

РАЧУНАРСКА ГИМНАЗИЈА

МАТУРСКИ РАД

из напредних техника програмирања

ВЕБ АПЛИКАЦИЈА ЗА БОЛНИЦУ

Ученик:

Алекса Прокић, IV-1

Ментори:

др Филип Марић

Душа Вуковић

Београд, мај 2020.

# Садржај

## Садржај

1. Увод.....	- 1 -
2. Кориснички интерфејс и функционалност .....	- 2 -
2.1 Логовање .....	- 2 -
2.2 Главна страна.....	- 2 -
2.3 Налаз и дијагноза .....	- 3 -
2.4 Заказани прегледи .....	- 5 -
2.5 Пацијенти.....	- 5 -
3. База података .....	- 7 -
3.1 Пројектовање базе .....	- 7 -
3.2 Логички модел базе .....	- 8 -
4. Коришћене технологије .....	- 9 -
4.1 <i>PhpMyAdmin</i> .....	- 9 -
4.2 <i>MariaDB</i> .....	- 9 -
4.3 <i>XAMPP</i> .....	- 10 -
5. Програмски код апликације.....	- 10 -
5.1 Регистрација лекара у систем.....	- 10 -
5.2 Код странице пацијенти.....	- 13 -
5.3 Код странице налаз/дијагноза .....	- 19 -
6. Закључак.....	- 22 -
7. Литература.....	- 23 -

# 1. Увод

Веб апликације су програми, који корисницима пружају могућност да шаљу или примају податке преко Интернета користећи веб претраживаче. Чине их група или више група повезаних фајлова који се чувају на интернет серверима.

Веб апликације није потребно инсталирати, доступне су у било ком делу света и могуће им је приступити било кад у току дана. Централизована структура их чини безбедним и погодним за прављење резервних копија.

У данашњем свету веб програмирања, веб апликације су веома популарне и тражене. Велике и успешне компаније данас је немогуће замислити без веб апликација, али исто тако и државне институције, банке, школе.

У овом матурском раду биће речи о организацији, имплементацији, као и изради веб апликације која је намењена за рад у болницама, домовима здравља и осталим медицинским установама.

Ова апликација је фокусирана да испуни више циљева, а основни је да олакша лекарима рад при прегледању пацијената.

Медицинске установе се развијају и имају све већи број пацијената. Старомодан начин вођења евиденције о тим пацијентима помоћу папира и фасцикала, јако је спор и неефикасан. Такође, идеја ове апликације је: дигитална евиденција о лекарским извештајима, подацима пацијената и запослених лекара, као и евиденција заказаних прегледа. То је брз и поуздан начин о бележењу информација.

## 2. Кориснички интерфејс и функционалност

### 2.1 Логовање

Сваки лекар има своју лозинку и корисничко име помоћу којег региструје у систем. Апликација је намењена лекарима и сваки лекар при запослењу добија корисничко име и лозинку.

The login form is titled "Login" and is set against a dark blue background. It features a light blue rectangular area containing two input fields and a button. The first input field is preceded by a green icon of a person and is labeled "Korisničko ime". The second input field is preceded by a green icon of a padlock and is labeled "Lozinka". Below these fields is a blue button labeled "Login". Red arrows point from text labels to the corresponding UI elements: "Поље за унос корисничког имена" to the first input field, "Поље за унос лозинке" to the second input field, and "Дугме за регистрацију" to the "Login" button.

### 2.2 Главна страна

Након регистрације у систем доктору се приказује главна страница. На њој лекар може видети своје опште податке као што су име, презиме, корисничко име, датум рођења, број телефона и датум запослења. Такође, поред општих података, лекару се приказују статистички подаци везани за рад (број пацијената, прегледа и заказаних прегледа). Поред података о лекару на овој страници се налазе линкови ка осталим страницама.

The main page has a dark blue header with three navigation links: "Nalaz /dijagnoza", "Zakazani pregledi", and "Pacijenti". A red arrow points from the text "Линкови ка осталим страница" to the "Pacijenti" link. Below the header, the page is titled "Doktore Aleksa Prokić ,dobrodošli!". On the left, there are two red-bordered boxes. The first box, titled "Opšti podaci lekara:", contains the following information: "Ime: Aleksa", "Prezime: Prokić", "Broj telefona: 0665453434", "Datum rođenja: 2019-09-12", and "Datum zaposlenja: 2020-01-08". A red arrow points from the text "Општи подаци лекара" to this box. The second box, titled "Statistički podaci:", contains: "Ukupan broj pacijenata: 7", "Ukupan broj održanih pregleda: 10", and "Broj zakazanih pregleda: 5". A red arrow points from the text "Статистички подаци регистрованог лекара" to this box. On the right side of the page is a large blue caduceus symbol.

## 2.3 Налаз и дијагноза

На овој страници постоје две картице (енг. *Tabs*). Картица *podaci pacijenta* везана је за пацијента, док је картица *nalaz i dijagnoza* везана за преглед. Уколико пацијент до сада није био у болници, доктор га на основу броја здравствене књижице уноси у систем, док на картици *nalaz i dijagnoza* на основу истог броја за пацијента попуњава лекарски извештај. При уношењу пацијента у систем поред броја књижице наводе се и име, презиме, контакт телефон, као и датум рођења пацијента и то у формату месец, дан, година.

**Пример целокупне картице *podaci pacijenta*:**

The screenshot shows a web form for entering patient data. At the top, there are two tabs: 'podaci pacijenta' (highlighted with a red box) and 'nalaz i dijagnoza'. Below the tabs, the form contains several input fields: 'Ime:' (with a red box around the input), 'Prezime:' (with a red box around the input), 'Broj zdravstvene knjižice:' (with a red box around the input), 'Datum rođenja:' (with a date format 'mm/dd/yyyy'), and 'Kontakt telefon:' (with a red box around the input). At the bottom, there is a 'Sačuvaj pacijenta' button (highlighted with a red box). Red arrows point from text annotations to these elements: 'Табови за унос пацијената и дијагноза' points to the tabs; 'Пример поља за унос личног податка пацијента' points to the 'Ime' field; 'Поље за унос броја здравствене књижице' points to the 'Broj zdravstvene knjižice' field; and 'На клику дугмета се пацијент уписује у базу' points to the 'Sačuvaj pacijenta' button.

На картици *nalaz i dijagnoza* потребно је унети број здравствене књижице пацијента, а затим се попуњава налаз. Лекарски извештај састоји се од следећих ставки, а то су:

- Врста прегледа
- Анамнеза
- Лабораторијске анализе
- Фиизикални налаз
- Дијагноза
- Терапија
- *Imaging* процедуре
- Лекови
- Опција за заказивање прегледа

The screenshot shows a dropdown menu for selecting a medical procedure. The dropdown is open, showing a search bar with the text 'Претрага по називу' (highlighted with a red box). Below the search bar, there is a list of procedures: 'rendgen', 'СТ', 'ultrazvuk', and 'magnetna rezonanca'. The 'rendgen' and 'ultrazvuk' items have checkmarks next to them. Below the list, there is a text annotation: 'Пример поља вишеструког избора'.

Поља за унос текста као што су: анамнеза, дијагноза, физикални налаз и терапија, лекар попуњава самостално. Лекови, лабораторијске анализе и *imaging* процедуре су поља вишеструког избора, где лекар одабиром ставки која та поља нуде, пацијенту преписује одређене лекове и налаже да обави одабране лабораторијске анализе или *imaging* процедуре. Пример поља вишеструког избора приказан је на слици изнад.

Последњи тип поља на овој страници је поље за одабир датума, помоћу којег се заказује следећи преглед, при чему се прво бира месец, а потом дан и година заказаног прегледа.

### Пример целокупне картице *nalaz i dijagnoza*:

podaci pacijenta

nalaz i dijagnoza

Unesite broj zdravstvene knjižice:

Broj zdravstvene Knjizice

Izaberite vrstu pregleda:

Prvi/Kontrolni

Anamneza:

Unesite anamnezu

Izaberite laboratorijsku analizu:

Nothing selected

Fizikalni nalaz:

Unesite fizikalni nalaz

Dijagnoza:

Unesite dijagnoza

Terapija:

Unesite terapiju

Izaberite imageing proceduru:

Nothing selected

Izaberite lek:

Nothing selected

Zakaži sledeći pregled: mm/dd/yyyy

Sačuvaj nalaz

Примери поља  
за унос налаза

Примери поља  
вишеструког избора

Заказивање прегледа

Дугме за чување налаза у базу

## 2.4 Заказани прегледи

Ова страница доктору пружа могућност да види распореде заказаних прегледа за одређени дан, а такође је могућа претрага заказаних прегледа по датуму. Уколико за одабрани датум нема заказаних прегледа, лекару се приказује одговарајућа порука, а у супротном се излиставају пацијенти који су заказани за тај датум. За сваког пацијента приказује се број здравствене књижице, име, презиме, датум рођења и контакт телефон.

Примери странице заказани прегледи у зависности од резултата претраге:

**Zakazani pregledi** Izaberite datum: 05/22/2020 Ucitaj pregledе

Nema zakazanih pregleda za danas

Резултат претраге

Поље за одабир датума

Дугме за претрагу по датуму

---

**Zakazani pregledi** Izaberite datum: 05/28/2020 Ucitaj pregledе

Pacijent: Dusan Tisma  
Broj knjizice: 555333, Telefon: 0669898767, Datum rođenja: 2001-09-15 Pregled

Pacijent: Pera Peric  
Broj knjizice: 12343214, Telefon: 2342131231, Datum rođenja: 1999-01-01 Pregled

Картица са подацима пацијента који има заказан преглед за одабрани датум





Дугме које преусмерава доктора на страницу за писање налаза

## 2.5 Пацијенти

На овој страници се приказују сви пацијенти који су долазили у болницу, која користи ову апликацију. Пацијенти се излиставају у виду картица са информацијама о пацијенту, као на страници „Заказани прегледи“ за сваког пацијента се приказују општи подаци као што су: име, презиме, контакт телефон, датум рођења и број књижице. Такође, притиском на дугме “налаз”, приказују се сви бивши прегледи, односно лекарски извештаји, одабраног пацијента. Ова страница пружа могућност претраживања пацијената по имену или презимену и на тај начин олакшава доктору претрагу, уколико постоји велики број пацијената.


## Слике странице „пацијенти“:

Пацијенти Pretraži:

<b>Pacijent: Andrija Nikić</b> Broj knjižice: 01010101, Telefon: 0667875432, Datum rođenja: 2001-03-02	Nalazi
Pacijent Andrija Nikić pregledan je datuma 2020-04-29. Pregled je bio Prvi 	
Pacijent Andrija Nikić pregledan je datuma 2020-05-22 	
Pacijent Andrija Nikić pregledan je datuma 2020-05-25. Pregled je bio Kontrolni 	
<b>Pacijent: Mileta Jovanović</b> Broj knjižice: 433433, Telefon: 0655433432, Datum rođenja: 2001-05-05	Nalazi
Pacijent nema zabeleženih pregleda.	
<b>Pacijent: Dusan Tisma</b> Broj knjižice: 555333, Telefon: 0669898767, Datum rođenja: 2001-09-15	Nalazi
Pacijent Dusan Tisma pregledan je datuma 2020-04-29. Pregled je bio Kontrolni 	

Пацијенти Pretraži:

<b>Pacijent: Mitar Mirić</b> Broj knjižice: 2536475, Telefon: 6246222362, Datum rođenja: 1999-01-01	Nalazi
<b>Pacijent: Mihailo Trajković</b> Broj knjižice: 2536432, Telefon: 84573462462, Datum rođenja: 2000-12-31	Nalazi
<b>Pacijent: Aleksa Prokić</b> Broj knjižice: 2536111, Telefon: 0667876543, Datum rođenja: 2005-01-01	Nalazi
<b>Pacijent: Pera Perić</b> Broj knjižice: 12343214, Telefon: 2342131231, Datum rođenja: 1999-01-01	Nalazi
<b>Pacijent: Natalija Branković</b> Broj knjižice: 12345555, Telefon: 53423241, Datum rođenja: 1999-04-30	Nalazi
<b>Pacijent: Ana Savić</b> Broj knjižice: 12345577, Telefon: 53423241, Datum rođenja: 1999-04-30	Nalazi
<b>Pacijent: Dusan Tisma</b> Broj knjižice: 555333, Telefon: 0669898767, Datum rođenja: 2001-09-15	Nalazi
<b>Pacijent: Andrija Nikić</b> Broj knjižice: 01010101, Telefon: 0667875432, Datum rođenja: 2001-03-02	Nalazi

Притиском на дугме  лекару се приказује целокупни лекарски извештај прегледа који је одржан наведеног датума. На извештају се може видети следеће:

- Име и презиме пацијента
- Анамнеза, дијагноза, физикални налаз и терапија
- Лекови који су преписани пацијенту
- Потребне лабораторијске анализе
- Потребне *imaging* процедуре
- Име и презиме лекара који је извршио преглед



Pacijent: Ana Savić

Datum pregleda: 2020-05-25

Vrsta pregleda: Prvi

Anamneza:

Unazad 2 meseca, dete ima bolove u trbuhu. Bolovi su nakon obroka. Ima noćni bol iza grudne kosti, rani jutranji bol iznad pupka. Stolica je uredna, svakodnevne, mokri uredno U poridici, otac ima ulkusnu bolest želuca i helicobacter je pozitivan

Fizikalni nalaz:

ždrelo: mirno, tragusi neosetljiv, pulmo obostrano uredan disajni zvuk cor: kacija ritmična tonovi jasni, bez patološkog šuma, SF: 72/min, TA: 110/70mmHg, abdomen je palpatarno mek,lako bolno osjetljiv u predlu epigastijuma i ispod ksifoidnog nastavka, jetra i slezina se palpiraju u fiziološkim granicama. Neurološki status je uredan

Pacijent je radio sledeće laboratorijske analize: krvna slika .

Dijagnoza:

Dyspepsia K30.0 Refluksa bolest jednjaka K21.0

Pacijentu su prepisani sledeći lekovi: Nolpaza ,Famotidin ,Riopan .

Terapija:

Nolpaza 1 x 40mg ujutro pola sta pre obroka, Famotidin 1 x 40mg uveče pola sata pre obroka Riopan 1 x 1 dnevno 2-3 x dnevno između obroka - dopunski uraditi helicobacter antigen u stolici - kontrola za tri nedelje, u slučaju pozitivnog nalaza na HBP antigen u stolici, započeti triplu eradikacionu terapiju. U slučaju negativnog nalaza na HBP antigen u stolici, a perzistiranju tegoba, indicovana je gastroskopija

Pregledao: dr. Aleksa Prokić

### 3.1 Пројектовање базе

За сваког пацијента знамо: име, презиме, датум рођења, контакт телефон и број здравствене књижице. Сваки пацијент, као и лекар, има јединствени идентификациони код под којим је обележен у бази.

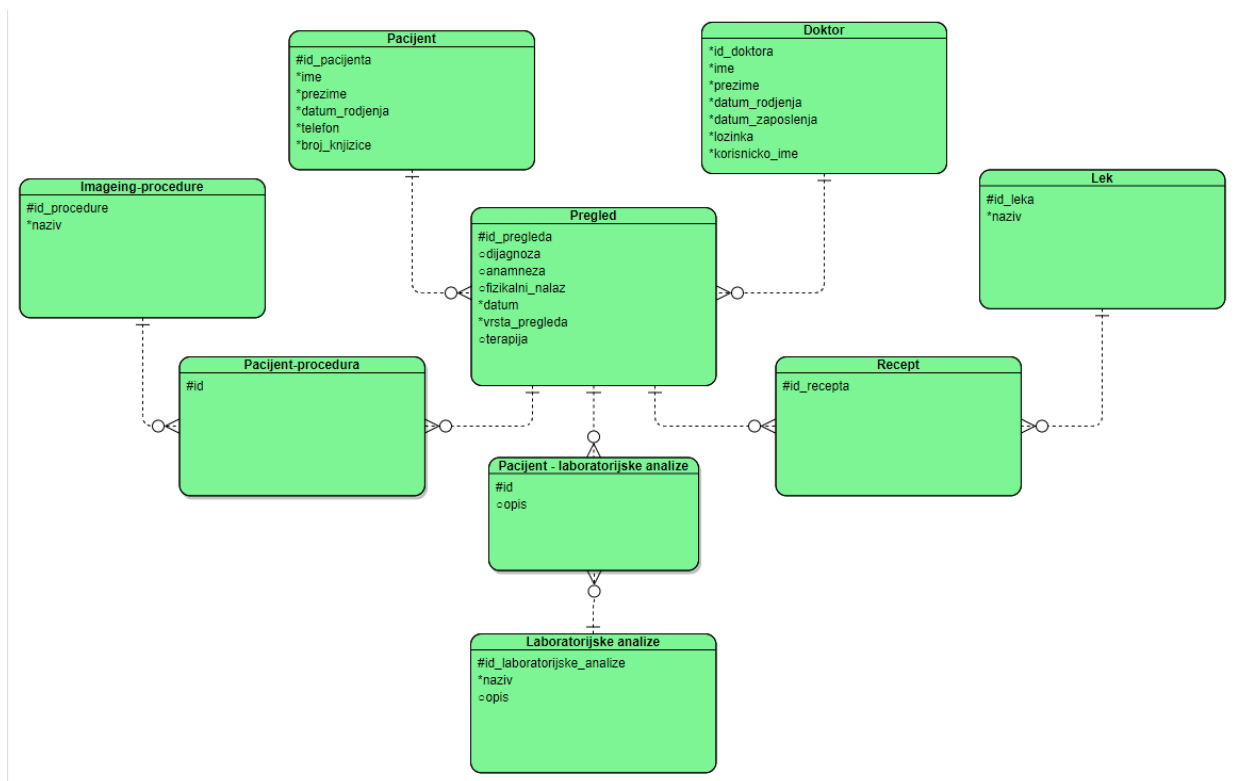
Уколико пацијент први пут долази на преглед, уноси се у базу на основу броја здравствене књижице, а уколико је пацијент раније већ био у болници, на основу истог броја књижице попуњава се за њега лекарски извештај.

Ентитет преглед представља везу *више-више* између пацијента и лекара. Састоји се из јединственог идентификационог броја (*id*), датума и разних атрибута, који одговарају лекарском извештају (дијагноза, анамнеза, физикални налаз...). Ставке лекарског извештаја, као нпр. дијагноза, су опције и уколико су попуњене преглед се сматра урађеним, а у супротном преглед се сматра заказаним. При попуњавању прегледа који су претходно евидентирани као заказани преглед, у бази се не креира нови преглед, већ се постојећи ажурира, односно поља која су претходно била празна (као нпр. дијагноза) добију вредност.

**\*Пословна напомена:** Лекарски извештај са непопуњеним пољима за дијагнозу, не би имао смисла и управо зато је у том случају погодно преглед евидентирати као заказан.

Остале везе и ентитети логичког модела базе могу се видети са следеће слике:

## 3.2 Логички модел базе



## 4. Коришћене технологије

Ова апликација развијена је помоћу програмских језика *php* и *JavaScript*. За изглед и дизајн коришћени су *HTML* и *CSS*, као и *framework Bootstrap* и библиотека *Font Awesome*.

### 4.1 PhpMyAdmin

*PhpMyAdmin* је бесплатан софтвер развијен помоћу *php* програмског језика, осмишљен за администрацију и управљање база података *web*-апликација, креираних помоћу *MySQL* програмског језика. *PhpMyAdmin* користи *MariaDB* базу података, и пружа велики избор операција над базом помоћу *MySQL* програмског језика. *MySQL* упити могу се позивати из развијане апликације, али такође их је могуће покренути у самом *phpMyAdmin-u*, који има богат кориснички интерфејс.

The screenshot displays the phpMyAdmin web interface. On the left, a sidebar shows the database structure with 'bolnica\_baza' selected. The top navigation bar includes options like 'Browse', 'Structure', 'SQL', 'Search', 'Insert', 'Export', 'Import', 'Privileges', 'Operations', and 'Triggers'. The main area shows a successful SQL query execution: 'UPDATE `pacijenti` SET `ime` = 'Nemanja', `prezime` = 'Joksimović' WHERE `pacijenti`.`id\_pacijenta` = 1;'. Below the query, it states 'Showing rows 0 - 8 (9 total, Query took 0.0003 seconds.)'. A table view of the 'pacijenti' table is shown below, with columns: id\_pacijenta, ime, prezime, datum\_rođenja, telefon, and broj\_knjizice. The table contains 10 rows of patient data. Red annotations highlight the 'bolnica\_baza' database in the sidebar, the SQL query execution area, and the 'pacijenti' table.

База података коришћена у апликацији за болницу

Могућности које phpMyAdmin пружа

Покренути "SQL" упити

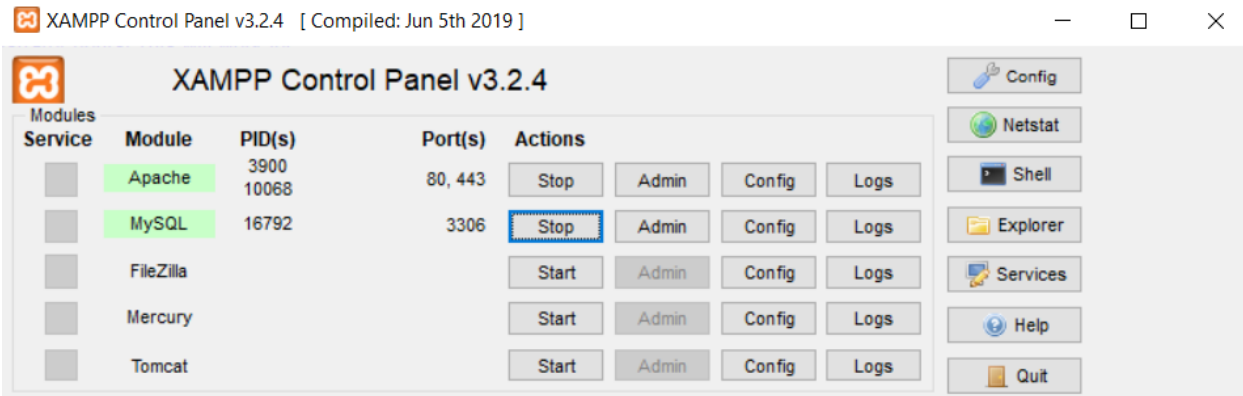
Табела пацијенти

### 4.2 MariaDB

*MariaDB* је једна од најпопуларнијих и најкоришћенијих база података. Развили су је оснивачи *MySQL* програмског језика и управо је то чини веома погодном и поузданом за употребу. *MariaDB* се користи зато што је брз и има велики број могућности погодних за рад у различитим ситуацијама. *MariaDB* је развијен као отворени софтвер (енг. *Open source*), осмишљен за рад са релационим базама података, који такође поседује кориснички интерфејс за покретање и извршавање *SQL* упита.

### 4.3 XAMPP

XAMPP је бесплатан и отворени софтвер (енг. *Open source*), дизајниран и осмишљен од стране компаније *Apache Friends*. Користи се при раду са серверима, базама података и при покретању *php* и *perl* програмских језика. Овај програм користи се у комбинацији са претходно поменути *MariaDB* и *phpMyAdmin*. Фирма *Apache Friends* је осмислила овај програм, како би помогла програмерима да свој рад могу да тестирају на својим рачунарима, без икаквог приступа интернету. Другим речима, XAMPP симулира сервер и комуникацију са сервером на рачунарау, без подизања сервера на Интернет.



## 5. Програмски код апликације

### 5.1 Регистрација лекара у систем

Форма за регистрацију лекара:

```
<form action="authentication.php" method="post">
  <label for="username">
    <i class="fas fa-user"></i>
  </label>
  <input type="text" name="username" placeholder="Korisnicko ime"
  id="username" required>
  <label for="password">
    <i class="fas fa-lock"></i>
  </label>
  <input type="password" name="password" placeholder="Lozinka"
  id="password" required>
  <input type="submit" value="Login">
</form>
```

Помоћу корисничког имена и лозинке, притиском на дугме типа *submit* доктор се пријављује у систем, при чему дугме активира *php(authentication.php)* код, који на основу унетих података врши регистрацију корисника у базу.

***authentication.php:***

```
<?php
session_start();

$DATABASE_HOST = 'localhost';
$DATABASE_USER = 'root';
$DATABASE_PASS = '';
$DATABASE_NAME = 'bolnica_baza';

$con = mysqli_connect($DATABASE_HOST, $DATABASE_USER,
    $DATABASE_PASS, $DATABASE_NAME);

if ( mysqli_connect_errno() ) {
    exit('Greska pri povezivanju na server: ' . mysqli_connect_error());
}

if ( !isset($_POST['username'], $_POST['password']) ) {
    exit('Korisnicko ime i lozinka nisu uneti');
}

if ($stmt = $con-
>prepare('SELECT id_doktora, lozinka FROM doktori WHERE korisnicko_ime = ?')) {

    $stmt->bind_param('s', $_POST['username']);
    $stmt->execute();
    $stmt->store_result();

    if ($stmt->num_rows > 0) {
        $stmt->bind_result($id, $lozinka);
        $stmt->fetch();
        if (password_verify($_POST['password'], $lozinka)) {

            session_regenerate_id(); // ovo je preventivna mera protiv presretanj
a komunikacije baze i nas
            $_SESSION['loggedin'] = TRUE;
            $_SESSION['korisnicko_ime'] = $_POST['username'];
            $_SESSION['id_doktora'] = $id;

            echo header("Location: ../medical_system/main_page.php");
        } else {
```

```

        echo 'Incorrect password!';
    }
} else {
    echo 'Incorrect username!';
}

$stmt->close();
}

?>

```

*SESSION (sesija)* је један од начина за складиштење и приступање подацима и информација који се користе на различитим страницама у оквиру једне апликације. За разлику од колачића (енг. *Cookies*), подаци се не чувају на рачунарима корисника, већ се чувају на интернету, што је једна од мана сесије, јер су подаци рањиви. Један од начина заштите јесте метода *session\_regenerate\_id()*, која спречава пресретање комуникације корисника са сесијом.

Да би приступ подацима сесије био могућ у оквиру неке странице, на почетку те странице је неопходно покренути сесију позивом методе *session\_start()*.

Сама регистрација лекара одвија се помоћу *SQL* упита, којим се проверава да ли унета лозинка и корисничко име постоје у бази. При позивању ових упита користи се заштита од пресретања података у виду *MySQL Prepared Statements-a* који имају посебне методе као што је метода *bind\_param* која врши евалуацију података *SQL* упита и спречава злонамерне нападе на базу.

## Читање информација о лекару

Након успешног пријављивања у систем, доктору се приказују лични и статистички подаци.

**Читање података о лекару извршава се помоћу следећег кода:**

```

//opsti podaci doktora
$query1 = "SELECT * FROM doktori WHERE id_doktora = '$id_doktora'";
$result1 = mysqli_query($db, $query1);
//broj odrzanih pregleda doktora
$query2 = "SELECT COUNT(id_doktora) AS 'br_preg' FROM pregledi WHERE id_doktora
'$id_doktora' AND dijagnoza != ''";
$result2 = mysqli_query($db,$query2);
//broj pacijenata doktora
$query3 = "SELECT COUNT(DISTINCT id_pacijenta) AS 'br_pac' FROM pregledi WHERE id
_doktora = '$id_doktora' ";
$result3 = mysqli_query($db,$query3);

```

```
//broj zakazanih pregleda doktora
$query4 = "SELECT COUNT(id_doktora) AS 'br_zak_preg' FROM pregledi
          WHERE id_doktora = '$id_doktora' AND dijagnoza = ''";
$result4 = mysqli_query($db,$query4);
```

Резултати ових упита се помоћу класе *stdClass()* смештају у низ који је се потом шаље назад ка клијентској страни. *StdClass()* је празна класа *PHP* програмског језика која се користи за претварање осталих типова података у објекте. Веома је погодна за рад, јер инстанце ове класе поред вредности имају и назив.

**Пример кода који користи *stdClass()*:**

```
$el = new stdClass();
$el->broj_pregleda = $row;
$ret[]=$el;
```

## 5.2 Код странице пацијенти

Лекару се на страници пацијенти приказују картице са информацијама о пацијенту и њиховим лекарским извештајима. Такође, на страници се налази и поље за претрагу пацијената по имену или презимену.

**Поље за претрагу по имену или презимену:**

```
<input type="text" id="ime_pacijenta" name="ime_pacijenta"
        placeholder="ime_pacijenta" class="mt-1">
```

Уношењем тескта у то поље, односно променом вредности тог поља, активира се догађај *ime\_prezime.addEventListener*. Унутар овог догађаја прво се позива код *load\_pacijenti\_with\_keywords.php*, а затим се на основу резултата позваног кода динамички креирају картице са информацијама о пацијенту, а такође се динамички креирају и лекарски извештаји. Читање података из базе, односно позивање поменутог *php* кода, врши се помоћу уграђене функције *javascript* језика која се назива *Ajax*. Он је веома погодан за рад због разних функционалности који носи са собом, а једна од њих је читање података из базе без освежавања и поновног учитавања отворене странице.

**Део кода са *ajax-ом*:**

```
var pacijenti_ajax = new XMLHttpRequest();
pacijenti_ajax.onreadystatechange = function() {
if (this.readyState == 4 && this.status == 200)
{
    console.log("success! ucitan pri ulasku");
    var podaci_pacijenata = JSON.parse(pacijenti_ajax.responseText);
//Остатак кода је изостављен ради веће прегледности
};
pacijenti_ajax.open("POST", "load_pacijenti_with_keywords.php", true);
```

```
pacijenti_ajax.send();
```

Након позива из *ajax-a* извршава се већ поменути код, који најпре из базе учитава све пацијенте чија имена и презимена одговарају критеријуму унетом у поље за претрагу.

*load\_pacijenti\_with\_keywords.php (kod) :*

Претрага пацијената по задатом критеријуму:

```
for($i = 0; $i < count($keywords); $i++)
{
    $word = $keywords[$i];
    $query = "SELECT * FROM pacijenti
              WHERE ime LIKE '%{$word}%' OR prezime LIKE '%{$word}%'";
    $result = mysqli_query($db, $query);
    //Остатак кода је изостављен ради веће прегледности.
}
```

Након што су из базе учитани одговарајући пацијенти, за сваког од њих се даље читају подаци и информације о свим досадашњим лекарским извештајима, преписаним лековима, као и извршеним лабораторијским анализама и *imaging* процедурама.

**Код који за пацијента наведеног идентификационог кода чита информације о свим бившим прегледима:**

```
$query1 = "SELECT * FROM pregledi WHERE id_pacijenta = '$id_pacijenta' ";
$result1 = mysqli_query($db,$query1);
```

Након извршења овог кода, за сваки од учитаних прегледа се даље читају информације о лекару који је извршио преглед, лековима који су за тај преглед преписани, као и потребним лабораторијским анализама и *imaging* процедурама.

**Примери читања поменутих информација везаних за појединачан преглед:**

```
$query2 = "SELECT * FROM doktori
WHERE id_doktora = '$id_doktora' ";
$result2 = mysqli_query($db,$query2);
$podaci_lekara = [];
while($row2 = mysqli_fetch_object($result2))
{
    $podaci_lekara = $row2;
}
$pregled->podaci_lekara = $podaci_lekara;

$query2 = "SELECT naziv FROM pacijent_lek
JOIN lekovi USING(id_leka)
WHERE id_pregleda = '$id_pregleda' ";
$result2 = mysqli_query($db,$query2);
```



```

$lekovi = [];
while($row2 = mysqli_fetch_object($result2))
{
    $lekovi[]=$row2;
}
$pregled->lekovi = $lekovi;

```

Све прочитане информације се додељују као вредности објекта *\$pregled* поменуте класе *stdClass()*. Сваки објекат *\$pregled* се затим додаје као вредност објекта *\$pac\_pregled*, такође класе *stdClass()*, који у себи чува информације о свим прегледима везаним за једног пацијента. На самом крају се *\$pac\_pregled* надовезује на низ *\$ret*, који је резултат рада ове *php* скрипте.

#### Поједини делови кода:

```

while($row = mysqli_fetch_assoc($result1))
{
    $pregled = new stdClass();
    $pregled->podaci_pregleda = $row;
    $pregled->podaci_lekara = $podaci_lekara;
    $pregled->lekovi = $lekovi;
    $pregled->procedure = $procedure;
    $pregled->lab_analize = $lab_analize;
    $pomNiz[]=$pregled;
}

$pac_pregled->podaci_o_pregledima = $pomNiz;
$ret[]=$pac_pregled;

```

Оператором *[]=* се врши надовезивање, односно додавање једног низа као елемент другог низа, што се може видети на примеру *\$ret[]=\$pac\_pregled*, где *\$pac\_pregled* постаје један од елемената низа *\$ret*.

Низ *\$ret* се враћа као резултат овог кода, а његова структура може се видети на следећој шеми:

У поље за претрагу је уписана слова „СА“ и као резултат су учитана 3 пацијента, као и сви њихови прегледи, са свим информацијама.

pacijenti.php:272

```

▼ Array(3)
  0: {info: {...}, podaci_o_pregledima: Array(1)}
  1: {info: {id_pacijenta: "8", ime: "Dusan", prezime: "Tisma", datum_rođenja: "2001-09-15", telefon: "0669898767", ...}, podaci_o_pregledima: Array(3)}
    0: {podaci_pregleda: {...}, podaci_lekara: {...}, lekovi: Array(0), procedure: Array(0), lab_analize: Array(0)}
      1: {lab_analize: [], lekovi: [], podaci_lekara: {id_doktora: "1", ime: "Aleksa", prezime: "Prokić", datum_rođenja: "2019-09-12", datum_zaposlenja: "2020-01-08", ...}, podaci_pregleda: {id_pregleda: "10", vrsta_pregleda: "", anamneza: "", fizikalni_nalaz: "", datum: "2020-05-13", ...}, procedure: []}
      __proto__: Object
    2: {podaci_pregleda: {...}, podaci_lekara: {...}, lekovi: Array(0), procedure: Array(0), lab_analize: Array(0)}
      length: 3
      __proto__: Array(0)
  2: {info: {...}, podaci_o_pregledima: Array(0)}
    length: 3
    __proto__: Array(0)

```

Подаци о пацијенту и свим његовим прегледима

Сви подаци везани за први преглед

Садржај једног прегледа

```

▼ Array(3)
  0: {info: {...}, podaci_o_pregledima: Array(1)}
  1: {info: {...}, podaci_o_pregledima: Array(3)}
  2: {info: {...}, podaci_o_pregledima: Array(0)}
  length: 3
  __proto__: Array(0)

```

Након што је *php* скрипта *load\_pacijenti\_with\_keywords.php* извршена, на клијентској страни се помоћу резултата ове скрипте (помињани низ *\$ret*), генеришу картице са подацима пацијента и такође се генеришу *div* елементи са подацима о лекарским извештајима. Сви елементи који се креирају динамички се састоје из више мањих, такође динамички креираних елемената.

#### Метода којом се динамички креирају елементи:

```

function createElement(elementTag, elementId, html, elementClass)
{
    var newElement = document.createElement(elementTag);
    if(elementId != "") newElement.setAttribute('id', elementId);
    newElement.innerHTML = html;
    if(elementClass != "") newElement.setAttribute('class', elementClass);
    return newElement;
}

```

#### Метода којом се динамички генерише садржај једног лекарског извештаја:

```

function generateNalaz(podaci_pregleda, podaci_doktora, podaci_pacijenta, lekovi,
procedure, lab_analize)

```

У овој функцији се помоћу прослеђених параметара и функције *createElement()* креирају мањи елементи који се потом групишу у крупније елементе. Резултат рада ове функције је елемент *div* класе „*moj\_container*“. Након што је налаз креиран, он се затим додељује картицама са информацијама о пацијенту које су финални исход динамичког генерисања елемената. Неки од динамички креираних елемената садрже елемент *button*, *icon* и *span*. Пошто су ови елементи, такође креирани дианмички, приступ њиховом *click* догађају је мало специфичан. Да би се догађај *click* активирао, неопходно је да знамо за који елемент је везан, односно потребно је знати *id* тог елемента. У овој апликацији овај проблем је решен помоћу функционалности *framework-a jQuery*.

#### Пример:

```
$('#div_za_prikaz').on('click', 'span', function(e)
{
    var id = e.target.id;
    var klasa = e.target.className;
    console.log(klasa);
    var id1 = e.target.id;
    var i_j = id1.split(":");
    console.log(i_j);
    var i = i_j[1];
    var j = i_j[2];
    var id2 = "myModal:"+i+":"+j;
    console.log(id2);
    modal = document.getElementById(id2);
    modal.style.display = "none";
});
```

Оно што можемо видети на овом примеру кода написаном помоћу *jQuery-a*, јесте начин на који се за неки елелмент добија жељени *id*. У елементу *id-a* *'#div\_za\_prikaz'* се посматрају сви елементи са тагом „*i*“. Уколико је неки од њих притиснут, активира се догађај *e.target* који враћа *id* оног елемента који је довео до позивања овог догађаја.

У овом делу кода такође можемо приметити имплементирану метуду језика *JavaScript* која се користи у раду са подацима типа *string*. Ова метода почетни *string* дели на више мањих,

```
▼<div class="moj_container">
  ▼<div class="nalaz_stavke">
    ▼<p class="moj_paragraf display-4">
      <strong>Pacijent: </strong>
      "Andrija Nikić"
    </p>
    ▶<p class="moj_paragraf">...</p>
    ▶<p class="moj_paragraf">...</p>
    ▶<p class="moj_paragraf">...</p>
  </div>
  ▶<div class="nalaz_blok">...</div>
  ▶<div class="nalaz_stavke">...</div>
  ▶<div class="nalaz_blok">...</div>
  ▼<div class="nalaz_stavke">
    ▼<p class="moj_paragraf">
      <strong>Pacijentu su prepisani sledeći
      lekovi: </strong>
      "berodual ,pulmikord ."
    </p>
  </div>
  ▶<div class="nalaz_blok">...</div>
  ▶<div class="nalaz_stavke">...</div>
</div>
```

на оним местима где се у полазном *string-u* налази прослеђени параметар, у овом случају карактер “:” ( `var i_j = id1.split(":")` ).

**Коначан изглед једне картице након завршетка динамичког генерисања:**

```
▼<div id="pregled-7">
  ▶<div id="myModal:7:0" class="modalmy">...</div>
  ▼<div id="myModal:7:2" class="modalmy">
    ▼<div id="modalContent:7:2" class="modal-content">
      <span id="span:7:2" class="close">x</span>
      ▼<div id="nalaz:7:2">
        ▼<div class="moj_container">
          ▶<div class="nalaz_stavke">...</div>
          ▶<div class="nalaz_blok">...</div>
          ▶<div class="nalaz_stavke">...</div>
          ▶<div class="nalaz_blok">...</div>
          ▶<div class="nalaz_stavke">...</div>
          ▶<div class="nalaz_blok">...</div>
          ▶<div class="nalaz_stavke">...</div>
        </div>
      </div>
    </div>
  </div>
  ▼<div id="myModal:7:4" class="modalmy">
    ▼<div id="modalContent:7:4" class="modal-content">
      <span id="span:7:4" class="close">x</span>
      ▶<div id="nalaz:7:4">...</div>
    </div>
  </div>
  ▶<div id="inside_pregled-7" class="row" style="background-color: rgb(207, 218, 255);">...</div>
  ▼<div id="collapseDiv-7" class="collapse">
    ▼<div id="inside_collapseDiv-7" class="card card-body">
      ▶<p id="paragraf-7:0">...</p>
      ▼<p id="paragraf-7:2">
        "Pacijent Andrija Nikić pregledan je datuma 2020-05-22. Pregled je bio Kontrolni."
        <i id="icon:7:2" class="fas fa-book-medical fa-lg"></i>
      </p>
      ▶<p id="paragraf-7:4">...</p>
    </div>
  </div>
</div>
```

**Напомена:** Странице „пацијенти” и „заказани прегледи” су имплементиране на исти начин уз минималне разлике у структури.

### 5.3 Код странице налаз/дијагноза

Ова страница сачињена је од многобројних поља за унос текста, као и поља вишеструког избора. Након што су унети потребни подаци пацијента, или након што је попуњен лекарски извештај, унете информације се уписују у базу. Као и на претходним страницама при позивању *php* скрипти користи се функционалност *JavaScript* језика – *ajax*.

**Примери поља за унос и одабир података:**

```
<label for="anamneza" style="font-size: 20px">Anamneza:</label><br>
<textarea id="anamneza" class = "nalaz_data shadow
p-3" name="anamneza" placeholder="Unesite anamnezu" ></textarea><br><br>
```

```
<label for="lab_anal" style="font-
size: 20px">Izaberite laboratorijsku analizu:</label><br>
```

```
<select id="cbbox1" name="cb_lab_anal" class="selectpicker"
multiple data-live-search="true"></select><br><br>
```

```
<label for="zakazivanje_pregleda"
style="font-size: 20px">Zakaži sledeći pregled:</label>
```

```
<input type="date" class = "nalaz_data " id="datum_sled_pregleda"
name="datum_sled_pregleda"><br><br>
```

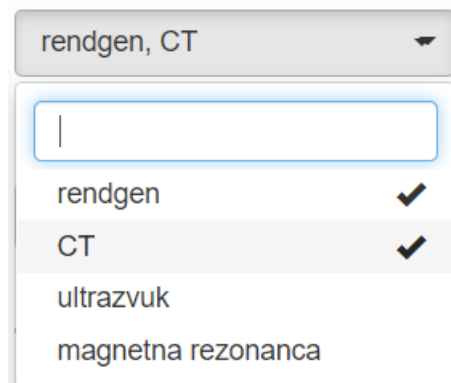
При креирању појединих елемената коришћене су класе *Bootsrtap-a*, коју је бесплатан *CSS framework* погодан за дизајнирање сајтова. Неке од често коришћених класа су класе *row* и *col*, које се користе при расподели простора унутар неког елемента. Класе *mt*, *mr*, *pl*, *display* и остале се користе за лако постављање маргина, величина наслова и фонта итд...

Посебну пажњу посветио бих елементу *select* класе „*selectpicker*“, који за разлику од обичног *select* елемента пружа могућност вишеструког одабира података, као и претраге понуђених опција. У овој апликацији користе се три оваква елемента који служе за одабир лекова, лабораторијских анализа и *imaging* процедура.

Вредности одабраних опција читају се и узимају помоћу методе *frameworka jQuery* која се зове *serializeArray()*. Ова метода као резултат враћа низ елемената које чине одабране опције елемента *select*.

```
var form_data_lekovi =
$('#cbbox3').serializeArray();
```

„*#cbbox2*“ је *id* елемента *select* који служи за одабир потребних *imaging* процедура. На слици поред је пример једног *select* елемента класе „*selectpicker*“, у овом случају за одабир *imaging* процедура



The image shows a 'selectpicker' dropdown menu. The selected option is 'rendgen, CT'. The dropdown list is open, showing four options: 'rendgen' (checked), 'CT' (checked), 'ultrazvuk', and 'magnetna rezonanca'.

Option	Selected
rendgen	✓
CT	✓
ultrazvuk	
magnetna rezonanca	

Након што су сви подаци из поља за унос текста и поља вишеструког избора прочитани, њихове вредности се смештају у објекат *formData*, који је веома погодан за рад са *XMLHttpRequest* захтевима, јер елементи овог објекта имају кључ и вредност (може се направити поређење са асоцијативним низовима), што га чини веома адекватним за рад са „POST“ захтевима.

**Следи пример кода који користи поменуте елементе:**

```
var formData = new FormData();
for(var i=0; i<elements.length; i++)
{
    formData.append(elements[i].name, elements[i].value);
}
var form_data_imgpcd = $('#cbbox2').serializeArray();
var pom = [];
for(var i = 0; i <form_data_imgpcd.length; i++)
{
    pom[i] = form_data_imgpcd[i].value;
}
var form_data_lekovi = $('#cbbox3').serializeArray();
var pom2 = [];
for(var i = 0; i <form_data_lekovi.length; i++)
{
    pom2[i] = form_data_lekovi[i].value;
}
var form_data_lab = $('#cbbox1').serializeArray();
var pom3 = [];
for(var i = 0; i <form_data_lab.length; i++)
{
    pom3[i] = form_data_lab[i].value;
}

formData.append('procedure',pom);
formData.append('lekovi',pom2);
formData.append('analyze',pom3);
formData.append('zak_datum_iz_sessiona',zak_datum_iz_sessiona);
formData.append('br_knjiz_iz_sessiona',br_knjiz_iz_sessiona);
formData.append('id_pregleda',id_pregelda_iz_sessiona);
var ajax = new XMLHttpRequest();
ajax.onreadystatechange = function() {
    ajax.open("post", "insert_diagnosis.php");
    ajax.send(formData);
}
```

Након што су потребни подаци прослеђени ка серверској страни, следи упис тих података у базу.

### ***insert\_diagnosis.php:***

Прослеђеним подацима приступа се помоћу „POST“ захтева што се може видети на следећем примеру:

```
$niz = $_POST["procedure"];
$niz2 = $_POST["lekovi"];
$niz3 = $_POST["analize"];
$br_knjizice = $_POST["broj_knjizice"];
$vrsta_pregleda = $_POST["vrsta_pregleda"];
$anamneza = $_POST["anamneza"];
$fiz_nalaz = $_POST["fiz_nalaz"];
$dijagnoza = $_POST["dijagnoza"];
$terapija = $_POST["terapija"];
```

Пре самог уписа података у базу, врши се њихова евалуација и исправност и уколико неки од њих не задовољавају потребне услове, резултат овог кода је број -1. Касније на клијентској страни, на основу вредности тог броја се кориснику приказује порука о успешном или неуспешном уписивању у базу.

Већ је поменуто да се преглед води евидентиран као заказан уколико је вредност атрибута „дијагноза“ празна. Уколико се попуњава лекарски извештај заказаног прегледа, у бази се ажурирају атрибути постојећег прегледа, који су претходно били празни. У супротном се једноставно креира нови ред у табели „прегледи“.

**Следи неколико примера уписивања података везаних за један преглед у базу:**

```
$query1 = "INSERT INTO pregledi (vrsta_pregleda,anamneza,fizikalni_nalaz,datum,
dijagnoza,terapija,id_doktora,id_pacijenta)
VALUES('$vrsta_pregleda','$anamneza',
'$fiz_nalaz','$datum','$dijagnoza','$terapija','$id_doktora','$id_pacijenta')";
mysqli_query($db, $query1);
```

```
$query3 = "INSERT INTO pacijent_procedura (id_pregleda,id_procedure)
VALUES('$id_pregleda','$id_procedure')";
mysqli_query($db, $query3);
```

```
$query7 = "UPDATE pregledi
SET vrsta_pregleda = '$vrsta_pregleda',anamneza = '$anamneza',
fizikalni_nalaz = '$fiz_nalaz',dijagnoza='$dijagnoza',
datum = '$datum',terapija='$terapija'
WHERE id_pregleda = '$id_pregleda_iz_sessiona'";
mysqli_query($db,$query7);
```

## **6. Закључак**

Веб програмирање је једно од најтраженијих послова данашњице.

Развијање ове апликације помогло ми је да схватим које изазове и проблеме оно са собом носи. Рад са овом апликацијом, такође ме је упознао са системима које користе здравствене установе као и потребним технологијама и програмским језицима потребним за креирање једне веб апликације.



## 7. Литература

1. Рад са елементима вишеструког избора класе *select*:  
<https://developer.snapappointments.com/bootstrap-select/>
2. Документације *php* програмског језика:  
<https://www.php.net/docs.php>
3. Документације *JavaScript* програмског језика:  
<https://devdocs.io/javascript/>
4. Литература везана за *HTML* и *CSS*:  
<https://www.w3schools.com/>
5. Документација *Bootstrap framework-a*:  
<https://getbootstrap.com/>
6. Документација библиотеке *Fontawesome* за рад са фонтовима и иконама:  
<https://fontawesome.com/>
7. Сајт и форум коришћен за решавање недоумица и проблема приликом програмирања:  
<https://stackoverflow.com/>
8. Документација о бази података *MariaDB*:  
<https://mariadb.org/about/>
9. Документација о раду са програмом *XAMPP*:  
<https://www.apachefriends.org/docs/>
10. Документација о раду са програмом *phpMyAdmin*:  
<https://www.phpmyadmin.net/docs/>