Универзитет у Београду

Електротехнички факултет

ДИПЛОМСКИ РАД

**СИСТЕМ ЗА ЗАКАЗИВАЊЕ ТЕРМИНА**

Ментор Студент

проф. др Милош Цветановић Лазар Лазић 2016/0245

Београд, 2020.

Садржај

[1 Увод 1](#_Toc51762559)

[2 Преглед постојећих решења 2](#_Toc51762560)

[2.1 Заказивање преко форме 2](#_Toc51762561)

[2.2 Аутоматизовано заказивање 2](#_Toc51762562)

[2.3 Груписање према области пословања 2](#_Toc51762563)

[3 Формирање решења 4](#_Toc51762564)

[3.1 Циљне карактеристике система 4](#_Toc51762565)

[3.2 Почетне претпоставке 4](#_Toc51762566)

[3.3 Предлог решења 4](#_Toc51762567)

[4 Преглед детаља решења 5](#_Toc51762568)

[4.1 Коришћене технологије 5](#_Toc51762569)

[4.2 Корисници система 5](#_Toc51762570)

[4.3 Функционалности 6](#_Toc51762571)

[4.4 База података 9](#_Toc51762572)

[4.5 Хеширање лозинки 11](#_Toc51762573)

[4.6 Отпорност сервера на неисправне захтеве 12](#_Toc51762574)

[4.7 Детаљан преглед заказивања термина 13](#_Toc51762575)

[4.8 Дијаграм стања термина 23](#_Toc51762576)

[5 Корисничко упутство 25](#_Toc51762577)

[5.1 Корисничко упутство за госта 25](#_Toc51762578)

[5.2 Корисничко упутство за фирму 26](#_Toc51762579)

[5.3 Корисничко упутство за запосленог 30](#_Toc51762580)

[5.4 Корисничко упутство за клијента 31](#_Toc51762581)

[5.5 Упутство за покретање система 34](#_Toc51762582)

[6 Закључак 35](#_Toc51762583)

[7 Литература 36](#_Toc51762584)

[8 Спискови 37](#_Toc51762585)

[8.1 Слике 37](#_Toc51762586)

[8.2 Исечци кода 37](#_Toc51762587)

[8.3 Скраћенице 37](#_Toc51762588)

1 Увод

Данас живимо све ужурбанијим животним стилом, где је време кључни ресурс за постизање зацртаних циљева. Често у фризерским салонима, код персоналних тренера, доктора, итд. професионалци морају да се баве организацијом и распоређивањем термина. То им одузима много времена, скреће пажњу са примарног посла и доводи до грешака у раду. Муштерије које заказују термине немају преглед слободних термина, већ добијају информацију телефоном о неком од слободних термина и затим померају остале своје обавезе, потенцијално жртвујући друге планове без разлога.

Управо због горе наведених проблема, постоји потреба за системом за заказивање, који ће растеретити професионалце беспотребних брига и омогућити муштеријама већи избор и бољи преглед доступности услужних делатности. На тај начин, уштеђено је време обема странама и постигнут је ефикаснији начин комуникације и организације.

Постојећи системи су углавном направљени за конкретну област или, још специфичније, за конкретну фирму. Цена развијања и одржавања толико различитих решења је велика. Практичније је обезбедити једно решење, довољно флексибилно да подмири потребе различитих делатности.

Да би људи користили систем, потребно је правити га управо према корисничким потребама. Зато је фокус на интуитивном дизајну са довољно опција за заказивање термина. Систем треба да буде доступан свима, да му се може брзо приступити и због тога је интернет апликација право решење за овај проблем. Како се интернету данас приступа преко мобилних телефона, компјутера, таблета, и других уређаја различитих величина екрана, потребно је обезбедити да апликација буде функционална и доброг изгледа на сваком од њих, односно да буде *responsive*.

У наставку документа прво ће бити приказана постојећа решења слична овом уз коментарисање њихових добрих и лоших страна. Након тога се изводи закључак какве карактеристике треба да има решење и предлаже се на који начин ће то бити постигнуто. Затим следи преглед најбитнијих детаља решења уз пропратне слике и дијаграме за лакше разумевање. Потом је на реду корисничко упутство, и на крају је закључак у коме стоји процена успешности испуњења захтева и генерални преглед решења.

2 Преглед постојећих решења

У предстојећим тачкама биће прокоментарисана постојећа решења. За свако постојоће решење биће објашњено како функционише и које су му добре и лоше стране. Сврха овога јесте да се утврде пожељне и непожељне карактеристике развијаног решења.

2.1 Заказивање преко форме

Постоје разне интернет апликације које омогућавају заказивање термина преко интернета. Најједноставније од њих заказивање врше тако што клијент унесе слободан текст у форму и то се пошаље запосленима. Затим запослени то прочитају, пронађу слободан термин и одговоре клијенту када може доћи. Уколико клијенту одговара то време, он ће прихватити и потврдити свој долазак.

Наизглед једноставна ствар је подељена у три корака и та размена може потрајати дуго у зависности од слободног времена клијента и радног времена запосленог код ког је заказиван термин. Уколико пак клијенту не одговара предложено време, потребна су још два додатна корака где би клијент изразио незадовољство одабраним термином, а запослени затим предложио неки други термин. Та секвенца се може продужити у недоглед.

Добитак коришћењем оваквог система је минималан или никакав. Традиционално заказивање коришћењем телефона би вероватно било брже. Поред тога, уложено је време, труд и новац у развијање решења који се користи само за ту фирму и ни за шта више. Предности таквог система су максимална прилагођеност потребама фирме и једноставност развијања. Код општијих решења се често изгуби та прилагођеност зарад покривања различитих проблема.

2.2 Аутоматизовано заказивање

Мало напреднија верзија система за заказивање је интернет апликација која је аутоматизована, тако да уклони непотребну комуникацију између клијента и запосленог. Заказивање започиње уносом слободних термина запосленог од стране запосленог или фирме, а клијент одабира један од тих фиксних термина. Након успешног заказивања, систем евидентира промену и уклања заказани термин из слободних термина. На тај начин нема беспотребне комуникације и синхронизације. Запослени када гледа своје обавезе, само му се излистају заказани термини и зна када шта треба да ради.

Ово решење је знатно боље јер заправо обезбеђује уштеду у времену. Добро је за фирме које желе да развијају своје решење и имају ресурса за то. Међутим, уколико један запослени пружа више различитих услуга, које не морају нужно да трају исто време, систем није одговарајући. Наиме, фиксни термини не могу истовремено бити две различите дужине и систем није способан да се прилагоди захтевима.

2.3 Груписање према области пословања

Затим постоје системи који пружају резервисање у различитим фирмама исте области пословања. Оваква решења су углавном боље развијена, јер представљају посао за себе, независно од неке одређене фирме. Фирмама је мања цена коришћења јер много фирми нуди заказивање термина, па је исплативост добра.

Проблем ових решења је често то што је изглед страница различитих фирми униформан, па нема простора да се страница истакне изгледом. Такође, ограничени су на одређену област пословања.

3 Формирање решења

У наредним тачкама биће установљене циљне карактеристике система и почетне претпоставке, а на основу та два ће бити дат предлог решења. Циљне карактеристике представљају нешто чему треба тежити како би систем био што бољи, а почетне претпоставке су разумне претпоставке које су неопходне да би систем функционисао.

3.1 Циљне карактеристике система

Посматрајући системе из претходног поглавља, намеће се мисао: направити систем који обједињује предности и изоставља мане тих система. Као кључне карактеристике се издвајају:

* аутоматизација – није смислено нити исплативо правити систем који није аутоматизован
* прилагодљивост – систем треба да буде прилагодљив и предвиди сценарије у којима један запослени врши више различитих услуга и може вршити било коју од њих у било ком тренутку радног времена, те дозволи да закажу било коју од тих услуга у истом тренутку
* општост – идеја је привући фирме из различитих области пословања, али тако да сви имају довољно могућности да се прикажу на добар начин, са одговарајућим опцијама за заказивање термина

3.2 Почетне претпоставке

Претпоставља се да постоје фирме које имају запослене, да фирме одређују услуге које пружају, радна времена својих запослених, као и услуге које пружа свако од њених запослених. Такође, фирма контролише који људи раде за њу, када се запошљавају и отпуштају. Налогом фирме управља особа, за то надређена у фирми. Да би систем могао да функционише, сви корисници морају имати приступ интернету, што је разумно очекивати у данашње време.

3.3 Предлог решења

Увести концепт услуга. Једна фирма пружа једну или више услуга, а сваки њен запослени пружа једну или више фирминих услуга. Услуга може представљати персонални тренинг, шишање, прање косе, офталмолошки преглед, технички преглед за аутомобил итд. У суштини услуга може бити било шта и зато лепо обезбеђује карактеристику општости.

Приликом заказивања термина, клијент прво бира услугу коју жели, а затим има преглед радног времена сваког од запослених који пружају ту услугу. Тачније, клијент види радно време запосленог издељено на термине на основу трајања одабране услуге. На тај начин, потребно је унети само радно време запосленог, нема фиксних термина и постигнута је флексибилност у одабиру термина. Овај механизам обезбеђује прилагодљивост система трајању услуга.

4 Преглед детаља решења

У оквиру овог поглавља биће објашњено како и за кога је систем направљен. То подразумева опис делова система, технологија коришћених за сваки од њих и најбитније технике при развоју решења. Поред тога, наведено је за кога је систем намењен као и који корисник шта може да ради у систему.

4.1 Коришћене технологије

Систем представља интернет апликацију која се састоји из више делова који међусобно комуницирају. *Frontend* део служи за интеракцију корисника са остатком система. *Backend* део служи за обраду корисничких захтева, извршавање логике апликације и комуникацију са *frontend*-ом и базом података. База података је део који служи за складиштење података и извршавање упита над њима.

*Frontend* је урађен у *Angular framework-*у. То је *framework* којим се праве *single-page* апликације, где није потребно при сваком захтеву послатом серверу да се поново учитава цела страна, већ се по потреби убацују и избацују компоненте апликације, у складу са тим шта је потребно. Управо та карактеристика пружа брже извршавање апликације.

*Backend* је урађен коришћењем *Express* сервера који се извршава у извршном окружењу *Node.js*. Те технологије су одабране јер користе *JavaScript* програмски језик, који је згодан за коришћење у комбинацији са *Angular*-ом и одабраном базом података (видети следећи пасус).

За базу података одабран је *MongoDB*. То је нерелациона база података која подржава неструктуриране типове података, који су погодни за решавање датог проблема. Сви подаци се складиште у *JSON* формату, па је због тога погодна за коришћење заједно са *backend* делом.

Коришћење ових технологија у комбинацији назива се *MEAN stack (MongoDB, Express, Angular, Node.js)*. Представља добру комбинацију приликом израда интернет апликација и зато се често користи у индустрији. Дато решење се могло реализовати и у другим технологијама на *frontend*-у и *backend-*у, као и коришћењем релационих и других типова нерелационих база података.

4.2 Корисници система

Да би користили релевантне делове система, корисници морају имати налог преко ког приступају систему. У зависности од типа, корисник може добити налог тако што се сам региструје или тако што га неко други региструје. Типови корисника овог система су:

* Фирма – корисник који представља једну фирму. Његова улога јесте да представи фирму у којој ради на најбољи могући начин, да управља услугама које пружа фирма и запосленима у фирми.
* Запослени – корисник који представља запосленог неке фирме. Он нема много могућности у систему и његова задужења прописује фирма.
* Клијент – корисник који представља било кога ко жели да закаже термин. Он представља центар апликације, и оно зашто апликација постоји.
* Гост – корисник који није пријављен на систем. Он нема приступ релевантним деловима система.

4.3 Функционалности

Следи преглед функционалности система. Оне су подељене према типу корисника у систему, односно према томе који тип корисника шта може да ради. За сваку функционалност ће бити објашњено чему служи и зашто постоји.

4.3.1 Функционалности госта

* Пријављивање на систем – корисник претходно мора имати налог у систему. Након успешног пријављивања, корисник више није гост, већ постаје клијент, фирма или запослени у зависности од типа налога. Тада може видети опције које има, као и своје име и презиме, односно назив фирме у горњем десном углу.
* Регистрација – корисник који жели да користи систем, мора направити налог који ће му пружити те могућности. Постоје две врсте регистрације које гост може обавити:
  + Регистрација клијента – представља регистрацију којом корисник добија налог преко кога приступа функционалностима клијента.
  + Регистрација фирме - представља регистрацију којом корисник добија налог преко кога приступа функционалностима фирме.

4.3.2 Функционалности фирме

* Креирање услуге – фирма може да креира услугу коју пружа, тако да она постоји у систему. Ту услугу касније може доделити неком од запослених. Она ће бити видљива клијентима приликом прегледа фирме.
* Преглед услуга – фирма може да види све услуге које пружа. Том приликом се улсуге приказују табеларно, где за сваку услугу пише њен назив, дужина трајања термина и цена по термину.
* Креирање запосленог – фирма може да дода запосленог из реалног живота у систем. Приликом додавања запосленог, фирма креира нови налог свом запосленом, преко кога он може приступати систему. Такође, том приликом фирма може да специфира које услуге пружа запослени, а то може урадити и касније. Након успешног додавања, запосленом се шаље мејл са креденцијалима помоћу којих се пријављује на систем.
* Уклањање запосленог – фирма може да уклони свог запосленог из система. Разлог за тако нешто може бити отпуштање. Фирма не може отпустити запослене који имају заказане термине у будућности.
* Преглед запослених – фирма може да излиста све своје запослене и одатле даље управља њиховим налозима.
* Додељивање радног времена запосленом – фирма може да додели радно време својим запосленима. То подразумева да одабере једног запосленог и дефинише његово радно време у неком предстојећем периоду. Том приликом специфицира који је то период (датум почетка и краја), којим данима запослени ради и које му је радно време.
* Преглед радног времена запосленог – фирма може да види радно време сваког свог запосленог, за одређени дан. Том приликом одабира датум од интереса и приказују сви термини (заузети и слободни) тог запосленог за тај дан.
* Додела услуга запосленом – фирма може сваком свом запосленом да промени услуге које пружа. Том приликом бира услуге које је запослени способан да пружи. Може додати нове и/или уклонити постојеће услуге. Све услуге које пружају запослени мора првенствено да пружа фирма.
* Одјављивање са система – фирма, као и сваки други корисник, на крају рада може се одјавити са система.

4.3.3 Функционалности запосленог

* Преглед термина – запослени може да прегледа своје радно време. У оквиру прегледа излистани су термини са временом почетка и временом завршетка. Уколико је термин заказан, пише ко га је заказао као и коју услугу је заказао. Уколико термин није заказан само пише да је слободан.
* Промена лозинке – запослени може да промени лозинку за приступ свом налогу. Ова функционалност је корисна зато што се приликом креирања налога генерише насумична лозинка која се шаље запосленом на мејл. Може се десити да запослени не жели да користи ту лозинку, па је може променити.
* Одјављивање са система – запослени, као и сваки други корисник, на крају рада може се одјавити са система.

4.3.4 Функционалности клијента

* Претрага фирми – клијент може да претражује све фирме које су регистроване у систему. Може их претраживати према називу или према услугама које пружају. Након избачених резултата претраге, клијент може прегледати фирме.
* Преглед фирме – клијент може да прегледа одабрану фирму. Ту ће видети како фирма се зове, где се налази, опис фирме, као и услуге које фирма пружа. Одатле може даље заказати термин за услугу коју хоће.
* Заказивање термина – клијент може да закаже термин. Приликом заказивања термина, клијент одабира услугу коју заказује, датум и време када ће се термин одржати и запосленог који врши услугу. Уколико клијент закаже термин тог истог дана, мејл са обавештењем биће послат одговарајућем запосленом како би знао да му је дневни распоред измењен.
* Преглед заказаних термина – клијент може да прегледа термине које је заказао а који се још нису одржали. Приликом прегледа, излистани су термини, укључујући датум одржавања, време почетка, назив услуге и опцију за детаљан преглед термина (следећа функционалност).
* Преглед заказаног термина – клијент може да прегледа свој заказани термин. Том приликом пише назив фирме која обавља услугу, адреса фирме, дан и датум одржавања термина, време почетка термина, време завршетка термина и име запосленог који пружа услугу. Поред тога, постоји опција за отказивање термина.
* Отказивање термина – клијент може да откаже термине који се још нису одржали. Уколико то учини истог дана када термин треба да буде одржан, биће послат мејл са обавештењем запосленом како би знао да му је дневни распоред измењен.

4.3.5 Дијаграм случајева употребе

Дијаграм случајева употребе се може видети на [слици 1](Систем%20за%20заказивање%20термина.docx#слика_1). Он лепо приказује случајеве употребе разврстане према типу корисника. Такође приказује значајне детаље попут информација о томе када се шаље мејл запосленом.



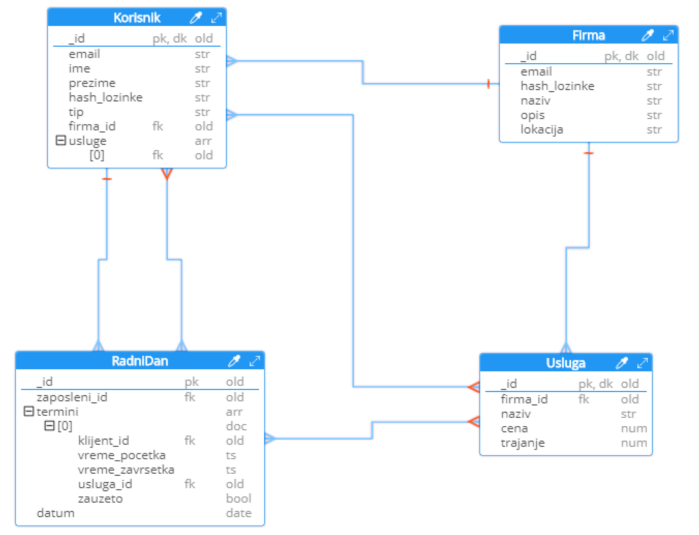
Слика 1. Дијаграм случајева употребе

4.4 База података

Као што је речено у [тачки 4.2](Систем%20за%20заказивање%20термина.docx#tacka_4_2), за базу података је одабран *MongoDB*. Пошто је овај систем намењен за широку употребу и може имати много корисника, па самим тим и много података, *MongoDB* је одличан избор јер има добру подршку за хоризонталну скалабилност. Метода која то пружа зове се *Sharding*. *Sharding* није коришћен у овом решењу јер је систем развијан локално, међутим уколико за то буде потребе, увек се може додати.

4.4.1 Модел базе података

Модел базе података се може видети на [слици 2](Систем%20за%20заказивање%20термина.docx#слика_2). На њему се виде колекције у оквиру базе података, као и информације које ће бити чуване у њиховим документима. Дати модел направљен је помоћу алата *Hackolade*.



Слика 2. Модел базе података

Иако дате релације нису праве релације, тј. не ограничавају унос нити измену поља која су у релацијама, додате су јер јасније приказују у којим су односима документи из одговарајућих колекција. Такође, поља и типови поља у документима не спречавају унос других поља или вредности других типова, међутим потпомажу разумевању модела. Детаљније о свакој од колекција се може видети у наредним тачкама.

4.4.2 Колекција Korisnik

У питању је колекција која чува документе са информацијама о клијентима и запосленима. Разлог за обједињење ова два типа корисника у једну колекцију је то што имају јако слична поља. Поље \_id служи за идентификацију документа и њега генерише база података аутоматски. Поља ime и prezime представљају одговарајуће податке о кориснику. Поља email и hash\_lozinke садрже адресу електронске поште и хеш лозинке, респективно, и користе се за аутентикацију. Више о томе зашто се чува хеш, а не лозинка се може видети у [тачки 4.6](Систем%20за%20заказивање%20термина.docx#тачка_4_6). Поље tip садржи информацију о томе да ли је у питању документ са информацијама о клијенту или запосленом.

Сва досад наведена поља садрже и документи клијената и документи запослених. Поља firma\_id и usluge садрже само документи запослених. Разлог за то је тај што је потребно имати информацију о томе за кога који запослени ради, као и које услуге сваки од запослених пружа. Поље firma\_id референцира поље \_id из колекције Firma и служи за идентификацију фирме у којој ради запослени. Поље usluge представља низ референци на поља \_id из колекције Usluga и служи за идентификацију услуга које пружа запослени.

4.4.3 Колекција Firma

У питању је колекција која чува документе са информацијама о фирмама. Разлог зашто није обједињена са колекцијом Korisnik је то што немају довољно сличних поља. Поље \_id служи за идентификацију документа и њега генерише база података аутоматски. Поља email и hash\_lozinke садрже адресу електронске поште и хеш лозинке, респективно, и користе се за аутентикацију. Више о томе зашто се чува хеш, а не лозинка се може видети у [тачки 4.6](file:///C:\Users\Lazar\Desktop\diplomski\Систем%20за%20заказивање%20термина.docx#тачка_4_6). Поља naziv, opis и lokacija садрже одговарајуће информације о фирми.

4.4.4 Колекција Usluga

У питању је колекција која чува документе са информацијама о услугама. Како услуге додаје и пружа одређена фирма, поставља се питање да ли су услуге могле бити уграђене у документ фирме па самим тим и да ли је ова колекција потребна. Пошто се услугама често приступа независно од фирме, односно фирми независно од услуга, и пошто колекција Korisnik реферише услуге, боље је решење одвојити их у засебну колекцију. На тај начин је постигнуто убрзање приликом извршавања упита и потенцијално је уштеђен простор (који би се користио за чување редудантних информација у колекцији Korisnik).

Поље \_id служи за идентификацију документа и њега генерише база података аутоматски. Поље firma\_id референцира поље \_id из колекције Firma и служи за идентификацију фирме која пружа дату услугу. Поља naziv, cena и trajanje садрже одговарајуће информације о услузи.

4.4.5 Колекција RadniDan

У питању је колекција која чува документе са информацијама о радним данима запослених. Један документ чува све термине у једном радном дану једног запосленог неког датума. Поље \_id служи за идентификацију документа и њега генерише база података аутоматски. Поље zaposleni\_id референцира поље \_id из колекције Korisnik и служи за идентификацију запосленог чији је то радни дан. Поље datum садржи дан за који важе термини из поља termini.

Поље termini представља низ докумената где сваки документ представља један термин и садржи поља:

* klijent\_id – референца на поље \_id из колекције Korisnik и служи за идентификацију клијента који је заказао термин
* usluga\_id – референца на поље \_id из колекције Usluga и служи за идентификацију услуге која се пружа у датом термину
* vreme\_pocetka – садржи време почетка термина
* vreme\_zavrsetka – садржи време завршетка термина
* zauzeto – садржи информацију о томе да ли је термин заузет или слободан

Уколико термин није заузет поља klijent\_id и usluga\_id не постоје, а поље zauzeto je false. Уколико је термин заузет klijent\_id и usluga\_id су попуњени, а поље zauzeto је true. У суштини, поље zauzeto је редудантно, јер се може извести из тога да ли постоје поља klijent\_id и usluga\_id, међутим заузима само један бит и доприноси читљивијем коду па је задржано.

Приликом пројектовања базе података постављало се питање да ли је боље имати колекцију са документима који чувају само један термин уместо термине за цео дан. Имајући у виду да се приликом прављења, прегледа (од стране запосленог) и заказивања термина сваки пут приступа терминима по датуму, добија се на убрзању приликом извршавања датих упита. Такође се штеди на простору јер би у супротном сваки термин морамо да има поља datum и zaposleni\_id, уместо по једном за цео дан. Негативна страна овога јесте што су упити знатно спорији када се претражују заказани термини за одређеног клијента. Тада је потребно претражити све термине свих радних дана свих запослених и упарити их са пољем \_id одговарајућег клијента. Међутим, како су ти упити много ређи и постоје само за један случај коришћења, повољно је направити овакву колекцију.

4.5 Хеширање лозинки

На [слици 2](Систем%20за%20заказивање%20термина.docx#слика_2) се може видети да у колекцијама Korisnik и Firma постоји поље hash\_lozinke. У питању је поље у ком се чува хеш лозинке одговарајућег корисника. За хеширање се користи хеш функција *sha256* која производи хеш дужине 256 бита, односно 32 бајта. Поставља се питање: зашто се не би чувала лозинка у изворном облику?

Данас су напади на интернет апликације изузетно чести. Неретко се дешавају напади на социјалне мреже као што су *Facebook*, *Instagram* и сл. које садрже много информација о својим корисницима. Том приликом процуре многе информације, па тако и лозинке корисника. Како су људи склони томе да имају исту лозинку на више различитих налога, компромитовани су и ти други налози.

Управо из тих разлога се не памти лозинка у изворном облику. Згодна особина хеш функција је то што су *one-way*, односно лако се израчунава хеш лозинке, али се јако тешко може открити лозинка на основу хеша. На тај начин, чак ни особе које имају увид у базу података не могу открити корисничку лозинку, па је постигнута сигурност и тајност корисника. Такође, време за проверу лозинке, тј. време израчунавања хеша и поређења са пољем из документа је релативно кратко и не оптерећује сервер.

Дакле, последице хеширања лозинке су добијање на сигурности и тајности корисника науштрб мало процесорске снаге. Пошто је у питању ефикасна метода чувања лозинки, данас се често примењује у најразличитијим системима.

4.6 Отпорност сервера на неисправне захтеве

Систем је направљен тако да му корисници приступају преко *Frontend* дела јер он пружа интуитиван дизајн и лак је за употребу. Међутим, то не спречава никога да шаље захтеве директно серверу. Зато треба предвидети разне сценарије у којима потенцијални нападач жели да угрози систем и спречити их. На следећем исечку кода из изворног кода сервера, види се колико је потребно провера како бисмо били сигурни да је захтев за креирање нове услуге исправан.

router.post('/usluga', async function(req, res, next) {

    const result = validacija.validirajUslugu(req.body);

    if (result.error) console.log(result.error);

    if (result.error) return res.status(400).json({ poruka : 'Neispravan zahtev.' });

    const firma = await FirmaModel.find({ \_id : req.body.firma\_id }).exec();

    if (firma.length === 0) return res.status(400).json({ poruka : 'Neispravan zahtev: Firma ne postoji. '});

    const u = await UslugaModel.find({ naziv : req.body.naziv, firma\_id : req.body.firma\_id }).exec();

    if (u.length !== 0) return res.json({ poruka: 'Već imate uslugu sa tim nazivom.' });

    let usluga = new UslugaModel({

        naziv : req.body.naziv,

        trajanje : req.body.trajanje,

        cena : req.body.cena,

        firma\_id : req.body.firma\_id

    });

    usluga.save();

    res.json(usluga);

});

*Исечак кода 1. Обрада захтева за креирање нове услуге на серверу*

Прво се валидира сам захтев, тј. да ли има информације одговарајућег назива и типа. То се ради помоћу пакета *Joi* који пружа интуитиван интерфејс за валидацију. Овај корак се налази на првој линији кода у оквиру функције зато што у случају да захтев нема потребне информације, нема разлога да се провера садржаних информација настави. На следећем исечку се може видети како је то урађено.

exports.validirajUslugu = function (usluga) {

    const schema = Joi.object({

        naziv: Joi.string()

            .min(1)

            .required(),

        trajanje: Joi.number()

            .integer()

            .positive()

            .required(),

        cena: Joi.number()

            .integer()

            .positive()

            .required(),

        firma\_id: Joi.objectId()

            .required()

    });

    return schema.validate(usluga);

}

*Исечак кода 2. Валидација документа услуге*

Дакле, као што се види, за креирање услуге су потребна четири поља: naziv, trajanje, cena и firma\_id. Свако од њих мора да испуњава задате критеријуме, иначе ће бити пријављена грешка и враћен одговор са статусом *Bad Request*. Уколико су подаци успешно валидирани, наставља се провера.

Након тога се проверава да ли постоји фирма у систему која има поље \_id чија је вредност иста као прослеђени параметар firma\_id. Уколико нема, биће враћен одговор са описом проблема, у супротном, значи да фирма постоји, па се може наставити са провером. Следећи корак је провера да ли та фирма већ има услугу која се тако зове. Ако има, захтев се не може прихватити јер касније неће бити јасно која је разлика између те две услуге и систем неће исправно радити. Коначно, ако је и тај услов задовољен, тј. нема услуге са истим именом, захтев се прихвата и чува у бази података. Враћа се одговор са новокреираном услугом, што значи да је успешно креирана.

За наизглед једноставан захтев, као што је овај, потребно је много провера, а за комплексније захтеве далеко више. Приликом развијања система треба посветити пажњу исправности захтева, јер и најмањи пропуст може довести до неконзистенције у систему и изазвати велике проблеме.

4.7 Детаљан преглед заказивања термина

Заказивање термина је најбитнији и најсложенији случај употребе у систему. Из тог разлога му је посвећено поглавље у овом документу. Пред овај случај употребе улоговани клијент претражује фирме од интереса и кликне на дугме за преглед неке од фирми. Тада ће му, између осталог, бити приказан списак свих услуга које пружа фирма. Кликом на дугме за заказивање услуге почиње овај случај употребе.

4.7.1 Сценарио случаја употребе „заказивање термина“

* **Циљ**:

Заказивање термина жељене услуге од стране клијента.

* **Предуслов**:

Клијент је пријављен на систем. Постоје фирме у систему које пружају услуге.

* **Последица** **успеха**:

Клијент је заказао термин.

* **Последица** **неуспеха**:

Клијент није заказао термин.

* **Примарни** **актор**:

Клијент

* **Секундарни** **актор**:

-

* **Улаз**:

Подаци:

1. Фирма која пружа услугу за коју се термин заказује
2. Услуга чији се термин заказује
3. Запослени који пружа услугу за коју се термин заказује
4. Време термина у ком се услуга врши

* **Излаз**:

Успешност заказивања термина

* **Главни** **сценарио** **успеха**:

1. Клијент одабира фирму у којој жели да добије услугу
2. Систем приказује детаље фирме и услуге које пружа
3. Клијент одабира услугу за коју жели да закаже термин кликом на дугме
4. Систем приказује форму за одабир жељеног датума
5. Клијент одабира жељени датум и потврђује избор кликом на дугме
6. Систем приказује форму за одабир запосленог који ће вршити услугу и термина када ће се услуга вршити
7. Клијент одабира запосленог и термин и потврђује избор кликом на дугме
8. Систем приказује све одабране информације као преглед одабраног термина
9. Клијент потврђује заказани термин
10. Систем проверава исправност захтева и доступност термина
11. Ажурира се стање система
12. Систем потврђује успешност заказивања термина

* **Проширења**:

Успеха:

7.а Клијент одлучи да ипак жели да закаже термин неког другог датума и кликне на дугме које води на претходни корак

7.а.1 Систем приказује корак 5

9.а Клијент одлучи да ипак жели да измени термин и кликне на дугме које води на претходни корак

9.а.1 Систем приказује корак 7

Неуспеха:

3.а Фирма није још додала услуге које пружа, па клијент не може заказати термин

5.а Клијент није одабрао датум и кликне на дугме за прелазак на следећи корак

5.а.1 Систем исписује поруку да клијент треба да одабере датум и остаје у кораку 5

5.б Клијент уноси неисправан датум и кликне на дугме за прелазак на следећи корак

5.б.1 Систем исписује поруку да клијент треба да унесе исправан датум и остаје у кораку 5

6.а Одабраног датума нема слободних термина

6.а.1 Систем обавештава клијента и препоручује му да проба неки други датум, остаје у кораку 6 без могућности преласка на корак 7

7.а Клијент није одабрао запосленог и кликне на дугме за прелазак на следећи корак

7.а.1 Систем исписује поруку да клијент треба да одабере запосленог и остаје у кораку 7

7.б Клијент је одабрао запосленог али није одабрао време термина и кликне дугме за прелазак на следећи корак

7.б.1 Систем исписује поруку да клијент треба да одабере време термина и остаје у кораку 7

10.а Захтев из неког разлога није исправан

10.а.1 Систем исписује поруку и остаје у кораку 10

10.б Неки други клијент је заказао тај термин или неки део тог термина у међувремену, па он више није доступан

10.б.1 Систем исписује поруку и враћа клијента на поновни одабир термина

4.7.2 Образложење сценарија случаја употребе „заказивање термина“

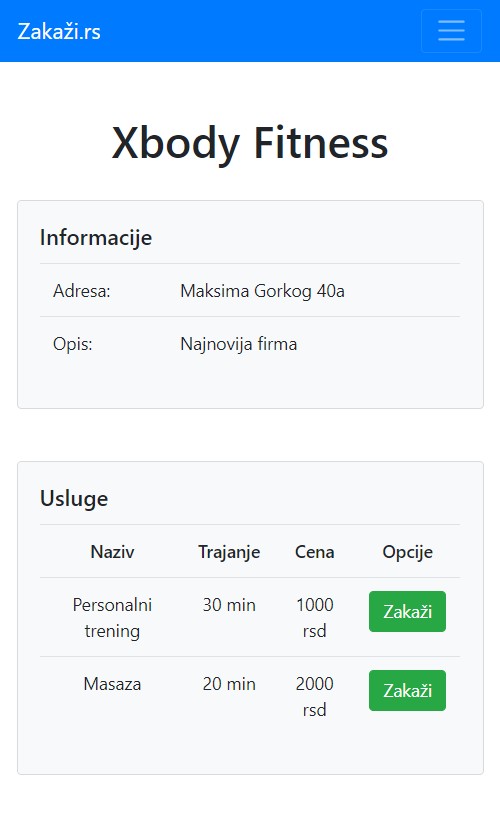
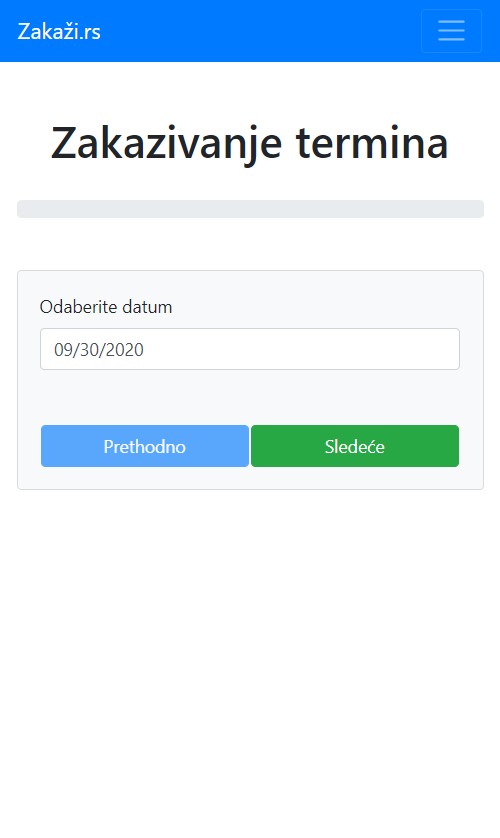
Овај сценарио је формиран пре свега имајући у виду потребе клијената. Како се систем може користити на уређајима који имају екране различитих величина, а на сваком од њих треба да изгледа добро, сценарио је прилагођен и најмањим (вероватно и најчешће коришћеним) мобилним телефонима. Да би форме за попуњавање изгледале добро, биле лаке за употребу и прегледне, издељене су на више корака у којима се уносе информације. На тај начин је и сам процес одабира рашчлањен и у сваком кораку је могуће пружити повратне информације клијенту о томе да ли је унос добар, као и да ли има слободних термина. Такође је омогућено обострано кретање по корацима (и напред и назад), тако да клијент има избора да промени нешто уколико се предомисли. Последњи корак пред потврду заказивања је убачен како би клијент могао на једном месту да види све што је одабрао, па не мора да се враћа на претходне кораке како би проверио унос.

Поставља се питање: да ли је потребно одабирати запосленог код ког се заказује термин или је боље да систем сам одабере коме ће доделити термин? Оно што је честа појава приликом заказивања термина јесте да клијент жели да иде код запосленог код ког иначе иде. Добар пример за то би била услуга шишања код фризера или персонални тренинг код тренера. Управо из тог разлога је омогућен одабир жељеног запосленог.

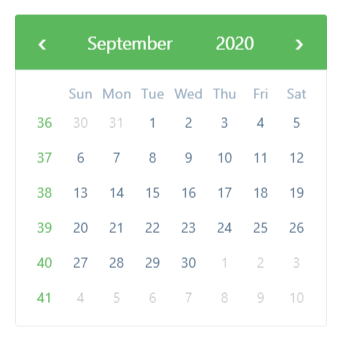
4.7.3 Интерфејс за заказивање термина

Од изузетне је важности да изглед екрана приликом извршавања кључних функционалности система буде примамљив његовим корисницима. То директно утиче на утисак који корисници стичу о апликацији. Поред тога, коришћење система мора бити једноставно. Да би се то постигло, потребно је да интерфејс буде једноставан, интуитиван, али и довољно информативан како би корисник у сваком тренутку знао шта ради. На бази тих начела направљен је интерфејс који се може видети на сликама 3, 4, 5, 6 и 7.

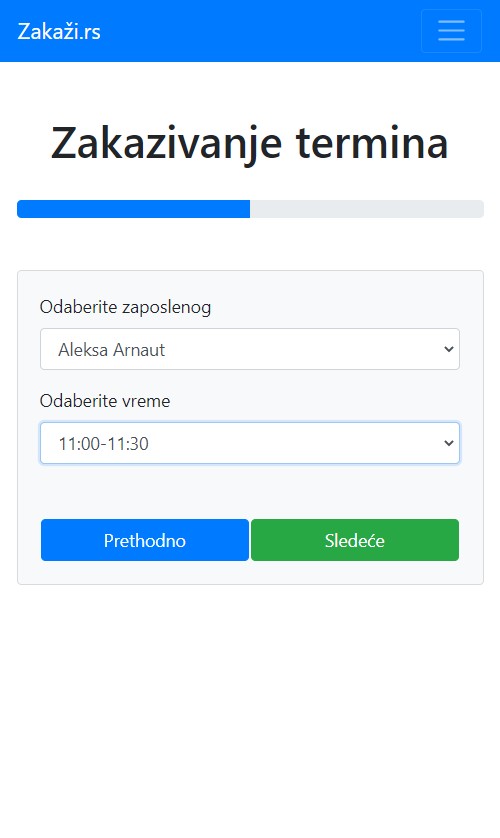
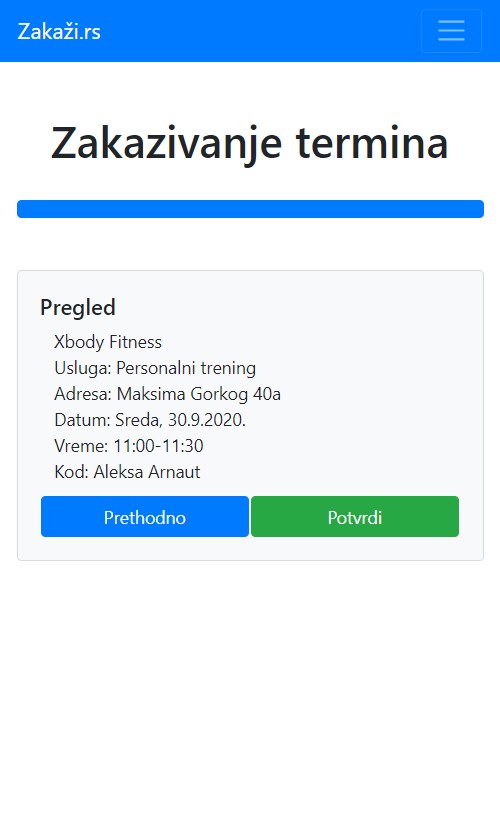
[Слика 3](file:///C:\Users\Lazar\Desktop\diplomski\Систем%20за%20заказивање%20термина.docx#слика_3), односно корак 1 при заказивању термина представља изглед екрана приликом прегледа одабране фирме. Поред основних информација о фирми, постоје и услуге, а кликом на дугме „Zakaži“ прелази се на корак 2. У овом примеру одабрана је услуга „Personalni trening“. У кораку 2 је додат *Progress bar* који је присутан у свим даљим корацима. Он приказује напредак приликом заказивања термина и пружа јасан увид клијенту колико му је остало до краја процеса. Приликом уноса датума у [кораку 2](file:///C:\\Users\\Lazar\\Desktop\\diplomski\\Систем%20за%20заказивање%20термина.docx" \l "слика_4), искаче мали прозор који олакшава одабир. Он приказује за сваки датум ког је дана, као и у којој је недељи. Његов изглед се може видети на [слици 5](Систем%20за%20заказивање%20термина.docx#слика_5).

Слика 3. Заказивање термина, корак 1 Слика 4. Заказивање термина, корак 2



*Слика 5. Прозор за одабир датума*

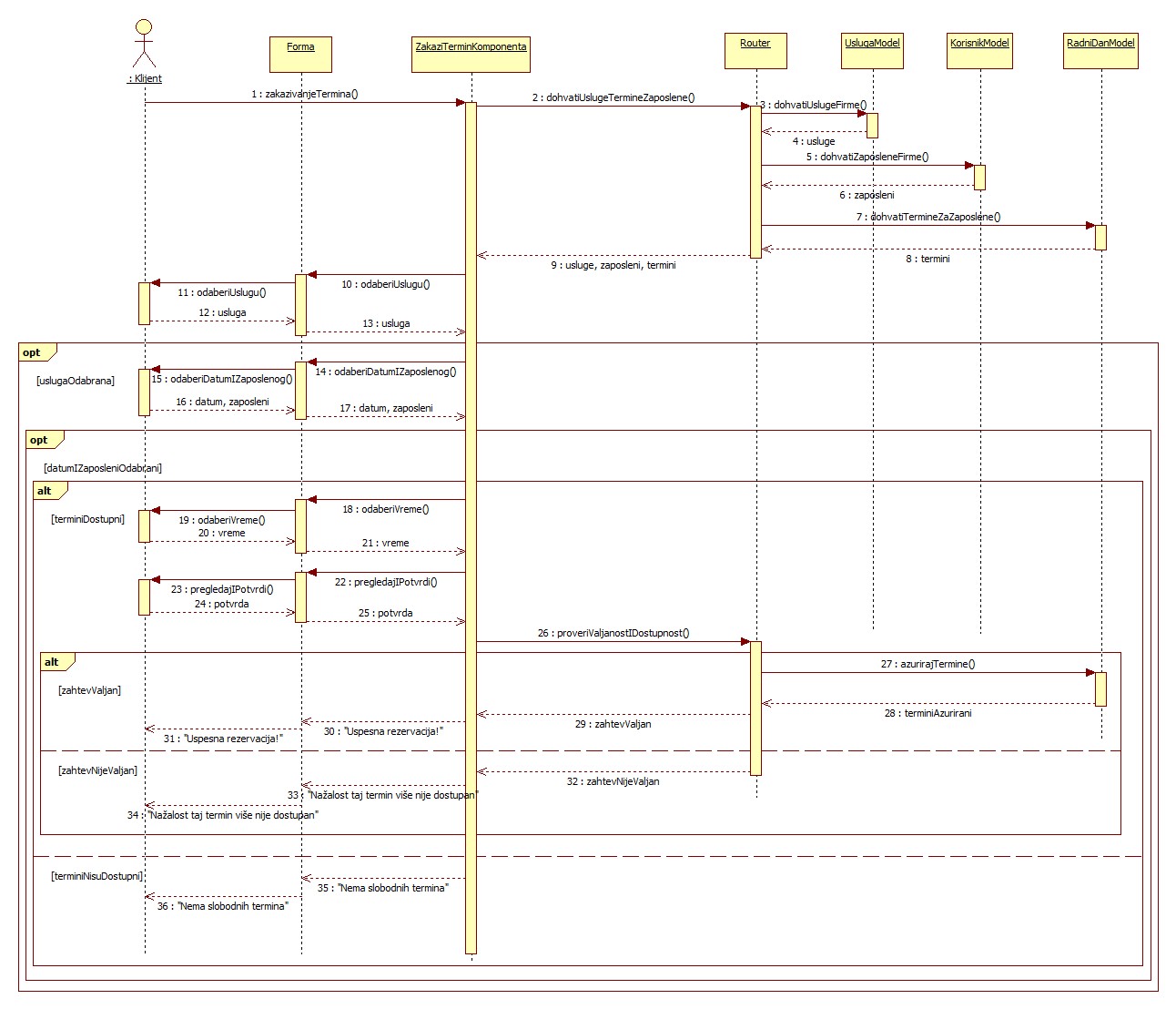
 

*Слика 6. Заказивање термина, корак 3* *Слика 7. Заказивање термина, корак 4*

Након одабраног датума и клика на дугме „Sledeće“, прелази се на [корак 3](Систем%20за%20заказивање%20термина.docx#слика_6). У том кораку је потребно изабрати запосленог, а сходно изабраном запосленом ће се променити вредности у падајућој листи за одабир времена. Након одабира жељених вредности и клика на дугме „Sledeće“, клијент прелази на [корак 4](Систем%20за%20заказивање%20термина.docx#слика_7). Клијент сада види преглед свега што је изабрао, и проверава да ли је сигуран да је то термин који је хтео. Уколико жели нешто да промени, увек се може вратити корак назад кликом на дугме „Prethodno“. Уколико је то управо оно што је желео, кликом на дугме „Potvrdi“ потврђује свој захтев. Систем затим обрађује захтев и као резултат обрађивања исписује поруку у оквиру искачућег прозора *alert* након чега га систем преусмерава на одговарајућу страну.

4.7.4 Детаљи имплементације заказивања термина

На дијаграму секвенце се може лепо видети како систем функционише. Он прати интеракцију корисника са системом, као и како интерагују унутрашњи модули система. Дијаграм је приказан на [слици 8](Систем%20за%20заказивање%20термина.docx#слика_8). Модули Forma и ZakaziTerminKomponenta представљају модуле који припадају *frontend* делу апликације, и они се извршавају унутар претраживача корисника. Модули Router, UslugaModel, KorisnikModel и RadniDanModel се налазе на *backend* делу апликације и извршавају се на серверу. Модели UslugaModel, KorisnikModel и RadniDanModel служе за приступ бази података.



*Слика 8. Дијаграм секвенце за случај употребе заказивање термина*

Цела секвенца почиње клијентовим захтевом за преглед фирме. Након тога, претраживач шаље HTTP захтев серверу да му достави доступне услуге, запослене који их пружају као и термине у којима могу бити одржане. Сервер тада прихвата захтев модулом који се зове *Router*. У питању је саставни део сервера *Express* који служи за рутирање захтева. Он позива методу са изворним кодом која пре свега проверава ваљаност захтева, а затим довлачи одговарајуће информације из базе користећи споменуте моделе. Затим су те информације враћене претраживачу одговарајућим HTTP одговором.

Тада претраживач клијенту приказује пружане услуге. Од тада, из клијентовог угла, секвенца изгледа као што је описано у [тачки 4.7.3](Систем%20за%20заказивање%20термина.docx#тачка_4_8_3). Оно што тамо није описано, јесте како се валидирају унете информације у форми. Прво се проверава да ли је свако од поља у форми попуњено, а онда у зависности од типа информације која се уноси се раде даље провере. Приликом уноса датума, коришћена је компонента *bsDatePicker* која има уграђену валидацију датума. Уколико унета информација није формата датума или јесте формата датума али није исправан датум (нпр. 30. фебруар 2020.) компонента ће вратити као вредност датума Invalid Date. Тада неће бити могућ прелазак на следећи корак заказивања. Што се тиче осталих валидација приликом заказивања термина, оне нису потребне јер се све остале информације уносе путем падајуће листе, па могу имати или валидну вредност или уопште нису унете.

Такође, постоји још један слој који се користи приликом слања захтева серверу, а није приказан овде како би се повећала видљивост осталих модула, а ни на који начин не мења логику функционисања. У питању су класе типа *Service*, односно сервиси. За сваки тип корисника постоји по један сервис чије име одговара имену типа корисника. Оне служе да издвоје комуникацију са сервером од компонената, односно делова *frontend*-а који служе за управљање подацима и изгледа странице. Осим сервиса, постоји и модул који служи за валидацију информација и користи се из истих разлога. Пошто су ти делови издвојени, компоненте имају стриктна задужења. Приликом слања захтева серверу, компоненте зову одговарајуће методе свог сервиса, а приликом валидације зову одговарајуће функције модула за валидацију. На тај начин различите компоненте користе заједничке сервисе и модуле па је постигнута модуларност система, као и већа реупотребљивост кода, а такође је код смањен јер нема потребе копирати га у сваку компоненту у којој се користи.

Након што је клијент прошао кроз све кораке заказивања, и кликнуо на дугме потврди, формира се захтев за заказивање термина који се шаље серверу. Тај захтев се сатоји из следећих информација:

* klijent\_id – идентификатор клијента који заказује термин
* zaposleni\_id – идентификатор запосленог који пружа услугу
* usluga\_id – идентификатор услуге коју клијент заказује
* datum – датум ког ће термин бити одржан
* vreme\_pocetka – број минута од почетка дана када ће термин почети

Време завршетка термина није потребно послати јер се то може закључити на основу почетка термина и на основу трајања услуге, а трајање услуге се може прочитати из базе података на оснобу идентификатора услуге. Време почетка се шаље и чува у бази као број минута од почетка дана јер је тако лакше за коришћење и валидацију него помоћу коришћења два поља, за сат и минут почетка. Преглед целе обраде на серверу је дат у исечку изворног кода испод.

router.post('/termin/zakazi', async (req, res, next) => {

    const result = validacija.validirajTermin(req.body);

    if (result.error) console.log(result.error);

    if (result.error) return res.status(400).json({ poruka: 'Neispravan zahtev.' });

    const zaposleni = await KorisnikModel.findById(req.body.zaposleni\_id).exec();

    if (!zaposleni || zaposleni.tip !== 'zaposleni') return res.status(404).json({ poruka: 'Zaposleni nije pronađen.' });

    const usluga = await UslugaModel.findById(req.body.usluga\_id).exec();

    if (!usluga) return res.status(404).json({ poruka: 'Usluga nije pronađena.' });

    const klijent = await KorisnikModel.findById(req.body.klijent\_id).exec();

    if (!klijent || klijent.tip !== 'klijent') return res.status(404).json({ poruka: 'Klijent nije pronađen.' });

    const vreme\_pocetka = Number(req.body.vreme\_pocetka);

    const vreme\_zavrsetka = vreme\_pocetka + usluga.trajanje;

    const satPocetka = Math.floor(vreme\_pocetka / 60);

    const minutPocetka = vreme\_pocetka % 60;

    let datum = new Date(req.body.datum);

    datum.setHours(satPocetka, minutPocetka, 0, 0);

    let sada = new Date();

    if (datum <= sada) return res.json({ poruka: 'Termin koji se zakazuje mora biti u budućnosti.' });

    let saljiMejl = false;

    datum.setHours(0, 0, 0, 0);

    let danas = new Date();

    danas.setHours(0, 0, 0, 0);

    //ako se zakazuje termin koji je danas kasnije u toku dana, treba obavestiti zaposlenog o tome

    if (datum.getTime() === danas.getTime()) saljiMejl = true;

    let radniDan = await RadniDaniModel.findOne({ datum: datum, zaposleni\_id: zaposleni.\_id }).exec();

    if (!radniDan) return res.status(404).json({ poruka: 'Odabrani zaposleni ne radi odabranog datuma.' });

    let i = 0;

    for (; i < radniDan.termini.length; i++) {

        let termin = radniDan.termini[i];

        if (!(termin.zauzeto) && termin.vreme\_pocetka <= vreme\_pocetka && termin.vreme\_zavrsetka >= vreme\_zavrsetka) {

            let noviTermin = {

                klijent\_id: klijent.\_id,

                usluga\_id: usluga.\_id,

                vreme\_pocetka: vreme\_pocetka,

                vreme\_zavrsetka: vreme\_zavrsetka,

                zauzeto: true

            };

            radniDan.termini.push(noviTermin);

            if (termin.vreme\_pocetka < vreme\_pocetka) {

                let terminPre = {

                    vreme\_pocetka: termin.vreme\_pocetka,

                    vreme\_zavrsetka: vreme\_pocetka,

                    zauzeto: false

                };

                radniDan.termini.push(terminPre);

            }

            if (termin.vreme\_zavrsetka > vreme\_zavrsetka) {

                let terminPosle = {

                    vreme\_pocetka: vreme\_zavrsetka,

                    vreme\_zavrsetka: termin.vreme\_zavrsetka,

                    zauzeto: false

                }

                radniDan.termini.push(terminPosle);

            }

            radniDan.termini.splice(i, 1);

            radniDan.termini.sort((a, b) => { return a.vreme\_pocetka - b.vreme\_pocetka; });

            radniDan.save();

            res.json(radniDan);

            if (saljiMejl) {

                const satZavrsetka = Math.floor(vreme\_zavrsetka / 60);

                const minutZavrsetka = vreme\_zavrsetka % 60;

                slanjeMaila.saljiMail('Klijent je upravo zakazao termin kod Vas za danas',

                    `Poštovani, \n\n` +

                    `Vaš klijent ${klijent.ime} ${klijent.prezime} je zakazao današnji termin kod Vas ` +

                    `od ${ispisNaDveCifre(satPocetka)}:${ispisNaDveCifre(minutPocetka)} do ${ispisNaDveCifre(satZavrsetka)}:${ispisNaDveCifre(minutZavrsetka)} ` +

                    `za uslugu ${usluga.naziv}.\n\n` +

                    `Pozdrav,\n` +

                    `Vaš Zakazi.rs`,

                    zaposleni.email);

            }

            return;

        }

    }

    res.json({ poruka: 'Nažalost, neko je zauzeo taj termin u međuvremenu.' });

});

*Исечак кода 3. Обрада захтева за заказивање термина на серверу*

Када захтев пристигне на сервер, прво се проверава да ли садржи све потребне информације, да ли је свака од њих одговарајућег типа (klijent\_id, zaposleni\_id и usluga\_id морају бити типа ObjectId, односно типа *BSON* дужине 12 бајтова, datum мора бити типа Date, а vreme\_pocetka позитиван цео број, не већи од броја минута у дану) као и да ли садржи само те информације. Уколико то није случај биће враћен одговор пошиљаоцу са статусом *Bad Request* и обрада ће бити прекинута. Затим се проверава у бази података да ли постоје запослени, услуга и клијент са одговарајућим идентификаторима из захтева. Ако било шта од тога није испуњено, захтев није добар и биће враћена одговарајућа порука.

Након тога следи провера датума и времена почетка термина. Почетак термина мора бити у будућности у односу на моменат заказивања термина. Дозвољено је заказивати термин истог дана, али том приликом ће бити послат мејл запосленом како би знао да му је распоред промењен за данашњи дан. Томе служи променљива saljiMejl. Затим следи провера да ли дати запослени уопште ради датог датума. Уколико ради, обрада се наставља. Последња провера јесте да ли је одабрани термин слободан одабраног датума. Уколико није, не мора да значи да је лош захтев, већ је могуће да је неки други клијент у међувремену резервисао термин који се преклапа да временом темрина тренутног захтева.

Коначно, уколико је захтев прошао све те провере, формираће се нови термин. Пошто се у једном радном дану прате и слободни и заузети термини, потребно је изделити слободни термин на један заузети и потенцијално један или два слободна термина. Пример када долази до тога јесте ако је претходно слободан термин био од 10:00 до 12:00 и заказује се термин од 10:30 до 11:00. Тада је потребно уместо слободног термина од 10:00 до 12:00 саставити три одвојена термина и то два слободна (10:00-10:30 и 11:00-12:00) и један заузет (10:30-11:00). Томе служе додатне провере у коду. Након тога, брише се постојећи термин из радног дана и додају се новоформирани термини. Они се затим сортирају по времену почетка и чувају у бази података. Клијенту се шаље одговор да је термин успешно заказан, и врши се слање обавештења запосленом на адресу електронске поште уколико је термин заказан истог дана.

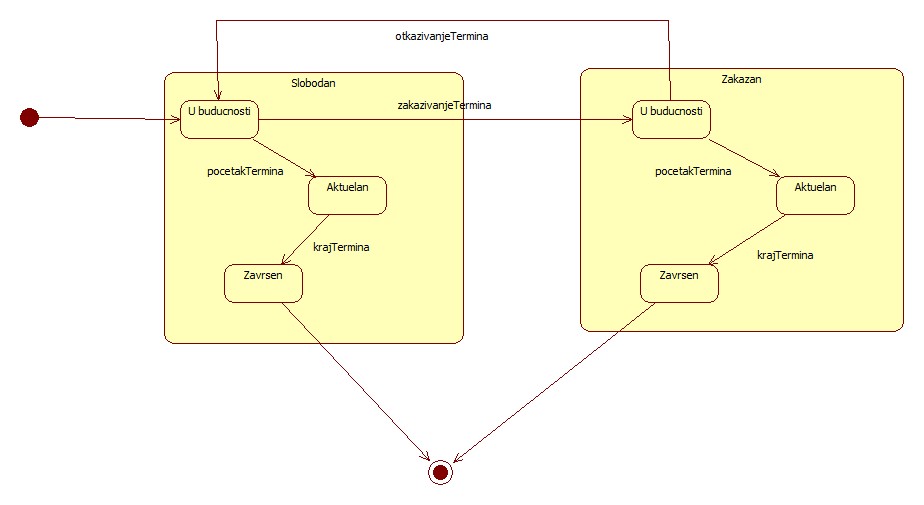
Дакле, у питању је изузетно комплексна провера са много случајева када захтев није исправан. Због тога, уместо угнеждавања *if*-ова са позитивним условима, коришћени су *if*-ови са негативним условима и искакањем из методе у случају испуњења. На тај начин је добијен прегледнији и разумљивији код који је лакши за одржавање.

Такође, добро је напоменути да је реверзни поступак потребно одрадити приликом отказивања термина. Наиме, тада је потребно проверити да ли постоје слободни термини непосредно пре и непосредно после отказаног термина како би се ти везани слободни термини склопили у један. Пример за то би била управо малопређашња три термина, који би се након отказивања резервисаног термина (10:30-11:00) спојила у један, од 10:00 до 12:00. Уколико се отказује термин истог дана одржавања термина, биће послат мејл са обавештењем запосленом.

4.8 Дијаграм стања термина

На дијаграму стања се може лепо видети кроз која стања пролази термин током свог животног века. То је значајно приликом развијања решења како би се знала егзактна стања сваког термина и како не би дошло до тога да термин заврши у недефинисаном стању. Дијаграм стања се може видети на [слици 9](Систем%20за%20заказивање%20термина.docx#слика_9).

На дијаграму се види да термин у погледу заказаности може бити у два различита стања – да буде слободан или заказан. Такође, у оба стања постоје три иста подстања која се односе на животни век термина. Дакле термин може бити у будућности (што значи да још није почео), може бити актуелан (почео је али се није завршио) и може бити завршен. Време је фактор који одређује у којем ће се, од та три подстања, наћи термин. Иницијално је термин слободан и у будућности. Када дође време почетка термина он ће постати актуелан. Од тог тренутка, он не може прећи у друго стање, већ само у подстање завршен стања у ком се налази. Разлог за то је што клијент никако не може заказати нити отказати термин који је већ почео.



*Слика 9. Дијаграм стања термина*

Термин између различитих стања прелази тако што га клијент закаже или откаже. Међутим, важно је нагласити да термин приликом преласка између стања не мора да има исту форму. Може се десити случај као што је описан у [пасусу тачке 4.7.4](Систем%20за%20заказивање%20термина.docx#пасус_тачке_4_8_4). Тада се један термин раставља на три нова термина, од којих један прелази у стање заказан, а друга два остају у стању слободан. Слична ситуација се дешава приликом отказивања термина, када се два слободна и један заказан стапају у један и он је у стању слободан.

Термин прелази из стања актуелан у стање завршен онда када дође време завршетка термина. Код заказаног термина то време зависи од времена трајања услуге која се пружа током датог термина. Када се термин заврши, више не може прећи ни у друго стање ни у било које друго подстање.

5 Корисничко упутство

Корисничко упутство намењено је свим корисницима система и изедељено је према типу корисника. Систем је направљен да буде лак за коришћење и интуитиван, али упутство служи у случају да нешто није довољно јасно. Иако је могуће приступати систему директном комуникацијом са сервером, предвиђено је приступање преко *frontend* дела апликације, за шта је и намењено упутство. На крају, биће дато упутство за покретање система.

5.1 Корисничко упутство за госта

У овој тачки ће бити преглед случајева употребе за госта. За сваки од њих ће бити описано шта су предуслови за његово извршење, затим по корацима шта је потребно урадити како би се успешно извршио, који потенцијални проблеми могу настати приликом извршавања корака и коначно шта је резултат тог извршавања.

5.1.1 Пријављивање на систем

* Предуслови: Гост мора поседовати налог у систему и знати његове креденцијале. Гост се мора налазити у апликацији на почетној страни. Уколико се ту не налази потребно је да кликне на лого апликације како би дошао на страну за пријављивање.
* Кораци:

1. Гост уноси адресу електронске поште
2. Гост уноси лозинку
3. Гост кликне на думге „Prijavi se“

* Могући проблеми: Може се десити да гост не унесе неко поље или неисправно унесе адресу електронске поште или лозинку. У том случају ће бити исписана порука црвеним словима са описом грешке.
* Резултат: Гост се успешно пријавио на систем и бива преусмерен на почетну страну одговарајућег типа корисника.

5.1.2 Регистрација клијента

* Предуслови: Уколико се гост не налази на одговарајућој страници, до ње мора доћи кликом на лого апликације, а затим кликом на дугме „Napravi novi nalog“.
* Кораци:

1. Гост уноси своје име
2. Гост уноси своје презиме
3. Гост уноси своју адресу електронске поште
4. Гост уноси жељену лозинку за приступ налогу
5. Гост кликне на дугме „Региструј се“

* Могући проблеми: Може се десити да гост не унесе неко поље или неисправно унесе адресу електронске поште или унесе лозинку дужине мање од 8 карактера, колико захтева систем због сигурности. У тим случајевима ће бити исписана порука црвеним словима са описом грешке. Уколико гост унесе адресу електронске поште која се већ користи на неком другом налогу, неће моћи помоћу ње да направи нови налог.
* Резултат: Гост се регистровао као клијент у систему и може се пријавити на систем.

5.1.3 Регистрација фирме

* Предуслови: Уколико се гост не налази на одговарајућој страници, до ње мора доћи кликом на лого апликације, затим кликом на дугме „Napravi novi nalog“ и коначно кликом на линк који води на регистрацију фирме.
* Кораци:

1. Гост уноси назив фирме
2. Гост уноси опис фирме
3. Гост уноси адресу на којој фирма послује
4. Гост уноси адресу електронске поште помоћу које ће приступати налогу
5. Гост уноси жељену лозинку за приступ налогу
6. Гост кликне на дугме „Региструј фирму“

* Могући проблеми: Може се десити да гост не унесе неко од поља или неисправно унесе адресу електронске поште или унесе лозинку дужине мање од 8 карактера, колико захтева систем због сигурности. У тим случајевима ће бити исписана порука црвеним словима са описом грешке. Уколико гост унесе адресу електронске поште која се већ користи на неком другом налогу или назив фирме који већ постоји у систему као назив неке друге фирме, добиће поруку о грешци и мораће да промени неисправне параметре како би могао да се региструје.
* Резултат: Гост је регистровао фирму у систему и може се пријавити на систем.

5.2 Корисничко упутство за фирму

У овој тачки ће бити преглед случајева употребе за фирму. За сваки од њих ће бити описано шта су предуслови за његово извршење, затим по корацима шта је потребно урадити како би се успешно извршио, који потенцијални проблеми могу настати приликом извршавања корака и коначно шта је резултат тог извршавања.

5.2.1 Креирање услуге

* Предуслови: Фирма мора бити пријављена на систем.
* Кораци:

1. Фирма кликне на линк у навигацији под називом „Usluge“ и затим се појављује падајућа листа
2. Фирма кликне на линк у падајућој листи по називом „Dodaj uslugu“ и затим се приказује страна са формом за додавање услуге
3. Фирма уноси назив услуге
4. Фирма уноси дужину трајање услуге у минутима
5. Фирма уноси цену пружања услуге по термину у динарима
6. Фирма кликне на дугме „Dodaj uslugu“

* Могући проблеми: Може се десити да фирма не унесе неко од поља или унесе погрешан тип податка. У том случају ће бити исписана порука црвеним словима са описом грешке. Може се десити да та фирма већ пружа услугу са истим називом, па није могуће поново додати исту услугу. Биће исписана порука о томе.
* Резултат: Фирма је додала нову услугу у своју понуду.

5.2.2 Преглед услуга

* Предуслови: Фирма мора бити пријављена на систем.
* Кораци:

1. Фирма кликне на линк у навигацији под називом „Usluge“ и затим се појављује падајућа листа
2. Фирма кликне на линк у падајућој листи по називом „Pregled usluga“

* Могући проблеми: Не постоје могући проблеми.
* Резултат: Фирма види табеларни приказ свих својих услуга, где за сваку услугу пише њен назив, дужина трајања термина као и цена по термину. Уколико фирма није додала ниједну услугу, биће исписана порука о томе.

5.2.3 Креирање запосленог

* Предуслови: Фирма мора бити пријављена на систем.
* Кораци:

1. Фирма кликне на линк у навигацији под називом „Zaposleni“ и затим се појављује падајућа листа
2. Фирма кликне на линк у падајућој листи по називом „Dodaj zaposlenog“ и затим се приказује страна са формом за додавање запосленог
3. Фирма уноси име запосленог
4. Фирма уноси презиме запосленог
5. Фирма уноси адресу електронске поште запосленог
6. Фирма чекира услуге које запослени може да пружи
7. Фирма кликне на дугме „Dodaј“

* Могући проблеми: Може се десити да фирма не унесе неко од поља или унесе погрешан тип податка. У том случају ће бити исписана порука црвеним словима са описом грешке. Може се десити да постоји корисник у систему који већ користи ту адресу електронске поште. У том случају фирма мора променити адресу електронске поште.
* Резултат: Фирма је додала новог запосленог. Систем шаље мејл запосленом у ком га обавештава о томе шта се десило и даје му насумично генерисану лозинку којом се може пријавити на систем.

5.2.4 Преглед својих запослених

* Предуслови: Фирма мора бити пријављена на систем.
* Кораци:

1. Фирма кликне на линк у навигацији под називом „Zaposleni“ и затим се појављује падајућа листа
2. Фирма кликне на линк у падајућој листи по називом „Pregled zaposlenih“

* Могући проблеми: Не постоје могући проблеми.
* Резултат: Фирма види своје запослене табеларно. За сваког запосленог види његово име и презиме као и дугмад за управљање радним временом и услугама запосленог, као и дугме за отпуштање запосленог.

5.2.5 Уклањање запосленог

* Предуслови: Фирма мора бити пријављена на систем. Фирма се мора налазити на страни за преглед својих запослених. Уколико се ту не налази, видети тачку 5.2.4 и доћи дотле.
* Кораци:

1. Фирма кликне на дугме „Otpusti“ запосленог кога жели да уклони

* Могући проблеми: Не постоје могући проблеми.
* Резултат: Фирма је уклонила запосленог из система. Систем шаље мејл са обавештењем клијенту како се више не може пријавити на систем помоћу тог налога.

5.2.6 Преглед радног времена запосленог

* Предуслови: Фирма мора бити пријављена на систем. Фирма се мора налазити на страни за преглед својих запослених. Уколико се ту не налази, видети тачку 5.2.4 и доћи дотле.
* Кораци:

1. Фирма кликне на дугме „Radno vreme“ запосленог коме жели да види радно време и тада се приказује страна за приказ радног времена
2. Фирма одабира датум за који жели да види радно време одабраног запосленог

* Могући проблеми: Може се десити да фирма неисправно унесе датум. Тада ће бити приказана порука о томе.
* Резултат: Фирма види радно време запосленог за одабрани датум. Том приликом су приказани сви термини тог запосленог за тај дан, тако да уколико је термин слободан пише да је слободан, а уколико је термин заузет пише који клијент је заказао коју услугу.

5.2.7 Додељивање радног времена запосленом

* Предуслови: Фирма мора бити пријављена на систем. Фирма се мора налазити на страни за преглед радног времена одабраног запосленог. Уколико се ту не налази, видети тачку 5.2.6 и доћи дотле.
* Кораци:

1. Фирма кликне на дугме „Dodeli radno vreme“ и приказује се форма за унос датума важења тог радног времена
2. Фирма уноси датуме почетка и краја важења радног времена и кликне на дугме „Sledeće“, када се приказује форма за унос почетка и краја радног времена
3. Фирма уноси време почетка и време краја радног времена одабирујући сате и минуте и кликне на дугме „Sledeće“, када се приказује форма за одабир радних дана
4. Фирма чекира радне дане којима ће запослени радити и кликне на дугме „Sledeće“, када се приказује преглед радног времена које ће бити додељено одговарајућем запосленом
5. Фирма прегледа информације и кликне на дугме „Potvrdi“

* Могући проблеми: Може се десити да фирма не унесе неко од поља неке од форми или унесе погрешан тип податка. У том случају ће бити исписана порука црвеним словима са описом грешке. Може се десити да је запосленом већ додељено радно време неког од тих дана, па се неће поново доделити, али ће се исписати порука о томе.
* Резултат: Фирма је доделила радно време одабраном запосленом.

5.2.8 Додела услуга запосленом

* Предуслови: Фирма мора бити пријављена на систем. Фирма се мора налазити на страни за преглед својих запослених. Уколико се ту не налази, видети тачку 5.2.4 и доћи дотле.
* Кораци:

1. Фирма кликне на дугме „Usluge“ жељеног запосленог и тада се приказују све услуге које пружа фирма, док су оне које тренутно пружа одабрани запослени чекиране
2. Фирма одабира услуге које пружа запослени, тако што чекира оне које пружа и одчекира оне које не пружа

* Могући проблеми: Може се десити да фирма не пружа тренутно услуге, па не може доделити ниједну услугу запосленом.
* Резултат: Фирма је изменила услуге које пружа запослени.

5.2.9 Одјављивање са система

* Предуслови: Фирма мора бити пријављена на систем.
* Кораци:

1. Фирма кликне на дугме „Odjavi se“ у навигацији

* Могући проблеми: Не постоје могући проблеми.
* Резултат: Фирма бива одјављена са система и преусмерена на страну за пријављивање на систем.

5.3 Корисничко упутство за запосленог

У овој тачки ће бити преглед случајева употребе за запосленог. За сваки од њих ће бити описано шта су предуслови за његово извршење, затим по корацима шта је потребно урадити како би се успешно извршио, који потенцијални проблеми могу настати приликом извршавања корака и коначно шта је резултат тог извршавања.

5.3.1 Преглед термина

* Предуслови: Запослени мора бити пријављен на систем.
* Кораци:

1. Запослени кликне на лого апликације и приказује му се страна за преглед термина
2. Запослени одабере датум за који жели да види своје термине

* Могући проблеми: Може се десити да запослени неисправно унесе датум. Том приликом ће бити исписана порука о томе.
* Резултат: Запослени види своје термине одабраног датума. Уколико нема термине тог датум биће исписана порука о томе.

5.3.2 Промена лозинке

* Предуслови: Запослени мора бити пријављен на систем.
* Кораци:

1. Запослени кликне на линк у навигацији под називом „Promeni lozinku“ и приказује му се страна са формом за промену лозинке
2. Запослени уноси своју стару лозинку
3. Запослени уноси нову жељену лозинку
4. Запослени кликне на дугме „Promeni lozinku“

* Могући проблеми: Може се десити да запослени не унесе неко од поља, унесе неисправну стару лозинку или да унесе нову лозинку која има мање од 8 карактера. Том приликом ће бити исписана порука црвеним словима са описом грешке.
* Резултат: Запослени је променио своју лозинку, добија потврду о тому у виду *alert*-а и бива одјављен са система и преусмерен на страну за пријављивање на систем.

5.3.3 Одјављивање са система

* Предуслови: Запослени мора бити пријављен на систем.
* Кораци:

1. Запослени кликне на дугме „Odjavi se“ у навигацији

* Могући проблеми: Не постоје могући проблеми.
* Резултат: Запослени бива одјављен са система и преусмерен на страну за пријављивање на систем.

5.4 Корисничко упутство за клијента

У овој тачки ће бити преглед случајева употребе за клијента. За сваки од њих ће бити описано шта су предуслови за његово извршење, затим по корацима шта је потребно урадити како би се успешно извршио, који потенцијални проблеми могу настати приликом извршавања корака и коначно шта је резултат тог извршавања.

5.4.1 Претрага фирми

* Предуслови: Клијент мора бити пријављен на систем. Клијент се мора налазити нa почетној страни за клијента. Уколико се ту не налази потребно је да кликне на лого апликације како би дошао на страну за претрагу фирми.
* Кораци:

1. Клијент уноси назив или део назива фирме или услугу или део назива услуге у поље за претрагу
2. Клијент кликне на дугме „Traži“

* Могући проблеми: Може се десити да клијент не унесе ништа у поље за претрагу. У том случају ће бити исписана порука црвеним словима са описом грешке.
* Резултат: Клијент је претражио фирме. Уколико постоји један или више резултата претраге, они ће се видети испод поља за претрагу. Уколико ништа није пронађено, биће исписана порука са том информацијом.

5.4.2 Преглед фирме

* Предуслови: Клијент мора бити пријављен на систем. Клијент се мора налазити нa страни за претрагу фирми, да је претходно претражио фирме и да је добио бар један резултат претраге. Уколико то није одрадио, клијент треба прво да погледа тачку 5.4.1 за претрагу фирми.
* Кораци:

1. Клијент кликне на дугме „Vidi“ за фирму коју жели да прегледа

* Могући проблеми: Не постоје могући проблеми.
* Резултат: Клијенту је приказан преглед фирме. На тој страници клијент може видети назив, адресу и опис фирме. Испод тога стоји листа услуга за које клијент може да закаже термин.

5.4.3 Заказивање термина

* Предуслови: Клијент мора бити пријављен на систем. Клијент се мора налазити нa страни за преглед фирме. Уколико се не налази ту, треба да прати упутство из тачке 5.4.2 и да дође до прегледа фирме.
* Кораци:

1. Клијент одабира услугу за коју жели да закаже термина и кликне на дугме „Zakaži“ одговарајуће услуге
2. Клијенту се приказује форма за унос датума, затим он уноси датум када жели да закаже термин и кликне на дугме „Sledeće“
3. Клијенту се приказује форма за одабир запосленог, он одабира запосленог и у оквиру исте форме се приказује опција за одабир жељеног времена, затим клијент одабира жељено време и кликне на дугме „Sledeće“
4. Клијенту се приказује преглед информација о заказаном термину, а он након провере да ли је све у реду кликне на дугме „Potvrdi“

* Могући проблеми: Може се десити да клијент не попуни неко од поља неке од форми. У том случају ће бити исписана порука црвеним словима са описом грешке. Може се десити да не постоје термини за одабрани датум. Тада ће систем приказати поруку о томе и препоручити клијенту да одабере неки други датум. Може се десити да након потврде захтева он ипак буде одбијен јер је неки други клијент у међувремену заказао термин који се преклапа са жељеним термином.
* Резултат: Клијент је заказао термин. Биће му приказана порука у виду *alert*-а и биће преусмерен на страну за преглед својих термина.

5.4.4 Преглед заказаних термина

* Предуслови: Клијент мора бити пријављен на систем.
* Кораци:

1. Клијент кликне на линк у навигацији под називом „Moji termini“

* Могући проблеми: Не постоје могући проблеми.
* Резултат: Клијент види своје термине који нису још одржани. За сваки термин пишу датум и време одржавања, назив услуге која је заказана и дугме које води на преглед тог термина. Уколико нема заказаних термина у бућуности биће исписана порука о томе.

5.4.5 Преглед заказаног термина

* Предуслови: Клијент мора бити пријављен на систем. Клијент се мора налазити нa страни за преглед заказаних термина. Уколико се ту не налази треба да прати упутство из тачке 5.4.4 и да дође до прегледа заказаних термина. Клијент мора имати заказане термине у будућности.
* Кораци:

1. Клијент кликне на дугме „Pregled“ термина који жели да прегледа

* Могући проблеми: Не постоје могући проблеми.
* Резултат: Клијенту се приказује страна са прегледом заказаног термина. На тој страни може видети назив и адресу фирме која пружа услугу, затим име и презиме запосленог који пружа услугу и датум, дан и време одржавања термина. Такође, има дугме за отказивање термина.

5.4.6 Отказивање термина

* Предуслови: Клијент мора бити пријављен на систем. Клијент се мора налазити нa страни за преглед заказаног термина. Уколико се ту не налази треба да прати упутство из тачке 5.4.5 и да дође до прегледа заказаног термина.
* Кораци:

1. Клијент кликне на дугме „Otkaži“

* Могући проблеми: Клијенту страница стоји отворена довољно дуго тако да је прошло време одражавања термина. Тада неће бити могуће отказати термин.
* Резултат: Клијенту је отказао жељени термин.

5.4.7 Одјављивање са система

* Предуслови: Клијент мора бити пријављен на систем.
* Кораци:

1. Клијент кликне на дугме „Odjavi se“ у навигацији

* Могући проблеми: Не постоје могући проблеми.
* Резултат: Клијент бива одјављен са система и преусмерен на страну за пријављивање на систем.

5.5 Упутство за покретање система

Систем се састоји из три дела, и то од базе података, *backend* дела апликације и *frontend* дела апликације. Сваки од та три дела покреће се независно. У наставку следи опис покретања сваког од њих.

5.5.1 Покретање сервера за приступ бази података

За сервер базе података коришћен је *MongoDB Server* верзија 4.2.6. Пре свега га је потребно инсталирати, а затим се покреће помоћу команде *mongod.exe*. У систему је коришћен порт 27017, али се то може променити.

5.5.2 Покретање *express* сервера

Пошто *express* ради помоћи *Node.js*-а, потребно је прво инсталирати *Node.js*. Коришћена верзија је 12.14.1. Како систем користи пакете са *Node Package Manager*-а, треба и њега инсталирати, верзију 6.13.4. Након тога, потребно је довући све коришћене пакете, командом *npm install*. Коначно, сервер се покреће командом *node src/server.js*.

5.5.3 Покретање *frontend*-а

*Frontend* је урађен коришћењем *Angular*-а који ради помоћу *Node.js*-а и *Node Package Manager*-а, па је потребно одрадити кораке из претходног пасуса, а затим га инсталирати командом *npm install -g @angular/cli*. Након тога, потребно је довући све коришћене пакете, командом *npm install*. Коначно, командом *ng serve* покреће се *Angular* пројекат.

6 Закључак

На почетку рада су установљени општи захтеви, као и кључне карактеристике које систем треба да поседује. То су биле аутоматизација, прилагодљивост и општост. У следећим пасусима следи преглед успешности обезбеђивања тих карактеристика.

Аутоматизација је дефинитивно постигнута. Како систем сам обезбеђује клијенту преглед слободних термина и исписује одговарајуће поруке, нема сумње да је систем аутоматизован.

Прилагодљивост је постигнута, барем у оним оквирима који су зацртани. Заиста се термини кроје према клијентима и радно време запосленог дели према тим терминима. Постоји простор за побољшање у виду уграђивања више функционалности које могу бити корисне, на пример флексибилније одабирање почетка термина. Могуће је остварити да клијент може у минут или чак и секунд да одабере када почиње његов термин, међутим то би могло довести до појављивања слободних термина у распореду који се не могу попунити, па тако и до мањег искоришћења радног времена и коначно до горе зараде.

Општост је такође постигнута. Било која фирма која се бави пружањем услуга може да користи систем. Међутим, систем није практичан за нека решења. На пример, за фирму која се бави сервисирањем аутомобила, цена сервиса може варирати у зависности од модела возила. Да би фирма понудила сва могућа сервисирања, морала би да дода по једну услугу за сервисирање сваког од возила (нпр. једна услуга би била „сервисирање модела Fiat Punto“, друга би била „сервисирање модела Fiat Panda“ итд.). Иако је могуће тако додавати услуге, није практично. Дакле колико је општост добра ствар, толико се губи на специфичностима различитих области. Управо због тога и имају смисла решења која се групишу према области пословања.

Такође, као општи захтев је било потребно обезбедити угодно корисничко искуство на свим уређајима и интуитиван интерфејс. Како је систем направљен по *mobile-first* принципу, систем је могуће користити на уређајима различитих величина. Напори ка што интуитивнијем интерфејсу су остварени, међутим реалан увид у успешност тих напора је могуће добити тек након примене система над правим корисницима и добијањем повратних информација.

Систем се, наравно, може надоградити. Једна обавезна надоградња пре пуштања система у реалну употребу би било додавање ауторизације. Као једно од добрих решења за коришћене технологије намеће се *JWT*. Добра надоградња би била додавање опција за измену постојећих услуга, и радног времена запослених.

Након пројектовања, имплементације и писања овог документа, уочена је сложеност ове теме. Тежина лежи у балансирању између решења које је потпуно опште и решења које пружа добру подршку за специфичности различитих пословања. Примећено је да остваривање конзистентности система није једноставно и захтева много провера пре свега на серверској страни, као и да модел базе података има велики утицај на комплексност одређених операција.

7 Литература

* Цветановић Милош, “Informacioni sistemi 1”, <http://si3is1.etf.rs/>, приступано септембар 2020.
* “Node.js documentation”, <https://nodejs.org/en/docs/>, приступано септембар 2020.
* “Angular documentation”, <https://angular.io/docs>, приступано септембар 2020.
* “Node Package Manager documentation”, <https://docs.npmjs.com/>, приступано септембар 2020.
* “Bootstrap examples”, <https://getbootstrap.com/docs/4.5/examples/>, приступано септембар 2020.
* “MongoDB documentation”, <https://docs.mongodb.com/>, приступано септембар 2020.
* “Mongoose documentation”, <https://mongoosejs.com/docs/>, приступано септембар 2020.

8 Спискови

8.1 Слике

1. [Дијаграм случајева употребе](Систем%20за%20заказивање%20термина.docx#слика_1) – страна 8.
2. [Модел базе података](Систем%20за%20заказивање%20термина.docx#слика_2) – страна 9.
3. [Заказивање термина, корак 1](Систем%20за%20заказивање%20термина.docx#слика_3) – страна 17.
4. [Заказивање термина, корак 2](Систем%20за%20заказивање%20термина.docx#слика_4) – страна 17.
5. [Прозор за одабир датума](Систем%20за%20заказивање%20термина.docx#слика_5) – страна 17.
6. [Заказивање термина, корак 3](#слика_6) – страна 18.
7. [Заказивање термина, корак 4](#слика_7) – страна 18.
8. [Дијаграм секвенце за случај употребе заказивање термина](Систем%20за%20заказивање%20термина.docx#слика_8) – страна 19.
9. [Дијаграм стања термина](Систем%20за%20заказивање%20термина.docx#слика_9) – страна 24.

8.2 Исечци кода

1. [Обрада захтева за креирање нове услуге на серверу](Систем%20за%20заказивање%20термина.docx#исечак_кода_1) – страна 12.
2. [Валидација документа услуге](Систем%20за%20заказивање%20термина.docx#исечак_кода_2) – страна 13.
3. [Обрада захтева за заказивање термина на серверу](Систем%20за%20заказивање%20термина.docx#исечак_кода_3) – страна 20.

8.3 Скраћенице

* итд. – и тако даље
* нпр. – на пример
* и сл. – и слично
* тј. – то јест