# Конфликтне ситуације – Студент 1

## 1. Конкурентно креирање прве оцене

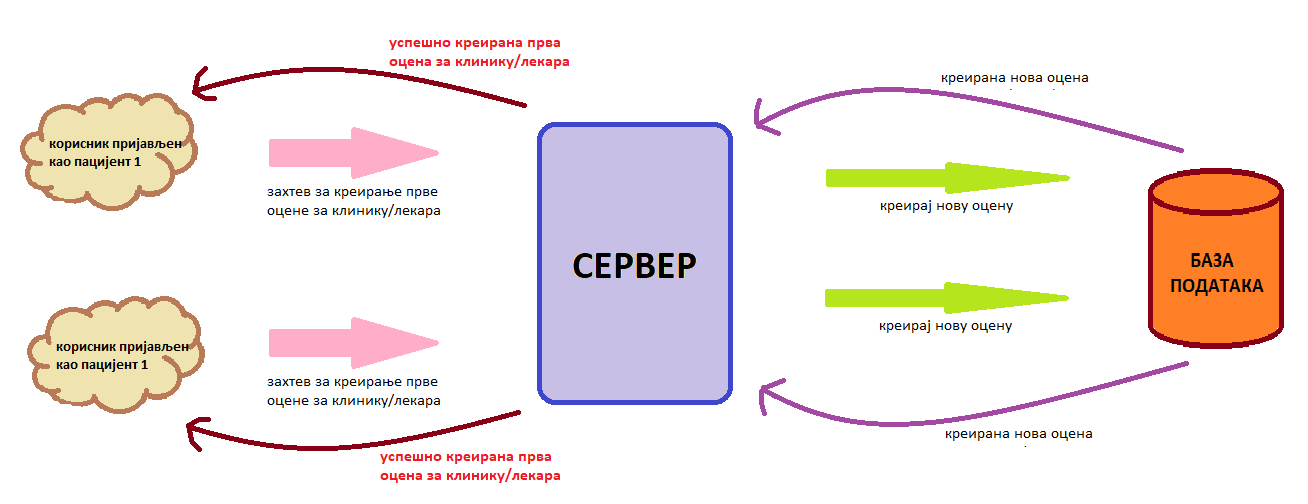
Функционалност *оцењивање клиника и лекара* представља две могуће ситуације:

* креирање нове оцене уколико пацијент претходно ниједном није оценио клинику/лекара
* измена постојеће оцене уколико пацијент јесте претходно оценио клинику/лекара

**Напомена:** У имплементираној апликацији дозвољена је истовремена пријава више корисника под истим корисничким налогом.

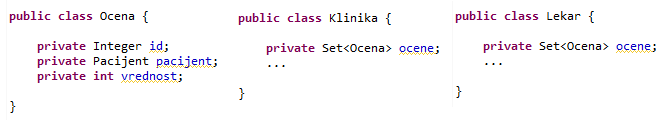
У другом случају, када се мења стара оцена, не постоје конфликтне ситуације. Може се десити да истовремено два корисника пријављена под истим налогом оцене исту клинику/лекара, али то није конфликтна ситуација јер ново стање оцене не зависи од претходног (за разлику од на пример ситуације када се мења стање банковног рачуна тако што се количина новца смањује за одређену суму – тада ново стање рачуна зависи од претходног).

У првом случају може се десити да два корисника пријављена под истим налогом истовремено први пут оцене клинику/лекара. Тада се може десити да се у бази појаве две оцене које се односе на истог пацијента и клинику/лекара. То је недопустиво стање и потребно га је спречити. Цртеж који демонстрира наведени конфликт је следећи:



Слика 1. Конкурентно креирање прве оцене

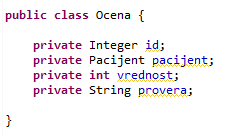
Да би се објаснио концепт решења уоченог конфликта, неопходно је анализирати (првобитни) излгед класа *Ocena, Klinika* и *Lekar*. Изглед је следећи:



Слика 2. Зависност класа Ocena, Klinika и Lekar

Како класа *Ocena* не садржи референцу на клинику/лекара, не врши се пропагирање страног кључа клинике/лекара у табелу која чува оцене, већ се формирају нове табеле *KLINIKA\_OCENA* и *LEKAR\_OCENA*. Обе табеле садрже две колоне – *id* клинике/лекара и *id* оцене.

Проблем уочене конфликтне ситуације решава се увођењем огрганичења јединствености, на нивоу шеме базе података. Ограничење подразумева јединственост парова пацијент-клиника (пацијент-лекар), у смислу везе оцењивања. Како ниједна наведена табела не садржи у себи кључеве оба ентитета (и пацијента и клинике/лекара), класа *Ocena* проширује се на следећи начин:



Слика 3. Проширена класа Ocena

Додаје се поље *provera* које представља *id* клинике/лекара конкатениран са карактером ‘K’/’L’. Када се над табелом *Ocena* уведе ограничење јединствености пара *pacijent-provera*, обезбедиће се на нивоу шеме базе немогућност креирања више од једне оцене која се односи на истог пацијента и клинику/лекара. Разлог зашто се користи претходно наведена конкатенација, а не само *id* клинике/лекара, јесте тај што би се у том случају могло десити да када пацијент оцени клинику која има *id* 1, никад неће моги да оцени лекара који има *id* 1.

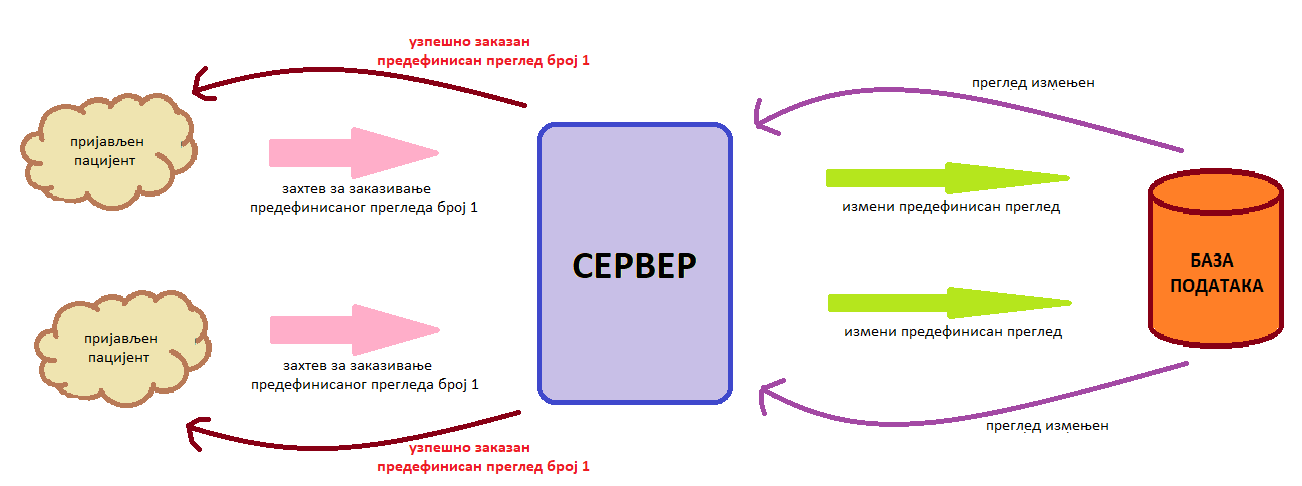
**Напомена:** Класа *Ocena* не садржи референцу на клинику/лекара јер они не наслеђују истог претка, а имплементација истог интерфејса не би решила уочену конфликтну ситуацију јер интерфејси не могу да садрже приватне атрибуте/кључеве, а који се користи у решавању уоченог конфликта. Зато се користи „вештачка“ референца (атрибут *provera*) која представља *id* клинике/лекара (додатно специфициран типом ентитета који се оцењује помоћу наведене конкатенације).

## 2. Конкурентно заказивање предефинисаног прегледа

Функционалност *заказивање предефинисаног прегледа* подразумева измену одабраног прегледа на следећи начин:

* поље *stanje* прегледа поставља се на *ZAUZETO* (могуће вредности су *SLOBODNO*, *ZAUZETO, U\_TOKU* и *OBAVLJENO*)
* поље *karton* прегледа поставља се на картон пријављеног пацијента који је одабрао заказивање прегледа

Конфликтна ситуација која се може десити у овом случају јесте истовремено заказивање истог предефинисаног прегледа од стране више пријављених пацијената. Цртеж који демонстрира наведени конфликт је следећи:



Слика 4. Конкурентно заказивање предефинисаног прегледа

Уочена конфликтна ситуација решава се оптимистичним закључавањем класе *Poseta* (која представља преглед или операцију) тако што се класа прошири пољем *long version* које је анотирано са *@Version*. На тај начин, прва трансакција која промени стање прегледа инкрементираће то поље и тако спречити наредну трансакцију да уради исту измену.

## 3. Конкурентност при заказивању индивидуалног прегледа

Функционалност *заказивање индивидуалног прегледа од стране пацијента* представља креирање захтева за преглед код одабраног лекара, за одабран датум и време, а који ће накнадно бити обрађен од стране администратора клинике. Пацијент претражује и филтрира клинике/лекаре на основу типа прегледа и датума када жели да га закаже, а сервер генерише сатницу за одабраног лекара и враћа је пацијенту. Пацијент из добијене сатнице бира једно време и шаље захтев.

**Напомена:** У даљем тексту, под *индивидуалним прегледом/операцијом* подразумевају се посете за које пацијент/лекар шаље захтев, а администратор га обрађује (резервише салу и остало). Под *предефинисаним прегледима* подразумевају се прегледи које је администратор унапред креирао, а који се могу накнадно заказати од стране пацијента. Предефинисане операције, према спецификацији, не постоје.

Да би се анализирале конфликтне ситуације које се могу десити приликом пацијентовог слања захтева за индивидуалан преглед, неопходно је анализирати на основу чега све сервер генерише сатницу за одабраног лекара и одабран датум. Параметри на основу којих сервер генерише сатницу су следећи:

* радно време лекара
* захтеви за одмор лекара
* претходно креирани прегледи/операције код лекара
* претходно послати захтеви за индивидуални преглед/операцију код лекара

На основу наведеног, може се закључити да постоје четири конфликтне ситуације које се могу десити приликом пацијентовог заказивања индивидуалног прегледа, а то је да се приликом слања захтева за индивидуални преглед истовремено десила нека од следећих промена:

* радно време лекара измењено
* креиран нови захтев за одмор лекара
* креиран нови преглед/операција код лекара (креирањем предефинисаног прегледа или обрадом претходно послатог захтева за индивидуални преглед/операцију)
* креиран нови захтев за индивидуални преглед/операцију код лекара

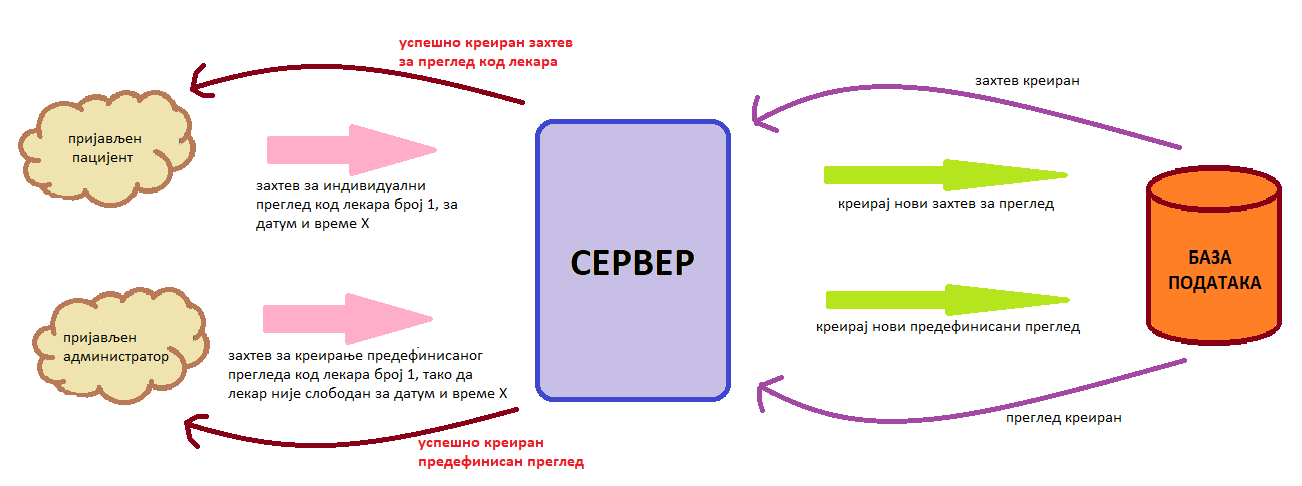
Цртежи који демонстрирају описане конфликтуне ситуације су следећи:



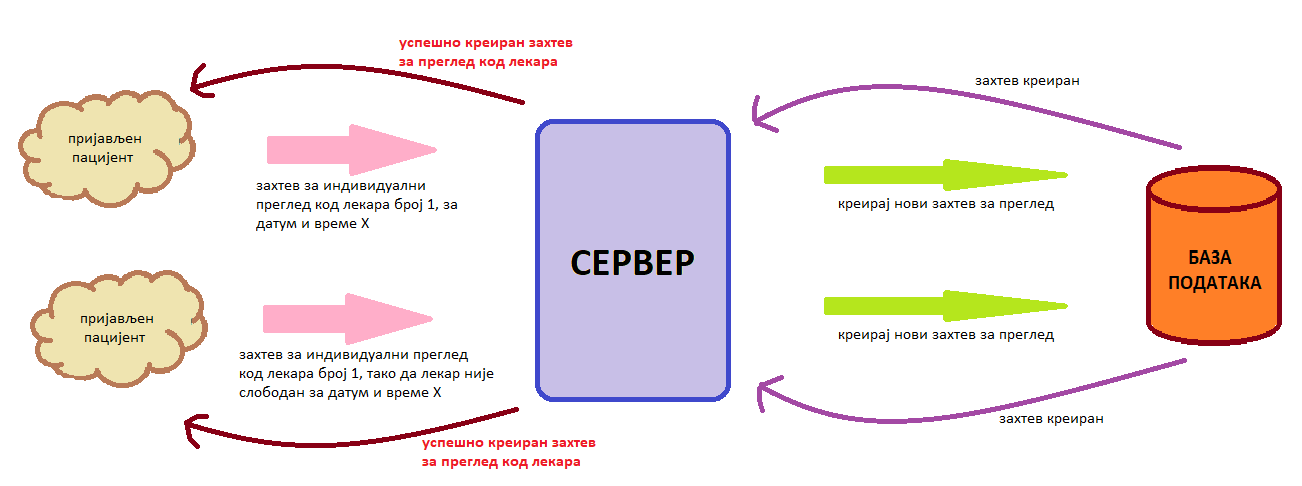
Слика 5. Истовремено заказивање индивидуалног прегледа код лекара и измена радног времена лекара



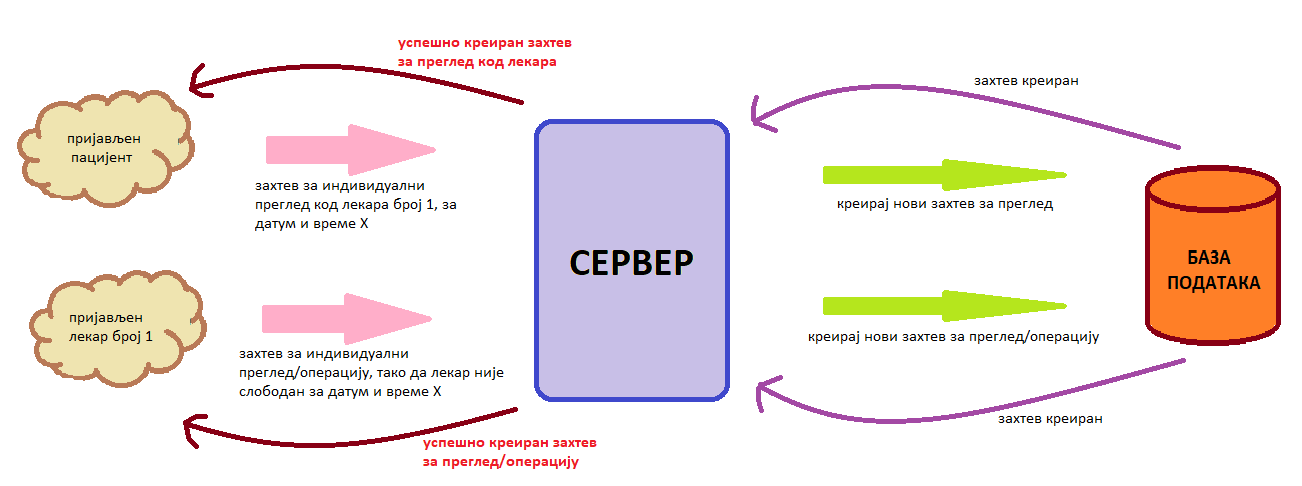
Слика 6. Истовремено заказивање индивидуалног прегледа и креирање захтева за одмор код истог лекара



Слика 7. Истовремено заказивање индивидуалног прегледа и креирање предефинисаног прегледа код истог лекара



Слика 8. Истовремено заказивање два индивидуална прегледа код истог лекара



Слика 9. Истовремено заказивање два индивидуална прегледа или индивидуалног прегледа и операције код истог лекара



Слика 10. Истовремено заказивање индивидуалног прегледа и креирање нове посете код истог лекара

Уочене конфликтне ситуације решавају се на следећи начин:

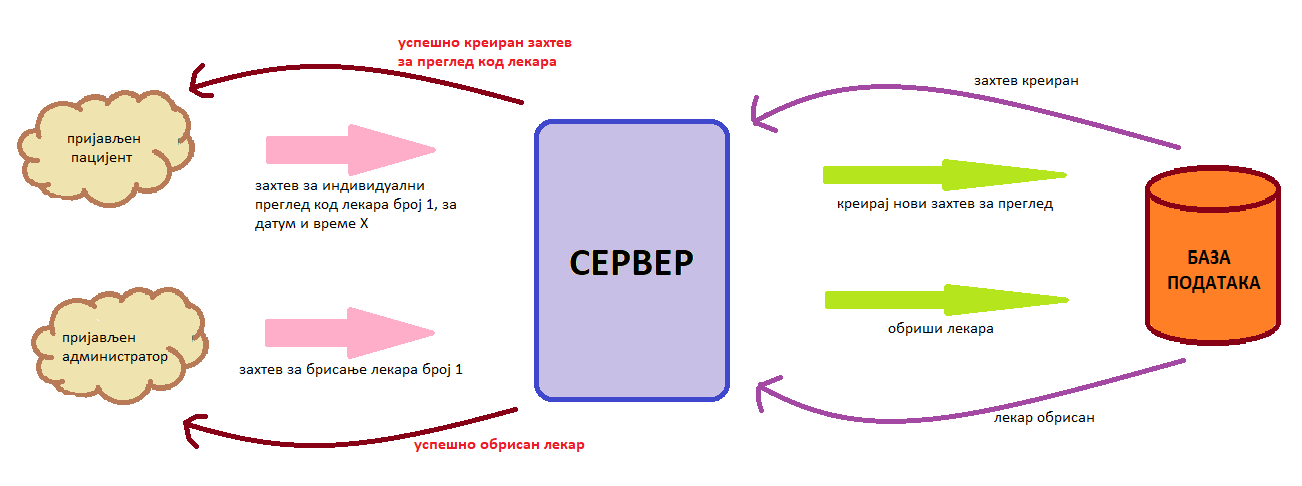
* класа *Korisnik* проширује се пољем *long version* које је анотирано са *@Version* (а како класа *Lekar* наслеђује класу *Korisnik*, она ће исто имати то поље)
* класа *Lekar* проширује се пољем *Date poslednjaIzmena*
* сервисне методе које креирају захтев за одмор лекара, захтев за индивидуални преглед/операцију код лекара или нову посету код лекара (креирањем предефинисаног прегледа или обрадом претходно послатог захтева за индивидуални преглед/операцију) на самом свом крају мењају лекара/лекаре на које се односи захтев/посета, тако што му постављају поље *poslednjaIzmena* на тренутно време

На овај начин обезбеђено је да се сваки пут када се креира нови захтев/посета, обележи у бази да су лекари са којима су они у вези измењени, а онда се може применити класично оптимистино закључавање. Овако се неће креирати захтев за индивидуални преглед затражен од стране пацијента, ако се истовремено десила нека од претходно анализираних ситуација. Тада ће пацијент бити обавештен о грешци, његова страница освежена, а самим тим приказана и најсвежија сатница лекара за одабран датум. Додатно, од претходно наведених сервисних метода, увек ће се извршити само она за коју се утврди да у међувремену нико није мењао лекаре са којима су у вези захтеви/посете које се креирају, тако да ће ресурси базе остати конзистентни.

**Напомена:** Под обрадом претходно послатог захтева за индивидуални преглед/операцију подразумева се и обрада од стране администратора клинике (када администратор резервише салу за посету) и обрада од стране система (када се аутоматски резервише сала).

**Напомена:** Како је уведено оптимистично закључавање класе *Korisnik,* биће спречена истовремена измена личних података једног корисника од стране више клијената пријављених под његовим налогом. Дакле, ако постоје на пример два клијента пријављена под налогом истог пацијента, неће моћи истовремено да измене податке тог пацијента.

Конфликтна ситуација која се може додатно десити, поред претходно анализираних четири, јесте истовремено брисање лекара и заказивање индивидуалног прегледа код њега. Цртеж који демонстрира наведени конфликт је следећи:



Слика 11. Истовремено заказивање индивидуалног прегледа код лекара и брисање лекара

Обе ситуације, и када се прво обрише лекар па пошаље захтев код њега и када се прво пошаље захтев па обрише лекар, биће спречене из следећих разлога:

* уведено је оптимистично закључавање класе *Korisnik*, односно класе *Lekar*
* брисање лекара представља логично брисање (односно измену лекара)
* сервисна метода која креира захтев за инвдивидуални преглед код лекара мења лекара непосредно пре него што се заврши

Дакле, прва трансакција која се изврши ће изменити лекара (инкрементирати његово поље version), а то ће спречити наредну трансакцију да се успешно заврши.

## 4. Успут решени конфликти

Решавајући конфликте који постоје приликом *индивидуалног заказивања прегледа од стране пацијента*, решени су успут и следећи конфликти:

1. истовремено креирање два прегледа/операције код истог лекара – креирањем предефинисаног прегледа или обрадом претходно послатог захтева за преглед/операцију (ова ситуације је конфликтна јер се може дефисити да се у бази појаве два прегледа/операције код истог лекара, а чији се термини преклапају)

2. истовремена измена радног времена лекара и креирање прегледа/операције код њега – креирањем предефинисаног прегледа или обрадом претходно послатог захтева за преглед/операцију (ова ситуације је конфликтна јер се може десити да се у бази појави преглед/операција код лекара који у траженом термину није слободан)

3. истовремено слање захтева за одмор лекара и креирање прегледа/операције код њега – креирањем предефинисаног прегледа или обрадом претходно послатог захтева за преглед/операцију (ова ситуације је конфликтна јер се може десити да се у бази појави преглед/операција код лекара који има креиран захтев за одмор у исто или преклапајуће време)

4. истовремено слање захтева за индивидуални преглед/операцију **од стране лекара**, а не пацијента, и креирање прегледа/операције њега – креирањем предефинисаног прегледа или обрадом претходно послатог захтева за преглед/операцију (ова ситуација је конфликтна јер се може десити да се у бази појави захтев за преглед/операцију и преглед/операција код истог лекара, а чији термини се преклапају)