|  |  |  |
| --- | --- | --- |
|  | УНИВЕРЗИТЕТ У НОВОМ САДУ  **ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У НОВОМ САДУ** |  |

Лазар Стојчевић

**Систем за анонимизацију података из AWS DynamoDB базе података**

ДИПЛОМСКИ РАД

- Основне академске студије -

Нови Сад, 2023.

|  |  |  |
| --- | --- | --- |
|  | Универзитет у Новом Сад, **Факултет техничких наука**  21000 НОВИ САД, Трг Доситеја Обрадовића 6 | Датум: |
|  |
| **ЗАДАТАК ЗА ИЗРАДУ ДИПЛОМСКОГ (BACHELOR) РАДА** | Лист/Листова: |
|  |

*(Податке уноси предметни наставник - ментор)*

| Врста студија: | Основне академске студије |
| --- | --- |
| Студијски програм: | **Рачунарство и аутоматика** |
| Руководилац студијског програма: | **проф. др Милан Рапаић** |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент: | **Лазар Стојчевић** | Број индекса: | **RA 84/2018** |
| Област: | **Безбедност у системима електронског пословања** | | |
| Ментор: | **проф. др Горан Сладић** | | |
| НА ОСНОВУ ПОДНЕТЕ ПРИЈАВЕ, ПРИЛОЖЕНЕ ДОКУМЕНТАЦИЈЕ И ОДРЕДБИ СТАТУТА ФАКУЛТЕТА ИЗДАЈЕ СЕ ЗАДАТАК ЗА ДИПЛОМСКИ (Bachelor) РАД, СА СЛЕДЕЋИМ ЕЛЕМЕНТИМА:   * проблем – тема рада; * начин решавања проблема и начин практичне провере резултата рада, ако је таква провера неопходна; * литература | | | |

**НАСЛОВ ДИПЛОМСКОГ (BACHELOR) РАДА:**

|  |
| --- |
| **Систем за анонимизацију података из AWS DynamoDB базе података** |

**ТЕКСТ ЗАДАТКА:**

|  |
| --- |
| Анализирати принципе анонимизације података. Упознати се са AWS платформом са посебним освртом на AWS DynamoDB. Специфицирати и имплементирати сервис за анонимизацију података за DynamoDB базу података. Реализовати веб апликацију која ће за користити наведени сервис за анонимизацију података. Документовати решење. |

|  |  |
| --- | --- |
| Руководилац студијског програма: | Ментор рада: |
|  |  |

|  |
| --- |
| Примерак за:  - Студента;  - Ментора |

**САДРЖАЈ**

[1. УВОД 7](#_Toc125278289)

[2. Основни појмови анонимизације 9](#_Toc125278290)

[2.1 Начини анонимизације 9](#_Toc125278291)

[2.1.1 Маскирање података 9](#_Toc125278292)

[2.1.2 Псеудонимизација 10](#_Toc125278293)

[2.1.3 Генерализација 10](#_Toc125278294)

[2.1.4 Пермутација података 11](#_Toc125278295)

[2.1.5 Пертурбација 11](#_Toc125278296)

[2.1.6 Синтетички подаци 12](#_Toc125278297)

[2.2 Избор метода за анонимизацију 12](#_Toc125278298)

[3. NoSQL базе података 13](#_Toc125278299)

[3.1 Предности NoSQL база података 13](#_Toc125278300)

[3.2 Мане NoSQL база података 14](#_Toc125278301)

[3.3 Amazon DynamoDB база података 14](#_Toc125278302)

[3.3.1 Моделовање података 15](#_Toc125278303)

[4. Апликација за заказивање лекарских прегледа 17](#_Toc125278304)

[4.1 Опис апликације 17](#_Toc125278305)

[4.2 Архитектура система 18](#_Toc125278306)

[4.3 Спецификација система 19](#_Toc125278307)

[5. Имплементација система 23](#_Toc125278308)

[5.1 Нерегистровани корисник 23](#_Toc125278309)

[5.2 Пацијент 24](#_Toc125278310)

[5.3 Лекар 28](#_Toc125278311)

[5.4 Рачуновођа 31](#_Toc125278312)

[5.5 Администратор 33](#_Toc125278313)

[6. ЗАКЉУЧАК 47](#_Toc125278314)

[ЛИТЕРАТУРА 49](#_Toc125278315)

[БИОГРАФИЈА 51](#_Toc125278316)

[КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА 53](#_Toc125278317)

[KEY WORDS DOCUMENTATION 55](#_Toc125278318)

# УВОД

У информационој ери лична приватност је све значајнија тема, како за кориснике, тако и за компаније. Све више услуга на које се ослањамо постају дигитализоване и маркетинг постаје све више персонализован, па компаније увек траже интелигентније начине да користе податке. Ове иновације не смеју бити на штету сигурног чувања података и заштите приватности. Ако компаније користе податке корисника оне имају обавезу да их заштите. Анонимизација података је кључни део заштите и сигурности података. Ниједна компанија не жели да буде намерни или случајни преступник. Такође обезбеђивање сигурности података ја све више кључни начин да се стекне поверење у бренд компаније, исто тако цурење података је брз начин да се изгуби репутација на тржишту [33].

Анонимизација је процес чувања приватних или осетљивих информација брисањем или шифровањем идентификатора који повезују појединца са податком. Међутим, чак и кад су идентификатори из података, нападачи могу да користе методе деанонимизације да обрну процес анонимизације. Пошто подаци обично пролазе кроз неколико извора, од којих су неки јавни, технике деанонимизације могу да испрате изворе и открију личне информације уколико анонимизација није правилно изведена [3].

Тема овог рада је имплементација сервиса за анонимизацију података који су складиштени у AWS DynamoDB бази података [1] као и демонстрација рада сервиса за анонимизацију унутар *web* апликације за заказивање лекарских прегледа.

Као пример за демонстрацију одабран је овај тип апликације зато што овај тип услуге свакодневно користе милиони људи и као таква генерише велику количину осетљивих података. Ова апликација представља симулацију једног таквог система.

Апликација за заказивање лекарских прегледа имплементира различите врсте корисника где свака врста корисника има приступ одређеном делу информација, код ког се информације којима корисник не сме да има приступ приказују анонимизовано. Администратор система има могућност анонимизације целе базе података у фајл JSON формата [2].

У другом поглављу ће бити објашњени основни појмови анонимизације, као и начини на које је могуће обавити исту.

У трећем поглављу ће акценат бити стављен на AWS DynamoDB базу података и начин на који она функционише.

У четвртом поглављу ће бити описан модел *web* апликације за заказивање лекарских прегледа и њеног сервиса за анонимизацију података, као и случајеви коришћења апликације.

У петом поглављу ће бити описана конкретна имплементација система. Биће приказани делови апликације и код којим је сама апликација и сервис за анонимизацију имплементиран, као и изглед анонимизованих података.

# Основни појмови анонимизације

Анонимизација података је уклањање или модификовање осетљивих података са циљем да се уклоне подаци који би открили идентитет лица коме ти подаци припадају. Анонимизација података минимализује шансу цурења података када се подаци шаљу изван оригиналног система [4].

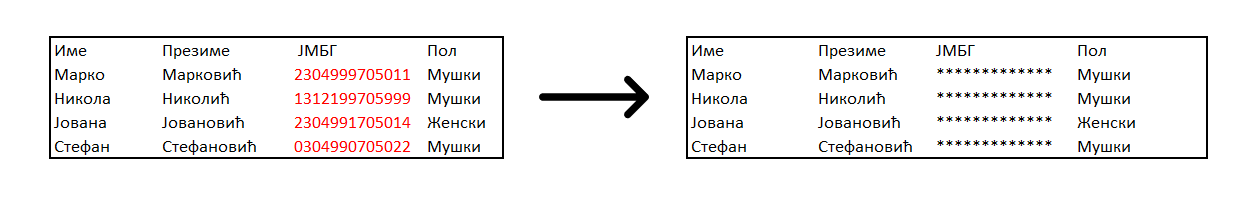
## 2.1 Начини анонимизације

### Маскирање података

Маскирање података је метод анонимизације који за циљ има креирање лажних података, али довољно „реалних“ да они могу да се користе у обучавању особља, демоа апликације или тестирању апликације. Циљ овог приступа је да у потпуности уклони могућност обрнутог инжењеринга. Може се обављати следећим техникама [5]:

* Замена целог или дела податка карактером као што су “x” или “\*”. Маскирање дела неког податка се може обавити над фиксним бројем карактера, као у случају кредитне картице, или над променљивим бројем карактера као у случају мејла. Потребно је анализирати податке како би се одлучило који део и величина податка је погодна за сакривање, како би се онемогућила деанонимизација, а са друге стране очувала семантичка вредност података
* Замена места карактерима унутар податка
* Замена речи или карактера
* Постављање *null* вредности
* Шифровање са очувањем формата (једини метод који је реверзибилан, уколико се открије кључ шифровања, што представља потенцијални сигурносни ризик) [6]

На слици 2.1 приказано је како подаци изгледају пре и како после маскирања осетљивих података.



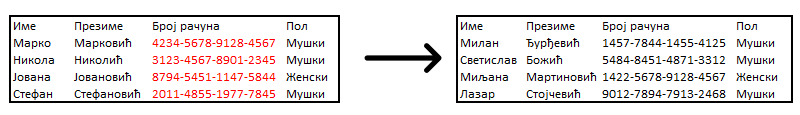
Слика 2.1. Резултат маскирања осетљивих података

### Псеудонимизација

Псеудонимизација је сигурносна техника за замену осетљивих података са реалистичним лажним подацима који се не могу повезати са оригиналним корисником без додатних информација. Такође подаци треба да очувају референцијални интегритет и статистичку тачност како би даљи пословни процеси са њима били могући [7].

Псеудонимизација се користи када су за потребе даљег тестирања, приказивања демоа апликације, анализе или тренинга запослених потребни што реалистичнији подаци [8].

На слици 2.2 приказани су подаци пре и после псеудонимизације.

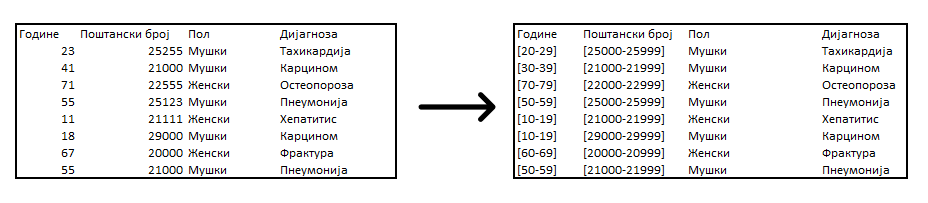


Слика 2.2. Резултат псеудонимизације података

### 2.1.3 Генерализација

Генерализација представља сигурносну технику заштите података где се из оригиналних податка уклањају одређени детаљи како би био мање препознатљив. Најчешће се обавља тако што се податак представља као опсег унутар ког припада или простим уклањањем делова податка. Циљ је елиминација било какве могућности идентификације корисника преко датог податка, али и одржавање одређене тачности податка. [3]

На слици 2.3. приказано је како подаци изгледају пре, а како после поступка генерализације.



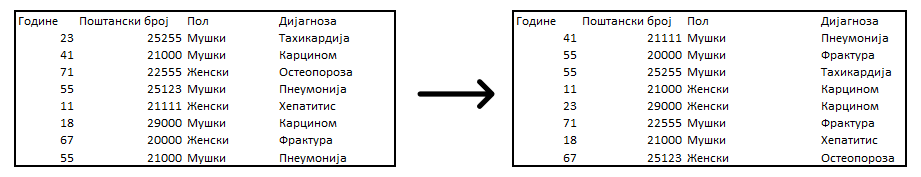
Слика 2.3. Резултат генерализације података

### 2.1.4 Пермутација података

Сигурносна техника која се користи да би се преуредио скуп података како он не би више представљао оригинални скуп података. При пермутовању података води се рачуна о томе које се колоне података пермутују јер неки подаци имају већи утицај на анонимизацију од других. Овај приступ се најчешће примењује на великим скуповима података [9].

Предност ове методе је што подаци остају погодни за даље тестирање система, док је мана што уколико се открије алгоритам пермутације подаци се могу деанонимизовати. [10].

На слици 2.4. је приказан пример података пре и после поступка пермутације података.



Слика 2.4. Резултат пермутације података

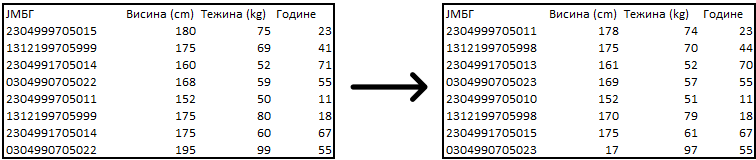
### 2.1.5 Пертурбација

Представља сигурносну технику анонимизовања података тако што се оригиналном скупу података додаје одређена количина шума. За разлику од генерализације, овај метод не може да гарантује истинитост података па чак и уколико је промена минимална [11].

Ниво шума који се додаје подацима треба бити пропорционалан величини опсега вредности података, другим речима, уколико је основа мала промена ће бити мања, док уколико је основа велика разлика између почетног и анонимизованог податка може бити велика [11].

Овом методом се чувају неки статистички подаци као што су средња вредност или корелација. Често се у пракси пертурбација комбинује са пермутацијом за категоричке вредности. Комбинацијом ова два метода добију се анонимизовани подаци, али се њихова семантичка вредност губи [11].

На слици 2.5. приказан је пример података пре и после пертурбације.



Слика 2.5. Резултат пертурбација података

### 2.1.6 Синтетички подаци

Синтетички подаци су у суштини лажни подаци добијени коришћењем вештачке интелигенције на основу података из реалног система. Уколико се процес исправно обави, подаци ће очувати сакривену структуру података и очуваће статистичке параметре почетног сета података. Често се овај метод примењује парцијално, само над оним подацима који се сматрају осетљивим [12].

Предност овог приступа је та што у потпуности очувава статистичку структуру података и самим тим нови скуп података је погодан за машинско учење над њим или тренинг запослених [12].

## 2.2 Избор метода за анонимизацију

Један од главних фактора који утичу на избор коју технику анонимизације треба користи је контекст у коме ће се анонимизовани подаци користити. Зависно од карактеристика података, неке од метода анонимизације ће бити мање ефективне или неће имати смисла над одређеним групама података. Стога пре самог почетка анонимизације врло је битно разумети оквир података. Још једна битна одлика податка коју треба узети у обзир је да ли је сам атрибут који се анонимизује нумерички или категорички [11].

Нумерички подаци су, као што им само име каже, бројеви. Према томе технике анонимизације које сакривају или замењују вредност атрибута изазивају значајан пад корисности података [32].

Категорички подаци су они који могу бити виђени као различите и коначне групе засноване на квалитативним обележјима [33].

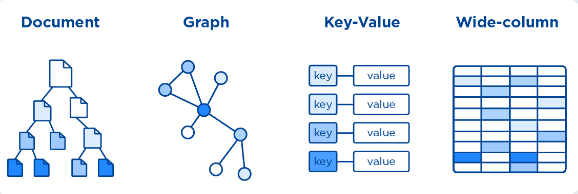
У оба случаја анализа садржаја атрибута ће одредити која техника је најпогоднија на основу корисности атрибута после анонимизације датом методом и могућности реидентификације [11].

# NoSQL базе података

Сам назив овог типа база података је потекао из “non-SQL” или “non-relational”, што се односи на то да се ове базе података за чување података не ослањају на релационе табеле. Појавиле су се крајем 2000-их како се цена складиштења података драстично смањивала а примарни трошак развоја и одржавања софтвера су постали програмери. Тако су и NoSQL базе података оптимизоване како би повећале продуктивност програмера. Типови NoSQL база података [16]:

* **Складишта докумената** – чува податке у документима облика сличног JSON (JavaScript Object Notation), где сваки документ садржи парове кључ-вредност.
* **Кључ**-**вредност** – сваки објекат сачуван у бази података је представљен као пар кључ-вредност.
* **Прошириве табеле** – вредности се чувају у табелама али за разлику од релационих табела формати колона могу да варирају од реда до реда.
* **Граф базе** – Подаци се чувају у чворовима, док се везе између података представљају везама између чворова.

Илустрације структура података NoSQL базе података могу се видети на слици 3.1.



Слика 3.1. Илустрације структура података NoSQL база података [22]

## 3.1 Предности NoSQL база података

NoSQL базе података не представљају савршено решење складиштења података али са собом носе значајна побољшања на одређеним пољима [17]:

* **Скалабилност** постигнута је растављањем једне основне базе података на кластере. За повећање капацитета је довољно само додати нове рачунаре у кластер.
* **Не захтева структуиране податке –** за разлику од релационих база података где се база служи предефинисаном шемом података где се подаци трансформишу при читању и писању, NoSQL базе података чувају податке на начин на који се они користе у апликацију, без обавезе да пратимо стриктно шему података.
* **Developer-friendly –** Подаци који су сачувани у бази података су много сличнији оним што се користи у апликацији па су самим тим лакши за тумачење.

## 3.2 Мане NoSQL база података

Поред бројних предности овај приступ носи поједине потешкоће са собом. Неке од мана овог приступа чувању података [20]:

* Свака NoSQL база података има јединствену синтаксу управљања подацима и упитима, насупрот SQL базама података.
* Уклањањем шеме базе података уклоњена су и ограничења над подацима које је она обезбеђивала.
* Пошто база података не води рачуна о структури података, апликација мора имплицитно то да ради.
* Не пружају исти ниво конзистентности као SQL базама података, пошто се већина NoSQL база података заснива на парадигми евентуалне конзистентности [21]. Подаци ће повремено бити неконзистентни, што је у одређеним системима недопустиво.

## 3.3 Amazon DynamoDB база података

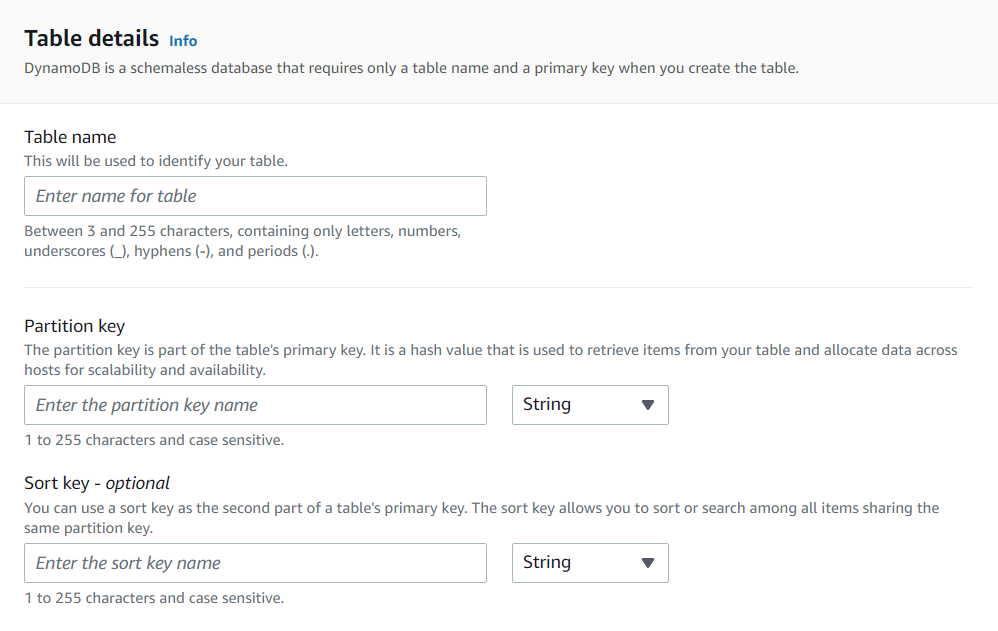
DynamoDB је NoSQL (Not SQL) база података чије се услуге пружају путем AWS-а (Amazon Web Services) [13]. Иницијално је представљена 18. јануара 2012. године и представљена је као унапређена верзија претходне Амазонове NoSQL базе података Sampled [14]. DynamoDB је кључ-вредност тип NoSQL базе података [15]. Септембра 2013. године Амазон је представио локалну верзију DynamoDB базе података како би програмери могли да тестирају своје апликације и без приступа интернету [18].

### 3.3.1 Моделовање података

Структура базе података се моделује тако да једна табела буде повезана са једним моделом, нешто слично као код релационих база података, са тим да табела нема стриктно дефинисану шему. Табела се проширује како се унесе нови објекат са пољем које до тада није унето. Осталим објектима се додаје то поље али нема вредност (“unknown”) [34].

Свака табела има примарни кључ којег формирају један или више њених атрибута. Дефинисани примарни кључ уводи ограничење да сваки креирани ред у табели мора имати обележја која чине примарни кључ. Примарни кључ мора бити скаларна вредност. Партициони кључ, мора бити део примарног кључа и мора бити дефинисан, одређује на којој партицији ће податак бити сачуван (уколико их има више) [34].

Уколико партициони кључ није уједно и примарни кључ, корисник може да дефинише и кључ за сортирање (Sort key), који ће омогућити претрагу и сортирање података који деле исти партициони кључ [19]. На слици 3.2. приказана је форма креирања нове табеле.



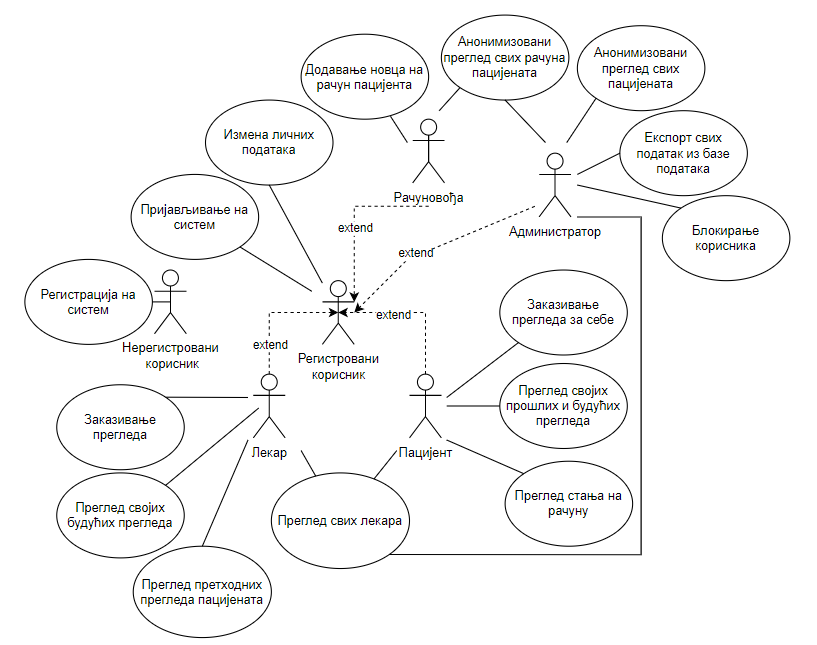
Слика 3.2. Креирање нове табеле у бази података

# Апликација за заказивање лекарских прегледа

## 4.1 Опис апликације

Апликација за заказивање лекарских прегледа је *web* апликација која за циљ има да пацијентима омогући што лакше заказивање термина за лекарски преглед. Пацијенти бирају жељени датум прегледа и трајање где након тога добијају слободне термине одговарајуће дужине за тај дан. Пацијенти имају на увид све своје претходне прегледа са њиховим извештајима и све будуће прегледе. Пацијенти имају могућност регистрације путем форме. Поред улоге пацијента у апликацији постоје још улоге лекара, администратора и књиговође. Лекар на увид има све своје будуће термине, има на увид све извештаје прегледа пацијента који има термин код њега и може да закаже тренутном пацијенту преглед код неког другог лекара или себе за време трајања прегледа. Лекар такође пише извештај за тренутни преглед, одређује цену прегледа и завршава преглед. Рачуновођа има на увид финансијске рачуне пацијената и може да мења износе на рачунима уколико пацијент уплати новац. Администратор има на увид све кориснике система и има могућност да блокира корисника уколико за то има потребе. Администратор такође има могућност извлачења целе базе података анонимизоване маскирањем или псеудонимизацијом у JSON формату. Сваки корисник има могућност мењања својих основних података.

Слика 4.1. приказује случајеве коришћења апликације за заказивање лекарских прегледа.



Слика 4.1. Случајеви коришћења апликације за заказивање лекарских прегледа

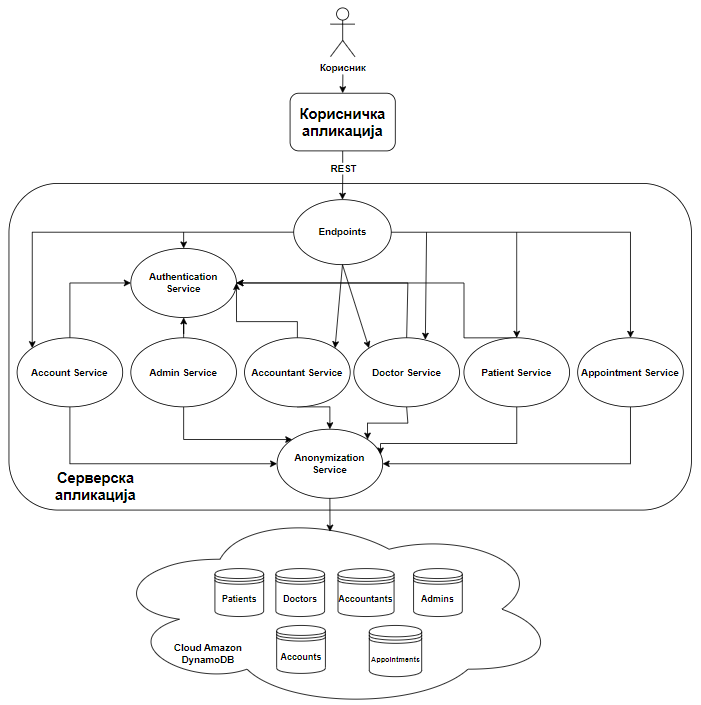
## 4.2 Архитектура система

Апликација за заказивање лекарских прегледа састоји се из 2 главна могула:

* Клијентска апликација – Апликација која обезбеђује графички интерфејс за коришћење апликације.
* Серверска апликација – Имплементира целокупну пословну логику система. Имплементирана је као монолитна апликација. Сервис за анонимизацију је имплементиран као саставни део серверске апликације [22].

Комуникација између ова два модула врши се путем HTTP протокола [25].

На слици 4.2. приказан је модел система апликација за заказивање лекарских прегледа.



Слика 4.2. Архитектура апликације за заказивање лекарских прегледа

## 4.3 Спецификација система

Како је већ наведено у претходном поглављу апликација за заказивање лекарских прегледа састоји се из два модула, клијентске апликације и серверске апликације.

Клијентска апликације имплементирана је помоћу Angular 13 [24] радног оквира. На основу радног оквира коришћене су додатне библиотеке које су олакшале развој саме апликације. Коришћена је библиотека Angular Material [25], библиотека корисничког интерфејса која садржи готову имплементацију дизајна компоненти корисничког интерфејса и RxJS [26] библиотека која олакшава рад са асинхроним позивима.

Серверска апликације имплементирана је у .NET 6 [25] радном оквиру. За имплементацију конкретних *endpoint-a* коришћена је библиотека FastEndpoints [28] са циљем да имплементира REPR дизајн шаблон (Request-Endpoint-Response) [29]. Серверска апликација као складиште података користи *cloud* верзију AWS DynamoDB базе података [1]. Сервис за анонимизацију података имплементиран је у оквиру серверске апликације, при чему као помоћна библиотека при анонимизације је коришћена Lorem.NET библиотека [30].

На слици 4.3. приказан је дијаграм модела ентитета којима манипулише апликација за заказивање лекарских прегледа.

Класа *Patient* представља модел пацијента. Она садржи идентификациона поља (име, презиме, корисничко име, шифра за пријаву на систем, адреса, лични идентификациони број, датум рођења и да ли је корисник активан) као и контакт податке као што су мејл адреса и број телефона. Класа такође садржи поља са подацима од медицинског значаја (висина, тежина и крвна група).

Свака класа у систему има поље “Id” путем ког се њени објекти јединствено идентификују у систему.

Класа “Accountant” представља модел рачуновође и садржи поља која представљају његове идентификационе информације, као и контакт податке (број телефона и мејл адреса).

Класа “Doctor” моделује лекара и поред идентификационих и контакт поља, као претходно две описане класе, она сржи поље *MedicalSpeciality* које представља ужу специјализацију лекара, уколико је има.

Класа “Admin” моделује администратора и поседује идентификациона и контакт поља као три претходно описане класе.

Класа “Account” представља модел рачуна пацијента. Сваки рачун је повезан са тачно једним пацијентом и сваки пацијент је повезан са тачно једним рачуном. Класа има поља која представљају број рачуна и тренутно стање на рачуну.

Класа “Appointment” представља модел једног термина лекарског прегледа. Сваки преглед је повезан са једним пацијентом и једним доктором. Сваки пацијент и доктор могу бити повезани са више термина. Ова класа садржи поља почетка и краја термина, као и извештај и цену уколико је термин обављен.

На слици 4.4. приказан је дијаграм активности главних функционалности сваке улоге у систему за заказивање лекарских прегледа.

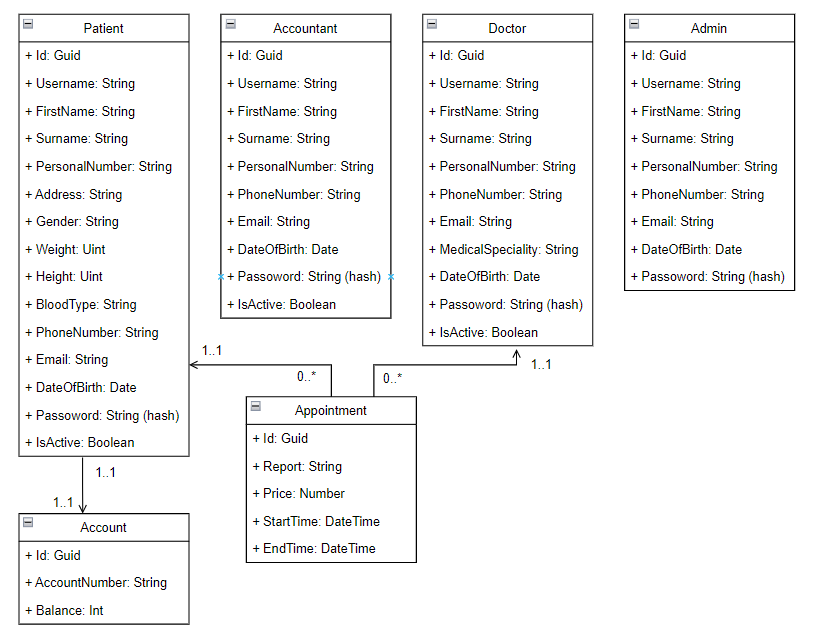
Сваки корисник да би успешно обавио било коју акцију први мора да унесе исправне креденцијале. Након чега му сервер доставља токен за аутентификацију.

Пацијент да би заказао нови лекарски преглед прво мора да оде на страницу свих својих прегледа где му серверска апликација доставља све његове прошле и будуће прегледа. Пацијент бира опцију да закаже нови лекарски преглед. Након чега бира жељеног лекара и датум прегледа. Серверска апликација му затим добавља све слободне термине тог лекара за изабрани датум. Пацијент након одабира термина и чувања истог се враћа на преглед прегледа и може заказати нови преглед уколико жели.

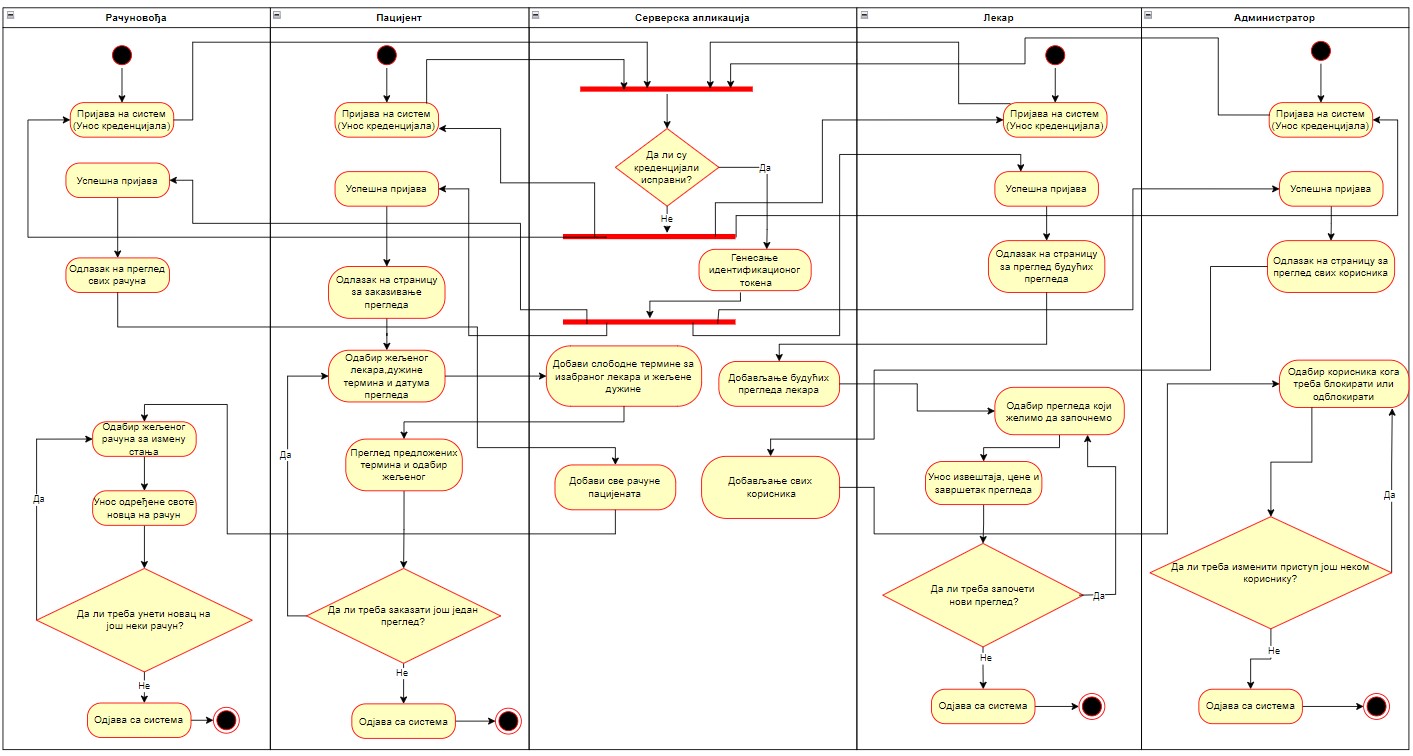
Рачуновођа да би унео новац на неки од рачуна пацијената прво одлази на преглед свих рачуна пацијената где му серверска апликација добавља анонимизоване податке рачуна пацијената. Рачуновођа бира жељени рачун и на њега уноси извесну своту новца.

Лекар да би обавио неки од прегледа за које је задужен прво одлази на страницу са свим својим будућим прегледима где му серверска апликација добавља све његове будуће прегледа. Лекар бира неки од својих будућих прегледа и започиње изабрани преглед. Да би завршио преглед лекар уноси извештај и цену прегледа, након чега се враћа на страницу свих својих будућих прегледа одакле може започети следећи преглед.

Администратор да би блокирао или одблокирао корисника прво одлази на страницу за приказ свих корисника где му серверска апликација добавља анонимизоване податке свих корисника од којих администратор проналази жељеног корисника кога жели да блокира или одблокира.



Слика 4.3. Дијаграм класа ентитета у систему



Слика 4.4. Дијаграм активности система за заказивање лекарских прегледа

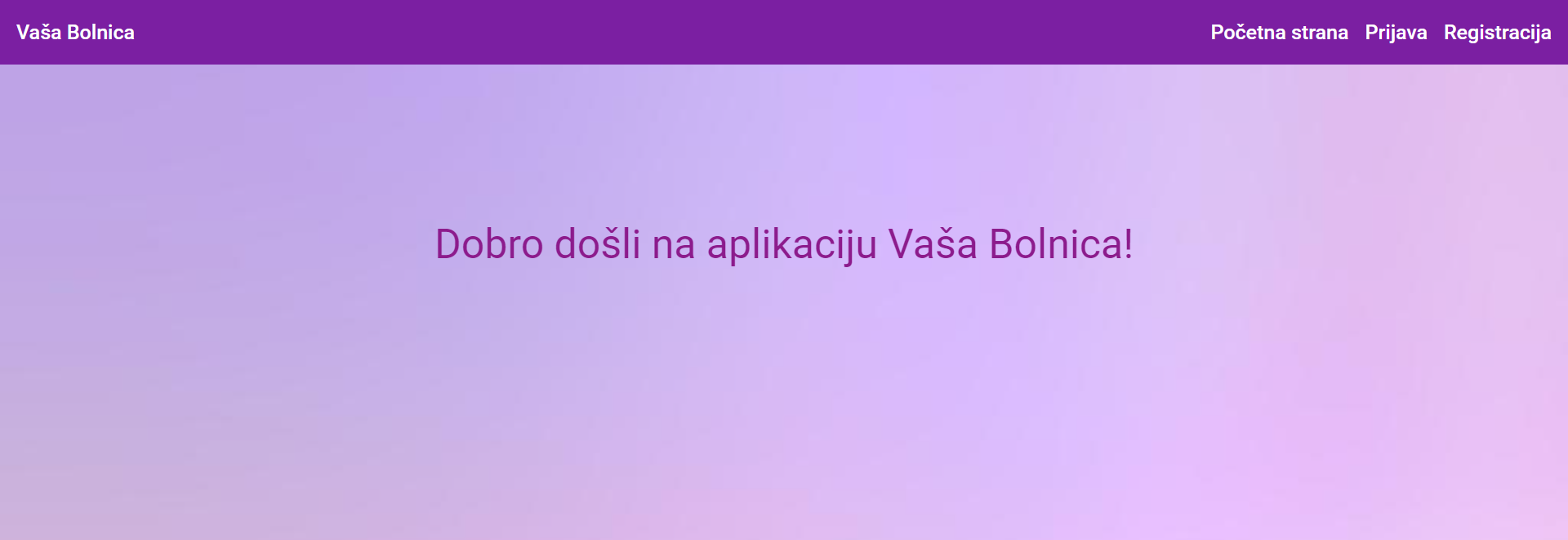
# 5. Имплементација система

У наставку ће имплементација система бити описана кроз улоге коју корисници могу имати у систему и функционалности које су им доступне.

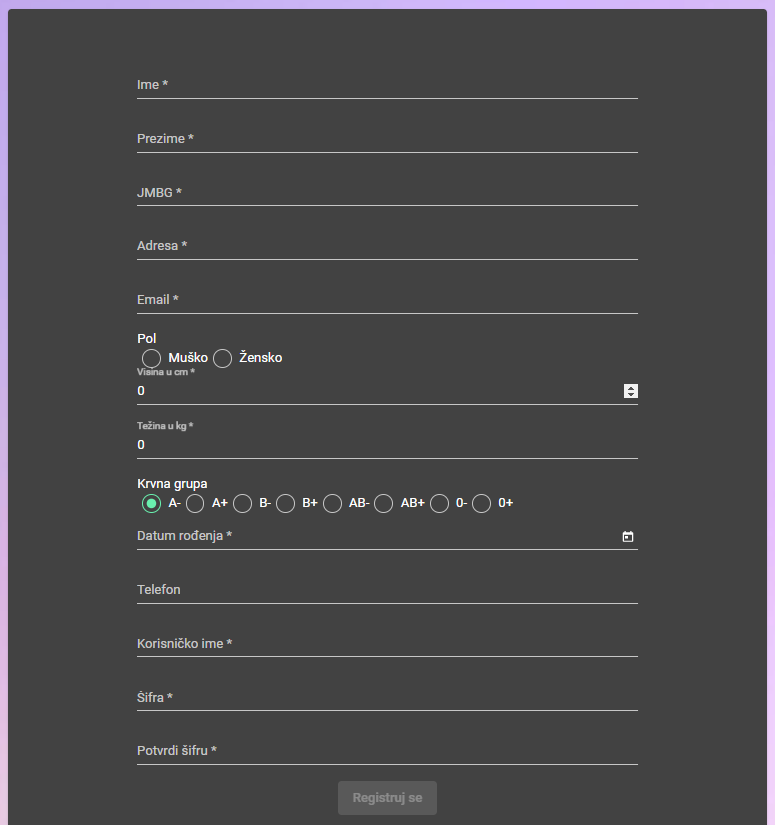
### 5.1 Нерегистровани корисник

Нерегистровани корисник при отварању апликације има на увид почетну страницу приказану на слици 5.1.

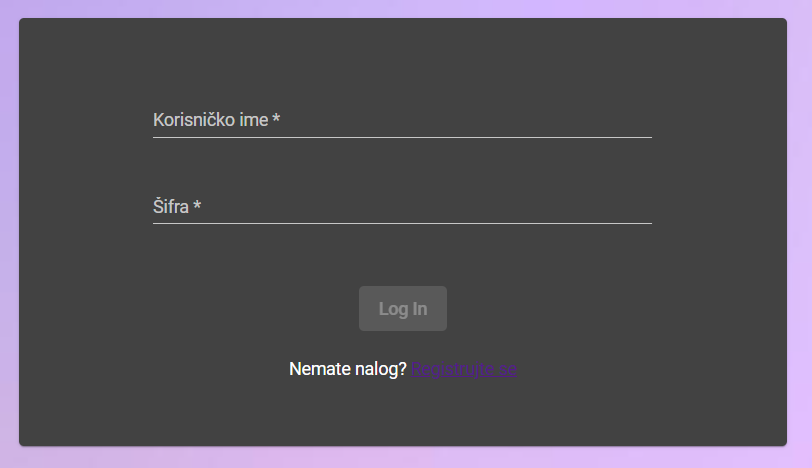
Слика 5.1. Почетна страна апликације



Поред почетног екрана, кориснику су доступна дугмад која воде ка страницама за регистрацију новог корисника која је приказа на слици 5.2. и ка страници за пријављивање на систем, која је приказана на слици 5.3. Након што корисник валидно попуни форму са слике 5.3 и кликне на дугме (*Registruj se*, слика 5.3) региструје се на систем и потом одласком на форму са слике 5.2 и уносом креденцијала које је унео на пређашњој форми пријављује се на систем и има могућност заказивања лекарских прегледа. Само се пацијенти могу регистровати путем форме са слике 5.3.



Слика 5.2. Форма за регистрацију новог корисника

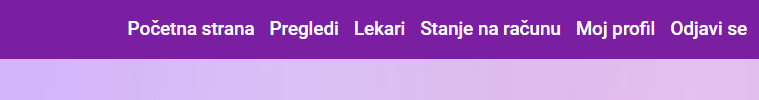


Слика 5.3. Форма за пријаву на систем

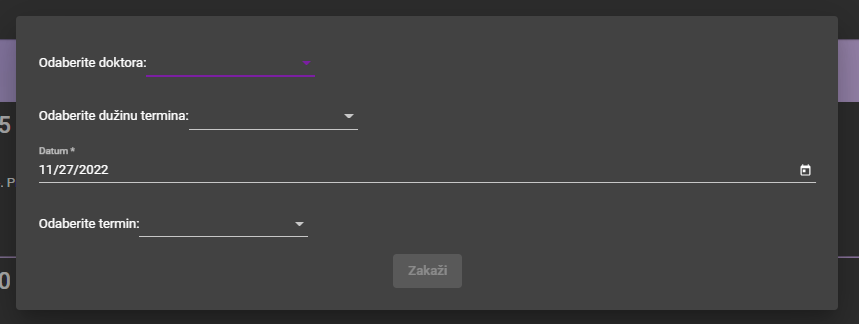
### 5.2 Пацијент

При успешној регистрацији пацијент се враћа на почетну страницу али му се појављују нове опције у заглављу странице. Пацијент има опцију прегледа својих прегледа са могућношћу заказивања нових, преглед свих лекара, преглед свој стања на рачуну и преглед свих профила. Опције у заглављу странице пријављеног пацијента приказано је на слици 5.4.

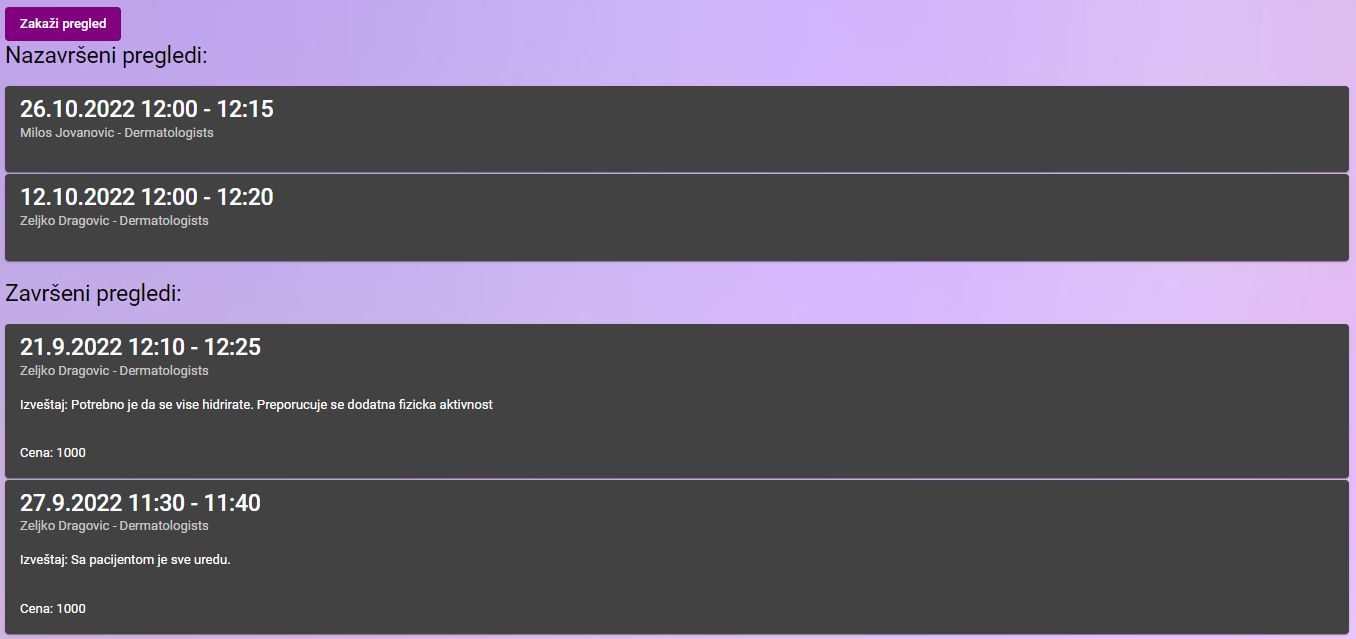
Слика 5.4. Заглавље пријављеног пацијента



На страници прегледа свих претходних и будућих прегледа пацијент има могућност заказивања новог прегледа путем форме приказане на слици 5.5. На слици 5.6. приказана је страница приказа свих будућих и претходних прегледа пацијента.



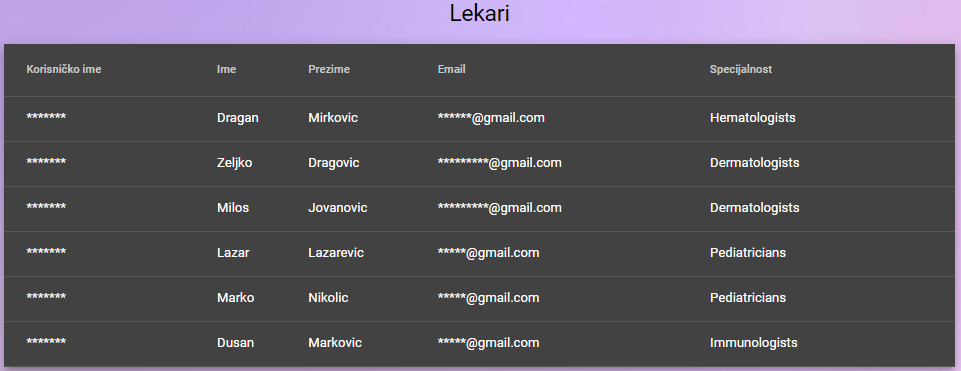
Слика 5.5. Заказивање новог прегледа



Слика 5.6. Преглед прошлих и будућих прегледа

Пацијент такође има приказ свих активних лекара у систему, што је приказано на слици 5.7. Над подацима из табеле приказане на слици 5.7 примењена је анонимизације маскирањем.

Код за маскирање података лекара приказан је на листингу 5.8. где је маскирање корисничког имена имплементирано путем замене сваког карактера “\*”, док је код анонимизације мејл адресе примењено парцијално маскирање где су сви карактери до “@” замењени “\*”, док је домен мејл адресе није анонимизован. Име, презиме и специјализација лекара се анонимизују јер представљају јавни податак установе.



Слика 5.7. Преглед свих лекара

Пацијент има преглед стања на свом тренутном рачуну, што је приказано на слици 5.9. као и преглед својих личних података са могућношћу да их промени. Форма за измену личних података пацијента приказана је на слици 5.10.

Листинг 5.8. Код за анонимизацију маскирањем свих доктора

public GetAllDoctorsResponse AnonymiseAllDoctors(GetAllDoctorsResponse allDoctors, bool forAdmin)

{

var response = new GetAllDoctorsResponse();

foreach (var doctor in allDoctors.Doctors)

{

response.Doctors.Add(AnonymiseDoctorResponse(doctor, forAdmin));

}

return response;

}

public DoctorResponse AnonymiseDoctorResponse(DoctorResponse doctor, bool forAdmin)

{

return new DoctorResponse

{

Id = doctor.Id,

Email = AnonymiseEmail(doctor.Email),

DateOfBirth = doctor.DateOfBirth,

FirstName = doctor.FirstName,

Surname = doctor.Surname,

PersonalNumber = AnonymiseString(doctor.PersonalNumber),

PhoneNumber = AnonymiseString(doctor.PhoneNumber),

Username = forAdmin ? doctor.Username : AnonymiseString(doctor.Username),

MedicalSpeciality = doctor.MedicalSpeciality,

IsActive = doctor.IsActive

};

}

private string AnonymiseString(string report)

{

return new Regex("\\S").Replace(report, "\*");

}

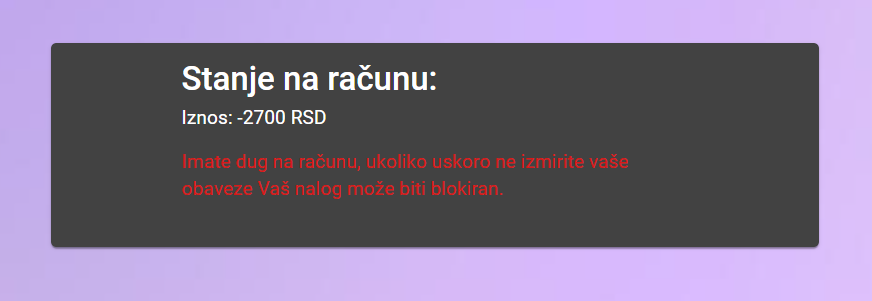
private string AnonymiseEmail(string email)

{

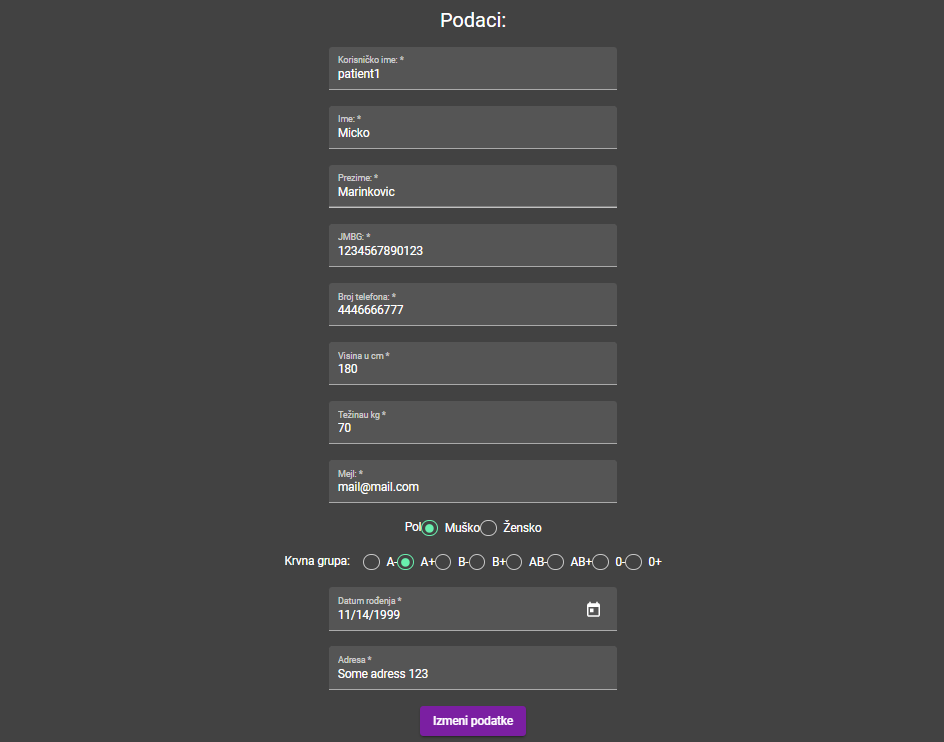
var domain = email.Split('@');

return $"{AnonymiseString(domain[0])}@{domain[1]}";

}



Слика 5.9. Преглед стања на рачуну пацијента

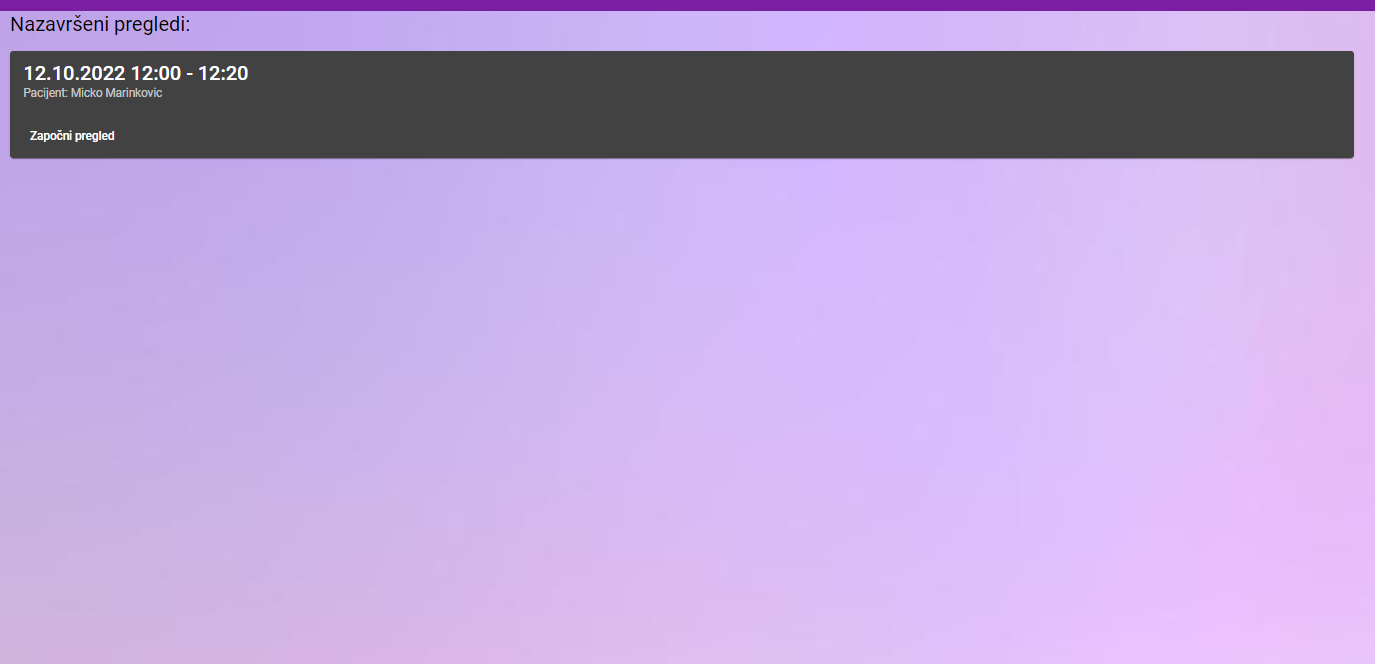


Слика 5.10. Форма за измену података пацијента

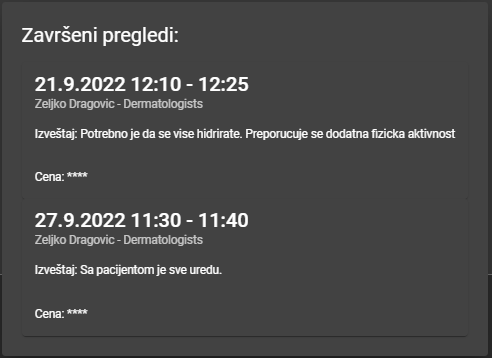
### 5.3 Лекар

Након успешног пријављивања лекар се враћа на почетну страну, где му се у заглављу стране омогућавају прелази на странице за преглед свих лекара, преглед свих будућих прегледа где може започети жељени преглед и преглед свог профила.

При прегледу будућих прегледа, слика 5.11, лекар има могућност да започне преглед. Када је преглед започет, лекар кликом на дугме *Započni pregled* са слике 5.11. отвара листу свих претходних прегледа пацијента који је тренутно на прегледу. При приказивању претходних прегледа примењена је метода маскирања над финансијским подацима везаних за преглед. Пример листе претходних прегледа који су приказани лекару налази се на слици 5.12.

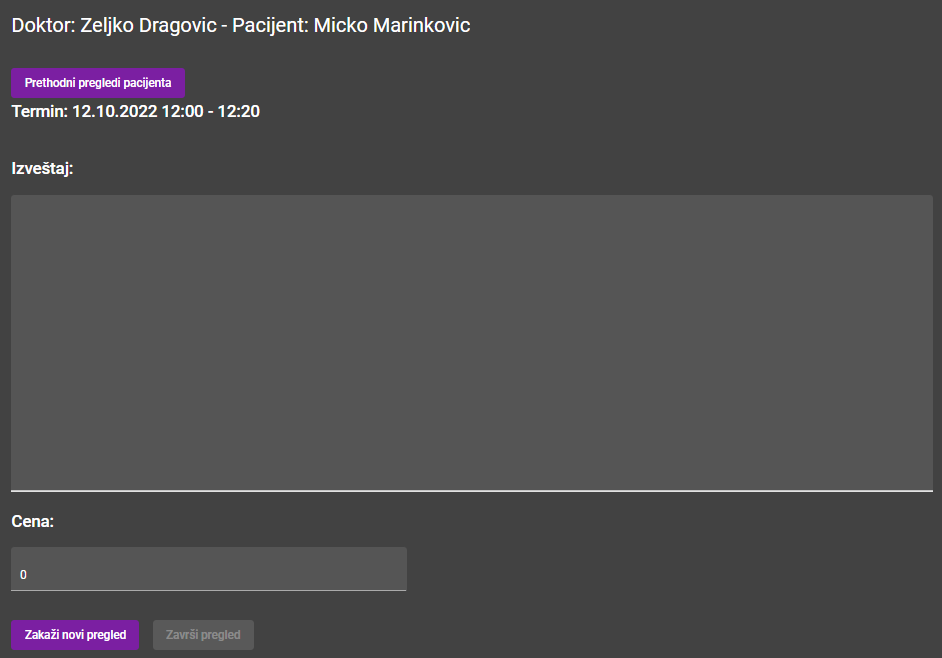


Слика 5.11. Будући прегледи лекара



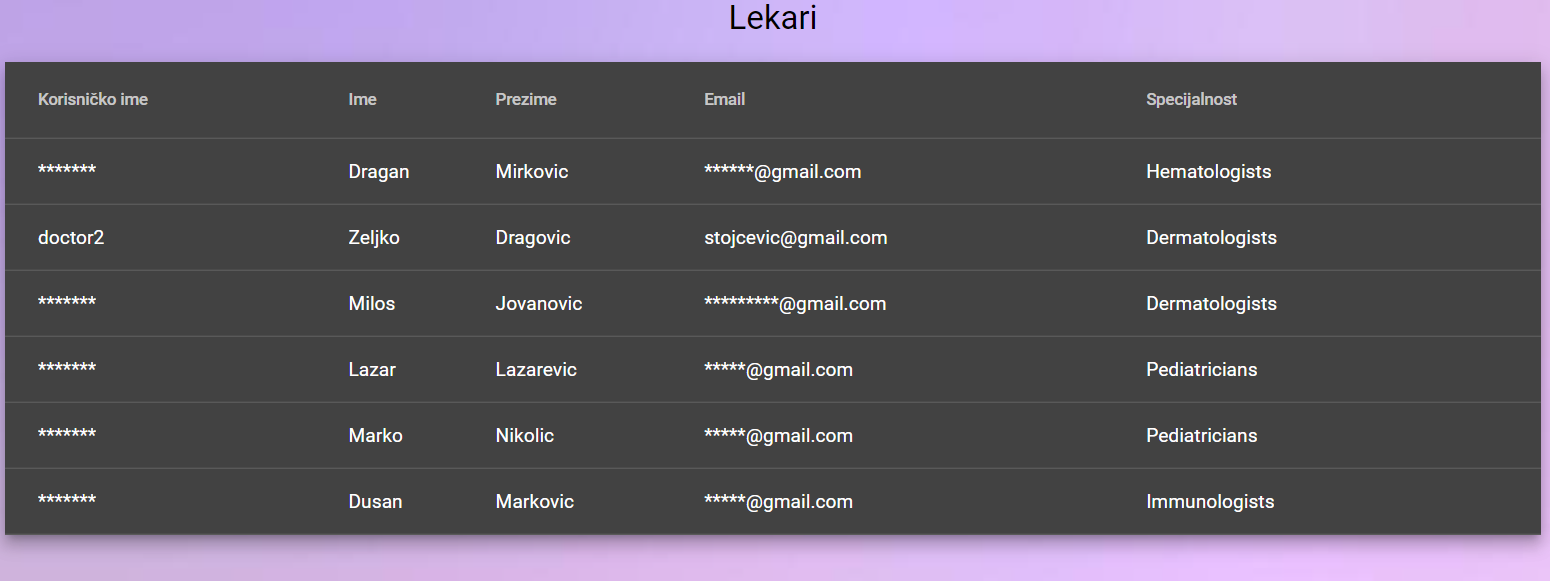
Слика 5.12. Претходни прегледи пацијента који је на прегледу

При обављању прегледа, лекар је дужан да унесе извештај како би могао да заврши преглед, такође није дозвољено унети цену прегледа мању од нуле. Страница прегледа приказана је на слици 5.13. Лекар такође има могућност да закаже нови преглед пацијенту које је на прегледу путем форме са слике 5.5. кликом на дугме *Zakaži novi pregled* са слике 5.13.



Слика 5.13. Форма за завршетак прегледа

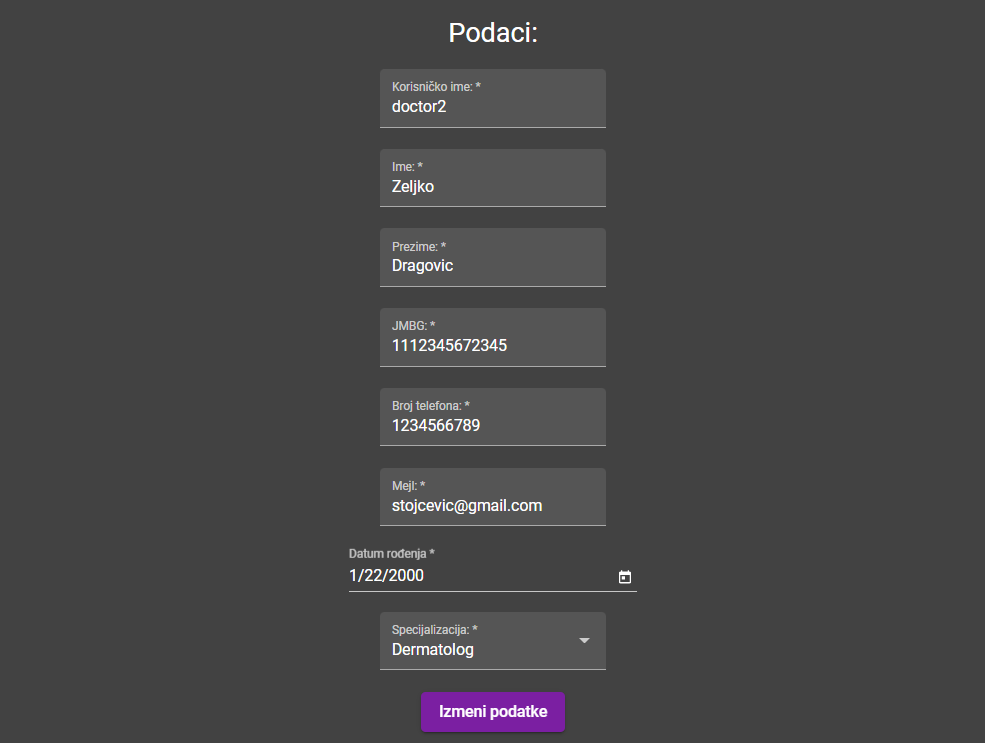
Лекар има могућност прегледа осталих лекара са тим да му се приказују анонимизовани подаци осталих лекара осим његових. Листа лекара приказана је на слици 5.14.



Слика 5.14. Анонимизовани лекари осим тренутно пријављеног

Анонимизација се обавља на идентичан начин као и при приказу свих лекара, листинг 5.8, пацијенту са изузетком тренутног корисника који се не анонимизује.

Лекар има могућност измене личних података путем форме приказане на слици 5.15.



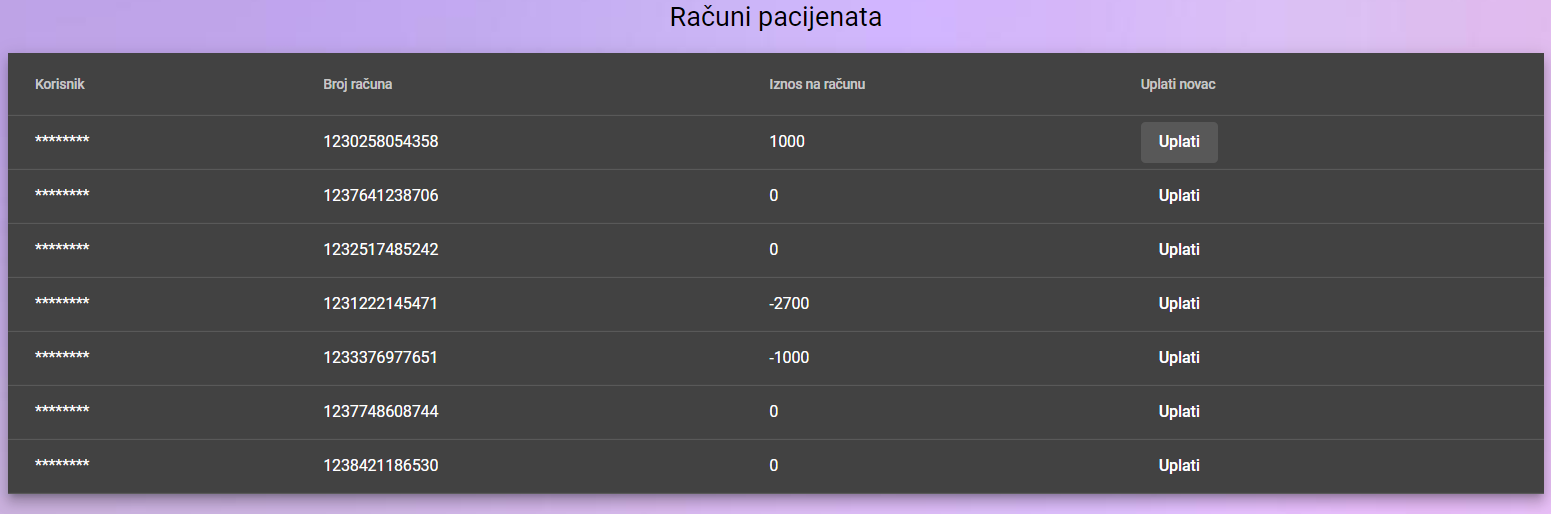
Слика 5.15. Форма измене личних података лекара

### 5.4 Рачуновођа

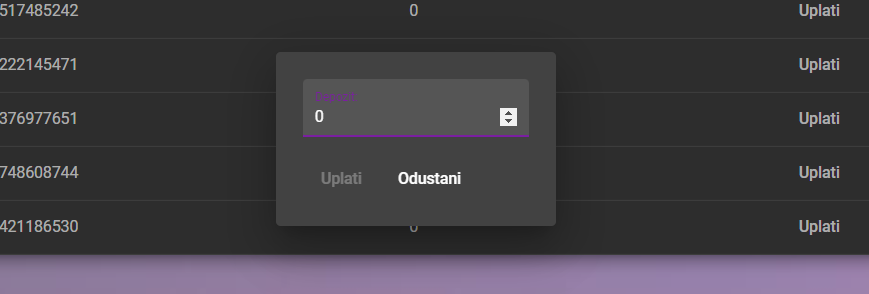
Након успешног пријављивања рачуновођа се враћа на почетну страну где се у заглављу појављују опције прегледа свих лекара, свих рачуна пацијената где има могућност уплате новца на њихов рачун и могућност измене личних података.

Рачуновођа има преглед свих лекара на исти начин као што то има и пацијент, што је приказано на слици 5.7.

Рачуновођа има на увид све рачуне пацијената и има могућност уплате новца на њих. Изглед табеле приказа свих рачуна приказан је на слици 5.16. док је модални дијалог за уплату новца на пацијентов рачун приказан на слици 5.17. који се отвара кликом на дугме *Uplati* у табели на слици 5.16. Да би рачуновођа успешно повећао износ на рачуну пацијента мора да унесе број већи од нуле.

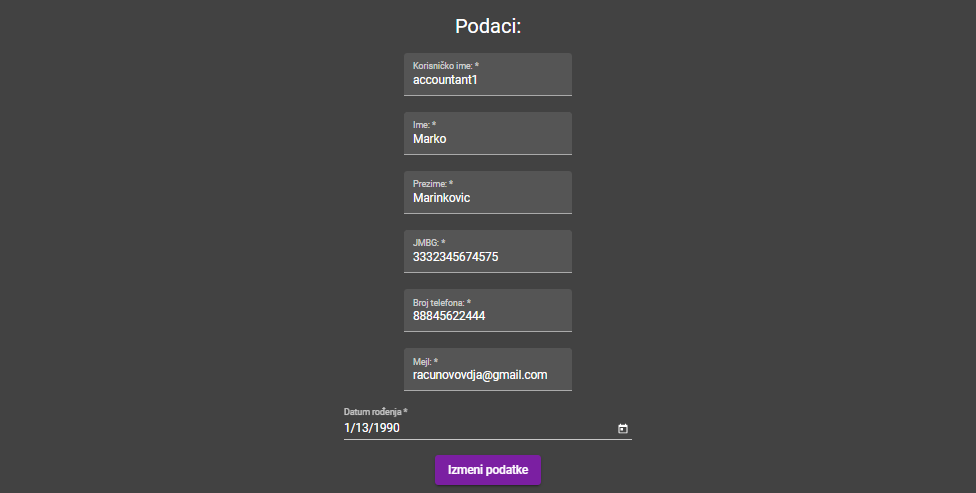


Слика 5.16. Приказ свих рачуна пацијената



Слика 5.17. Дијалог за уплату новца на рачун пацијента

Рачуновођа, као и сви други пријављени корисници система има могућност прегледа свог профила и измену личних података, што је приказано на слици 5.18.



Слика 5.18. Дијалог за измену личних података рачуновође

### 5.5 Администратор

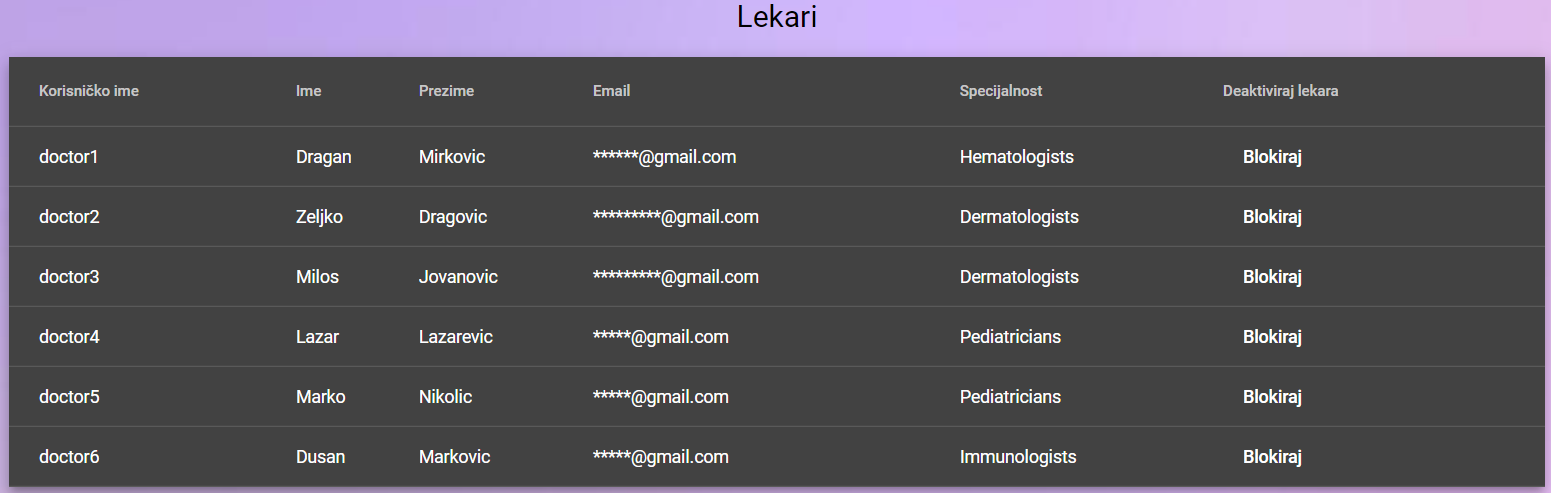
Након успешног пријављивања, администратор се пребацује на почетну страну, где му се на заглављу странице појављују нове опције, преглед свих корисника, преглед свих лекара, преглед рачуна и експорт базе података.

На страници прегледа свих пацијената, администратор има могућност да блокира жељеног пацијента. Након што је пацијент блокиран, он више не може да се улогује на апликацију. Поред блокирања, администратор може и да одблокира блокиране кориснике. Администратор на страници види само корисничко име пацијента док су остали подаци анонимизовани методом маскирања. Приказ табеле свих корисника коју на увид има администратор система приказана је на слици 5.19.



Слика 5.19. Приказ свих пацијената администратору

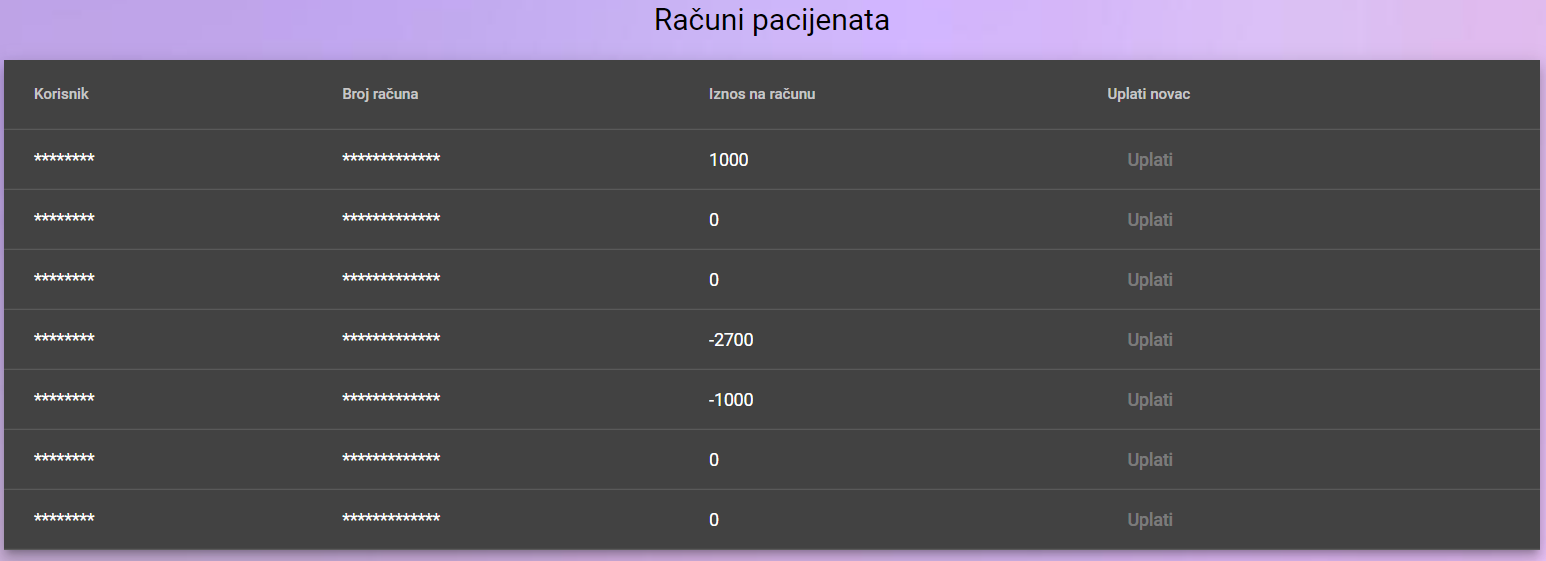
Страница прегледа свих лекара приказује табелу свих лекара са могућношћу да администратор блокира жељеног лекара. Блокирани лекар не може више да се пријави на систем. Администратор након блокирања може да одблокира блокираног лекара. Приказ табеле свих лекара коју види администратор може се видети на слици 5.20.



Слика 5.20. Приказ свих лекара администратору

Администратор такође има преглед стања на рачунима пацијената. Администратору су сви подаци анонимизовани маскирањем оним конкретног стања на рачуну. Табела свих рачуна пацијената приказана је на слици 5.21.

Код за анонимизацују рачуна приказан у листингу 5.20. на ком је приказана анонимизација маскирањем где се сви карактери корисничког имена пацијента коме припада рачун и карактери броја рачуна замењују са “\*”. Стање на рачуну се не маскира јер оно не представља податак који може да открије идентитет пацијента.



Листинг 5.20. Код за анонимизацију маскирањем рачуна пацијената

public AccountsResponse AnonymiseMultipleAccounts(AccountsResponse accounts)

{

var retVal = new AccountsResponse();

foreach (var account in accounts.Accounts)

{

retVal.Accounts.Add(new AccountResponse()

{

Id = account.Id,

AccountNumber = AnonymiseString(account.AccountNumber),

PatientId = account.PatientId,

Balance = account.Balance,

PatientUsername = AnonymiseString(account.PatientUsername ?? "")

});

}

return retVal;

}

private string AnonymiseString(string report)

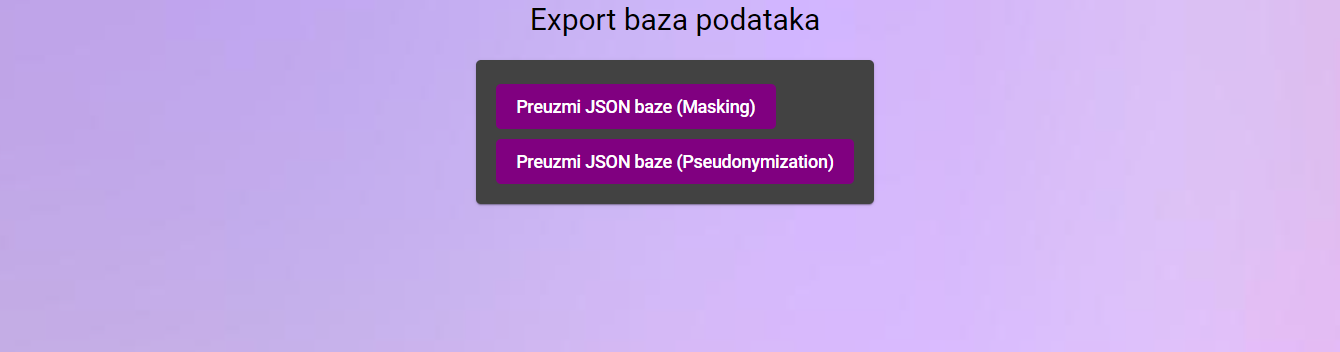
{

return new Regex("\\S").Replace(report, "\*");

}

Слика 5.21. Приказ свих рачуна администратору

Администратор има опцију преузимања анонимизоване целе базе података. Администратору се на избор даје да ли жели да анонимизација буде обављена методом маскирања или псеудонимизацијом. Након тога администратор преузима фајл у JSON формату. Изглед странице за одабир методе анонимизације приказан је на слици 5.22.



Слика 5.22. Избор методе анонимизације базе података

Код “endpoint-a” за анонимизацију маскирањем је приказан на листингу 5.22. где је приказано како се за сваку листу ентитета позива сервис за анонимизацију где се обавља анонимизације конкретне листе. У коду је такође приказана анотација ауторизације која не дозвољава да овој функционалности приступа било ко, ко није администратор.

Док је код конкретне анонимизације података приказан на листингу 5.23. где се приказује имплементација анонимизације маскирањем где се сви карактери осетљивих података замењују са “\*”. Изузетак од овог приступа представљају датуми, који се замењују насумичним датумом и мејл адреса где се анонимизује само део пре “@” карактера. “Id” вредности се не анонимизују, јер би се тако изгубиле везе између података.

[HttpGet("export"), Authorize(Roles = "ADMIN")]

public class ExportDatabaseWithMaskingEndpoint : Endpoint<EmptyRequest, ExportDatabaseResponse>

{

private readonly IDoctorService \_doctorService;

private readonly IAppointmentService \_appointmentService;

private readonly IAccountService \_accountService;

private readonly IPatientService \_patientService;

private readonly IAccountantService \_accountantService;

private readonly IAdminService \_adminService;

private readonly IAnonymizationService \_anonymizationService;

public ExportDatabaseWithMaskingEndpoint(

IAppointmentService appointmentService,

IAccountService accountService,

IDoctorService doctorService,

IPatientService patientService,

IAnonymizationService anonymizationService,

IAccountantService accountantService, IAdminService adminService)

{

\_appointmentService = appointmentService;

\_accountService = accountService;

\_doctorService = doctorService;

\_patientService = patientService;

\_anonymizationService = anonymizationService;

\_accountantService = accountantService;

\_adminService = adminService;

}

Листинг 5.22. Код имплементације “endpoint-a” за анонимизацију маскирањем целе базе података

public override async Task HandleAsync(EmptyRequest req, CancellationToken ct)

{

var patients = await \_patientService.GetAllAsync();

var doctors = await \_doctorService.GetAllAsync(false);

var accounts = await \_accountService.GetAllAsync();

var appointments = await \_appointmentService.GetAllAsync();

var accountants = await \_accountantService.GetAllAsync();

var admins = await \_adminService.GetAllAsync();

await SendOkAsync(new ExportDatabaseResponse

{

Patients = \_anonymizationService.AnonymisePatientsByMasking(patients)?.ToPatientsResponse() ?? new List<PatientResponse>(),

Doctors = \_anonymizationService.AnonymiseDoctorsByMasking(doctors)?.ToDoctorsResponse() ?? new List<DoctorResponse>(),

Accounts = \_anonymizationService.AnonymiseAccountsByMasking(accounts)?.ToAccountsResponse() ?? new List<AccountResponse>(),

Appointments = \_anonymizationService.AnonymiseAppointmentsByMasking(appointments)?.ToMultipleAppointmentResponse() ?? new List<AppointmentResponse>(),

Accountants = \_anonymizationService.AnonymiseAccountantsByMasking(accountants)?.ToAccountantsResponse() ?? new List<AccountantResponse>(),

Admins = \_anonymizationService.AnonymiseAdminsByMasking(admins)?.ToAdminsResponse() ?? new List<AdminResponse>(),

},

ct) ;

}

}

public ICollection<PatientDto>? AnonymisePatientsByMasking(ICollection<Patient>? patients)

{

var retVal = new List<PatientDto>();

retVal.AddRange(patients!.Select(patient => new PatientDto

{

Id = patient.Id.Value.ToString(),

Email = AnonymiseEmail(patient.Email.Value),

DateOfBirth = patient.DateOfBirth.Value.ToDateTime(TimeOnly.MinValue),

FirstName = AnonymiseString(patient.FirstName.Value),

Surname = AnonymiseString(patient.Surname.Value),

PersonalNumber = AnonymiseString(patient.PersonalNumber.Value),

PhoneNumber = AnonymiseString(patient.PhoneNumber.Value),

Password = "/",

Username = patient.Username.Value,

AccountId = patient.AccountId.Value.ToString(),

Adress = AnonymiseString(patient.Adress.Value),

BloodType = patient.BloodType.Value,

Gender = patient.Gender.Value,

Height = patient.Height.Value,

Weight = patient.Weight.Value,

IsActive = patient.IsActive

}));

return retVal;

}

public ICollection<DoctorDto>? AnonymiseDoctorsByMasking(ICollection<Doctor>? doctors)

{

var retVal = new List<DoctorDto>();

retVal.AddRange(doctors!.Select(doctor => new DoctorDto

{

Id = doctor.Id.Value.ToString(),

Email = AnonymiseEmail(doctor.Email.Value),

DateOfBirth = doctor.DateOfBirth.Value.ToDateTime(TimeOnly.MinValue),

FirstName = AnonymiseString(doctor.FirstName.Value),

Surname = AnonymiseString(doctor.Surname.Value),

MedicalSpeciality = doctor.MedicalSpeciality.Value.ToString(),

PersonalNumber = AnonymiseString(doctor.PersonalNumber.Value),

PhoneNumber = AnonymiseString(doctor.PhoneNumber.Value),

Password = "/",

Username = doctor.Username.Value,

IsActive = doctor.IsActive,

}));

return retVal;

}

public ICollection<AccountDto>? AnonymiseAccountsByMasking(ICollection<Account>? accounts)

{

var retVal = new List<AccountDto>();

retVal.AddRange(accounts!.Select(account => new AccountDto

{

Id = account.Id.Value.ToString(),

AccountNumber = AnonymiseString(account.AccountNumber.Value),

Balance = account.Balance.Value,

PatientId = account.PatientId.Value.ToString(),

}));

return retVal;

}

public ICollection<AccountantDto>? AnonymiseAccountantsByMasking(ICollection<Accountant>? accountants)

{

var retVal = new List<AccountantDto>();

retVal.AddRange(accountants!.Select(accountant => new AccountantDto

{

Id = accountant.Id.Value.ToString(),

Email = AnonymiseEmail(accountant.Email.Value),

DateOfBirth = accountant.DateOfBirth.Value.ToDateTime(TimeOnly.MinValue),

FirstName = AnonymiseString(accountant.FirstName.Value),

Surname = AnonymiseString(accountant.Surname.Value),

PersonalNumber = AnonymiseString(accountant.PersonalNumber.Value),

PhoneNumber = AnonymiseString(accountant.PhoneNumber.Value),

Password = "/",

Username = accountant.Username.Value

}));

return retVal;

}

public ICollection<AppointmentDto>? AnonymiseAppointmentsByMasking(ICollection<Appointment>? appointments)

{

var retVal = new List<AppointmentDto>();

retVal.AddRange(appointments!.Select(appointment => new AppointmentDto

{

Id = appointment.Id.Value.ToString(),

PatientId = appointment.PatientId.Value.ToString(),

DoctorId = appointment.DoctorId.Value.ToString(),

EndTime = appointment.EndTime.Value,

StartTime = appointment.StartTime.Value,

Price = appointment.Price?.Value ?? 0,

Report = AnonymiseString(appointment.Report?.Value ?? string.Empty),

}));

return retVal;

}

Листинг 5.23. Код имплементације анонимизације маскирањем

public ICollection<AdminDto>? AnonymiseAdminsByMasking(ICollection<Admin>? admins)

{

var retVal = new List<AdminDto>();

retVal.AddRange(admins!.Select(admin => new AdminDto

{

Id = admin.Id.Value.ToString(),

Email = AnonymiseEmail(admin.Email.Value),

DateOfBirth = admin.DateOfBirth.Value.ToDateTime(TimeOnly.MinValue),

FirstName = AnonymiseString(admin.FirstName.Value),

Surname = AnonymiseString(admin.Surname.Value),

PersonalNumber = AnonymiseString(admin.PersonalNumber.Value),

PhoneNumber = AnonymiseString(admin.PhoneNumber.Value),

Password = "/",

Username = admin.Username.Value

}));

return retVal;

}

Код “endpoint-a” за анонимизацију псеудонимизацијом је приказан на листингу 5.24. где је приказана анотација која обезбеђује да овој функционалности може да приступи само администратор и такође је приказано добављање свих података путом одговарајућих сервиса да би се потом ти подаци проследили сервису за анонимизацију где се обавља анонимизација података.

Код конкретне анонимизације података приказан на листингу 5.25. Анонимизација је обављена методом псеудонимизације где се оригиналне вредности замењују насумичним вредностима. За генерисање насумичне вредности лекарских извештаја, мејл адреса и датума коришћена је библиотека Lorem. За насумични одабир имена и презимена коришћен је генератор насумичних бројева за одабир одговарајућег имена или презимена из предефинисане листе имена и презимена.

[HttpGet("export-pseudonymization"), Authorize(Roles = "ADMIN")]

public class ExportDatabaseWithPseudonymizationEndpoint : Endpoint<EmptyRequest, ExportDatabaseResponse>

{

private readonly IDoctorService \_doctorService;

private readonly IAppointmentService \_appointmentService;

private readonly IAccountService \_accountService;

private readonly IPatientService \_patientService;

private readonly IAccountantService \_accountantService;

private readonly IAdminService \_adminService;

private readonly IAnonymizationService \_anonymizationService;

public ExportDatabaseWithPseudonymizationEndpoint(

IAppointmentService appointmentService,

IAccountService accountService,

IDoctorService doctorService,

IPatientService patientService,

IAnonymizationService anonymizationService,

IAccountantService accountantService, IAdminService adminService)

{

\_appointmentService = appointmentService;

\_accountService = accountService;

\_doctorService = doctorService;

\_patientService = patientService;

\_anonymizationService = anonymizationService;

\_accountantService = accountantService;

\_adminService = adminService;

}

public override async Task HandleAsync(EmptyRequest req, CancellationToken ct)

{

var patients = await \_patientService.GetAllAsync();

var doctors = await \_doctorService.GetAllAsync(false);

var accounts = await \_accountService.GetAllAsync();

var appointments = await \_appointmentService.GetAllAsync();

var accountants = await \_accountantService.GetAllAsync();

var admins = await \_adminService.GetAllAsync();

await SendOkAsync(new ExportDatabaseResponse

{

public ICollection<AccountDto>? AnonymiseAccountsByPseudonymization(ICollection<Account>? accounts)

{

var retVal = new List<AccountDto>();

retVal.AddRange(accounts!.Select(account => new AccountDto

{

Id = account.Id.Value.ToString(),

AccountNumber = $"123{gen.Next(1000, 9999)}{gen.Next(100, 999)}{gen.Next(100, 999)}",

Balance = account.Balance.Value,

PatientId = account.PatientId.Value.ToString(),

}));

return retVal;

}

Листинг 5.24. Код имплементације “endpoint-a” за анонимизацију преудонимизацијом целе базе података

Patients = \_anonymizationService.AnonymisePatientsByPseudonymization(patients)?.ToPatientsResponse() ?? new List<PatientResponse>(),

Doctors = \_anonymizationService.AnonymiseDoctorsByPseudonymization(doctors)?.ToDoctorsResponse() ?? new List<DoctorResponse>(),

Accounts = \_anonymizationService.AnonymiseAccountsByPseudonymization(accounts)?.ToAccountsResponse() ?? new List<AccountResponse>(),

Appointments = \_anonymizationService.AnonymiseAppointmentsByPseudonymization(appointments)?.ToMultipleAppointmentResponse() ?? new List<AppointmentResponse>(),

Accountants = \_anonymizationService.AnonymiseAccountantsByPseudonymization(accountants)?.ToAccountantsResponse() ?? new List<AccountantResponse>(),

Admins = \_anonymizationService.AnonymiseAdminsByPseudonymization(admins)?.ToAdminsResponse() ?? new List<AdminResponse>(),

},

ct) ;

}

}

public ICollection<AppointmentDto>? AnonymiseAppointmentsByPseudonymization(ICollection<Appointment>? appointments)

{

var retVal = new List<AppointmentDto>();

retVal.AddRange(appointments!.Select(appointment => new AppointmentDto

{

Id = appointment.Id.Value.ToString(),

PatientId = appointment.PatientId.Value.ToString(),

DoctorId = appointment.DoctorId.Value.ToString(),

EndTime = appointment.EndTime.Value,

StartTime = appointment.StartTime.Value,

Price = appointment.Price?.Value ?? 0,

Report = appointment.Report is null ? "" : Lorem.Paragraph(5, 12, 2, 10),

}));

return retVal;

}

public ICollection<DoctorDto>? AnonymiseDoctorsByPseudonymization(ICollection<Doctor>? doctors)

{

var retVal = new List<DoctorDto>();

retVal.AddRange(doctors!.Select(doctor => new DoctorDto

{

Id = doctor.Id.Value.ToString(),

Email = Lorem.Email(),

DateOfBirth = Lorem.DateTime(new DateTime(1950, 1, 1), new DateTime(2000, 1, 1)),

FirstName = GetRandomFirstName(),

Surname = GetRadnomSurname(),

MedicalSpeciality = doctor.MedicalSpeciality.Value.ToString(),

PersonalNumber = GetRandomPersonalNumber(),

PhoneNumber = GetRandomPhoneNumber(),

Password = "password",

Username = GetRandomUsername(),

IsActive = doctor.IsActive,

})); ;

return retVal;

}

public ICollection<PatientDto>? AnonymisePatientsByPseudonymization(ICollection<Patient>? patients)

{

var retVal = new List<PatientDto>();

retVal.AddRange(patients!.Select(patient => new PatientDto

{

Id = patient.Id.Value.ToString(),

Email = Lorem.Email(),

DateOfBirth = Lorem.DateTime(new DateTime(1950, 1, 1), new DateTime(2020, 1, 1)),

FirstName = patient.Gender.Value.Equals("Male") ? GetRandomMaleFirstName() : GetRadnomFemaleFirstName(),

Surname = GetRadnomSurname(),

PersonalNumber = GetRandomPersonalNumber(),

PhoneNumber = GetRandomPhoneNumber(),

Password = "password",

Username = GetRandomUsername(),

AccountId = patient.AccountId.Value.ToString(),

Adress = GetRandomAddress(),

BloodType = patient.BloodType.Value,

Gender = patient.Gender.Value,

Height = patient.Height.Value,

Weight = patient.Weight.Value,

IsActive = patient.IsActive

}));

return retVal;

}

public ICollection<AccountantDto>? AnonymiseAccountantsByPseudonymization(ICollection<Accountant>? accountants)

{

var retVal = new List<AccountantDto>();

retVal.AddRange(accountants!.Select(accountant => new AccountantDto

{

Id = accountant.Id.Value.ToString(),

Email = Lorem.Email(),

DateOfBirth = Lorem.DateTime(new DateTime(1950, 1, 1), new DateTime(2000, 1, 1)),

FirstName = GetRandomFirstName(),

Surname = GetRadnomSurname(),

PersonalNumber = GetRandomPersonalNumber(),

PhoneNumber = GetRandomPhoneNumber(),

Password = "password",

Username = GetRandomUsername()

}));

return retVal;

}

Листинг 5.25. Код имплементације анонимизације псеудонимизацијом

public ICollection<AdminDto>? AnonymiseAdminsByPseudonymization(ICollection<Admin>? admins)

{

var retVal = new List<AdminDto>();

retVal.AddRange(admins!.Select(admin => new AdminDto

{

Id = admin.Id.Value.ToString(),

Email = Lorem.Email(),

DateOfBirth = Lorem.DateTime(new DateTime(1950, 1, 1), new DateTime(2000, 1, 1)),

FirstName = GetRandomFirstName(),

Surname = GetRadnomSurname(),

PersonalNumber = GetRandomPersonalNumber(),

PhoneNumber = GetRandomPhoneNumber(),

Password = "password",

Username = GetRandomUsername(),

}));

return retVal;

}

Помоћне методе које су коришћене при псеудонимизацији су приказани у листингу 5.26. За прецизнију расподелу насумичних имена као генератор насумичне вредности коришћен је генератор насумичне вредности из библиотеке Lorem. Приказане су метода за генерисање насумичног имена, презимена, ЈМБГ-а, броја телефона и адресе.

private string GetRandomFirstName()

{

if (Lorem.Chance(50, 100))

{

return Constants.MaleNames.ElementAt((int)Lorem.Number(0, Constants.MaleNames.Count - 1));

}

else

{

return Constants.FemaleNames.ElementAt((int)Lorem.Number(0, Constants.FemaleNames.Count - 1));

}

}

private string GetRandomMaleFirstName()

{

return Constants.MaleNames.ElementAt((int)Lorem.Number(0, Constants.MaleNames.Count - 1));

}

private string GetRandomFemaleFirstName()

{

return Constants.FemaleNames.ElementAt((int)Lorem.Number(0, Constants.FemaleNames.Count - 1));

}

Листинг 5.26. Код помоћних метода псеудонимизације

private string GetRandomSurname()

{

return Constants.Surnames.ElementAt((int)Lorem.Number(0, Constants.Surnames.Count - 1));

}

private string GetRandomPhoneNumber()

{

return $"+3816{ Lorem.Number(100, 900)}{ Lorem.Number(10, 99)}{ Lorem.Number(100, 999)}";

}

private string GetRandomPersonalNumber()

{

return $"{ Lorem.Number(100, 999)}{ Lorem.Number(100, 999)}{ Lorem.Number(100, 999)}{ Lorem.Number(100, 999)}";

}

private string GetRandomUsername()

{

return $"{Lorem.Words(1, false)}{Lorem.Number(1, 100)}";

}

private string GetRandomAddress()

{

return $"{Constants.Towns.ElementAt((int)Lorem.Number(0, Constants.Towns.Count - 1))}," +

$" {Constants.Streets.ElementAt((int)Lorem.Number(0, Constants.Streets.Count - 1))} {Lorem.Number(0, 50)}";

}

private string GetRandomFemaleFirstName()

{

return Constants.FemaleNames.ElementAt((int)Lorem.Number(0, Constants.FemaleNames.Count - 1));

}

# 6. ЗАКЉУЧАК

У овом раду описана је реализација апликације за заказивање лекарских прегледа са нагласком на сервис са анонимизацију података унутар ње, који омогућава адекватну анонимизацију података система. Апликација треба да обезбеди сигурно екстраховање података како би се ти подаци касније могли користи у тестирању или обучавању будућих корисника система. Подаци ни на који начин не смеју да одају идентитет корисника са којим су оригинално повезани. То представља одговоран задатак тима који развија апликацију јер уколико би дошло до цурења осетљивих података компанија која развија софтвер се може суочити за законским последицама.

Описано је шта је то анонимизације и на које начине се она може обавити, са примерима како подаци изгледају након сваког типа анонимизације, као и када је који начин анонимизације погодан. Описано је шта подаци треба да задовољавају како би се могли сматрати анонимизованим.

На крају је приказана конкретна имплементација система. Описане су улоге које корисници могу да обављају у систему, акције које могу да извршавају као и подаци који им се приказује, анонимизовани или не. Приказан је програмски код имплементације сервиса за анонимизацију података.

У даљем развоју апликације би се могле имплементирати још неке од метода анонимизације. Унапређења невезана за анонимизацију би могла бити увођење програмског регистровања нових лекара и рачуновођа као и повећање опште сигурности апликације коришћењем HTTPS протокола за комуникацију корисничке апликације и серверске стране уместо тренутног HTTP протокола.

# ЛИТЕРАТУРА

[1] Amazon Documentation “What is Amazon DynamoDB?”, (Amazon DynamoDB)

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Introduction.html>

[2] JSON (JavaScript Object Notation),

<https://www.json.org/json-en.html>

[3] What is Data Anonymization?

<https://www.imperva.com/learn/data-security/anonymization/>

[4] Data anonymization: Use cases and 6 Common Techniques

<https://satoricyber.com/data-masking/data-anonymization-use-cases-and-6-common-techniques/>

[5] What is Data masking?

<https://www.imperva.com/learn/data-security/data-masking/>

[6] What is Data masking? | TIBCO

<https://www.tibco.com/reference-center/what-is-data-masking>

[7] Pseudonymization

<https://www.imperva.com/learn/data-security/pseudonymization/>

[8] What is pseudonymization?

<https://www.cloudflare.com/learning/privacy/what-is-pseudonymization/>

[9] 6 personal data anonymization techniques

<https://blog.pangeanic.com/6-personal-data-anonymization-techniques>

[10] Minimizing application privacy risk

<https://developer.ibm.com/articles/s-gdpr3/>

[11] Data anonymization: What, why, and how is it done?

<https://www.sidechannel.blog/en/data-anonymization-what-why-and-how-is-it-done/>

[12] Synthetic Data: Applications in Data Privacy and Machine Learning

<https://towardsdatascience.com/synthetic-data-applications-in-data-privacy-and-machine-learning-1078bb5dc1a7>

[13] Cloud Computing Services | Amazon Web Services

<https://aws.amazon.com/>

[14] AWS | Amazon SimpleDB

<https://aws.amazon.com/simpledb/>

[15] Amazon DynamoDB

<https://aws.amazon.com/dynamodb/>

[16] What is NoSQL?

<https://www.mongodb.com/nosql-explained>

[17] Advantages of NoSQL

<https://www.mongodb.com/nosql-explained/advantages>

[18] DynamoDB Local for Desktop Development

<https://aws.amazon.com/blogs/aws/dynamodb-local-for-desktop-development/>

[19] What is Amazon DynamoDB?

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Introduction.html>

[20] NoSQL Database

<https://logicmojo.com/nosql-database>

[21] Eventual-consistency

<https://www.scylladb.com/glossary/eventual-consistency/>

[22] Monolithic architecture

<https://www.techtarget.com/whatis/definition/monolithic-architecture>

[23] What is HTTP?

<https://www.cloudflare.com/learning/ddos/glossary/hypertext-transfer-protocol-http/>

[24] Angular

<https://angular.io/>

[25] Angular Material

<https://material.angular.io/>

[26] What is .NET 6?

<https://learn.microsoft.com/en-us/dotnet/core/whats-new/dotnet-6>

[27] RxJS

https://rxjs.dev/

[28] FastEndpoints - Documentation

<https://fast-endpoints.com/docs/get-started#create-project-install-package>

[29] REPR Design Pattern

<https://deviq.com/design-patterns/repr-design-pattern>

[30] Lorem.NET

<https://github.com/dochoffiday/Lorem.NET>

[31] Data Anonymization: The What, Why, and How of Data Anonymization

[https://usercentrics.com/knowledge-hub/data-anonymization/](https://usercentrics.com/knowledge-hub/data-anonymization/%20)

[32] Numerical data: Types and Characteristics

<https://www.voxco.com/blog/numerical-data-types-and-characteristics>

[33] Categorical Data

<http://www.stat.yale.edu/Courses/1997-98/101/catdat.htm>

[34] Getting started with DynamoDB

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GettingStartedDynamoDB.html>

# БИОГРАФИЈА

Лазар Стојчевић рођен је 13. јануара 2000. године у Новом Саду. Основну школу “Бора Станковић” у Каравукову завршио је 2014. након чега уписује Средњу техничку школу у Сомбору коју завршава 2018. Исте године уписује се на Факултет техничких наука, смер Рачунарство и аутоматика. Школске 2020/21. се опредељује за усмерење Примењене рачунарске науке и информатика, након чега се школске 2021/22. опредељује за модул Интернет и електронско пословање. Положио је све испите предвиђене планом и програмом.

# КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

|  |  |
| --- | --- |
| Редни број, **РБР**: |  |
| Идентификациони број, **ИБР**: |  |
| Тип документације, **ТД**: | монографска публикација |
| Тип записа, **ТЗ**: | текстуални штампани документ |
| Врста рада, **ВР**: | дипломски рад |
| Аутор, **АУ**: | Лазар Стојчевић |
| Ментор, **МН**: | проф. др Горан Сладић, ФТН Нови Сад |
| Наслов рада, **НР**: | Систем за анонимизацију података из AWS DynamoDB базе података |
| Језик публикације,**ЈП**: | Српски |
| Језик извода, **ЈИ**: | српски / енглески |
| Земља публиковања, **ЗП**: | Србија |
| Уже географско подручје, **УГП**: | Војводина |
| Година, **ГО**: | 2023 |
| Издавач, **ИЗ**: | ауторски репринт |
| Место и адреса, **МА**: | Нови Сад, Факултет техничких наука,  Трг Доситеја Обрадовића 6 |
| Физички опис рада, **ФО**: | бр. поглавља 6/ страница 55/ цитата / табела / слика 32/ графикона / прилога / листинга 7 |
| Научна област, **НО**: | Електротехничко и рачунарско инжењерство |
| Научна дисциплина, **НД**: | Безбедност у системима електронског пословања |
| Предметна одредница/кључне речи, **ПО**: | анонимизација, псеудонимизација, база података |
| **УДК** |  |
| Чува се у, **ЧУ**: | Библиотека Факултета техничких наука, Трг Доситеја Обрадовића 6, Нови Сад |
| Важна напомена, **ВН**: |  |
| Извод, **ИЗ**: | У раду је описано шта је то анонимизација података, начини на које се она обавља и зашто је важна. Примери анонимизације приказани су кроз апликацију за заказивање лекарских прегледа. |
| Датум прихватања теме, **ДП**: |  |
| Датум одбране, **ДО**: |  |
| Чланови комисије, **КО**: |  |
| председник | др Бранко Милосављевић, ред. проф, ФТН Нови Сад |
| члан | др Мирослав Зарић, ванр. проф, ФТН Нови Сад |
| ментор | др Горан Сладић, ред. проф, ФТН Нови Сад |
| Потпис ментора | |

# KEY WORDS DOCUMENTATION

|  |  |
| --- | --- |
| Accession number, **ANO**: |  |
| Identification number, **INO**: |  |
| Document type, **DT**: | monographic publication |
| Type of record, **TR**: | textual material |
| Contents code, **CC**: | BSc thesis |
| Author, **AU**: | Lazar Stojčević |
| Mentor, **MN**: | Goran Sladić, PhD, full. prof., FTN Novi Sad |
| Title, **TI**: | Application of AWS IAM and Cognito services for access control |
| Language of text, **LT**: | Serbian |
| Language of abstract, **LA**: | serbian / english |
| Country of publication, **CP**: | Serbia |
| Locality of publication, **LP**: | Vojvodina |
| Publication year, **PY**: | 2023 |
| Publisher, **PB**: | author’s reprint |
| Publication place, **PP**: | Novi Sad, Faculty of Technical Sciences, Square Dositej Obradović 6 |
| Physical description, **PD**: | no. of chapters 6/ pages 55/ quotes / tables / pictures 32/ graphs / appendix / listings 7 |
| Scientific field, **SF**: | Electrical and computer engineering |
| Scientific discipline, **ND**: | Security in e-business systems |
| Subject / Keywords, **S/KW**: | authentification, pseudonymisation, database |
| **UDC** |  |
| Holding data, **HD**: | Library of the Faculty of Technical Sciences, Trg Dositeja Obradovića 6, Novi Sad |
| Note, **N**: |  |
| Abstract, **AB**: | The article describes what data anonymization is, the ways it is performed, and why it is important. Examples of anonymization are shown through the application of ordering medical reviews. |
| Accepted by sci. board on, **ASB**: |  |
| Defended on, **DE**: |  |
| Defense board, **DB**: |  |
| president | Branko MIlosavljević, PhD, full. prof, FTN Novi Sad |
| member | Miroslav Zarić, PhD, assoc. prof, FTN Novi Sad |
| mentor | Goran Sladić, PhD, full. prof, FTN Novi Sad |
| Mentor’s signature | |