Web aplikacija za pomoć pri učenju programskih jezika

Krajačić, Matija

Undergraduate thesis / Završni rad

2021

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Polytechnic of Međimurje in Čakovec / Međimursko veleučilište u Čakovcu**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:110:198996

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-02-17**

Repository / Repozitorij:

Polytechnic of Međimurje in Čakovec Repository - Polytechnic of Međimurje Undergraduate and Graduate Theses Repository

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU
PREDDIPLOMSKI STRUČNI STUDIJ RAČUNARSTVA

MATIJA KRAJAČIĆ

WEB APLIKACIJA ZA POMOĆ PRI UČENJU PROGRAMSKIH JEZIKA

ZAVRŠNI RAD

ČAKOVEC, 2021.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU
PREDDIPLOMSKI STRUČNI STUDIJ RAČUNARSTVA

MATIJA KRAJAČIĆ

WEB APLIKACIJA ZA POMOĆ PRI UČENJU PROGRAMSKIH JEZIKA

WEB APPLICATION FOR LEARNING PROGRAMMING LANGUAGES

ZAVRŠNI RAD

Mentor:
dr. sc. Sanja Brekalo

ČAKOVEC, 2021.

ZAHVALA

Zahvaljujem mentorici dr. sc. Sanji Brekalo na pomoći pri izradi ovog rada te odabiru teme.

Zahvaljujem majci Tatjani na podršci i razumijevanju tijekom studiranja i ocu Ivanu na svoj njegovoj pomoći.

Zahvaljujem prijateljima koji su uljepšali vrijeme studiranja, te svima ostalima koji su me podržavali i vjerovali u mene.

Matija Krajačić

SAŽETAK

Tema rada je izrada web aplikacije za pomoć pri učenju programskih jezika. Cilj aplikacije je olakšati učenje programskih jezika kroz kvizna pitanja. Tečaj programskog jezika unutar aplikacije podijeljen je na lekcije kao što su sintaksa jezika, varijable, funkcije, objekti, klase itd. Svaka lekcija sadrži opis u kojem se nalaze informacije vezane uz lekciju. Temeljem sadržaja iz opisa lekcije kreirano je 10-15 pitanja na koja korisnik mora točno odgovoriti kako bi prošao lekciju. Vrste pitanja koje lekcija može sadržavati su pitanja s ponuđenim odgovorima od kojih jedan ili više odgovora može biti točan, pitanja koja zahtjevaju od korisnika upis teksta te pitanja koja zahtjevaju od korisnika da upiše programski kod. Korisnik prolazi lekciju ukoliko je netočno odgovorio na manje od 3 postavljena pitanja. Ukoliko je korisnik prošao svaku od lekcija unutar jezika, smatra se da je uspješno savladao gradivo. Pitanjima iz lekcije pristupa se pritiskom na gumb koji se nalazi na stranici s opisom lekcije. Ukoliko je korisnik prošao bilo koju od lekcija unutar jezika, ili je na bilo koje pitanje netočno odgovorio, otključava mu se opcija „Vježba" koja mu postavlja do 15 pitanja iz lekcija koje je prošao. Pri odgovaranju na pitanja, informacije o netočno odgovorenim pitanjima spremaju se u bazu podataka. Tijekom svakog pokretanja kviza korisnik ima 3 života. Svakim netočnim odgovorom gubi jedan život. Izgubi li korisnik sva 3 života, kviz se prekida te korisnik ne prolazi lekciju na čija je pitanja odgovarao. Na netočno odgovorena pitanja korisnik može ponovo odgovarati preko opcije „Vježba", te ukoliko ih riješi, prolazi lekciju.

U radu je opisan postupak izrade aplikacije, te korištene tehnologije među kojima su HTML, CSS i AJAX za korisničko sučelje (engl. front-end), PHP, MySQL, Node.js, CORS Anywhere za arhitekturu i logiku sustava skrivenu od krajnjeg korisnika i JDoodle i CodeMirror za izradu editora za kodiranje i izvršavanje programskog koda upisanog od strane korisnika.

Ključne riječi: *PHP, AJAX, kviz, tečaj, programiranje, učenje*

SADRŽAJ

1. Uvod

Razvojem i sve češćim korištenjem digitalnih tehnologija, vještine i znanja programiranja svakim danom postaju sve traženija. Vjerojatno najbolji način učenja i svladavanja pojedinog programskog jezika je izradom aplikacija te povećavanjem njihovih kompleksnosti svakim idućim programskim zadatkom. Početnici u programiranju prije razvoja samih aplikacija trebaju usvojiti osnovna znanja vezana uz sintaksu i tehnike programiranja i izvršavanja koda. Učiti osnove jezika moguće je čitanjem teorije programiranja te isprobavanjem osnovnih naredbi iz programskih jezika, praćenjem video sadržaja, nastave ili korištenjem multimedijalnih aplikacija. Učenjem preko multimedijalnih aplikacija, proces učenja programiranja može postati zabavniji. Odgovaranjem na pitanja kviza, praćenjem lekcija te otključavanjem novih lekcija, može se stvoriti dojam učenja kroz igru te samim tim proces učenja učiniti zabavnijim. Web aplikacija za pomoć pri učenju programskih jezika osmišljena je upravo na navedeni način.

Istraživanje koje je uspoređivalo rezultate studenata koji su učili gradivo isključivo preko video sadržaja s rezultatima studenata koji su isto gradivo učili preko video sadržaja te interaktivnih sadržaja pokazalo je da je interaktivno učenje šest puta efikasnije od učenja isključivo preko video sadržaja. Rezultati su ukazali da je količina naučenog kroz svaku aktivnost šest puta veća u odnosu na svaki odgledani video zapis ili pročitanu stranicu. Studenti koji su učili preko interaktivnih sadržaja skloniji su sudjelovanju u nastavi te je manja vjerojatnost da napuste studij. [1]

2. Cilj rada

Standardni načini učenja programiranja ponekad mogu biti neefikasni. Alternativnim načinima učenja, kao što su kvizovi, učenje može postati zanimljivije. Uz opise lekcija koji iznose bitne činjenice i pitanja koja potiču shvaćanje gradiva te lakše pamćenje istog, efikasnost učenja može se povećati. Cilj rada je kreiranje aplikacije za učenje programiranja kojom proces učenja postaje zabavniji i efikasniji. Istovremeno, cilj je izraditi funkcionalnu aplikaciju korištenjem web tehnologija.

3. Upotrijebljene tehnologije

3.1. HyperText Markup Language

HyperText Markup Language, ili skraćeno HTML, prezentacijski je jezik za izradu web stranica. Izgled web stranica u HTML-u opisuje se pomoću HTML oznaka (engl. *tagova*). HTML oznake određuju položaj i način prikazivanja sadržaja web stranice. HTML oznake interpretira, tj. tumači web preglednik te stvara prikaz web stranice. Oznake u HTML-u zapravo su upute web pregledniku kojima se određuje prikaz sadržaja web stranice. HTML oznake navode se unutar znakova „< “ i „> “ [2]. Nakon što se otvori HTML oznaka, primjerice <*body*>, oznaku je potrebno zatvoriti na isti način kao što se i otvara, uz dodavanje desne kose crte ispred naziva oznake, primjerice </*body*>.

Početna oznaka HTML dokumenta je <*html*>, u kojoj se nalaze dva osnovna dijela: zaglavlje i tijelo. Zaglavlje se definira oznakom <*head*> te sadrži osnovne informacije o dokumentu kao što su naslov i metapodaci te pozive potrebnih CSS ili skriptnih datoteka. HTML oznake mogu sadržavati atribute koji dodatno opisuju svojstva oznake. Primjerice, za element paragrafa, oznake <*p*>, moguće je atributom *align=“right“* odrediti poravnanje odlomka, u spomenutom primjeru to bi bilo desno poravnjanje. Prva standardna verzija HTML-a kreirana je 1995. godine. Danas se preporučuje korištenje najnovije verzije HTML5 koja je donijela veću podršku za multimedijske datoteke, nove semantičke strukturne elemente te nove atribute. Na slici 1. prikazan je HTML kod za strukturu jednostavnog HTML dokumenta sa zaglavljem i tijelom.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <p>Text</p>
</body>
</html>
```

Slika 1. Jednostavan HTML dokument sa zaglavljem i tijelom
Izvor: Autor

3.2. Cascading Style Sheets

Cascading Style Sheets, ili skraćeno CSS, stilski je jezik korišten za opis izgleda dokumenta izrađenog u HTML-u, XML-u i povezanim jezicima [3]. CSS-om je moguće birati HTML elemente (oznake) te mijenjati svojstva prikaza odabranih elemenata.

Slika 2. Primjer dohvaćanja elementa unutar CSS-a
Izvor: Autor

Slika 2. prikazuje dohvaćanje elementa preko CSS-a i mijenjanje svojstva prikaza. U primjeru dohvaćeni je paragraf element p. Unutar vitičastih zagrada navodi se svojstvo koje se želi promijeniti. Nakon zapisa svojstva koje se želi izmijeniti potrebna je dvotočka prije upisivanja željene vrijednosti te znak točke sa zarezom.

3.3. JavaScript

JavaScript je najpoznatiji kao skriptni jezik za web stranice, međutim, koristi se i u okolinama izvan weba, kao što su Node.js ili Adobe Acrobat. JavaScript podržava proceduralno i objektno orijentirano programiranje. Objekti se kreiraju dodavanjem metoda i atributa u vrijeme izvođenja programa, za razliku od klasa definiranih u kompajliranim jezicima kao što su C++ ili Java. JavaScript često se koristi za animacije, asinkrono dodavanje sadržaja na web stranicu, izradu jednostraničnih web sučelja u nekom od programskih okvira, igre unutar web preglednika, kontrolu reproduciranja multimedijskih sadržaja, validaciju upisanih podataka prije slanja serveru itd. Preko 80% web stranica koristi JavaScript bibilioteke i programske okvire. Neki od najpopularnijih programskih okvira su jQuery, React i Angular. Osim za web, može se koristiti za izradu desktop ili mobilnih aplikacija.

3.3.1. JQuery

JQuery je JavaScript biblioteka koja olakšava izmjenu HTML sadržaja, rukovanje događajima (klik mišom, prijelaz mišem preko sadržaja itd.), kreiranje animacija i korištenje AJAX-a. Oko 73% od 10 milijuna najpopularnijih web stranica koristi biblioteku JQuery. Dohvaćanje elemenata

web stranice pomoću jQuerya obično započinje pozivanjem *$* funkcije uz CSS selektor. Time se stvara jQuery objekt koji je referenca na sve HTML elemente koji odgovaraju CSS selektoru [4]. Na taj objekt moguće je pozivati funkcije ili rukovati događajima. Na slici 3. prikazan je primjer dodavanja funkcionalnosti pomoću jQuery za slučaj pritiska na gumb s identifikatorom *usernameSubmit*.

```
$('#usernameSubmit').click(function() {
    var fd = new FormData();
    var user_id = $('#user-name-id').val();
    var username = $('#usr-username').val();

    fd.append('user-name-id', user_id);
    fd.append('usr-username', username);
```

Slika 3. Primjer dodavanja funkcionalnosti pomoću jQuerya
Izvor: Autor

3.3.2. AJAX

Asynchronous JavaScript and XML ili AJAX, set je web tehnologija za kreiranje asinkronih web aplikacija. Pomoću AJAX-a, web aplikacijama je omogućeno slanje i primanje podataka sa servera u pozadini bez utjecaja na prikazivanje i ponašanje već učitane stranice. AJAX omogućuje web stranicama i aplikacijama da mijenjaju sadržaj dinamički bez potrebe za ponovnim učitavanjem cijele stranice. HTML i CSS koriste se u AJAX-u za stiliziranje informacija. AJAX koristi JavaScript, JSON i XMLHttpRequest za AJAX zahtjeve [5]. U samom nazivu AJAX-a spominje se XML, međutim u modernim aplikacijama i web stranicama češće se koristi JSON umjesto XML-a.

3.3.3. XMLHttpRequest

XMLHttpRequest je aplikacijsko programsko sučelje (engl. *API*) u obliku objekta čije metode prenose podatke između web preglednika i web servera. Ovaj objekt ugrađen je u JavaScript okolinu unutar web preglednika. Dohvaćanje podataka iz XMLHttpRequest objekta temeljna je sastavnica AJAX dizajna. XMLHttpRequest moguće je koristiti i uz protokole različite od HTTP-a. Podaci ne moraju biti samo u obliku XML-a, nego mogu biti i u JSON ili HTML obliku ili u

obliku običnog teksta. Slika 4. prikazuje JavaScript kod za korištenje XMLHttpRequest objekta za dohvaćanje odgovora servera. [6]

```
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
  if (this.readyState == 4 && this.status == 200) {
    document.getElementById("demo").innerHTML =
    this.responseText;
  }
```

Slika 4. Primjer korištenja XMLHttpRequest objekta za dohvaćanje odgovora servera [7]

3.3.4. JavaScript Object Notation

JavaScript Object Notation ili JSON tekstualni je format za prikaz strukturiranih podataka baziran na sintaksi JavaScript objekata. Koristi se u web aplikacijama za prijenos podataka između korisnika i servera. U JSON datoteku moguće je uključiti tekstualne podatke, brojeve, polja, boolean podatke ili objekte. [8]

3.4. PHP

Hypertext Preprocessor ili PHP skriptni je jezik koji se izvršava na serveru. PHP je moguće koristiti za prikupljanje podataka iz formi (obrazaca), dinamičko generiranje sadržaja na web stranici, slanje i primanje kolačića, stvaranje sesija i dr. [9]. PHP omogućuje pristup bazama podataka i komuniciranje s raznim Internet servisima kao što su HTTP ili POP3. Interpretiran i izvršen PHP kod zapravo formira HTTP odgovor (engl. *response*). PHP se koristi u gotovo 80% web stranica čiji je serverski jezik poznat javnosti [10]. Najnovija PHP verzija danas je PHP 8.0. PHP kod se ubacuje u HTML dokument otvaranjem *<?php* oznake i završnom oznakom *?>*. Varijable u PHP jeziku započinju $ znakom.

3.4.1. PHP Data Objects

PHP Data Objects ili PDO, sučelje je za pristup bazama podataka dostupno od PHP verzije 5. PDO pruža apstrakcijski sloj za pristup podacima, što znači da bez obzira na korištenu bazu podataka, koriste se iste funkcije za postavljanje upita i dohvaćanje podataka. PDO povećava sigurnost i upotrebljivost koda te ga čini jednostavnijim. Na slici 5. prikazan je kod za povezivanje

na bazu podataka koristeći PDO sučelje.

```php
function connect()
{
  $servername = "localhost";
  $username = "admin";
  $password = "admin5";
  $db = "coding";

  try {
    $conn = new PDO("mysql:host=$servername;dbname=$db", $username, $password);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  } catch (PDOException $e) {
    echo "Connection failed: " . $e->getMessage();
  }
  return $conn;
}
```

Slika 5. Primjer povezivanja na bazu podataka koristeći PDO sučelje
Izvor: Autor

3.4.2. HTMLPurifier

HTMLPurifier je PHP biblioteka za filtriranje i čišćenje HTML teksta. Korisna je za pročišćavanje teksta napisanog pomoću uređivača bogatog teksta od zlonamjernih HTML tagova koji bi se mogli koristiti pri XSS (*Cross-Site Scripting)* napadima. Biblioteka rastavlja dani tekst u tokene, te iz njih uklanja elemente koji se ne nalaze na listi prihvatljivih elemenata. Istovremeno, biblioteka provjerava formiranost oznaka te osigurava usklađenost HTML-a s W3C standardima [11]. Slika 6. prikazuje kod za inicijalizaciju i korištenje HTMLPurifier biblioteke.

```php
require_once '../vendor/htmlpurifier/library/HTMLPurifier.auto.php';
$config = HTMLPurifier_Config::createDefault();
$purifier = new HTMLPurifier($config);

$clean_desc = $purifier->purify($description);
```

Slika 6. Primjer korišenja HTMLPurifier biblioteke
Izvor: Autor

3.5. MySQL

MySQL je sustav za upravljanje relacijskim bazama podataka otvorenog koda. Podaci u

MySQL bazi podataka spremljeni su u tablice – kolekcije povezanih podataka koje sadrže redove i stupce. Podacima u bazi podataka moguće je pristupiti MySQL upitima. Slika 7. prikazuje MySQL upit za prikaz vrijednosti iz stupca *image* unutar tablice.

```
$query = "SELECT l.image FROM " . $this->table . " l WHERE l.id=?";
```

Slika 7. Primjer MySQL upita u PHP-u
Izvor: Autor

3.6. JDoodle

JDoodle kompajler je online aplikacijsko programsko sučelje (engl. *API*) za kompajliranje i izvršavanje programskog koda raznih programskih jezika kao što su Java, C, C++, Python, Ruby i mnogi drugi. JDoodle API baziran je na JSON REST arhitekturi. Za korištenje API servisa potrebni su parametri *clientId* i *clientSecret*, koji se mogu dobiti registracijom na web stranicu servisa. Za 200 izvršenih dnevnih poziva API servisa nema naplate. Za 1000 dnevnih poziva nadalje potrebno je plaćanje pretplate na servis. Uz *clientId* i *clientSecret*, potrebni parametri za API poziv su *language*, programski jezik čiji se kod želi izvršiti, *versionIndex*, verzija programskog jezika te *script*, kod koji se želi izvršiti. Na slici 8. prikazan je primjer ulaznih podataka za API poziv, gdje je u varijablu *script* upisan kod unutar postavljene forme koji će se izvršavati.

```
            const proxy = "http://localhost:8080/";
            const url = "https://api.jdoodle.com/v1/execute/";
            var script = code_answer.getValue();

 var data = {
    clientId: "ef873qqrh32tigf8q247tz56z363dc245c",
    clientSecret: "qnco7624t8597bc8374qb6589666669696v726669c67nv83476",
    language: '<?php if (isset($compiler_mode)) {echo $compiler_mode;}  ?>',
                script: script,
versionIndex: '<?php if (isset($language_version)) {echo $language_version;}
?>'
            };
```

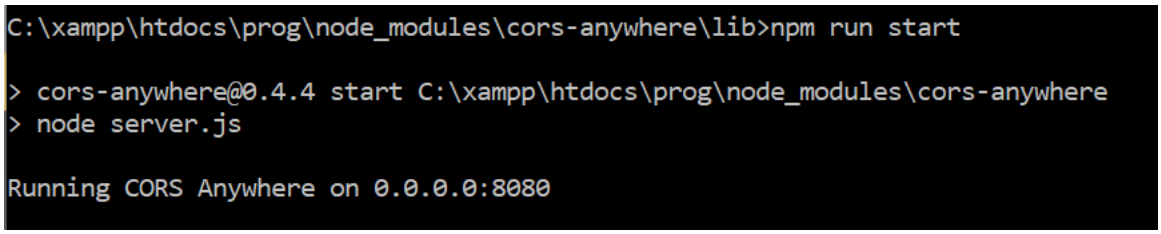
Slika 8. Primjer ulaznih podataka za API poziv
Izvor: Autor

3.7. Node.js

Node.js je izvršno okruženje otvorenog koda koje izvršava JavaScript programski jezik. Node.js koristi asinkrono programiranje. Omogućuje generiranje dinamičnog sadržaja web stranice, manipulaciju datotekama i kreiranje datoteka na serveru, prikupljanje podataka s obrazaca, dodavanje podataka u bazu podataka i manipulaciju podacima iz baze podataka.

3.7.1. CORS Anywhere

Cross-Origin Resource Sharing ili CORS je mehanizam baziran na HTTP zaglavljima koji omogućuje serverima da domenama ili portovima ograniče učitavanje resursa sa servera. JDoodle API servis namijenjen je za pozive između servera. CORS mehanizam servisa onemogućuje izvršavanje API poziva direktno s web stranice, jer bi se time povjerljivi ulazni podaci *clientID* i *clientSecret* izložili nesigurnosti. Zbog toga se unutar aplikacije koristi Node.js proxy server CORS Anywhere, koji dodaje potrebna CORS zaglavlja na zahtjev. Proxy se pokreće unutar komandne linije izmjenom direktorija na direktorij *lib* biblioteke CORS Anywhere, te upisivanjem naredbe *npm run start*. Na slici 9. prikazan je primjer uspješnog pokretanja CORS Anywhere proxy servera preko komandne linije.



Slika 9. Uspješno pokrenut CORS Anywhere proxy server
Izvor: Autor

3.8. CodeMirror

CodeMirror je uređivač za kodiranje koji je moguće ugraditi u web stranicu. Implementiran je u JavaScriptu. CodeMirror sadrži modove specifične za pojedinačne programske jezike. Modovi dodavaju funkcionalnosti prilagođene odabranom programskom jeziku kao što su automatsko dodavanje uvlaka ili označavanje sintakse. Osim modova, instanci CodeMirror editora moguće je dodati i *value* parametar kojim se instanci dodaje upisani kod ili tekst tijekom učitavanja. Između ostalog, moguće je promijeniti izgled editora biranjem između brojnih tema i dodavanjem

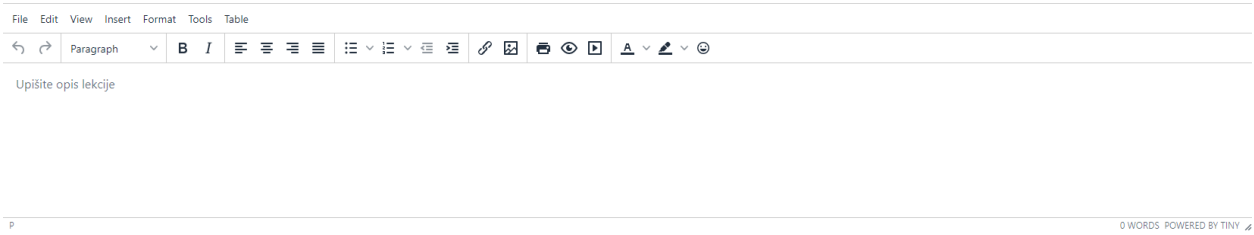
parametra *theme*. Najnovija stabilna CodeMirror verzija je 5.61.1, dok je verzija 6 u beta fazi [12]. Slika 10. prikazuje JavaScript kod za kreiranje instance CodeMirror uređivača koda.

```
CodeMirror(document.querySelector('#my-div'), {
  lineNumbers: true,
  tabSize: 2,
  value: 'console.log("Hello, World");',
  mode:'javascript',
  theme: "base16-dark"
});
```

Slika 10. Kod za kreiranje instance CodeMirror editora
Izvor: Autor

3.9. TinyMCE

TinyMCE je JavaScript dodatak koji omogućuje pretvorbu HTML elemenata u instance uređivača bogatog teksta (engl. *Rich text editor*). Uređivač bogatog teksta omogućuje formatiranje teksta (izmjenu boja, fonta, veličine itd.), dodavanje multimedijskog sadržaja, dodavanje poveznica te mnoge druge opcije koje se mogu dodavati pomoću dodataka za TinyMCE. TinyMCE trenutno koristi više od 100 milijuna aplikacija i web stranica, a dodatak je preuziman više od 350 milijuna puta godišnje [13]. Na slici 11. prikazan je izgled instance TinyMCE uređivača bogatog teksta.



Slika 11. Instanca TinyMCE uređivača teksta
Izvor: Autor

3.10. Bootstrap

Bootstrap je razvojno okruženje za lakše i brže razvijanje web stranica i aplikacija. Sadrži HTML i CSS predloške za obrasce, gumbe, tablice, menije, modalne prozore, te mnoge druge elemente. Uključuje i JavaScript dodatke. Uz Bootstrap uvelike je olakšano kreiranje responzivnog

dizajna, prilagodljivog za sve veličine ekrana. Da bi se postigli željeni efekti ili izgledi komponenata, potrebno je na HTML elemente postaviti namijenjene Bootrap klase. Primjerice, ukoliko želimo promijeniti boju paragrafa *p* u crvenu, potrebno je samo u *class* atribut elementa dodati klasu *text-danger*.

3.11. Slick.js

Slick.js je jQuery dodatak za izradu „klizača“ (engl. *Slidera*). Dodatak nudi brojne opcije prilagodbe kao što su specificiranje broja prikazanih stavki, specificiranje pomaka klikom na navigacijske tipke, prilagodba visine po veličini stavke. Navigacija kroz stavke moguća je korištenjem navigacijskih tipki ili povlačenjem miša u željenu stranu. Omogućena je izmjena opcija ovisno o širini ekrana. Na web aplikaciji je za prikaz popisa programskih jezika i lekcija korišten navedeni dodatak, što je prikazano na slici 12.

Slika 12. Prikaz programskih jezika Slick.js "klizačem"
Izvor: Autor

3.12. Typed.js

JavaScript biblioteka Typed.js omogućuje jednostavnu izradu animacija upisivanja i ispisivanja teksta. Biblioteka unutar odabranog HTML elementa kreira animaciju utipkavanja riječi zadanih unutar polja. Završetkom animacije utipkavanja riječi, započinje animacija brisanja te riječi te nakon što je riječ obrisana, započinje nova animacija utipkavanja iduće riječi unutar polja. Slika

13. prikazuje JavaScript kod za inicijalizaciju Typed.js biblioteke.

```
var string = $('.details-desc span').html();
var typed_eror = new Typed('.details-desc span', {
    strings: [string, '', string],
    typeSpeed: 60,
    backSpeed: 40,
    backDelay: 1000,
    startDelay: 1000,
    smartBackspace: false,
    loop: true
});
```

Slika 13. Primjer inicijalizacije Typed.js biblioteke
Izvor: Autor

4. Razvojno okruženje i alati

4.1. Visual Studio Code

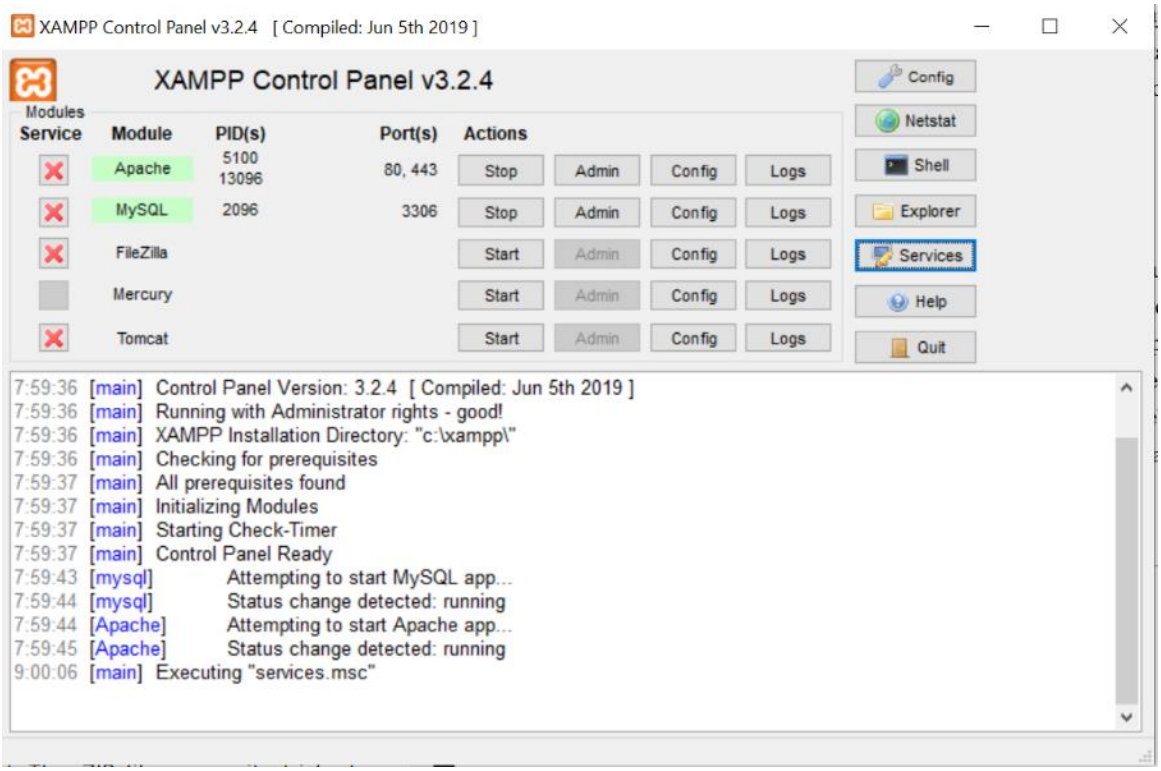
Visual Studio Code je uređivač koda otvorenog tipa kreiran od strane Microsofta. Dolazi s ugrađenom podrškom za JavaScript, TypeScript i Node.js. Omogućuje pronalažanje pogrešaka u kodu (debagiranje), označavanje sintakse, inteligentno dovršavanje koda, nudi gotove isječke koda, refaktoriranje te ugrađenu GitHub podršku. [14] Moguće je biranje između tema, moguća je izmjena prečaca na tipkovnici, izmjena postavki i instalacija brojnih dodataka za dodatne funkcionalnosti i podršku za programske jezike kao što su C++, C#, Java, Python, PHP, GO i dr.

4.2. NPM

Node Package Manager ili NPM najveća je biblioteka programa, s više od 800 000 paketa. Koristi se za upravljanje paketima i instalaciju paketa. Kako bi se koristio NPM, prvo je potrebno instalirati Node.js. [15] NPM naredbe upisuju se preko komandne linije. Neke od mogućnosti NPM-a su prilagođavanje paketa za potrebe aplikacije, preuzimanje alata, dijeljenje koda s ostalim NPM korisnicima, dijeljenje koda samo odabranim korisnicima.

4.3. XAMPP

XAMPP je više platformski web server koji pomaže pri kreiranju i testiranju aplikacija na računalu na kojem se aplikacija izvršava. Razvijen je od strane Apache Friends. Uključuje Apache HTTP Server, MariaDB bazu podataka te interpretere za programske jezike kao što su PHP i Pearl. U instalaciju je uključen i phpMyAdmin, alat za rukovanje bazama podataka, OpenSSL, implementacija otvorenog koda za Secure Socket Layer Protocol te Transport Layer Protocol, XAMPP Control Panel, kontrolna ploča za upravljanje XAMPP komponentama, Webalizer, programsko rješenje za web analitiku i detalje o korištenju, Mercury, server za slanje mailova, Tomcat, implementacija Java funkcionalnosti i tehnologija te Filezilla, FTP server za slanje podataka [16]. Slika 14. prikazuje izgled XAMPP upravljačke ploče.



Slika 14. XAMPP upravljačka ploča
Izvor: Autor

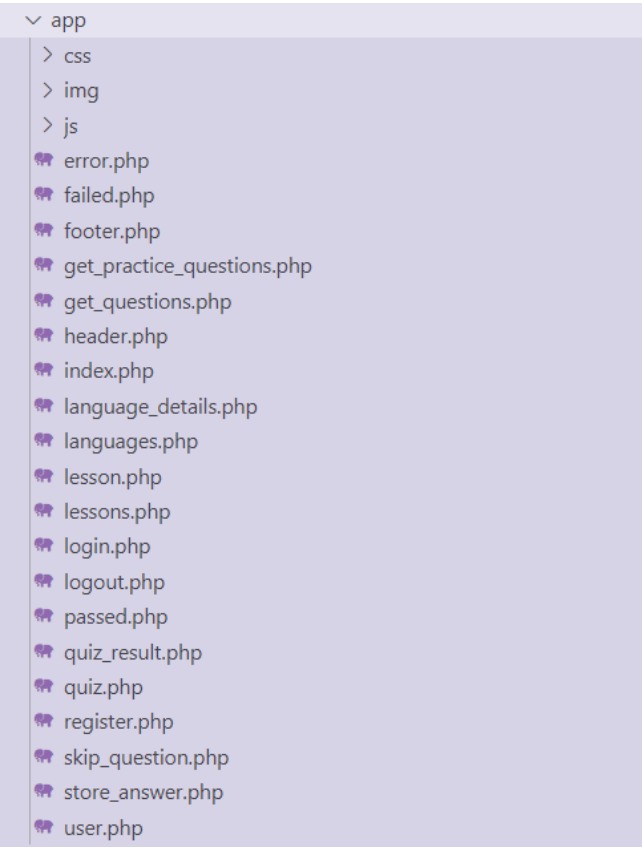
5. Razvoj aplikacije

Na početnoj stranici izrađene aplikacije korisnik može vidjeti dostupne programske jezike za učenje te gumbe za registraciju i prijavu. Nakon uspješne prijave, učitava se lista dostupnih jezika. Ispod naziva i fotografije svakog od jezika nalazi se gumb za prikaz detalja o jeziku i gumb za prikaz dostupnih lekcija unutar odabranog jezika. Ukoliko je korisnik već prošao cijeli jezik ili neku od lekcija iz jezika, iznad naziva jezika prikazat će ikona koja označava da je jezik riješen, ili natpis „U tijeku“ ukoliko je riješena barem jedna lekcija iz jezika. Klikom na gumb „Lekcije“ ispod naziva i fotografije jezika, učitava se lista dostupnih lekcija unutar odabranog jezika. Pokraj naziva odabranog jezika pojaviti će se gumb za opciju „Vježba“. Samo lekcije koje nemaju naveden preduvjet, ili lekcije čiji je preduvijet korisnik riješio, sadržat će gumb za prikaz opisa lekcije. Na prikazu opisa lekcije, pokraj naziva lekcije prikazat će se gumb za početak rješavanja pitanja iz odabrane lekcije. Klikom na gumb pokreće se kviz. Korisniku se učitava stranica uspjeha ili neuspjeha, ovisno o rezultatu kviza. S tih stranica korisnik može birati lokaciju povratka koju želi ovisno o vrsti kviza koji je rješavao.

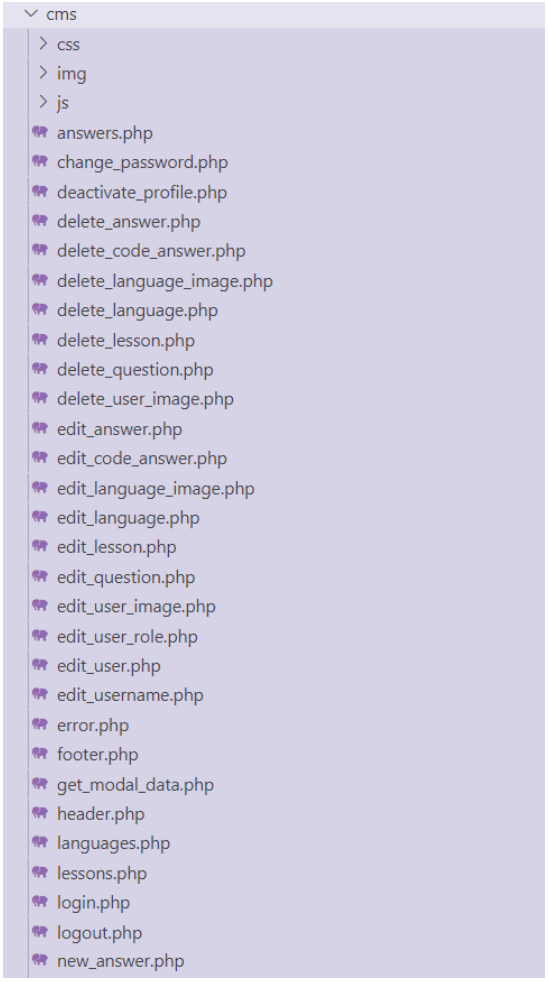
5.1. Struktura datoteka

Slika 15. Struktura datoteka
Izvor: Autor

Slika 15. prikazuje strukturu datoteka unutar aplikacije. U mapi *app*, čija je struktura prikazana na slici 16., nalaze se PHP datoteke potrebne za stvaranje aplikacijskog djela, korištene CSS i JavaScript datoteke spremljene u podmape *css* i *js*, te korištene slikovne datoteke spremljene u podmapu *img*. Mapa *class* sadrži PHP klase za objekte korištene kroz aplikaciju i administrativni dio. Unutar mape *cms*, čija je struktura prikazana na slici 17., nalaze se PHP datoteke potrebne za stvaranje administrativnog dijela, zajedno sa korištenim CSS, JavaScript i slikovnim datotekama spremljenim u podmape *css, js* te *img*. *Node_modules* je mapa koja sadrži datoteke proxy servera CORS Anywhere. U mapi *vendor* nalaze se datoteke korištenih JavaScript biblioteka jQuery, CodeMirror i TinyMCE, datoteke PHP biblioteke HTMLPurifier i datoteke razvojnog okruženja Bootstrap. *Coding.sql* i *database.sql* datoteke su sa strukturom baze podataka i njenim podacima. *Functions.php* je datoteka sa PHP funkcijama korištenim unutar aplikacijskog i administrativnog dijela. *Package-lock.json* i *package.json* su datoteke potrebne za funkcioniranje Node.js proxy servera CORS Anywhere.



Slika 16. Datoteke unutar mape app
Izvor: Autor



Slika 17. Datoteke unutar mape cms
Izvor: Autor

5.2. Način prikazivanja lekcija

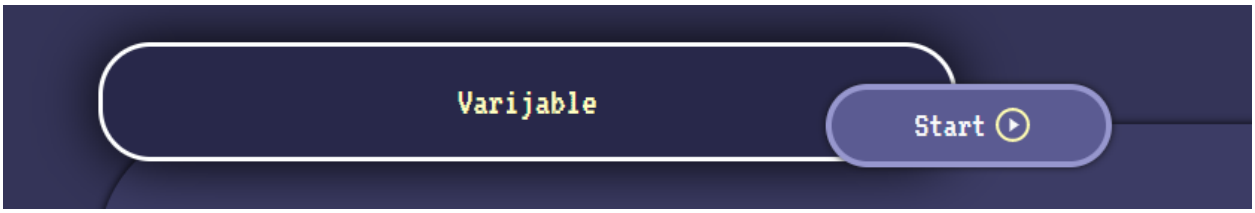
Svaka lekcija u bazi podataka sadrži polje „preduvjet“ (engl. *precondition*) koje označava koja lekcija mora biti riješena da bi lekcija postala dostupna. Ukoliko za prikaz lekcije ne treba ispuniti preduvjet (što je slučaj za prvih nekoliko lekcija), tada vrijednost polja „preduvjet“ iznosi 0. U suprotnom, ukoliko je za prikaz lekcije potrebno ispuniti preduvjet (riješiti određenu lekciju), tada se u polje „preduvjet“ upisuje identifikator lekcije koja se mora riješiti kako bi lekcija bila dostupna.

Odabirom prikaza lekcija koje se nalaze unutar odabranog jezika prikazuju se sve lekcije koje spadaju pod taj jezik. Međutim, ukoliko preduvjet lekcije ne iznosi 0, ili se identifikator lekcije naveden kao preduvjet ne nalazi u lekcijama koje je korisnik prošao, tada se vidljivost (prozirnost) lekcije smanjuje i ne pojavljuje se gumb za prikaz više detalja o lekciji. Ukoliko je korisnik prošao jednu od lekcija, iznad naziva lekcije prikazat će se ikona koja označava da je lekcija riješena. Na slici 18. prikazan je izgled prikaza lekcija u kojem za lekciju „Tipovi podataka“ nije ispunjen preduvjet. Na gornjem rubu sekcije za lekciju „Varijable“ vidljiva je ikona koja označava da je lekcija riješena.

Slika 18. Prikaz lekcija
Izvor: Autor

5.3. Sustav prikupljanja pitanja

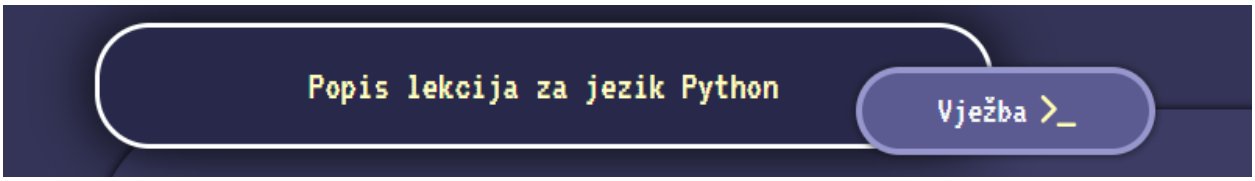
Klikom na gumb za pokretanje kviznih pitanja, koji se nalazi na stranici s detaljnim opisom lekcije programskog jezika, pokreće se sustav za prikupljanje pitanja. Gumb sadrži tekst „Start“ te je prikazan na slici 19.



Slika 19. Gumb za pokretanje kviznih pitanja iz lekcije
Izvor: Autor

Pokreće se AJAX POST zahtjev prema datoteci *get_questions.php*, kojoj se unutar JavaScript FormData objekta šalje identifikator odabrane lekcije, postavljen na atributu gumba *data-name*. U datoteci *get_questions.php* nalaze se sva pitanja koja su povezana s odabranom lekcijom te se identifikatori pitanja spremaju u polje. Inicijaliziraju se sesijske varijable koje se koriste u toku rješavanja kviza. Datoteka *get_questions.php* vraća odgovor JSON oblika s vrijednošću *status*. Ukoliko je *status* postavljen na 1, to znači da su pitanja uspješno prikupljena te je sustav spreman za početak kviza. Korisnika se tada preusmjerava na stranicu *quiz.php* gdje započinje rješavanje kviza.

Klikom na gumb „Vježba“, prikazanog na slici 20, koji se nalazi na stranici za prikaz lekcija iz odabranog jezika, pokreće se sustav za prikupljanje pitanja za vježbu.



Slika 20. Gumb za pokretanje opcije "Vježba"
Izvor: Autor

Pokreće se AJAX POST zahtjev prema datoteci *get_practice_questions.php*, kojoj se šalje identifikator jezika koji je bio odabran tijekom pritiska na gumb. Opcija „Vježba“ dostupna je ako je korisnik prošao barem jednu lekciju iz odabranog jezika, ili ako je netočno odgovorio na barem jedno pitanje iz jezika. U datoteci se prikupljaju identifikatori odgovarajućih pitanja te se inicijaliziraju potrebne sesijske varijable. Redoslijed prikupljenih pitanja izmiješan je pomoću PHP funkcije *shuffle*.

5.4. Način prikazivanja pitanja i odgovaranja na pitanja

Ukoliko su pitanja uspješno prikupljena, AJAX zahtjev prema datotekama *get_questions.php* ili *get_practice_questions.php* dobiva odgovor JSON oblika u kojem je atribut *status* postavljen na 1. U tom slučaju, korisnik je preusmjeren na datoteku *quiz.php*, gdje započinje rješavanje kviza s prikupljenim pitanjima. U suprotnom, ukoliko je atribut *status* postavljen na 0, korisnik je preusmjeren na datoteku *error.php* koja prikazuje poruku pogreške.

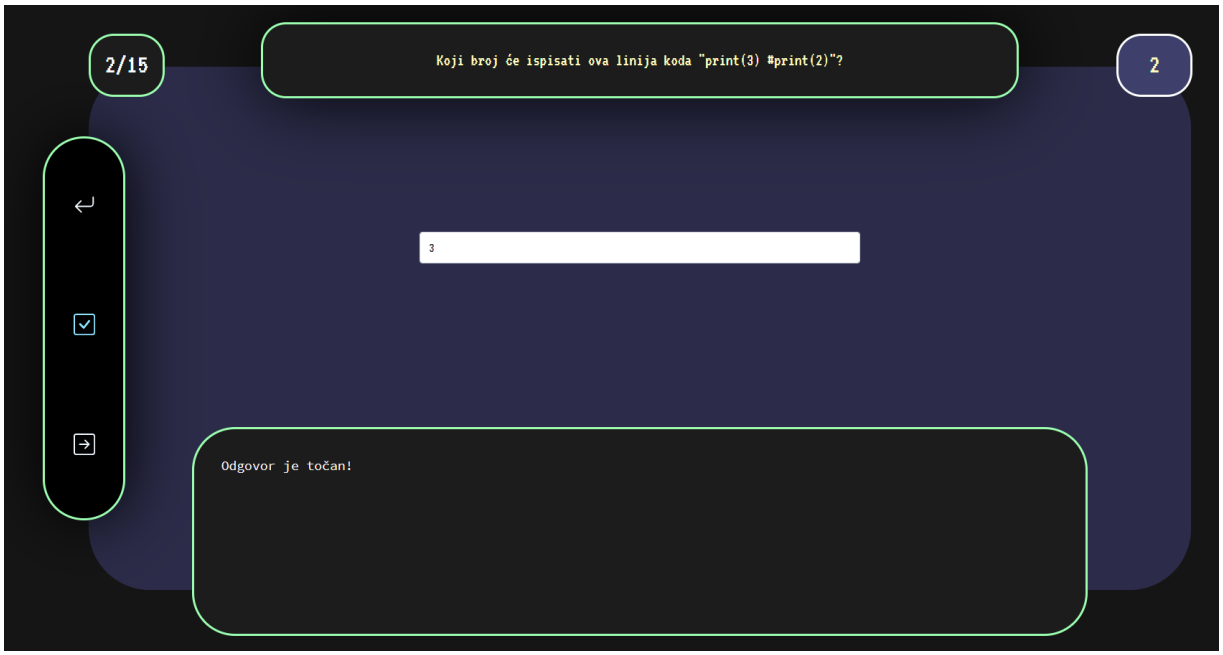
U datoteci *quiz.php*, iz sesijske varijable s prikupljenim pitanjima, uzima se pitanje na poziciji polja specificiranoj sesijskom varijablom *index*. Slika 21. prikazuje izgled postavljenog kviznog pitanja. Za svako pitanje u gornjem lijevom kutu prikazan je broj pitanja u odnosu na ukupan broj postavljenih pitanja, zatim sam tekst pitanja u sredini te broj preostalih života u gornjem desnom kutu. S lijeve strane nalaze se gumbi za izlaz iz kviza, slanje odgovora i preskakanje odgovora. U sredini su, ovisno o vrsti pitanja, ponuđeni odgovori ili forme za upis odgovora ili koda.

Slika 21. Postavljeno kvizno pitanje
Izvor: Autor

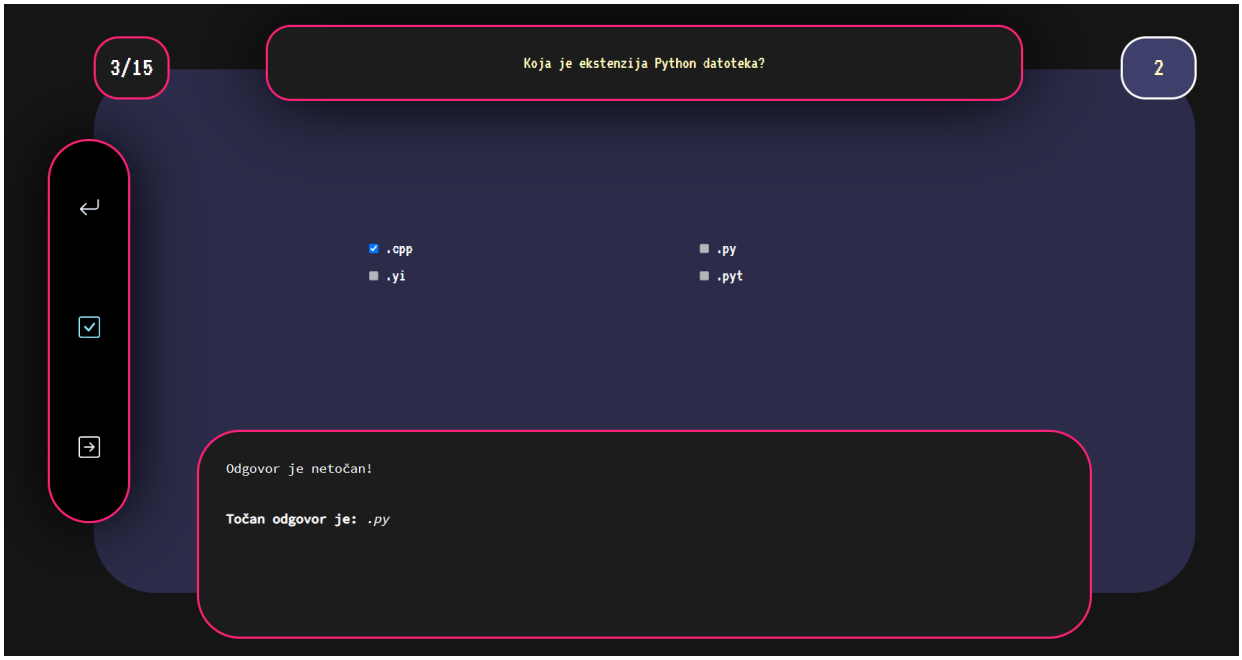
Gumb za slanje odgovora onemogućen je dokle god nije odabran barem jedan ponuđeni odgovor, ili forme za upis odgovora nemaju upisan tekst. Gumb za izlaz vraća korisnika na stranicu s koje je došao. Treći gumb za preskakanje pitanja dostupan je samo ako korisnik nije pritisnuo gumb za slanje odgovora. Pritiskom na gumb za preskakanje pitanja korisniku se prikazuje tekst

točnog odgovora te gumb za preskakanje pitanja mijenja izgled ikone i postaje gumb za prelazak na iduće pitanje. Gumb za prelazak na iduće pitanje osvježava stranicu čime se učitava novo pitanje. Osvježavanje stranice učitava novo pitanje jer se pri slanju odgovora ili preskakanju pitanja uvećava sesijska varijabla *index* koja označava poziciju pitanja koje se dohvaća iz polja s prikupljenim pitanjima.

Pritiskom na gumb za slanje odgovora, treći gumb za preskakanje pitanja zamjenjuje se s gumbom za prelazak na iduće pitanje. Poslani odgovor analizira se te se korisniku prikazuje je li odgovor točan ili netočan, uz tekst ispravnih odgovora ukoliko je poslani odgovor netočan. Slike 22. i 23. prikazuju izgled prikaza ovisno o tome je li poslani odgovor točan ili netočan.



Slika 22. Prikaz ukoliko je poslani odgovor točan
Izvor: Autor



Slika 23. Prikaz ukoliko je poslani odgovor netočan
Izvor: Autor

Korisniku će se ispod teksta pitanja prikazati potvrdni okviri ako je postavljeno pitanje s jednim ili više ponuđenih točnih odgovora, kao na Slici 21. Ako je pitanje vrste „nadopunjavanje“, prikazat će se forma za upis odgovora kao na Slici 22. Za pitanja vrste „kodiranje“, dodaje se potreban JavaScript kod te CSS stilovi za stvaranje uređivača koda (engl. *code editora*) pomoću biblioteke CodeMirror. Primjer pitanja vrste „kodiranje“ prikazan je na slici 24.

Slika 24. Pitanje sa kodiranjem i postavljenim editorom
Izvor: Autor

5.5. Prikupljanje i analiza odgovora

Gumb za slanje odgovora pokreće AJAX zahtjev prema datoteci *store_answer.php*. Zahtjevu se u obliku JavaScript FormData objekta šalju podaci o pitanju te odgovori. Za pitanja vrste „kodiranje“ prije slanja zahtjeva datoteci *store_answer.php,* šalje se API poziv programskom sučelju JDoodle da bi se izvršio upisani programski kod. Podaci koji se šalju API pozivu prikazani su na slici 8. Nakon kreiranja API poziva, na ekran se ispisuje prikaz koji stvara upisani kod. JQuery metodom *then*, završetkom API poziva prema sučelju JDoodle, pokreće se AJAX zahtjev prema datoteci *store_answer.php*.

U datoteci *store_answer.php* provjerava se točnost poslanog odgovora. Proces provjere drugačiji je za svaku vrstu pitanja. Ako se radi o pitanju s ponuđenim odgovorima, iz baze podataka dohvaća se odgovor s identifikatorom koji je jednak vrijednosti poslanog odgovora. Za pronađeni odgovor testira se je li oznaka točnosti *correct* postavljena na 1 jedan, što bi značilo da je odabran odgovor točan.

Za pitanja na koja se odgovora upisivanjem teksta pronalaze se točni odgovori zapisani u bazi podataka. Za svaki od pronađenih točnih odgovora, PHP funkcijom *similar_text* provjerava se sličnost između upisanog odgovora i odgovora iz baze podataka.

Ako pitanje od korisnika zahtjeva upis programskog koda, pronalaze se točni odgovori zapisani u bazi podataka. Kao i kod pitanja s upisivanjem teksta, za svaki od pronađenih točnih odgovora, PHP funkcijom *similar_text* provjerava se sličnost između upisanog programskog koda i programskog koda iz baze podataka.

Za netočno odgovorena pitanja, u tablicu *user_progress_question* dodaje se redak s identifikatorom netočno odgovorenog pitanja te identifikatorom korisnika. Ukoliko je korisnik točno odgovorio na pitanje koje je jednom netočno odgovorio, iz iste tablice se briše redak s identifikatorom pitanja i identifikatorom korisnika.

5.6. Određivanje rezultata kviza

Kviz se prekida ukoliko je korisnik netočno odgovorio na 3 pitanja te se učitava stranica

neuspjeha, *failed.php*, na kojoj se prikazuje poruka o neuspješnom rješavanju kviza. Ukoliko je korisnik došao do kraja kviza s manje od 3 netočno odgovorena pitanja, učitava se stranica uspjeha, *passed.php*, na kojoj se prikazuje poruka o uspješnom rješavanju kviza, te ukoliko je prošao lekciju ili jezik, informacija o uspješnom prolazu lekcije ili jezika. Stranice uspjeha i neuspjeha prikazane su na slikama 25. i 26. Podaci o lekcijama ili jezicima koje je korisnik prošao spremaju se u tablice *user_progress_lesson* i *user_progress_language*, gdje je u svakom redku spremljen identifikator lekcije ili jezika te identifikator korisnika.

Slika 25. Izgled stranice uspjeha
Izvor: Autor

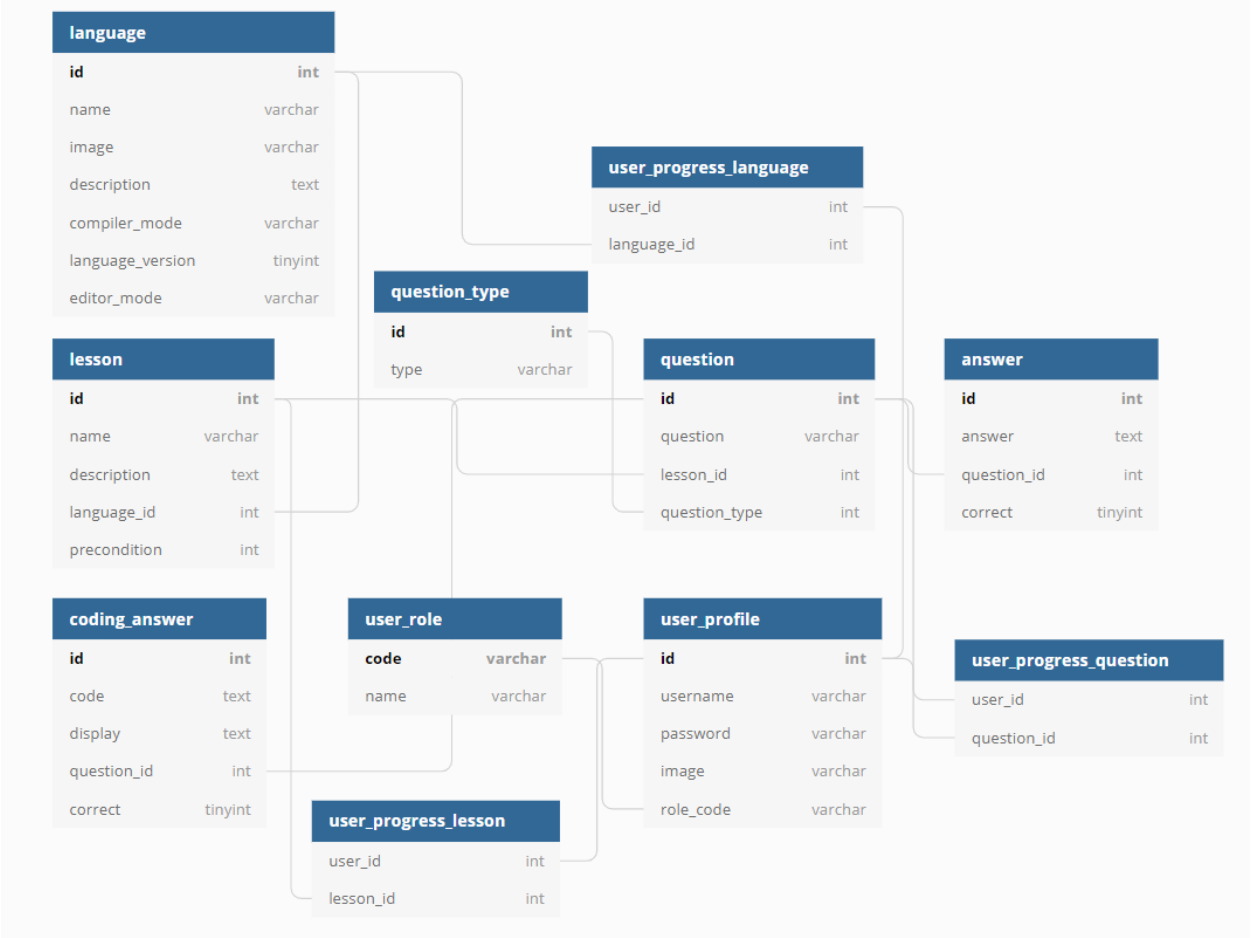
Slika 26. Izgled stranice neuspjeha
Izvor: Autor

5.7. Baza podataka

Baza podataka aplikacije sadrži tablicu *answer* s ponuđenim i mogućim točnim odgovorima. U tablicu *language* spremljeni su programski jezici. Tablica *lesson* sadrži lekcije za svaki programski jezik. Tablica *question* sadrži pitanja za svaku lekciju. Vrste pitanja zapisane su u tablicu *question_type*. U tablici *user_profile* nalaze se podaci o korisničkim profilima. Lozinka se pretvara u kriptirani oblik i šalje u bazu podataka pri registraciji ili dodavanju novog korisnika pomoću PHP funkcije *password_hash* i algoritma *PASSWORD_BYCRIPT*. Rezultat funkcije uvijek je podatak od 60 znakova, a započinje znakovima *„$2y$“*.

Tablice *user_progress_lesson* i *user_progress_language* služe za bilježenje jezika ili lekcija koje je korisnik uspješno prošao. Tablica *user_progress_question* služi za bilježenje pitanja koja je korisnik netočno odgovorio. Kada korisnik netočno odgovori na kvizno pitanje ili ga preskoči, u tablicu *user_progress_question* dodaje se redak s identifikatorom netočno odgovorenog ili preskočenog pitanja te identifikatorom korisnika. Ukoliko je korisnik točno odgovorio na pitanje koje je bilo netočno odgovoreno ili preskočeno, iz tablice se briše redak s identifikatorom tog pitanja i identifikatorom korisnika. U tablice *user_progress_lesson* i *user_progress_language* novi redovi s identifikatorima lekcija ili jezika te identifikatorom korisnika dodaju se u slučaju da je korisnik prošao lekciju ili jezik. Prolaz lekcije određuje se ukoliko je korisnik prvim rješavanjem kviza prošao lekciju, ili ako se među korisnikovim netočno odgovorenim pitanjima više ne nalazi nijedno pitanje iz lekcije. Prolaz jezika određuje se ukoliko je korisnik prošao sve lekcije koje se nalaze unutar programskog jezika.

U tablici *user_role* nalaze se moguće korisničke ovlasti. Dodani podaci u tablici su Administrator, Moderator i Korisnik. Shema baze podataka prikazana je na slici 27.

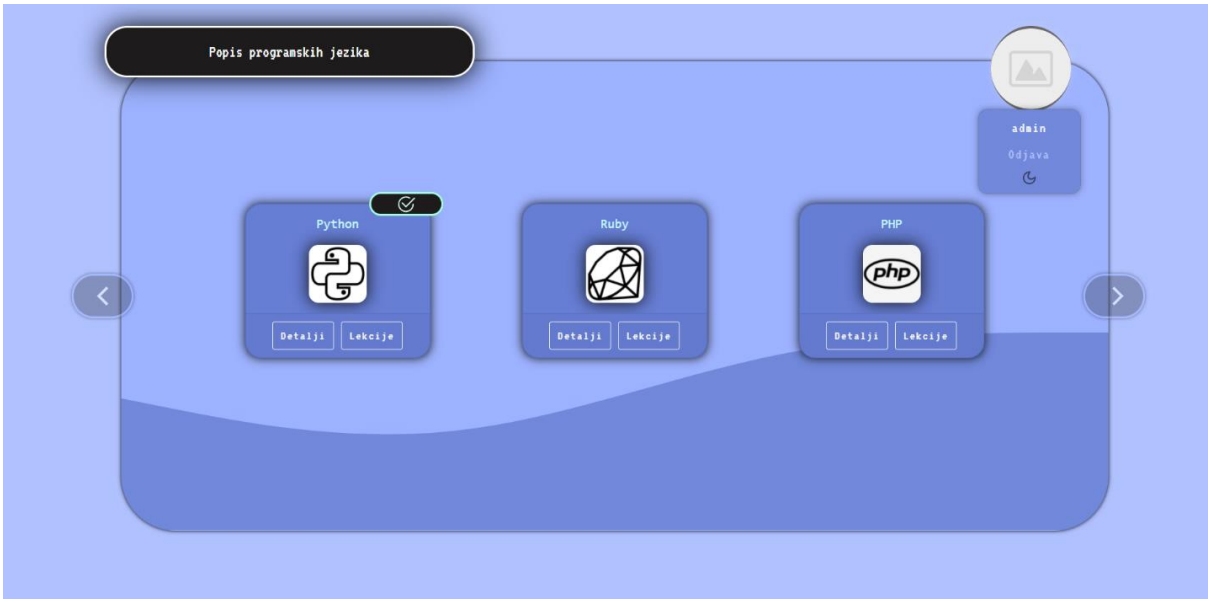


Slika 27. Shema baze podataka
Izvor: Autor

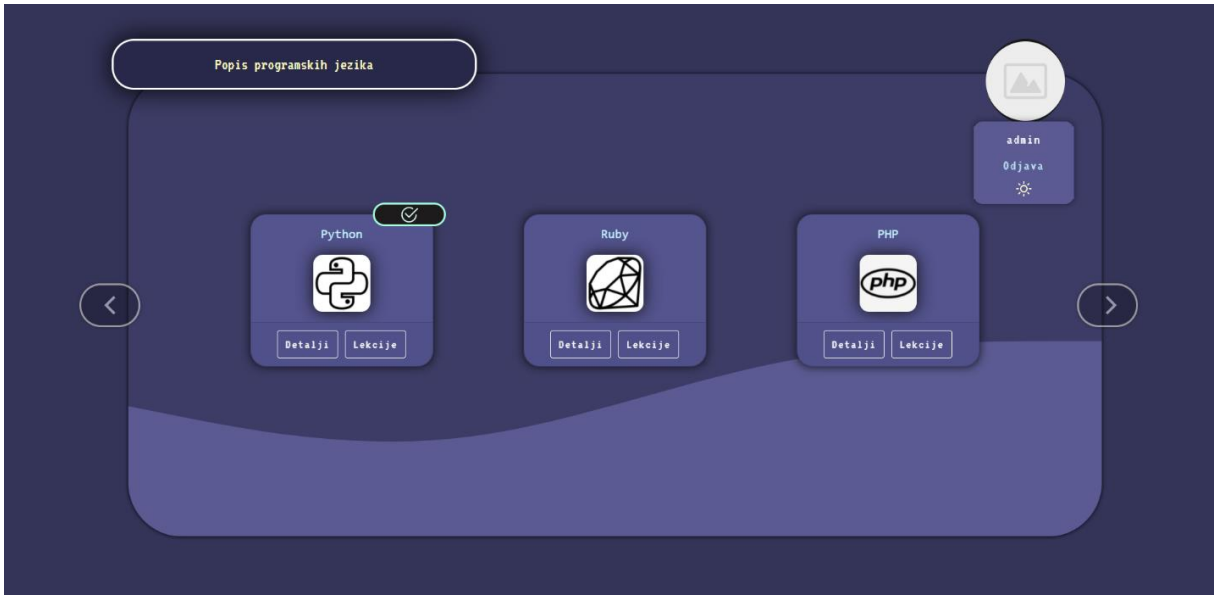
5.8. Funkcionalnost izmjene između tamnog i svijetlog načina rada

Tamni način rada postao je jedna od najtraženijih sastavnica dizajna web stranica i aplikacija. Prednosti tamnog načina rada mogu biti smanjena potrošnja baterije uređaja, smanjenje naprezanja očiju ili jednostavno estetika dizajna. U aplikaciji je implementirana funkcionalnost odabira između tamnog i svijetlog načina rada pomoću JavaScript svojstva *localStorage* koje omogućuje spremanje vrijednosti definiranih specificiranim nazivom u web preglednik. Spremljeni podaci ne brišu se zatvaranjem preglednika, nego ostaju dostupni kroz duže vrijeme.

Klikom na ikonu mjeseca ili sunca, koja se mijenja ovisno o odabranom načinu rada, na HTML element *body* dodaje se ili uklanja klasa *dark*. Promjena načina rada popraćena je i CSS animacijom pozadinske boje web aplikacije. Slike 28. i 29. prikazuju razliku između svijetlog i tamnog načina rada aplikacije.



Slika 28. Svijetli način rada
Izvor: Autor



Slika 29. Tamni način rada
Izvor: Autor

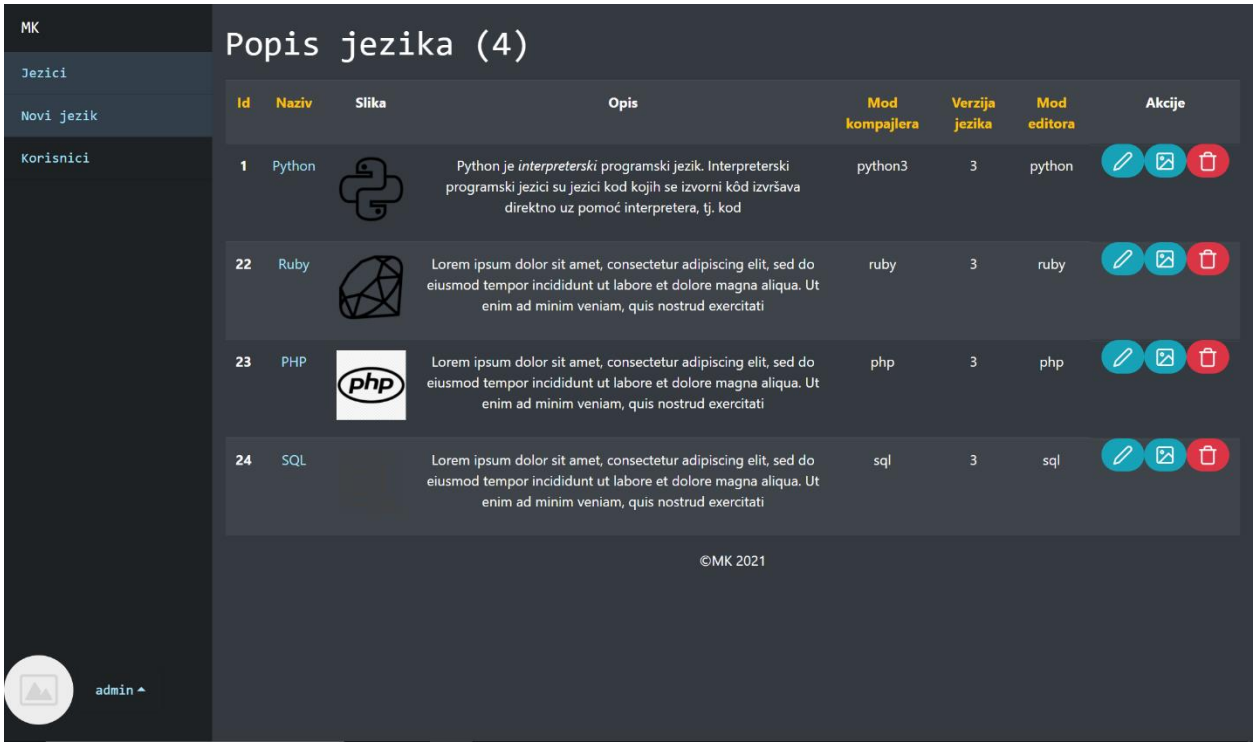
6. Administracija

Aplikacija sadrži administrativni dio koji služi za pregled i manipulaciju podataka o jezicima, lekcijama, pitanjima, odgovorima te korisnicima. Moguće je uređivanje postojećih podataka, brisanje podataka ili dodavanje novih jezika, lekcija, pitanja, odgovora ili korisnika. Pristup administrativnom dijelu dostupan je korisnicima s ovlastima Moderator i Administrator.

Registracijom na web aplikaciju, korisniku se automatski dodjeljuje ovlast Korisnik. Da bi novom korisniku bio dozvoljen pristup administrativnom dijelu, korisnik ovlasti Administrator mora mu postaviti ovlast na Moderator ili Administrator. Korisnicima ovlasti Administrator omogućen je pristup svim dijelovima administratorskog dijela. Ovlast Moderator može pristupiti svim dijelovima administrativnog dijela aplikacije osim dijelu namijenjenom za manipulaciju podacima o korisnicima.

6.1. Prikaz podataka

Na slici 30. prikazan je primjer prikaza popisa jezika u administracijskom dijelu aplikacije.



Slika 30. Popis jezika u administracijskom dijelu
Izvor: Autor

Izbornik s lijeve strane bojom označava kategoriju podataka koja je trenutno odabrana te se poveznica ispod odabrane kategorije mijenja ovisno o kategoriji. Primjerice, za odabranu kategoriju jezika, prikazana je poveznica za stvaranje novog jezika. Klikom na naziv jezika iz tablice (prikazan plavom bojom), prikazuje se lista lekcija iz odabranog jezika te su kategorija podataka sada lekcije, a prikazana je poveznica za dodavanje nove lekcije itd. Na slici je ulogirani korisnik ovlasti Administrator, pa je zato prikazana i poveznica za prikaz podataka o korisnicima.

Podaci iz odabrane kategorije prikazani su u tablici. Iznad tablice u zagradi naslova prikazan je ukupan broj prikazanih podataka. Žutom bojom prikazani su stupci tablice po kojima je moguće klikom na naziv stupca sortirati podatke. U primjeru na slici 31. moguće je sortirati pitanja po identifikatoru, tekstu pitanja i vrsti pitanja. Slika 31. prikazuje popis pitanja iz odabrane lekcije sortiran silazno po stupcu „Vrsta pitanja“.

Slika 31. Primjer popisa pitanja sortiranog po vrsti pitanja
Izvor: Autor

Na slici su ispod tablice s podacima vidljivi i linkovi za paginaciju. Ukoliko je broj prikazanih podataka veći od 10, podaci se dijele po stranicama od kojih svaka sadrži 10 podataka. Paginacija je implementirana na način da su u prikaz podataka uključeni i postavljeni kriteriji sortiranja.

Korisnicima ovlasti Administrator omogućen je prikaz i manipulacija podataka o korisnicima aplikacije. Prikazani podaci o korisnicima su identifikator korisnika, korisničko ime, slika, ovlast korisnika, broj riješenih jezika, broj riješenih lekcija, broj jezika u tijeku, broj lekcija u tijeku te broj netočno odgovorenih pitanja. Klikom na naziv ovlasti unutar tablice učitava se lista korisnika s odabranom ovlasti. Među gumbima za moguće akcije s lijeve strane pojavljuje se gumb za izmjenu korisničke ovlasti. Na slikama 32. i 33. prikazan je izgled popisa svih korisnika te popis

korisnika odabrane ovlasti.

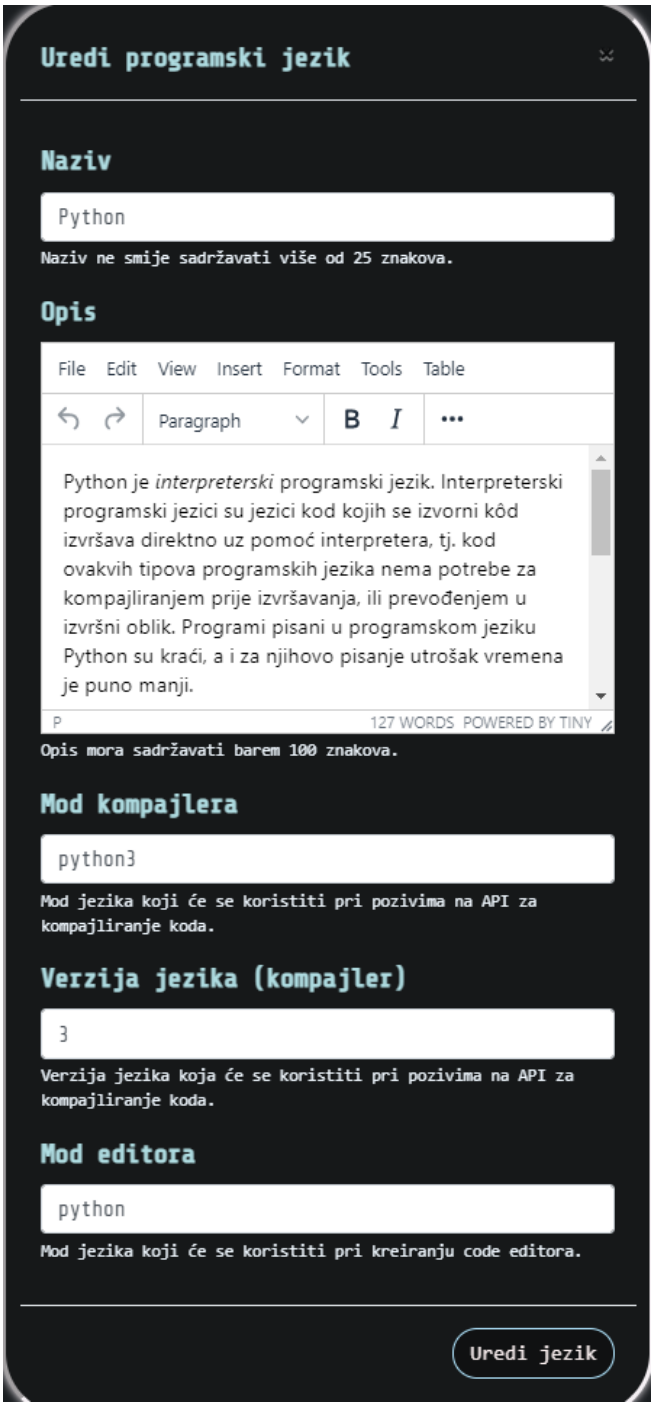
Slika 32. Popis korisnika u administracijskom dijelu
Izvor: Autor

Slika 33. Popis korisnika s ovlasti moderator
Izvor: Autor

6.2. Manipulacija podacima

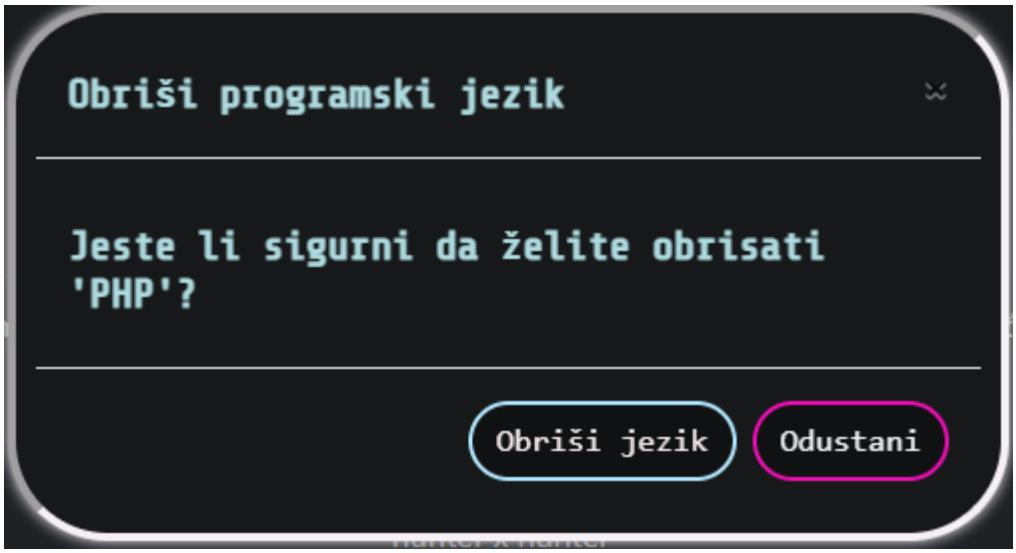
Posljednji stupac tablice s podacima sadrži gumbe za uređivanje pojedinačnih podataka iz odabrane kategorije. Klikom na gumbe otvara se modalni prozor u kojem je moguće izvršiti odabranu akciju. Modalni prozor za uređivanje programskog jezika sadrži uređivač bogatog teksta za uređivanje opisa jezika. Kreiran je pomoću JavaScript dodatka *tinymce*. Modalni prozor za izmjenu fotografije jezika ima onemogućen gumb za uređivanje fotografije. To je zato jer u

prikazanom primjeru programski jezik već ima postavljenu fotografiju jezika. Da bi se gumb omogućio, prvo je potrebno obrisati postojeću fotografiju klikom na gumb „Obriši fotografiju“. Navedeni modalni prozori, te modalni prozor za brisanje programskog jezika prikazani su na slikama 34., 35. i 36.



Slika 34. Modalni prozor za uređivanje fotografije programskog jezika
Izvor: Autor

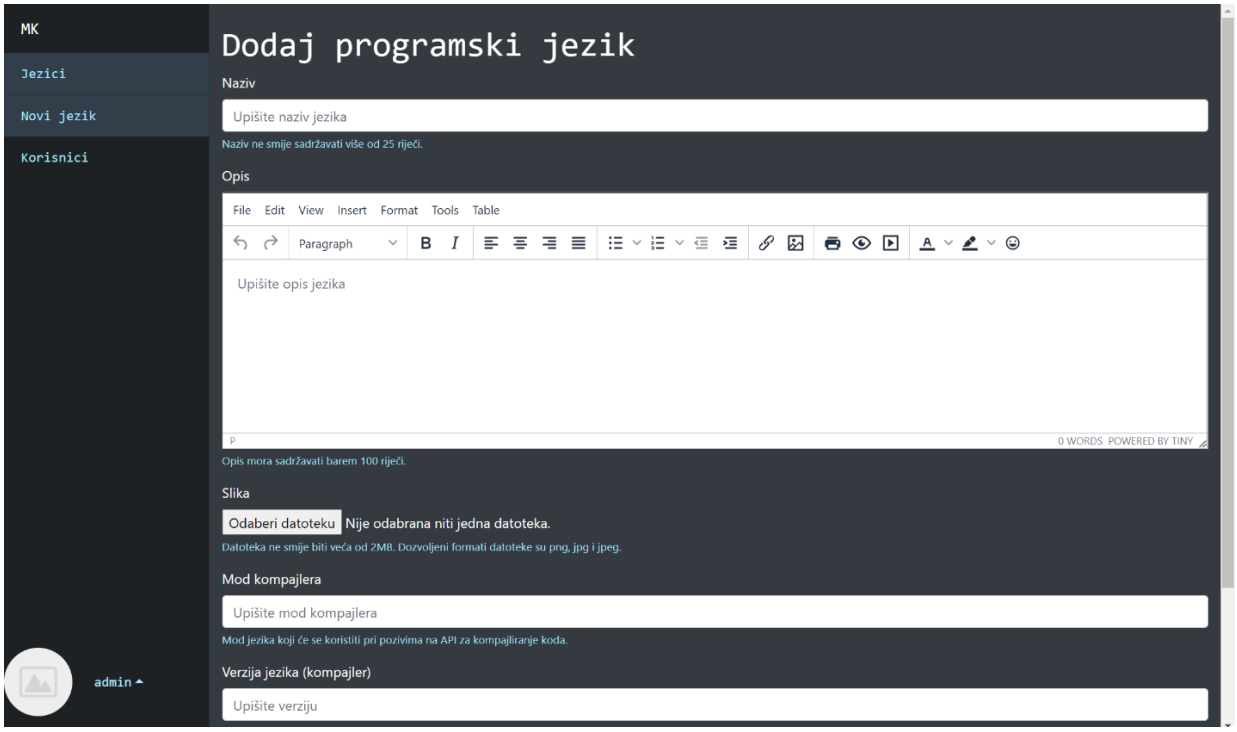
Slika 35. Modalni prozor za uređivanje programskog jezika
Izvor: Autor



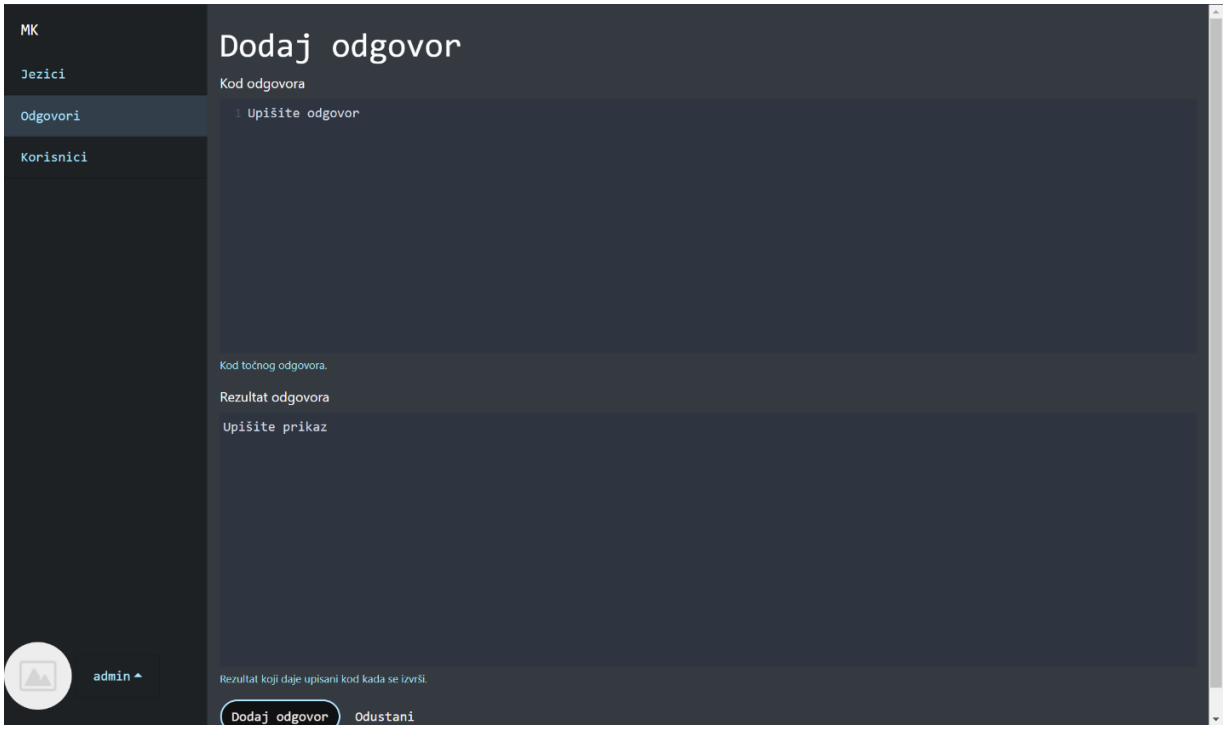
Slika 36. Modalni prozor za brisanje programskog jezika
Izvor: Autor

Klikom na gumbe za izvršavanje akcija unutar modalnih prozora u JavaScriptu prikupljaju se upisani podaci, dodaju u *FormData* objekt, te se pokreće AJAX poziv prema datoteci namijenjenoj za odabranu akciju. U datoteci se obrađuju poslani podaci, vrši odabrana akcija, te ovisno o uspješnosti izvršene akcije, unutar modalnog prozora prikazuje se prikladna poruka.

Osim uređivanja postojećih podataka, u administrativnom dijelu moguće je i dodavanje novih jezika, lekcija, pitanja, odgovora ili korisnika. Na slikama 37. i 38. prikazani su obrasci za dodavanje novog jezika te dodavanje točnog odgovora za pitanje koje zahtijeva upis programskog koda. Polja za upis programskog koda i rezultata izvršavanja programskog koda pretvorena su u uređivače koda pomoću biblioteke *CodeMirror*.



Slika 37. Obrazac za dodavanje programskog jezika
Izvor: Autor



Slika 38. Obrazac za dodavanje odgovora za pitanje s kodiranjem
Izvor: Autor

7. Sigurnost sustava

Uzevši u obzir napade kao što su Cross Site Scripting, Cross-Site Request Forgery i SQL Injection, poduzete su akcije za sigurnije prikazivanje sadržaja, slanje obrazaca te manipuliranje podacima iz baze podataka.

7.1. Cross Site Scripting (XSS)

Cross Site Scripting (XSS) je vrsta napada u kojem se zlonamjerne skripte „ubacuju“ u web stranicu preko obraza za upis na web stranici. „Ubačene“ skripte napisane su u HTML-u, JavaScriptu ili bilo kojem drugom jeziku koji web preglednik može izvršiti. Ovim napadom moguće je prikupiti privatne podatke s web stranice kao što su podaci o kolačićima ili informacije o sesiji. Moguća je i izmjena izgleda i ponašanja napadnute web stranice. Da bi se web aplikacija osigurala, potrebno je obratiti pozornost na sva mjesta preko kojih bi korisnikov upis mogao biti prikazan kroz HTML [17]. PHP funkcijom *htmlspecialchars*, određeni znakovi koji bi se mogli koristiti u XSS napadu pretvaraju se u HTML entitete. Time se postiže da web preglednik ne koristi tekst kao HTML element, što bi bio slučaj da znakovi nisu pretvoreni. Sadržaj teksta prikazuje se u cijelosti, ali dio teksta koji bi inače bio interpretiran kao HTML element, gubi svoju funkciju. Ovim se spriječava da napadači „ubace“ dijelove programskog koda preko HTML elemenata kao što su *script* ili *style*.

U administrativnom dijelu aplikacije omogućeno je dodavanje teksta (opisa jezika i lekcija) preko uređivača bogatog teksta (engl. *Rich text editora*). Tekst kreiran preko takvog uređivača teksta izgubio bi formatiranje ako se njegov tekst pretvori u HTML entitete. Da bi se na siguran način prikazivao tekst kreiran preko *Rich text editora,* koristi se dodatak za PHP *HTMLPurifier*, koji filtrira i čisti HTML tekst. Ovaj dodatak dozvoljava elemente i atribute koji bi se inače koristili u formatiranju teksta iz *Rich text editora* kao što su *<a>, , <b>, <code>, <h1>, <i>* itd, ali briše elemente i atribute koji bi se mogli koristiti pri XSS napadima.

7.2. Cross-Site Request Forgery

CSRF (engl. *Cross-Site Request Forgery*) je napad kod kojeg napadač u ime ovlaštenog korisnika pristupa nekom web odredištu. [18] Web stranica podložna je napadu ako ne provjerava izvor HTTP zahtjeva prije nego ga obradi i izvede, ili ako ne postoje nepredvidljivi parametri HTTP zahtjeva koje napadač ne može pogoditi. Napadom se mogu poslati podaci web stranici s

treće strane, ili pokrenuti zlonamjerne skripte bez znanja korisnika. Radi prevencije CSRF napada, u aplikaciji je generiran token pomoću PHP funkcije *random_bytes* koja izbacuje 32 nasumično generirana bajta. Bajtovi se iz binarnog u heksadekadski oblik pretvaraju funkcijom *bin2hex* te se nakon pretvorbe dobiva token od 64 nasumično generiranih znakova. Token se šalje pri svakom obrascu koji zahtjeva korisnikov upis, osim pri odgovaranju na kvizna pitanja. Ako je token istekao, izbacuje se poruka pogreške, te je potrebno osvježiti stranicu kako bi se generirao novi važeći token. Novi token generira se samo u slučaju da je postojeći već istekao, ili ako su sesijske varijable s informacijama o tokenu prazne, što je slučaj nakon uspješne validacije poslanog tokena.

7.3. SQL Injection

SQL *Injection* napadima moguće je zaobići potrebnu autentifikaciju te uređivati i brisati iz baze podataka. Ova vrsta napada može se spriječiti koristeći *PHP Data Objects (PDO)* pripremljene naredbe (engl. *Prepared statements*). Pripremljene naredbe pomažu jer se pri slanju SQL upita odvojeno šalju podaci i naredba. Dio unutar naredbe namijenjen za podatke označava se upitnikom te se za svaki upitnik dodaje odgovarajući podatak. Podaci se šalju odvojeno te ih sustav baze podataka interpretira isključivo kao podatke. Ovo nije slučaj kod običnih naredbi u kojima bi se bazi podataka poslao string iz kojeg sustav baze podataka naredbe i podatke razlikuje samo po navodnicima i sintaksi. Napadač bi kod „obične“ naredbe mogao upisati dijelove SQL naredbe kao podatak koji bi sustav baze podataka interpretirao kao naredbu te ju izvršio na bazi podataka. U primjeru koda PDO pripremljene naredbe sa slike 39., vrijednost identifikatora objekta je podatak, a funkcijom *bindParam* dodaje se vrijednost podatka.

```
        $query = "SELECT l.image FROM " . $this->table . " l WHERE l.id=?";

        $stmt = $this->conn->prepare($query);
        $stmt->bindParam(1, $this->id);
```

Slika 39. Primjer PDO pripremljene naredbe
Izvor: Autor

8. Zaključak

Rad prikazuje proces izrade edukativne aplikacije za učenje programskih jezika. Korišteno je više web tehnologija i alata. HTML, CSS i JavaScript, uz biblioteke JQuery, CodeMirror, Bootstrap, Typed.js te Slick.js korišteni su za stvaranje korisničkog sučelja, poboljšavanje korisničkog iskustva te dodavanje dinamičnosti web aplikaciji. JDoodle, Node.js i CORS Anywhere pozadinske su tehnologije kojima je ostvarena funkcionalnost izvršavanja upisanog programskog koda unutar aplikacije. Sve navedene tehnologije spojene su u funkcionalnu cjelinu preko PHP jezika, dok se potrebni podaci spremaju i dohvaćaju iz MySQL baze podataka.

Učenje programiranja može biti zahtjevno, ali uz aplikacije slične opisanoj u radu, proces učenja može postati zabavniji i efikasniji. Prolaskom lekcija i jezika kroz kvizna pitanja stvara se dojam učenja kroz igru. Pitanja su kreirana s ciljem da se bolje upamti sadržaj lekcije. Odgovarajući na pitanja koja zahtijevaju donošenje zaključaka temeljem sadržaja iz lekcije, korisnika se potiče da bolje zapamti ključne dijelove lekcije nego što bi to možda bio slučaj isključivo čitanjem teorije.

9. Popis literature

[1] https://www.educationworld.com/a_news/interactive-learning-helps-students-learn-six-times-more-moocs-study-says-1058054973 (30. 8. 2021.)

[2] http://www.oblakznanja.com/2013/04/html-dokument-i-osnove-html-jezika/ (12. 6. 2021.)

[3] https://hr.wikipedia.org/wiki/CSS (12. 6. 2021.)

[4] https://en.wikipedia.org/wiki/JQuery (12. 6. 2021.)

[5] https://en.wikipedia.org/wiki/Ajax_(programming) (12. 6. 2021.)

[6] https://en.wikipedia.org/wiki/XMLHttpRequest (12. 6. 2021.)

[7] https://www.w3schools.com/xml/tryit.asp?filename=tryxml_httprequest (30. 8. 2021.)

[8] https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/JSON (12. 6. 2021.)

[9] https://w3techs.com/technologies/details/pl-php (30. 8. 2021.)

[10] https://www.w3schools.com/php/php_intro.asp (14. 6. 2021.)

[11] http://htmlpurifier.org/ (24. 8. 2021.)

[12] https://codemirror.net/ (14. 6. 2021.)

[13] https://github.com/tinymce/tinymce (24. 8. 2021.)

[14] https://code.visualstudio.com/docs (15. 6. 2021.)

[15] https://www.w3schools.com/whatis/whatis_npm.asp (15. 6. 2021.)

[16] https://www.javatpoint.com/xampp (15. 6. 2021.)

[17] https://owasp.org/www-community/attacks/xss/ (19. 8. 2021.)

[18] https://www.cis.hr/www.edicija/CSRFnapadi.html (19. 8. 2021.)

Prilozi

Popis slika