**MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU**

**PREDDIPLOMSKI STRUČNI STUDIJ RAČUNARSTVA**

**MATIJA KRAJAČIĆ**

**WEB APLIKACIJA ZA POMOĆ PRI UČENJU PROGRAMSKIH JEZIKA**

**SEMINAR A**

**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.**

**SADRŽAJ**

[1. UPOTREBLJENE TEHNOLOGIJE 4](#_Toc74909494)

[1.1. HTML5 4](#_Toc74909495)

[1.2. CSS3 6](#_Toc74909496)

[1.3. AJAX 7](#_Toc74909497)

[1.3.1 JavaScript 7](#_Toc74909498)

[1.3.3.1. JQuery 8](#_Toc74909499)

[1.3.2. XMLHttpRequest 8](#_Toc74909500)

[1.3.3. JSON 9](#_Toc74909501)

[1.4. PHP 9](#_Toc74909502)

[1.4.1. PDO 9](#_Toc74909503)

[1.5. MySQL 10](#_Toc74909504)

[1.6. JDoodle 10](#_Toc74909505)

[1.7. Node.js 11](#_Toc74909506)

[1.7.1. CORS Anywhere 11](#_Toc74909507)

[1.8. CodeMirror 12](#_Toc74909508)

[1.9. Bootstrap 12](#_Toc74909509)

[2. KORIŠTENO RAZVOJNO OKRUŽENJE I ALATI 14](#_Toc74909510)

[2.1. Visual Studio Code 14](#_Toc74909511)

[2.2. NPM 14](#_Toc74909512)

[2.3. XAMPP 14](#_Toc74909513)

[3. PLAN APLIKACIJE 15](#_Toc74909514)

[3.1. BAZA PODATAKA 16](#_Toc74909515)

[3.2. SUSTAV POSTAVLJANJA PITANJA 17](#_Toc74909516)

**1. UPOTREBLJENE TEHNOLOGIJE**

**1.1. HTML5**

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 elemenata web stranice. HTML oznake interpretira, tj. tumači web preglednik, te stvara prikaz web stranice. Oznake u HTML-u zapravo su naredbe web pregledniku kojima se određuje prikaz sadržaja web stranice. Svaka HTML oznaka navodi se unutar znakova *<* i *>*. 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>*. Negi tagovi, kao što su *<br>* za novi red ili *<hr>* za liniju ne moraju se zatvarati, međutim, radi kompatibilnosti sa XHTML jezikom preporučljivo ih je zatvarati dodavanjem desne kose crte prije *>* oznake pri otvaranju taga, primjerice *<br />*.

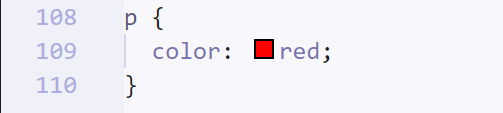
Početni tag HTML dokumenta je *<html>*, u kojem se obično nalaze dva osnovna dijela: zaglavlje i tijelo. Zaglavlje se definira oznakom *<head>* te sadrži osnovne informacije o dokumentu kao što su naslov i metapodaci te pozive potrebnih CSS ili skriptnih datoteka. HTML oznake mogu sadržavati atribute, koji dodatno opisuju određena 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.



Slika 1 Jednostavan HTML dokument sa zaglavljem i tijelom

**1.2. CSS3**

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



Slika 2 Primjer dohvaćanja elementa unutar CSS-a

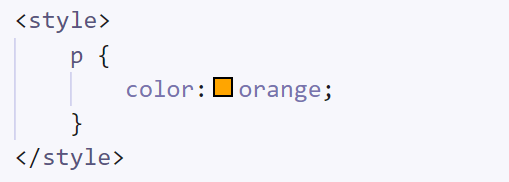
Na prikazanom primjeru dohvaćen je paragraf element p. Unutar vitičastih zagrada navodi se svojstvo elementa koje se želi promijeniti. Nakon navođenja svojstva elementa potrebna je dvotočka prije upisivanja željene vrijednosti svojstva. Nakon vrijednosti svojstva potrebno je dodati znak točke sa zarezom.

Postoji nekoliko načina uvođenja CSS stilova u HTML dokument. Jedan od načina je upis svojstava unutar HTML atributa *style*. Željenom HTML elementu unutar atributa style upisuju se CSS svojstva i vrijednosti.



Slika 3 Primjer upisa CSS svojstva i atributa pomoću HTML atributa style

Idući način uvođenja CSS stilova je pomoću HTML oznake *<style>*. Ova oznaka obično se dodaje u zaglavlje HTML dokumenta. Između otvaranja i zatvaranja oznake upisuju se CSS svojstva i vrijednosti.



Slika 4 Primjera upisa CSS svojstva i atributa pomoću HTML taga style

Posljednji mogući način uvođenja CSS stilova je pomoću HTML oznake *<link>*. I ova oznaka obično se dodaje u zaglavlje HTML dokumenta. Atributom *rel* specificira se odnos između trenutnog dokumenta i dokumenta koji se poziva oznakom *<link>*. Atributom *href* navodi se putanja do CSS datoteke koja se želi učitati.



Slika 5 Primjer učitavanja CSS svojstva i atributa pomoću HTML taga link

**1.3. AJAX**

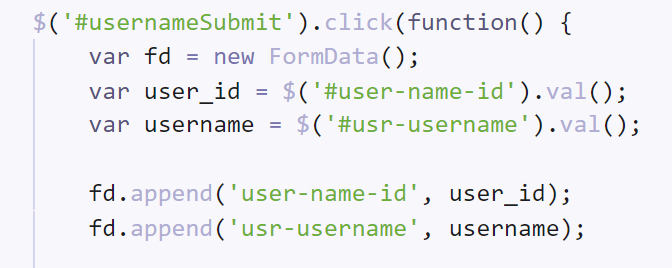
Asynchronous JavaScript and XML ili AJAX, set je web tehnologija za kreiranje asinkronih web aplikacija. Pomoću AJAX-a, web aplikacije mogu slati i primati podatke sa servera u pozadini bez utjecaja na prikaz i ponašanje postojeće stranice. AJAX omogućuje web stranicama i aplikacijama da mijenjaju sadržaj dinamički bez potrebe za ponovnim učitavanjem cijele stranice. HTML i CSS ubrajaju se u AJAX tehnologije jer se koriste za stiliziranje informacija. Uz njih, u AJAX tehnologije spadaju JavaScript, JSON i XMLHttpRequest. U samom nazivu AJAX-a spominje se XML, međutim u modernim aplikacijama i web stranicama češće se koristi JSON umjesto XML-a.

**1.3.1 JavaScript**

JavaScript je najpoznatiji kao skriptni jezik za web stranice, međutim, korišten je i u mnogim okolinama izvan weba, kao što su Node.js ili Adobe Acrobat. JavaScript istovremeno 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. Primjeri ponašanja koja je moguće implementirati pomoću JavaScripta su animacije nestajanja ili povećavanja sadržaja na web stranici, igre unutar web preglednika, kontrola reproduciranja multimedijskih sadržaja, validacija upisanih podataka prije slanja serveru itd. Preko 80% web stranica koristi JavaScript bibilioteke i programske okvire. Samo neki od najpopularnijih su jQuery, React i Angular.

**1.3.3.1. 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. Na taj objekt zatim je moguće pozivati funkcije ili rukovati događajima.



Slika 6 Primjer dodavanja funkcionalnosti pomoću jQuerya za slučaj pritiska na gumb

**1.3.2. XMLHttpRequest**

XMLHttpRequest je aplikacijsko programsko sučelje (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 7 Primjer korištenja XMLHttpRequest objekta za dohvaćanje odgovora servera

**1.3.3. JSON**

JavaScript Object Notation ili JSON je tekstualni 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.

**1.4. PHP**

Hypertext Preprocessor ili PHP je skriptni jezik koji se izvršava na serveru, za razliku od jezika kao što su JavaScript. PHP je moguće koristiti za prikupljanje podataka iz formi (obrazaca). dinamičko generiranje sadržaja na web stranici, slanje i primanje kolačića, stvaranje sesija i mnoge druge stvari. PHP omogućuje pristup bazama podataka i komuniciranje sa raznim Internet servisima kao što su POP3, HTTP, SNMP, NNTP itd. Interpretiran i izvršen PHP kod zapravo formira HTTP odgovor (*response*). PHP se koristi u gotovo 80% web stranica čiji je serverski jezik poznat javnosti. Najnovija PHP verzija danas je PHP 8.0. PHP kod se ubacuje u HTML dokument otvaranjem *<?php* taga. Tag nije potrebno zatvoriti, ali je to moguće učiniti znakovima *?>*. Varijable u PHP jeziku započinju *$* znakom.

**1.4.1. PDO**

PHP Data Objects ili PDO, sučelje je za pristup bazama podataka dostupno od PHP verzije 5. PDO pruža abstrakcijski 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.



Slika 8 Primjer povezivanja na bazu podataka koristeći PDO sučelje

**1.5. 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 9 Primjer MySQL upita u PHP-u

**1.6. JDoodle**

JDoodle kompajler je online aplikacijsko programsko sučelje (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.



Slika 10 Primjer ulaznih podataka za API poziv, gdje je script upisan kod unutar postavljene forme

**1.7. Node.js**

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

**1.7.1. 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 sa web stranice, jer bi se time povjerljivi ulazni podaci *clientID* i *clientSecret* izložili nesigurnosti. Zbog toga se koristi Node.js proxy server CORS Anywhere, koji dodaje potrebna CORS zaglavlja na zahtjev.

**1.8. 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 boju i tabulatore kodu napisanom u odabranom programskom jeziku. Osim moda, 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.



Slika 11 Primjer kreiranja instance CodeMirror editora

**1.9. 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 da u *class* atribut elementa dodamo klasu *text-danger*.



Slika 12 Primjer dodavanja crvene boje na element

**2. KORIŠTENO RAZVOJNO OKRUŽENJE I ALATI**

**2.1. Visual Studio Code**

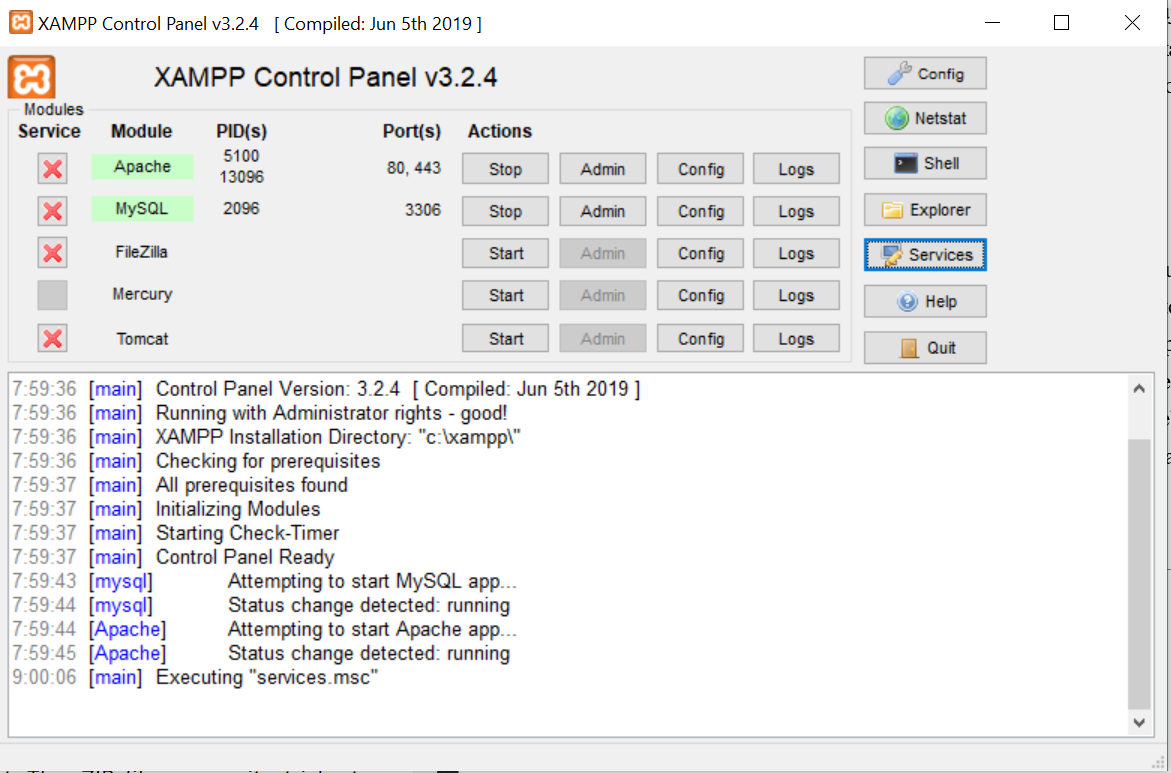
Visual Studio Code je uređivač koda otvorenog koda kreiran od strane Microsofta. Dolazi sa ugrađenom podrškom za JavaScript, TypeScript i Node.js. Omogućuje pronalažanje pogrešaka u kodu (debagiranje), označavanje sintakse, inteligentno dovršavanje koda, gotove isječke koda, refaktoriranje te ugrađenu GitHub podršku. Moguće je biranje između brojnih 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 i GO.

**2.2. NPM**

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

**2.3. XAMPP**

XAMPP je više platformski web server koji pomaže pri kreiranju i testiranju aplikacija na lokalnom web serveru. Razvijen je od strane Apache Friends. Uključuje Apache HTTP Server, MariaDB bazu podataka, te interpreter 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.



Slika 13 XAMPP upravljačka ploča

**3. PLAN APLIKACIJE**

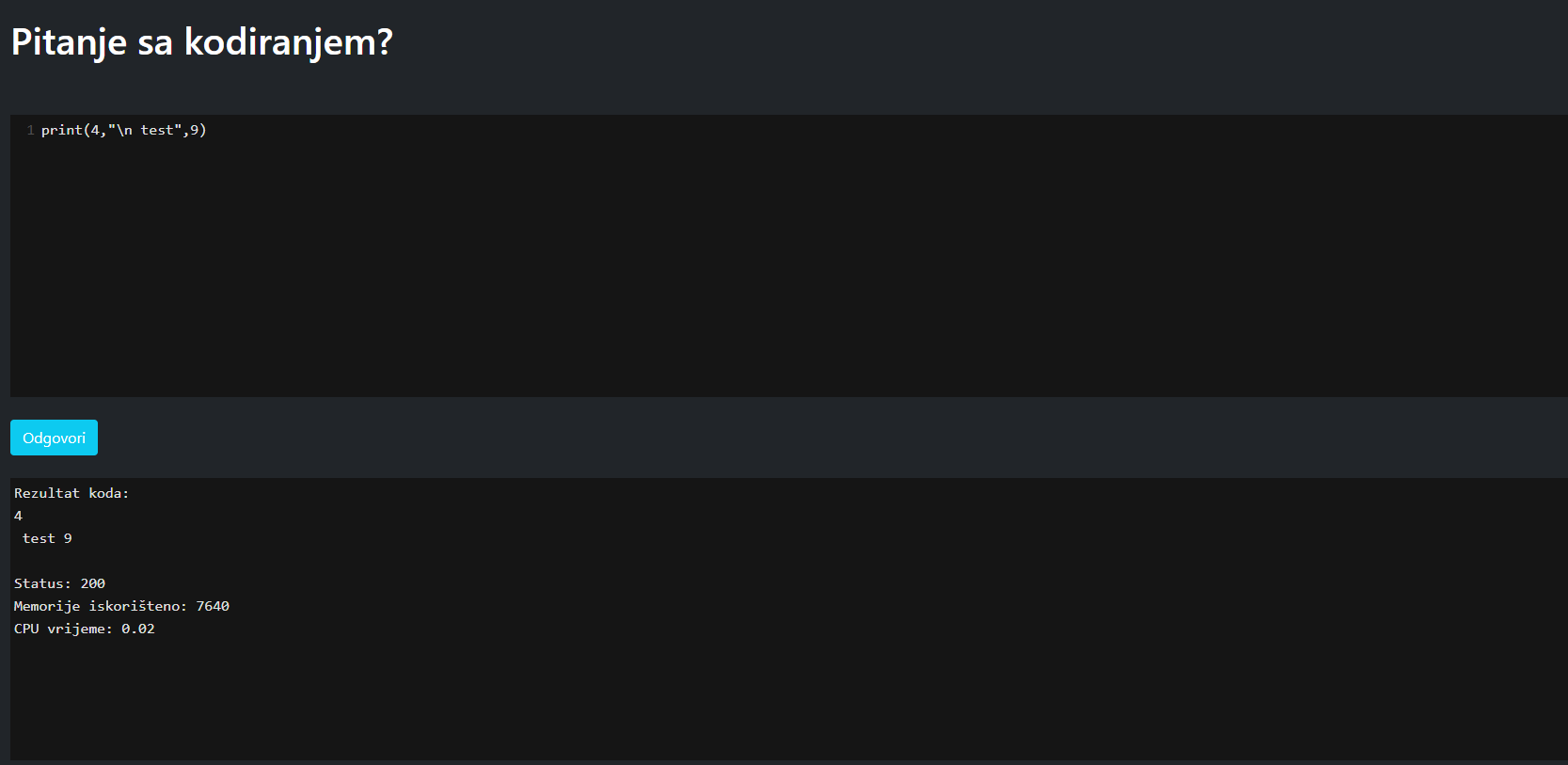
Cilj aplikacije je olakšavanje procesa učenja programskih jezika. Umjesto učenja sintakse i naredbi napamet, u ovoj se aplikaciji želi potaknuti učenje na pasivan, zabavniji način, kroz kvizna pitanja. Planirane vrste pitanja su pitanja sa jednim točnim odgovorom, pitanja sa više točnih odgovora, pitanja sa nadopunjavanjem (preko upisa) i pitanja koja uključuju kodiranje u jeziku koji je odabran za učenje. Svaki od programskih jezika podijeljen je na lekcije koje obuhvaćaju cjeline kao što su varijable unutar programskog jezika, izrada funkcija, tipovi podataka, strukture podataka, klase i objekti… Svaka od lekcija sadrži od 10-15 pitanja od kojih minimalno 5 pitanja zahtjeva upis koda programskog jezika u sustav. Svako pitanje za koje su ponuđeni odgovori sadrži netočne i točne odgovore, čiji se broj razlikuje ovisno o vrsti pitanja. Pitanja sa nadopunjavanjem i upisom koda sadrže moguće točne odgovore, sa kojima se uspoređuje upisani odgovor od strane korisnika. Pitanjima je moguće pristupiti ulaskom na željeni jezik, biranjem željene lekcije te klikom na gumb za pokretanje kviznih pitanja. Klikom na gumb pokreće se PHP sesija. Identifikatori netočno odgovorenih pitanja spremaju se u bazu podataka zajedno sa identifikatorom korisnika. Na njih se ponovo može odgovarati odabirom opcije *Vježba*. Odabirom te opcije, korisniku se postavljaju 15 pitanja na koja je ranije netočno odgovorio, te nasumično odabrana pitanja na koja je točno odgovorio, čiji broj ovisi o broju netočno odgovorenih pitanja. Da bi se lekcija uspješno prošla, potrebno je barem jednom točno odgovoriti na sva pitanja iz lekcije. Sve lekcije unutar programskog jezika korisniku nisu odmah dostupne. Prolaskom lekcije korisniku se otključavaju iduće lekcije čiji su preduvijet prolaz te lekcije. Kada je korisnik uspješno prošao lekciju, u bazu podataka se sprema identifikator korisnika, uz identifikator lekcije koju je prošao. Kada je korisnik uspješno prošao sve lekcije za programski jezik, u bazu podataka se sprema identifikator korisnika, uz identifikator jezika koji je prošao.

**3.1. BAZA PODATAKA**

Baza podataka aplikacije sadrži tablicu *answer* sa ponuđenim i mogućim točnim odgovorima. Svaki redak tablice *answer* sadrži identifikator *id*, odgovor *answer*, identifikator povezanog pitanja *question\_id*, povezan na tablicu sa pitanjima *question*, te oznaku točnosti *correct,* zadanu na 0. U tablicu *language* spremljeni su programski jezici. Uz identifikator *id*, tablica sadrži ime programskog jezika *name*, ograničeno na 25 znakova, putanju do fotografije programskog jezika *image* sa ograničenjem od 255 znakova te opis programskog jezika *description*. Tablica *lesson* sadrži lekcije za svaki programski jezik. Uz identifikator *id*, tablica *lesson* sadrži ime lekcije *name*, ograničeno na 100 znakova, opis lekcije *description*, identifikator povezanog jezika *language\_id*, povezan na tablicu sa jezicima *language* te identifikator lekcije čiji je prolaz preduvijet za dostupnost lekcije *precondition*. Ukoliko je vrijednost polja *precondition* 0, tada lekcija nema preduvijete za dostupnost. Tablica *question* sadrži pitanja za svaku lekciju. Uz identifikator *id*, tablica *question* sadrži pitanje *question,* identifikator lekcije za koju je pitanje namijenjeno *lesson\_id*, te identifikator vrste pitanja *question\_type*. Vrste pitanja zapisane su u tablicu *question\_type*. Svaki redak te tablice sadrži identifikator *id*, te vrstu pitanja *type*, ograničenu na 20 znakova. U tablici *user\_profile* nalaze se podaci o korisničkim profilima. Svaki redak tablice sadrži identifikator *id*, korisničko ime *username* koje ne smije biti duže od 15 znakova, kriptiranu lozinku *password* koja je ograničena na 60 znakova, putanju do slike profila *image* koja je ograničena na 255 znakova, te oznaku ovlasti, *role\_code,* povezanu na tablicu sa korisničkim ovlastima *user\_role*. Tablica *user\_progress\_language* služi za bilježenje jezika koje je korisnik uspješno prošao. Sadrži poveznicu na tablicu sa korisničkim profilima preko polja *user\_id*, poveznicu na tablicu sa jezicima preko polja *language\_id*, te oznaku izvršenosti *completed*, koja je zadana na 1, što označava prolaz. Tablica *user\_progress\_lesson* služi za bilježenje lekcija koje je korisnik uspješno prošao. Sadrži poveznicu na tablicu sa korisničkim profilima preko polja *user\_id*, poveznicu na tablicu sa lekcijama preko polja *lesson\_id*, te oznaku izvršenosti *completed*, koja je zadana na 1. Tablica *user\_progress\_question* služi za bilježenje pitanja koja je korisnik netočno odgovorio. Sadrži poveznicu na tablicu sa korisničkim profilima preko polja *user\_id*, poveznicu na tablicu sa pitanjima preko polja *question\_id*, te oznaku izvršenosti *completed*, koja je zadana na 0, što označava netočan odgovor. U tablici *user\_role* nalaze se moguće korisničke ovlasti. Svaki redak tablice sadrži skraćenu oznaku ovlasti *code*, te puno ime ovlasti *name* ograničeno na 15 znakova. Dodani podaci u tablici su Administrator, Moderator i Korisnik.

**3.2. SUSTAV POSTAVLJANJA PITANJA**

Klikom na gumb za pokretanje kviznih pitanja, koji se nalazi na stranici sa detaljnim opisom lekcije programskog jezika, pokreće se sustav za postavljanje pitanja. Pokreće se AJAX POST zahtjev prema datoteci *get\_questions.php*, kojoj se šalje identifikator odabrane lekcije. U toj datoteci pronalaze se sva pitanja koja su povezana sa odabranom lekcijom, te se identifikatori tih pitanja spremaju u sesijsku varijablu. Istovremeno, kreira se sesijska varijabla *index*, koja označava redni broj pitanja na koje korisnik odgovara. Datoteka *get\_questions.php* vraća odgovor JSON oblika sa 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 prebacuje na stranicu *quiz.php*. Pitanja se postavljaju pojedinačno. U datoteci *quiz.php* prolazi se kroz ranije kreiranu sesijsku varijablu sa identifikatorima pitanja. Ta sesijska varijabla je polje, a pozicija identifikatora koji se koristi specificirana je sesijskom varijablom *index*. Nakon što se pronađe pitanje sa identifikatorom na poziciji specificiranoj varijablom *index*, izgled obrasca za upis odgovora mijenja se ovisno o vrsti pitanja. Klikom na gumb za slanje odgovora, pokreće se AJAX POST zahtjev prema datoteci *store\_answer.php*, kojoj se šalju upisani (ili označeni) odgovori. Datoteka provjerava točnost poslanih odgovora, te ovisno o točnosti vraća JSON vrijednosti i dodaje ili briše iz baze podataka podatke o korisnikovom napredku. Ukoliko je odgovor netočan, u tablicu *user\_progress\_question* dodaje se novi redak sa identifikatorom korisnika i identifikatorom pitanja. Ukoliko je odgovor točan, a u tablici postoji zapis o netočnom odgovoru, taj zapis se iz tablice briše. Ukoliko poslani odgovor nije prazan, neovisno o tome je li korisnik odgovorio točno ili netočno, sesijska varijabla *index* povećava se za 1, te se datoteka *quiz.php* ponovno učitava sa idućim pitanjem unutar sesijskog polja sa identifikatorima pitanja. Kada je sesijska varijabla *index* jednaka veličini sesijskog polja sa identifikatorima pitanja, to znači da je posljednje pitanje odgovoreno, te da je kviz za odabranu lekciju gotov. U tom slučaju korisika se šalje natrag na detalje odabrane lekcije.



Slika 14 Primjer testnog pitanja sa upisivanjem koda te ispisa izvršenog koda