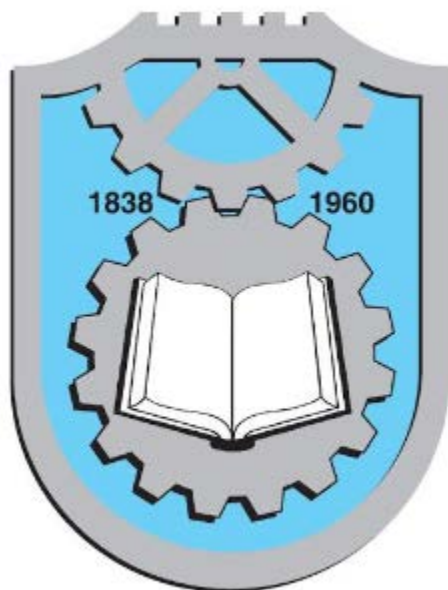


Универзитет у Крагујевцу
Факултет инжењерских наука



Пројекат:
Стоматолошка ординација

Студент:

Урош Милошевић 628/2019

Предметни професор:

Ненад Филиповић

Садржај

Увод.....	3
Опис пројекта.....	3
Опис кода	14
Функције администратора	15
Функције техничара.....	17
Функције стоматолога	18
Генерисање извештаја	19
UML Дијаграми.....	20
Use Case дијаграм	20
Дијаграм класа	23
Дијаграм редоследа.....	24
Дијаграм активности.....	25
Дијаграм стања	26
Литература	27

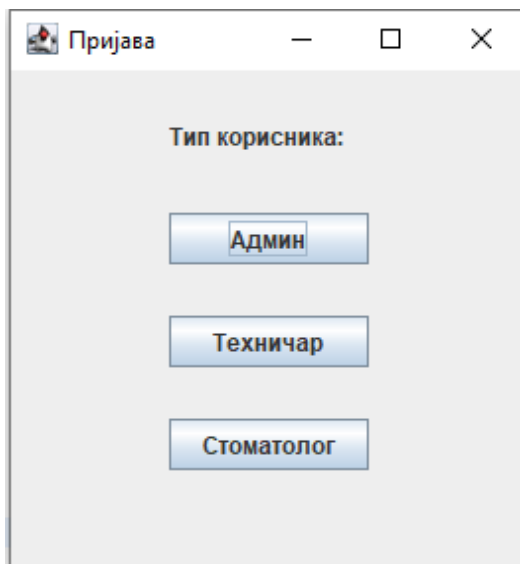
Увод

Стоматолошка ординација има потребу за уредном евиденцијом својих клијената. Софтвер нуди решење које ради у сврси чувања података о клијентима. У зависности од привилегија датог корисника, корисник има могућност креирања листе пацијената, заказивање прегледа, сам преглед заказаних термина, корисник може уносити детаље прегледа и генерисати извештаје. Поред основног решења софтвер нуди и додатне опције за организацију особља, односно одређивања привилегија корисницима. Софтвер нуди и основне методе превенције неадекватних уноса и контролу грешака.

Опис пројекта

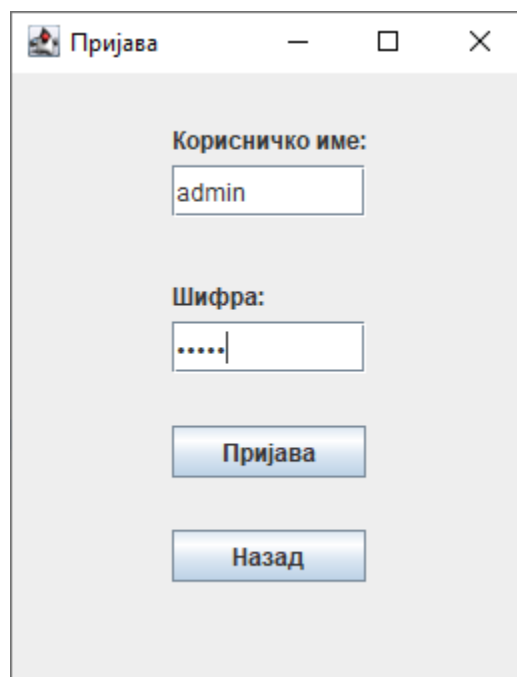
Пројектни задатак представља апликацију креирану у програмском језику Јава. Апликација функционише по принципу различитих типова корисника и њихових привилегија. Као резултат целокупног пројекта добијамо систем за управљање стоматолошком клиником. Пројекат садржи фолдер „датотеке“ који садржи фолдере „login“, „стоматолог“ и „техничар“ који даље садрже текстуалне датотеке за пријављивање корисника, фолдере са извештајима прегледа пацијената и као и листу пацијента и њихове заказане прегледе, респективно.

При покретању апликације кориснику се нуди одабир типа корисника.(слика 2.1.1)



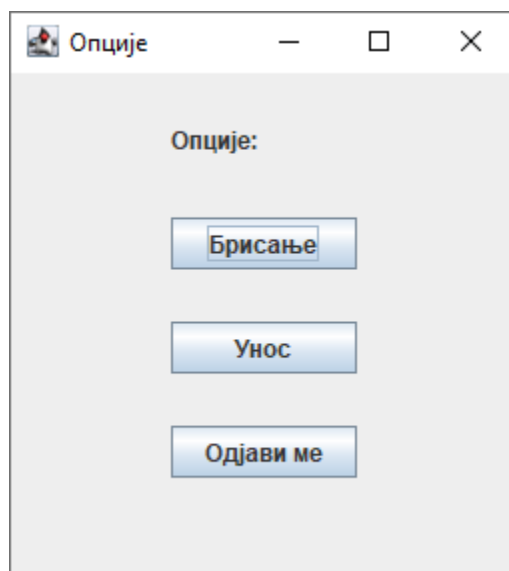
Слика 2.1.1 : Одабир корисника

Након одабира типа корисника, од корисника се захтева ферификација идентитета у виду потврде корисничког имена и лозинке.(Слика 2.2) Почећемо са описом администраторске функције. Подразумевни подаци за пријаву администратора система су **admin/admin**.



Слика 2.2 : Пријављивање

Кликом на дугме „Назад“ се враћамо на почетни мени док клик на дугме „Пријава“ врши верификацију корисника и прелазак на следећи мени.



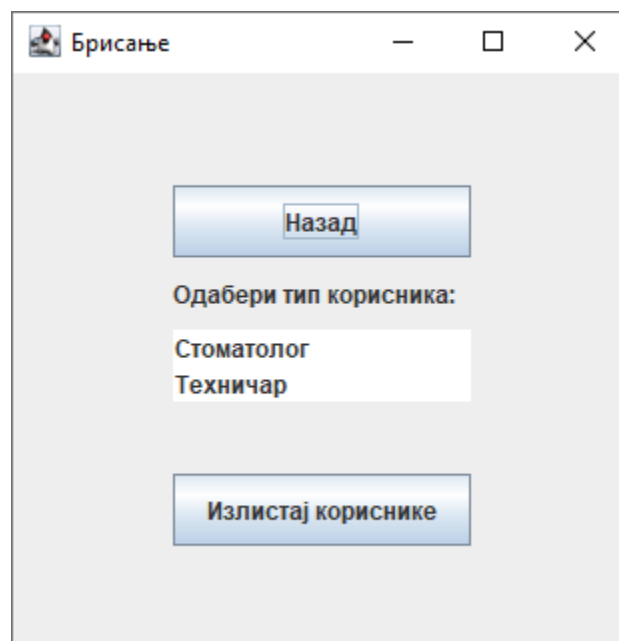
Слика 2.3: Опције
администратора

Опције које се нуде администратору су: (Слика 2.3)

- Брисање корисника из система
- Унос нових корисника у систем

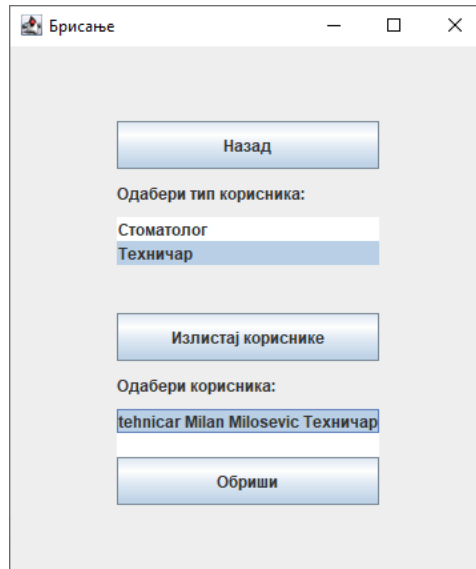
Трећа понуђена опција је одјављивање из апликације.

Кликом на дугме „Брисање“ се приказује мени са листом типова корисника који се налазе у систему.(Слика 2.4)



Слика 2.4 : Одабир типа корисника

Кликом на одређени тип корисника вршимо одабир типа корисника који желимо да излистамо док кликом на дугме „Изслистај кориснике“ добијамо у наставку прозора излистане кориснике и дугме „Обриши“ које врши функцију брисања корисника из система. Након извршеног брисања остајемо у истом менију. (Слика 2.5)

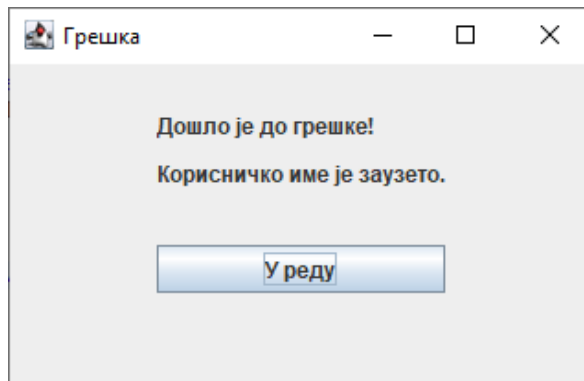


Слика 2.5 : Листа корисника

Одабир корисника и потврда брисања се врши на исти начин као горе описани процес одабира типа корисника и изслиставања истих.

Коришћење опције уноса нових корисника функционише на следећи начин.

Попуњавањем текстуалних улаза за основне податке (Корисничко име, шифра, име, презиме и уже области рада у случају стоматолога) се уносе подаци. Из листе „Функција“ је кориснику понуђена селекција доступних функција у ординацији. Кликом на функцију „Стоматолог“ се поље „Ужа област рада“ откључава за унос текста док је у почетку то поље онемогућено. (Слика 2.6.1) У склопу система је доступна провера доступности корисничких имена и уколико је дошло до преклапања корисничких имена добијамо следећу проуку: (Слика 2.6.2)



Слика 2.6.2 : Грешка/заузето к.
име

И систем на враћа на почетни мени администратора.

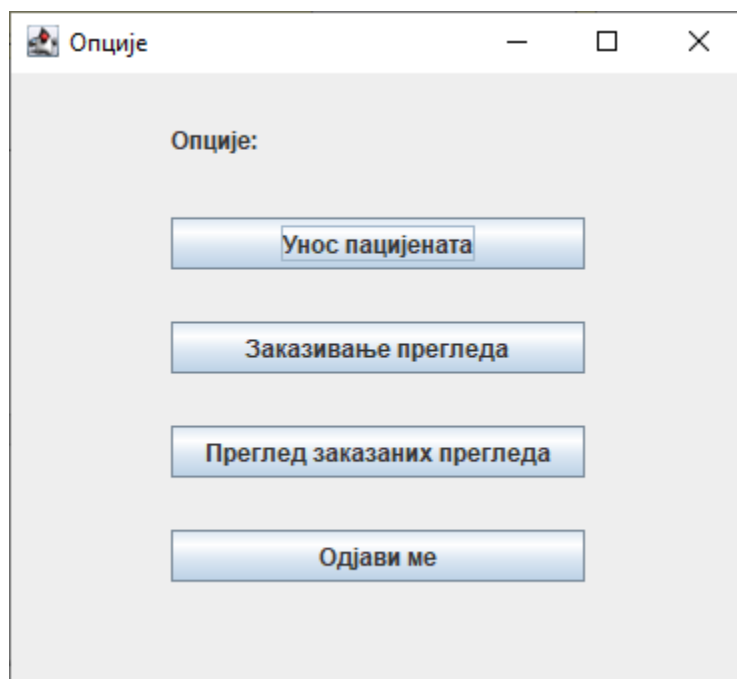
Након уноса коректних података нови корисник се чува у базу података и систем нас враћа на почетни мени администратора.

The 'Унос' (Entry) window contains the following elements:

- Корисничко име:** Text input field.
- Шифра:** Text input field.
- Име:** Text input field.
- Презиме:** Text input field.
- Функција:** A list box with two options: 'Стоматолог' (Dentist) and 'Техничар' (Technician). 'Стоматолог' is currently selected.
- Ужа област рада:** Text input field, which is disabled when 'Техничар' is selected.
- Унеси** (Add): Button to save the new user.
- Назад** (Back): Button to return to the previous screen.

Слика 2.6.1 : Унос корисника

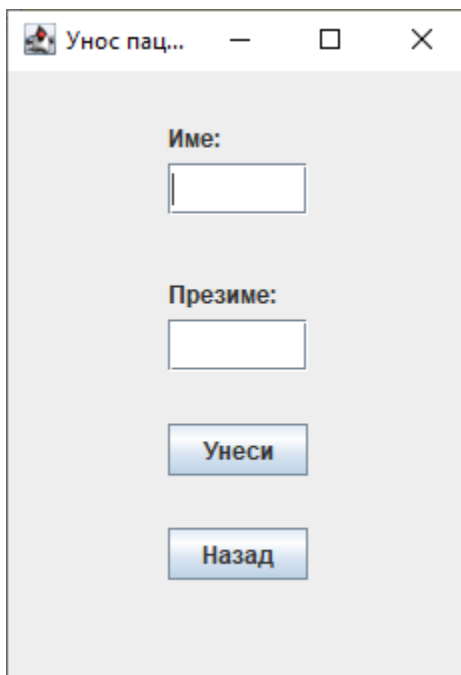
Корисник „Техничар“ након уноса корисничког имена и шифре добија следећи мени:
(Подразумевни подаци за пријаву техничара система су **tehnicar/ tehnicar**)



Слика 2.7 : Опције техничара

Опције су приказане на слици 2.7.

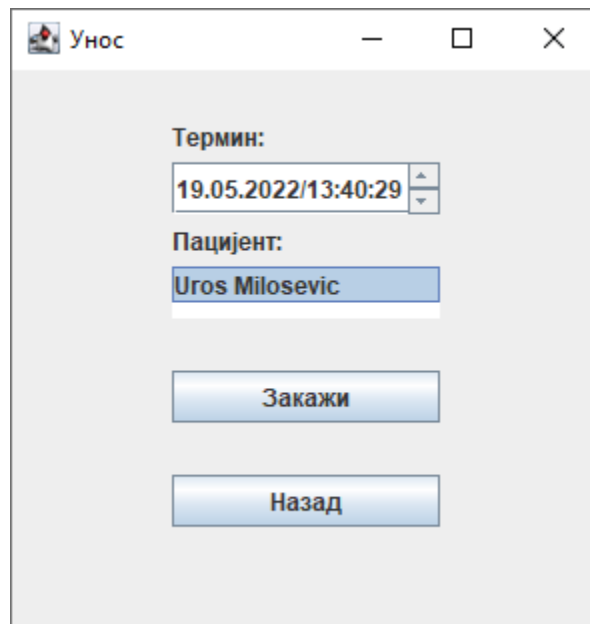
Кликом на дугме „Унос пацијената“ добијамо форму за унос нових пацијената.



Слика 2.8 : Унос пацијента

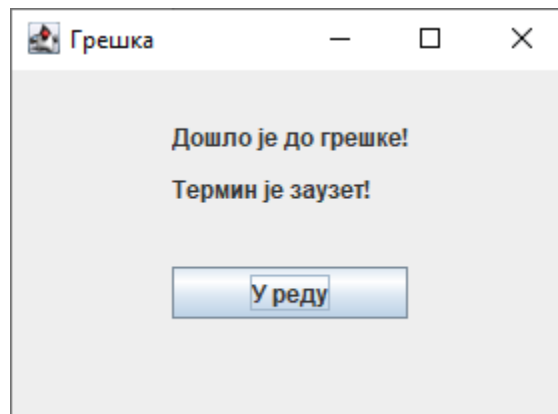
Након клика на дугме „Унеси“ нов пацијент је додат у базу података.(Слика 2.8)

Кликом на дугме „Заказивање прегледа“ дијамо мени са излистаним пацијентима из базе података и одабирачем термина. Кликом на пацијента из листе вршимо селекцију пацијента док означавањем конкретно дана/месеца/године или сата и минута добијамо могућност подешавања конкретног термина. Кликовима на стрелице са десне стране кутије добијамо померање термина. (Слика 2.9.1)



Слика 2.9.1 : Заказивање термина

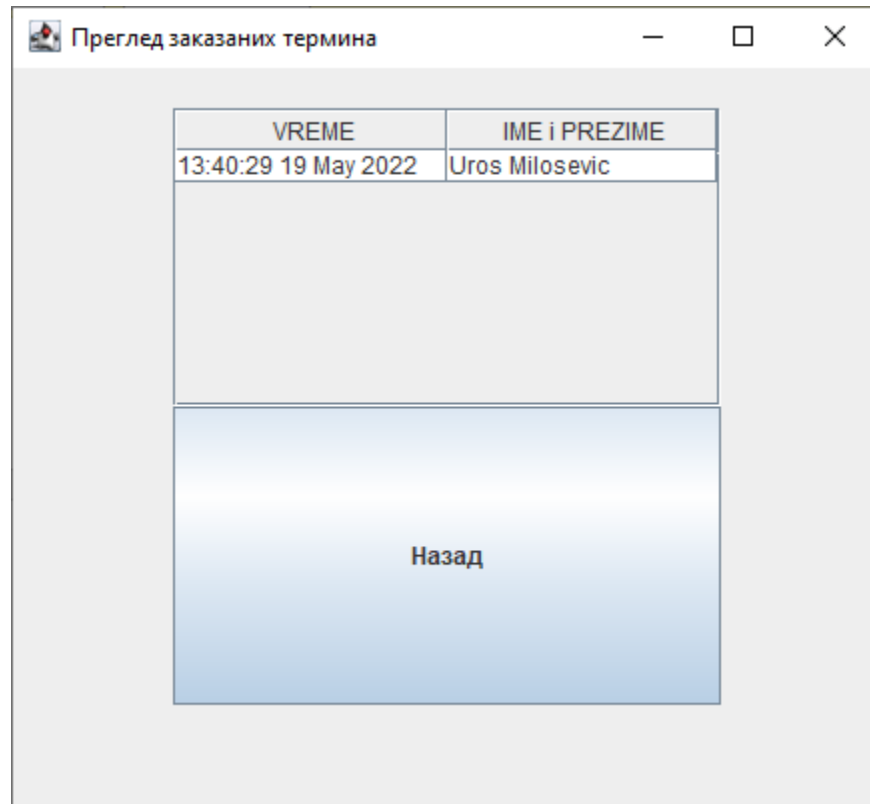
Кликом на дугме „Закажи“ се врши провера и унос у базу, уколико је термин слободан. У случају да јер термин заузет добијамо следећу поруку: (Слика 2.9.2)



Слика 2.9.2 : Грешка/термин заузет

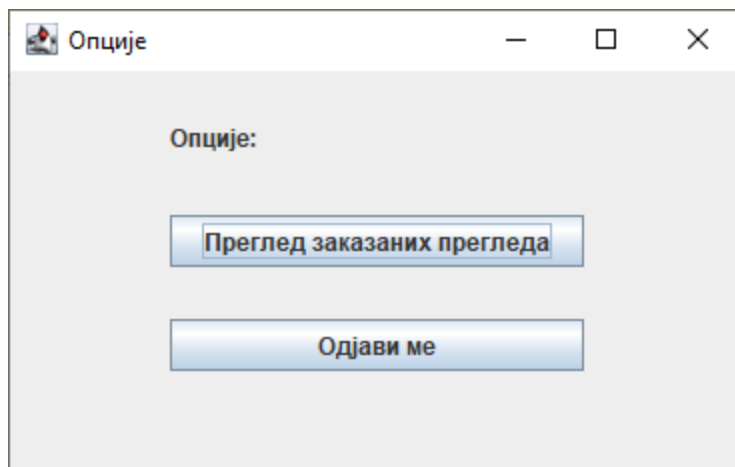
Систем нас у том случају враћа на почетни мени техничара.

Кликом на дугме „Преглед заказаних термина“ систем отвара прозор са табелом пацијената и њихових термина где се техничару даје на увид распоред прегледа. (Слика 2.10)



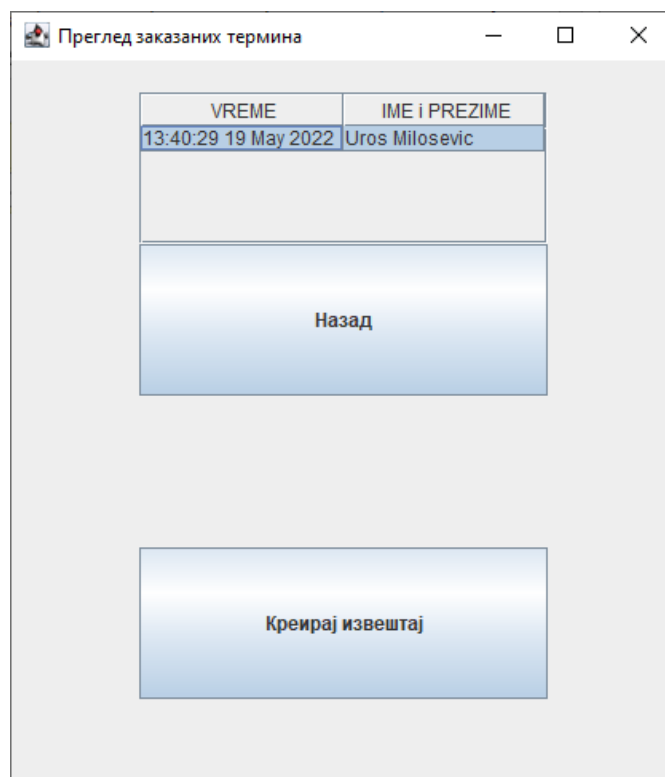
Слика 2.10 : Преглед термина

Корисник „Стоматолог“ након уноса корисничког имена и шифре добија следећи мени: (Подразумевни подаци за пријаву техничара система су **stomatolog/ stomatolog**) (Слика 2.11)

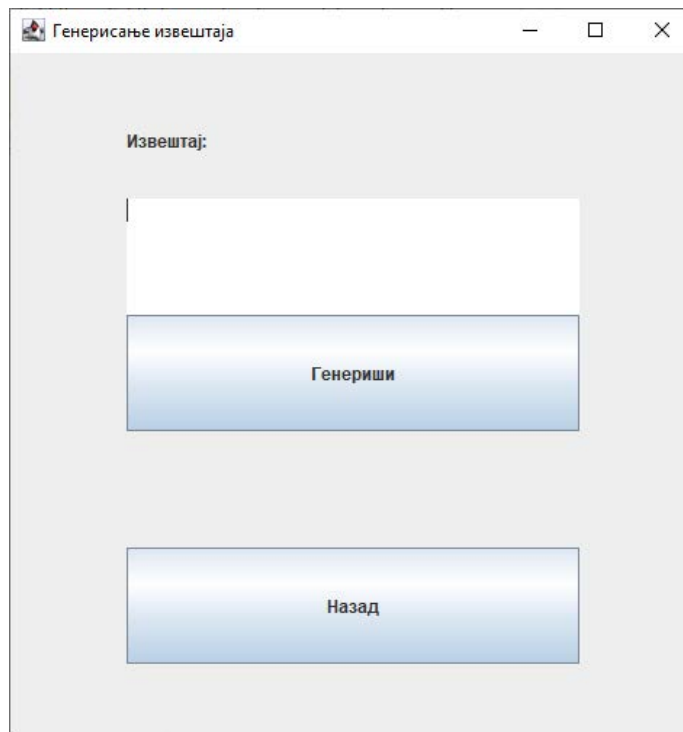


Слика 2.11 : Опције стоматолога

Након клика на „Преглед заказаних прегледа“ стоматолог добија листу заказаних. (Слика 2.12) Кликом на ред са термином и именом пацијента и дугмета „Креирај извештај“ стоматолог добија окружење за крерирање PDF извештаја о прегледу. (Слика 2.13)

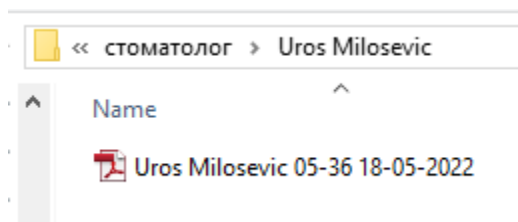


Слика 2.12 : Преглед термина са опцијом креирања извештаја



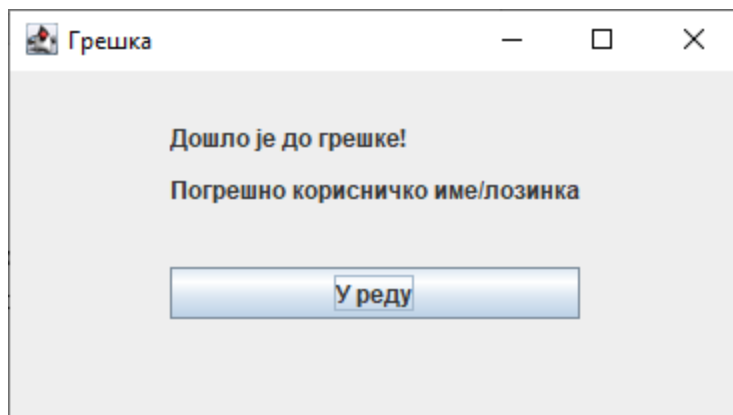
Слика 2.13 : Извештај

Након клика на дугме „Генериши“ фолдер са именом пацијента се креира, уколико већ не постоји, и ту се смешта PDF са датог прегледа. У фолдеру то изгледа овако: (Слика 2.14)



Слика 2.14 : Резултат креирања извештаја

У случају да корисник унесе погрешне податке за пријаву, корисник добија обавештење. (Слика 2.1.2)



Слика 2.1.2 : Погрешно унети
подаци за пријаву

Опис кода

За израду графичко корисничког интерфејса је коришћена библиотека `javax.swing` док је за генерисање PDF извештаја коришћен `itext 7.0.0`.

Пројекат садржи 12 класа и у даљем тексту ће бити описани кључни делови кода за функционисање система.

Свака класа заправо представља један од прозора пројекта, увоз `javax.swing` [1] библиотеке је поприлично једноставан и не захтеба никакве додатне операције сем базичне `import` комадне. (Слика 3.1)

```
import javax.swing.BorderFactory;
import javax.swing.Box;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JList;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
```

Слика 3.1 : Типична употреба
`javax.swing` библиотеке




Кретање кроз систем и извршавање метода је омогућено помоћу `ActionListener` класе `java.awt` [2] која се такође поприлично лако увози у пројекат.

Пример употребе `ActionListener`-а у пројекту: (Слика 3.2)

```
public void actionPerformed(ActionEvent e) {
    prozor.dispose();
    if(e.getSource() == button) {
        new prijava("Админ");
    }
    if(e.getSource() == button2) {
        new prijava("Техничар");
    }
    if(e.getSource() == button3) {
        new prijava("Стоматолог");
    }
}
```

Слика 3.2 : Метода која прихвата
догађај и одређује које дугме је
послало команду

Верификација корисника се врши провером унетих података са базом података. Базу података сачињавају текстуалне датотеке за сваки тип корисника. (Слика 3.3.2)

<< datoteke > login
Name ^
 админ
 Стоматолог
 Техничар

Слика 3.3.2 : Структура базе корисника

Излиставање корисника се врши читањем сваке линије из одређеног текстуалног документа а затим дељењем на бланко карактер како бисмо издвојили понаособ име, презиме,... Очитане делове убацујемо у нис, потом у листу која се приказује кориснику. [3] (Слика 3.4)

```
int i=0;
File myObj = new File("datoteke\\login\\" + tip + ".txt" );
Scanner myReader1 = new Scanner(myObj);
while (myReader1.hasNextLine()) {
    myReader1.nextLine();
    i++;
}
myReader1.close();

String osobe[] = new String[i];
int b =0;
Scanner myReader = new Scanner(myObj);
while (myReader.hasNextLine()) {
    String data = myReader.nextLine();
    String delovi[] =data.split(" ");
    if(tip == "Стоматолог") {
        osobe[b]=delovi[0] + " " + delovi[2] + " " + delovi[3] + " " + delovi[4] + " " + delovi[5];
    }
    else {
        osobe[b]=delovi[0] + " " + delovi[2] + " " + delovi[3] + " " + delovi[4];
    }
    b++;
}
myReader.close();

listaKorisnika = new JList<String>(osobe);
```

Слика 3.4 : Део кода за учитавање листе корисника/пацијената

Функције администратора

Брисање се реализује на следећи начин:

Одабрани корисник долази у методу где се учитава текстуална датотека и упоређуј редови датотеке са одабраним корисником. У привремену датотеку се уписују сви корисници а изузев одабраног од стране корисника. На послетку се почетна датотека брише а нова преимењује у назив почетне датотеке. `delovi[0]` и `delovi2[0]` у текстуалној датотеци представљају јединствени идентификатор сваког корисника [4] (Слика 3.5)

```
File inputFile = new File("datoteke\\login\\" + tipKorisnika + ".txt");
File tempFile = new File("datoteke\\login\\temp.txt");
File rename = new File("datoteke\\login\\" + tipKorisnika + ".txt");

BufferedReader reader = new BufferedReader(new FileReader(inputFile));
BufferedWriter writer = new BufferedWriter(new FileWriter(tempFile));

String currentLine;

while((currentLine = reader.readLine()) != null) {
    String trimmedLine = currentLine.trim();
    String delovi[] = trimmedLine.split(" ");
    String delovi2[] = korisnik.split(" ");
    if(delovi[0].equals(delovi2[0])) continue;
    writer.write(currentLine + System.getProperty("line.separator"));
}

writer.close();
reader.close();
inputFile.delete();
tempFile.renameTo(rename);
prozor.dispose();
new brisanje();
```

Слика 3.5 : Функција за брисање
корисника

Унос корисника се реализује на следећи начин:

Учитавамо датотеку и креирамо објекте за читање и писање у датотеку. Најпре проверавамо да ли је корисничко име слободно и уколико јесте, уписујемо нови ред са новим подацима у текстуалну датотеку. (Слика 3.6)


```

try(FileWriter fw = new FileWriter("datoteke\\login\\" + rolaKorisnik + ".txt", true);
    BufferedWriter bw = new BufferedWriter(fw);
    PrintWriter out = new PrintWriter(bw))
{
    int flag =1;
    File myObj = new File("datoteke\\login\\" + rolaKorisnik + ".txt" );
    Scanner myReader = new Scanner(myObj);
    while (myReader.hasNextLine()) {
        String data = myReader.nextLine();
        String delovi[] =data.split(" ");
        if(korisnicko.equals(delovi[0])) {
            flag = 0;
        }
    }
    if(flag == 1) {
        String str = korisnicko + " " + String.valueOf(pass) + " " + imeKorisnik + " " + prezimeKorisnik + " " + rolaKorisnik;
        if(rolaKorisnik == "Стоматолог") {
            str = str + " " + oblast;
        }
        out.println(str);
        System.out.println(str);
    }
    else {
        new greska("Корисничко име је заузето.");
    }
    myReader.close();
}

```

Слика 3.6 : Провера доступности
корисничког имена и упис новог
корисника

Функције техничара

Унос пацијената се реализује на исти начин као и упис нових корисника, код је описан у објашњењу функције уписа нових корисника.

Заказивање прегледа се реализује уписивањем термина и података пацијента у текстуалну датотеку под називом *прегледи.txt*.

Одабир термина је релаизован коришћењем JSpinner објекта [5] и подешавањем одговарајућег формата времена и датума:

```

JSpinner timeSpinner = new JSpinner( new SpinnerDateModel() );
JSpinner.DateEditor timeEditor = new JSpinner.DateEditor(timeSpinner, "dd.MM.yyyy/HH:mm:ss");

```

Слика 3.7 : Кутија за селекцију
времена

Контрола доступности термина се врши по истом принципу као и провера доступности корисничког имена, описана у делу уноса нових корисника.

Преглед заказаних термина подразумева табелу која излистава све термине из текстуалне датотеке *прегледи.txt*.

```

String kolona[]={ "VREME", "IME i PREZIME"};

int i=0;
File myObj = new File("Datoteke\\техничар\\" + "прегледи" + ".txt" );

Scanner myReader1 = new Scanner(myObj);
while (myReader1.hasNextLine()) {
    myReader1.nextLine();
    i++;
}
myReader1.close();

String podaci[][] = new String[i][2];

int b =0;
Scanner myReader = new Scanner(myObj);
while (myReader.hasNextLine()) {
    String data = myReader.nextLine();
    String delovi[] =data.split(" ");

    podaci[b][0]=delovi[3] + " " + delovi[2] + " " + delovi[1] + " " + delovi[5];
    podaci[b][1]=delovi[6] + " " + delovi[7];

    b++;
}
myReader.close();

jt=new JTable(podaci,kolona);
JScrollPane sp=new JScrollPane(jt);

```

Слика 3.8 : Део кода за
излиставање прегледа

Функције стоматолога

У случају стоматолога, исти код за излиставање прегледа се позива као при захтеву техничара, код описан у делу излиставање прегледа – функције техничара. Међутим, део кода који контролише са чије стране долази захтев за излиставање прегледа додаје одређене опције стоматологу – опцију генерисања извештаја.

```

if(e.getSource() == button2) {
    int red=jt.getSelectedRow();
    String ime=String.valueOf(jt.getValueAt(red, 1));
    new generisanjeizvestaja(ime);
}

```

Слика 3.9 : Део ActionListener кода за
генерисање извештаја одабраног
прегледа

Генерисање извештаја

За реализацију овог дела кода је коришћена библиотека iText 7.0.0. Упутство за интегрисање ове библиотеке у пројекат погледати [овде](#).

JAR датотеке можете преузети са овог [линка](#).

Код најпре провери да ли је „картон“ односно фолдер корисника већ постоји и уколико не постоји, креира фолдер са називом пацијента. Након тога креира PDF документ са називом пацијента и временом прегледа, уноси у датотеку сам извештај и чува датотеку у фолдеру пацијента. [6]

```
SimpleDateFormat formatter = new SimpleDateFormat("hh-mm dd-MM-yyyy");
Date date = new Date();
String datum=String.valueOf(formatter.format(date));

File directory = new File("datoteke\\стоматолог\\"+pacijent);
if (! directory.exists()){
    directory.mkdir();
}
String putanja="datoteke\\стоматолог\\"+pacijent+"\\ "+pacijent + " " +datum+".pdf";
PdfWriter pisac = new PdfWriter(putanja);

PdfDocument pdfDokument=new PdfDocument(pisac);

pdfDokument.addNewPage();
Paragraph paragraph = new Paragraph(tekst);

Document document = new Document(pdfDokument);
document.add(paragraph);
```

Слика 3.10 : Део кода задужен за
креирање извештаја

UML Дијаграми

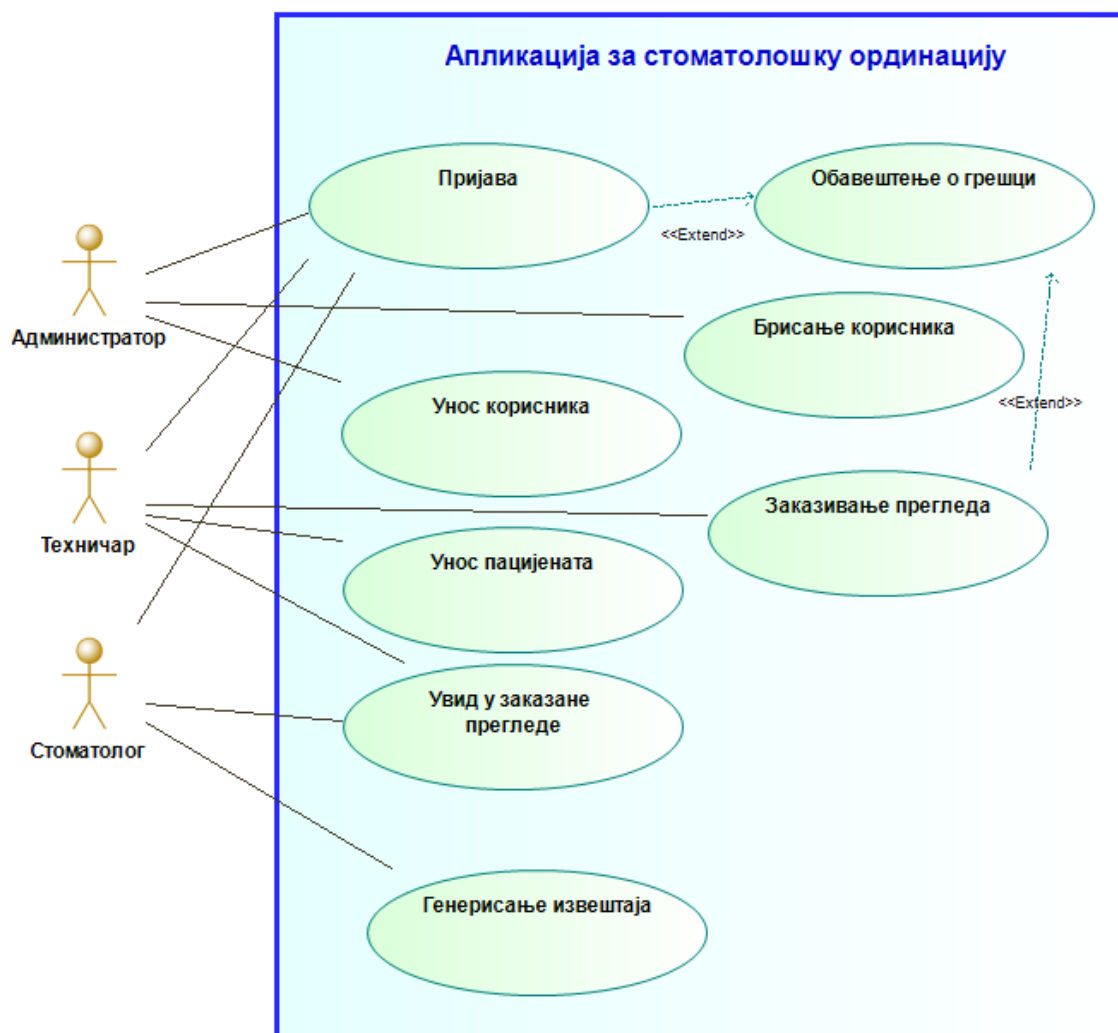
UML (Unified Modeling Language) је обједињени, визуелни језик за пословно и софтверско моделовање у свим фазама развоја и за све типове система, као и за генерално моделовање којим се дефинишу статичке структуре и динамичко понашање. [7]

Use Case дијаграм

Овај тип дијаграма представља најједноставнији приказ интеракције корисника са системом. У овом систему постоје следећи случајеви:

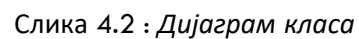
1. UseCase: Пријава
 - a. Кратак опис: Пријава корисника у систем
 - b. Актер: Администратор, техничар, стоматолог
 - c. Предуслови: Апликација је покренута
 - d. Опис: Уношењем својих корисничких података систем верификује корисника и додљује му одобрене могућности
 - e. Последице: Да би користио систем неопходно је да потврди свој идентитет
2. UseCase: Унос корисника
 - a. Кратак опис: Унос нових корисника у систем
 - b. Актер: Администратор
 - c. Предуслови: Администратор се успешно пријавио у систем
 - d. Опис: Администратор додаје нове кориснике у систем као и информације о корисницима
 - e. Последице: Да би техничари и стоматолози могли да користе систем, неопходно је да их администратор најпре региструје у систем
3. UseCase: Брисање корисника
 - a. Кратак опис: Брисање постојећих корисника из система
 - b. Актер: Администратор
 - c. Предуслови: Администратор се успешно пријавио у систем
 - d. Опис: Администратор уређује кориснике и има могућност брисања корисника из система, тиме им онемогућавајући приступ систему
 - e. Последице: Одређени запослени више немају приступ систему
4. UseCase: Унос пацијената
 - a. Кратак опис: Унос нових клијената у систем
 - b. Актер: Техничар
 - c. Предуслови: Техничар се успешно пријавио у систем
 - d. Опис: Техничар додаје нове кориснике услуга стоматолошке ординације у систем.

- е. Последице: Да би пацијенту могао бити заказан преглед, неопходно је да буде унет у систем
- 5. UseCase: Заказивање прегледа
 - а. Кратак опис: Додељивање слободних термина за преглед
 - б. Актер: Техничар
 - с. Предуслови: Техничар се успешно пријавио у систем
 - д. Опис: Техничар заказује прегледе пацијентима
 - е. Последице: Да би пацијенту могао бити извршен преглед и написан извештај, неопходно је да му преглед буде заказан
 - ф. Изузеци: Термин који је техничар одабрао је заузет
- 6. UseCase: Увид у заказане прегледе
 - а. Кратак опис: Увид у заказане прегледе
 - б. Актер: Техничар, Стоматолог
 - с. Предуслови: Техничар/стоматолог се успешно пријавио у систем
 - д. Опис: Техничар има могућност прегледа свих заказаних прегледа док стоматолог добија и опцију генерисања извештаја за одабраног пацијента
 - е. Последице: Да би пацијенту могао бити написа извештај, неопходно је да има заказан преглед
- 7. UseCase: Генерисање извештаја
 - а. Кратак опис: Писање извештаја прегледа
 - б. Актер: Стоматолог
 - с. Предуслови: Стоматолог се успешно пријавио у систем
 - д. Опис: Након обављеног прегледа, стоматолог пише извештај о прегледу
 - е. Последице: Извештај је написан



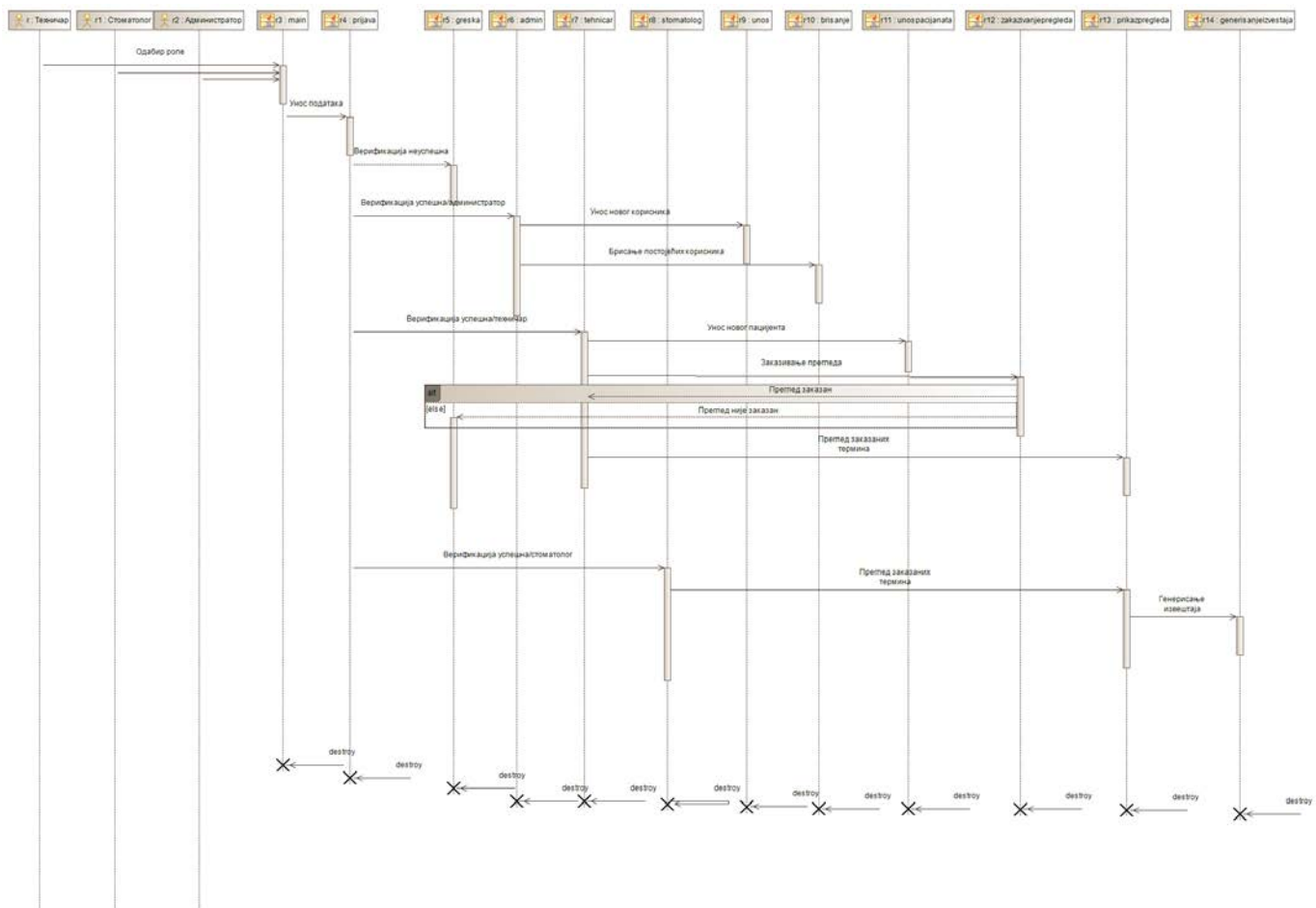
Слика 4.1 : UseCase дијаграм

У софтверском инжењерингу, дијаграм класе у језику обједињеног моделовања (UML) представља врсту дијаграма статичне структуре који описује структуру система приказивањем класа система, њихових атрибута, операција (или метода) и односа између објеката. [8]



Дијаграм редоследа

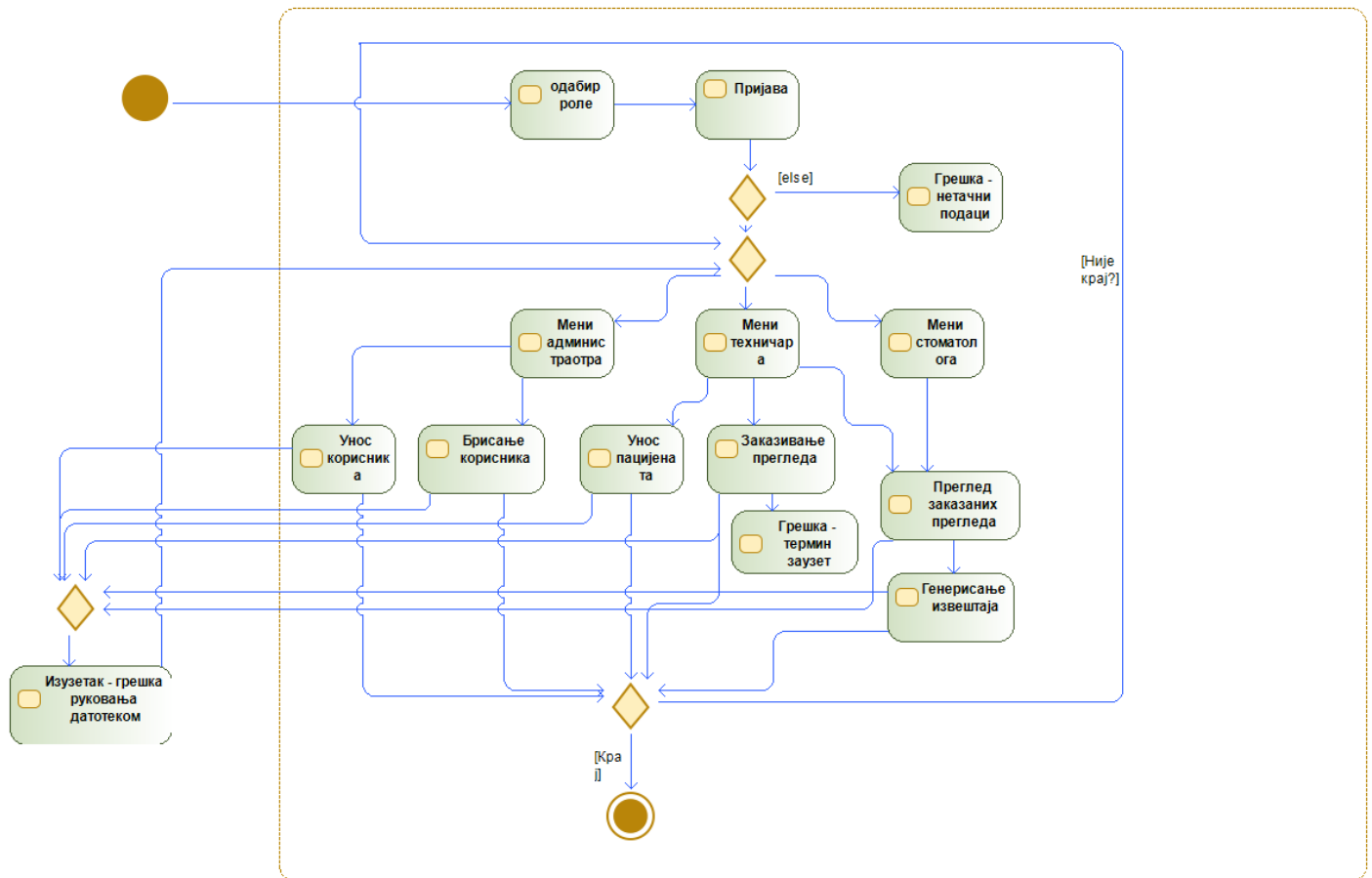
Дијаграм редоследа је тип дијаграма интеракције зато што описује како – и којим редоследом – група објеката функционише заједно. Ове дијаграме користе пројектанти софтвера и пословни професионалци да би разумели захтеве за нови систем или да би документовали постојећи процес. [9]



Слика 4.3 : Дијаграм редоследа

Дијаграм активности

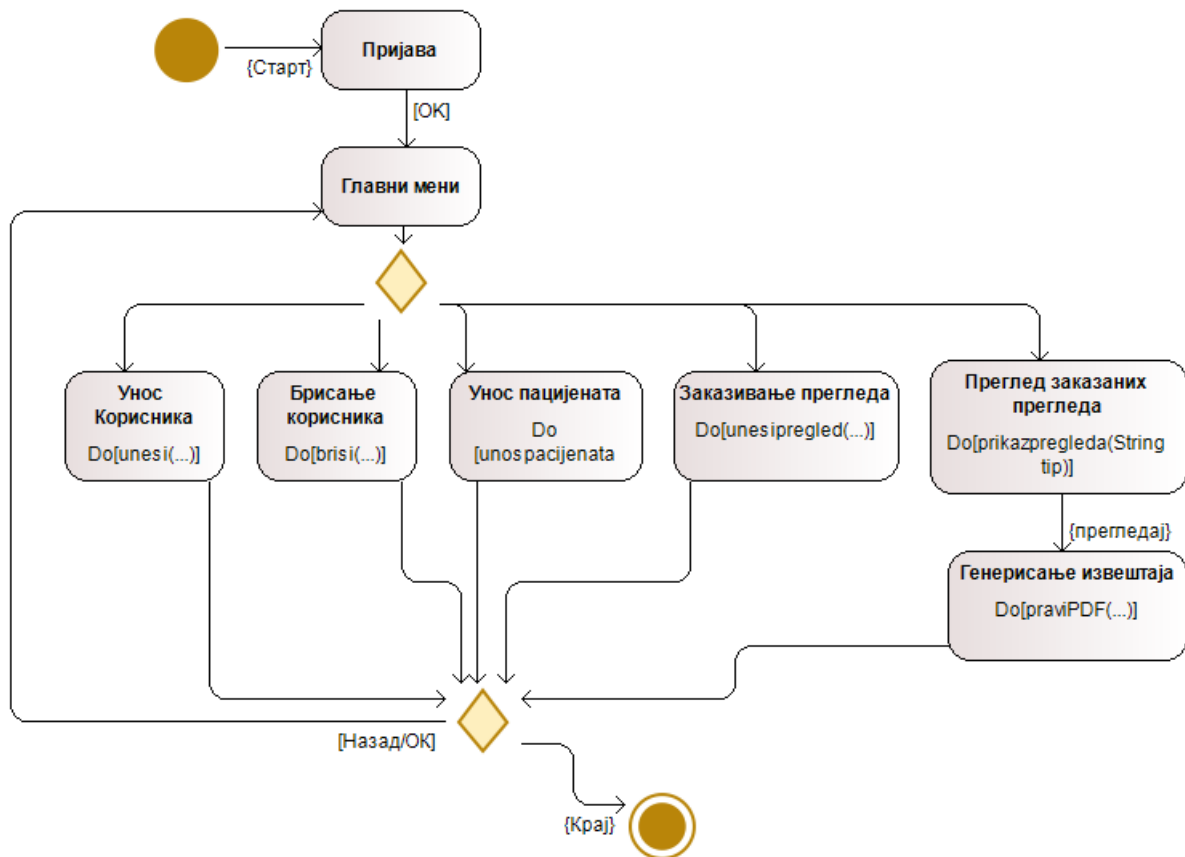
Дијаграм активности визуелно представља низ радњи или проток контроле у систему сличном дијаграму тока или дијаграму тока података. Дијаграми активности се често користе у моделирању пословних процеса. Они такође могу да опишу кораке у дијаграму предмета употребе. Активности по узору на модел могу бити секвенцијалне и упоредне. [10]



Слика 4.4 : Дијаграм активности

Дијаграм стања

Дијаграм стања се обично користи за описивање понашања објекта зависног од стања. Објекат различито реагује на исти догађај у зависности од тога у каквом је стању. Дијаграми стања се обично примењују на објекте, али се могу применити на било који елемент који има понашање према другим ентитетима као што су: актери, коришћење предмета, методе, подсистеми и сл. [11]



Слика 4.5 : Дијаграм стања

Литература

- [1] <https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html> - 25. Maj 2022
- [2] <https://docs.oracle.com/javase/7/docs/api/java/awt/package-summary.html> - 25. Maj 2022
- [3] https://www.w3schools.com/java/java_files_read.asp - 26. Maj 2022
- [4] <https://stackoverflow.com/questions/5360209/how-to-delete-a-specific-string-in-a-text-file>
- 26. Maj 2022
- [5] <https://www.codejava.net/java-se/swing/how-to-use-jdatepicker-to-display-calendar-component>
- 26. Maj 2022
- [6] <https://knpcode.com/java-programs/generating-pdf-java-using-itext-tutorial/> - 26. Maj 2022
- [7] *Unified Modeling Language User Guide, The* (2 ed.). Addison-Wesley. 2005. p. 496. ISBN 0321267974.
- [8] Scott W. Ambler (2009) *UML 2 Class Diagrams*. Webdoc 2003-2009.
- [9] *System Sequence Diagrams*
- [10] *Glossary of Key Terms* at McGraw-hill.com.
- [11] Samek, Miro (2008). *Practical UML Statecharts in C/C++, Second Edition: Event-Driven Programming for Embedded Systems*. Newnes