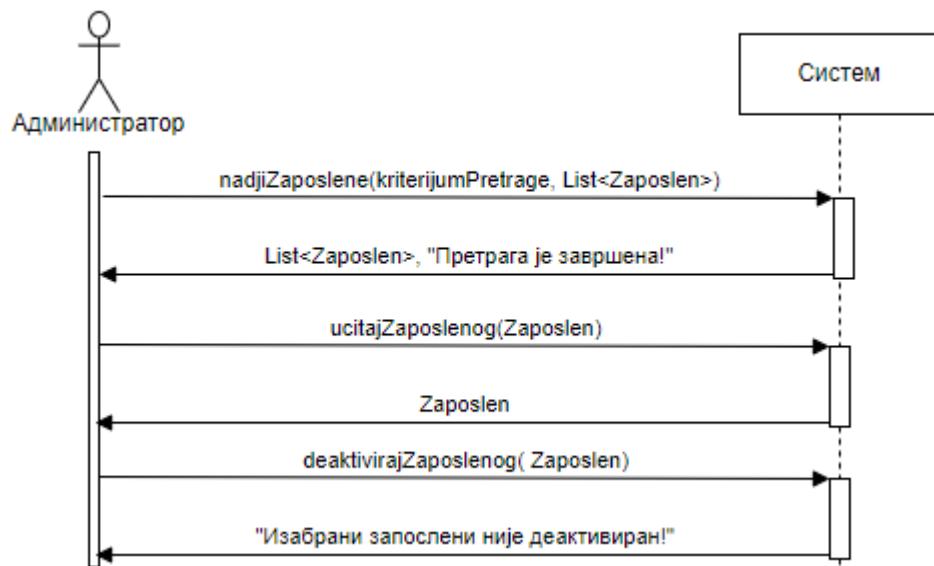
Слика 53 - ДС Деактивирање запосленог АС2

4.1. Уколико систем не може да учита запосленог, он приказује администратору поруку: "Систем није учитао запосленог!". Прекида се извршење сценарија. (ИА)



Слика 54 - ДС Деактивирање запосленог АС3

6.1. Уколико систем не може да деактивира запосленог он приказује администратору поруку: "Запослени није деактивиран!". (ИА)



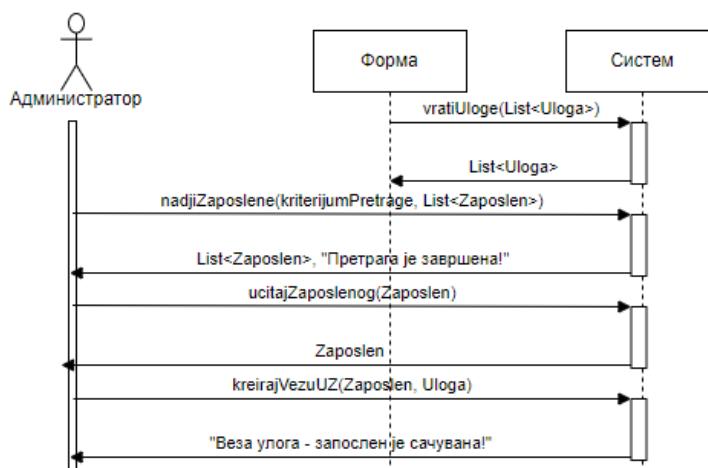
Слика 55 - ДС Деактивирање запосленог АС4

1. signal **NadjiZaposlene** (kriterijumPretrage, List<Zaposlen>)
2. signal **UcitajZaposlenog** (Zaposlen)
3. signal **DeaktivirajZaposlenog** (Zaposlen)

ДС11: Додавање улоге запосленом

Основни сценарио

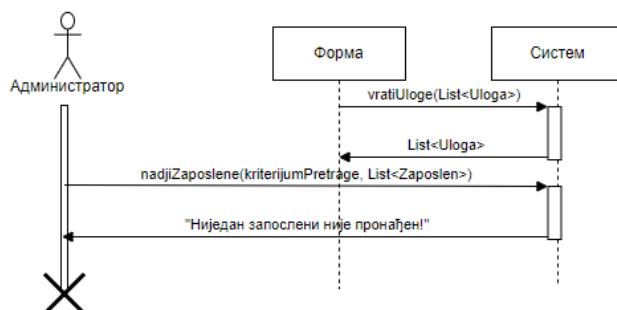
1. Форма позива систем да учита листу улога. (АПСО)
2. Систем приказује на форми листу улога. (ИА)
3. Администратор позива систем да нађе запослене по задатој вредности. (АПСО)
4. Систем приказује администратору листу запослених и поруку: "Претрага је завршена!". (ИА)
5. Администратор позива систем да учита податке о одабраном запосленом. (АПСО)
6. Систем приказује администратору податке о запосленом. (ИА)
7. Администратор позива систем да запамти податке о вези улога - запослен. (АПСО)
8. Систем приказује кориснику поруку: "Веза улога - запослен је сачувана!". (ИА)



Слика 56 - ДС Додавање улоге запосленом ОС

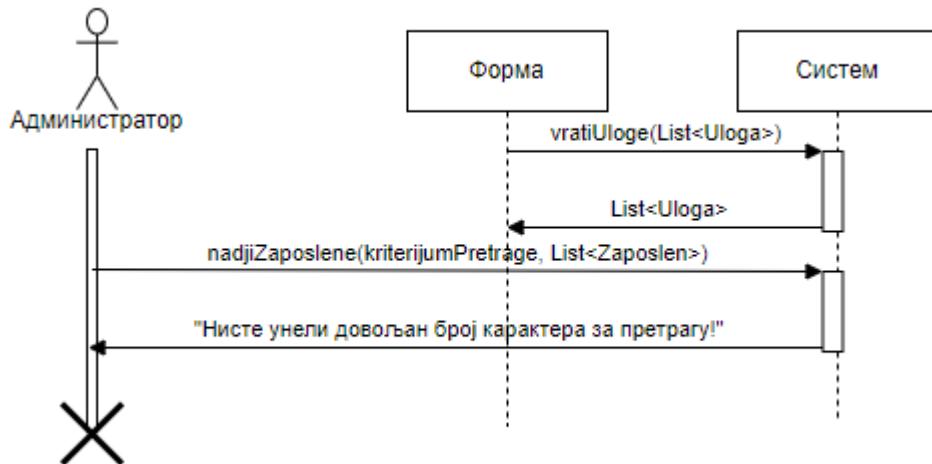
Алтернативна сценарија

- 4.1. Уколико систем не може да нађе запослене он приказује администратору поруку: "Ниједан запослени није пронађен!". Прекида се извршење сценарија. (ИА)



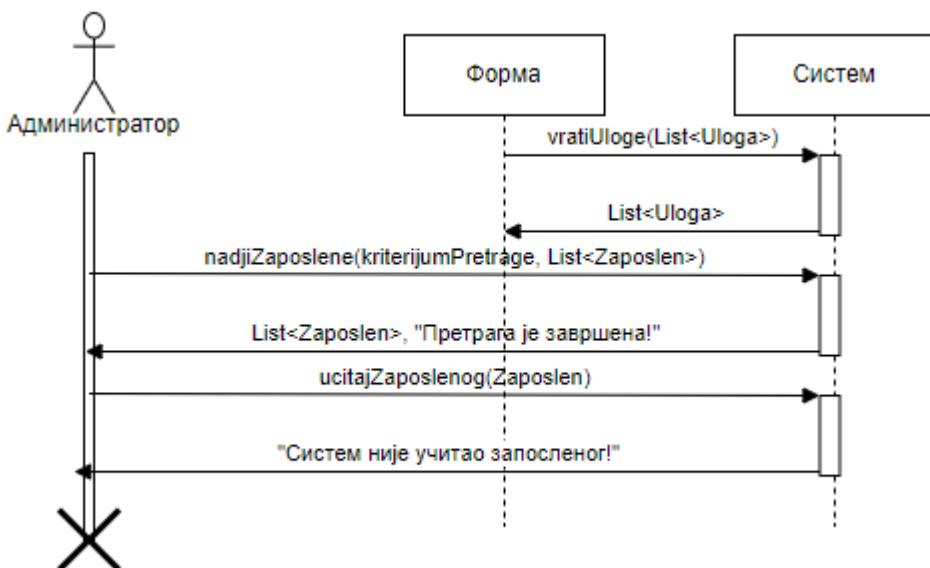
Слика 57 - ДС Додавање улоге запосленом АС1

4.2. Уколико администратор није унео довољан број карактера за претрагу запослених систем приказује администратору поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



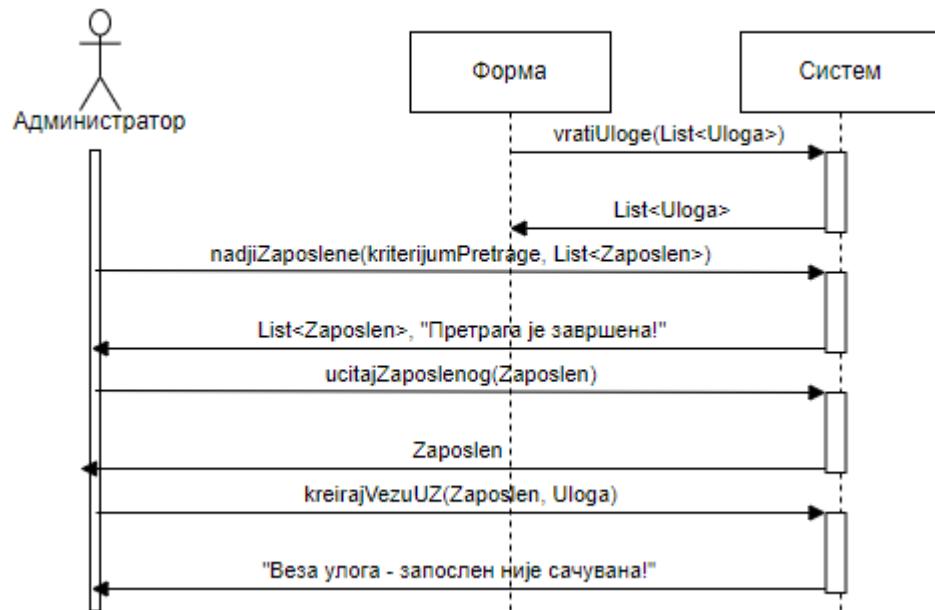
Слика 58 - ДС Додавање улоге запосленом АС2

6.1. Уколико систем не може да учита запосленог, он приказује администратору поруку: "Систем није учитао запосленог!". Прекида се извршење сценарија. (ИА)



Слика 59 - ДС Додавање улоге запосленом АС3

8.1. Уколико систем не може да запамти податке о вези улога - запослен он приказује администратору поруку: "Веза улога - запослен није сачувана. Проверите унете податке!". (ИА)



Слика 60 - ДС Додавање улоге запосленом АС4

1. signal **VratiUloge** (**List<Uloga>**)
2. signal **Nadjizaposlene** (**kriterijumPretrage, List<Zaposlen>**)
3. signal **UcitajZaposlenog** (**Zaposlen**)
4. signal **KreirajVezuUZ** (**Zaposlen, Uloga**)

Како резултат анализе сценарија добијене су 22 системских операција које треба пројектовати:

1. signal **KreirajKlijenta**(Klijent)
2. signal **VratiLoyaltyTipove**(List<Loyalty>)
3. signal **NadjiKlijente** (kriterijumPretrage, List<Klijent>)
4. signal **UcitajKlijenta** (Klijent)
5. signal **IzmeniKlijenta**(Klijent)
6. signal **VratiKategorije** (List<Kategorija>)
7. signal **KreirajUslugu**(Usluga)
8. signal **NadjiUsluge** (kriterijumPretrage, List<Usluga>)
9. signal **UcitajUslugu** (Usluga)
10. signal **IzmeniUslugu**(Usluga)
11. signal **VratiUsluge**(List<Usluga>)
12. signal **VratiNacinePlacanja**(List<NaciniPlacanja>)
13. signal **KreirajRacun**(Racun)
14. signal **NadjiRacun**(kriterijumPretrage, List<Racun>)
15. signal **UcitajRacun** (Racun)
16. signal **StornirajRacun** (Racun)
17. signal **NadjiZaposlene** (kriterijumPretrage, List<Zaposlen>)
18. signal **UcitajZaposlenog** (Zaposlen)
19. signal **KreirajZaposlenog** (Zaposlen)
20. signal **DeaktivirajZaposlenog** (Zaposlen)
21. signal **VratiUloge** (List<Uloga>)
22. signal **KreirajVezuUZ** (Zaposlen, Uloga)

3.2. Понашање софтверског система

Дефинисање уговора о системским операцијама

1. Уговор UG1: *VratiLoyaltyTipove*

Операција: *VratiLoyaltyTipove* (*List<Loyalty>*): signal;

Веза са СК: СК1, СК3

Предуслови:

Постуслови:

2. Уговор UG2: *KreirajKlijenta*

Операција: *KreirajKlijenta* (*Klijent*): signal;

Веза са СК: СК1

Предуслови: Вредносна и структурна ограничења над објектом Клијент морају бити задовољена.

Постуслови: Креиран је нови клијент.

3. Уговор UG3: *NadjiKlijente*

Операција: *NadjiKlijente* (*kriterijumPretrage, List<Klijent>*): signal;

Веза са СК: СК2, СК3, СК7

Предуслови:

Постуслови:

4. Уговор UG4: *IzmeniKlijenta*

Операција: *IzmeniKlijenta* (*Klijent*): signal;

Веза са СК: СК3

Предуслови: Вредносна и структурна ограничење над објектом Клијент морају бити задовољена.

Постуслови: Подаци о клијенту су изменењени.

5. Уговор UG5: *UcitajKlijenta*

Операција: *UcitajKlijenta* (*Klijent*): signal;

Веза са СК: CK2, CK3

Предуслови:

Постуслови:

6. Уговор UG6: *KreirajUslugu*

Операција: *KreirajUslugu* (*Usluga*): signal;

Веза са СК: CK4

Предуслови: Вредносна и структорна ограничење над објектом Услуга морају бити задовољена.

Постуслови: Креирана је нова услуга.

7. Уговор UG7: *VratiKategorije*

Операција: *VratiKategorija* (*List<Kategorija>*): signal;

Веза са СК: CK6

Предуслови:

Постуслови:

8. Уговор UG8: *NadjiUsluge*

Операција: *NadjiUsluge* (*kriterijumPretrage, List<Usluga>*): signal;

Веза са СК: CK5, CK6

Предуслови:

Постуслови:

9. Уговор UG9: *IzmeniUslugu*

Операција: *IzmeniUslugu* (*Usluga*): signal;

Веза са СК: CK6

Предуслови: Вредносна и структорна ограничење над објектом Услуга морају бити задовољена.

Постуслови: Подаци о услуги су изменењени.

10. Уговор UG10: UcitajUslugu

Операција: *UcitajUslugu (Usluga): signal;*

Веза са СК: CK5, CK6

Предуслови:

Постуслови:

11. Уговор UG11: KreirajRacun

Операција: *KreirajRacun (Racun): signal;*

Веза са СК: CK7

Предуслови: Вредносна и структорна ограничење над објектом Рачун морају бити задовољена.

Постуслови: Креиран је нови рачун.

12. Уговор UG12: VratiUsluge

Операција: *VratiUsluge (List<Usluga>): signal;*

Веза са СК: CK7

Предуслови:

Постуслови:

13. Уговор UG13: VratiNacinePlacanja

Операција: *VratiNacinePlacanja (List<NaciniPlacanja>): signal;*

Веза са СК: CK7

Предуслови:

Постуслови:

14. Уговор UG14: StornirajRacun

Операција: *StornirajRacun (Racun): signal;*

Веза са СК: CK8

Предуслови: Вредносно ограничење над објектом Рачун мора бити задовољено. Ако је рачун већ сторниран не може се извршити системска операција.

Постуслови: Рачун је сторниран.

15. Уговор UG15: *UcitajRacun*

Операција: *UcitajRacun* (*Racun*): signal;

Веза са СК: CK8

Предуслови:

Постуслови:

16. Уговор UG16: *NadjiRacun*

Операција: *NadjiRacun* (*kriterijumPretrage, List<Racun>*): signal;

Веза са СК: CK8

Предуслови:

Постуслови:

17. Уговор UG17: *KreirajZaposlenog*

Операција: *KreirajZaposlenog* (*Zaposlen*): signal;

Веза са СК: CK9

Предуслови: Вредносна и структорна ограничење над објектом Запослен морају бити задовољена.

Постуслови: Креиран је нови запослен.

18. Уговор UG18: *DeaktivirajZaposlenog*

Операција: *DeaktivirajZaposlenog* (*Zaposlen*): signal;

Веза са СК: CK10

Предуслови: Вредносно ограничење над објектом Запослен мора бити задовољено.

Постуслови: Запослен је деактивиран.

19. Уговор UG19: *UcitajZaposlenog*

Операција: *UcitajZaposlenog* (*Zaposlen*): signal;

Веза са СК: CK10, CK11

Предуслови:

Постуслови:

20. Уговор UG20: NadjiZaposlene

Операција: NadjiZaposlene (*kriterijumPretrage, List<Zaposlen>*): signal;

Веза са СК: CK10, CK11

Предуслови:

Постуслови:

21. Уговор UG21: KreirajVezuUZ

Операција: KreirajVezuUZ (*Zaposlen, Uloga*): signal;

Веза са СК: CK11

Предуслови: Вредносна и структорна ограничење над објектом ЗапосленУлога морају бити задовољена.

Постуслови: Креиран је нова веза улога – запослен.

22. Уговор UG22: VratiUloge

Операција: VratiUloge (*List<Uloga>*): signal;

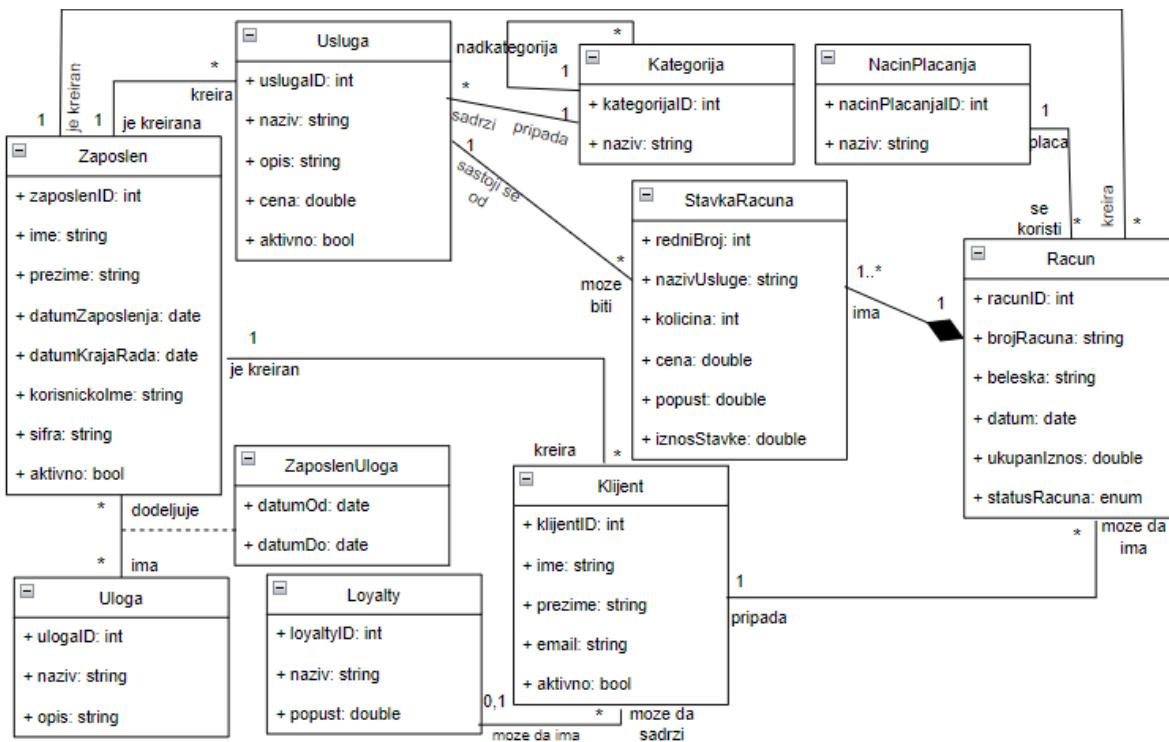
Веза са СК: CK11

Предуслови:

Постуслови:

3.3. Структура софтверског система

Концептуални модел



Слика 61 - Концептуални дијаграм класа

Табела <i>Zaposlen</i>		Просто вредносно ограничење		Сложено вредносно ограничење		Структурн о ограничењ е
Атрибути	Име	Тип атрибута	Вреднос т атрибут а	Међузависно ст атрибута једне табеле	Међузависно ст атрибута више табела	INSERT / UPDATE CASCADES <i>Racun, Usluga, Klijent, ZaposlenUloga</i> DELETE RESTRICTED <i>Racun, Usluga, Klijent, ZaposlenUloga</i>
	zaposlenID	Integer	Not null			
	ime	String				
	prezime	String				
	datumZapo sljenja	Date	Not null			
	datumKraja Rada	Date				
	korisnickoIme	String				
	sifra	String				
	aktivno	Boolean	Default: true			

Таблица 1 – Ограниченија Запослен

Табела <i>Uloga</i>		Просто вредносно ограничење		Сложено вредносно ограничење		Структурн о ограничењ е
Атрибути	Име	Тип атрибута	Вреднос т атрибут а	Међузависно ст атрибута једне табеле	Међузависно ст атрибута више табела	INSERT / UPDATE CASCADE <i>ZaposlenUloga</i> DELETE RESTRICTED <i>ZaposlenUloga</i>
	ulogalID	Integer	Not null			
	naziv	String				
	opis	String				

Таблица 2 – Ограниченија Улога

Табела <i>ZaposlenUloga</i>		Просто вредносно ограничење		Сложено вредносно ограничење		Структурно ограничење
Атрибути	Име	Тип атрибута	Вредност атрибута	Међувисно ст атрибута једне табеле	Међувисно ст атрибута више табела	INSERT / UPDATE / DELETE /
	zaposlenID	Integer	Not null			
	ulogaID	Integer	Not null			
	datumOd	Date	Not null			
	datumDo	Date				

Таблица 3 – Ограниченија ЗапосленУлога

Табела <i>Klijent</i>		Просто вредносно ограничење		Сложено вредносно ограничење		Структурно ограничење
Атрибути	Име	Тип атрибута	Вредност атрибута	Међувисно ст атрибута једне табеле	Међувисно ст атрибута више табела	INSERT RESTRICTED <i>Zaposlen,</i> <i>Loyalty</i> UPDATE CASCADE <i>Racun,</i> RESTRICTED <i>Zaposlen,</i> <i>Loyalty</i> DELETE RESTRICTED <i>Racun</i>
	klijentID	Integer	Not null			
	ime	String				
	prezime	String				
	email	String				
	aktivno	Boolean	Default: true			
	loyaltyID	Integer			Tabela:Loyalty поле:loyaltyID	
	zaposlenID	Integer	Not null		Tabela:Zaposlen поле:zaposlenID	

Таблица 4 – Ограниченија Клијент

Табела Loyalty		Просто вредносно ограничење		Сложено вредносно ограничење		Структурно ограничење
Атрибути	Име	Тип атрибута	Вредност атрибута	Међузависност атрибута једне табеле	Међузависност атрибута више табела	INSERT / UPDATE CASCADE <i>Klijent</i> DELETE RESTRICTED <i>Klijent</i>
	loyaltyID	Integer	Not null			
	naziv	String				
	popust	Double	>=0, default:0			

Таблица 5 – Ограниченија Лојалти

Табела Usluga		Просто вредносно ограничење		Сложено вредносно ограничење		Структурно ограничење
Атрибути	Име	Тип атрибута	Вредност атрибута	Међузависност атрибута једне табеле	Међузависност атрибута више табела	INSERT RESTRICTED <i>Kategorija, Zaposlen</i> UPDATE CASCADE <i>StavkaRacuna, RESTRICTED Kategorija, Zaposlen</i> DELETE RESTRICTED <i>StavkaRacuna</i>
	uslugalID	Integer	Not null			
	naziv	String				
	opis	String				
	cena	Double				
	aktivno	Boolean	Default: true			
	kategorijaID	Integer	Not null		Tabela:Kategorija Polje:kategorijaID	
zaposlenID	Integer	Not null			Tabela:Zaposlen Polje:zaposlenID	

Таблица 6 – Ограниченија Услуга

Табела <i>Kategorija</i>		Просто вредносно ограничење		Сложено вредносно ограничење		Структурно ограничење
Атрибути	Име	Тип атрибута	Вредност атрибута	Међузависност атрибута једне табеле	Међузависност атрибута више табела	INSERT / UPDATE CASCADES <i>Usluga, Kategorija</i> DELETE RESTRICTED <i>Usluga, Kategorija</i>
	kategorijalID	Integer	Not null			
	naziv	String				
	nadKategorij aID	Integer				

Таблица 7- Ограничена Категорија

Табела <i>Racun</i>		Просто вредносно ограничење		Сложено вредносно ограничење		Структурно ограничење
Атрибути	Име	Тип атрибута	Вредност атрибута	Међузависност атрибута једне табеле	Међузависност атрибута више табела	INSERT RESTRICTED <i>Klijent, NačinPlaćanja, Zaposlen</i> UPDATE CASCADE <i>StavkeRacuna, StavkeRacuna, RESTRICTED Klijent, NačinPlaćanja, Zaposlen</i> DELETE RESTRICTED <i>StavkeRacuna</i>
	racunID	Integer	Not null			
	brojRacuna	String				
	beleska	String				
	datum	Date				
	ukupanIznos	Double	Default: 0		ukupanIznos= SUM(StavkaRacuna.IznosStavke)	
	statusRacuna	Enum	Obradjen, Storniran			
	klijentID	Integer			Tabela:Klijent polje:klijentID	
	nacinPlacanjaID	Integer	Not null		Tabela: NacinPlacanja polje: nacinPlacanjaID	

	zaposlenID	Integer	Not null		Tabela:Zaposlen polje:zaposlenID	
--	------------	---------	----------	--	-------------------------------------	--

Таблица 8 - Ограничења Рачун

Табела <i>StavkeRacuna</i>		Просто вредносно ограничење		Сложено вредносно ограничење		Структурно ограничење
Атрибути	Име	Тип атрибута	Вредност атрибута	Међузависност атрибута једне табеле	Међузависност атрибута више табела	
	racunID	Integer	Not null			INSERT RESTRICTED <i>Račun, Usluga</i>
	redniBroj	Integer	Not null and >0			UPDATE RESTRICTED <i>Račun, Usluga</i>
	nazivUsluge	String				DELETE /
	kolicina	Integer				
	cena	Double				
	popust	Double			Tabela:Loyalty Polje:popust	
	iznosStavke	Double		iznosStavke=cena*kolicina-cena*kolicina*popust/100		
	uslugaid	Integer	Not null		Tabela:Usluga polje:uslugaid	

Таблица 9 - Ограничења Ставка рачуна

Табела <i>NacinPlacanja</i>		Просто вредносно ограничење		Сложено вредносно ограничење		Структурно ограничење
Атрибути	Име	Тип атрибута	Вредност атрибута	Међузависност атрибута једне табеле	Међузависност атрибута више табела	INSERT / UPDATE CASCADE Racun DELETE RESTRICTED Racun
	nacinPlacanjaID	Integer	Not null			
	naziv	String				

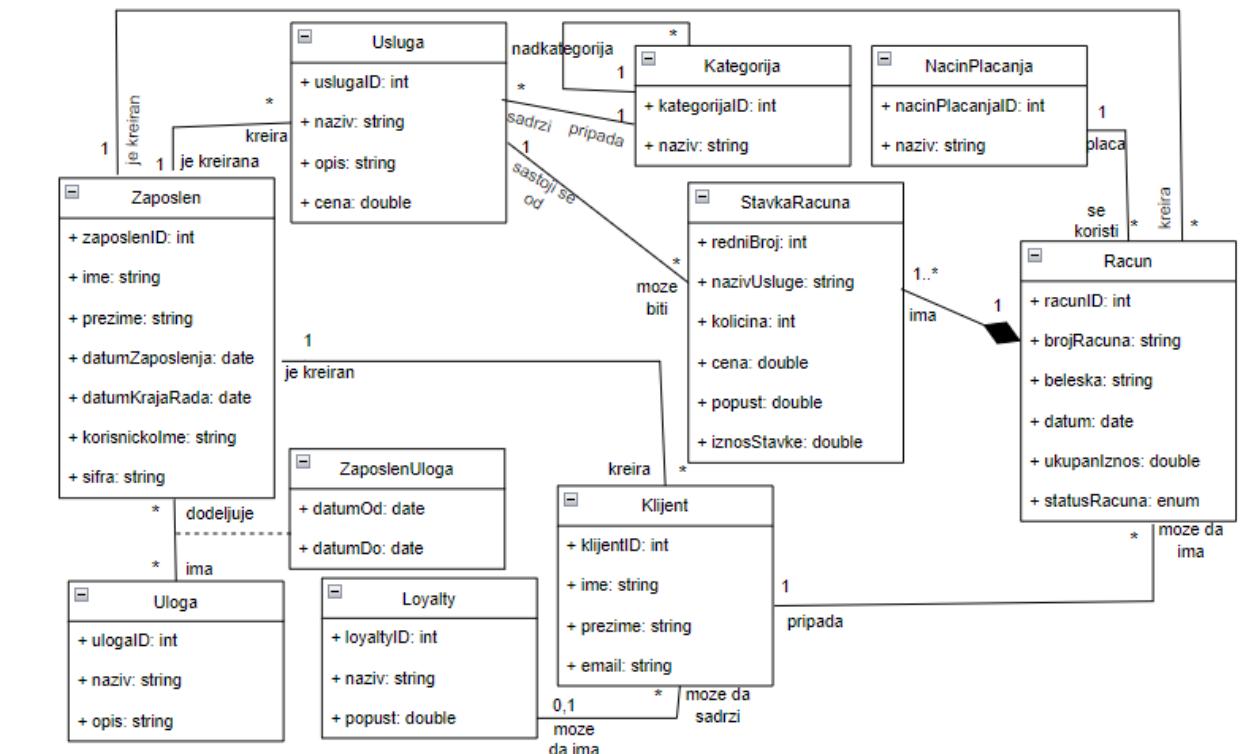
Таблица 10 - Ограниченија Начин Плаћања

Резултат анализе

Како резултат анализе сценарија СК и направљеног концептуалног модела добија се логичка структура и понашање софтверског система:

Софтверски систем

Структура система



Слика 62 - КМ структура система

Понашање система

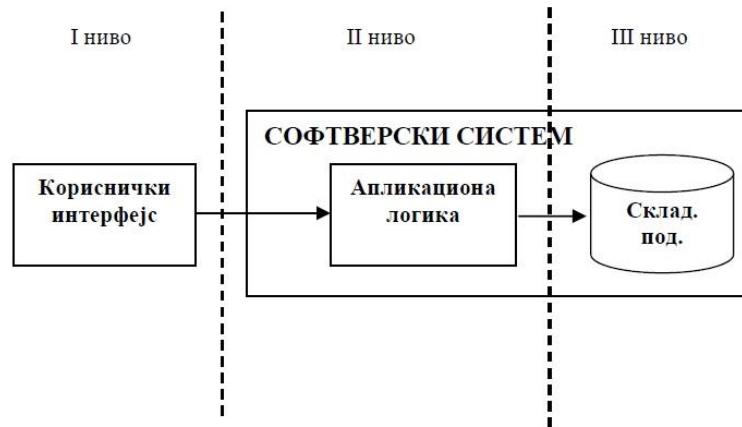
Sistemske operacije (SO)
+ VratiLoyaltyTipove(List<Loyalty>)(Korisnik): signal + KreirajKlijenta(Klijent): signal + NadjiKlijente (kriterijumPretrage, List<Klijent>): signal + IzmeniKlijenta(Klijent): signal + UcitajKlijenta (Klijent): signal + KreirajUslugu(Usluga): signal + VratiKategorija (List<Kategorija>): signal + NadjiUsluge (kriterijumPretrage, List<Usluga>): signal + IzmeniUslugu (Usluga): signal + UcitajUslugu (Usluga): signal + KreirajRacun(Racun): signal + VratiUsluge (List<Usluga>): signal + VratiNacinePlacanja(List<NaciniPlacanja>): signal + StornirajRacun (Racun): signal + UcitajRacun (Racun): signal + NadjiRacun(kriterijumPretrage, List<Racun>): signal
+KreirajZaposlenog (Zaposlen): signal +DeaktivirajZaposlenog (Zaposlen): signal +UcitajZaposlenog (Zaposlen): signal +NadjiZaposlene (kriterijumPretrage, List<Zaposlen>): signal +KreirajVezuUZ (Zaposlen, Uloga): signal +VratiUloge (List<Uloga>): signal

Слика 63 - КМ Понашање система

4. Пројектовање

У фази пројектовања описује се архитектура софтверског система. Она је тронивојска и састоји се од:

- Корисничког интерфејса
- Апликационе логике
- Складишта података



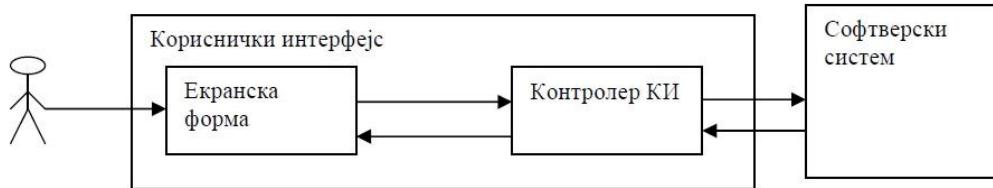
Слика 64- Тронивојска архитектура

Кориснички интерфејс се састоји од скупа екранских форми и контролера корисничког интерфејса. Апликациона логика се састоји од брокера базе података, пословне логике и контролера апликационе логике. Складиште података представља базу података или неки други вид складиштења података. Ниво корисничког интерфејса се налази на страни клијента, док се апликациона логика и складиште података налазе на страни сервера.

4.1. Пројектовање корисничког интерфејса – пројектовање екранских форми

Кориснички интерфејс представља реализацију улаза и/или излаза софтверског система. Кориснички интерфејс се састоји од:

- Екранске форме
- Контролера корисничког интерфејса



Слика 65 - Структура корисничког интерфејса

СК1: Случај коришћења – Унос новог клијента

Назив СК

Унос новог клијента

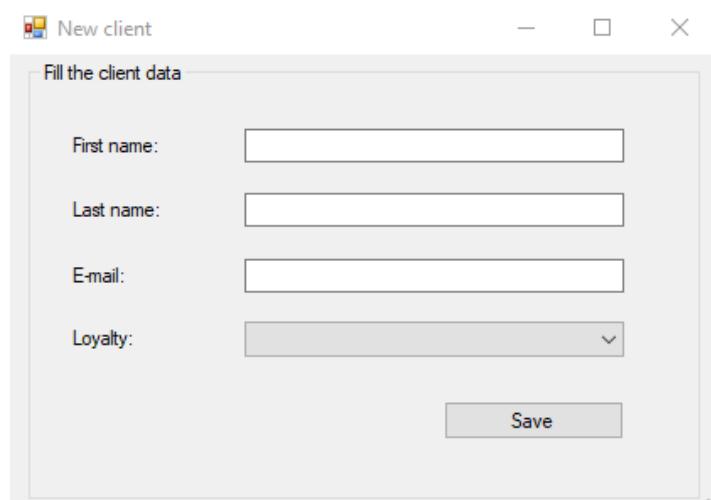
Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за унос клијента. Учитана је листа са типовима лојалти програма.

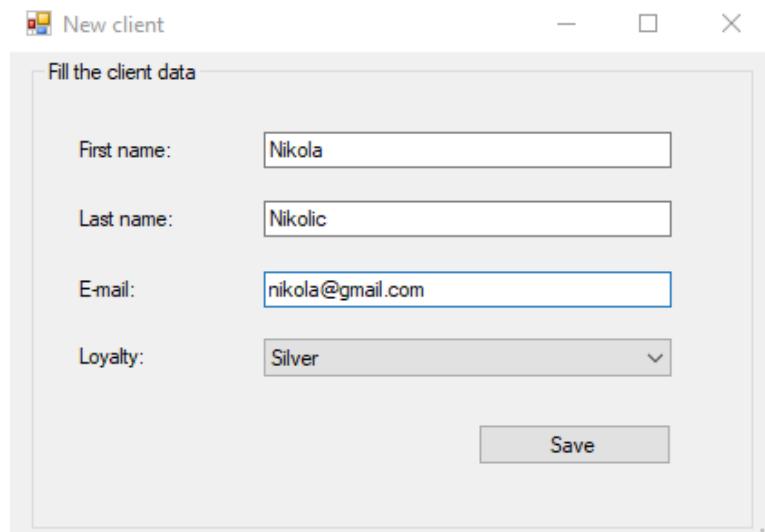


Слика 66 - Форма за унос новог клијента

Основни сценарио СК

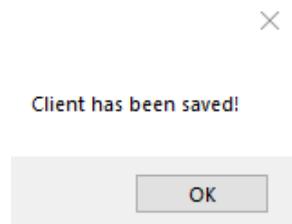
1. Корисник уноси податке о новом клијенту. (АПУСО)
2. Корисник контролише да ли је коректно унео податке о новом клијенту. (АХСО)
3. Корисник позива систем да запамти податке о клијенту. (АПСО)

Опис акције: Корисник кликом на дугме Save позива системску операцију KreirajKlijenta (Klijent) која памти новог клијенту.



Слика 67 - Унос новог клијента

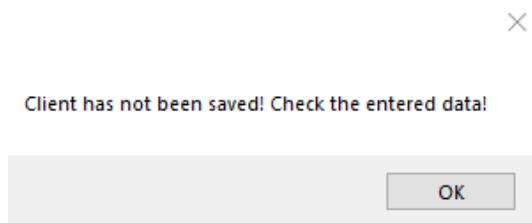
4. Систем памти податке о клијенту. (СО)
5. Систем приказује поруку: "Клијент је сачуван!". (ИА)



Слика 68 - Клијент је сачуван

Алтернативна сценарија

- 5.1. Уколико систем не може да запамти податке о клијенту он приказује кориснику поруку "Клијент није сачуван. Проверите унете податке!". (ИА)



Слика 69 - Клијент није сачуван

СК2: Случај коришћења – Претраживање клијента

Назив СК

Претраживање клијента

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са клијентима.

	First name	Last name	Email	Loyalty name
▶	Nikola	Nikolic	nikola@gmail.com	Silver
	Marko	Petrovic	marko.petrovic@gmail	
	Nevena	Mladenovic	nmladenovic@gmail.com	
	Gordana	Vukomanovic	vukomanovic.gordana70...	Gold
	Dragan	Vukomanovic	dragan@gmail.com	Silver
	Milan	Milenkovic	mm@gmail	
	Nikola	Djordjevic	nikola.djordjevic@gmail	

Слика 70 - Форма за претрагу клијената

Основни сценарио СК

1. Корисник уноси вредност по којој претражује клијенте. (АПУСО)
2. Корисник позива систем да нађе клијенте по задатој вредности. (АПСО)

Опис акције: Корисник кликом на дугме Find clients позива системску операцију NadjiKlijente (kriterijumPretrage, List<Klijent>) која приказује клијенте. Клијенте је могуће претражити по имениу, презимену, е-маилу или изабрати лојалти из падајућег менија.

Search clients

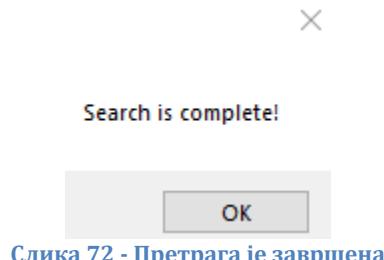
	First name	Last name	Email	Loyalty name
▶	Nikola	Nikolic	nikola@gmail.com	Silver
	Marko	Petrovic	marko.petrovic@gmail	
	Nevena	Mladenovic	nmladenovic@gmail.com	
	Gordana	Vukomanovic	vukomanovic.gordana70...	Gold
	Dragan	Vukomanovic	dragan@gmail.com	Silver
	Milan	Milenkovic	mm@gmail	
	Milica	Djordjevic	milicadjordjevic@gmail	

Enter part of first name, last name or email:

Select loyalty:

Слика 71 - Претрага клијената

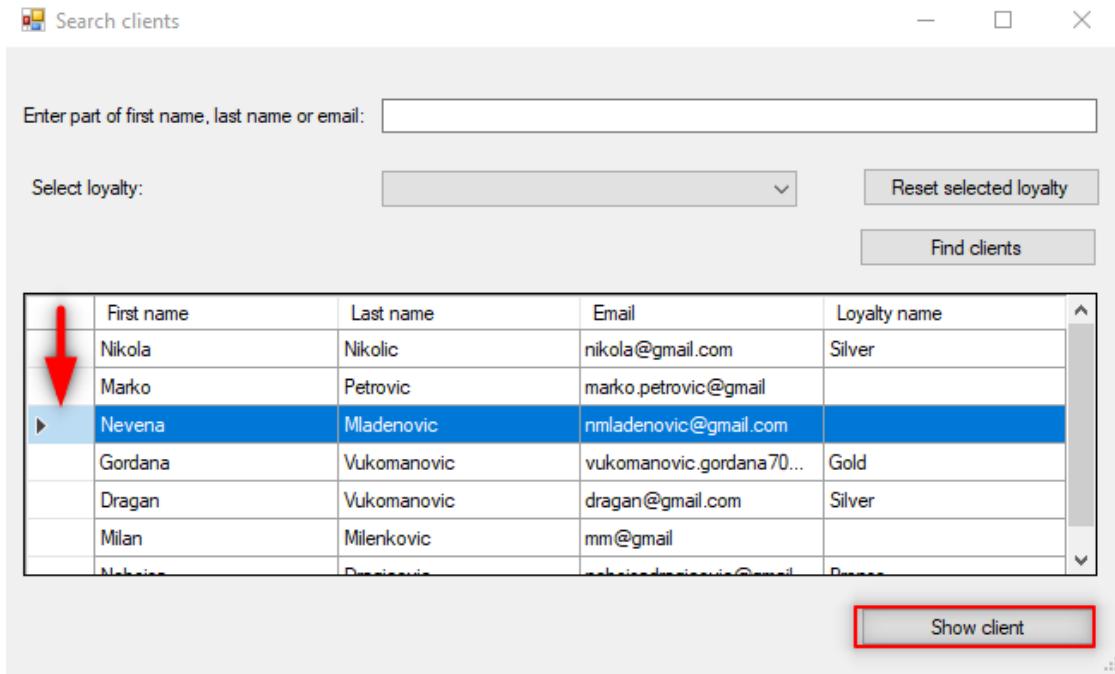
3. Систем претражује клијенте по задатој вредности. (СО)
4. Систем приказује кориснику листу клијената и поруку "Претрага је завршена!". (ИА)



Слика 72 - Претрага је завршена

5. Корисник бира једног клијента. (АПСО)

Опис акције: Корисник кликом на дугме Show clients позива системску операцију UcitajKlijenta (Klijent) која учитава податке о изабраном клијенту.



Слика 73 - Избор клијента

Show client

Cleint data

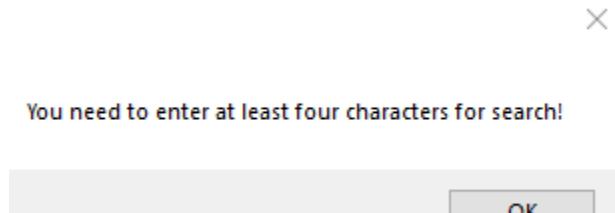
First name:	Nevena
Last name:	Mladenovic
E-mail:	nmladenovic@gmail.com
Loyalty:	Gold

Delete **Edit**

Слика 74 - Приказ података о клијенту

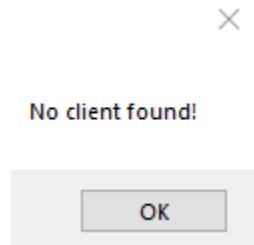
Алтернативна сценарија

- 4.1. Уколико корисник није унео довољан број карактера за претрагу систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!". (ИА)



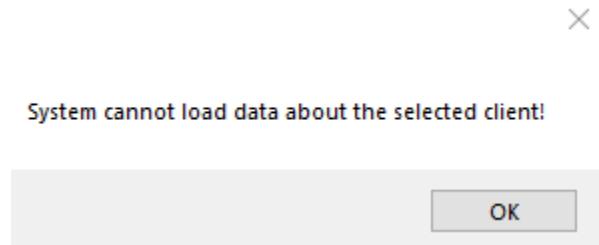
Слика 75 - Недовољно карактера за претрагу

- 4.2. Уколико систем не може да нађе клијенте систем приказује кориснику поруку: "Ниједан клијент није пронађен!" (ИА)



Слика 76 - Клијенти нису пронађени

- 7.1. Уколико систем не може да учита податке о одабраном клијенту, он приказује кориснику поруку: "Систем не може да учита податке о изабраном клијенту!".



Слика 77 - Клијент није учитан

СК3: Случај коришћења – Измена података о клијенту

Назив СК

Измена података о клијенту

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са клијентима. Учитана је листа са типовима лојалти програма.

Основни сценарио СК

1. Корисник уноси вредност по којој претражује клијенте. (АПУСО)
2. Корисник позива систем да нађе клијенте по задатој вредности. (АПСО)
3. Систем тражи клијенте по задатој вредности. (СО)
4. Систем приказује кориснику листу клијената и поруку "Претрага је завршена!". (ИА)

Search clients

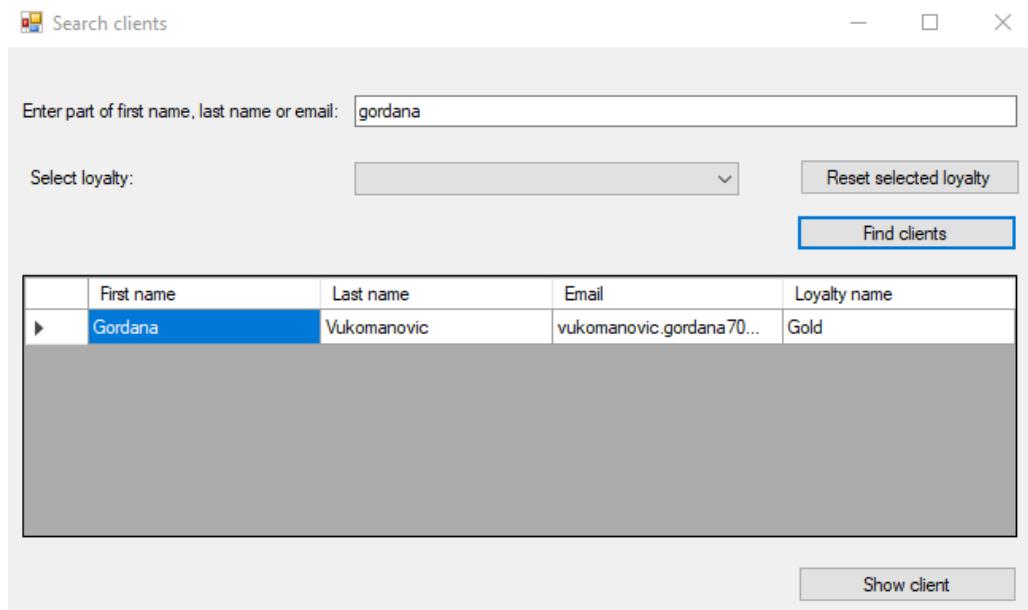
Enter part of first name, last name or email:

Select loyalty:

	First name	Last name	Loyalty name
▶	Nikola	Nikolic	Silver
	Marko	Petrovic	
	Nevena	Mladenovic	
	Gordana	Vukomanovic	vukomanovic.gordana70... Gold
	Dragan	Vukomanovic	dragan@gmail.com Silver
	Milan	Milenkovic	mm@gmail.com
	Milana	Djordjevic	milajordanovic@gmail.com

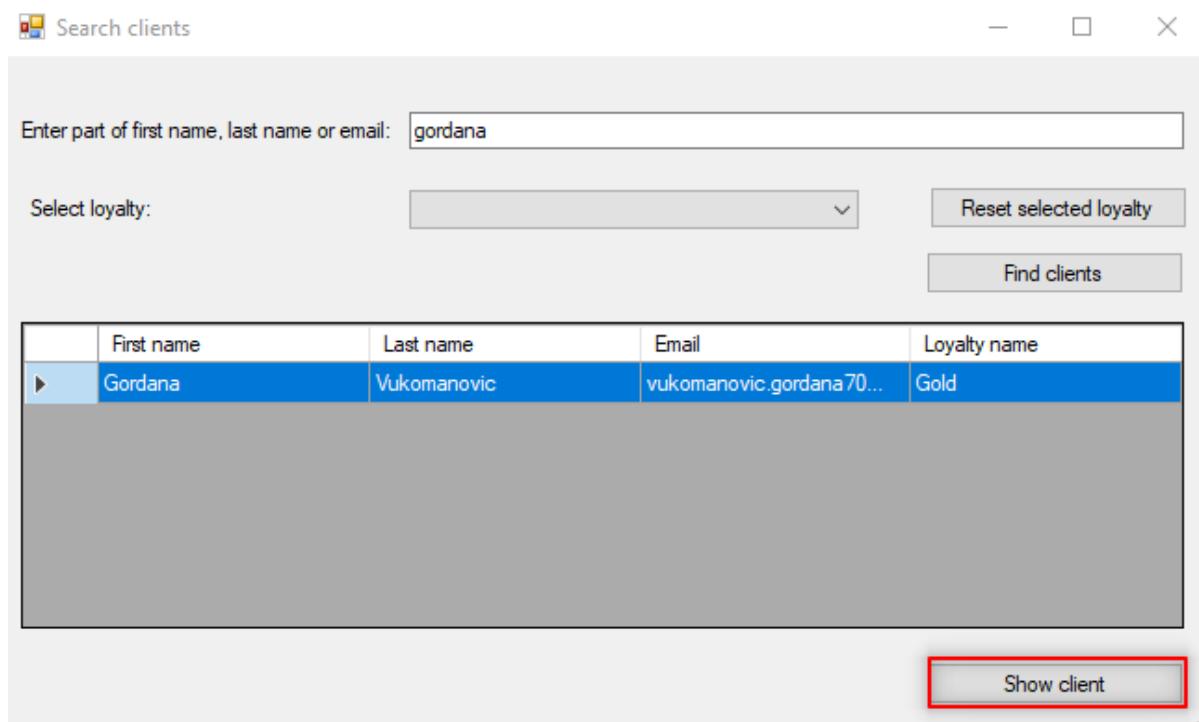
Show client

Слика 78 - Форма за претрагу клијената



Слика 79 - Приказ клијената добијених претрагом

5. Корисник бира клијента чије податке жели да измени. (АПУСО)
6. Корисник позива систем да учита податке о одабраном клијенту. (АПСО)



Слика 80 - Приказ података о изабраном клијенту

7. Систем учитава податке о одабраном клијенту. (CO)
8. Систем приказује кориснику податке о клијенту. (IA)

Client data

First name: Gordana

Last name: Vukomanovic

E-mail: vukomanovic.gordana70@gmail.com

Loyalty: Gold

Delete Edit

Слика 81 - Приказани подаци о изабраном клијенту

9. Корисник уноси (мења) податке о клијенту. (АПУСО)
10. Корисник контролише да ли је коректно унео податке о клијенту. (АНСО)
11. Корисник позива систем да запамти податке о клијенту. (АПСО)

Опис акције: Корисник кликом на дугме Edit позива системску операцију IzmeniKlijenta (Klijent) која мења податке о изабраном клијенту.

Client data

First name: Gordana

Last name: Vukomanovic

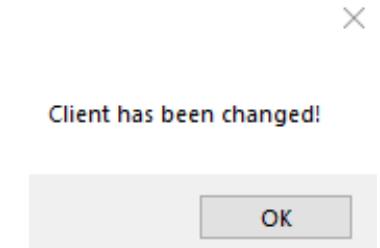
E-mail: vukomanovic.gordana70@gmail.com

Loyalty: Silver

Delete Edit

Слика 82 - Измена клијента

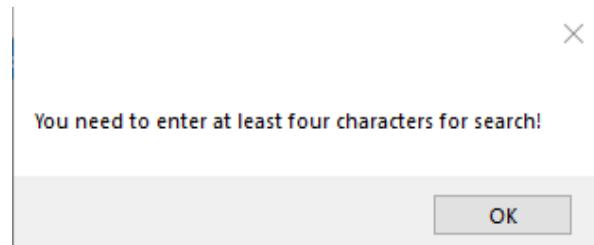
12. Систем памти податке о клијенту. (CO)
13. Систем приказује кориснику поруку: "Подаци о клијенту су изменењени!". (IA)



Слика 83 - Подаци о клијенту су изменјени

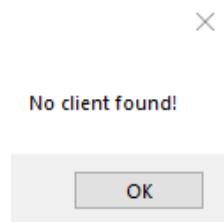
Алтернативна сценарија

4.1. Уколико корисник није унео довољан број карактера за претрагу систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



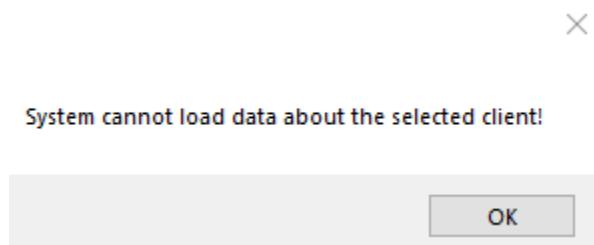
Слика 84 - Неодвољан унос карактера за претрагу

4.2. Уколико систем не може да нађе клијента он приказује кориснику поруку: "Ниједан клијент није пронађен!". Прекида се извршење сценарија. (ИА)



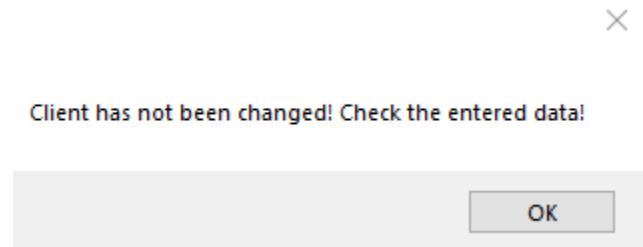
Слика 85 - Клијенти нису пронађени

8.1. Уколико систем не може да учита клијента он приказује кориснику поруку: "Систем не може да учита податке о изабраном клијенту!". Прекида се извршење сценарија. (ИА)



Слика 86 - Клијент није учитан

13.1. Уколико систем не може да запамти податке о клијенту он приказује кориснику поруку: "Клијент није изменењен. Проверите изменењене податке!". (ИА)



Слика 87 - Клијент није изменењен

СК4: Случај коришћења – Унос нове услуге

Назив СК

Унос нове услуге

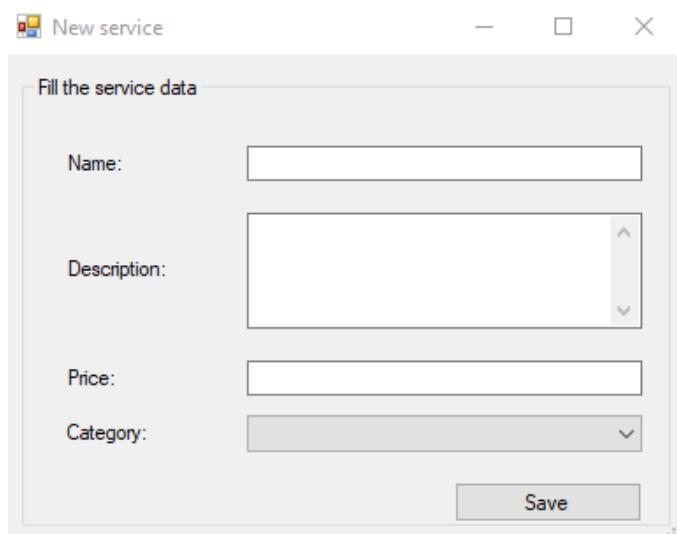
Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за унос услуге. Учитана је листа са категоријама.



The screenshot shows a Windows application window titled "New service". Inside, there's a form titled "Fill the service data". The form has four fields: "Name:" with an input box, "Description:" with a rich text area containing scroll bars, "Price:" with an input box, and "Category:" with a dropdown menu. At the bottom right is a "Save" button.

Слика 88 - Форма за унос нове услуге

Основни сценарио СК

1. Корисник уноси податке о услуги. (АПУСО)
2. Корисник контролише да ли је коректно унео податке о услуги. (АНСО)
3. Корисник позива систем да запамти податке о услуги. (АПСО)

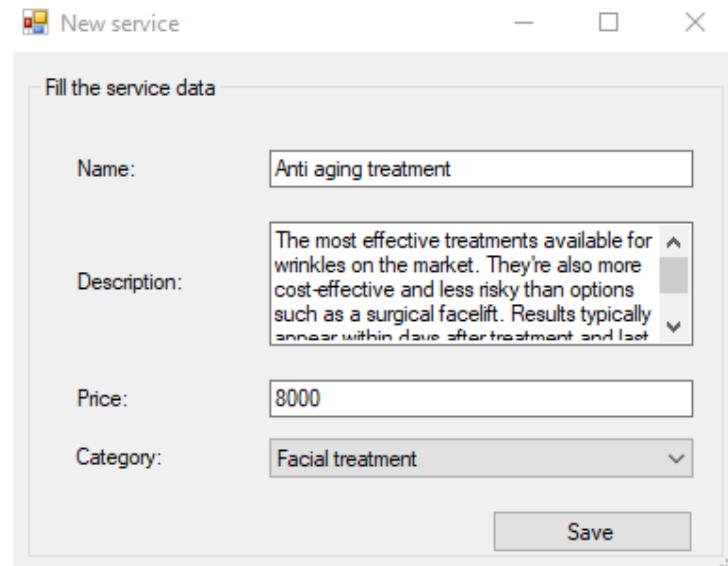
Opis akcije: Korisnik klikom na dugme Save poziva sistemsку operaciju KreirajUslugu(Usluga) koja pamti podatke o novom klijentu.

New service

Fill the service data

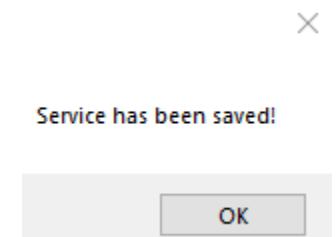
Name:	Anti aging treatment
Description:	The most effective treatments available for wrinkles on the market. They're also more cost-effective and less risky than options such as a surgical facelift. Results typically appear within days after treatment and last
Price:	8000
Category:	Facial treatment

Save



Слика 89 - Унос нове услуге

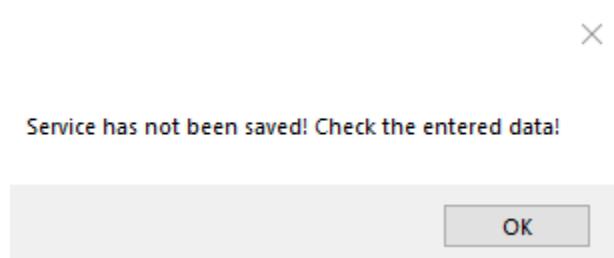
4. Систем памти податке о услуги. (СО)
5. Систем приказује кориснику поруку: "Услуга је сачуван!". (ИА)



Слика 90 - Услуга је сачувана

Алтернативна сценарија

- 4.1. Уколико систем не може да запамти податке о услуги он приказује кориснику поруку: "Услуга није сачувана. Проверите унете податке!". (ИА)



Слика 91 - Услуга није сачувана

СК5: Случај коришћења – Претраживање услуге

Назив СК

Претраживање услуге

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са услугама.

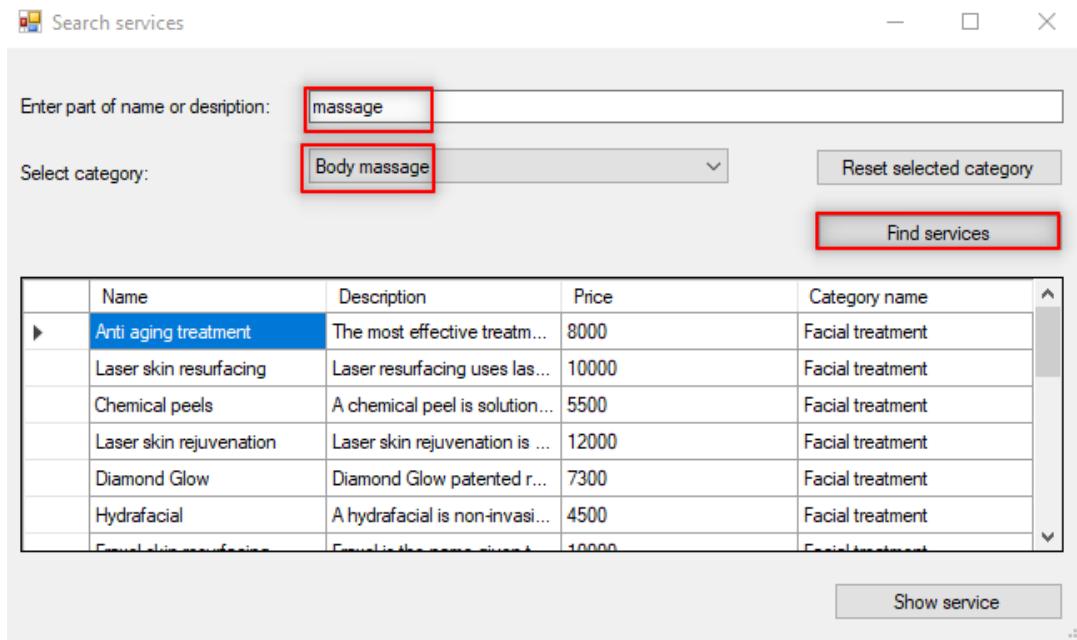
	Name	Description	Price	Category name
▶	Anti aging treatment	The most effective treatment for anti-aging... The most effective treatment for anti-aging...	8000	Facial treatment
	Laser skin resurfacing	Laser resurfacing uses lasers to remove the top layer of skin...	10000	Facial treatment
	Chemical peels	A chemical peel is a treatment that removes the top layer of skin...	5500	Facial treatment
	Laser skin rejuvenation	Laser skin rejuvenation is a procedure that uses a laser to improve the appearance of the skin...	12000	Facial treatment
	Diamond Glow	Diamond Glow patented technology...	7300	Facial treatment
	Hydrafacial	A hydrafacial is a non-invasive facial treatment...	4500	Facial treatment
	General skin treatments	General skin treatments...	10000	Facial treatment

Слика 92 - Форма за претрагу услуге

Основни сценарио СК

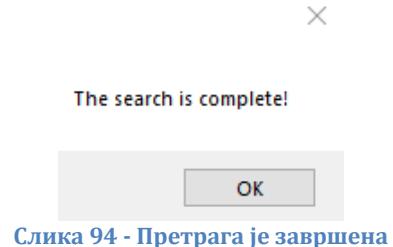
1. Корисник уноси вредност по којој претражује услуге. (АПУСО)
2. Корисник позива систем да нађе услуге по задатој вредности. (АПСО)

Опис акције: Корисник кликом на дугме Find services позива системску операцију NadjiUsluge (kriterijumPretrage, List<Usluga>) која приказује услуге.

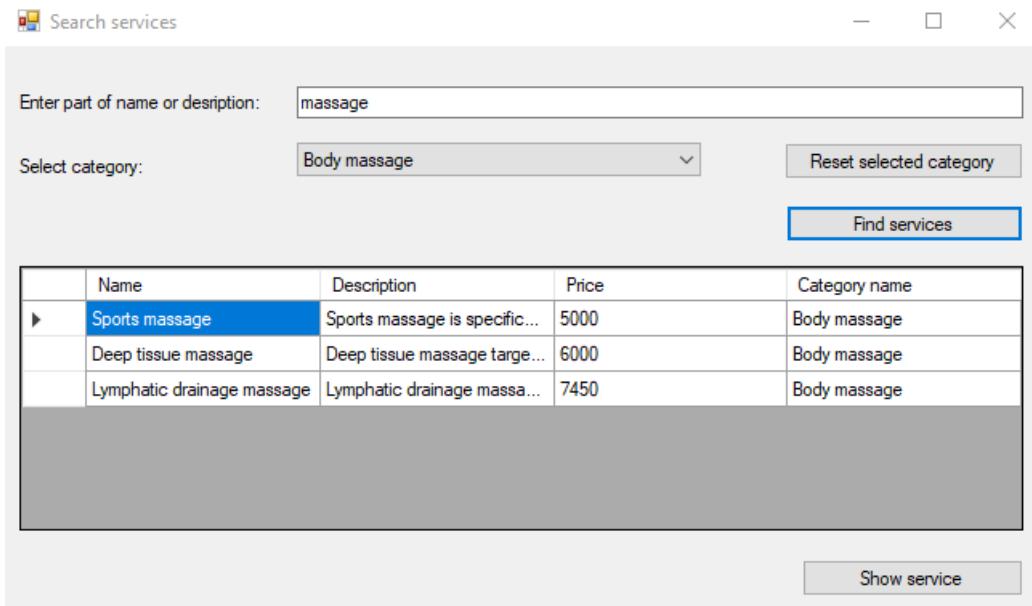


Слика 93 - Претраживање услуга

3. Систем претражује услуге по задатој вредности. (СО)
4. Систем приказује кориснику листу услуга и поруку "Претрага је завршена!". (ИА)



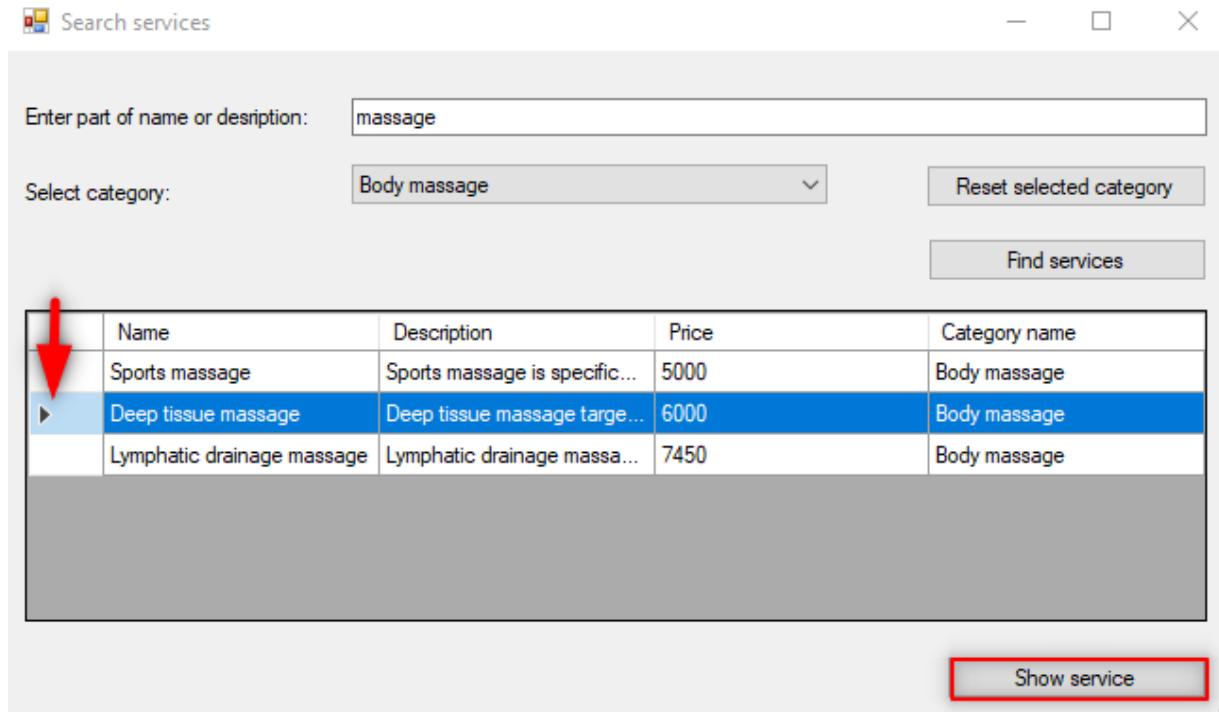
Слика 94 - Претрага је завршена



Слика 95 - Приказ улига добијених претрагом

5. Корисник бира једну услугу. (АПСО)

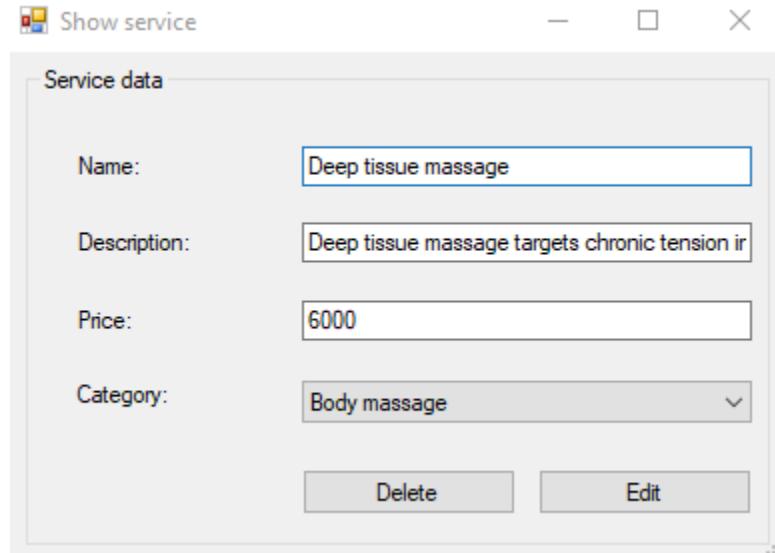
Опис акције: Корисник кликом на дугме *Show service* позива системску операцију *UcitajUslugu (Usluga)* која учитава податке о изабраној услуги.



Слика 96 - Избор услуге

6. Систем учитава податке о одабраној услуги. (СО)

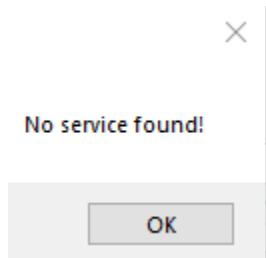
7. Систем приказује кориснику податке о одабраној услуги. (ИА)



Слика 97 - Приказ података о услуги

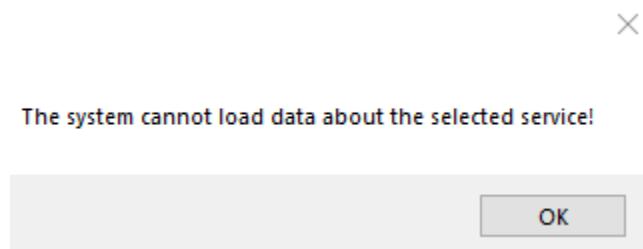
Алтернативна сценарија

4.1. Уколико систем не може да нађе услуге систем приказује кориснику поруку: "Ниједна услуга није пронађена!" (ИА)



Слика 98 - Услуга није пронађена

7.1. Уколико систем не може да учита податке о одабраној услуги, он приказује кориснику поруку: "Систем не може да учита податке о одабраној услуги!".



Слика 99 - Услуга не може да се учита

СК6: Случај коришћења – Измена података о услуги

Назив СК

Измена података о услуги

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са услугама. Учитана је листа са категоријама.

The screenshot shows a Windows-style search interface titled "Search services". It includes a search bar labeled "Enter part of name or description:" and a dropdown menu labeled "Select category:". A "Find services" button is located below these controls. The main area displays a table of service categories:

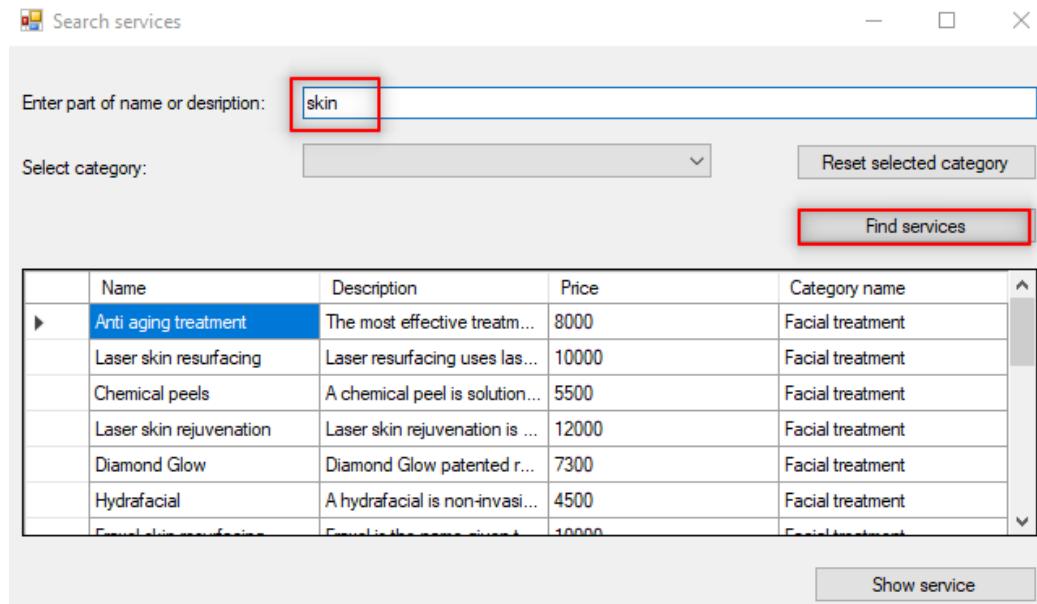
Name	Description	Price	Category name
Anti aging treatment	The most effective treatm...	8000	Facial treatment
Laser skin resurfacing	Laser resurfacing uses las...	10000	Facial treatment
Chemical peels	A chemical peel is solution...	5500	Facial treatment
Laser skin rejuvenation	Laser skin rejuvenation is ...	12000	Facial treatment
Diamond Glow	Diamond Glow patented r...	7300	Facial treatment
Hydrafacial	A hydrafacial is non-invasi...	4500	Facial treatment
General skin treatments	General skin treatments are...	10000	Facial treatment

A "Show service" button is located at the bottom right of the table area.

Слика 100 - Форма за претрагу услуга

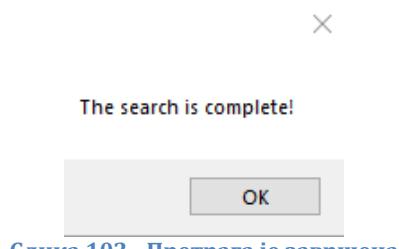
Основни сценарио СК

1. Корисник уноси вредност по којој претражује услуге. (АПУСО)
2. Корисник позива систем да нађе услуге по задатој вредности. (АПСО)

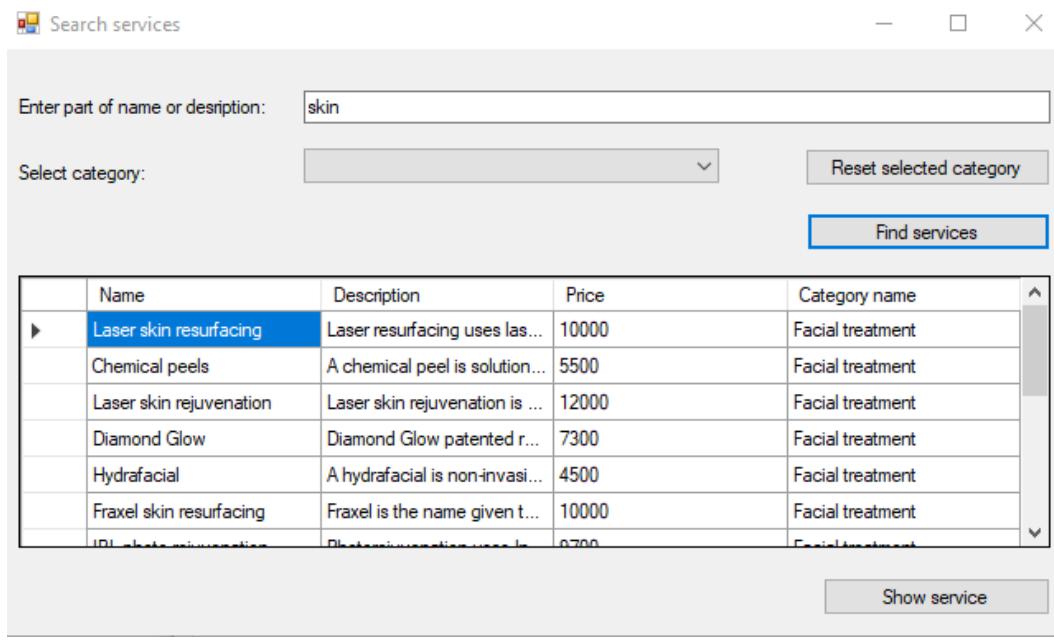


Слика 101 - Претрага услуга

3. Систем тражи услуге по задатој вредности. (СО)
4. Систем приказује кориснику листу услуга и поруку "Претрага је завршена!". (ИА)

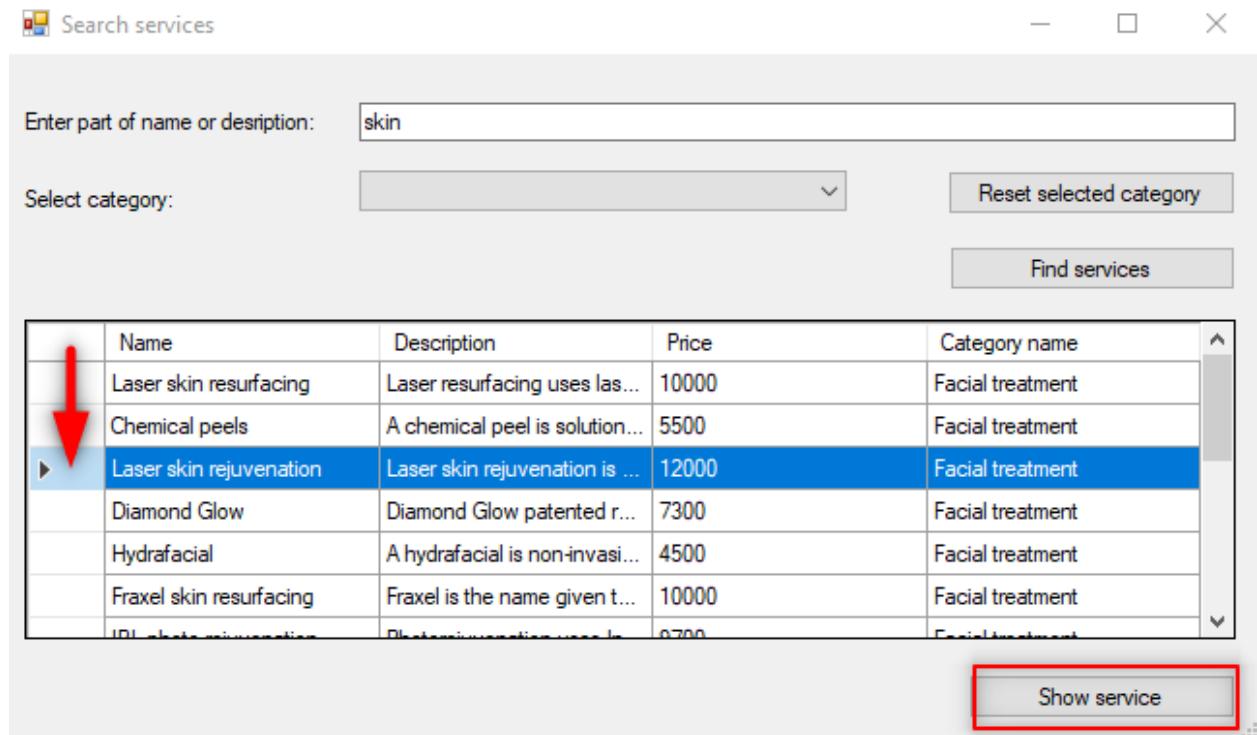


Слика 102 - Претрага је завршена



Слика 103 - Приказ услуга добијених претрагом

5. Корисник бира услугу чије податке жели да измени. (АПУСО)
6. Корисник позива систем да учита податке о одабраној услуги. (АПСО)



Слика 104 - Избор услуге

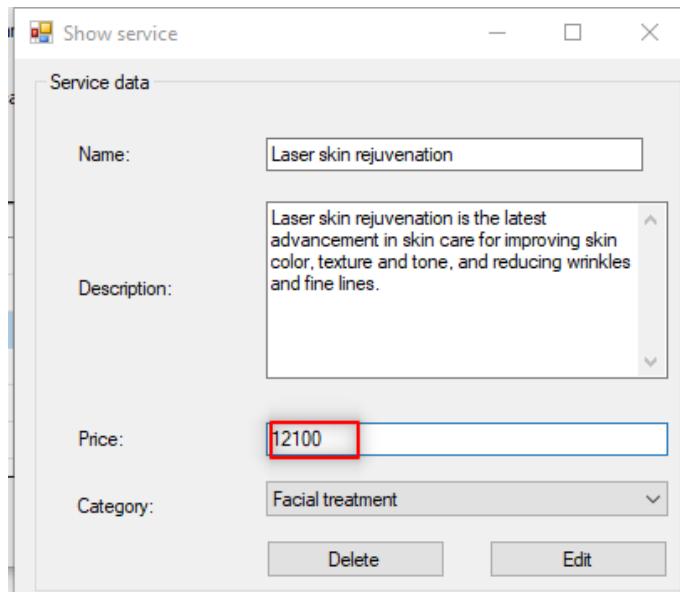
7. Систем учитава податке о одабраној услуги. (СО)
8. Систем приказује кориснику податке о услуги. (ИА)

The screenshot shows a software window titled "Show service". Inside, there is a form titled "Service data". The form has fields for "Name" (containing "Laser skin rejuvenation"), "Description" (containing a detailed text about laser skin rejuvenation), "Price" (containing "12000"), and "Category" (containing "Facial treatment"). At the bottom of the form are two buttons: "Delete" and "Edit".

Слика 105 - Приказ података о изабраној услуги

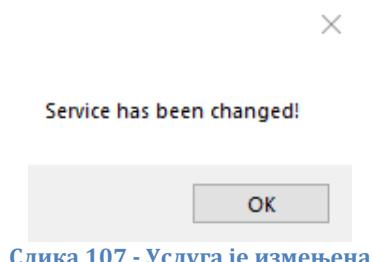
9. Корисник уноси (мења) податке о услуги. (АПУСО)
10. Корисник контролише да ли је коректно унео податке о услуги. (АНСО)
11. Корисник позива систем да запамти податке о услуги. (АПСО)

Опис акције: Корисник кликом на дугме Edit позива системску операцију IzmeniUslugu (Usluga) која мења податке о изабраној услуги.



Слика 106 - Измена услуге

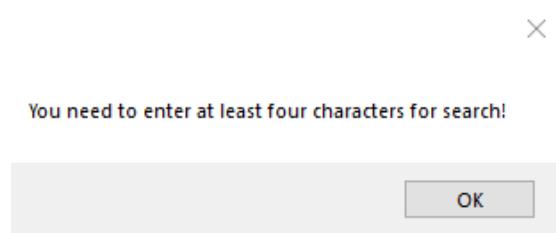
12. Систем памти податке о услуги. (СО)
13. Систем приказује кориснику поруку: "Подаци о услуги су изменењени!". (ИА)



Слика 107 - Услуга је изменењена

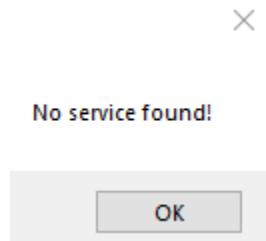
Алтернативна сценарија

- 4.1. Уколико корисник није унео довољан број карактера за претрагу систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



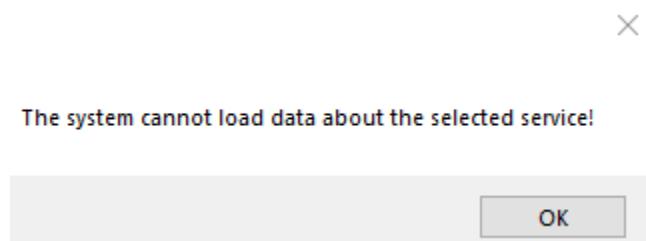
Слика 108 - Недовољан број карактера за претрагу

- 4.3. Уколико систем не може да нађе услугу он приказује кориснику поруку: "Ниједна услуга није пронађена!". Прекида се извршење сценарија. (ИА)



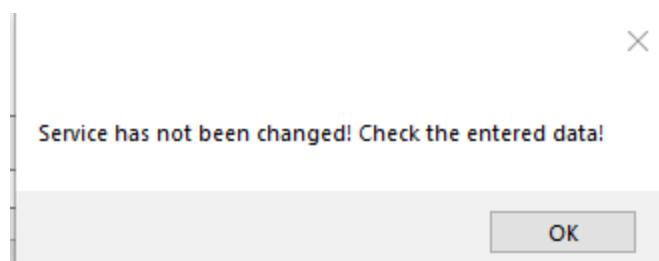
Слика 109 - Услуга није пронађена

- 8.2. Уколико систем не може да учита услугу он приказује кориснику поруку: "Систем не може да учита податке о одабраној услуги!". Прекида се извршење сценарија. (ИА)



Слика 110 - Услуга није учитана

- 13.1. Уколико систем не може да запамти податке о услуги он приказује кориснику поруку: "Услуга није изменењена. Проверите изменењене податке!". (ИА)



Слика 111 - Услуга није изменењена

СК7: Случај коришћења – Креирање рачуна

Назив СК

Креирање рачуна

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са рачунима. Листа Начини плаћања је учитана. Листа Услуге је учитана.

The screenshot shows a Windows-style application window titled "New receipt". The main area is labeled "Fill receipt data" and contains the following fields:

- Receipt number: REC2023411-8709
- Receipt date: 4/11/2023
- Payment method: (dropdown menu)
- Note: (text area)
- Add client: (button)
- Discount: (text input field)
- Amount: (text input field)
- Add receipt line: (button)
- Delete receipt line: (button)

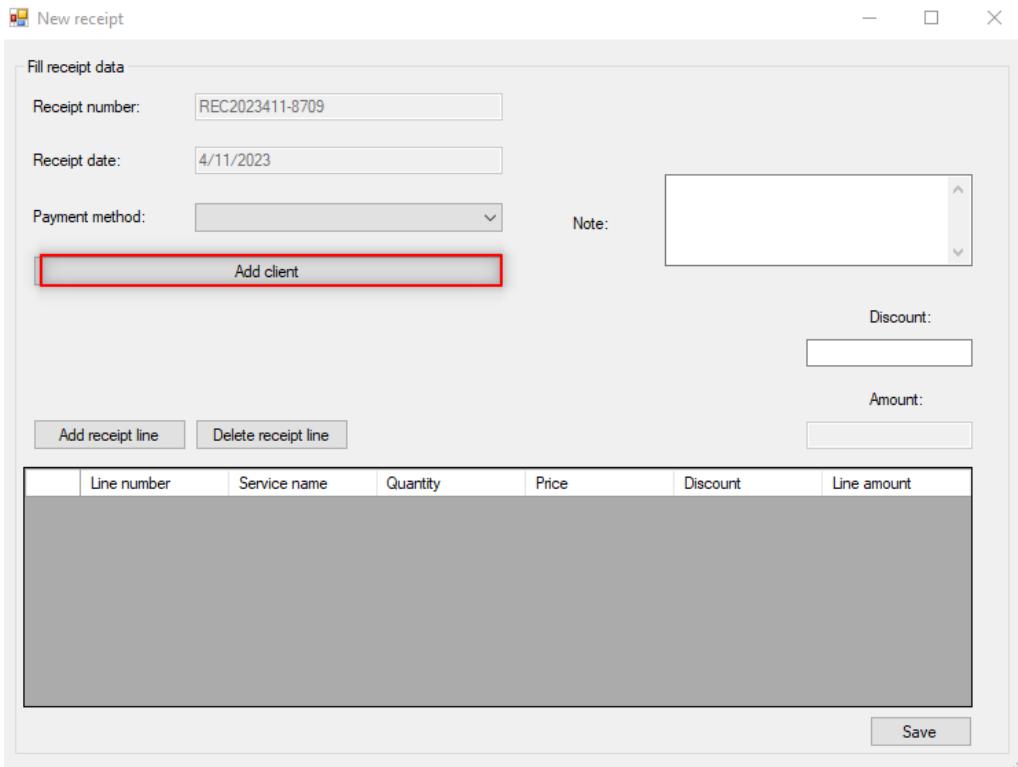
Below these controls is a table grid for entering receipt lines, with columns: Line number, Service name, Quantity, Price, Discount, and Line amount. The entire grid is currently grayed out. At the bottom right of the window is a "Save" button.

Слика 112 - Форма за унос рачуна

Основни сценарио СК

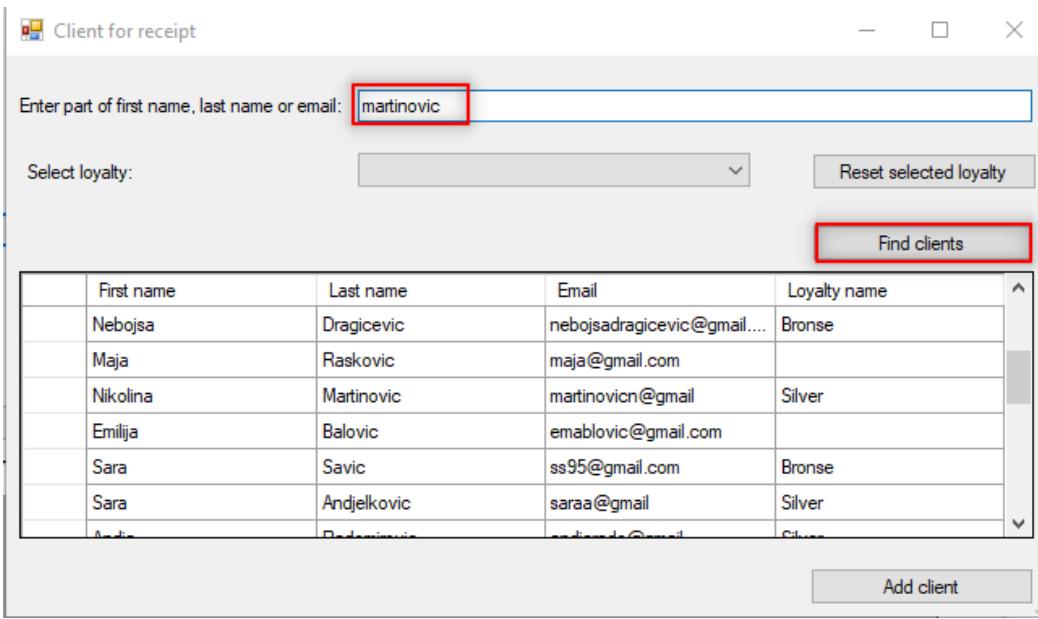
1. Корисник уноси вредност по којој претражује клијенте. (АПУСО)
2. Корисник позива систем да нађе клијенте по задатој вредности. (АПСО)

Опис акције: Корисник кликом на дугме Add client отвара форму за претрагу клијената.



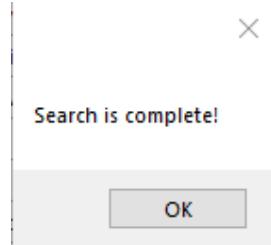
Слика 113 - Додавање клијента на рачун

Опис акције: Корисник кликом на дугме Find clients позива системску операцију NadjiKlijente (kriterijumPretrage, List<Klijent>) која приказује клијенте.



Слика 114 - Претрага клијената

3. Систем тражи клијенте по задатој вредности. (CO)
4. Систем приказује кориснику листу клијената и поруку: "Претрага је завршена!". (ИА)



Слика 115 - Претрага је завршена

5. Корисник бира клијента ког жели да дода на рачун. (АПУСО)

Опис акције: Корисник кликом на дугме Add client додаје изабраног клијента на рачун.

	First name	Last name	Email	Loyalty name
▶	Nikolina	Martinovic	martinovicn@gmail.com	Silver

Слика 116 - Приказ клијената добијених претрагом

The screenshot shows the 'New receipt' application window. In the 'Fill receipt data' section, the 'Client' field is populated with 'Nikolina Martinovic' and has a red border around it. Below the data entry fields is a table header for entering receipt lines:

Line number	Service name	Quantity	Price	Discount	Line amount
-------------	--------------	----------	-------	----------	-------------

At the bottom of the window are 'Add receipt line' and 'Delete receipt line' buttons.

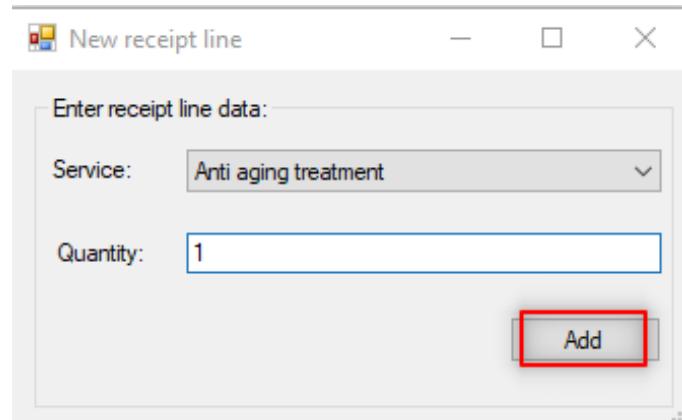
Слика 117 - Приказ клијента на рачуну

6. Корисник бира услуге које жели да дода на рачун. (АПУСО)

Опис акције: Корисник кликом на дугмe Add receipt lines отвара форму за унос услуга.

This screenshot is identical to Screenshot 117, but the 'Add receipt line' button is highlighted with a red border.

Слика 118 - Додавање услуга



Слика 119 - Унос нове услуге на рачун

Опис акције: Корисник кликом на дугме Add уноси услугу на рачун.

	Line number	Service name	Quantity	Price	Discount	Line amount
▶	1	Anti aging treatment	1	8000	8	7360

Слика 120 - Приказ услуге која је додата на рачун

7. Корисник уноси податке о рачуну. (АПУСО)
8. Корисник контролише да ли је коректно унео све потребне податке. (АХСО)
9. Корисник позива систем да запамти податке о рачуну. (АПСО)

Опис акције: Корисник кликом на дугме Save позива системску операцију operaciju KreirajRacun(Racun) која памти нов рачун.

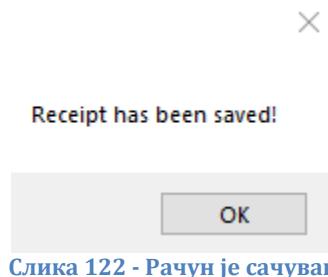
The screenshot shows a software interface titled 'New receipt'. The 'Fill receipt data' section includes fields for 'Receipt number' (REC2023411-2338), 'Receipt date' (4/11/2023), 'Payment method' (dropdown menu), 'Note' (text area), 'Client' (Nikolina Martinovic), 'Discount' (640), and 'Amount' (7360). Below this is a table with one row:

Line number	Service name	Quantity	Price	Discount	Line amount
1	Anti aging treatment	1	8000	8	7360

Buttons at the bottom include 'Add receipt line' (highlighted in blue), 'Delete receipt line', and a red-bordered 'Save' button.

Слика 121 - Унос новог рачуна

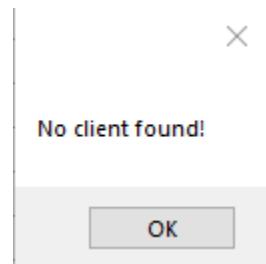
10. Систем памти податке о рачуну. (СО)
11. Систем приказује кориснику поруку: "Рачун је сачуван!". (ИА)



Слика 122 - Рачун је сачуван

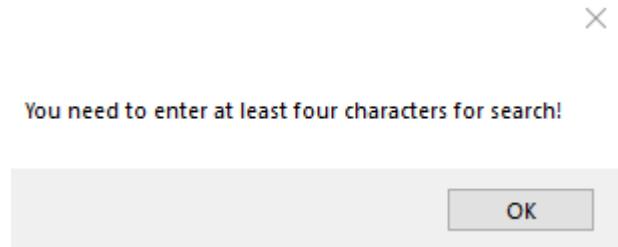
Алтернативна сценарија

- 4.1. Уколико систем не може да нађе клијенте он приказује кориснику поруку: "Ниједан клијент није пронађен!". Прекида се извршење сценарија. (ИА)



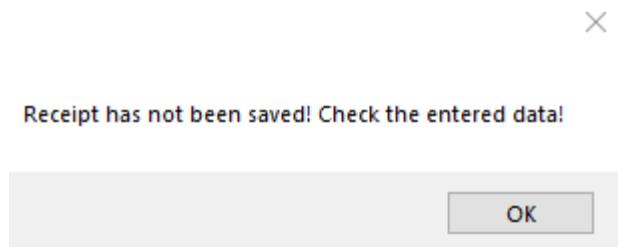
Слика 123 - Клијенти нису пронађени

4.2. Уколико корисник није унео довољан број карактера за претрагу клијената систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



Слика 124 - Недовољно карактера за претрагу

11.1. Уколико систем не може да запамти податке о рачуну он приказује корисику поруку: "Рачун није сачувана. Проверите унете податке!". (ИА)



Слика 125 - Рачун није сачуван

СК8: Случај коришћења – Сторнирање рачуна

Назив СК

Сторнирање рачуна

Актори СК

Корисник

Учесници СК

Корисник и систем (програм)

Предуслов: Систем је укључен и корисник је пријављен под својом шифром. Систем приказује форму за рад са рачунима.

The screenshot shows a Windows application window titled "Cancel receipt". At the top, there is a search bar labeled "Enter receipt number:" with an empty input field. Below it is a dropdown menu labeled "Select status:" with the value "None". To the right of the dropdown is a button "Reset selected status". A "Find receipt" button is located below these controls. The main area is a table listing seven receipts. The columns are: Receipt number, Note, Receipt date, Amount, Status, Client name, and Payment method name. The first six rows have standard blue borders, while the row for receipt "REC2023419-3367" is highlighted with a thicker blue border and has a small blue arrow icon to its left, indicating it is selected. The last row is partially visible. At the bottom right of the table area is a "Show receipt" button.

	Receipt number	Note	Receipt date	Amount	Status	Client name	Payment method name
	REC2023419-1917		4/19/2023	8000	Stomiran		Card
	REC2023419-5050	Reflexology is a t...	4/19/2023	8000	Stomiran		Card
▶	REC2023419-3367	Reflexology is a t...	4/19/2023	8000	Obradjen		Card
	REC2023419-7870	Reflexology is a t...	4/19/2023	16000	Obradjen		Card
	REC2023427-8551		4/27/2023	19400	Stomiran	Nevena Mladen...	Card
	REC2023427-1785		4/27/2023	29325	Stomiran	Marko Petrovic	Card

Слика 126 - Форма за претрагу рачуна

Основни сценарио СК

1. Корисник уноси вредност по којој претражује рачуне. (АПУСО)
2. Корисник позива систем да нађе рачуне по задатој вредности. (АПСО)

Опис акције: Корисник кликом на дугме Find receipts позива системску операцију NadjiRacun (kriterijumPretrage, List<Racun>) која приказује рачуне.

Cancel receipt

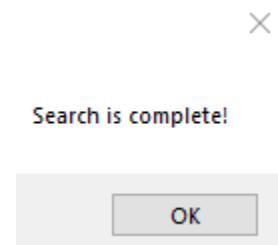
Enter receipt number:

Select status:

	Receipt number	Note	Receipt date	Amount	Status	Client name	Payment method name
	REC2023419-1917		4/19/2023	8000	Stomiran		Card
	REC2023419-5050	Reflexology is a t...	4/19/2023	8000	Stomiran		Card
▶	REC2023419-3367	Reflexology is a t...	4/19/2023	8000	Obradjen		Card
	REC2023419-7870	Reflexology is a t...	4/19/2023	16000	Obradjen		Card
	REC2023427-8551		4/27/2023	19400	Stomiran	Nevena Mladen...	Card
	REC2023427-1785		4/27/2023	29325	Stomiran	Marko Petrovic	Card

Слика 127 - Претрага рачуна

3. Систем тражи рачуне по задатој вредности. (СО)
4. Систем приказује кориснику листу рачуна и поруку: "Претрага је завршена!". (ИА)



Слика 128 - Претрага је завршена

5. Корисник бира рачун који жели да сторнира. (АПУСО)
 6. Корисник позива систем да учита податке о одабраном рачуну. (АПСО)
- Опис акције: Корисник кликом на дугме Show receipt позива системску операцију UcitajRacun(Racun).*

Enter receipt number:

Select status: Obradjen

	Receipt number	Note	Receipt date	Amount	Status	Client name	Payment method name
	REC2023419-3367	Reflexology is a ty...	4/19/2023	8000	Obradjen		Card
▶	REC2023419-7870	Reflexology is a ty...	4/19/2023	16000	Obradjen		Card

Слика 129 - Учитавање података о рачуну

7. Систем учитава податке о одабраном рачуну. (CO)
8. Систем приказује кориснику податке о рачуну и исписује поруку: "Тражени рачун је приказан!". (IA)

Receipt data

Receipt number:

Receipt date:

Payment method:

Note:

Reflexology is a type of therapy that uses gentle pressure on specific points along your feet (and possibly on your hands or ears as well) to help you feel better.
Можете унети највише 255 карактера текста.

Client:

Amount:

	Line number	Service name	Quantity	Price	Discount	Line amount
▶	1	Anti aging treatment	1	8000	0	8000
	2	Anti aging treatment	1	8000	0	8000

Слика 130 - Форма са подацима о рачуну

9. Корисник позива систем да сторнира рачун. (АПСО)

Опис акције: Корисник кликом на дугме *Cancel receipt* позива системску операцију *StornirajRacun (Racun)* која сторнира изабрани рачун.

The screenshot shows a Windows application window titled "ShowReceiptForm". Inside, there's a "Receipt data" section with fields for Receipt number (REC2023419-7870), Receipt date (4/19/2023), Payment method (Card), Client (empty), and a Note area containing text about reflexology and a character limit note. Below this is a table of receipt items:

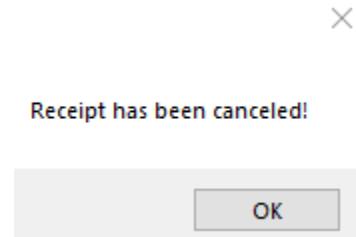
	Line number	Service name	Quantity	Price	Discount	Line amount
▶	1	Anti aging treatment	1	8000	0	8000
	2	Anti aging treatment	1	8000	0	8000

At the bottom right is a red-bordered "Cancel receipt" button.

Слика 131 - Сторнирање рачуна

10. Систем сторнира рачун. (СО)

11. Систем приказује кориснику поруку: "Изабрани рачун је сторниран! ". (ИА)



Слика 132 - Рачун је сторниран

Алтернативна сценарија

4.1. Уколико систем не може да нађе рачуне он приказује кориснику поруку: "Ниједан рачун није пронађен!". Прекида се извршење сценарија. (ИА)



No receipt found!

OK

Слика 133 - Рачуни нису пронађени

- 4.2. Уколико корисник није унео довољан број карактера за претрагу рачуна систем приказује кориснику поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



You need to enter at least four characters for search!

OK

Слика 134 - Недовољан број карактера за претрагу

- 8.1. Уколико систем не може да учита рачун, он приказује кориснику поруку: "Систем није учитао рачун!". Прекида се извршење сценарија. (ИА)



System cannot load data about the selected receipt!

OK

Слика 135 - Рачун није учитан

- 11.1. Уколико систем не може да сторнира рачун он приказује кориснику поруку: "Рачун није сторниран!". (ИА)



Receipt has not been canceled!

OK

Слика 136 - Рачун није сторниран

СК9: Случај коришћења – Унос новог запосленог

Назив СК

Унос новог запосленог

Актори СК

Администратор

Учесници СК

Администратор и систем (програм)

Предуслов: Систем је укључен и администратор је пријављен под својом шифром. Систем приказује форму за унос запосленог.

New employee

Fill the employee data

Username: _____

Password: _____

First name: _____

Last name: _____

Date of employment: Wednesday, April 12, 2023

Do you want to set termination date?

Слика 137 - Форма за унос новог запосленог

Основни сценарио СК

1. Администратор уноси податке о запосленом. (АПУСО)
2. Администратор контролише да ли је коректно унео податке о запосленом. (АНСО)
3. Администратор позива систем да запамти податке о запосленом. (АПСО)

Опис акције: Корисник кликом на дугме Save позива системску операцију KreirajZaposlenog(Zaposlen) која памти податке о новом запосленом.

Fill the employee data

Username: z005

Password: 55555

First name: Mihajlo

Last name: Milenkovic

Date of employment: Wednesday, April 12, 2023

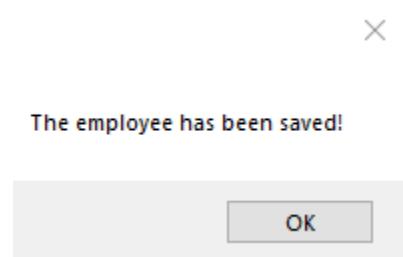
Do you want to set termination date?

Termination date: Saturday, April 12, 2025

Save

Слика 138 - Унос новог запосленог

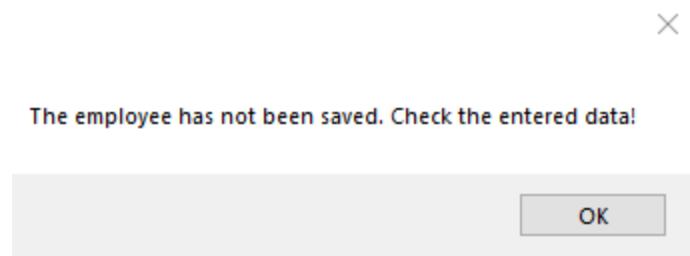
4. Систем памти податке о запосленом. (CO)
5. Систем приказује кориснику поруку: "Запослени је сачуван!". (IA)



Слика 139 - Запослени је сачуван

Алтернативна сценарија

- 5.1. Уколико систем не може да запамти податке о запосленом он приказује администратору поруку: "Запослени није сачуван. Проверите унете податке!". (IA)



Слика 140 - Запослени није сачуван

СК10: Случај коришћења – Деактивирање запосленог

Назив СК

Деактивирање запосленог

Актори СК

Администратор

Учесници СК

Администратор и систем (програм)

Предуслов: Систем је укључен и администратор је пријављен под својом шифром. Систем приказује форму за рад са запосленима.

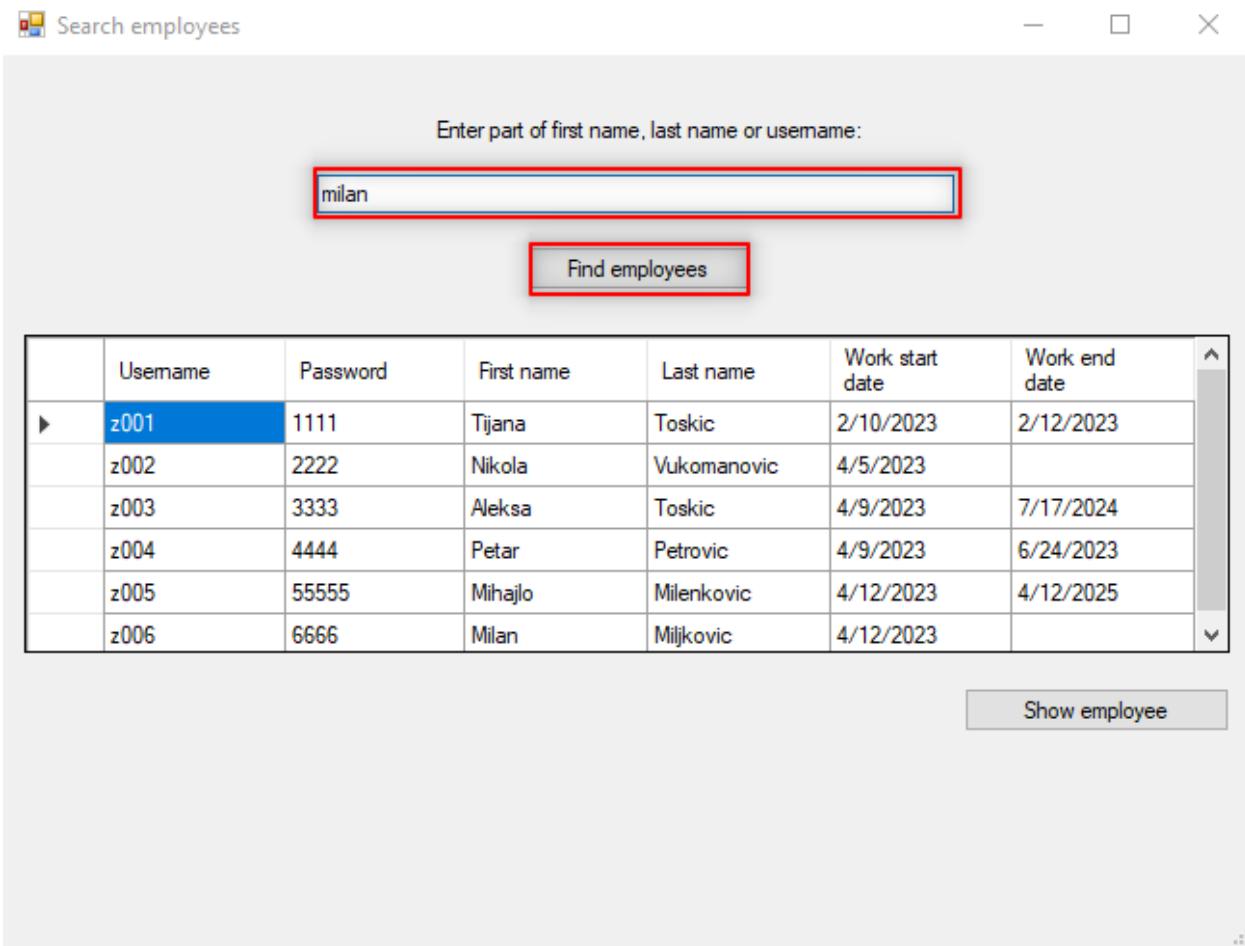
	Username	Password	First name	Last name	Work start date	Work end date
▶	z001	1111	Tijana	Toskic	2/10/2023	2/12/2023
	z002	2222	Nikola	Vukomanovic	4/5/2023	
	z003	3333	Aleksa	Toskic	4/9/2023	7/17/2024
	z004	4444	Petar	Petrovic	4/9/2023	6/24/2023
	z005	55555	Mihajlo	Milenkovic	4/12/2023	4/12/2025
	z006	6666	Milan	Miljkovic	4/12/2023	

Слика 141 - Форма за претрагу запосленог

Основни сценарио СК

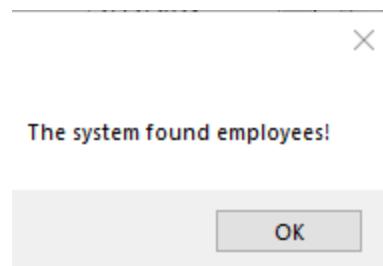
1. Администратор уноси вредност по којој претражује запослене. (АПУСО)
2. Администратор позива систем да нађе запослене по задатој вредности. (АПСО)

Опис акције: Корисник кликом на дугме Find employees позива системску операцију NadjiZaposlene(kriterijumPretrage, List<Zaposlen>) која приказује запослене.



Слика 142 - Претрага запосленог

3. Систем тражи запослене по задатој вредности. (CO)
4. Систем приказује администратору листу запослених и поруку: "Систем је нашао запослене по задатој вредности!". (ИА)



Слика 143 - Претрага је успешно извршена

Search employees

	Username	Password	First name	Last name	Work start date	Work end date
▶	z006	6666	Milan	Miljkovic	4/12/2023	

Enter part of first name, last name or username:
milan

Find employees

Show employee

Слика 144 - Приказ за запослених добијених претрагом

5. Администратор бира запосленог ког жели да деактивира. (АПУСО)
6. Администратор позива систем да учита податке о одабраном запосленом. (АПСО)

Опис акције: Корисник кликом на дугме Show employee позива системску операцију UcitajZaposnlenog(Zaposlen) која учитава податке о изабраном запосленом.

Search employees

	Username	Password	First name	Last name	Work start date	Work end date
▶	z006	6666	Milan	Miljkovic	4/12/2023	

Enter part of first name, last name or username:
milan

Find employees

Show employee

Слика 145 - Учитавање података о запосленом

7. Систем учитава податке о одабраном запосленом. (СО)
8. Систем приказује администратору податке о запосленом. (ИА)

The screenshot shows a Windows application window titled 'ShowEmployee'. Inside, there's a form titled 'Employee data' with the following fields:

- Username: z006
- Password: 6666
- First name: Milan
- Last name: Miljkovic
- Date of employment: Wednesday, April 12, 2023
- Do you want to set termination date?

At the bottom are four buttons: 'Show roles', 'Add role', 'Delete', and 'Edit'. The 'Delete' button is highlighted with a red rectangle.

Слика 146 - Приказ података о изабраном запосленом

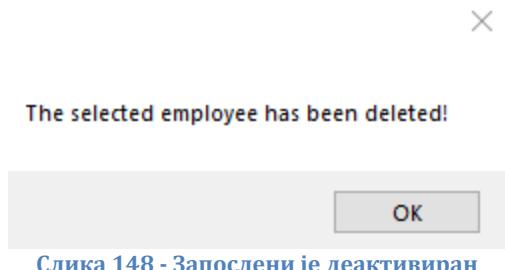
9. Администратор позива систем да деактивира запосленог. (АПСО)

Опис акције: Корисник кликом на дугме Delete позива системску операцију DeaktivirajZaposlenog (Zaposlen) која деактивира изабраног запосленог.

This screenshot is identical to Screenshot 146, showing the 'Employee data' form for user z006. The difference is that the 'Delete' button at the bottom of the form is now highlighted with a red rectangle, indicating it has been clicked.

Слика 147 - Деактивирање запосленог

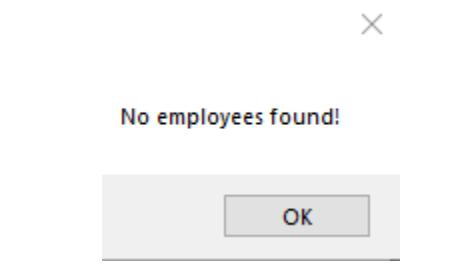
10. Систем деактивира запосленог. (СО)
11. Систем приказује администратор поруку: "Изабрани запослени је деактивиран!". (ИА)



Слика 148 - Запослени је деактивиран

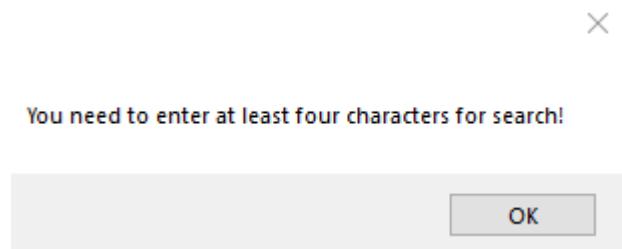
Алтернативна сценарија

- 4.1. Уколико систем не може да нађе запослене он приказује администратору поруку: "Ниједан запослени није пронађен!". Прекида се извршење сценарија. (ИА)



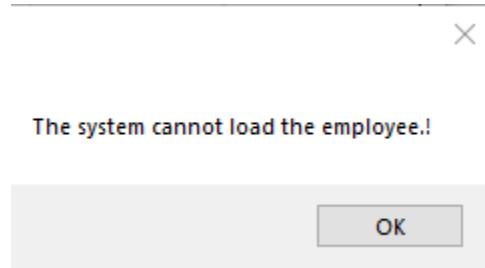
Слика 149 - запослени нису пронађени

- 4.2. Уколико администратор није унео довољан број карактера за претрагу запослених систем приказује администратору поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



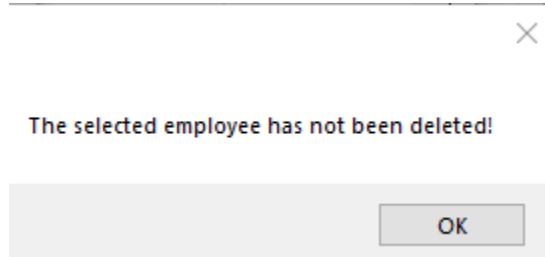
Слика 150 - Недовољан број карактера за претрагу

- 8.1. Уколико систем не може да учита запосленог, он приказује администратору поруку: "Систем није учитао запосленог!". Прекида се извршење сценарија. (ИА)



Слика 151 - Запослени не може да се учита

- 11.1. Уколико систем не може да деактивира запосленог он приказује администратору поруку: "Запослени није деактивиран!". (ИА)



Слика 152 - Запослени је деактивиран

СК11: Случај коришћења – Додавање улоге запосленом

Назив СК

Додавање улоге запосленом

Актори СК

Администратор

Учесници СК

Администратор и систем (програм)

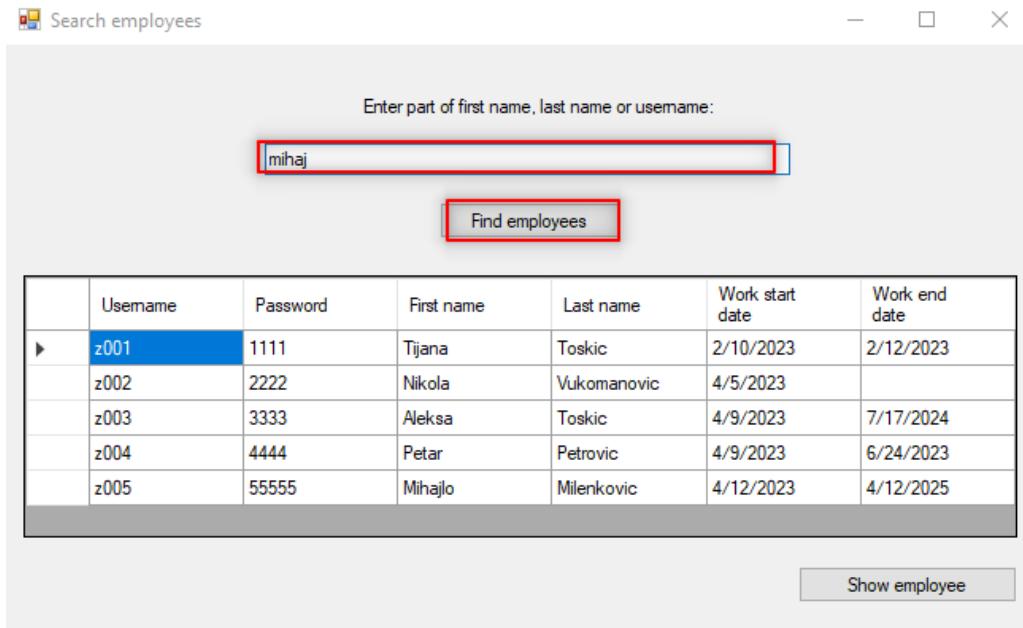
Предуслов: Систем је укључен и администратор је пријављен под својом шифром. Систем приказује форму за рад са запосленима. Листа Улога је учитана.

	Username	Password	First name	Last name	Work start date	Work end date
▶	z001	1111	Tijana	Toskic	2/10/2023	2/12/2023
	z002	2222	Nikola	Vukomanovic	4/5/2023	
	z003	3333	Aleksa	Toskic	4/9/2023	7/17/2024
	z004	4444	Petar	Petrovic	4/9/2023	6/24/2023
	z005	55555	Mihajlo	Milenkovic	4/12/2023	4/12/2025

Слика 153 - Форма за претрагу запослених

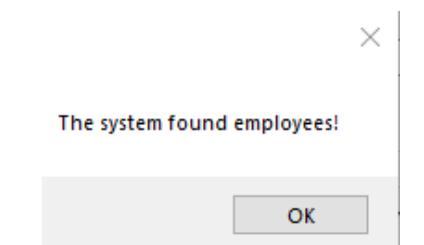
Основни сценарио СК

1. Администратор уноси вредност по којој претражује запослене. (АПУСО)
2. Администратор позива систем да нађе запослене по задатој вредности. (АПСО)

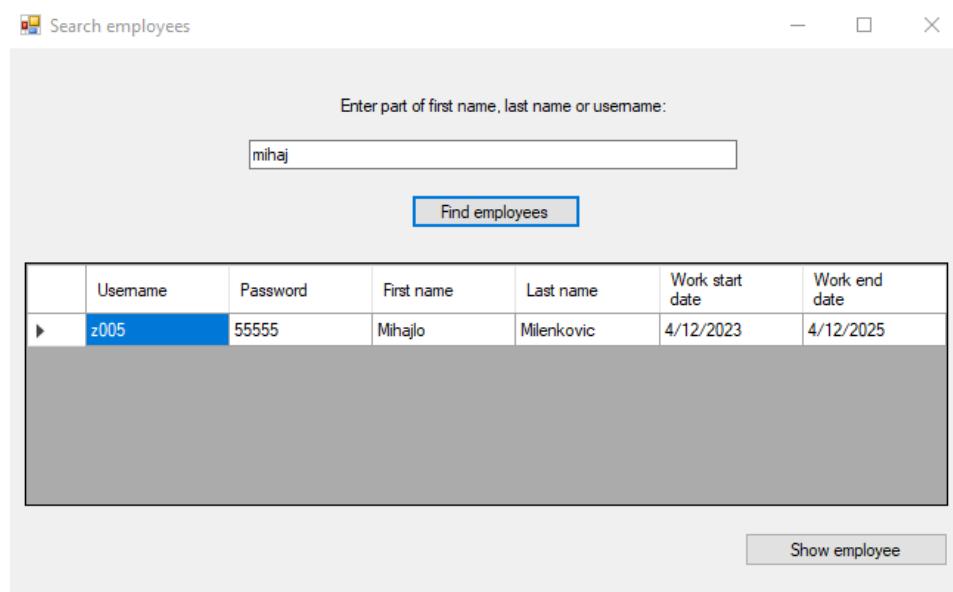


Слика 154 - Претрага запослених

3. Систем тражи запослене по задатој вредности. (СО)
4. Систем приказује администратору листу запослених и поруку: "Систем је нашао запослене по задатој вредности!". (ИА)

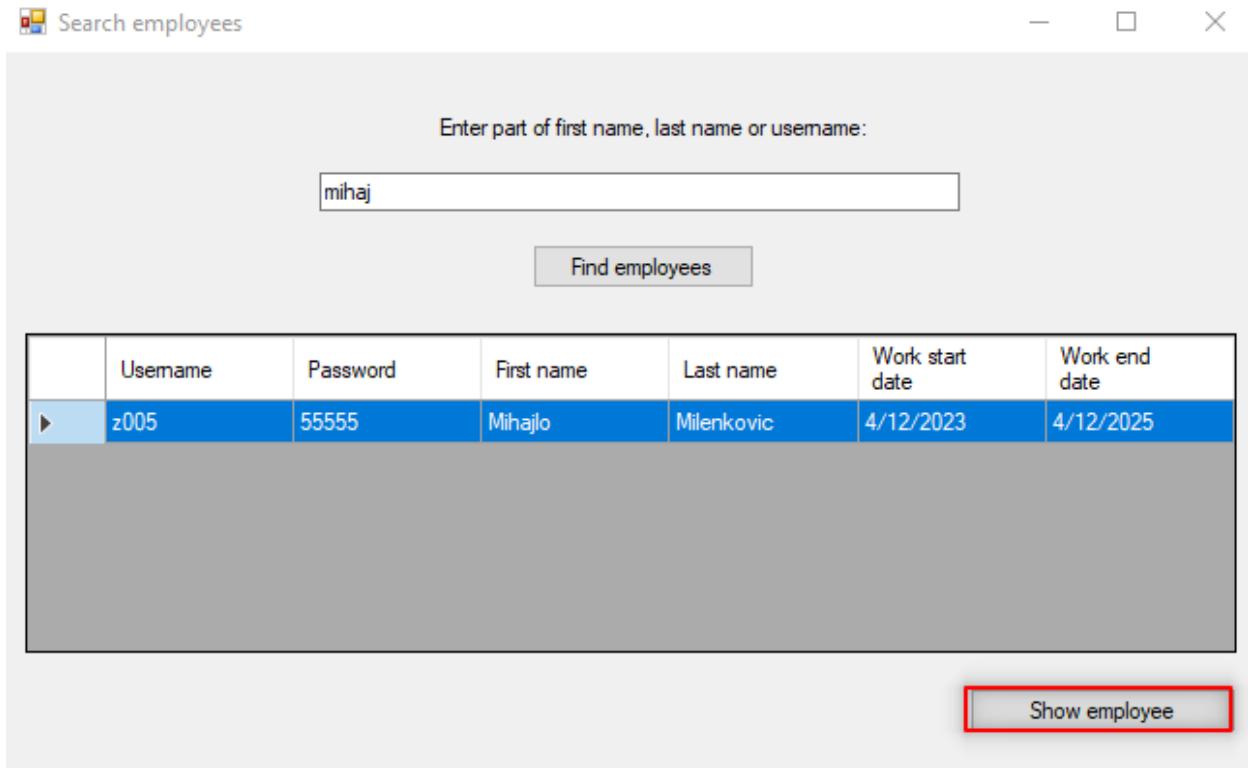


Слика 155 - Запослени су пронађени



Слика 156 - Приказ запослених

5. Администратор бира запосленог ком жељи да додели улогу. (АПУСО)
6. Администратор позива систем да учита податке о одабраном запосленом. (АПСО)



Слика 157 - Приказ изабраног запосленог

7. Систем учитава податке о одабраном запосленом. (СО)
8. Систем приказује администратору податке о запосленом. (ИА)

Employee data

Username: z005

Password: 55555

First name: Mihajlo

Last name: Milenkovic

Date of employment: Wednesday, April 12, 2023

Termination date: Saturday, April 12, 2025

Show roles Add role Delete Edit

Слика 158 - Додавање нове улоге запосленом

Опис акције: Корисник кликом на дугме Add role отвара форму за унос нове улоге за изабраног запосленог.

Employee: Mihajlo Milenkovic

Employee role data

Role:

Date from: Wednesday, April 12, 2023

Do you want to set date to?

Save

Слика 159 - Форма за додавање нове улоге

9. Администратор бира улогу коју жели да додели запосленом. (АПУСО)
 10. Администратор уноси податке о вези улога - запослени. (АПУСО)
 11. Администратор контролише да ли је коректно изабрао улогу и унео остале потребне податке. (АНСО)
 12. Администратор позива систем да запамти податке о вези улога - запослен. (АПСО)
- Опис акције: Корисник кликом на дугме Save позива системску операцију operaciju KreirajVezuUZ(Zaposlen, Uloga) која памти податке о новој вези.*

Employee: Mihajlo Milenkovic

Employee role data

Role: USER

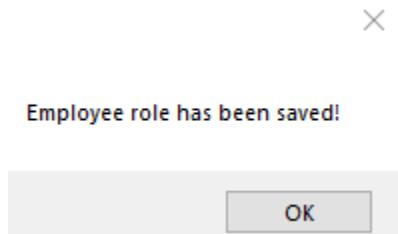
Date from: Wednesday, April 12, 2023

Do you want to set date to?

Save

Слика 160 - Унос нове улоге

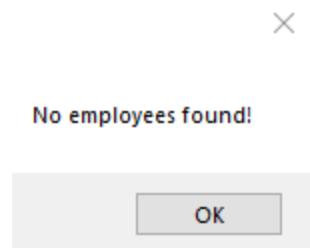
13. Систем памти податке о вези улога - запослен. (CO)
14. Систем приказује кориснику поруку: "Веза улога – запослен је сачувана!". (ИА)



Слика 161 - Нова улога је сачувана

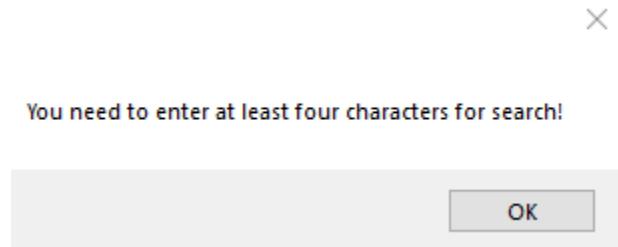
Алтернативна сценарија

- 4.1. Уколико систем не може да нађе запослене он приказује администратору поруку: "Ниједан запослени није пронађен!". Прекида се извршење сценарија. (ИА)



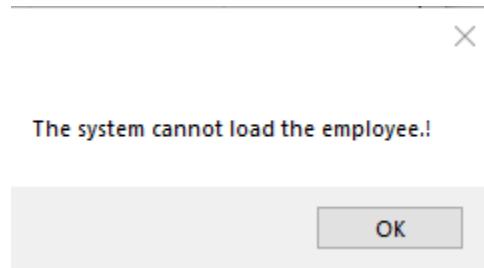
Слика 162 - Запослени нису пронађени

- 4.2. Уколико администратор није унео довољан број карактера за претрагу запослених систем приказује администратору поруку: "Нисте унели довољан број карактера за претрагу!" Прекида се извршење сценарија. (ИА)



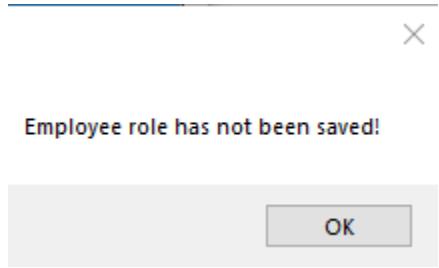
Слика 163 - Недовољан број карактера за претрагу

- 8.1. Уколико систем не може да учита запосленог, он приказује администратору поруку: "Систем није учитао запосленог!". Прекида се извршење сценарија. (ИА)



Слика 164 - Запослени није учитан

- 14.1. Уколико систем не може да запамти податке о вези улога - запослен он приказује администратору поруку: "Веза улога - запослен није сачувана. Проверите унете податке!". (ИА)



Слика 165 - Нова улога није сачувана

4.2.Пројектовање апликационе логике

Апликациони сервери треба да обезбеде сервисе који ће омогућити реализацију апликационе логике софтверског система. Пројектовани апликациони сервер садржи:

- део за комуникацију са клијентом
- контролер апликационе логике
- део за комуникацију са складиштем података (брокер базе података)
- део који садржи пословну логику

4.2.1.1. Комуникација са клијентима

Део за комуникацију подиже серверски сокет који даље ослушкује мрежу. Када клијент успостави конекцију, сервер генерише нит која ће бити одговорна за двосмерну везу са клијентом.

Клијент шаље захтев за извршење неке од системских операција, одговарајућа нит (додељена клијенту) приhvата захтев и прослеђује га до контролера апликационе логике. Након извршења системске операције, контролер враћа резултат “нити клијента”. Резултат се затим прослеђује клијенту.

Комуникација између клијента и сервера се обавља разменом објекта класе “*TransferClass*”.

4.2.1.2. Контролер апликационе логике

Контролер апликационе логике приhvата захтеве за извршење системских операција и исте прослеђује до конкретне системске операције. Након извршења системске операције, контролер приhvата одговор и враћа назад позиваоцу (нити клијента).

4.3. Пословна логика

4.3.1.1. Пројектовање понашања софтверског система – системске операције

Преко класе *ThreadClient*, прихватати се захтеви од клијента за извршење Системских операција, који се затим прослеђују до одговарајућих класа које су одговорне за извршење системских операција. За сваку системску операцију праве се софтверске класе које треба да реализују једну конкретну системску операцију (оне описују понашање система па се зато називају и софтверске класе понашања).

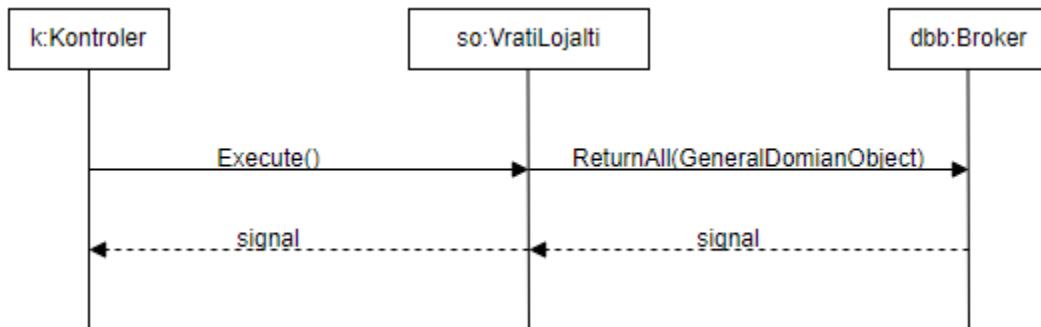
Уговор UG1: *VratiLoyaltyTipove*

Операција: *VratiLoyaltyTipove* (*List<Loyalty>*): signal;

Веза са СК: CK1, CK3

Предуслови:

Постуслови:



Слика 166 - Дијаграм УГ1

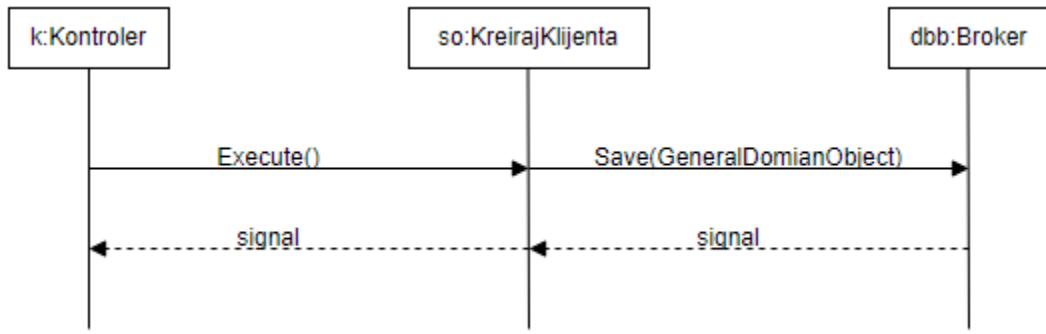
Уговор UG2: *KreirajKlijenta*

Операција: *KreirajKlijenta* (*Klijent*): signal;

Веза са СК: CK1

Предуслови: Вредносна и структурна ограничења над објектом Клијент морају бити задовољена.

Постуслови: Креиран је нови клијент.



Слика 167 - Дијаграм УГ2

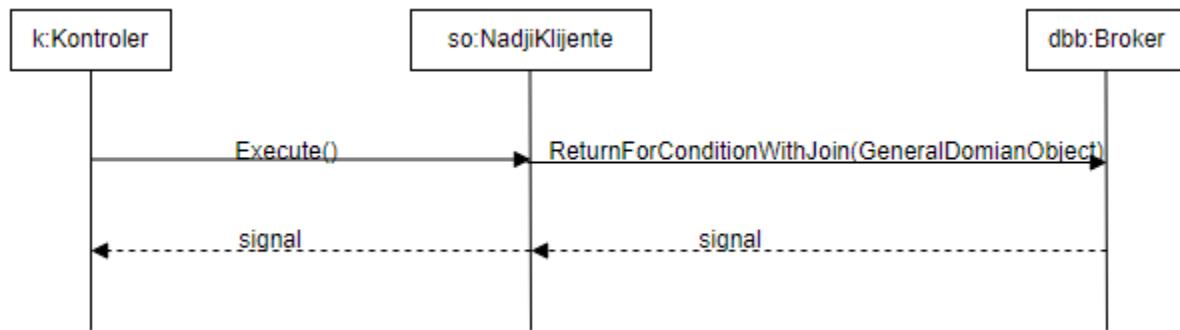
Уговор UG3: NadjiKlijente

Операција: `NadjiKlijente (kriterijumPretrage, List<Klijent>): signal;`

Веза са СК: СК2, СК3, СК7

Предуслови:

Постуслови:



Слика 168 - Дијаграм УГ3

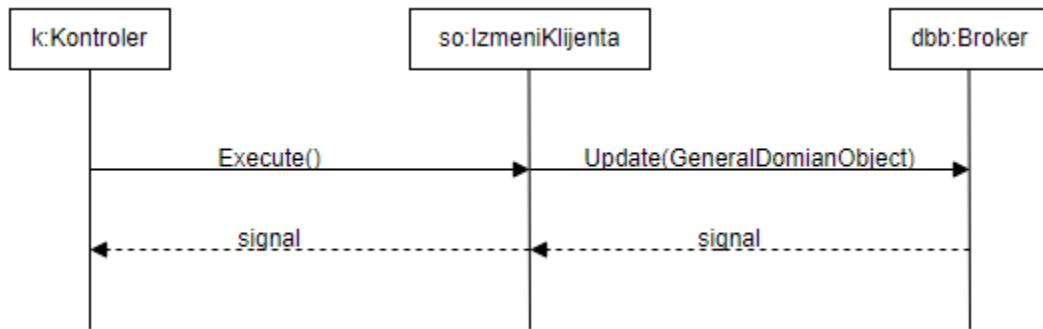
Уговор UG4: IzmeniKlijenta

Операција: IzmeniKlijenta (*Klijent*): signal;

Веза са СК: СК3

Предуслови: Вредносна и структорна ограничење над објектом Клијент морају бити задовољена.

Постуслови: Подаци о клијенту су изменењени.



Слика 169 - Дијаграм УГ4

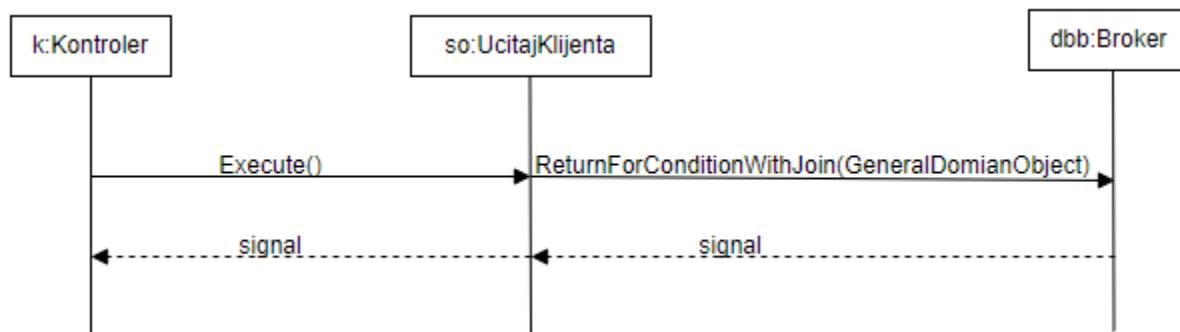
Уговор UG5: UcitajKlijenta

Операција: UcitajKlijenta (*Klijent*): signal;

Веза са СК: СК2, СК3

Предуслови:

Постуслови:



Слика 170 - Дијаграм УГ5

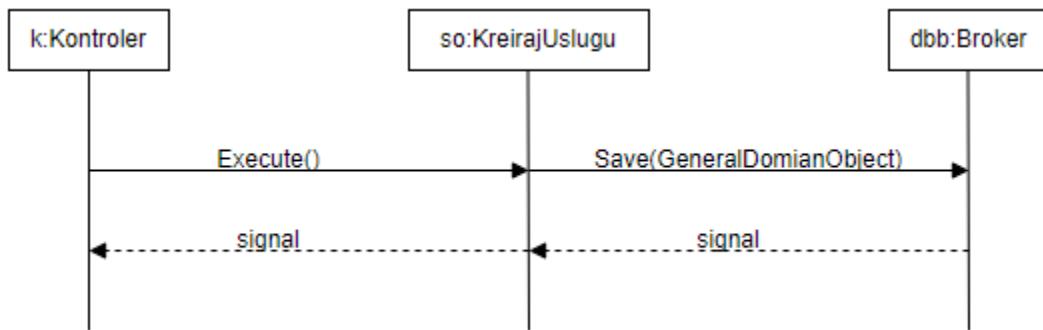
Уговор UG6: KreirajUslugu

Операција: KreirajUslugu (*Usluga*): signal;

Веза са СК: СК4

Предуслови: Вредносна и структорна ограничење над објектом Услуга морају бити задовољена.

Постуслови: Креирана је нова услуга.



Слика 171 - Дијаграм УГ6

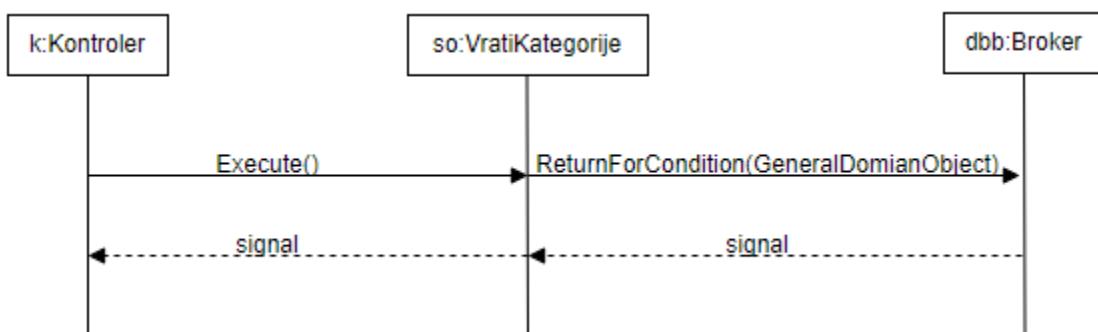
Уговор UG7: VratiKategorije

Операција: VratiKategorija (*List<Kategorija>*): signal;

Веза са СК: СК6

Предуслови:

Постуслови:



Слика 172-Дијаграм УГ7

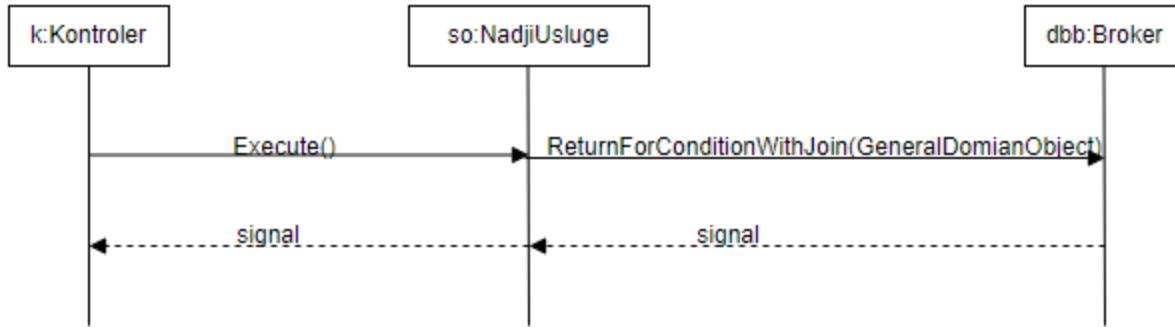
Уговор UG8: NadjiUsluge

Операција: NadjiUsluge (*kriterijumPretrage, List<Usluga>*): signal;

Веза са СК: CK5, CK6

Предуслови:

Постуслови:



Слика 173 - Дијаграм УГ8

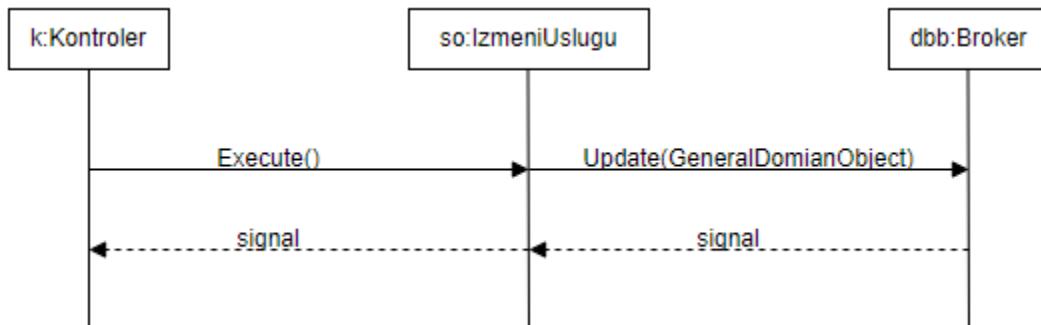
Уговор UG9: IzmeniUslugu

Операција: IzmeniUslugu (*Usluga*): signal;

Веза са СК: CK6

Предуслови: Вредносна и структорна ограничење над објектом Услуга морају бити задовољена.

Постуслови: Подаци о услуги су изменењени.



Слика 174 - Дијаграм УГ9

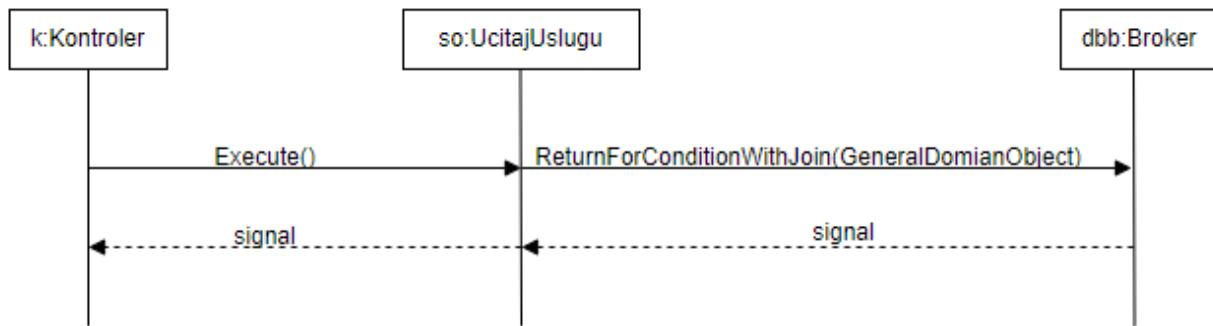
Уговор UG10: *UcitajUslugu*

Операција: *UcitajUslugu (Usluga): signal;*

Веза са СК: CK5, CK6

Предуслови:

Постуслови:



Слика 175 - Дијаграм УГ10

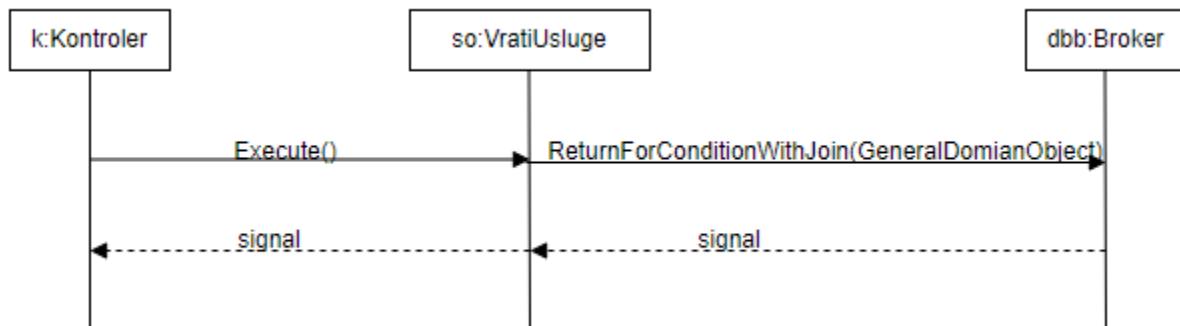
Уговор UG12: *VratiUsluge*

Операција: *VratiUsluge (List<Usluga>): signal;*

Веза са СК: CK7

Предуслови:

Постуслови:



Слика 176- Дијаграм УГ12

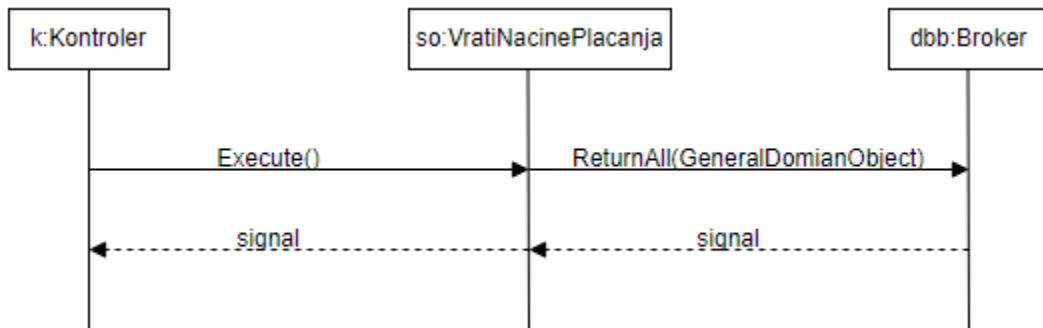
Уговор UG13: VratiNacinePlacanja

Операција: VratiNacinePlacanja (*List<NaciniPlacanja>*): signal;

Веза са СК: СК7

Предуслови:

Постуслови:



Слика 177 - Дијаграм УГ13

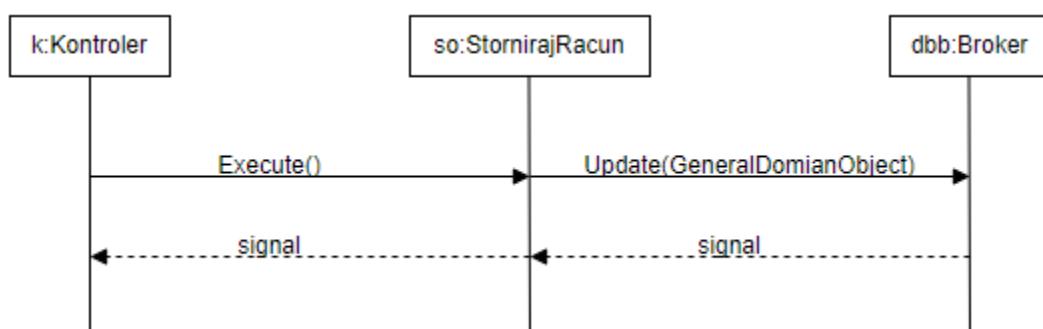
Уговор UG14: StornirajRacun

Операција: StornirajRacun (*Racun*): signal;

Веза са СК: СК8

Предуслови: Вредносно ограничење над објектом Рачун мора бити задовољено. Ако је рачун већ сторниран не може се извршити системска операција.

Постуслови: Рачун је сторниран.



Слика 178 - Дијаграм УГ14

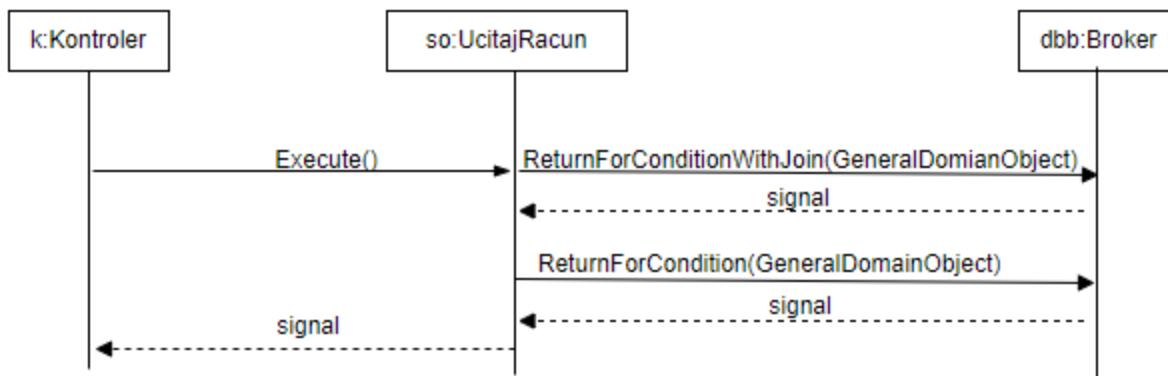
Уговор UG15: *UcitajRacun*

Операција: *UcitajRacun (Racun): signal;*

Веза са СК: СК8

Предуслови:

Постуслови:



Слика 179 - Дијаграм УГ15

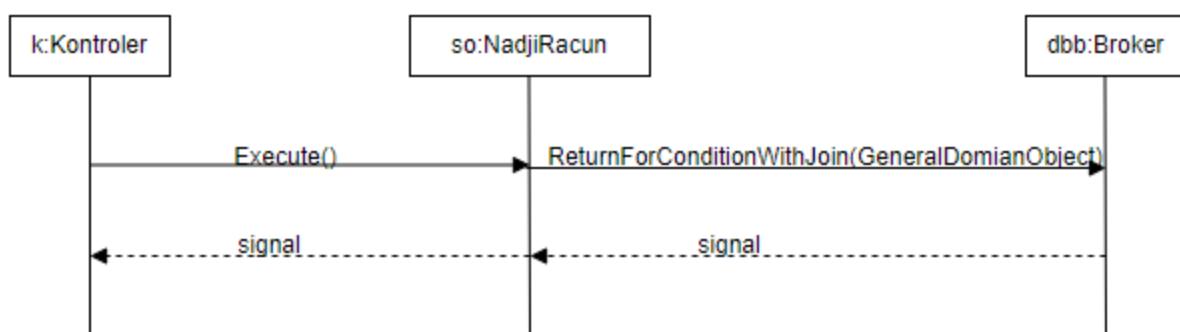
Уговор UG16: *NadjiRacun*

Операција: *NadjiRacun (kriterijumPretrage, List<Racun>): signal;*

Веза са СК: СК8

Предуслови:

Постуслови:



Слика 180 - Дијаграм УГ16

Уговор UG17: KreirajZaposlenog

Операција: KreirajZaposlenog (Zaposlen): signal;

Веза са СК: СК9

Предуслови: Вредносна и структорна ограничење над објектом Запослен морају бити задовољена.

Постуслови: Креиран је нови запослен.



Слика 181 - Дијаграм УГ17

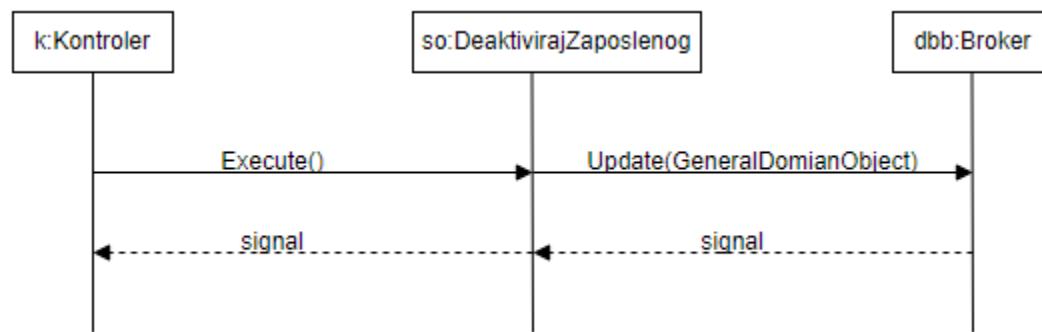
Уговор UG18: DeaktivirajZaposlenog

Операција: DeaktivirajZaposlenog (Zaposlen): signal;

Веза са СК: СК10

Предуслови: Вредносно ограничење над објектом Запослен мора бити задовољено.

Постуслови: Запослен је деактивиран.



Слика 182 - Дијаграм УГ18

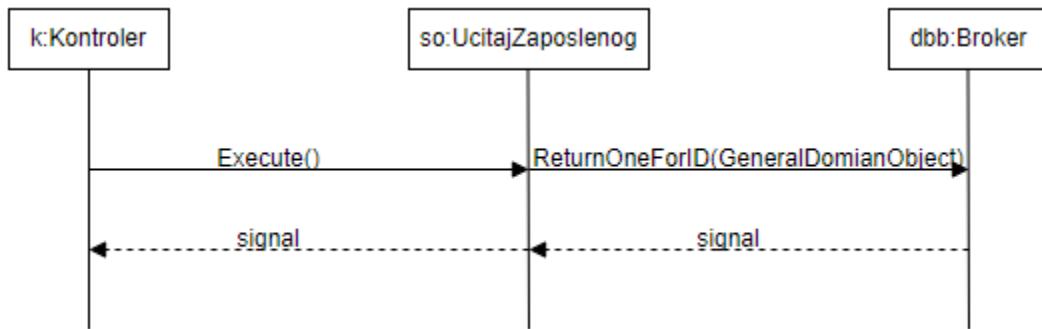
Уговор UG19: *UcitajZaposlenog*

Операција: *UcitajZaposlenog* (*Zaposlen*): signal;

Веза са СК: СК10, СК11

Предуслови:

Постуслови:



Слика 183 - Дијаграм УГ19

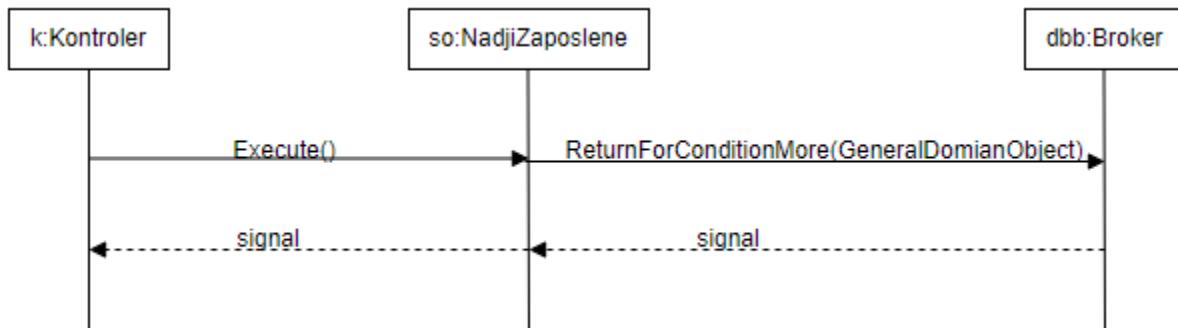
Уговор UG20: *NadjiZaposlene*

Операција: *NadjiZaposlene* (*kriterijumPretrage, List<Zaposlen>*): signal;

Веза са СК: СК10, СК11

Предуслови:

Постуслови:



Слика 184 - Дијаграм УГ20

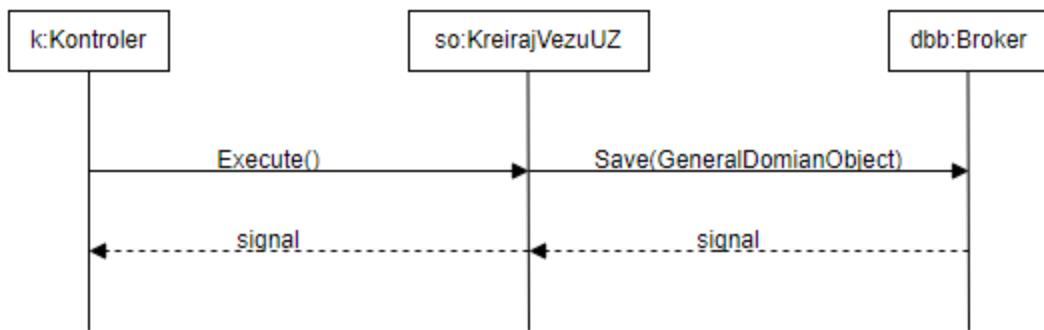
Уговор UG21: KreirajVezuUZ

Операција: KreirajVezuUZ (*Zaposlen, Uloga*): signal;

Веза са СК: CK11

Предуслови: Вредносна и структорна ограничење над објектом ЗапосленУлога морају бити задовољена.

Постуслови: Креиран је нова веза улога – запослен.



Слика 185 - Дијаграм УГ21

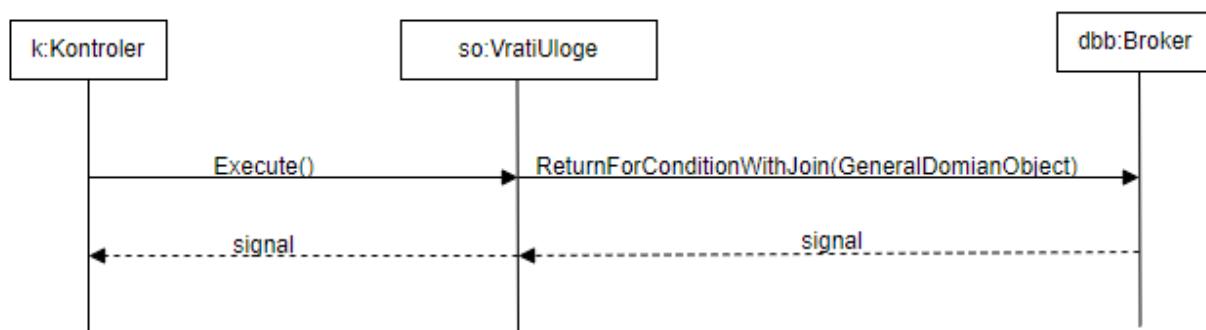
Уговор UG22: VratiUloge

Операција: VratiUloge (*List<Uloga>*): signal;

Веза са СК: CK11

Предуслови:

Постуслови:



Слика 186 - Дијаграм УГ22

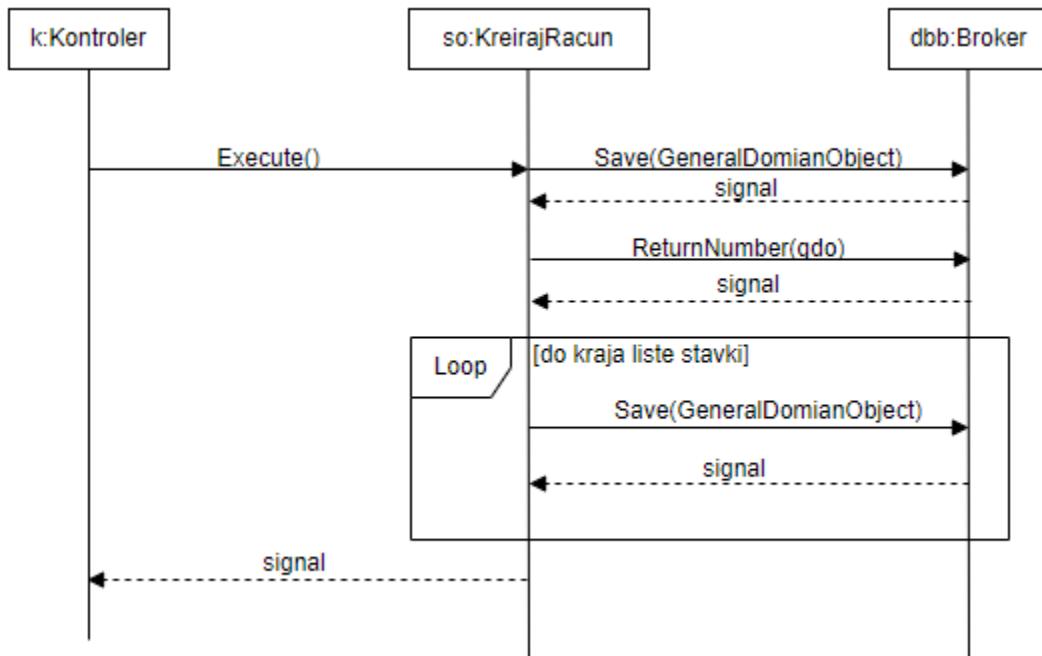
Уговор UG11: KreirajRacun

Операција: KreirajRacun (*Racun*): signal;

Веза са СК: CK7

Предуслови: Вредносна и структорна ограничење над објектом Рачун морају бити задовољена.

Постуслови: Креиран је нови рачун.



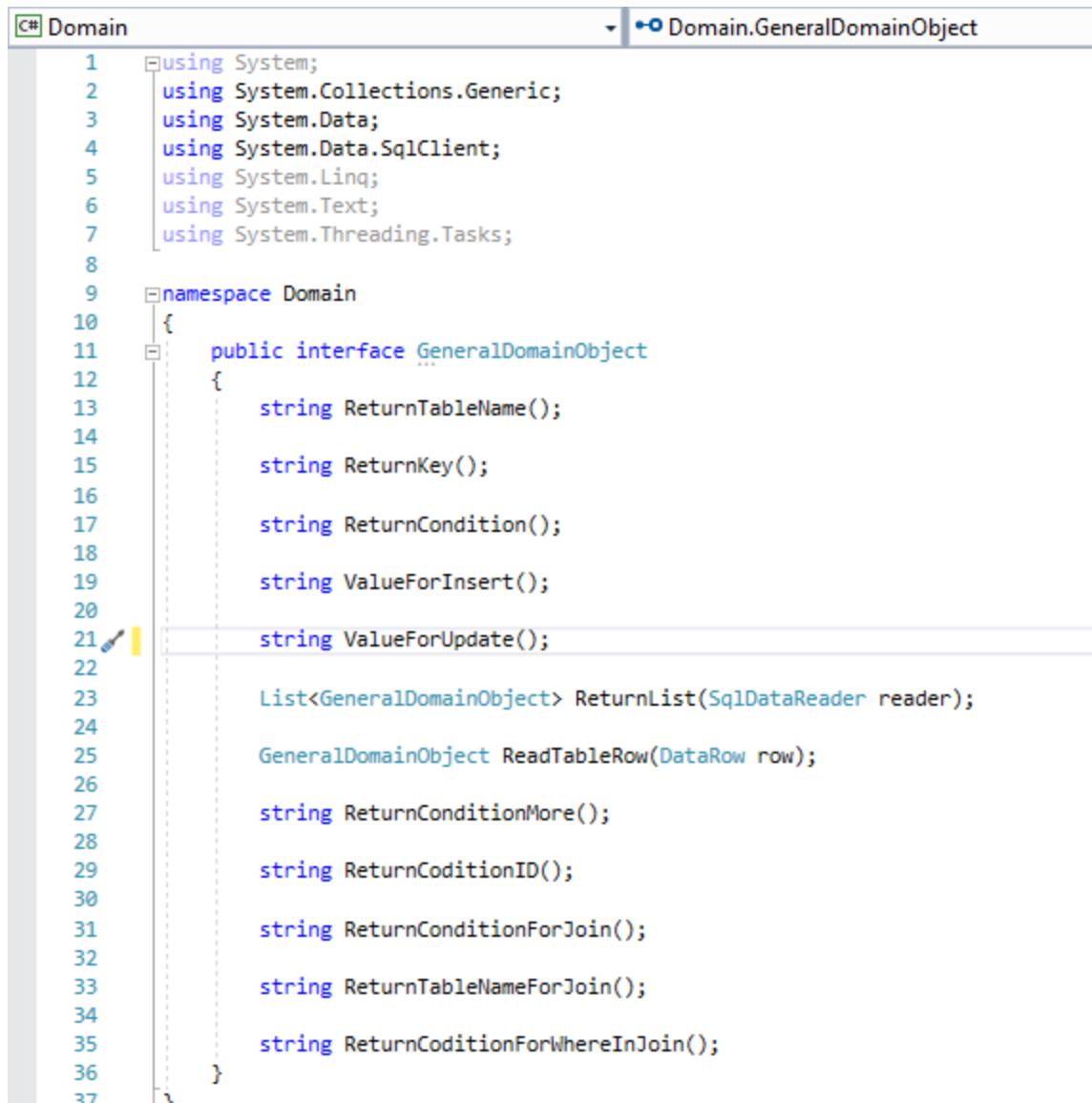
Слика 187 - Дијаграм УГ11



Слика 188 - Класе које су одговорне за СО наслеђују класу ОпштаСО

4.3.1.2. Пројектовање структуре софтверског система – Доменске класе

На основу концептуалних класа праве се софтверске класе структуре система. Свака класа садржи приватна поља атрибута, гетере и сетере за те атрибуте, конструктор, као и имплементацију методе Опште Доменске класе.



The screenshot shows a code editor window with the title bar 'C# Domain'. The code is defined in a class named 'GeneralDomainObject' located in the 'Domain' namespace. The code includes several methods for returning table names, keys, conditions, and values for insert and update operations, as well as methods for reading rows from a database reader and returning condition strings for joins. The code is numbered from 1 to 37.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Data;
4  using System.Data.SqlClient;
5  using System.Linq;
6  using System.Text;
7  using System.Threading.Tasks;
8
9  namespace Domain
10 {
11     public interface GeneralDomainObject
12     {
13         string ReturnTableName();
14
15         string ReturnKey();
16
17         string ReturnCondition();
18
19         string ValueForInsert();
20
21         string ValueForUpdate();
22
23         List<GeneralDomainObject> ReturnList(SqlDataReader reader);
24
25         GeneralDomainObject ReadTableRow(DataRow row);
26
27         string ReturnConditionMore();
28
29         string ReturnConditionID();
30
31         string ReturnConditionForJoin();
32
33         string ReturnTableNameForJoin();
34
35         string ReturnConditionForWhereInJoin();
36
37     }
}
```

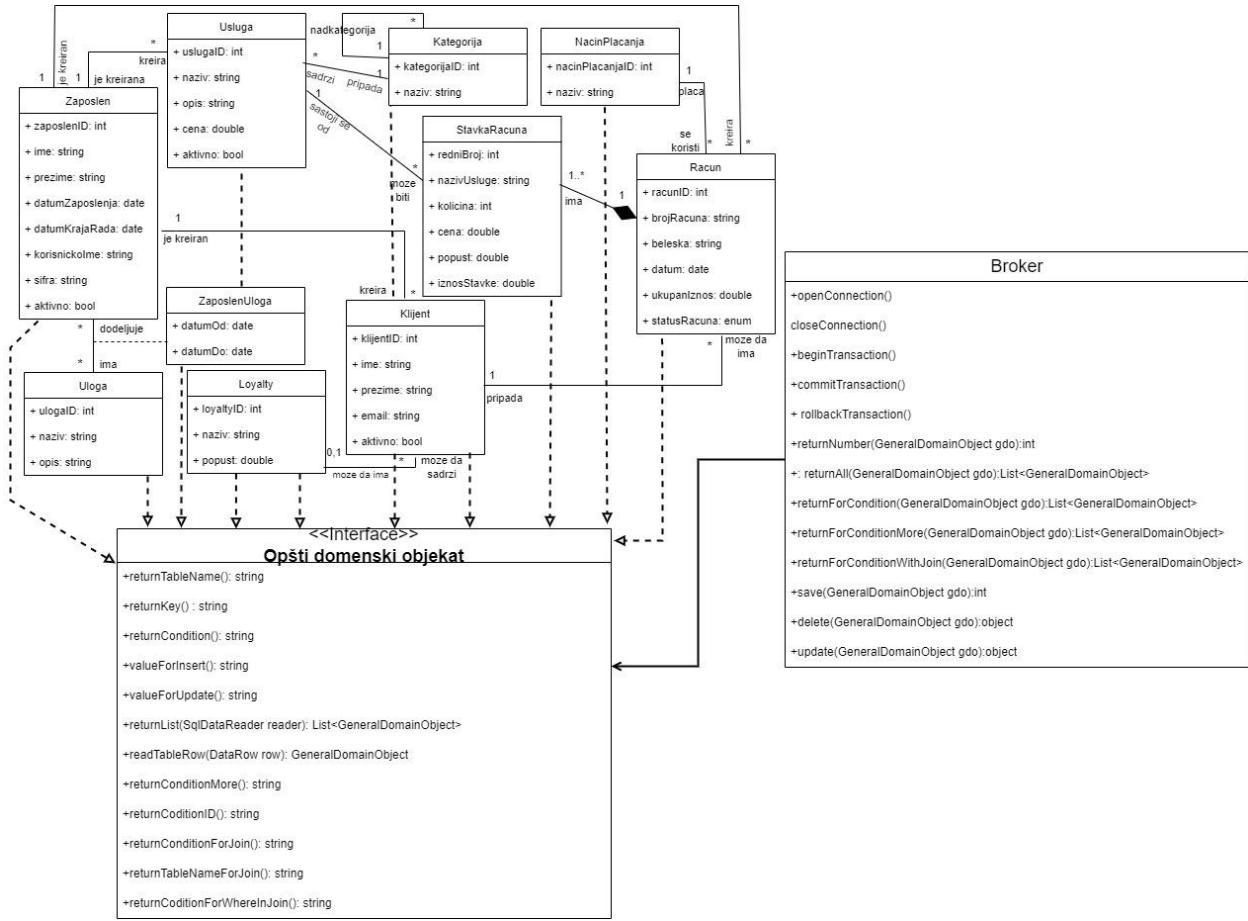
Слика 189 - код Опште доменске класе

4.4.Пројектовање брокера базе података

Брокер базе података је софтверска класа одговорна за комуникацију између пословне логике и складишта података. Односно, она се пројектује како би се обезбедио перзистентни сервис објектима доменских класа који се чувају у бази података. Класа Брокер представља перзистентни оквир који посредује у свим операцијама над базом података и реализује следеће методе:

```
public void OpenConnection()  
public void CloseConnection()  
public void BeginTransaction()  
public void RollbackTransaction()  
public void CommitTransaction()  
public int ReturnNumber(GeneralDomainObject gdo)  
public List< GeneralDomainObject > ReturnAll(GeneralDomainObject gdo)  
public List< GeneralDomainObject > ReturnForCondition(GeneralDomainObject gdo)  
public List< GeneralDomainObject > ReturnForConditionMore(GeneralDomainObject gdo)  
public List< GeneralDomainObject > ReturnForConditionWithJoin(GeneralDomainObject gdo)  
public GeneralDomainObject ReturnOneForID(GeneralDomainObject gdo)  
public object Save(GeneralDomainObject gdo)  
public object Delete (GeneralDomainObject gdo)  
public object Update (GeneralDomainObject gdo)
```

Методе које се налазе у класи Брокер пројектоване су као генеричке, што значи су улазни аргументи метода различити домански објекти. На овај начин се постиже значајно олакшање при одржавању кода зато што није потребно имплементирати појединачне методе за сваку доменску класу.



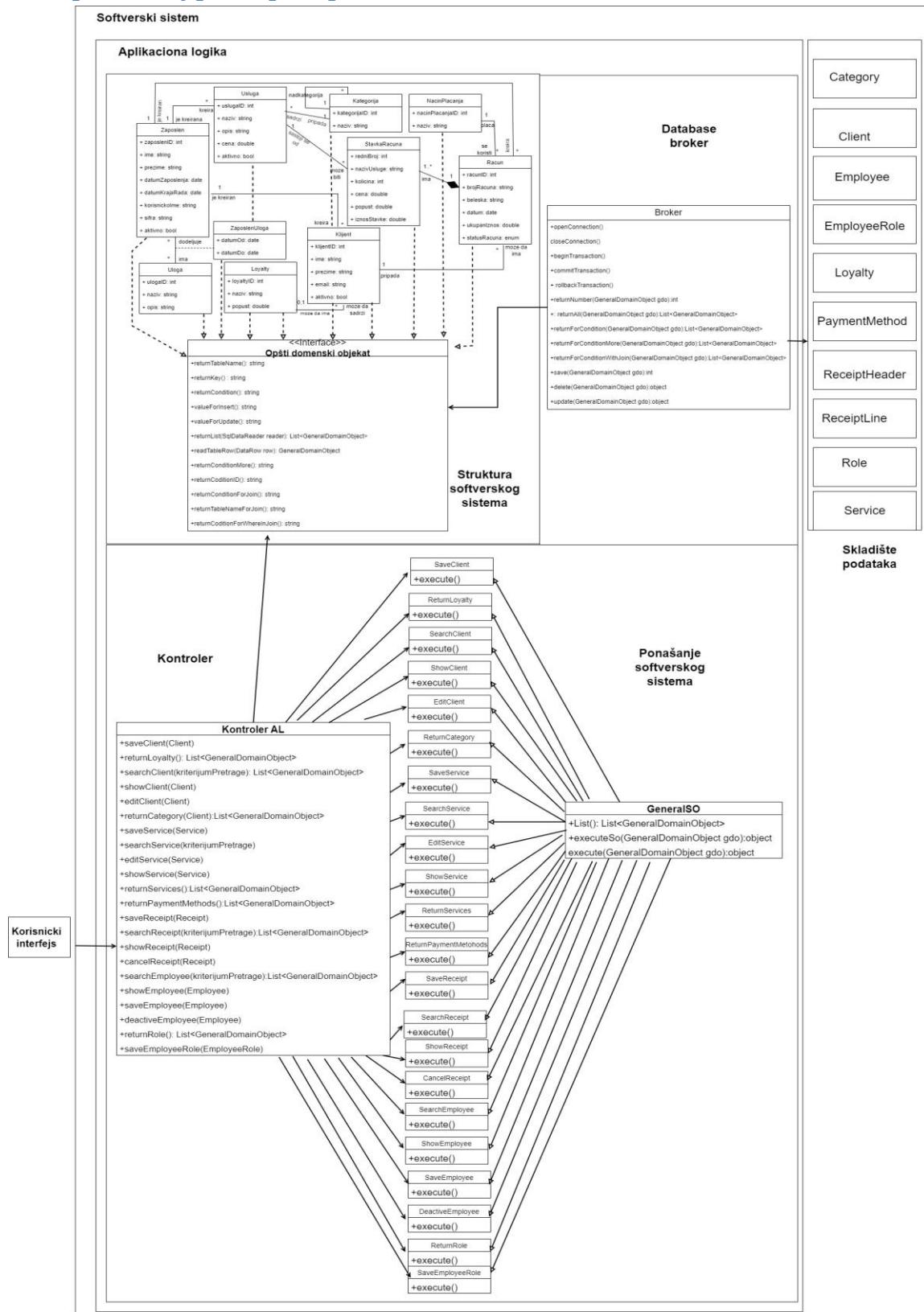
Слика 190 - Брокер класа се повезује са класом ОпштиДоменскиОбјекат

4.5. Релациони модел

На основу концептуланог модела добијен је следећи релациони модел:

- **Zaposlen**(zaposlenID, ime, prezime, datumZaposlenja, datumKrajaRada, korisničkoIme, šifra, aktivno)
- **Uloga**(ulogalID, naziv, opis)
- **ZaposlenUloga**(zaposlenID, ulogalID, datumOd, datumDo)
- **Klijent**(klijentID, ime, prezime, email, aktivno, *loyaltyID*, *zaposlenID*)
- **Loyalty**(loyaltyID, naziv, popust)
- **Usluga**(uslugaID, naziv, opis, cena, aktivno, *kategorijaID*, *zaposlenID*)
- **Kategorija** (kategorijaID, naziv, *nadKategorijaID*)
- **Racun** (racunID, brojRacuna, beleska, datum, ukupanIznos, statusRacuna, *klijentID*, *nacinPlacanjaID*, *zaposlenID*)
- **StavkeRacuna** (racunID, redniBroj, nazivUsluge, kolicina, cena, popust, iznosStavke, *uslugaID*)
- **NacinPlacanja** (nacinPlacanjaID, naziv)

4.6. Архитектура софтверског система



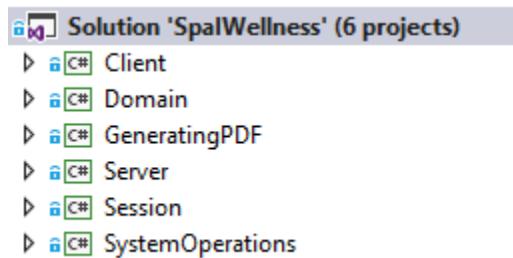
Слика 191 - Архитектура софтверског система

5. Имплементација

5.1. Имплементација пројеката из којих се састоји решење *SpaCenter*

Софтверски систем, резултат овог рада развијен је у програмском језику *C#*. Систем је пројектован као клијент-сервер апликација. За израду дијаграма пројектне документације коришћен је *Microsoft Word*. Као развојно окружење коришћен је *Microsoft Visual Studio 2017*.

Пројекат је реализован на основу шест пројекта, пројекти су приказани на слици:



Слика 192 - Пројекти из којих се састоји пројекат Спа центар

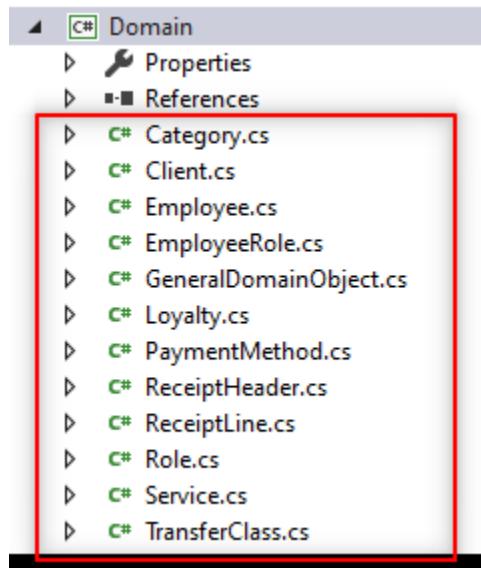
На основу архитектуре система добијене су следеће софтверске класе:

5.1.1.1. Класе *SpaWellness* – Domain

У пројекту *Domain* сачуване су све класе које се односе на сам Спа систем.

Поредих ових класа, постоји и класа *GeneralDomainObject*, ову класу наслеђују све горе поменуте класе и преко ове класе су имплементирају све мотеде које су потребне да би Брокер успешно повукао податке из базе.

Класа *TransferClass* се користи за комуникацију између пројектата.



Слика 193 - Доменске класе

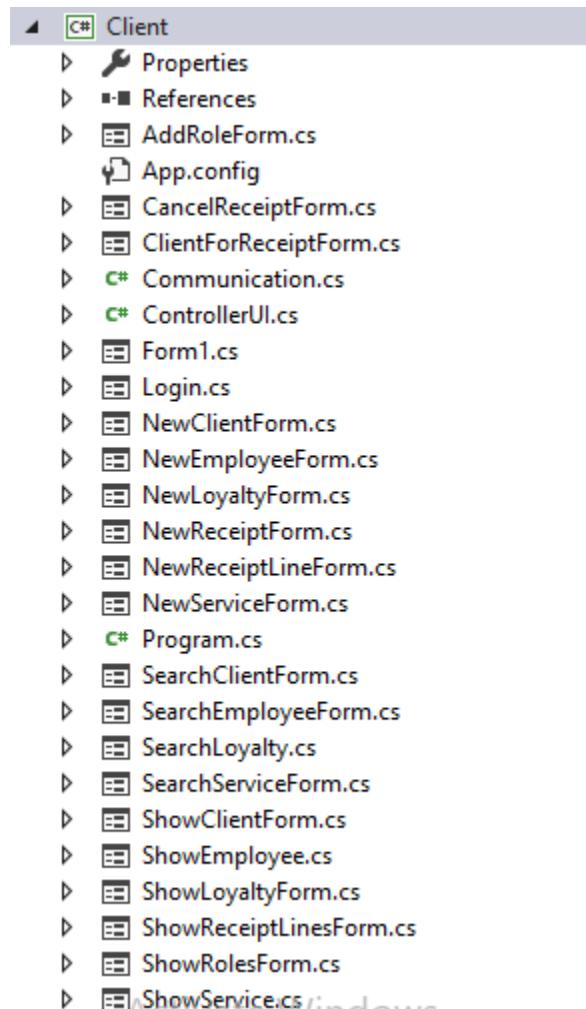
5.1.1.2. Класе SpaWellness – Client

Пројекат *Client* се састоји из следећих форми:

- *Login.cs*
- *Form1.cs*
- *NewClientForm.cs*
- *SearchClientForm.cs*
- *ShowClientForm.cs*
- *NewServiceForm.cs*
- *SearchServiceForm.cs*
- *ShowService.cs*
- *NewReceiptForm.cs*
- *New ReceiptLineForm.cs*
- *ClientForReceiptForm.cs*
- *CancelReceiptForm.cs*
- *ShowReceiptForm.cs*
- *NewLoyaltyForm.cs*
- *SearchLoyalty.cs*
- *ShowLoyaltyForm.cs*
- *NewEmployeeForm.cs*
- *SearchEmployeeForm.cs*
- *ShowEmployee.cs*
- *AddRoleForm.cs*
- *ShowRolesForm.cs*

Класе које садржи:

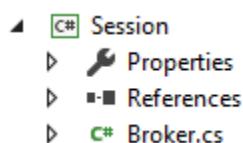
- *Communication.cs*
- *ControllerUI.cs*
- *Program.cs*



Слика 194 – Пројекат Клијент

5.1.1.3. Класе SpalWellness – Session

Пројекат *Session* садржи класу *Broker*.

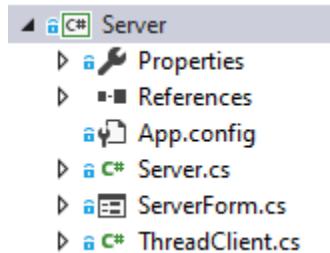


Слика 195 - Пројекат Сесија

5.1.1.4. Класе SpaWellness – Server

Пројекат *Server* садржи класе:

- *Server.cs*
- *ThreadClient.cs*.

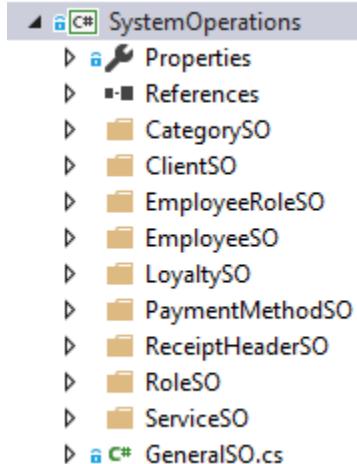


Слика 196 - Пројекта Сервер

Server садржи и форму *ServerForm.cs* која се користи за покретање сервера.

5.1.1.5. Класе SpaWellness – SystemOperations

Пројекат *SystemOperations* садржи све системске операције груписане по фолдерима који представљају објекте на који се те операције односе.

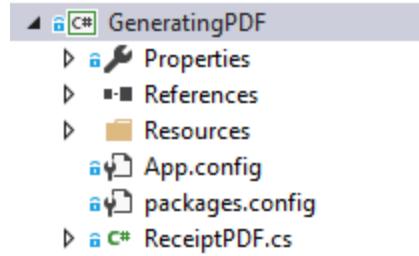


Слика 197 - Пројекат Системске операције

Садржи и класу *GeneralSO* коју наслеђују све системске операције. У овој класу се отвара и затвара конекција ка бази података као и трансакција.

5.1.1.6. Класе SpaWellness – GeneratingPDF

Пројекат *GeneratingPDF* садржи референцу на библиотеку iText7 која се користи за креирање ПДФ документа. У фолдеру *Resource* садржи импортовану слику која се користи за лого на ПДФ документима. У класу *ReceiptPDF* је имплементирано креирање ПДФ за рачуна, тј. креирање ПДФ-а за класу *ReceiptHeader*.



Слика 198- Пројекат за креирање ПДФ докумената

5.2. Имплементација складишта података

На основу софтверских класа структуре (доменских класа), имплементиране су табеле (складишта података) релационог система за управљање базом података. За развој овог софтверског система коришћен је *Microsoft SQL Server Management Studio 18*.

За креирање већине табеле коришћена је скрипа, овог типа:

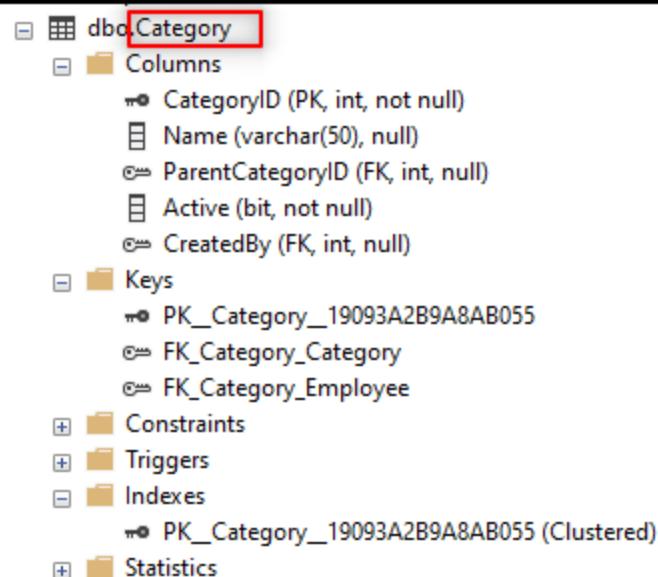
```
CREATE TABLE Category (
    CategoryID int IDENTITY(1,1) PRIMARY KEY,
    Name varchar(255) not null,
    ParentCategoryID int,
    Active bit default 1,
    CreatedBy int,
);
```

Слика 199 - скрипта за креирање табела

У већини табела колона која представља примарни кључ се аутоматски генерише, тако да при инсертовању рекорда у табеле није потребно да се проследи вредност за примарни кључ. Изузетак су табеле које имају сложени примарни кључ и које садрже примарни кључ неке друге табеле.

Креиран је дијаграм где су повезани сви спољни кључеви и дефинисане акције које се покрећу код *delete* и *update* метода.

Структура табела:



Слика 200 - Табела Категорије

dbo.Client

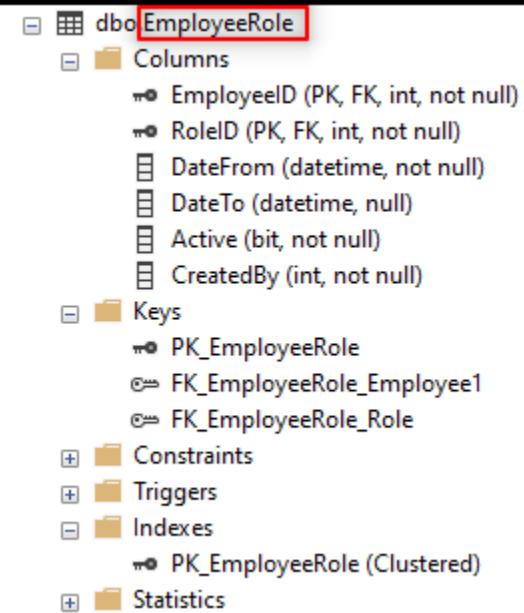
- Columns
 - ClientID (PK, int, not null)
 - FirstName (varchar(50), null)
 - LastName (varchar(50), null)
 - Email (varchar(255), null)
 - LoyaltyID (FK, int, null)
 - Active (bit, not null)
 - CreatedBy (FK, int, not null)
- Keys
 - PK_Client_E67E1A04DCF3A642
 - FK_Client_Employee
 - FK_Client_Loyalty
- Constraints
- Triggers
- Indexes
 - PK_Client_E67E1A04DCF3A642 (Clustered)
- Statistics

Слика 201 - Табела Клијент

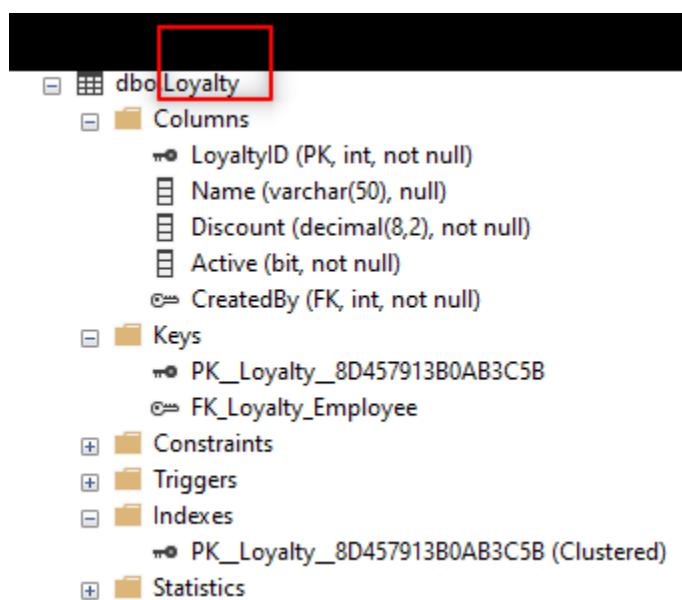
dbo.Employee

- Columns
 - EmployeeID (PK, int, not null)
 - Username (varchar(50), null)
 - Password (varchar(50), null)
 - FirstName (varchar(50), null)
 - LastName (varchar(50), null)
 - WorkStartDate (datetime, not null)
 - WorkEndDate (datetime, null)
 - Active (bit, not null)
 - CreatedBy (FK, int, null)
- Keys
 - PK_Employee_7AD04FF13B7F1E78
 - FK_Employee_Employee
- Constraints
- Triggers
- Indexes
 - PK_Employee_7AD04FF13B7F1E78 (Clustered)
- Statistics

Слика 202 - Табела Запослени



Слика 203 - Табела Запослени - Улога



Слика 204 - Табела Лојалти

The screenshot shows the structure of the `PaymentMethod` table in the `dbo` schema. The table has the following columns:

- `PaymentMethodID` (PK, int, not null)
- `Name` (varchar(255), null)
- `Active` (bit, not null)
- `CreatedBy` (FK, int, null)

It also contains the following keys:

- `PK_PaymentM_DC31C1F345E06521`
- `FK_PaymentMethod_Employee`

And the following indexes:

- `PK_PaymentM_DC31C1F345E06521` (Clustered)

Слика 205 - Табела Начини плаћања

The screenshot shows the structure of the `ReceiptHeader` table in the `dbo` schema. The table has the following columns:

- `ReceiptID` (PK, int, not null)
- `ReceiptNumber` (varchar(50), null)
- `Note` (varchar(255), null)
- `ReceiptDate` (datetime, null)
- `Amount` (decimal(8,2), null)
- `Status` (varchar(50), null)
- `ClientID` (FK, int, null)
- `PaymentMethodID` (FK, int, not null)
- `Active` (bit, not null)
- `CreatedBy` (FK, int, not null)

It also contains the following keys:

- `PK_ReceiptH_CC08C400B62A111C`
- `FK_ReceiptHeader_Client`
- `FK_ReceiptHeader_Employee`
- `FK_ReceiptHeader_PaymentMethod`

And the following indexes:

- `PK_ReceiptH_CC08C400B62A111C` (Clustered)

Associated objects include:

- `dbo.ReceiptLine`
- `dbo.Role`
- `dbo.Service`

Слика 206 - Табела Заглавље рачуна

The screenshot shows the structure of the `ReceiptLine` table in the `dbo` schema. The table has the following columns:

- `ReceiptID` (PK, FK, int, not null)
- `LineNumber` (PK, int, not null)
- `ServiceName` (varchar(50), null)
- `Quantity` (int, null)
- `Price` (decimal(8,2), null)
- `Discount` (decimal(8,2), null)
- `LineAmount` (decimal(8,2), null)
- `ServiceID` (FK, int, not null)
- `Active` (bit, not null)
- `CreatedBy` (FK, int, not null)

Keys defined for the table include:

- `PK_ReceiptLine`
- `FK_ReceiptLine_Employee`
- `FK_ReceiptLine_ReceiptHeader`
- `FK_ReceiptLine_Service`

Indexes defined for the table include:

- `PK_ReceiptLine` (Clustered)

Statistics defined for the table include:

Слика 207 - Табела Линије рачуна

The screenshot shows the structure of the `Role` table in the `dbo` schema. The table has the following columns:

- `RoleID` (PK, int, not null)
- `Name` (varchar(50), null)
- `Description` (varchar(255), null)
- `Active` (bit, not null)
- `CreatedBy` (FK, int, null)

Keys defined for the table include:

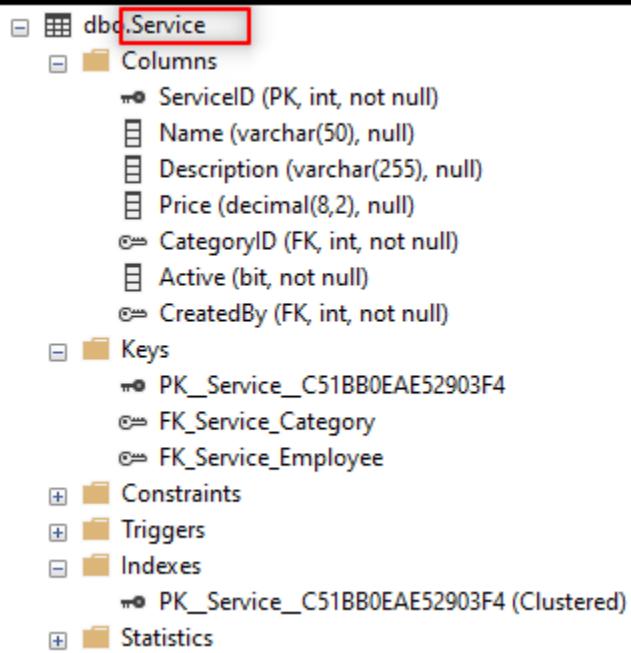
- `PK_Role_8AFACE3AB00900E1`
- `FK_Role_Employee`

Indexes defined for the table include:

- `PK_Role_8AFACE3AB00900E1` (Clustered)

Statistics defined for the table include:

Слика 208 - Табела Улоге



Слика 209 - Табела Услуге

5.3. Имплементација системских операција

5.3.1.1. CO KreirajKlijenta

```
C# SystemOperations
  1 using System;
  2 using System.Collections.Generic;
  3 using System.Linq;
  4 using System.Text;
  5 using System.Threading.Tasks;
  6 using Domain;
  7 using Session;
  8
  9 namespace SystemOperations.ClientSO
 10 {
 11     public class SaveClient : GeneralSO
 12     {
 13         protected override object Execute(GeneralDomainObject gdo)
 14         {
 15             return Broker.ReturnSession().Save(gdo);
 16         }
 17     }
 18 }
 19
```

Слика 210 - Код СО Креирај клијента

5.3.1.2. CO VratiLoyaltyTipove

```
C# SystemOperations
  1 using System;
  2 using System.Collections.Generic;
  3 using System.Linq;
  4 using System.Text;
  5 using System.Threading.Tasks;
  6 using Domain;
  7 using Session;
  8
  9 namespace SystemOperations.LoyaltySO
 10 {
 11     public class ReturnLoyalties : GeneralSO
 12     {
 13         protected override object Execute(GeneralDomainObject gdo)
 14         {
 15             return Broker.ReturnSession().ReturnAll(gdo).OfType<Loyalty>().ToList<Loyalty>();
 16         }
 17     }
 18 }
 19
```

Слика 211 - Код СО Врати лојалти типове

5.3.1.3. CO NadjiKlijente

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.ClientSO
10 {
11     public class SearchClients : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().ReturnForConditionWithJoin(gdo).OfType<Domain.Client>().ToList<Domain.Client>();
16         }
17     }
18 }
19

```

Слика 212 - Код CO Нађи клијенте

Клијенте је могуће претражити преко имена, презимена, лојалти типа. Апликација подржава и претрагу преко комбинације поброжаних критеријума. У доменском објекту Клијент који се прослеђује да би се системска операција извршила, чувају се критеријуми претраге у зависности од тога који критеријум претрага је клијент унео. Наведено је имплементирано у методи *ReturnConditionForWhereInJoin()* где се у зависности од наредбе *if* која се извршава креира *where* клаузула за генеричку методу *ReturnForConditionWithJoin(gdo)* која се позива за дату системску операцију.

```

}
public string ReturnConditionForWhereInJoin()
{
    if (this.firstName == null && this.loyalty == null) return "c.Active = 1";
    if (this.clientID>0) return "c.Active = 1 and c.ClientID = "+this.clientID+"";
    if (this.firstName == "" && this.loyalty != null) return "c.Active = 1 and c.LoyaltyID = "+this.loyalty.LoyaltyID+" ";
    if (this.firstName!="" && this.loyalty == null) return "c.Active = 1 and" +
        "(c.FirstName like '%" + this.firstName + "%' or c.LastName like '%" + this.firstName + "%' or c.Email like '%" + this.firstName + "%')";
    return "c.Active = 1 and" +
        "(c.FirstName like '%" + this.firstName + "%' or c.LastName like '%" + this.firstName + "%' or c.Email like '%" + this.firstName + "%') and" +
        "c.LoyaltyID = " + this.loyalty.LoyaltyID + " ";
}
#endregion

```

Слика 213 - Код CO имплементација различитих типова претраге клијента

5.3.1.4. CO UcitajKlijenta

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using Domain;
7 using Session;
8
9 namespace SystemOperations.ClientSO
10 {
11     public class SearchClients : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().ReturnForConditionWithJoin(gdo).OfType<Domain.Client>().ToList<Domain.Client>();
16         }
17     }
18 }
19
```

Слика 214 - Код СО Учитај клијента

5.3.1.5. CO IzmeniKlijenta

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using Domain;
7 using Session;
8
9 namespace SystemOperations.ClientSO
10 {
11     public class EditClient : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().Update(gdo);
16         }
17     }
18 }
```

Слика 215 - Код СО Измени клијента

5.3.1.6. CO VratiKategorije

The screenshot shows a code editor with the following code:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.CategorySO
10 {
11     public class ReturnCategories : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().ReturnForCondition(gdo).OfType<Category>().ToList<Category>();
16         }
17     }
18 }
```

Слика 216 - Код CO Врати категорије

5.3.1.7. CO KreirajUslugu

The screenshot shows a code editor with the following code:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.ServiceSO
10 {
11     public class SaveService : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().Save(gdo);
16         }
17     }
18 }
19 
```

Слика 217 - Код CO Креирај услугу

5.3.1.8. CO NadjiUsluge

```
C# SystemOperations
SystemOperations.ServiceSO.SearchServices
Execute(GeneralDomainObject gdo)

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.ServiceSO
10 {
11     public class SearchServices : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().ReturnForConditionWithJoin(gdo).OfType<Service>().ToList<Service>();
16         }
17     }
18 }
19
```

Слика 218 - Код CO Нађи услуге

Услуге је могуће претражити преко назива, опис и категорије кјој припадају. Апликација подржава и претрагу преко комбинације побројаних критеријума. У доменском објекту Услуга који се прослеђује да би се системска операција извршила, чувају се критеријуми претраге у зависности од тога који критеријум претрага је клијент унео. Наведено је имплементирано у методи *ReturnConditionForWhereInJoin()* где се у зависности од наредбе *if* која се извршава креира *where* клаузула за генеричку методу *ReturnForConditionWithJoin(gdo)* која се позива за дату системску операцију.

```
public string ReturnConditionForWhereInJoin()
{
    if (this.name == null && this.category == null) return "s.Active = 1";
    if (this.serviceID>0) return "s.Active = 1 and s.ServiceID = "+this.serviceID+"";
    if (this.name == "" && this.category!=null) return "s.Active = 1 and s.CategoryID = "+this.category.CategoryID+"";
    if (this.name != "" && this.category == null) return "s.Active = 1 and" +
        "(s.Name like '%" + this.name + "%' or s.Description like '%" + this.name + "%')";
    return "s.Active = 1 and (s.Name like '%" + this.name + "%' or s.Description like '%" + this.name + "%') and" +
        "s.CategoryID = "+this.category.CategoryID+"";
}
```

Слика 219 - Имплементација могућности претраге услуга по различитим критеријумима

5.3.1.9. CO UcitajUslugu

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.ServiceSO
10 {
11     public class SearchServices : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().ReturnForConditionWithJoin(gdo).OfType<Service>().ToList<Service>();
16         }
17     }
18 }
19
```

Слика 220 - Код СО Учитај услугу

5.3.1.10. CO IzmeniUslugu

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.ServiceSO
10 {
11     public class EditService : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().Update(gdo);
16         }
17     }
18 }
19
```

Слика 221 - Код СО Измени услугу

5.3.1.11. CO VratiUsluge

```
C# SystemOperations
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.ServiceSO
10 {
11     public class SearchServices : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().ReturnForConditionWithJoin(gdo).OfType<Service>().ToList<Service>();
16         }
17     }
18 }
19
```

Слика 222 - Код СО Врати услуге

5.3.1.12. CO VratiNacinePlacanja

```
C# SystemOperations
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.PaymentMethodSO
10 {
11     public class ReturnPaymentMethods : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().ReturnAll(gdo).OfType<PaymentMethod>().ToList<PaymentMethod>();
16         }
17     }
18 }
19
```

Слика 223 - Код СО Врати начине плаћања

5.3.1.13. CO KreirajRacun

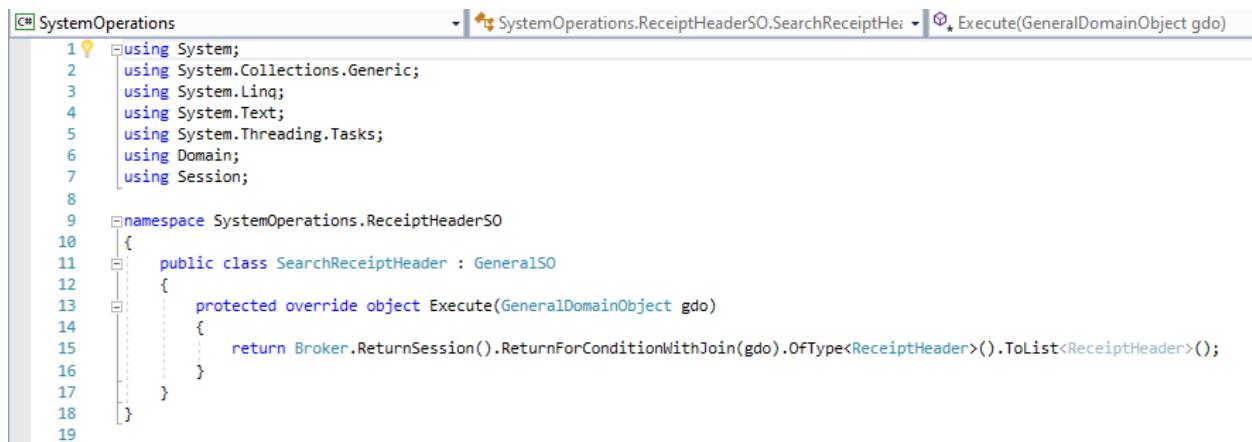
The screenshot shows a code editor window with the following details:

- Project: SystemOperations
- Class: SystemOperations.ReceiptHeaderSO.SaveReceiptHead
- Code Content:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.ReceiptHeaderSO
10 {
11     public class SaveReceiptHeader : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             ReceiptHeader rh = (ReceiptHeader)gdo;
16             Object o = Broker.ReturnSession().Save(gdo);
17             if (o != null)
18             {
19                 rh.receiptID = Broker.ReturnSession().ReturnNumber(gdo);
20
21                 foreach (Domain.ReceiptLine line in rh.receiptLines)
22                 {
23                     line.ReceiptHeader = rh.receiptID;
24                     o = Broker.ReturnSession().Save(line as GeneralDomainObject);
25                     if (o == null) return null;
26                 }
27             }
28             return o;
29         }
30     }
31 }
32 }
```

Слика 224 - Код СО Креирај рачун

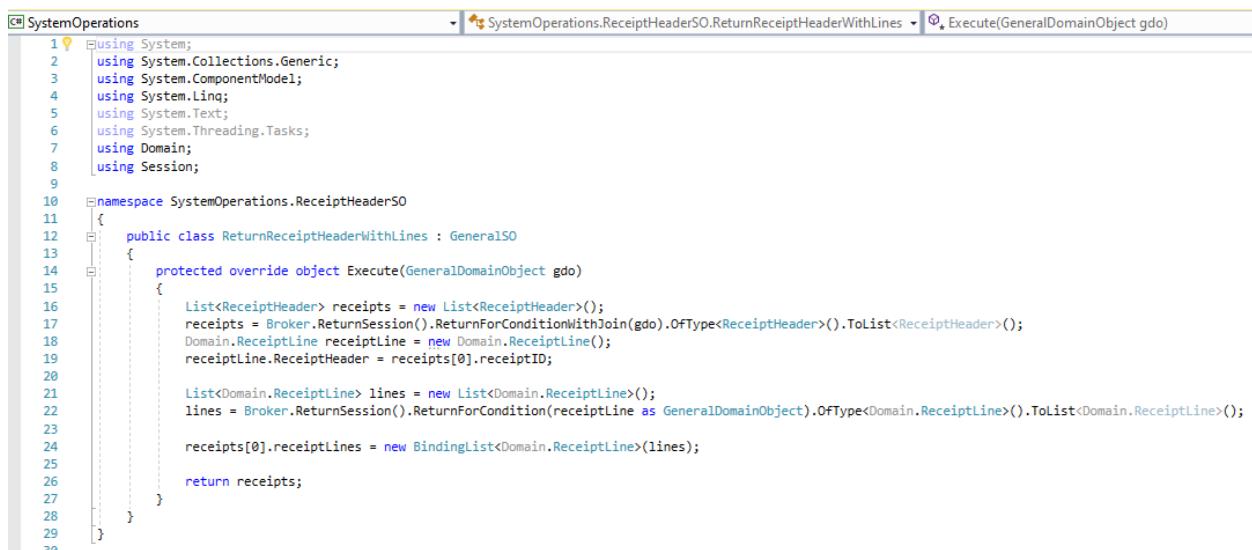
5.3.1.14. CO NadjiRacun



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using Domain;
7 using Session;
8
9 namespace SystemOperations.ReceiptHeaderSO
10 {
11     public class SearchReceiptHeader : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().ReturnForConditionWithJoin(gdo).OfType<ReceiptHeader>().ToList<ReceiptHeader>();
16         }
17     }
18 }
19
```

Слика 225 - Код СО Наджи рачун

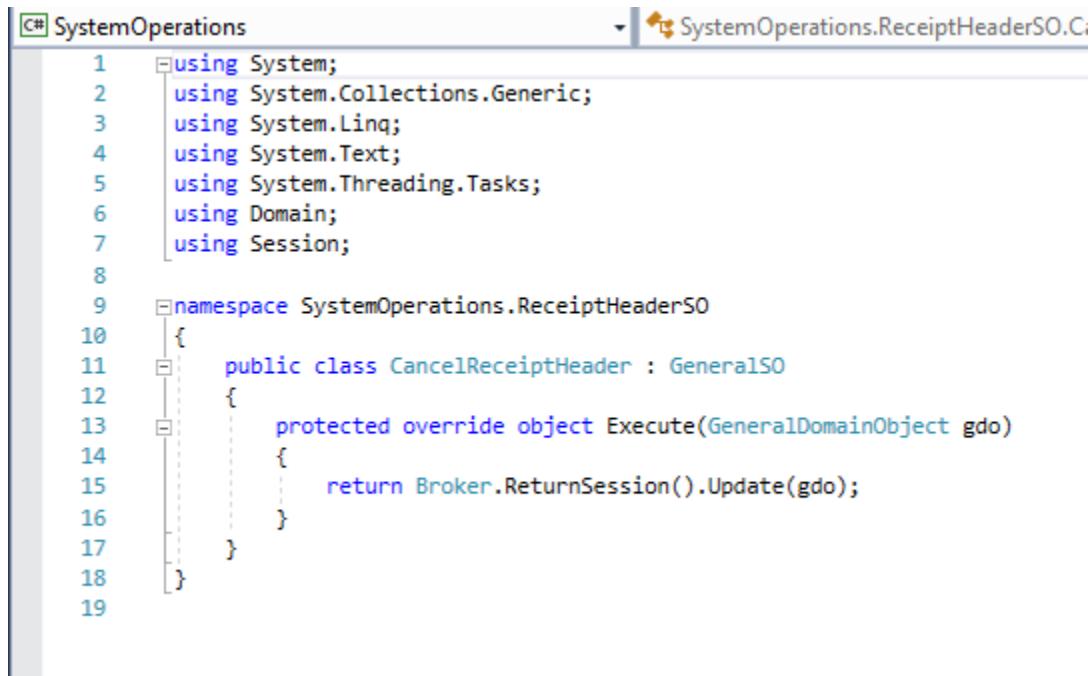
5.3.1.15. CO UcitajRacun



```
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Linq;
5 using System.Text;
6 using System.Threading.Tasks;
7 using Domain;
8 using Session;
9
10 namespace SystemOperations.ReceiptHeaderSO
11 {
12     public class ReturnReceiptHeaderWithLines : GeneralSO
13     {
14         protected override object Execute(GeneralDomainObject gdo)
15         {
16             List<ReceiptHeader> receipts = new List<ReceiptHeader>();
17             receipts = Broker.ReturnSession().ReturnForConditionWithJoin(gdo).OfType<ReceiptHeader>().ToList<ReceiptHeader>();
18             Domain.ReceiptLine receiptline = new Domain.ReceiptLine();
19             receiptline.ReceiptHeader = receipts[0].receiptID;
20
21             List<Domain.ReceiptLine> lines = new List<Domain.ReceiptLine>();
22             lines = Broker.ReturnSession().ReturnForCondition(receiptline as GeneralDomainObject).OfType<Domain.ReceiptLine>().ToList<Domain.ReceiptLine>();
23
24             receipts[0].receiptLines = new BindingList<Domain.ReceiptLine>(lines);
25
26             return receipts;
27         }
28     }
29 }
```

Слика 226 - Код СО Учитај рачун

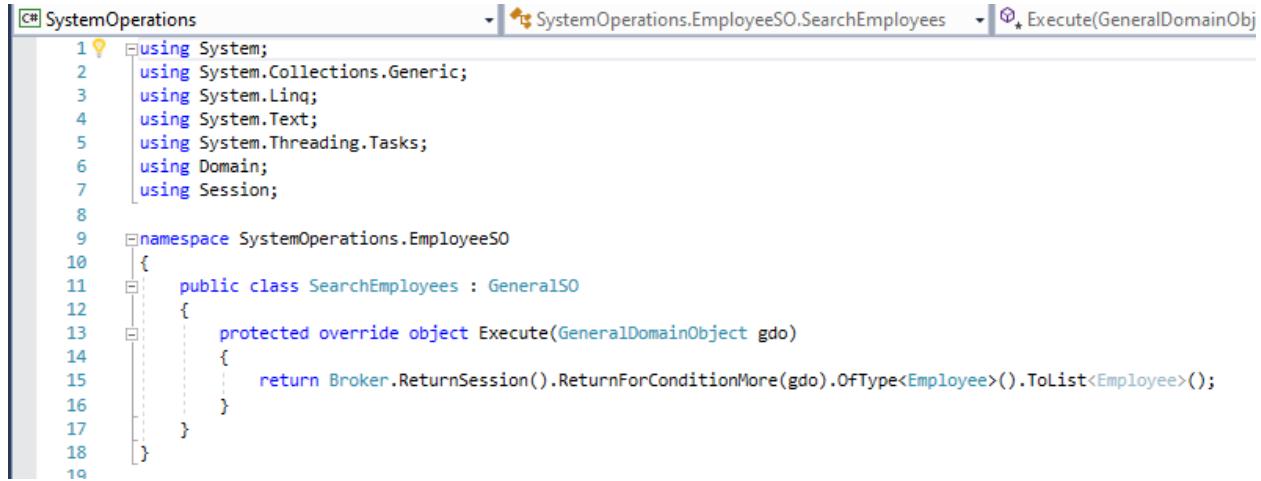
5.3.1.16. CO StornirajRacun



```
C# SystemOperations
  1  using System;
  2  using System.Collections.Generic;
  3  using System.Linq;
  4  using System.Text;
  5  using System.Threading.Tasks;
  6  using Domain;
  7  using Session;
  8
  9  namespace SystemOperations.ReceiptHeaderSO
 10 {
 11     public class CancelReceiptHeader : GeneralSO
 12     {
 13         protected override object Execute(GeneralDomainObject gdo)
 14         {
 15             return Broker.ReturnSession().Update(gdo);
 16         }
 17     }
 18 }
 19
```

Слика 227 - Код СО Сторнирај рачун

5.3.1.17. CO NadjiZaposlene



```
C# SystemOperations
  1  using System;
  2  using System.Collections.Generic;
  3  using System.Linq;
  4  using System.Text;
  5  using System.Threading.Tasks;
  6  using Domain;
  7  using Session;
  8
  9  namespace SystemOperations.EmployeeSO
 10 {
 11     public class SearchEmployees : GeneralSO
 12     {
 13         protected override object Execute(GeneralDomainObject gdo)
 14         {
 15             return Broker.ReturnSession().ReturnForConditionMore(gdo).OfType<Employee>().ToList<Employee>();
 16         }
 17     }
 18 }
```

Слика 228 - Код СО Нађи запослене

5.3.1.18. CO UcitajZaposlenog

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.EmployeeSO
10 {
11     public class ReturnSelectedEmployee : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().ReturnOneForID(gdo) as Employee;
16         }
17     }
18 }
19
```

Слика 229 - Код СО Учитај запосленог

5.3.1.19. CO KreirajZaposlenog

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.EmployeeSO
10 {
11     public class SaveEmployee : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().Save(gdo);
16         }
17     }
18 }
19
```

Слика 230 - Код СО Креирај запосленог

5.3.1.20. CO DeaktivirajZaposlenog

```
C# SystemOperations
SystemOperations.EmployeeSO.Edit
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.EmployeeSO
10 {
11     public class EditEmployee : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().Update(gdo);
16         }
17     }
18 }
19
```

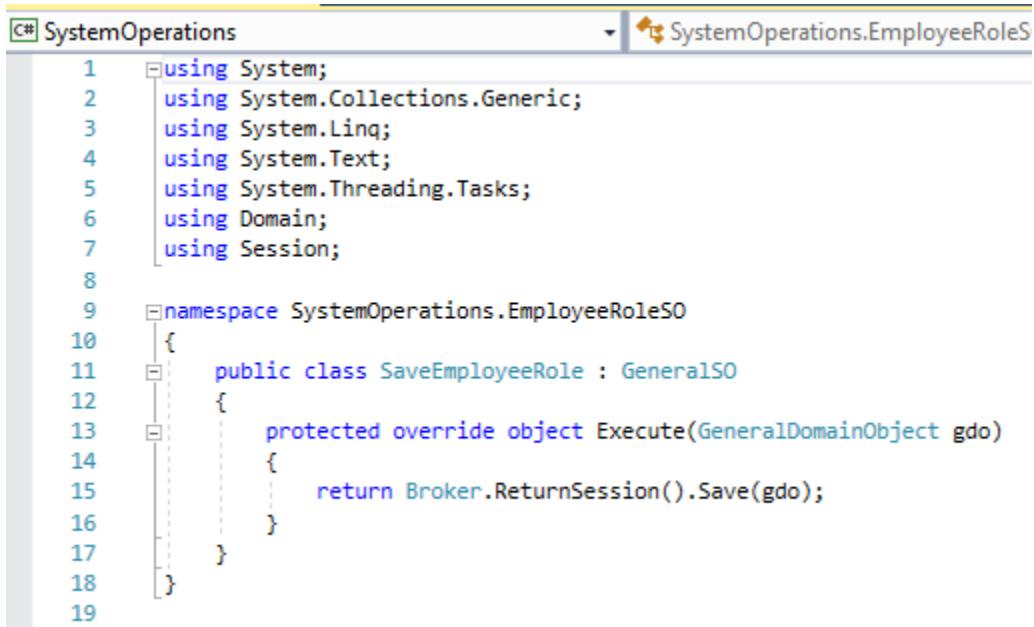
Слика 231 - Код СО Деактивирај запосленог

5.3.1.21. CO VratiUloge

```
C# SystemOperations
SystemOperations.RoleSO.ReturnRoles
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.RoleSO
10 {
11     public class ReturnRoles : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().ReturnAll(gdo).OfType<Role>().ToList<Role>();
16         }
17     }
18 }
19
```

Слика 232 - Код СО Врати улоге

5.3.1.22. CO KreirajVezuUZ



```
c# SystemOperations
SystemOperations.EmployeeRoleS
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Domain;
7  using Session;
8
9  namespace SystemOperations.EmployeeRoleS
10 {
11     public class SaveEmployeeRole : GeneralSO
12     {
13         protected override object Execute(GeneralDomainObject gdo)
14         {
15             return Broker.ReturnSession().Save(gdo);
16         }
17     }
18 }
19
```

Слика 233 - Код CO Креирај везу улога - запослен

5.4. Имплементација екранских форми

5.4.1.1. Главна екранска форма

```
using Domain;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Client
{
    public partial class Form1 : Form
    {
        private ControllerUI ui;
        public Form1(ControllerUI ui)
        {
            InitializeComponent();
            this.ui = ui;
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            ui.CheckRoles(employeeToolStripMenuItem, newLoyaltyToolStripMenuItem);
            ui.SetDateForNewReceipt(textBoxReceiptNumber, textBoxReceiptDate,
comboBoxPaymentMethods, dataGridViewReceiptLines);

        }

        private void MenuStrip1_ItemClicked(object sender, ToolStripItemClickedEventArgs e)
        {
        }

        private void NewClientToolStripMenuItem_Click(object sender, EventArgs e)
        {
            new NewClientForm(ui).ShowDialog();
        }

        private void NewServiceToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            new NewServiceForm(ui).ShowDialog();
        }

        private void NewLoyaltyToolStripMenuItem_Click(object sender, EventArgs e)
        {
            new NewLoyaltyForm(ui).ShowDialog();
        }

        private void NewEmployeeToolStripMenuItem_Click(object sender, EventArgs e)
```

```

{
    new NewEmployeeForm(ui).ShowDialog();
}

private void SearchToolStripMenuItem1_Click(object sender, EventArgs e)
{
    new SearchEmployeeForm(ui).ShowDialog();
}

private void SearchToolStripMenuItem2_Click(object sender, EventArgs e)
{
    new SearchLoyalty(ui).ShowDialog();
}

private void NewServiceToolStripMenuItem_Click(object sender, EventArgs e)
{
    new SearchServiceForm(ui).ShowDialog();
}

private void SearchToolStripMenuItem_Click(object sender, EventArgs e)
{
    new SearchClientForm(ui).ShowDialog();
}

private void AddRoleToolStripMenuItem_Click(object sender, EventArgs e)
{
    new SearchEmployeeForm(ui).ShowDialog();
}

private void NewReceiptToolStripMenuItem_Click(object sender, EventArgs e)
{
    new NewReceiptForm(ui).ShowDialog();
}

private void NewReceiptToolStripMenuItem1_Click(object sender, EventArgs e)
{
    new CancelReceiptForm(ui).ShowDialog();
}

private void ButtonFindClient_Click(object sender, EventArgs e)
{
    new ClientForReceiptForm(ui).ShowDialog();
    ui.CheckIfClientExist(textBoxClient, labelClient, buttonFindClient,
textBoxAmount, textBoxDiscountAmount);
    dataGridViewReceiptLines.Refresh();
}

private void Button1_Click(object sender, EventArgs e)
{
    new NewReceiptLineForm(ui).ShowDialog();
    textBoxAmount.Text = ui.receiptHeader.Amount.ToString();
    textBoxDiscountAmount.Text = ui.receiptHeader.discount.ToString();
}

private void ButtonRemoveRL_Click(object sender, EventArgs e)
{
    ui.RemoveReceiptLine(dataGridViewReceiptLines);
    textBoxAmount.Text = ui.receiptHeader.Amount.ToString();
    textBoxDiscountAmount.Text = ui.receiptHeader.discount.ToString();
}

```

```

        }

        private void ButtonSave_Click(object sender, EventArgs e)
        {
            ui.SaveReceipt(textBoxReceiptNumber, textBoxReceiptDate,
comboBoxPaymentMethods, textBoxClient, textBoxNote, textBoxAmount,
dataGridViewReceiptLines, textBoxDiscountAmount, labelClient, buttonFindClient, this);
        }
    }
}

```

У методи *Form1_Load* која служи за учитавање дате форме, позива се метода *CheckRoles*. Ова метода је имплементирана у класи *ControllerUI*. Метода проверава да ли улоговани корисник има *ADMIN* улогу и у зависности од тога омогућава кориснику коришћење свих или само одређеним функционалности апликације.

```

internal void CheckRoles(ToolStripMenuItem employeeToolStripMenuItem,
ToolStripMenuItem newLoyaltyToolStripMenuItem)
{
    if (adminRoleExist)
    {
        employeeToolStripMenuItem.Visible = true;
        newLoyaltyToolStripMenuItem.Visible = true;
    }
}

```

Објекат *adminRoleExist* из наведене методе се попуњава у тренутку када се корисник улогује.

```

internal void FindEmployee(TextBox textBoxUsername, TextBox textBoxPassword)
{
    string mandatoryValues = "Please enter:";
    bool thereIsEmptyValue = false;

    if (string.IsNullOrWhiteSpace(textBoxUsername.Text)) {
        if (mandatoryValues.Count() == 13) mandatoryValues = mandatoryValues +
"username";
        else mandatoryValues = mandatoryValues + ",username";
        textBoxUsername.BackColor = Color.MistyRose;
        thereIsEmptyValue = true;
    }

    if (string.IsNullOrWhiteSpace(textBoxPassword.Text))
    {
        if (mandatoryValues.Count() == 13) mandatoryValues = mandatoryValues +
"password";
        else mandatoryValues = mandatoryValues + ",password";
        textBoxPassword.BackColor = Color.MistyRose;
        thereIsEmptyValue = true;
    }

    if (thereIsEmptyValue)
    {
        MessageBox.Show(mandatoryValues + "!");
        return;
    }
}

```

```

List<Employee> employees = new List<Employee>();
Employee emplFromUI = new Employee();
emplFromUI.Username = textBoxUsername.Text;
emplFromUI.Password = textBoxPassword.Text;
employees = c.FindEmployee(emplFromUI);
if (employees == null || employees.Count() == 0)
{
    MessageBox.Show("The system cannot find the user based on the entered
values! Check the entered data!");
    return;
}
else
{
    e = employees[0];
    EmployeeRole emplRole = new EmployeeRole();
    emplRole.Employee = e;
    er = c.ReturnRolesForEmployee(emplRole);
    adminRoleExist = FindAdminRole();
    MessageBox.Show("You have successfully logged into the system!");
    new Form1(this).ShowDialog();
}

```

Метода *FindAdminRole()* проверава да ли улоговани корисник има *ADMIN* улогу.

```

public bool FindAdminRole()
{
    foreach (EmployeeRole employeeRole in er)
    {
        if (employeeRole.RoleName == adminRole)
        {
            return true;
        }
    }
    return false;
}

```

5.4.1.2. Форма Претрага услуга

```
using Domain;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Client
{
    public partial class SearchServiceForm : Form
    {
        ControllerUI ui;
        public SearchServiceForm(ControllerUI ui)
        {
            InitializeComponent();
            this.ui = ui;
        }

        private void ButtonSearchService_Click(object sender, EventArgs e)
        {
            ui.SearchServices(textBoxSearchService, comboBoxCategory,
dataGridViewService);
        }

        private void ButtonShowService_Click(object sender, EventArgs e)
        {
            ui.ReturnSelectedService(dataGridViewService);
        }

        private void SearchServiceForm_Load(object sender, EventArgs e)
        {
            ui.ReturnCategories(comboBoxCategory);
            ui.ReturnAllServices(dataGridViewService);
        }

        private void ButtonResetCategory_Click(object sender, EventArgs e)
        {
            comboBoxCategory.SelectedItem = null;
        }
    }
}
```

ButtonSearchService_Click је метода која се позива кликом на дугме за претрагу клијената (*Find services*). У овој методи се позива метода *SearchServices*. Ова метода је имплементирана у класи *ControllerUI*.

```

internal void SearchServices(TextBox textBoxSearchService, ComboBox
comboBoxCategory, DataGridView dataGridViewService)
{
    List<Service> services = new List<Service>();
    if (textBoxSearchService.Text.Count() == 0 && comboBoxCategory.SelectedItem
== null)
    {
        MessageBox.Show("Please enter a search value!");
        return;
    }

    if (textBoxSearchService.Text.Count() > 0 &&
textBoxSearchService.Text.Count() <= 3)
    {
        MessageBox.Show("You need to enter at least four characters for
search!");
        return;
    }

    Service s = new Service();
    s.Name = textBoxSearchService.Text;
    s.category = comboBoxCategory.SelectedItem as Category;
    services = c.SearchServices(s);

    if (services == null || services.Count == 0)
    {
        MessageBox.Show("No service found!");
        dataGridViewService.DataSource = null;

        return;
    }
    else
    {
        MessageBox.Show("The search is complete!");
        dataGridViewService.DataSource = new BindingList<Service>(services);
    }
}

```

6. Тестирање

Сваки од поменутих имплементираних случајева коришћења је тестиран током развоја овог софтверског система. Приликом тестирања сваког случаја коришћења поред унетих тачних података уношени су и неправилни подаци, како би се утврдило какав ће резултат извршења бити. На основу извршених тестирања отклоњене су све уочене грешке.

Поред тестирања случајева коришћења у овој фази развоја софтвера сам се трудали да што више узмем у обзир и корисничко искуство. Желела сам да олакшам коришћење апликације и апликацију прилагодим корисницима. Приликом овог типа тестирања уочила сам пар случајева коришћења апликације које сам покушала да што реалније модификујем како би цео систем био што реалнији. Следећи случајеви су модификовани да би се олакшало коришћење апликације:

1. Уколико корисник није попунио неко обавезно поље, а кликнуо је не дугме који позива системску операцију, приказаће се порука са информацијом које поље је потребно да се попуни и поље ће бити обојено светло црвеном бојом. Када корисник почне да уноси неку вредност у дато поље, нестаће црвена боја што је индикатор да је корисник предузео одговарајућу, жељену акцију.
2. Код поља где је потребно да се унесе бројчана вредност, одрађена је валидација која уколико је поље погрешно попуњено приказује корисику поруку да унесе бројчану вредност како би могао да изврши жељену акцију.
3. За унос датума који се односи на крај рада или период до (означава датум до) одрађен је checkbox који представља индикатор да корисник жели да унесе дати датум. Ово је омогућено како би били обухваћени и случајеви када корисник не зна или не жели да унесе дати датум.
4. Код уноса описа постоји ограничење у броју карактера које је могуће унети. Уколико се унесе више од 255 карактера кориснику се прикаже порука да је унео више од могућег броја карактера. Овим је избегнуто неуспешно извршење системске операције и корисник је обавештен о могућој грешци тако да је могућа акција са његове стране како би се грешка избегла.

Без обзира на жељу да се корисницима омогући што лакше коришћење апликације, од корисника се очекује да прочитају упутство о коришћењу апликације и да имају обуку како би што успешније користили апликацију. У овој фази сам се трудали да омогућим што лакше коришћење али да избегнем механичко коришћење апликације (без размишљања) од стране корисника јер колико год да је апликација једноставна такав начин коришћења апликације доводи до погрешних акција од стране корисника.

7. Закључак

У овом примеру приказане су све фазе развоја софтвера. Свака фаза је подједнако значајна и за развој и за одржавање апликације.

Замислила сам систем за који желим да креирајем софтвер. Тако замишљен систем сам разделила на целине од којих су настали случајеви коришћења. Трудила сам се да што реалније опишем случајеве коришћења, како бих олакшала себи касније кораке. У овом тренутку сам скицирала и прву верзију корисничког интерфејса како бих била сигурна да сам дефинисала потребне све случајеве коришћења.

Добро дефинисани случајеви коришћења значајно су ми олакшали све даље фазе развоја софтвера.

У фази пројектовања, највећи изазов ми је био да добро поставим и разумем архитектуру, пошто до сада нисам имала прилике да то радим. Постављена архитектура ми је помогла у имплементацији понашања софтвера јер сам многе методе могла лако да прилагодим коришћењу различитих класа. Након што сам имплементирала прве системске операције за инсерт, едит и за враћање података из базе све наредне методе сам имплементирала доста лакше и уз јако мало измена. Овде сам увидела огромне предности добре архитектуре.

У фази тестирања сам се трудила да што више узмем у обзир корисничко искуство, како бих олакшала коришћење апликације. Потрудила сам да систем буде што реалнији, како би задовољио потребе стварних корисника.

8. Литература

1. Др Синиша Влајић, (2020) *Пројектовање софтвера – Скрипта*
2. Pradeep Tapadiya, (2002) *.NET programing a particular guide using C#*
3. Ms. Poonam Verma, *Dot Net Technology Notes (BCA 602) BCA VI UNIT I & UNIT II*
4. Karnataka state open university, department of studies in information technologies, (2014) *MSIT-120 Dot Net Technologies*
5. Jozeph Albahari, *Threading in C#,* последњи пут ажурирано 2011-04-07.
6. David B. Makofske, Michael J. Donahoo, Kenneth L. Calvert, (2004) *TCP/IP Socket in C# Particular guide for*
7. *.NET fundamentals documentation,* доступно онлине на <https://learn.microsoft.com/en-us/dotnet/fundamentals/>
8. *.NET Framework архитектура,* доступно онлине на <https://www.radlovacki.com/net-framework-architektura/>
9. *Рад са нитима у C# програмској језику,* доступно онлине на:
<https://www.manuelradovanovic.com/2016/04/rad-sa-nitima-u-c-programskom-jeziku.html>
10. *.NET,* доступно онлине на <https://en.wikipedia.org/wiki/.NET>
11. *.NET Framework,* доступо онлине на https://sr.wikipedia.org/sr-ec/.NET_Framework
12. *C#,* доступо онлине на https://sr.wikipedia.org/sr-ec/C_Sharp
13. *Увод у C# програмски језик,* доступно онлине на
<https://www.manuelradovanovic.com/2015/11/uvod-u-c-programski-jezik.html>
14. *.NET Framework vs .NET Core vs .NET vs .NET Standard vs C#,* видео доступан онлине на
<https://www.youtube.com/@IAmTimCorey/videos>
15. *CS 60 Computer Networks – Lecture 3 and 4 – Socket Programming*
<https://www.cs.dartmouth.edu/~campbell/cs60/socketprogramming.html>
16. *MVC архитектура,* https://sr.wikipedia.org/wiki/MVC_arhitektura