Факултет инжењерских наука Универзитета у Крагујевцу

Назив студијског програма: Рачунарска техника и софтверско инжењерство

Ниво студија: Основне академске студије

Предмет: Софтверски инжењеринг 2

Број индекса: 556/2015

Јован Д. Петровић

Развој веб апликације за спровођење анкете

дипломски рад

Комисија за преглед и одбрану:

1. **Др Велибор Исаиловић, доцент - ментор**
2. _____ Датум одбране:_____
3. _____ Оцена: _____

Задатак дипломског рада

Током израде завршног рада потребно је да кандидат развије клијент – сервер апликацију за спровођење анкета. Апликација треба да омогући:

- састављање нових анкета и њихово чување на серверу
- приказ анкета које се налазе на серверу
- попуњавање анкета и чување резултата
- приказ резултата попуњених анкета

На почетној страни треба да буду приказане анкете које су тренутно актуелне. Када се кликне на неку од њих треба да се омогући приказ постојећих резултата анкете и попуњавање анкете уколико претходно није попуњена.

Препоручена литература:

[1] Rasmus Lerdorf, Kevin Tatroe, Peter MacIntyre, Programming PHP, 2nd Edition, електронско издање, САД, 2009.

[2] Larry Ullman, PHP and MySQL for Dynamic Web Sites: Visual QuickPro Guide, 4th Editioin, електронско издање, САД, 2017.

[3] Robin Nixon, Learning PHP, MySQL, JavaScript, CSS & HTML 5, Third Edition, електронско издање, САД, 2014.

[4] Matt Stauffer, Laravel: Up & Running, 2nd Edition, електронско издање, САД, 2019.

Крагујевац, октобар 2019. ментор:

Др Велибор Исаиловић

Abstract:

This thesis aims to present to its reader the concept and realization of a simple web survey application. Main tool used to produce the application is Laravel, a PHP web framework based on MVC (model-view-controller) architectural pattern. Main advantages of using Laravel are automation and facilitation of web application development process. Dynamic content produced by Laravel is transferred into HTML content used by the end-user's web browser. Internet enables effective distribution of content to the clients of application.

A survey is a way of collecting data from a target population of people, associated with a particular topic. Web application described in this thesis aims to simplify, automate and facilitate creation and distribution of surveys, while keeping and managing the data collected through the surveys using contemporary technology and advantages of modern computers and internet.

Key words: internet application, PHP, Laravel, survey, client-server

Резиме:

У овом раду је представљен и реализован концепт једноставне интернет апликације за анкетирање. За израду овог решења је коришћен *PHP* у форми програмског оквира (*framework*) *Laravel*, који се заснива на *MVC* (*model-view-controller*) софтверском архитектонском шаблону. Ларавел у великој мери олакшава писање програмског кода а одликује га аутоматизација и убрзавање процеса израде интернет апликација. Динамички садржај који он прави се на крају претвара у *HTML* код који се извршава на клијент страни апликације. Интернет омогућава ефикасну дистрибуцију садржаја до жељених корисника (клијената).

Анкета је вид прикупљања података везаних за одређену тему која се примењује над одређеном циљном популацијом. Ова интернет апликација има за циљ да поједностави, аутоматизује и олакша израду и одржавање анкета, а узгред омогући чување и управљање подацима прикупљених кроз попуњавање анкета коришћењем модерних технологија и предности савремених рачунара и интернета.

Кључне речи: интернет апликација, *PHP*, *Laravel*, анкета, клијент-сервер

Садржај

1.Увод

У природи одвајкада постоје непроменљиве законитости које неумитно утичу на сав живи свет. Као главни закон се издваја правило јачег, односно да најјача јединка у друштву има надмоћ над осталим физички инфериорним јединкама. Развојом свести о припадности заједници и утврђивањем заједничких циљева, појединачне јединке су откриле да удруживањем могу да надјачају јединке физички јаче од себе. То правило се из домена физичке снаге пренело и на остале аспекте живота једне заједнице, јединке се према заједничким интересима удружују у групе како би што боље задовољиле те интересе. Ове законитости се поред животињског света примењују и на људске заједнице: надмоћ више не припада физички најјачој јединки, већ бројчано најјачој групацији са заједничким интересима. Овај закон у већој или мањој мери доминира свим аспектима људског друштва, било да су то ратни сукоби (број људства и комада наоружања) или мирнодопски процеси (кључни принцип демократије јесте владавина већине над мањином). Мишљење већине формира такозване *мејнстрим (* енг. *main-главни, stream-ток)* токове који диктирају правац развоја људске културе и уметности. У светски распрострањеном претежно капиталистичком политичко-економском систему, главни фактор напретка је стицање и обрт капитала који је директно сразмеран броју потрошача неког производа односно прималаца услуга. Закон диктира да ће онај који стекне највише потрошача/поклоника имати највећу моћ и највећи капитал. Због ове пресудне особине људског друштва јавила се потреба за опажањем и тестирањем јавног мњења ради доношења бољих одлука или стицања предности над конкуренцијом, као и простог бележења промена у преференцијама једне заједнице.

Један од главних метода за испитивање јавног мњења јесте анкета. Анкета је вид прикупљања података везаних за одређену тему која се примењује над одређеном циљном популацијом. Пре развоја информационих технологија, анкете су биле везане за папир, што је значило да су постојале одређене препреке за њихово **успешно** обављање, пре свега дистрибуција и увид у резултате. Напретком информационих технологија, такви проблеми су превазиђени.

Систем описан у овом раду има за циљ да омогући корисницима да са лакоћом приступају јединственој бази података која чува анкете и податке о анкетама. Корисници су у могућности да праве анкете, праве измене над анкетама и бришу анкете. Такође имају увид у резултате анкета које су сами направили.

Израда програмског кода апликације је реализована у уређивачу текста *Visual Studio Code*. Веб апликације захтевају одвојене клијентске и серверске стране, а током развоја је у сврхе симулирања сервера коришћен *XAMPP* пакет са својим потпакетима. Програмски језик на којем је писана апликација је *PHP*, а *Laravel* је програмски оквир (енг. *framework*) коришћен за примену *MVC* (*model-view-controller*) софтверске архитектуре. За клијентску интеракцију са апликацијом се користи интернет претраживач Гугл Хром (*Google Chrome*).

2.Теоријске основе

У овом поглављу ће бити описани мотиви, предности и мане основних појмова који се тичу овог рада.

2.1 Анкета

Анкета је један од начина за прикупљање специфичниих података о циљаној популацији. Користи се за допуну знања у пољима као што су демографија и социјална истраживања и највише их користе психолози и социолози. Најзначајнији циљ анкете је стицање увида у мишљења, расположења и осећања циљне популације[1]. Анкете се могу разликовати по свом обиму, врстама питања и сврси. Могу бити усмене (лице у лице, телефонске) или писане на некој форми медијума (папир или електронским путем).

У наставку су наведене најзначајније предности коришћења анкета:

- релативно су лаке за реализацију
- простије су од осталих метода прикупљања података
- исплативе су у односу на уложена средства
- могућност спровођења електронским путем што неутралише стварање података са уске области око извора анкете
- истовремено прикупљање података од више јединки циљане популације истовремено
- стандардизоване су, што искључује одређене врсте грешака
- могућност анонимитета која подстиче испитане особе да дају искрене одговоре без страха од последица, са друге стране објективно процесирање информација од стране која спроводи анкету

Главне мане коришћења анкета су:

- особе које попуњавају анкету често нису мотивисане да дају искрене одговоре и мишљења
- често постављана питања су бројевне скале на којима се од испитане особе тражи да представи емоцију или мишљење једним бројем, што ван граничних вредности даје непрецизне резултате
- у зависности од различитих фактора, исто питање на анкети се може тумачити на више начина од стране више особа, што доводи до прикупљања погрешних информација[2].

Слика 1: *Пример класичне папирне анкете*

2.2 Интернет апликација

Интернет (веб) апликација је програм који се налази на удаљеном серверу, а интеракција са њим се остварује преко Интернета кроз интернет претраживач. Интернет сервиси су подскуп интернет апликација, а већина интернет страница садржи интернет апликације. По речима *Jarela Remicka*, компонента која обавља одређену функцију за корисника било које интернет странице сврстава ту страницу у интернет апликације. Интернет апликације имају широку примену, од прегледавања и писања електронске поште, преко радњи за наручивање свих врста робе, пријема и слања новца, праћења резултата, а у последње време и управљањем сензора и актуатора као део четврте индустријске револуције.

За разлику од програма који се инсталирају на рачунару, интернет апликацијама корисник приступа директно кроз интернет прегледач. На овај начин се избегава трошење времена на инсталацију апликације и чува меморијски простор који би иначе та апликација заузела. Интернет апликације су са клијентске стране искључиво везане за интернет прегледач, што их чини вишеплатформским (поседују могућност рада на свим оперативним системима који подржавају интернет прегледач). На овај начин се избегава развој исте апликације за различите оперативне системе и уређаје.

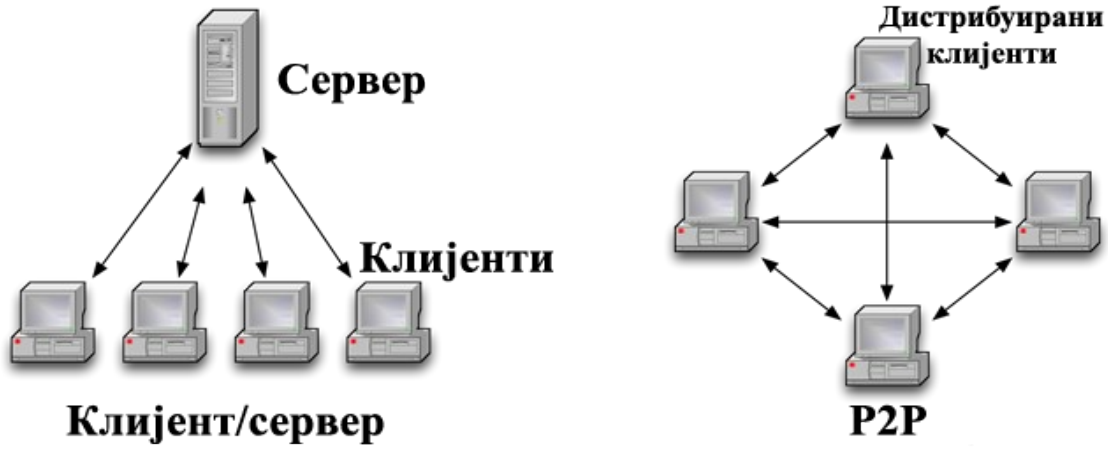
Да би интернет апликација функционисала, потребан је веб (интернет) сервер, апликацијски сервер и база података. Веб сервери управљају захтевима које клијенти шаљу, а апликацијски сервер обавља прослеђене задатке. База података се користи за складиштење података од значаја за апликацију.

Постоје и хибридне апликације које се инсталирају на уређају, односно оперативном систему, а понашају се као интернет апликације. Овакве апликације могу користити ресурсе које нуди дати систем, а који су иначе ограничени или нису доступни интернет апликацијама. Оне кумуницирају са интернет серверима путем *API*-ја (енг. *Application Programmng Interface* – апликациони програмски интерфејс), стандардизованих правила под којим апликације комуницирају са сервером[3].

Највећа мана хибридних и интернет апликација је њихова зависност од интернета, без којег губе сву своју функционалност.

2.2.1 Клијентско-серверска архитектура

У клијентско-серверској архитектури постији увек доступан рачунар, назван **сервер**, чије услуге тражи много других рачунара, који се називају **клијенти**. Рачунари клијенти могу бити повремено или стално укључени. Када од рачунара клијента прими захтев за одређени објекат, веб сервер одговара слањем траженог објекта до рачунара клијента. У овој архитектури клијенти не комуницирају непосредно један са другим, већ користе веб сервер за посредну комуникацију. Важна карактеристика сервера је да увек има постојану и добро познату *IP* (енг. *internet protocol*) адресу преко које клијенти у сваком тренутку могу да ступе у контакт са сервером. Архитектура супротна клијентско-серверској је *P2P* (енг. *peer-to-peer*) архитектура, код које клијенти међусобно непосредно комуницирају (слика 2).



Слика 2: *Клијент-сервер и P2P архитектура*

Клијент-сервер архитектура је централизованог типа, а *P2P* архитектура је дистрибуираног типа. Клијент-сервер има одређене предности у односу на *P2P*, али и одређене мане.

Често се дешава да један сервер не може да се избори са свим захтевима својих клијената (најчешће код апликација попут друштвених мрежа и апликација за дистрибуирање електронског садржаја), па се у тим ситуацијама поставља више сервера са заједничким задатком дистрибуирања података корисницима. Групе сервера се постављају у заједничке просторије (тзв. „серверске фарме") у којима се строго прате и контролишу радни услови. Мноштво сервера на истом месту са истом наменом чине један моћни виртуелни сервер[4].

У овом раду постоји потреба само за једним сервером који се виртуелно одржава на истом рачунару са клијентом.

3. Коришћене технологије и радна окружења

Коришћене технологије, језици и библиотеке за реализацију овог задатка су следеће:

- HTML
- CSS (MaterializeCSS)
- PHP
- Laravel
- XAMPP
  - Apache
  - MySQL
- Microsoft Visual Code
- Google Chrome

3.1 HTML

HTML (енг. *Hyper Text Markup Language*) је стандардни језик за означавање садржаја који се користи за израду интернет страница. Он описује структуру и садржај користећи се низом елемената које интернет претраживач користи за форматирање и прављење страница. Ови елементи су описани **ознакама** (енг. *tags*), а ознаке претраживач користи као инструкције и не приказује их кориснику на готовој страници. *HTML* ознаке чине знак **мање** (<), кључна реч и на крају знак **веће** (>). Већина ознака се мора употребљавати у пару, где прва ознака отвара текстуално окружење а друга ознака га затвара, при чему друга ознака пре кључне речи садржи **косу црту** (/) како би се разликовала од отварајуће ознаке.

```
<ознака>садржај иде овде...</ознака>
```

Прилог 1: *Пример једне HTML ознаке*

HTML пружа основне команде за приказ и уређивање страница, а за додатну контролу изгледа и понашања странице се ослања на остале језике који се увозе у страницу преко ознака **<style>** и **<script**>.

Ознаке имају вишеструку примену, а главне примене су:

- структурно означавање
- презентационо означавање
- семантичко означавање
- означавање веза ка другим документима

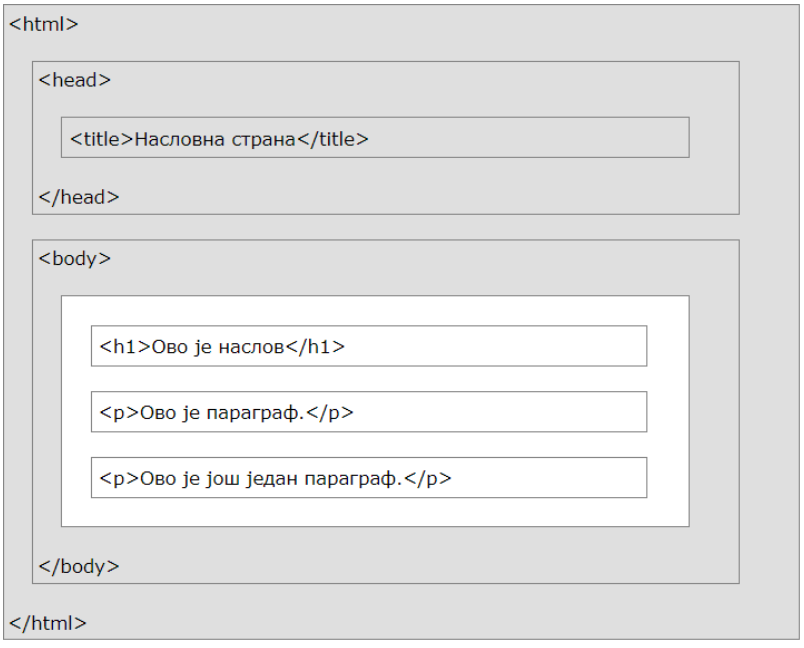
Структурно означавање се користи за прављење нивоа организације текста, примери ознака су <h1>,</h1>, које служе да означе наслов првог нивоа неке странице. Остали тагови служе за прављење наслова нижег нивоа, параграфа и мањих одељака текста.

Презентационо означавање се користи за издвајање одређених елемената из групе елемената, као што су одређене речи у реченици. Пример ознака за ту сврху су и , који респективно подебљавају и праве курзив слова.

Семантичко означавање је слично презентационом, али има више семантичку природу, и користи се за означавање битних речи у тексту. Стил ових измена се диктира у *CSS*-у и зато немају једноличан визуелни идентитет. Главне ознаке су и .

Означавање веза ка другим документима се врши коришћењем ознаке <a>. Ово је прва ознака која уводи динамику у интернет странице и служи као пречица ка осталим документима. Ова ознака прима атрибут који користи као путању на коју прослеђује корисника након што кликне на елемент обележен ознаком[5].

Аргументи ознака се пишу редом један за другим, а представљају уређени пар **назив_аргумента = “вредност_аргумента”**.



Слика 3: *Приказ структуре HTML документа*

3.2 CSS

CSS (*cascading style sheets*) је језик за обележавање семантике презентације *HTML* докумената и њихових појединачних елемената. Настао је из потребе да се створи већа моћ уређивања изгледа интернет страница. Омогућено је дефинисање поставки изгледа у засебној датотеци која носи наставак *.css*, а у *HTML* заглављу документа је потребно само позвати ту датотеку коришћењем ознака <link href=”датотека.css” rel=”stylesheet” type=”text/css”> . Коришћењем готових *CSS* решења се скраћује време потребно за израду визуелно богате и лепе апликације са стандардизованим изгледом без знања графичког дизајна[6].

Синтакса се разликује од *HTML* синтаксе, а ниједан текстуални елемент *CSS* изворног кода се не приказује кориснику, већ само служи као низ правила за украшавање *HTML* грађе. Изглед жељеног елемента се дефинише редом: име елемента, **отворена витичаста заграда** ({), парови својстава и вредности раздвојени **двотачком** (:) а који се завршавају **тачка-зарезом** (;) и **затворена витичаста заграда** (}) која затвара целину. Пример дефинисања плавог наслова првог реда величине фонта 12 пиксела је дат у слици 4[7].

Слика 4: *Изглед декларације изгледа елемента h1*

Развојем *CSS* скрипти се баве углавном дизајнери који након тога готова решења излажу на интернету у оквиру *CSS* програмских оквира да се њима служе веб програмери. Самостални елементи у тим библиотекама су бесплатни за коришћење, а целокупне странице са уређеним шаблонима се могу преузети уз одговарајућу новчану накнаду. Постоји одређен број популарних програмских оквира који се широко користе за израду интернет апликација, а за израду апликације из овог рада је коришћен *MaterializeCSS.*

3.3 MaterializeCSS

MaterializeCSS је клијентски (енг. *front end)* програмски оквир намењен олакшавању прављења корисничких страница приликом развоја интернет апликација.

Заснован је на материјалном дизајну који је у мејнстрим токове увела корпорација Гугл 2014. године. Одликује се равним (енг. *flat*) изгледом елемената са оштрим ивицама који чине страницу. Гуглов дизајнер *Matias Duarte* задужен за развој материјалног дизајна је објаснио да „упркос правом папиру, дигитални материјал може да се проширује и мења паметним путем. Материјал има физичке површине и ивице. Шавови и сенке пружају интуитивно значење о ономе што можете да додирнете."

Гугл је масовним коришћењем овог дизајна у својим апликацијама успоставио стандард који су и остали произвођачи почели да имплементирају или копирају. Кроз *API* токове, Гугл је дозволио прављење фрејмворка који имплементирају њихов дизајн.

Један од тих фрејмворка је и *Materialize*, који је коришћен у изради овог пројектног задатка.

Materialize у оквиру себе пружа кориснику приступ готовим елементима спремним за имплементацију:

- *CSS* датотеке са предодређеним изгледом
- компоненте
- *JavaScript* датотеке
- форме

Инсталација се може обавити на више начина:

- путем *CDN*-а (*content delivery network*)
- директним ручним преузимањем датотека

Да би се садржај учинио доступним унутар *HTML* документа, потребно га је навести унутар одговарајућих *HTML* елемената (прилог 2).

```
<link type="text/css" rel="stylesheet" href="css/materialize.min.css">

<script type="text/javascript" src="js/materialize.min.js"></script>
```

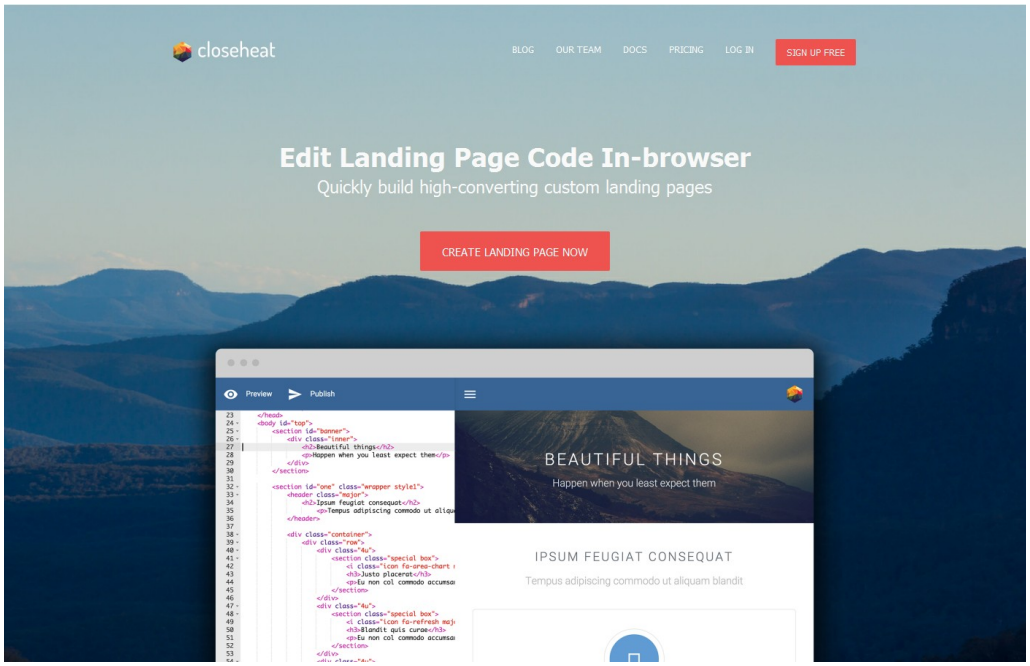
Прилог 2: *Позивање .css и .js датотека у документ*

Позивање елемената се врши уписивањем кључних речи унутар атрибута ***class*** жељених *HTML* ознака. *JavaScript* датотека се позиционира на крају документа да би се добило на брзини учитавања интернет странице[8].

Посебно олакшање позиционирања компоненти на корисничком приказу пружа мрежа (*grid*) која се састоји од 12 колона. Те колоне служе за распоређивање садржаја по ширини приказа у зависности од типа екрана уређаја који корисник користи за приступ интернет страници. Већина уређаја су рачунари, таблети, паметни телефони и паметни телевизори, па је зато битно да оријентација екрана не поквари корисничко искуство (*user experience*) и не искључи неке од података са екрана. У сврху вишеплатформског коришћења интернет страница, колоне које садрже одређене елементе се могу сврстати у:

- *.s* (*small* - мале)
- *.m* (medium – средње)
- *.l* (*large* – велике)
- *.xl* (*extra large* – највеће)

Да би се допринело изгледу и лакоћи коришћења, радни оквири обично садрже и *JavaScript* датотеке које су круна динамичности интернет странице. У овом раду није посвећен посебан одељак језику *JavaScript* јер је врло мало заступљен кроз стандардизоване форме.



Слика 5: *Изглед страниице дизазјниране уз помоћ Materialize програмског оквира*

3.4 PHP

PHP (*PHP:Hypertext Preprocessor*) је програмски језик опште намене који углавном служи у развоју интернет апликација. Настао је 1994. као лични пројекат Расмуса Лердорфа али је убрзо прешао под надлежност *PHP* групе (*The PHP Group*).

PHP ради из командне линије (енг. *CLI - command line interface*), може се писати директно унутар *HTML* кода или уз коришћење бекенд (*backend*) програмских оквира, као што је у овом случају *Laravel*. Поседује засебни интерпретатор имплементиран као модул на веб серверу. Веб сервер приказује резултате извршеног *PHP* кода који своје резултате убацује у HTML код (прилог 3).

Стандардни *PHP* преводилац је бесплатан и дистрибуира се под *PHP* лиценцом. Светски је распрострањен и користи се на многим веб серверима, а подржавају га сви оперативни системи и платформе. Еволуирао је без писане формалне спецификације и стандарда све до 2014. године, а тада је започет рад на стварању формалне спецификације.

До септембра 2019. године преко 60% интернет страница које користе *PHP* и даље користе застареле верзије 5.6 или ниже. Иако *PHP* група не пружа званичну подршку за верзије старије од 7.1, постоје треће стране попут одређених фондација које и даље пружају поменуту подршку[9].

На серверу, било који *PHP* код у траженој датотеци се извршава током рада и притом креира динамички садржај веб страница или динамичке слике на веб локацијама. Његов потенцијал се испољава и приликом прављења и извршавања скрипти на командној линији, као и приликом израде апликација са графичким корисничким интерфејсом (енг. *GUI – graphical user interface*) на страни клијента. Популаран је због своје једноставне имплементације са системима за управљање релационим базама података (*RDBMS – Relational DataBase Management System*).

Доступан је за коришћење бесплатно уз комплетан изворни код што омогућава корисницима широку слободу за прилагођавање сопственим пројектима. Екстензија *PHP* датотека је *.php.*

PHP наредбе нису осетљиве на мала и велика слова, зато ће се код из прилога 3 извршити и неће пријавити грешку. Насупрот тога, називи промењљивих јесу осетљиви на мала и велика слова. Такође постоје резервисани називи који се не смеју давати као имена промењљивама. При креирању промењљивих није потребно декларисати врсту промењљиве али је у последњим верзијама (7.0+) то промењено.

```
<!DOCTYPE html>
<html>
<body>

<?php
ECHO "Hello World!<br>";
echo "Hello World!<br>";
EcHo "Hello World!<br>";
?>

</body>
</html>
```

Прилог 3: *Пример уграђивања PHP кода у HTML код*

3.5 Laravel

Laravel је бесплатни *PHP* веб оквир (*framework*) отвореног кода који је направио *Taylor Otwell* и предвиђен је за развој веб апликација које се заснивају на принципма архитектонског обрасца модел-приказ-контролер (енг. *MVC - Model-View-Controller*). Неке од карактеристика Ларавела су модуларни систем структуирања и паковања ресурса са уграђеним менаџером за управљање зависностима (*Dependency Manager*), могућност различитих начина приступа релационим базама података, помоћни програми који помажу у активирању и одржавању апликација његова оријентација ка имплементирању најједноставније и најинтуитивније могуће синтаксе програмског језика. Изворни код Ларавела налази се на *GitHub*-у и лиценциран је MIT лиценцом[10]. Кључне карактеристике Ларавела су:

- *Bundles* пакети пружају модуларни систем паковања од верзије Ларавела 3.0, а у пакетима су већ доступне функције за једноставно додавање додатних функционалности апликацијама. Поред тога, од верзије Ларавела 4.0 па надаље, користи се *Composer* као менаџер за управљање пакетима (*Dependency Manager*) помоћу којег се додају *PHP* пакети доступни у *Packagist* репозиторијуму.
- *Eloquent ORM* (*object-relational mapping* - објектно-релацијско пресликавање) је напредна PHP имплементација обрасца активног снимања (*active record pattern*), која истовремено нуди уграђене методе за постављање ограничења на односе између објеката базе података. Према принципима обрасца активног снимања,

Eloquent ORM приказује табеле из базе података као класе, а инстанце њиховог објекта везане су за појединачне редове тих табела.

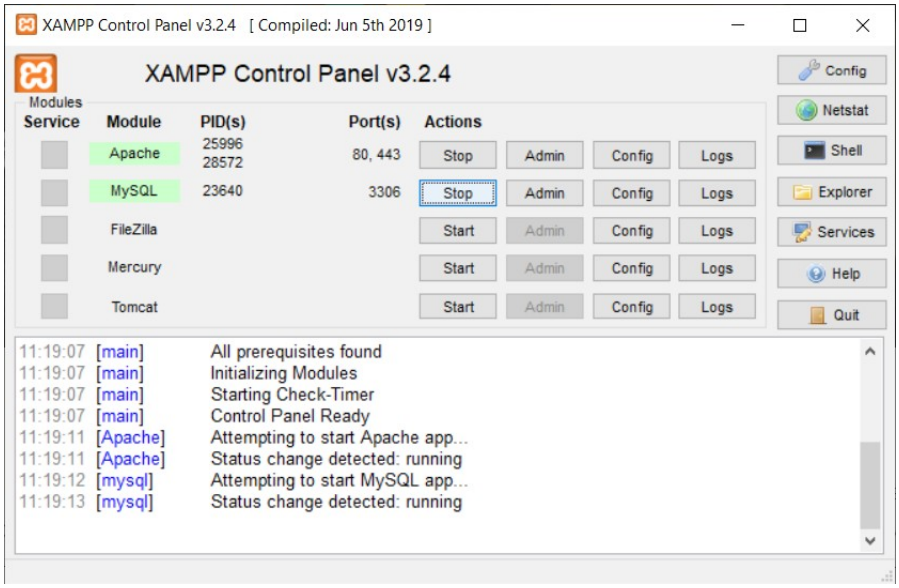
- *Query builder* (градитељ упита) доступан од верзије Ларавела 3.0, пружа директнију алтернативу приступу базама података Eloquent ORM-у. Уместо да захтева директно писање SQL упита, Ларавелов "градитељ упита" пружа скуп класа и метода који могу програмски да креирају SQL упите. Такође омогућава селективно кеширање резултата извршених упита.
- Логика апликације је саставни део развијених апликација, имплементираних било коришћењем контролера или као део декларације руте.
- *Reverse routing* (обрнуто усмеравање) дефинише однос између веза (линкова) и рута, омогућавајући тако да се евентуалне касније промене рута аутоматски изврше и над одговарајућим линковима. Када се линкови створе помоћу имена постојећих рута, Ларавел аутоматски прави одговарајуће јединствене идентификаторе ресурса (*Uniform resource identifiers – URI*).
- *Restful controllers* (контролери засновани на принципима *REST* модела), пружају опциони начин за одвајање позадинске логике која служи за опслуживање *HTTP GET* и *POST* захтева.
- Аутоматско учитавање класе омогућава аутоматско учитавање PHP класа без потребе за ручним одржавањем путања. Учитавање на захтев спречава укључивање непотребних компоненти, тако да се учитавају само компоненте које се заправо користе.
- *View composers* (композитори приказа) служе као прилагодљиве јединице логичког кода које се могу извршити када се учита приказ.
- *Blade templating engine* (погон блејд шаблона) комбинује један или више образаца са моделом података како би се добили одговарајући прикази, тако што преводи образац у кеширани PHP код и тиме омогућава побољшане перформансе. *Blade* такође пружа скуп сопствених контролних структура као што су *if* услови и петље, које се интерно мапирају на одговарајуће *PHP* делове кода. Такође, Ларавел услуге могу да се позивају из *Blade* шаблона, а сам уређај за обраду шаблона може да се прошири кориснички дефинисаним директивама.
- *IoC containers* (иоц контејнери) омогућавају стварање нових објеката применом принципа инверзије контроле (*Inversion of Control*), у ком *framework* позива код који се односи на извршавање апликације или задатка, са опционим инстанцирањем и референцирањем нових објеката на начин који прати *singleton* образац.
- *Migrations* (миграције) пружају систем контроле верзија за шеме база података, што омогућава једноставно извршавање промена у кодној бази апликације и

потребних промена у изгледу базе података. Као резултат, ова функција поједностављује употребу и ажурирање апликација заснованих на Ларавелу.

- *Database seeding* омогућава начин попуњавања табела база података одабраним подразумеваним подацима који се могу користити за тестирање апликација или се изводе као део почетног подешавања апликације.
- *Unit testing* (модуларно тестирање) обезбеђено је као саставни део Ларавела, који сам по себи садржи модуларне тестове који откривају и спречавају регресију у оквиру самог *framework*-а. Модуларни тестови могу се изводити кроз обезбеђени алат за командну линију.
- Аутоматско страничење (енг. *pagination*) поједностављује задатак имплементације пагинације, замењујући уобичајене ручне приступе са имплементацијом аутоматизованим методама интегрисаним у Ларавел.

3.6 XAMPP

Ксамп (енг. *XAMPP*) је бесплатни вишеплатформски веб стек (енг. *web stack*) пакет које развија удружење *Apache Friends*. Главне компоненте ксампа (од чијих се почетних слова и створио и назив *XAMPP* као акроним) су *Apache HTTP* сервер, *MariaDB* база података и интерпретатори за програмске језике *PHP* и *Perl*. Све ове компоненте су потребне за одржавање реалног сервера, тако да је врло лако пренети пројекат из пробног окружења на реални сервер. Ксамп је вишеплатформски и одлкује га једноставна инсталација на било ком оперативном систему. Садржи једноставан интерфејс за контролу својих сервиса (слика 6).



***Слика 6**: XAMPP контролни интерфејс*

3.6.1 Apache HTTP сервер

Један од кључних делова X*AMPP* пакета је *Apache* (*Apache HTTP Server*) који се користи за покретање веб сервера на *Windows* оперативном систему. Покретањем локалног *Apache* веб сервера на свом рачунару, веб програмер може тестирати веб странице у веб претраживачу без објављивања истих на Интернет. Пројекат *Apache HTTP Server* је настао као производ заједничког рада на развоју софтвера који има за циљ стварање робусне, комерцијалне, функционалне и доступне имплементације отвореног изворног кода *HTTP* (веб) сервера. Пројектом заједнички управља група волонтера широм света, који користе Интернет за комуникацију, планирање и развој сервера као и креирање пратеће документације. Овај пројекат је део *Apache* софтверске фондације. Поред тога, стотине волонтера константно доприносе писању кода и пројектне документације као што је и случај са већином пројеката отвореног кода.

3.6.2 MariaDB

MariaDB је систем за управљање релационим базама података (енг. *RDBMS – Relational DataBase Management System*). Представља директну копију *MySQL*-а. *Maria* и *My* су имена ћерки креатора оба система, *Michaela Wideniusa.* До одвајања је дошло након што је корпорацију која је у власништву држала *MySQL* купила приватна фирма *Sun Microsystems* (данашња фирма *Oracle*). Главни програмери који су радили у то време на пројекту *MySQL* су копирали програмски код (енг. *fork*) и наставили засебно да развијају своју верзију копије. Своје поступке су објаснили као бојазан од гашења или успоравања пројекта, јер су чести случајеви да фирме купују своју конкуренцију и након тога је угасе.

MariaDB и *MySQL* имају идентичну синтаксу тако да нема разлике у коришћењу ова два система. Како време пролази, разлике између њих се продубљују иако се програмери заздужени за развој *MariaDB* труде да у систему одрже заједничке особине рада са *MySQL*-ом.

MariaDB је бесплатни софтвер који је издат под условима *GNU General Public Licence*, а доступан је и под разним власничким лиценцама. Овај систем користе многе веб апликације које су засноване на коришћењу база података, укључујући *Drupal, Joomla*, *phpBB* и *WordPress*. Такође га користе и многи популарни веб-сајтови, међу којима су *Facebook, Flicker, Twitter,* и *YouTube.*

MariaDB је написан у *C* и *C++* програмским језицима. Ради на многим платформама, укључујући *AIX*, *BSDi*, *FreeBSD*, *HP-UX*, *IRIX*, *Linux*, *macOS*, *Microsoft Windows*, *NetBSD*, *OpenBSD*, *OpenSolaris*, *Oracle Solaris*, *Simbian*, *SunOS*. У *MariaDB* софтверу постоји и порт за конфигурисање *OpenVMS* виртуалног оперативног система.

MySQL софтвер, као и клијентске библиотеке, користи дистрибуцију двоструког лиценцирања. Нуди се под *GPL* лиценцом верзије 2, или власничком лиценцом. Подршка се може добити преко званичног приручника. Бесплатна подршка је такође доступна на различитим *IRC* (*Internet Relay Chat*) каналима и форумима. Велике корпорације, међу којима је и Гугл, су послале своје програмере да учествују у развоју система.

MariaDB је добио позитивне критике и рецензенти су приметили да „изузетно добро функционише у просечном случају“ и да „развојни интерфејси постоје (попут *phpMyAdmin* окружења), а документација је веома добро написана". Такође је тестирано да је „брз, стабилан и прави вишекориснички сервер са више слојева *SQL* базе података“.

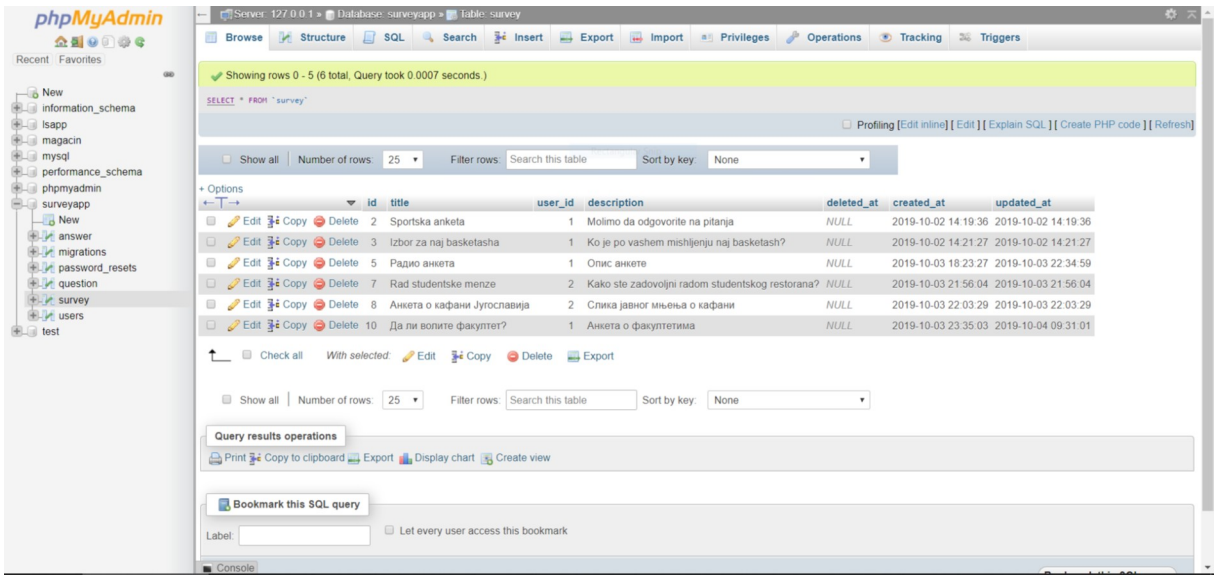
3.6.3 phpMyAdmin

У склопу *XAMPP*-а се налази и *PhpMyAdmin*, графички интерфејс за комуникацију са базом података. *PhpMyAdmin* је бесплатни софтвер написан у *PHP*-у, намењен за управљање администрацијом *MariaDB*/*MySQL* базе података преко Интернета.

Подржава широк спектар операција над *MariaDB* и *MySQL* системима за управљање базама података. Операције које се често користе (управљање базама података, табелама, колонама, релацијама, индексима, корисницима, дозволама итд.) могу се изводити преко корисничког интерфејса, а поред тога постоји могућност и да се директно извршава било који *SQL* упит у базу.

PhpMyAdmin долази уз обимну документацију, а његови корисници су подстакнути да ажурирају документацију на свом матерњем језку (уколико није енглески) како би разменили идеје и начине рада за разне операције. *PhpMyAdmin* је такође врло дубоко документован у књизи “*Mastering phpMyAdmin for Effective MySQL Management*”, која је доступна на енглеском језику. Да би се олакшала употреба широком кругу људи, *PhpMyAdmin* се преводи на 72 језика и подржава *LTR* (*Left-to-right*) и *RTL* (*Right-to-left*) језике. *PhpMyAdmin* је успешан пројекат са стабилном и флексибилном базом кодова. Пројекат *PhpMyAdmin* је и део организације *Software Freedom Conservancy* (*SFC*). *SFC* је непрофитна организација која помаже у промовисању, унапређењу, развоју и заступању *Free, Libre and Open-Source* (*FLOSS*) пројеката.

Графички кориснички интерфејс је приказан на слици 7.

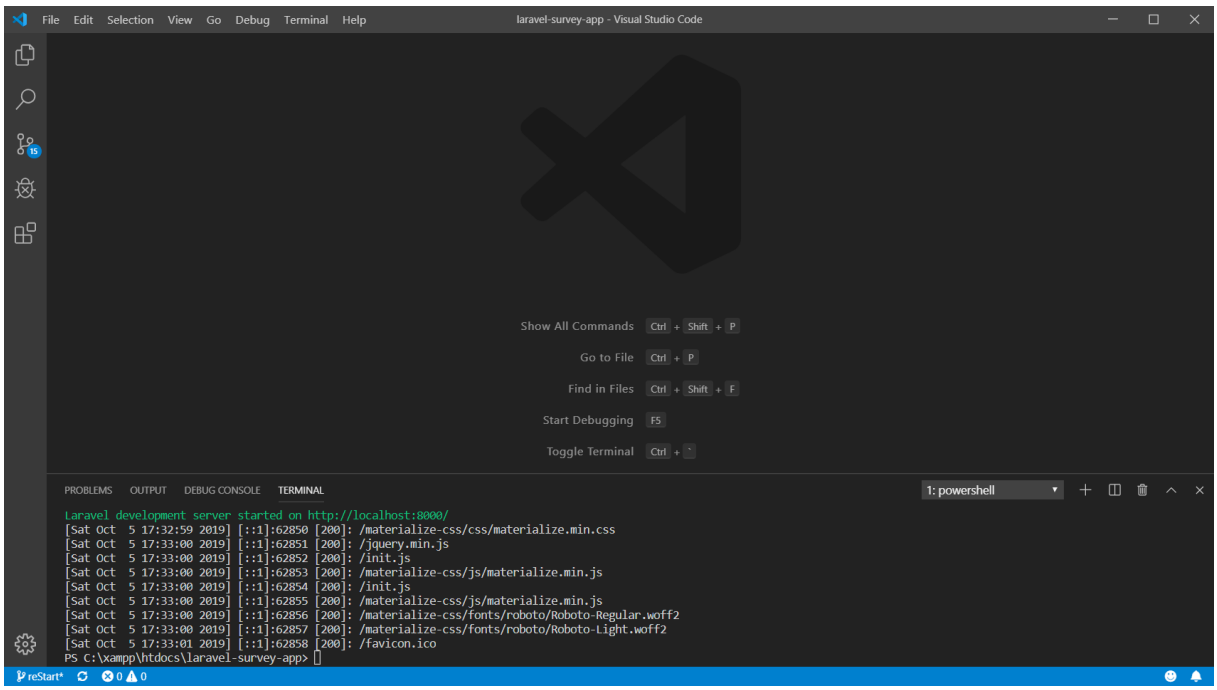


Слика 7: *Изглед графичког корисничког интерфејса phpMyAdmin-а*

3.7 Visual Studio Code

Visual Studio Code је уређивач изворног кода (*source-code editor*) који је развила корпорација *Microsoft* за *Windows*, *Linux* и *macOS* оперативне системе. Пружа подршку за уклањање грешака, уграђену *Git* и *GitHub* контролу развојне верзије, истицање синтаксе, паметно довршавање кода, шаблоне и рефакторисање кода. Веома је прилагодљив, омогућава корисницима да мењају тему интерфејса, пречице на тастатури, поставке и инсталирају додатке који доприносе функционалности програма. Изворни код је бесплатан и отворен, а издаје се под *MIT* лиценцом. *Visual Studio Code* заснован је на *Electron*-у, програмском оквиру који се користи за прављење *Node.js* апликација, чије се функционисање заснива на *Blink Layout Engine* прегледачком софтверском погону.

У анкети за програмере на форуму популарног интернет сајта за програмере *Stack Overflow,* 2019. године, *Visual Studio Code* је рангиран као најпопуларније окружење за програмере, а 50,7% од 87.317 испитаника је тврдило да га користи. *Visual Studio Code* је уређивач изворног кода који се може користити са различитим програмским језицима. Уместо пројектног система омогућава корисницима отварање једног или више директоријума, који се затим могу организовати у радне просторе за будућу употребу. То му омогућава да ради као општи уређивач кода за било који језик, супротно *Miscrosoft Visual Studio* окружењу које користи *.sln (*скраћено од *solution)* фајлове и пројектне фајлове који су стриктно везани за дати пројекат.

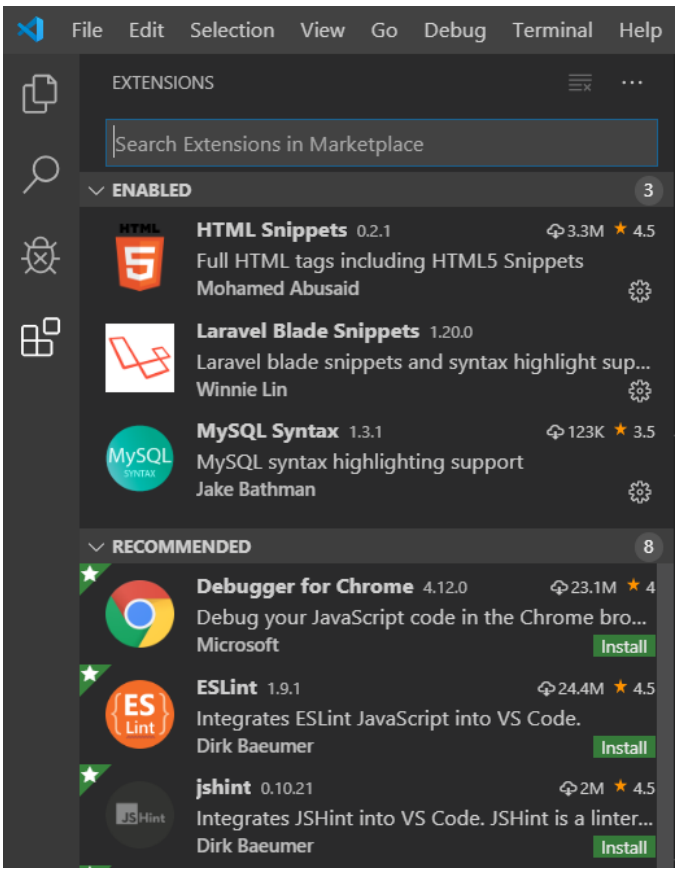


***Слика 8**: Графички кориснички интерфејс Visual Studio Code-а*

На слици 8 је представљен интерфејс *Visual Studio Code*-а, са основном црном темом и покренутим *powershell* терминалом у дну прозора .

Visual Studio Code укључује вишеструка проширења за протокол слања датотека (*FTP - File Transfer Protocol*), омогућавајући да се софтвер користи као бесплатна алтернатива за веб развој. Код се може синхронизовати између уређивача и сервера, без преузимања додатног софтвера. Такође омогућава корисницима да поставе кодну страницу на којој је сачуван активни документ, знак нове линије за *Windows*/*Linux* и програмски језик активног документа. То му омогућава да се користи на било којој платформи, на било којем локалитету и за било који одређени програмски језик.

Подржава бројне програмске језике и скуп функција које се разликују у зависности од тренутно коришћеног програмског језика. Нежељене датотеке и мапе могу се искључити из стабла пројекта путем поставки. Многе функције *Visual Studio Code*-а нису изложене кроз мениje или корисничко окружење, већ им се може приступити путем палете наредби. Visual Studio Code може да се прошири помоћу додатака који су доступни и могу да се преузму из централног репозиторијума. Ово укључује додатке за уређивач и подршке за програмске језике. Значајна карактеристика је могућност креирања додатака који додају подршку за нове језике, теме, исправљаче грешака као и извршавање статичке анализе кода (слика 9).



***Слика 9**: Прозор за избор додатних пакета*

3.8 Google Chrome

Гугл Хром (*Google Chrome*) је вишеплатформски интернет претраживач, а произвела га је и одржава корпорација Гугл (*Google*). Први пут се појавио 2008. године на оперативном систему *Windows*, а након тога је почео да се шири и на остале системе (*Linux, macOS, iOS, Android*). Представља главну компоненту оперативног система који развија Гугл, *ChromeOS*-а.

Највећи део изворног кода потиче од пројекта C*hromium,* који је отвореног кода. Првобитно је за погон коришћен *WebKit,* а тренутно се у свим верзијама *Chrome*-а осм у верзији за *iOS* користи *Blink*.

У проценама направљених лета 2019. се тврди да *Chrome* заузима 71% удела светског тржишта интернет прегледача на личним рачунарима (*PC*), а 63,34% удела прегледача на свим платформама заједно.

Главни разлози за коришћење овог претраживача су брзина и поузданост. Покретањем *Apache* сервера и уносом *localhost:8000* у поље за претрагу, апликација се покреће.

4. Прављење базе података

Комуникација са базом података се одвија целокупно на серверској страни и не тиче се корисника, јер он само прослеђује податке серверу који онда барата њима. Интеракцију са базом података сервер врши коришћењем јасно дефинисане *SQL* синтаксе. На тај начин се извршавају све наредбе, од дефинисања и манипулације подацима, преко репликације целокупних база и табела па све до контроле приступа.

За приступ бази су потребни подаци за пријављивање администратора (корисничко име и лозинка), који су у овом случају '*root'* и ''(празно поље).

Коришћењем *Laravel*-а, корисник у датотеци под називом *.env* (*environment* – окружење). У овој датотеци се налазе најважнији параметри који служе апликацији за коришћење ресурса на основу информација пружених у оквиру ње. Потребно је подесити шест параметара да би апликација могла приступити бази података:

- DB_CONNECTION = mysql
- DB_HOST = 127.0.0.1
- DB_PORT = 3306
- DB_DATABASE = surveyapp
- DB_USERNAME = root
- DB_PASSWORD =

Поред поменутих параметара за пријаву, потребно је дефинисати тип базе података (*mysql*/*mariadb*), назив табеле и подесити *IP* адресу и *port*.

Једина директна команда коју корисник мора саопштити *mysql*-у је прављење базе података. Ово се може извршити преко командне линије, или у овом случају преко графичког интерфејса *phpMyAdmin.* Прављење базе је крајње интуитивно коришћењем интерфејса, а одговарајућа *SQL* наредба је:

```
CREATE DATABASE surveyapp
```

Прилог 4: *програмски позив за прављење базе података*

Прављење табела се може извршавати писањем *SQL* наредби кроз командну линију, коришћењем *phpMyAdmin* интерфејса или дефинисањем наредби унутар *PHP* датотека.

Најефикаснији начин је коришћењем *Eloquent ORM*-а. Преко њега се табеле у бази података представљају као класе, а сваки ред у поменутим табелама се представља као инстанца класе. Све класе табела наслеђују класу *Migration* да би могле да користе готове методе. Свака класа табеле садржи две методе: *up()* и *down*(). Ове методе се позивају када се класа инстанцира и брише. За све класе, функција брисања има исти изглед, тако да неће бити понављана за сваку класу посебно:

```
public function down(){ Schema::drop('naziv_tabele'); }
```

***Прилог 5**: функција за брисање табеле*

У даљем тексту је дат код за прављење сваке табеле унутар функције *up()* уз еквивалентни *SQL* код:

```
Schema::create('users', function (Blueprint $table) {
            $table->increments('id');
            $table->string('name');
            $table->string('email')->unique();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
```

```
CREATE TABLE users (
 id int(10) unsigned NOT NULL AUTO_INCREMENT,
 name varchar(255) COLLATE utf8_unicode_ci NOT NULL,
 email varchar(255) COLLATE utf8_unicode_ci NOT NULL,
 password varchar(255) COLLATE utf8_unicode_ci NOT NULL,
 remember_token varchar(100) COLLATE utf8_unicode_ci DEFAULT NULL,
 created_at timestamp NULL DEFAULT NULL,
 updated_at timestamp NULL DEFAULT NULL,
 PRIMARY KEY (id),
 UNIQUE KEY users_email_unique (email)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci
```

***Прилог 6:** Прављење табеле users*

```
Schema::create('survey', function (Blueprint $table) {
            $table->increments('id');
            $table->string('title');
            $table->integer('user_id')->unsigned()->index();
            $table->string('description');
            $table->softDeletes();
            $table->timestamps();
        });
```

```
CREATE TABLE survey (
 id int(10) unsigned NOT NULL AUTO_INCREMENT,
 title varchar(255) COLLATE utf8_unicode_ci NOT NULL,
 user_id int(10) unsigned NOT NULL,
 description varchar(255) COLLATE utf8_unicode_ci NOT NULL,
 deleted_at timestamp NULL DEFAULT NULL,
 created_at timestamp NULL DEFAULT NULL,
 updated_at timestamp NULL DEFAULT NULL,
 PRIMARY KEY (id),
 KEY survey_user_id_index (user_id)
) ENGINE=InnoDB AUTO_INCREMENT=13 DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci
```

***Прилог 7:** Прављење табеле survey*

```
Schema::create('question', function (Blueprint $table) {
            $table->increments('id');
            $table->integer('survey_id')->unsigned()->index();
            $table->integer('user_id')->unsigned()->index();
            $table->string('title');
            $table->string('question_type');
            $table->string('option_name')->nullable();
            $table->timestamps();
        });
```

```
CREATE TABLE question (
 id int(10) unsigned NOT NULL AUTO_INCREMENT,
 survey_id int(10) unsigned NOT NULL,
 user_id int(10) unsigned NOT NULL,
 title varchar(255) COLLATE utf8_unicode_ci NOT NULL,
 question_type varchar(255) COLLATE utf8_unicode_ci NOT NULL,
 option_name varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
 created_at timestamp NULL DEFAULT NULL,
 updated_at timestamp NULL DEFAULT NULL,
 PRIMARY KEY (id),
 KEY question_survey_id_index (survey_id),
 KEY question_user_id_index (user_id)
) ENGINE=InnoDB AUTO_INCREMENT=29 DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci
```

Прилог 8: *Прављење табеле question*

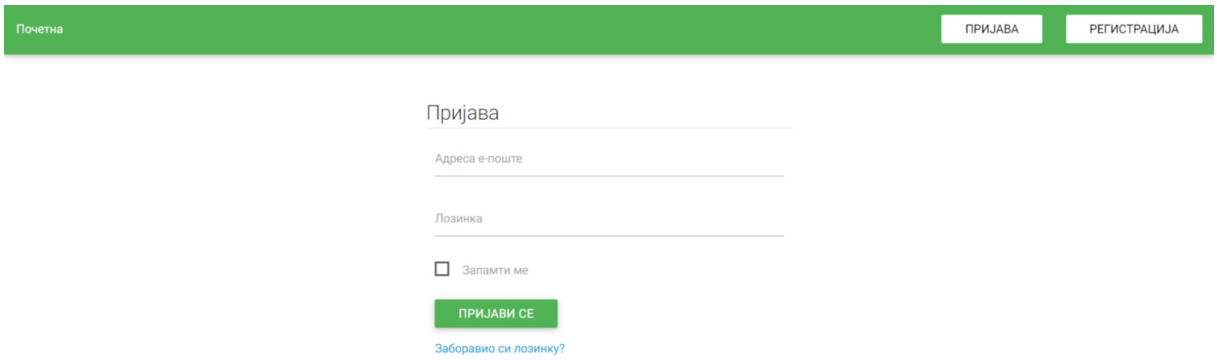
```
Schema::create('Answer', function (Blueprint $table) {
            $table->increments('id');
            $table->integer('user_id');
            $table->integer('question_id');
            $table->integer('survey_id');
            $table->string('answer');
            $table->timestamps();
        });
```

```
CREATE TABLE answer (
 id int(10) unsigned NOT NULL AUTO_INCREMENT,
 user_id int(11) NOT NULL,
 question_id int(11) NOT NULL,
 survey_id int(11) NOT NULL,
 answer varchar(255) COLLATE utf8_unicode_ci NOT NULL,
 created_at timestamp NULL DEFAULT NULL,
 updated_at timestamp NULL DEFAULT NULL,
 PRIMARY KEY (id)
) ENGINE=InnoDB AUTO_INCREMENT=56 DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci
```

Прилог 9: *Прављење табеле answer*

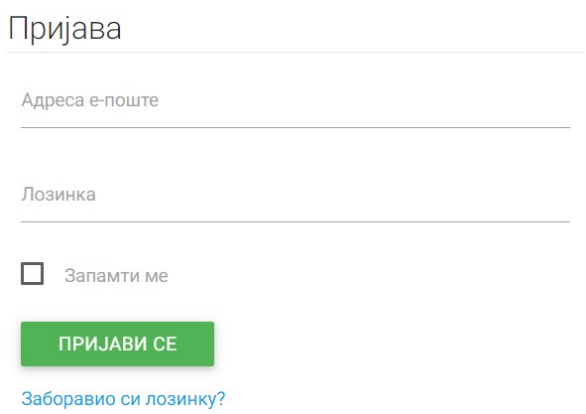
5. Софтверско решење

Почетна страница апликације води корисника директно до форме за пријављивање (слика 10). Заједнички елемент за све странице је заглавље које садржи непроменљиво дугме („**Почетна**”) које води на почетну страницу и дугмад за пријаву („**Пријава**”) и регистрацију („**Регистрација**”). Уколико корисник жели да заобиђе пријаву и покуша да унесе у *URL* адресу назив странице са актуелним анкетама, биће преусмерен на страницу за пријављивање. На тај начин се спречава неовлашћен приступ нерегистрованих корисника систему. На сликама у тексту ће бити поступно приказани релевантни елементи странице без заједничких елемената свих страница ради уштеде простора и бољег квалитета слика.

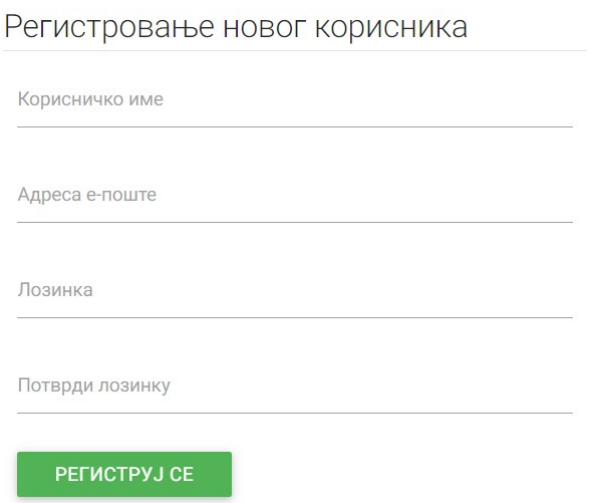


Слика 10: *Почетна страница*

У зависности од постојања корисничког налога, корисници се могу пријавити или регистровати у систем избором једног од дугмића у горњем левом углу. Притиском на одговарајуће дугме кориснику се отвара одговарајућа форма (слике 11 и 12).

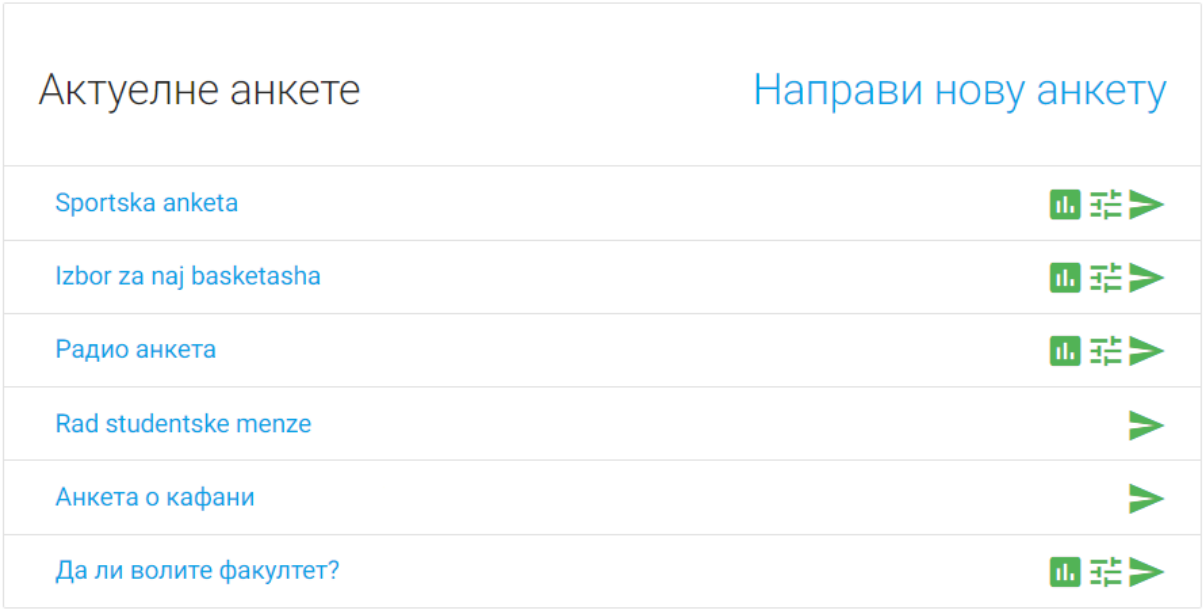


***Слика 11**: Пријава корисника*



***Слика 12:** Регистрација корисника*

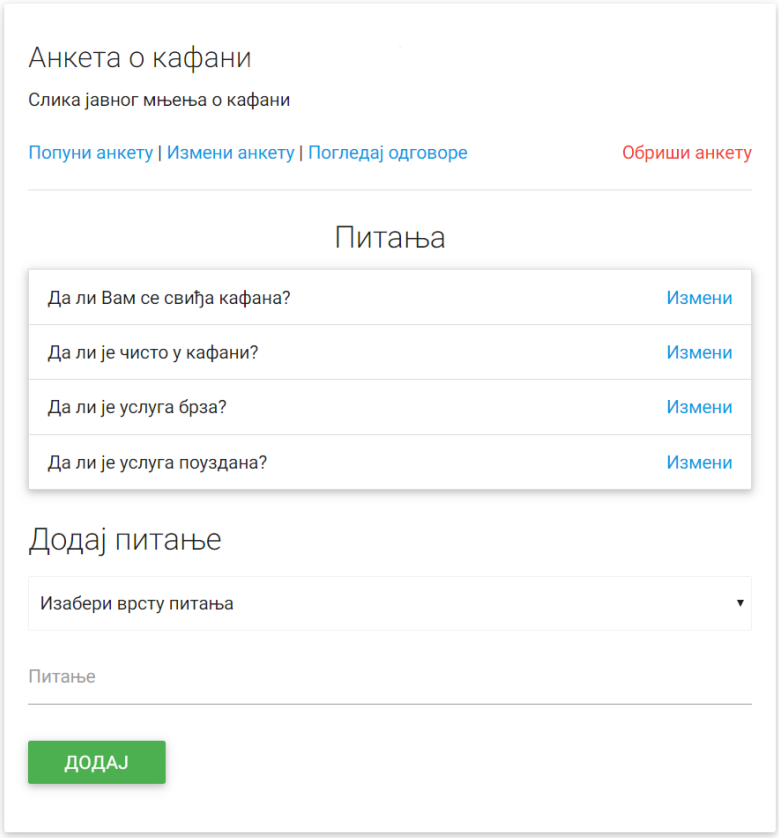
Након регистрације, корисник се мора пријавити да би приступио систему и анкетама. Након пријаве бива преусмерен на главну страну апликације где су излистане све анкете (слика 13). У овом прозору су излистане све анкете свих корисника, дугме за прављење нове анкете („**Направи нову анкету**”) као и операције које тренутно пријављени корисник може да чини над сваком од анкета.



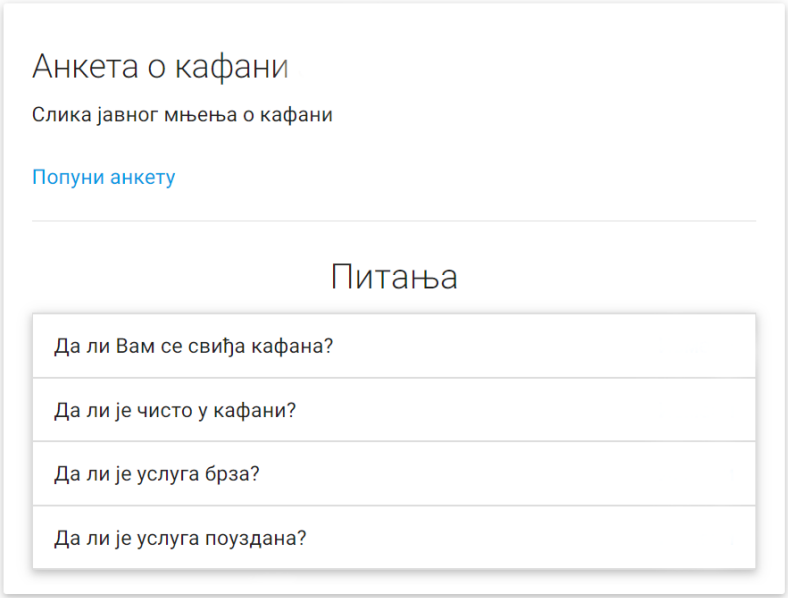
***Слика 13:** Елемент главне странице*

Корисник има право да извршава следеће операције над својим анкетама: преглед детаља анкете, уређивање назива и описа анкете, уређивање питања у анкети, преглед резултата анкете и брисање анкете. Тренутно пријављени корисник може на главној страни видети анкете осталих корисника, уз могућност прегледа детаља и попуњавања.

Кликом на назив једне од анкета са главне стране отвара се страница са детаљима анкете (слике 14 и 15).

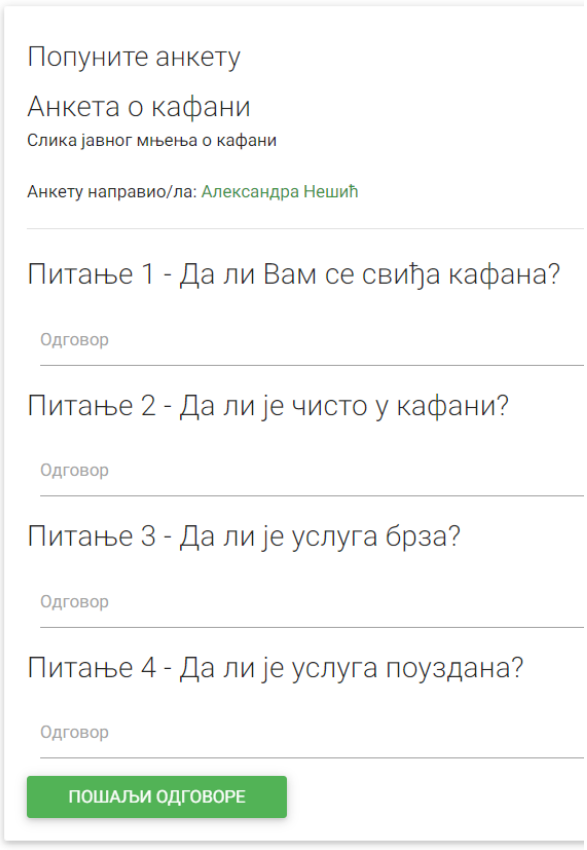


Слика 14: *Детаљи о анкети приказани кориснику који ју је направио*



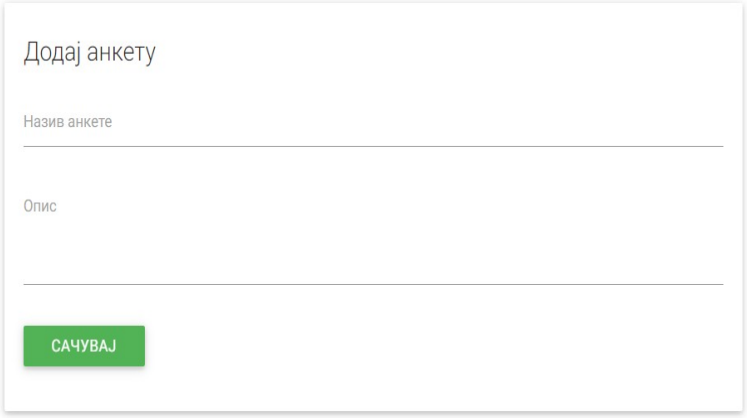
Слика 15: *Приказ детаља анкете страном кориснику*

Када корисник жели да попуни анкету, притиском на одговарајуће дугме му се отвара форма за попуњавање са излистаним свим питањима и пољима за унос садржаја. Након што попуни сва поља (нека може оставити празна), притиском на дугме („**Пошаљи одговоре**”) садржај из форми се прослеђује серверу а он их смешта у одговарајућу табелу у бази података.



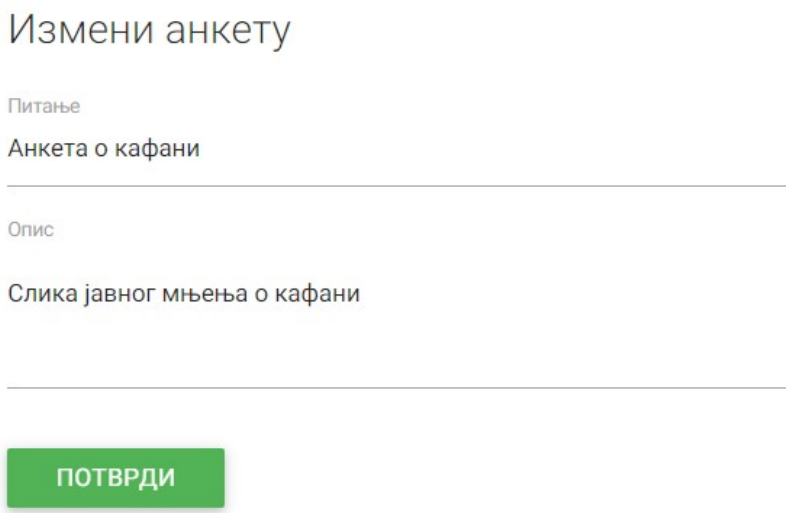
Слика 16: Форма за попуњавање анкете

Када корисник жели да направи нову табелу, то чини притиском на одговарајући текст на главној страници („**Направи нову анкету**”). Сервер тада прослеђује страницу са формом за прављење нове анкете, а први корак је дефинисање назива и описа (слика 17).



***Слика 17**: Прављење нове анкете*

Након прављења нове анкете, сервер корисника прослеђује на страницу са детаљима анкете на којој он има могућност да додаје питања или да мења назив и опис у случају да је погрешио при прављењу анкете. Притиском на дугме („**Измени анкету**”) отвара му се прозор идентичан прозору за прављење нове анкете, али овај пут су поља за име и опис већ попуњена постојећим подацима о анкети (слика 18).

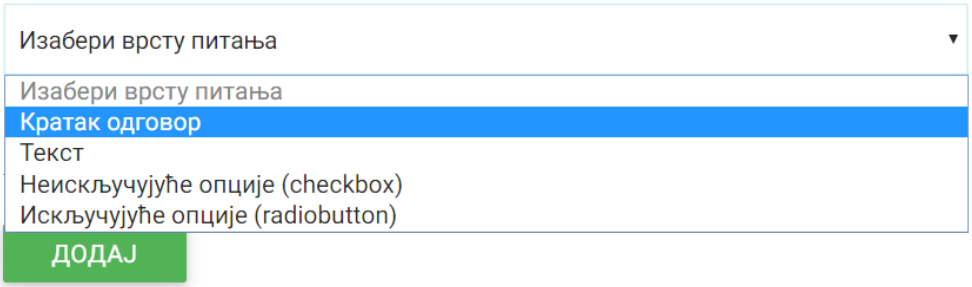


***Слика 18**: Форма за измену детаља о анкети*

Када додаје ново питање, кориснику се пружа избор од четири врсте питања (слика 19):

- кратак одговор
- текст (дуг одговор)
- неискључујуће опције (*checkbox*)
- искључујуће опције (*radiobutton*)

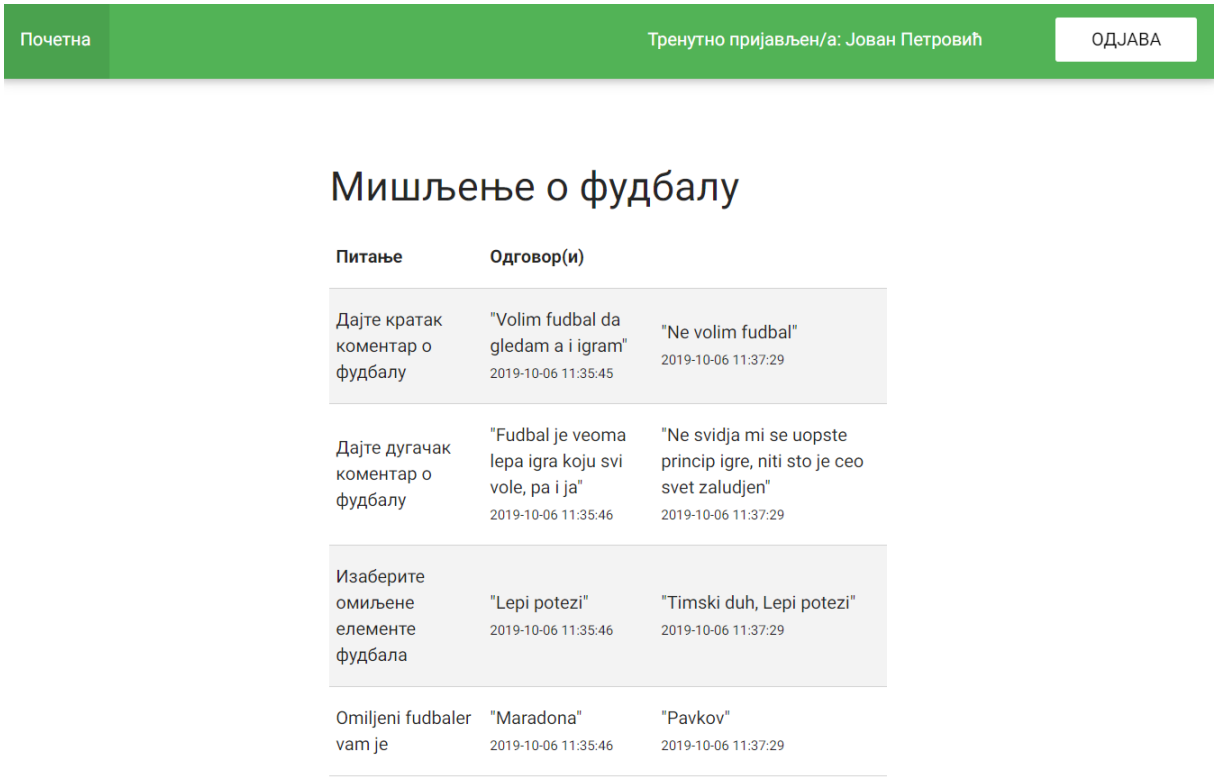
Једна анкета може имати неограничено много питања, а једно питање може имати неограничено много одговора различитих типова.



***Слика 19**: Падајући мени за избор врсте питања*

Корисник додаје нове опцијом притиском на одговарајуће дугме („**Додај**”), а у случају питања која имају опције, појављују се нова два дугмета у интерфејсу за додавање опције („**Додај нову опцију**”) и брисање опције („**Обриши**”).

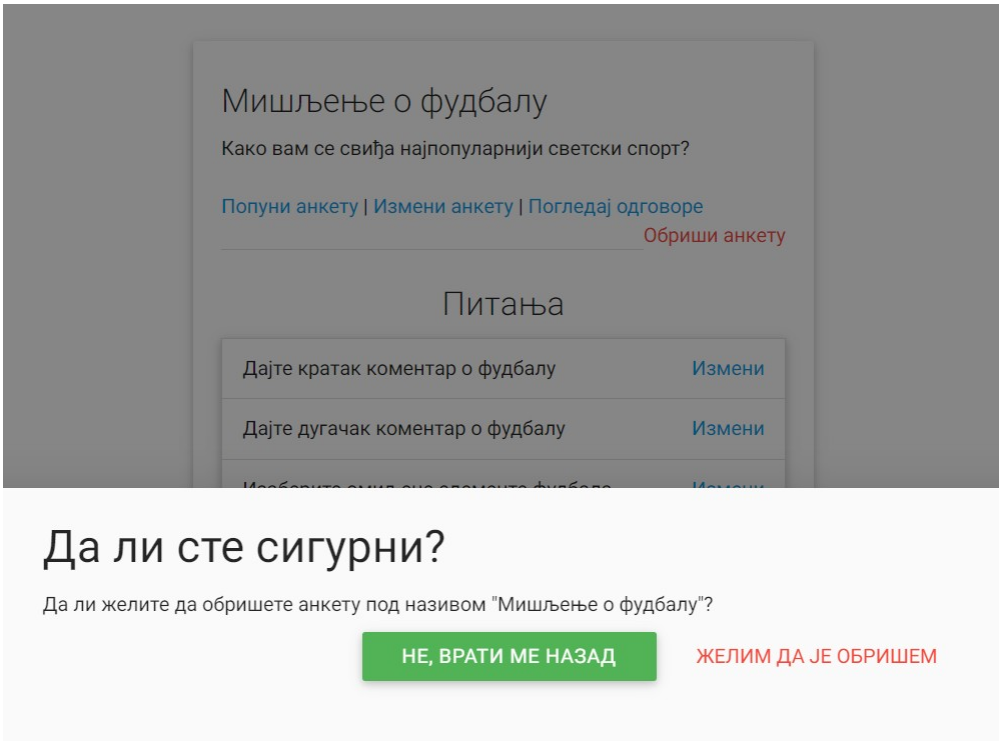
Како би приступио резултатима своје анкете, корисник на главној страници треба да притисне дугме за приказ резултата које изгледа као мали графикон са три различита стуба (када пређе показивачем миша преко дугмета, приказаће се текст „**Погледај резултате анкете**”). Дугме ће бити присутно само за анкете које је он направио. Притиском на то дугме отвара се страница са резултатима анкете у чијем је реду било дугме, а на страници ће се приказати табела са резултатима из сваке попуњене инстанце те анкете (слика 20).



***Слика 20**: Страница са резултатима анкете*

Анкетирање је анонимно и може се вршити више пута, а за свако попуњавање се бележи временска ознака (датум и време).

Када прође одређено време и анкета застари, корисник који је њу направио је може и обрисати тако што приступи њеним детаљима и изабере опцију („**Обриши анкету**”) (слика 14). Притом се отвара прозор обавештења који даје две опције: потврду брисања („**Желим да је обришем**”) и отказивање брисања („**Не, врати ме назад**”). Овај прозор је приказан на слици 21.



***Слика 21**: Приказ прозора за потврду брисања анкете*

6. Закључак

Овај рад приказује приступ изради решења софтвера за прављење анкета користећи се широко доступним средствима модерног информатичког доба.

Ера савремених иновација захтева побољшање постојећих метода дистрибуирања и прикупљања података. У овом раду је демонстрирано осавремењавање анкета модерним информатичким средствима што побољшава њихову дистрибуцију до циљне популације. Елиминацијом куповине папира, штампе и дистрибуције анкетних листова до циљне популације се практично анулирају средства потребна за реализацију било које анкете. Побољшава се учинак анкета превазилажењем проблема коју представља удаљеност циљне групе од извора анкете, а све чешће се циљне групе до танчина дефинишу користећи податке о особама прикупљене са друштвених мрежа.

Овај програм се може побољшати увођењем статистичког приказа и извоза резултата анкете, уз додатно побољшање изгледа апликације додавањем графичких елемената.

Литература

[1] Shaughnessy, J., Zechmeister, E., Jeanne, Z., Research methods in psychology (9th ed.). New York, NY: McGraw Hill, 2011.

[2] Предности и мане анкета, преузето октобра 2019. са https://www.snapsurveys.com/blog/advantages-disadvantages-surveys/

[3] О интернет апликацијама, преузето октобра 2019. са https://searchsoftwarequality.techtarget.com/definition/Web-application-Web-app

[4] James F. Kurose, Keith W. Ross, Умрежавање рачунара, превод четвртог издања, РАФ, ЦЕТ, 2009.

[5] HTML, W3 Schools, преузето октобра 2019. са https://www.w3schools.com/html

[6] Jennifer Niederst Robbins, Learning Web Design: A Beginner's Guide to (X)HTML, StyleSheets, and Web Graphics, електронско издање, САД, O'Reilly, 2007.

[7] CSS, W3 Schools, преузето октобра 2019. са https://www.w3schools.com/css

[8] MaterializeCSS, преузето октобра 2019. са https://materializecss.com/about

[9] Историја PHP-а, преузето октобра 2019. са https://www.php.net/manual/en/history.php.php

[10] Laravel, преузето октобра 2019. са https://laravel.com/docs/6.x