Web platforma za zadavanje i rješavanje programskih problema

Gudelj, Tomislav

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TENOLOGIJA

Sveučilišni studij

WEB PLATFORMA ZA ZADAVANJE I RJEŠAVANJE PROGRAMSKIH PROBLEMA

Diplomski rad

Tomislav Gudelj

Osijek, 2018.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 03.10.2018.

Ime i prezime studenta:	Tomislav Gudelj
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D 847 R, 26.09.2017.
Ephorus podudaranje [%]:	4

Ovom izjavom izjavljujem da je rad pod nazivom**: Web platforma za zadavanje i rješavanje programskih problema**

izrađen pod vodstvom mentora Izv. prof. dr. sc. Alfonzo Baumgartner

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA

Ja, Tomislav Gudelj, OIB: 06532544170, student/ica na studiju: Diplomski sveučilišni studij Računarstvo, dajem suglasnost Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijek da pohrani i javno objavi moj **diplomski rad**:

Web platforma za zadavanje i rješavanje programskih problema

u javno dostupnom fakultetskom, sveučilišnom i nacionalnom repozitoriju.

Osijek, 03.10.2018.

_____

potpis

SADRŽAJ

1. UVOD

Internet je prepun programskih prevoditelja namijenjenih prevođenju izvornog koda u raznim programskim jezicima. Navedeni prevoditelji uzeli bi korisnikov kod, preveli ga u izvršni (binarni) kod, izvršili i zatim rezultat izvršenja vratili korisniku. No, Internet definitivno nije prepun platformama koje istovremeno nude i opisanu funkcionalnost i repozitorij programskih problema i zadataka koje bi znatiželjni korisnik mogao riješiti pisanjem programskog koda. Tijekom posljednjeg desetljeća postojala je jedna takva platforma imenom Z-trening, ali je uslijed nepoznatih razloga danas nedostupna. Cilj ovog diplomskog rada je razviti jednu web platformu koja omogućava ne samo osnovno prevođenje i izvršenje programskog koda u jeziku C++, nego i razvijanje vlastitih vještina razvoja algoritama kroz rješavanje postojećih programskih problema. Koristeći ovu platformu, korisnik bi mogao ne samo pokrenuti svoj proizvoljni C++ kod na udaljenom računalu, nego i pokušati pisati programski kod posebno osmišljen radi rješavanja programskih problema zadanih od strane sustava, ali također i osmisliti i doprinijeti svoj vlastiti programski problem na mrežu. Ova platforma testirat će korisnikov programski kod na testne slučajeve, bodovati točnost programskog koda, korisnika izvijestiti o točnosti, rezultat dokumentirati na lokalnu i, indirektno, opću rang listu i korisnika prezentirati navedenim rang listama. Korisnik će moći koristiti sustav koristeći vlastiti, osobni račun kojega može stvoriti odmah na licu mjesta ili koristiti sustav kao posjetitelj, bez prijave.

1.1. Zadatak završnog rada

Potrebno je napraviti web aplikaciju koja će omogućiti zadavanje programskih problema po uređenom formatu (ulazi/izlazi). Platforma treba omogućiti korisnicima zadavanje, ali i rješavanje programskih problema u određenim programskim jezicima (npr. C/C++), te njihovo bodovanje te prikaz rezultata. Obratiti pozornost na sigurnost ovog višekorisničkog sustava.

1.2. Porijeklo ideje i vrijednost koju ovaj rad pruža

Hrvatsko otvoreno natjecanje u informatici (HONI, u daljnjem tekstu: HONI)[1] jedanaestogodišnji je neprofitni projekt pod pokroviteljstvom udruge Hrvatski savez informatičara (HSIN)[2]. Hrvatski savez informatičara – HSIN je najviše nacionalno strukovno tijelo, neprofitna, nevladina i nestranačka udruga, kojoj je sustavan rad s mladim informatičarima s naglaskom na darovite, jedna od temeljnih zadaća.[3] HONI je informatičko natjecanje u programiranju, odnosno rješavanju programskih problema pisanjem programskog koda u nekoliko programskih jezika, među kojima su C, C++ i Pascal. Natjecanje HONI sastojalo se od zadataka – programskih problema koje bi natjecatelj trebao riješiti pisanjem programskog koda na svojem vlastitom računalu, testiranjem vlastitog programskog koda koliko god mu je potrebno puta i konačnim slanjem tog istog programskog koda HSIN-u unutar vremenskog ograničenja tijekom trajanja natjecanja u sklopu dana tijekom kojeg se natjecanje održavalo. Program je trebao veoma precizno ispisati rezultate rada, a odstupanja od očekivanih rezultata uglavnom su se smatrala neispravnim rezultatima. Tijekom jednog kola, natjecatelj bi bio prezentiran nekolicinom zadataka i pripadajućim test primjerima kao i pomoćnim podacima. Sukladno podacima i uz pomoć test primjera, natjecatelj bi morao napisati programski kod za kojeg smatra da vraća ispravne i kvalitetne rezultate rada. Njegov programski kod bi ultimativno bio testiran na takozvane testne podatke – uređene parove podataka prema modelu ulaz – izlaz. Rukovoditelj HONI-ja, HSIN, ultimativni autor svih zadataka, definirao bi i navedene testne podatke čije ispravnosti jamči. Za svaki uređeni par ulaz – izlaz, natjecateljev program osvojio bi određenu svotu bodova ili bi osvojio nula bodova. Natjecateljev konačan rezultat predstavlja sumu svih ovih bodova, za sve zadatke koje je riješio. Primjer zadatka može se vidjeti na slici 2.1, a aktualni zadatak prvi je zadatak s prvog kola prvog natjecanja HONI, koje se održalo na datum 28.10.2006.

1. MODULO

Za neka dva prirodna broja A i B izraz A modulo B je jednak ostatku pri dijeljenju broja A s brojem B. Tako recimo brojevi 7, 14, 27 i 38 postaju 1, 2, 0, 2 kad se gledaju modulo 3.

Od 10 unesenih brojeva, odredite koliko je različitih kada se gledaju modulo 42.

Ulazni podaci

10 prirodnih brojeva manjih od 1000, svaki u svojem redu.

Izlazni podaci

U prvoj i jedinoj liniji potrebno je ispisati broj različitih brojeva modulo 42.

Test primjeri

tipkovnica	**tipkovnica**	**tipkovnica**
1	42	39
2	84	40
3	252	41
4	420	42
5	840	43
6	126	44
7	42	82
8	84	83
9	420	84
10	126	85
zaslon	**zaslon**	**zaslon**
10	1	6

Pojašnjenje test primjera:
U prvom test primjeru brojevi modulo 42 iznose 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, što je 10 različitih brojeva.
U drugom test primjeru svi brojevi modulo 42 iznose 0, što je jedan broj.
U trećem test primjeru brojevi modulo 42 iznose 39, 40, 41, 0, 1, 2, 40, 41, 0, 1, što je 6 različitih brojeva.

Slika 2.1. Prvi zadatak prvog kola prvog natjecanja HONI održanog na datum 28.10.2006.[4]

Na slici 2.1. vidljiv je model test podatka – uređeni skup podataka prema modelu ulaz – izlaz. Pod rubrikom **tipkovnica** navedeni su, radi primjera, podaci koje bi ispitivajuće računalo (računalo koje testira natjecateljev program na točnost) upisalo u natjecateljev program, a pod rubrikom **zaslon** navedeni su podaci koje isto računalo očekuje kao izlaz navedenog programa. Ukoliko se podaci pod rubrikom **zaslon** i podaci kojeg natjecateljev program vrati podudaraju, natjecateljev program nagrađen je određenom svotom bodova. Natjecatelj nikada tijekom

natjecanja nema uvid u stvarne testne podatke na koje će se njegov program testirati, a takozvani test primjer ponuđeni su samo radi pomoći prilikom razvoja rješenja, kako bi se izbjegle nenamjerne greške u oblikovanju ispisa. Natjecateljev program tretira se kao crna kutija, a u obzir se uzimaju sljedeći faktori:

1. ulazni podaci koji budu upisani u program
2. izlazni podaci koje program ispiše
3. vrijeme izvođenja koje je proteklo tijekom navedena dva procesa

Sve programe testiralo bi isto računalo, sekvencijalno, program za programom, kako bi se očuvala transparentnost i pravednost ocjenjivanja. Vrijeme izvođenja mjerilo bi se zbog činjenice da se neki programski problemi mogu riješiti jednostavnim, ali izuzetno vremenski neefikasnim algoritmima, što članovi HSIN-a nisu smatrali poantom natjecanja HONI. Poantom je bilo smatrano riješiti problem razvojem kvalitetnog rješenja, a ne najjednostavnijeg mogućeg. Zbog toga, zadnjih nekoliko testnih podataka sadržavalo bi izuzetno velike podatke, podatke za koje bi program loše vremenske efikasnosti obavljao posao zauvijek i ne dostigao rješenje, što znači da bi kompletne bodove unutar jednog programskog problema natjecatelj zavrijedio tek kada bi napisao uistinu vremenski efikasan algoritam, a u protivnom, osvojio bi samo dio bodova.

Natjecanje HONI bilo bi objavljivano na internetskim stranicama HSIN-a u obliku rezultata zadnjih kola, preslika zadataka koji su tada bili zadani i stvarnim testnim podacima koji su bili korišteni. Prvi ovakav skup podataka bio je objavljen radi natjecanja održanog na datum 28.10.2006, a posljednji na datum 04.03.2017[1]. Natjecanje HONI bilo je popularno pretežito među srednjim školama, a predstavljalo je temelj razvoja logičkog razmišljanja i vještina pisanja kvalitetnih algoritama natjecatelja. Autor ovog diplomskog rada i sam je redovito sudjelovao u ovom natjecanju, no, s obzirom da nije posjedovao računalo koje bi omogućavalo lokalno prevođenje programskog koda i stoga omogućavalo autoru vježbanje pisanja algoritama, to je činio uglavnom u školi, što smatra samo neoptimalnim kompromisom. Ovaj diplomski rad sanira ovaj kompromis tako što cijeli proces prevođenja i izvršenja C++ programskog koda poslužuje na Internetu, na udaljenom računalu, prema modelu klijent-poslužitelj. Softver razvijen u sklopu ovog diplomskog rada budućim bi natjecateljima na nekom sličnom natjecanju omogućio lako razvijanje algoritama i sličnih programskih rješenja, a do neke razine čak i simulirao samo natjecanje, kroz model zadavanja i rješavanja zadataka u sklopu samog softvera.

Donedavno, slično rješenje već je postojalo. Radilo se o internetskoj platformi jednake namjene pod imenom Z-trening. Nažalost, Z-trening je od nepoznatog trenutka kontinuirano nedostupan i nije moguće priložiti vizualne materijale koji bi navedeni servis pobliže opisali. Platforma Z-trening je uvelike inspirirala ovaj diplomski rad. Kao i Z-trening i HONI, platforma razvijena u sklopu ovog diplomskog rada izgrađena je na temelju modela ulaz – izlaz. Korisnik platforme može kreirati vlastiti zadatak kojeg će tada moći rješavati svi korisnici platforme. Korisnik je dužan kreirati i testne podatke i jamčiti njihovu ispravnost. Svaki programski kod predan od strane bilo kojeg korisnika u jednom se trenutku prevodi i izvršava na poslužiteljskom računalu. Naravno, ova činjenica otvara izuzetne mogućnosti hakerskog djelovanja protiv interesa same platforme i radi uništenja poslužiteljskog računala. Stoga, u sklopu ovog diplomskog rada također su implementirani i mehanizmi zaštite poslužiteljskog računala kroz onemogućavanje nekih funkcionalnosti izvršavanja predanih programa koje bi u scenariju izvršavanja istih na lokalnom računalu bile u potpunosti omogućene. Također, u sklopu ovog diplomskog rada demonstrirana je kvalitetna realizacija sustava za registraciju i prijavu, prema modelu složene kriptografije. Ovakva realizacija sustava za registraciju i prijavu u jednu je ruku i osvrt na još uvijek velik udio internetskih servisa koji barataju izuzetno ranjivim sustavima registracije i prijave.[5]

2. TEHNOLOGIJE I ALATI NA KOJIMA JE PLATFORMA IZGRAĐENA

2.1. Poslužiteljsko računalo

Za kontinuirano posluživanje aplikacije koja je bila razvijena prema konceptu prikazanom u poglavlju 1 trebalo je biti odabrano poslužiteljsko računalo. Prvobitni kandidat za ovu ulogu bio je PHP poslužitelj pod pokroviteljstvom Fakulteta elektrotehnike, računarstva i informacijskih tehnologija (FERIT). FERIT omogućava svakom svom studentu određeni memorijski prostor unutar kojeg može posluživati svoje PHP aplikacije i koje su zatim raspoložive na Internetu, vremenski onoliko koliko je raspoloživ i sam internetski resurs, drava.etfos.hr. No, ovaj PHP poslužitelj ne omogućava kreiranje datoteka na svojem datotečnom sustavu od strane PHP poslužiteljskog softvera (httpd.exe). Za realizaciju ovog koncepta bilo je poželjno posjedovati poslužitelj koji nema ovo ograničenje. Sukladno tome, za ovu ulogu odabrano je autorovo osobno računalo s obzirom na stopostotnu kontrolu konfiguracije i proizvoljnu raspoloživost. U tablici 2.1 navedena je sklopovna specifikacija navedenog računala.

Tablica 2.1. Hardverska specifikacija poslužiteljskog računala u vlasništvu autora

CPU	AMD Phenom2 1055T @ 2.8GHz x 6
GPU	AMD Radeon HD7850
HDD	Samsung 1TB @ 7200 RPM
RAM	8GB DDR3 @ 1600MHz
Matična ploča	Asrock 770 Extreme

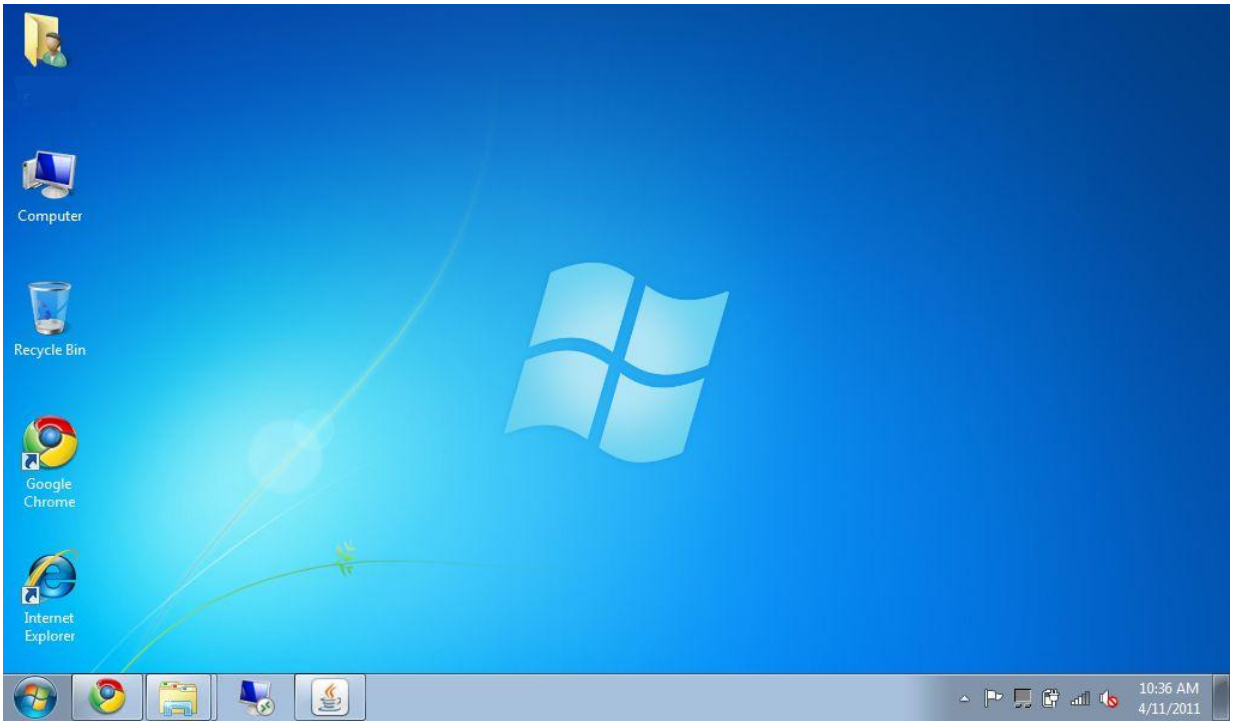
2.2. Operacijski sustav

Operacijski sustav jest softver čija je namjena sklopovlje učiniti funkcionalnim na način da korisnik računala, od trenutka instalacije operativnog sustava, na raspolaganju ima računalo pomoću kojeg može pregledavati i uređivati digitalni sadržaj, bilo to koristeći aplikacije ugrađene u operativni sustav ili aplikacije treće strane. Najpopularniji operativni sustavi današnjice su upravo oni koji pružaju zadovoljavajuću podršku za razvoj aplikacija treće strane.

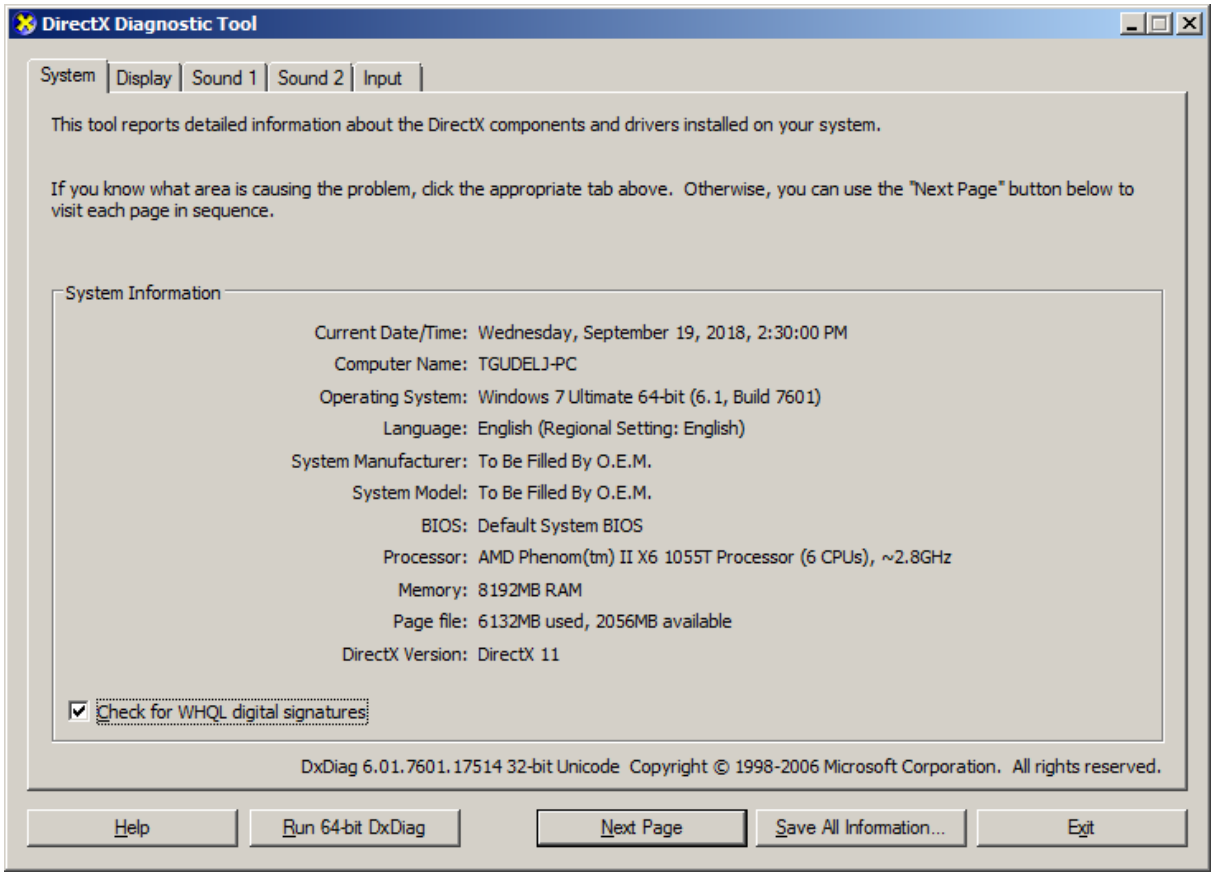
Operacijski sustav (OS) (još i radni sustav, operacijski sistem, operativni sustav) je skup osnovnih sustavnih programa koji upravljaju sklopovljem računala (eng. hardware) radi ostvarivanja osnovnih funkcija računala: ulaz, memoriranje, obrada i izlaz podataka.[6]

Operacijski sustav omogućuje vezu između sklopovlja i korisničkih programa. Mnogi korisnički programi u svom izvođenju pozivaju funkcije koje su sadržane u operacijskom sustavu kroz tzv. API (eng. application program interface).[6]

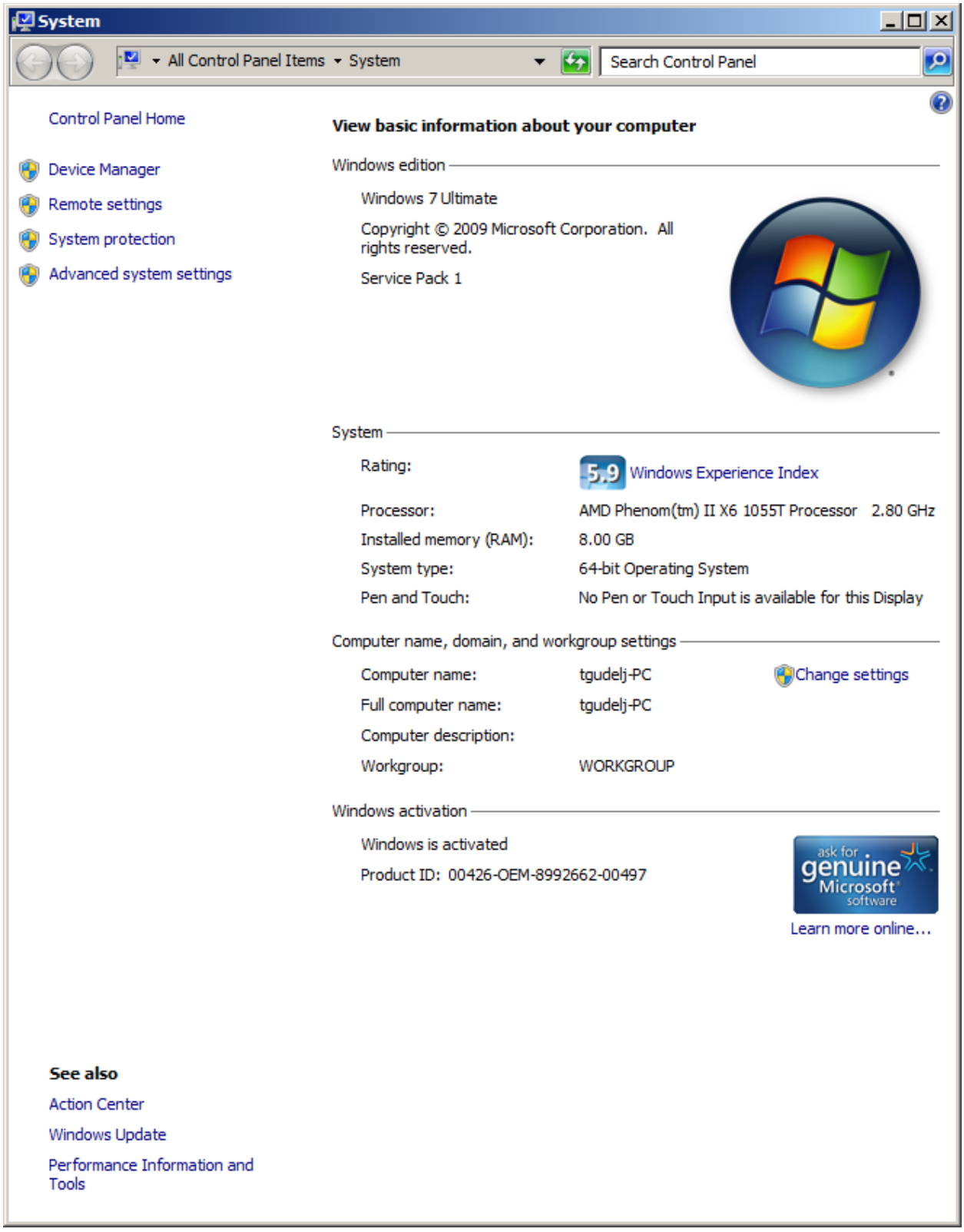
S obzirom da je za samo poslužiteljsko računalo odabrano autorovo osobno računalo, za operativni sustav bilo je neizbježno odabrati operativni sustav koji pokreće autorovo osobno računalo. U ovom slučaju to je Microsoft Windows 7 x64. Ovaj je projekt mogao biti izrealiziran i na drugim operativnim sustavima nalik UNIX-u, uz manje dorade, pa čak i na operativnom sustavu Mac OS X uz veće dorade. S obzirom da je Mac OS X komercijalno rješenje uz koje je asocirana određena značajna cijena, mogućnost realizacije ovog projekta na istom nije dublje uzeta u obzir uslijed posjedovanja već funkcionalnog operativnog sustava kako je ranije navedeno. Na slici 2.1. vidljiv je početni izgled radne površine (engl. *desktop*) operativnog sustava Microsoft Windows 7 x64, a na slici 2.2. vidljiv je izvještaj računalne konfiguracije iz perspektive korisničkog sučelja navedenog operativnog sustava.



Slika 2.1. Izgled početne radne površine operativnog sustava Microsoft Windows 7 x64.



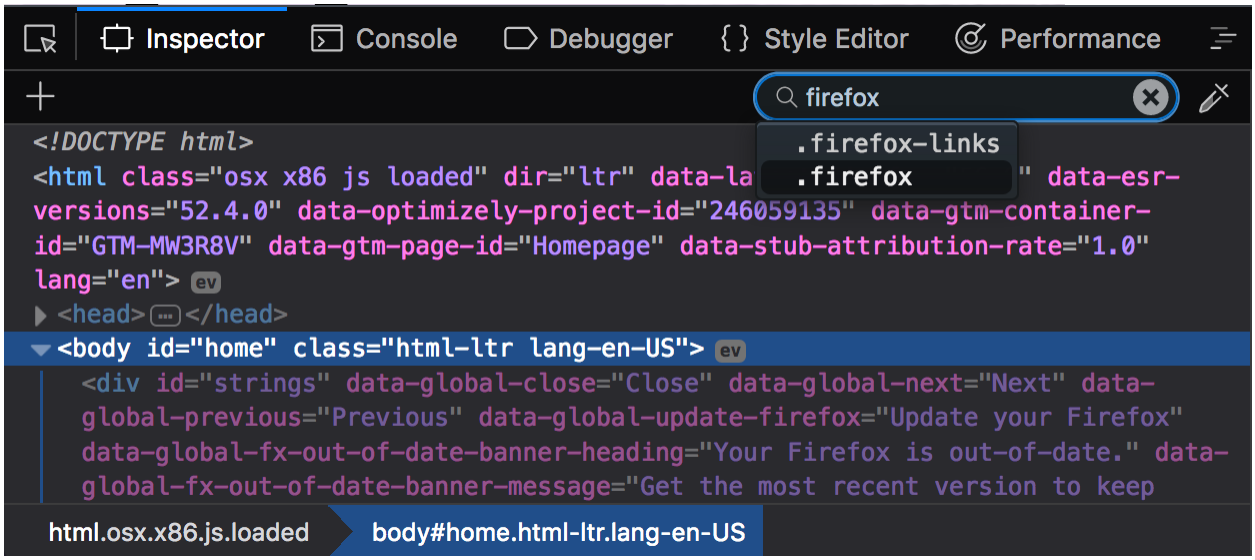
Slika 2.2. Izvještaj računalne konfiguracije iz perspektive korisničkog sučelja navedenog operativnog sustava otvaranjem aplikacije dxdiag.exe.



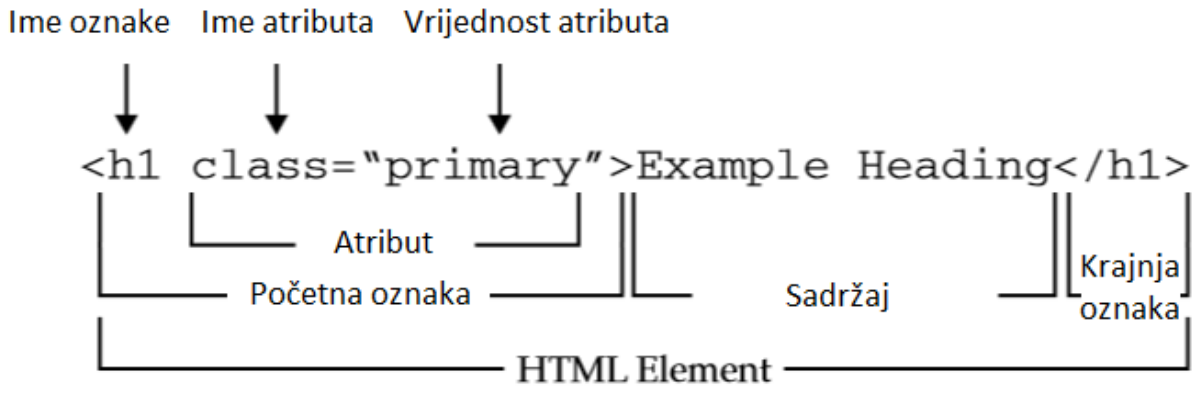
Slika 2.3. Izvještaj računalne konfiguracije iz perspektive korisničkog sučelja navedenog operativnog sustava korištenjem aplikacije explorer.exe.

2.3. Hypertext Markup Language (HTML)

Tehnologija HTML tehnologija je za definiranje razmještaja vizualnih i funkcionalnih elemenata *web* stranica. Ova izjava samo je istinita zato što je softver imenom *web* pretraživač, softver korišten za pretraživanje Interneta *web* stranicu po *web* stranicu, proizveden s namjenom interpretiranja HTML koda. HTML, uz vlastitu varijantu XHTML, predstavlja osnovu svih *web* stranica na Internetu. Za potrebe realizacije razmještaja vizualnih i funkcionalnih elemenata ovog projekta korišten je HTML. Elementi poput ulaznih rubrika obrazaca (engl. *form*), tablica, tipaka i većine vidljivog teksta realizirani su pomoću HTML-a.



Slika 2.4. Maleni primjer HTML koda. [7]



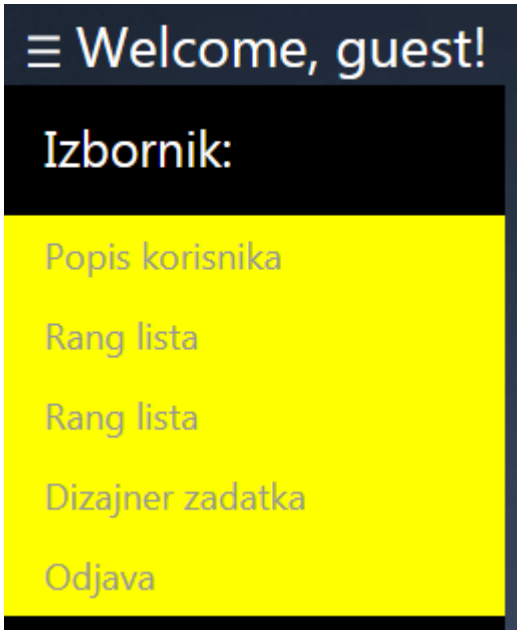
Slika 2.5. Rasčlanjenje jednog navoda jednog vizualnog elementa u HTML-u. [8]

2.4. Cascading Style Sheets (CSS)

Tehnologija CSS predstavlja tehnologiju kreiranu radi zadavanja pravila vizualnog izgleda i razmještaja elemenata definiranih pomoću nekog jezika označavanja (engl. *Markup language*). Danas, to nedvojbeno predstavlja HTML. Jedna CSS „skripta“ zapravo je kaskadna lista pravila stiliziranja elemenata. Jedan jedinični element CSS „skripte“ jest navod nekog HTML elementa ili skupine istih preko njihovih tzv. identifikatora, a zatim navod vizualnih pravila unutar vitičastih zagrada. Kao i sve internetske stranice na svijetu, za potrebe realizacije ovog projekta korišten je CSS kako bi se stilizirali korišteni HTML elementi projekta i kako isti ne bi ostali u potpunosti crno-bijele vanjštine i poravnati ulijevo, kako je uglavnom standard za novokreirane HTML elemente.

Slika 2.6. Rasčlanjenje jednog navoda jednog niza pravila stiliziranja u CSS-u. [9]

Rezultat primjene koda sa slike 2.6. vidljiv je na slici 2.7.. Ovaj primjer privremeno je iskorišten nad dijelom programskog koda iz ovog projekta.



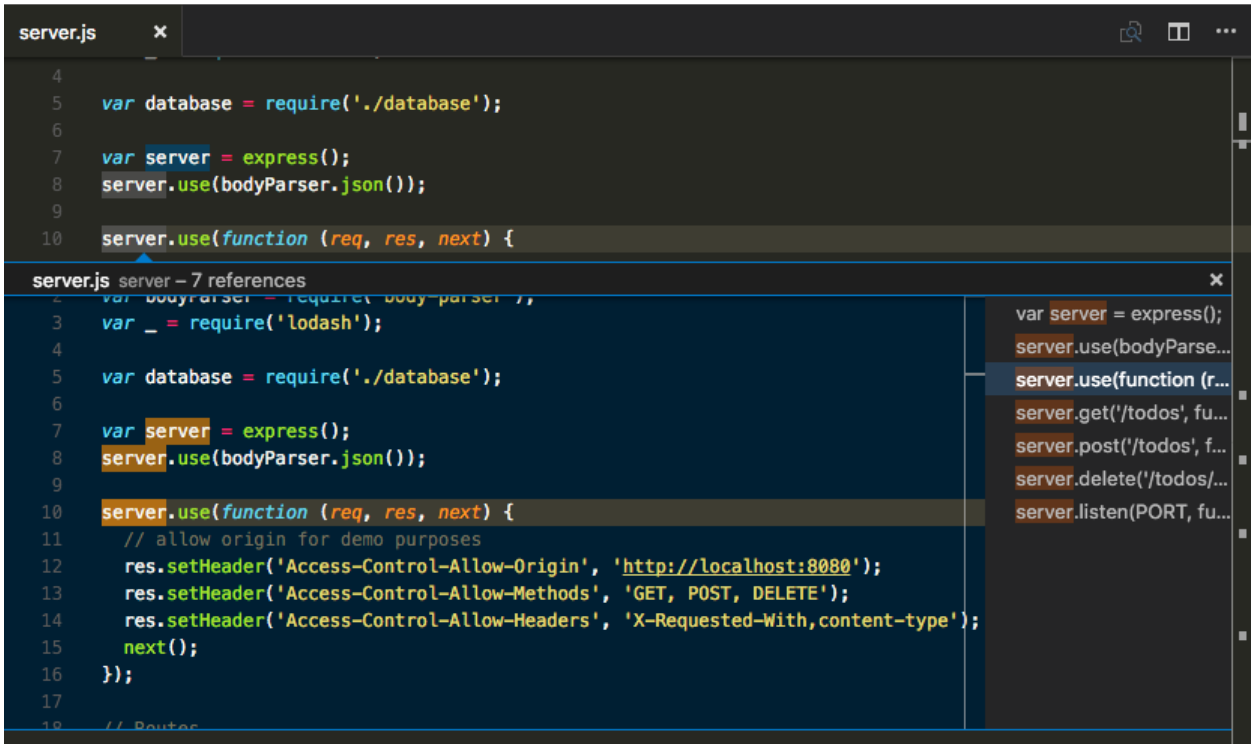
Slika 2.7. Rezultat primjene koda sa slike 2.5. na sve elemente tipa *a* unutar jedne od PHP datoteka projekta. Sve obojano žuto na ovoj slici neke su hiperveze.

2.5. JavaScript (JS)

JavaScript je programski jezik interpretiran „upravo na vrijeme“ (engl. *JIT – Just in time*). Implementiran u *web* pretraživače, JavaScript uglavnom se koristi kao programski jezik za upravljanje funkcionalnošću *web* stranica sa strane klijenta. To znači da se JavaScript koristi za upravljanje ponašanjem elemenata *web* stranica od trenutka kada preko TCP/IP protokola podaci o *web* stranici stignu na klijentsko računalo. Neki jednostavniji primjeri slučaja korištenja JavaScripta jesu upravljanje upozorenjima i prozorima poruke u trenutku kada klijent nešto klikne ili otvori na *web* stranici. Kao HTML, JavaScript sam po sebi ima ove funkcije jer ga *web* pretraživači uopće podržavaju.

JavaScript je skriptni programski jezik, koji se izvršava u web pregledniku na strani korisnika. Napravljen je da bude sličan Javi, zbog lakšega korištenja, ali nije objektno orijentiran kao Java, već se temelji na prototipu i tu prestaje svaka povezanost s programskim jezikom Java. Izvorno ga je razvila tvrtka Netscape (www.netscape.com). JavaScript je primjena ECMAScript standarda. [10]

JavaScript s AJAX (Asynchronous JavaScript and XML) tehnikom omogućuje web stranicama komunikaciju sa serverskim programom, što čini web aplikaciju interaktivnijom i lakšom za korištenje. [10]



Slika 2.8. Maleni primjer JavaScript koda. [11]

JavaScript je korišten u sklopu ovog projekta, ali minimalno. Korišten je radi premoštenja nekih inherentnih mana PHP-a i kao jednostavno i efektivno rješenje uslijed istih.

2.6. JQuery

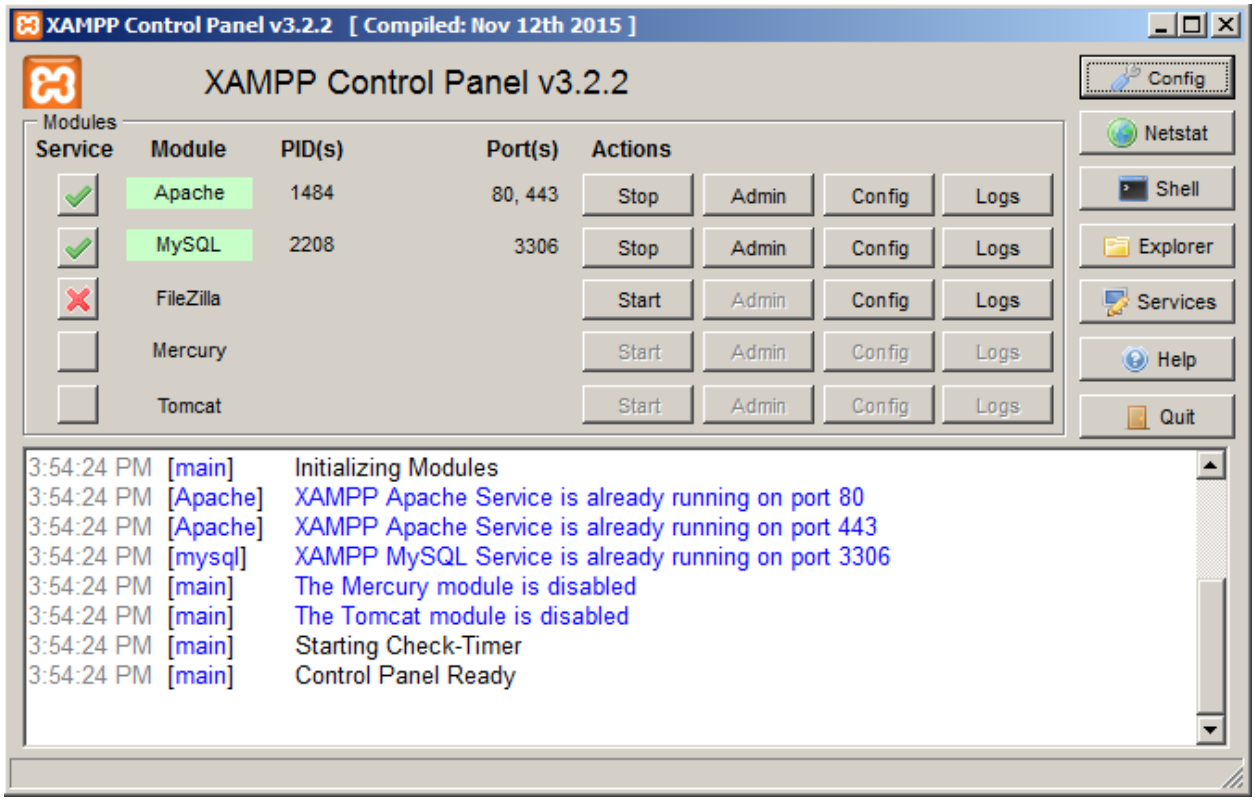
JQuery je brza i bogata JavaScript biblioteka. Osnovna ideja tehnologije jQuery bila je kreirati sloj programskog jezika koji radi na temelju JavaScripta, ali pojednostavljuje njegovo korištenje, a pritom pruža obogaćenje njegove funkcionalnosti. JQuery je bio korišten u sklopu izrade ovog projekta, ali minimalno. Korišten je radi premoštenja nekih inherentnih mana PHP-a i kao jednostavno i efektivno rješenje uslijed istih.

```
$.ajax({
  url: "/api/getWeather",
  data: {
    zipcode: 97201
  },
  success: function( result ) {
    $( "#weather-temp" ).html( "<strong>" + result + "</strong> degrees" );
  }
});
```

Slika 2.9. Maleni primjer jQuery koda. Ovaj programski kod koristi asinkroni JavaScript kako bi kontaktirao udaljeno računalo čak i nakon što je HTTP zahtjev (uslijed klika na neku *web* stranicu) u potpunosti dostavio traženi resurs. Ovaj primjer zatražuje podatke s nekog udaljenog računala prema podatku zapisanom u redu broj 2 i u slučaju uspješnog odziva, mijenja sadržaj nekog HTML elementa s proizvoljnim sadržajem. [12]

2.7. XAMPP

XAMPP je programski paket koji se sastoji od, među ostalim, Apache poslužiteljskog softvera, PHP pretprocesora (engl. *preprocessor*), sustava za vođenje baza podataka relacijskog modela MySQL i poslužiteljske inačice FileZilla FTP („*File transfer protocol*“ – protokol za prijenos datoteka preko mreže) softvera. XAMPP se sastoji i od drugih alata, ali su navedeni ovi alati koji su ključni za razvoj ovog projekta. XAMPP je multiplatformna (engl. *cross-platform*) aplikacija osmišljena za olakšanje pretvaranja nekog računala u računalo sposobno za posluživanje sadržaja na Internetu, pa čak i u odsustvu konkretno napisanih *web* stranica. Računalo samo po sebi (barem što se tiče operativnog sustava Microsoft Windows 7 x64) nije sposobno interpretirati PHP kod niti posluživati svoje resurse na Internetu. XAMPP predstavlja idealno rješenje s obzirom na odnos jednostavnosti i robusnosti. Na slici 2.10. vidljivo je korisničko sučelje aplikacije XAMPP Control Panel.



Slika 2.10. Korisničko sučelje aplikacije XAMPP Control Panel pokrenuto na poslužiteljskom računalu koje se koristi za potrebe ovog projekta.

2.7.1. Apache poslužitelj

Apache poslužitelj softver je za posluživanje resursa nekog računala na Internetu putem HTTP i TCP/IP protokola.

Apache je HTTP poslužitelj pod pokroviteljstvom Apache HTTP Server projekta. To je projekt otvorenog izvora (engl. *open-source*) i cilj mu je kontinuirano svijetu pružati siguran, efikasan i proširiv softver HTTP posluživanja za moderne operativne sustave poput UNIX-a i Windows-a. Prvobitno je stvoren 1995. godine, a od travnja 1996. godine najpopularniji je *web* poslužiteljski softver. [13]

U ovom projektu korišten je zbog dobrog omjera jednostavnosti implementacije i kvalitete pružene usluge.

2.7.2 Hypertext Preprocessor (PHP)

PHP je programski jezik pretprocesuiranja HTML resursa. U trenutku kada HTTP zahtjev stigne na poslužiteljsko računalo, nije nužno klijentu odmah vratiti nekakav unaprijed definirani HTML

kod. Sasvim je moguće na temelju okolnosti zahtjeva naknadno izračunati nekakav HTML kod dinamički, u danom trenutku, a zatim ga vratiti. Ovaj proces zove se pretprocesuiranje (engl. *preprocessing*). PHP je programski jezik točno osmišljen za ovu namjenu. PHP je korišten od najmanjih Internetskih projekata pa sve do najkompleksnijih svjetski prepoznatih *web* stranica. PHP nudi podršku za MySQL baze podataka, kao i za PDO (engl. *PHP Data Objects*) model podataka – međusloj modela podataka pomoću kojeg je moguće pristupiti velikom broju različitih pogona (engl. *engine*) baza podataka, među kojima je MySQL (MariaDB) samo jedan. PHP je prema sintaksi veoma nalik jezicima C i C++. PHP je slobodni softver otvorenog izvora.

2.7.3. MySQL

MySQL je inačica relacijskog modela baza podataka (engl. *Relational database management system – RDBMS*). Korišten je u sklopu ovog projekta zato što je efektivno i jednostavno omogućen od strane XAMPP paketa, a podržava ga PHP tehnologija.

2.7.4. FileZilla

FileZilla jest, u ovom slučaju, poslužiteljska inačica softvera koji implementira FTP protokol, protokol za razmjenu datoteka preko mrežnog sloja Interneta. U sklopu ovog projekta korišten je rano u razvoju kada je razvoj tekao s udaljenog računala. FileZilla FTP poslužitelj omogućavao je autoru razmjenu datoteka poslužitelja s udaljenog računala i razvoj s udaljenog računala. U konačnim trenucima razvoja, potreba za ovim je iščezla pa je na poslužiteljskom računalu ovaj servis u konačnici ugašen.

2.8. Datotečni sustav

Neki dijelovi funkcionalnosti ovog projekta i ove *web* aplikacije implementirani su tako da podatke spremaju u i čitaju iz baze podataka. No, ovo ne vrijedi za baš sve slučajeve spremanja i čitanja podataka, zbog prirode koncepta prevođenja programskog koda i izvršavanja rezultantnog binarnog koda na poslužiteljskom računalu. U barem nekom trenutku, ova *web* aplikacija morat će na datotečnom sustavu poslužiteljskog računala spremiti i izvršiti neke datoteke, tekstualne ili izvršne. Kao što je i navedeno u poglavlju 2.1., ukoliko ovo ne bi bilo moguće, ne bi bilo moguće na ovaj način izvršavati programski kod predan od strane klijenta i vratiti rezultat. Dakle, unutar ovog projekta programski kod ne ocjenjuje se opisno niti mu se daju kritike, nego

se ocjenjuje kroz stvarno prevođenje i izvršavanje programskog koda na poslužiteljskom računalu.

2.9. MinGW

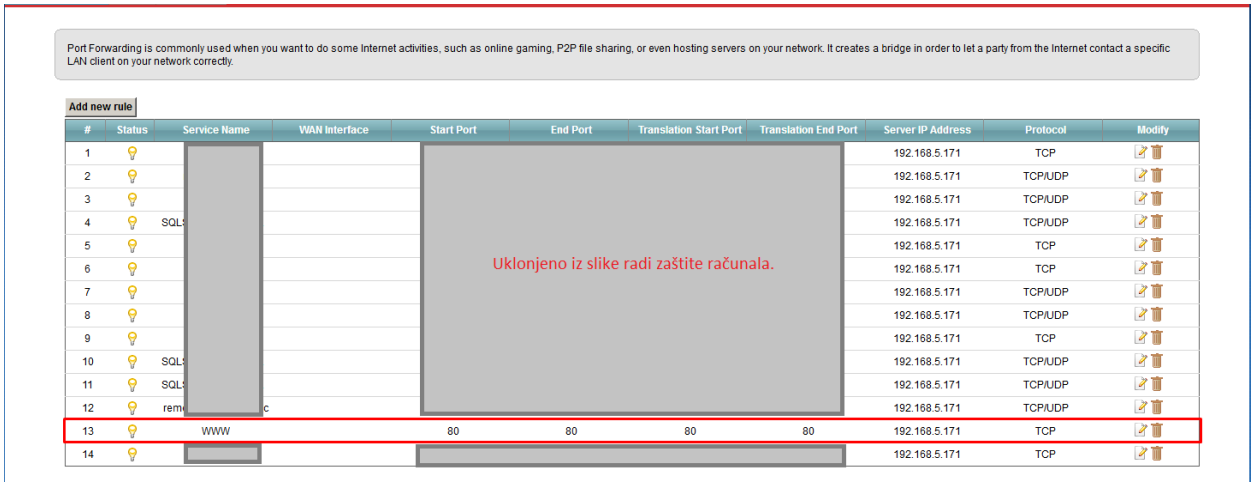
Minimalist GNU for Windows (MinGW) programski je paket koji na Windows operativnim sustavima omogućavaju korištenje prevoditeljskog softvera za C i C++ programske jezike, ali i neke druge. Ova podrška osmišljena je i realizirana po uzoru na istu unutar GNU sustava, podvrste UNIX operativnih sustava.

2.9.1 G++

G++ (g++.exe) jest dio MinGW paketa, a predstavlja prevoditelj za C++ programe. U sklopu ovog projekta potreban je i štoviše, ključan je dio projekta koji omogućava uopće razvoj ikakve daljnje ideje udaljenog izvršenja C++ programskog koda.

2.10. NAT

Prespajanje mrežnih adresa (engl. *Network Address Translation - NAT*) mrežni je koncept konfiguriranja lokalnih i ostalih mreža tako da se onemogući da pristigli HTTP upit ima višeznačno odredište. Kada računalo poslužitelj nudi svoje servise na Internetu i neki klijent se eventualno pokuša spojiti na unaprijed poznatu brojčanu adresu internetskog protokola (IP adresa) tog računala, s obzirom da cijela lokalna mreža u kojoj se to računalo nalazi zapravo dijeli istu tzv. vanjsku IP adresu s mrežno sposobnih uređaja u toj mreži, nije jednoznačno određeno na koji uređaj je potrebno proslijediti HTTP upit stvoren uslijed pokušaja spajanja na ovo računalo. Uređaj imenom usmjerivač (engl. *router*) mrežni je uređaj koji usmjerava upite izvan i unutar lokalne mreže, a na njega je spojeno poslužiteljsko računalo. Postoje i drukčiji modeli realizacije mreža, ali u slučaju ovog projekta, bilo je potrebno konfigurirati usmjerivač kako bi se ova višeznačnost adresa otklonila. Ovo se postiže uparivanjem tzv. *port-a* na kojoj se aplikacija nalazi (za HTTP protokol to je port 80) s tzv. unutarnjom IP adresom poslužiteljskog računala, IP adresom koju ono posjeduje unutar te lokalne mreže. Koristeći korisničko sučelje usmjerivača, to izgleda kao na slici 2.11..



Slika 2.11. Korisničko sučelje konfiguriranja NAT funkcionalnosti na usmjerivaču. Crveno uokvireno vidljiva je konfiguracija NAT s ciljem prespajanja HTTP zahtjeva koji stižu s Interneta – na poslužiteljsko računalo.

2.11. Dynu

Poslužiteljsko računalo igrom slučaja posjeduje takvu supskripciju usluge kod internetskog pružatelja usluge (engl. *Internet Service Provider*) koja ne omogućava uvijek istu IP adresu preko koje se navedenom računalu može pristupiti, tzv. statičnu IP adresu. S obzirom da se u bilo kojem trenutku IP adresa ovog računala može izmijeniti, eventualni korisnici nemaju stabilan način koristiti usluge ovog projekta. Radi toga, ugovorena je supskripcija kod pružatelja usluge imenom Dynu. Dynu jest internetski servis koji pruža uslugu dinamičnog DNS poslužitelja. DNS (engl. *Domain Name System*) internetski je sustav koji uz IP adrese asocira imena u tekstualnom obliku. Baš svaka internetska stranica koju šira populacija dostiže koristeći nekakav tekstualni format, poput *www.google.com*, za razliku od IP reprezentacije istog resursa, *172.217.17.142*, ima konfiguriranu nekakvu DNS funkcionalnost kako bi navedeno ime bilo ispravno asocirano uz navedenu adresu. U slučaju ovog projekta, ugovoreno je kako će mrežna adresa *tgudelj.dynu.net* kontinuirano pokazivati na koju god IP adresu poslužiteljsko računalo posjeduje u danom trenutku.

2.12. Infrastruktura veze na Internet

Internetska usluga na koju je spojeno poslužiteljsko računalo ovog projekta barata propusnošću od 7Mbps (Megabit po sekundi), gdje maksimalna propusnost preuzimanja predstavlja većinski udio ove vrijednost, a maksimalna propusnost slanja manjinski. S obzirom da se u ovom projektu

barata slanjem i primanjem malenih datoteka programskog koda, ovo ograničenje neće predstavljati osjetnu kompromitaciju performansi.

3. IMPLEMENTACIJA PROJEKTA

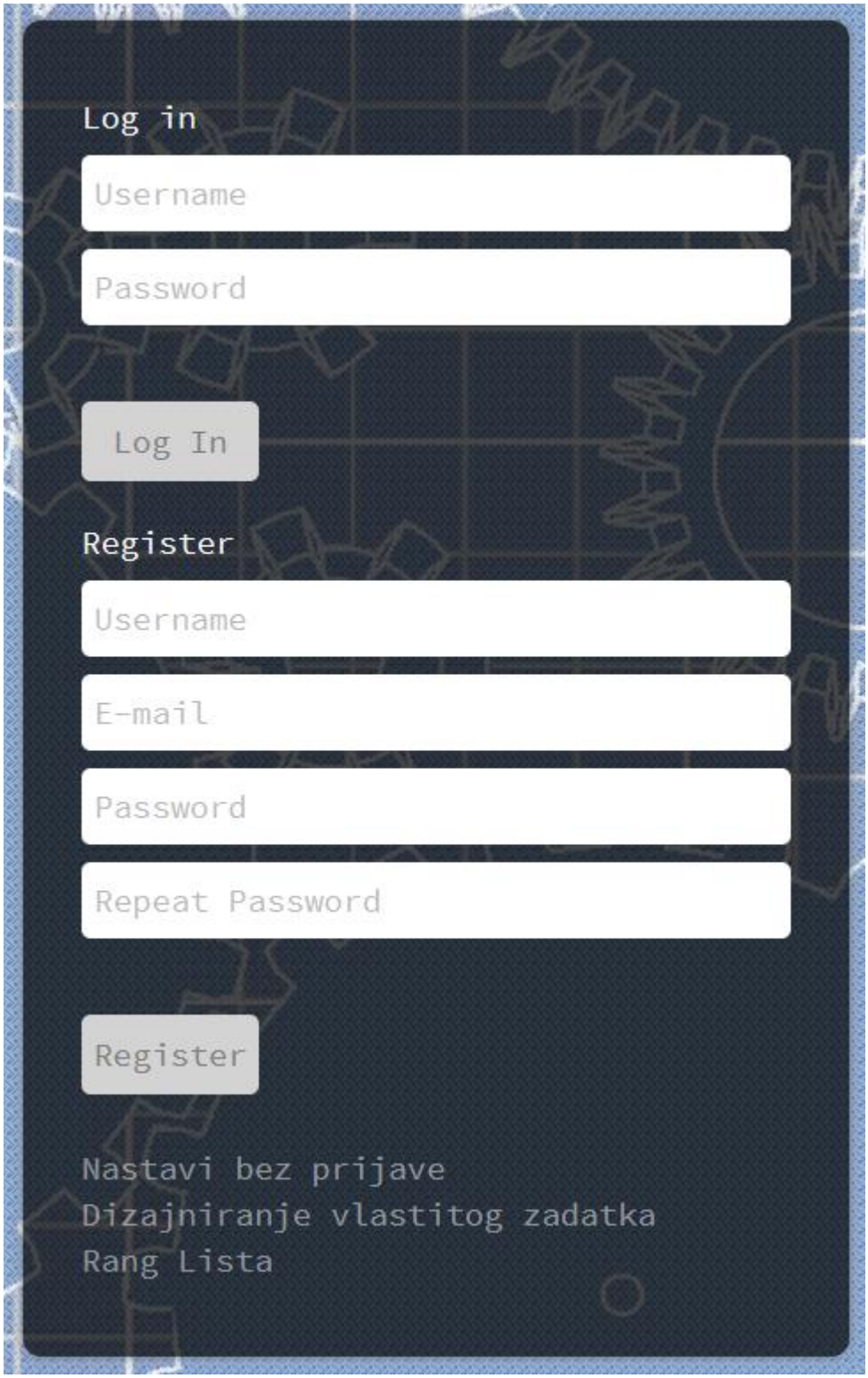
Kompletan programski kod svih vidljivih i funkcionalnih dijelova aplikacije koje je autor samostalno programirao, ali ne i programski kod programskih paketa treće strane, bit će raspoloživ u prilozima ovog diplomskog rada. No, neki isječci programskog koda bit će raspoloživi u ovom poglavlju također, radi potreba demonstracije i objašnjenja raznih funkcionalnosti ove aplikacije. Demonstracija implementacije projekta podijeljena je u funkcionalne cjeline i to počevši iz perspektive korisnika u sljedećim potpoglavljima pomoću kojih je u potpunosti objašnjena.

3.1. Sučelje za prijavu i registraciju

Kada se korisnik spoji na navedeni internetski resurs,

http://tgudelj.dynu.net/

odmah je predstavljen sučeljem za izbor iduće akcije. Sučelje je vidljivo na slici 3.1.



Slika 3.1. Sučelje za prijavu ili odabir neke druge akcije.

Slika 3.2. Slika 3.1., ali iz *full-screen* perspektive, neodrezano.

Na sučelju prvi obrazac predstavlja obrazac za prijavu. Kako bi se korisnik prijavio, mora posjedovati korisnički račun, kojeg ukoliko ne posjeduje, može odmah registrirati koristeći drugi obrazac.

Drugi obrazac je obrazac za registraciju. Korisnik može registrirati svoj novi korisnički račun. Ukoliko je korisničko ime koje korisnik unese već zauzeto, registracija se neće uspješno izvršiti, a na korisniku je da pokuša ponovo.

Nakon spomenuta dva obrasca, korisniku je ponuđeno preostale tri akcije koje bi ga odvele nekamo drugdje u sklopu aplikacije.

Ukoliko klikne na „Nastavi bez prijave“ hipervezu, bit će odveden dalje u aplikaciju na sučelje opisano u poglavlju 3.2., odnosno baš kao da se uspješno prijavio, ali će biti prijavljen na testni račun, takozvani gost račun (engl. *guest*).

Ukoliko klikne na „Dizajniranje vlastitog zadatka“ hipervezu, bit će odveden na sučelje opisano u poglavlju 3.3., odnosno sučelje za dodavanje vlastitog zadatka u sustav.

Ukoliko klikne na „Rang Lista“ hipervezu, bit će odveden na sučelje opisano u poglavlju 3.4., odnosno sučelje za prikaz rang liste svih korisnika u sustavu.

Ovo sučelje u idealnom slučaju prvo je sučelje kojim će korisnik biti prezentiran. Ukoliko je korisnik prvobitno prezentiran nekim drugim sučeljem, to može predstaviti nedozvoljene okolnosti rada aplikacije.

U prvom koraku razvoja ovog sučelja definirane s pomoćne varijable za pristup bazi podataka.

```php
<?php
    $DB_URI = "localhost";
    $DB_USERNAME = "root";
    $DB_PASSWORD = '';
    $DB_NAME = "cpp_programming_competition";
?>
```

Slika 3.2.1. Definiranje pomoćnih varijabli za pristup bazi podataka.

U neposredno idućem bloku koda definirana je veza na vanjski CSS dizajn napisan posebno radi potreba ovog projekta.

```html
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="index.css">
        <title>Natjecanje u programiranju - home</title>
    </head>
```

Slika 3.2.2. Povezivanje ovog sučelja i vanjskog CSS dizajna.

Obrazac za prijavu i registraciju realiziran je korištenjem *form* elemenata HTML tehnologije. Na slici 3.2.3. vidljiv je jedan od ovih obrazaca.

```html
<form id="register_form" action="index.php" method="POST">
        <input type="text" name="register_username" placeholder="Username" />
        <input type="text" name="register_email" placeholder="E-mail" />
        <input type="password" name="register_password" placeholder="Password" />
        <input type="password" name="register_password_repeat" placeholder="Repeat Password" />
        <br />
        <input type="submit" value="Register"/>
</form>
```

Slika 3.2.3. Stvarni primjer jednog od obrazaca iz ovog sučelja.

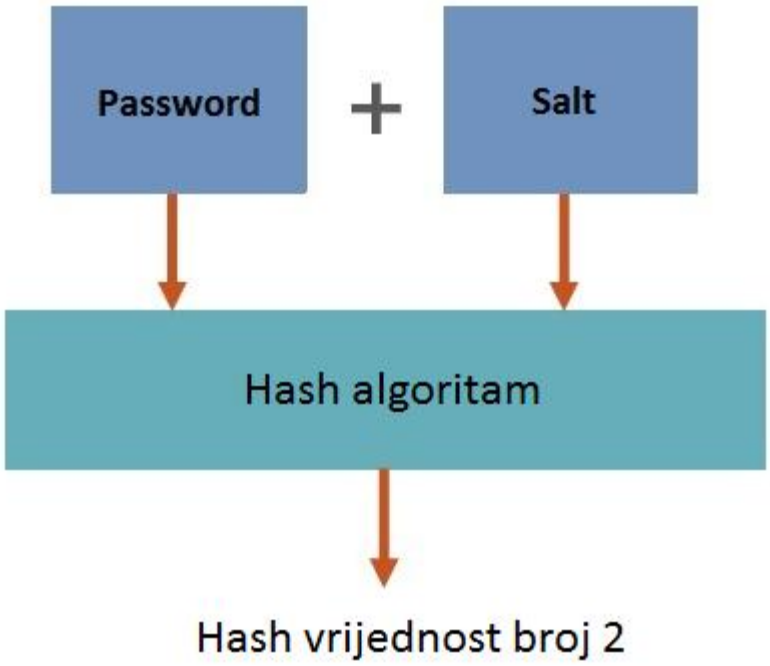
Nakon zatvarajućeg *html* elementa ovog sučelja naveden je jedan veći blok PHP koda. Ovaj blok koda, na slici 3.2.4., predstavlja jedno jedinično spajanje na bazu podataka, dohvaćanje podatka iz stupca *salt* za onog korisnika koji je predstavljen korisničkim imenom koje je upisano u obrazac za prijavu. Naravno, ovaj dio koda bit će aktivan samo ako je korisnik zapravo zatražio prijavu.

```php
<?php
    if(isset($_POST["username"]) && isset($_POST["password"])){
        $mysqli = new mysqli($DB_URI, $DB_USERNAME, $DB_PASSWORD, $DB_NAME);
        if (mysqli_connect_errno()){
            echo "Failed to connect to MySQL: " . mysqli_connect_error();
        }
        $salt = "SELECT salt FROM users WHERE username = '" . $_POST["username"] . "';";
        echo "<p>$salt</p>";
        if (!($salt = $mysqli->query($salt))){
            echo "salt query failed: (" . $mysqli->errno . ") " . $mysqli->error;
        }
        if($salt){
            $procedure_call = "SELECT * FROM users where username = '" . $_POST["username"] . "' AND password = SUBSTR(SHA2(CONCAT('" . $_POST["password
                "] . "','" . $salt->fetch_array()["salt"] . "'), 256), 1, 32)";
            //$procedure_call = 'SELECT SHA2(CONCAT(' . $_POST["password"] . ',' . $salt->fetch_array()["salt"] . '), 256)';
            echo "<p>$procedure_call</p>";
            $result;
            if (!($result = $mysqli->query($procedure_call))){
                echo "login query failed: (" . $mysqli->errno . ") " . $mysqli->error;
            }
            echo "<p> num_rows " . $result->num_rows . "</p>";
            if($result->num_rows == 1){
                //login successful
                session_start();
                $_SESSION["username"] = $_POST["username"];
                header("location: cpp_programming_competition.php");
            }
        }
    }
```

Slika 3.2.4. Veći blok PHP koda koji upravlja prijavom na sustav.

Salt predstavlja pomoćni podatak sastavljen od nasumičnih podataka. Koristi se u tzv. *hash* + *salt* modelu kriptografije. Navedeni model kriptografije podvrgava lozinku korisnika (prilikom registracije) tzv. *hash* proceduri – funkciji koja mijenja podatak na način da je nemoguće ili iznimno teško rezultirajući podatak pretvoriti natrag u originalni podatak. Stoga, ova procedura predstavlja dobar osnovni model za zaštitu lozinki u bazi podataka ukoliko sadržaj baze podataka ikad, uslijed bilo kakvog scenarija, bude pročitan od strane treće osobe.

Unatoč ovom modelu zaštite lozinki, postoje udruge i grupe koje ulažu trud u posjedovanje uređenih parova lozinki i njihovih *hash* vrijednosti. Ukoliko posjeduju *hash* vrijednost za neku lozinku, a zatim identičnu *hash* vrijednost pronađu u nekoj bazi podataka u čije posjedovanje dođu, ove grupe tada mogu unazadno usporediti poznate i upravo otkrivene *hash* vrijednosti i dešifrirati ih ne ulazeći u samu tehničku prirodu korištenog *hash* algoritma. Kako bi se ovo spriječilo, svakom pojedinom korisniku na lozinku dodaje se jedna nasumična vrijednost koja će za njega tada biti zapamćena, a tada se kombinacija te dvije vrijednosti podvrgava *hash* algoritmu. Ovo čini ranije opisanu ranjivost samostalnog provođenja *hash* algoritama daleko manje izraženom.



Slika 3.2.5. *Hash + salt* kriptografijski model.

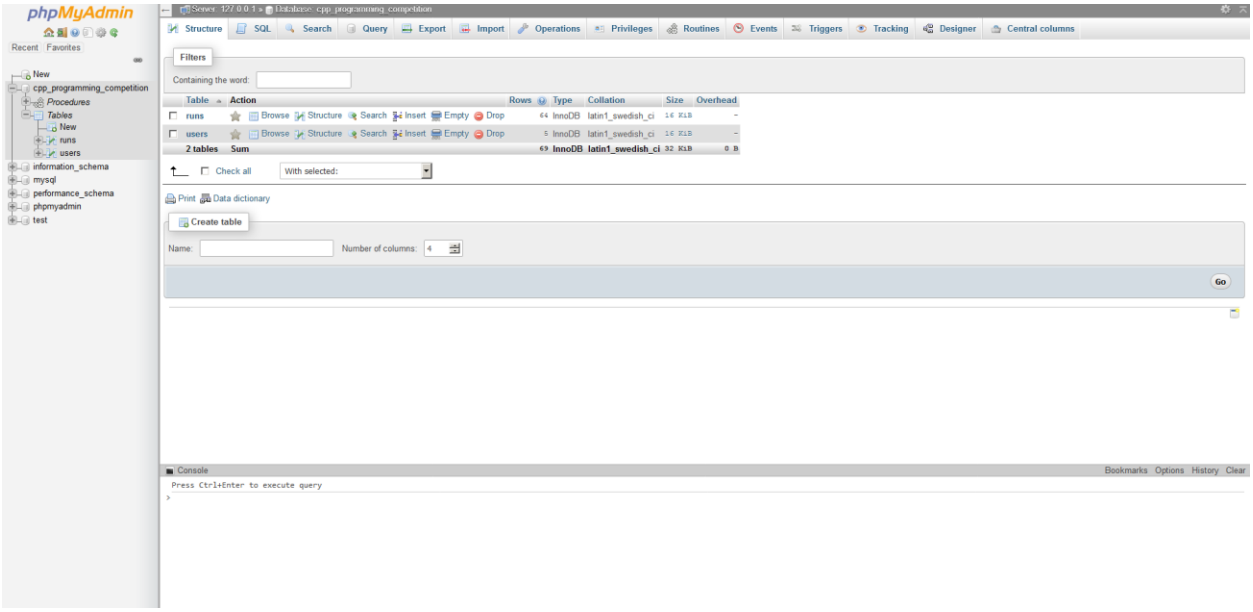
Na slici 3.2.4 (pretposljednjoj slici iz ove perspektive) vidljivo je kako red 75 postavlja nekakav SQL upit. Ovaj SQL upit nastoji iz baze podataka (konkretno, relacije users) otkriti sve zapise čiji *username* odgovara onom upisanom, ali istovremeno i upravo izračunat, dinamički predviđen *hash + salt* model lozinke upisane kroz obrazac odgovara onom u relaciji zapisanom *hash + salt* modelu lozinke spremljene tijekom registracije korisnika. Ukoliko takav slučaj postoji, to znači da je korisnik uspješno upisao ispravne podatke. Ukoliko ne, onda ili ne postoji takav *username*u sustavu, ili postoji, ali mu je predan neispravan *password*. Ukoliko je prijava uspješna, u redu 86 program će zapamtiti taj *username* kao mjerodavni podatak sjednice, a u redu 87 kontrola programa prijeći će na iduću PHP skriptu.

Registracija je veoma slična prijavljivanju, samo što umjesto ključne SQL riječi *SELECT* koristi riječ *INSERT*. Kompletan se kod može pogledati u prilogu ovog rada.

3.1.1. Dizajn baze podataka

U ovom trenutku potrebno je razumjeti modele podataka i relacija korištenih u bazi podataka.

Modeli podataka i same relacije kreirane su korištenjem phpMyAdmin grafičkog sučelja.



Slika 3.3. Izgled grafičkog sučelja phpMyAdmin, sastavnog dijela XAMPP paketa.

Kada se koristeći ovo sučelje pokrene SQL naredba *describe users;*, dobije se sljedeći pogled:

Your SQL query has been executed successfully.

```
describe users
```

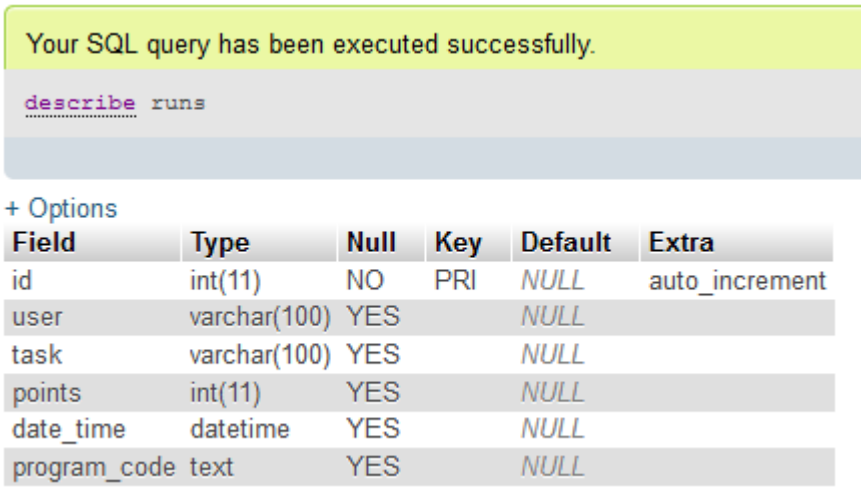
+ Options

Field	Type	Null	Key	Default	Extra
id	int(10) unsigned	NO		*NULL*	
username	varchar(30)	NO		*NULL*	
email	varchar(100)	NO		*NULL*	
password	binary(32)	YES		*NULL*	
salt	varchar(25)	YES		*NULL*	

Slika 3.3. Dizajn relacije imenom users – relacije koja drži podatke o pojedinim korisnicima sustava.

Kao što je vidljivo, *username, password, email* i *salt* podaci su koji se spremaju u ovu relaciju u kontekstu jednog korisnika.

Kada se koristeći ovo sučelje pokrene SQL naredba *describe runs;*, dobije se sljedeći pogled:


Your SQL query has been executed successfully.

describe runs

+ Options

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
user	varchar(100)	YES		NULL	
task	varchar(100)	YES		NULL	
points	int(11)	YES		NULL	
date_time	datetime	YES		NULL	
program_code	text	YES		NULL	

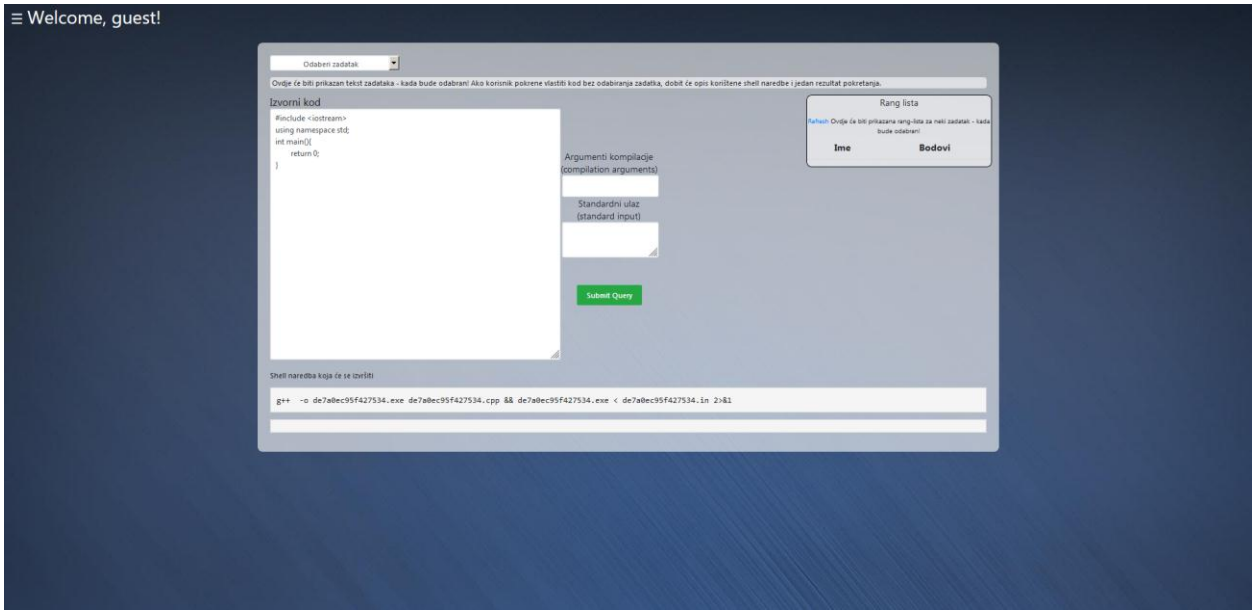
Slika 3.4. Dizajn relacije imenom runs – relacije koja drži podatke o pojedinim instancama pokretanja programskog koda u kontekstu nekog zadatka. O ovom će više riječi biti kasnije.

3.2. Sučelje za odabiranje zadatka i predaju programskog koda.

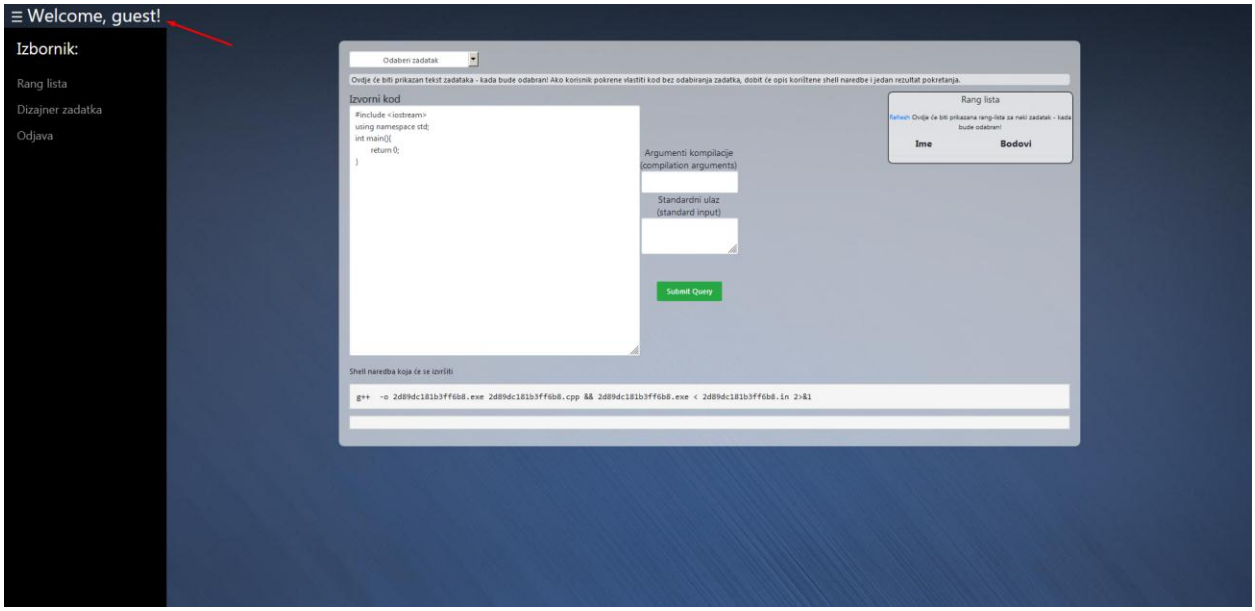
Sučelje na koje je korisnik odveden ukoliko je uspješno prijavljen (ili je odabrao opciju nastavljanja dalje kroz sustav bez prijavljivanja) jest sučelje za odabiranje zadatka i predaju programskog koda. Ovo je sučelje najkompleksnije po izradi, makar ne imalo takav izgled na prvi pogled.

Korisnik jest u mogućnosti predati ispravan ili neispravan C++ programski kod odmah, upisivanjem istog u najveće polje za unos vidljivih obrazaca. Korisnik nije obvezan odabrati zadatak, ali će tada ovaj sustav imati samo osnovnu funkcionalnost – funkcionalnost prevođenja predanog koda, izvršavanja programskog koda i vraćanja rezultata obrade. Pri dnu središnjeg dijela prozora vidljiva je takozvana *shell* naredba. To je zapravo ona naredba koja će biti izvršena na poslužiteljskom računalu. Ova naredba unaprijed je konfigurirana tako da koristi aplikaciju *g++.exe*. Ova aplikacija je dio paketa MinGW i predstavlja prevoditelj za C++ programski jezik. *Shell* naredba kakva je navedena na slici 3.5.1. prvo prevodi predani C++ (nastavka „.cpp“) program, a zatim novostvorene binarne datoteke pokušava izvršiti i njihov rezultat vratiti PHP poslužitelju, kako bi ga on mogao poslužiti u obliku HTML-a korisniku na zahtjev.

S gornje lijeve strane ovog pogleda nalazi se izbornik za navigaciju po ostalim funkcionalnostima ovog sustava.

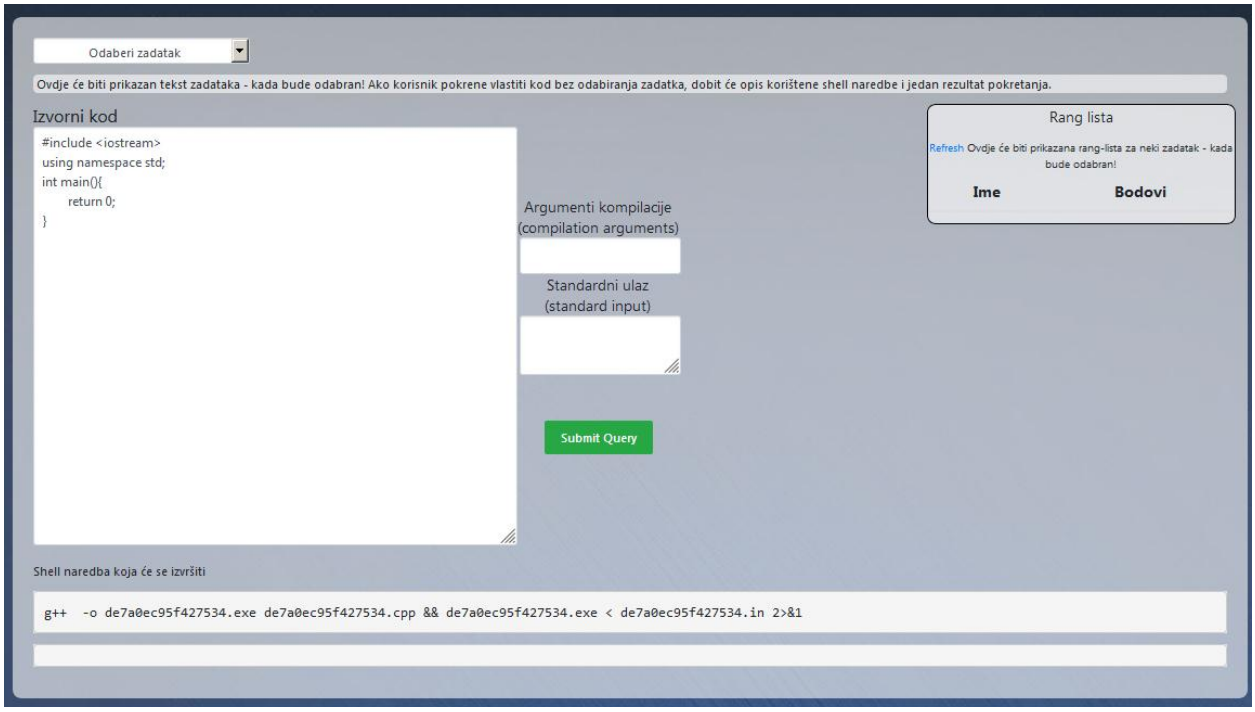


Slika 3.5.1. Vanjština sučelja za odabiranje zadataka i predaju programskog koda.

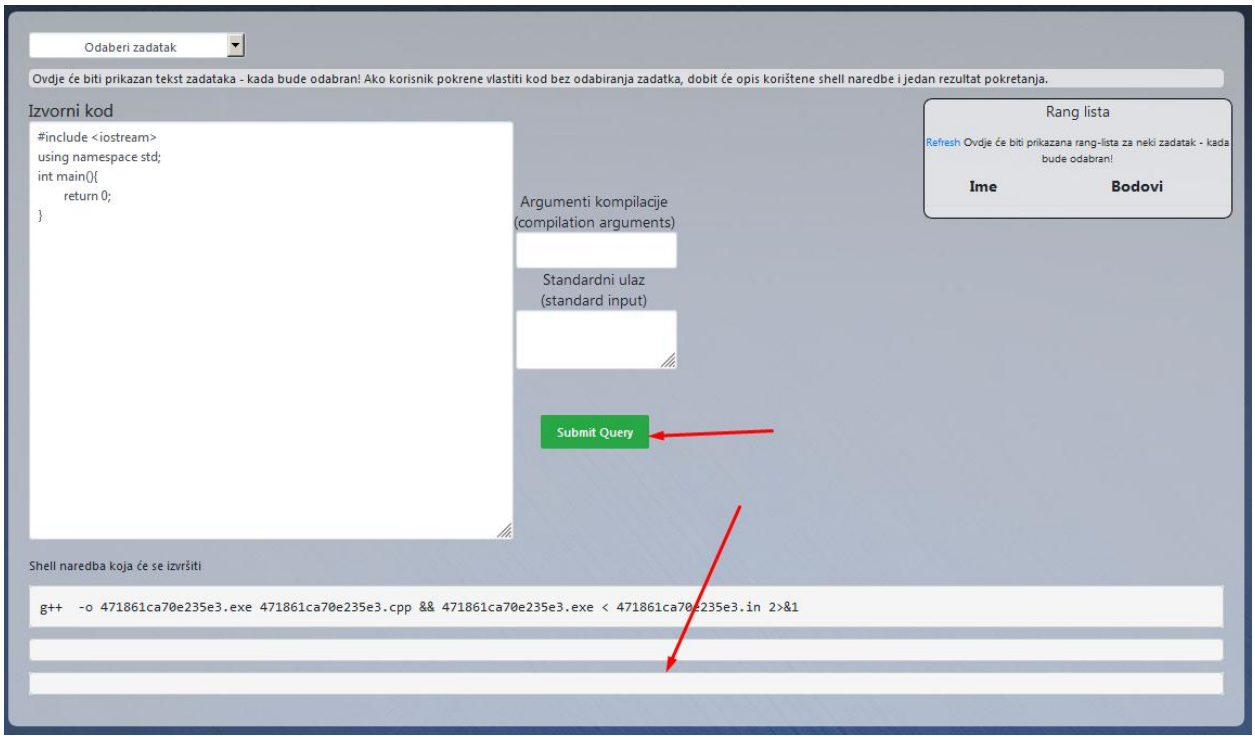


Slika 3.5.2. Izbornik za navigaciju po ostalim funkcionalnostima ovog sustava.

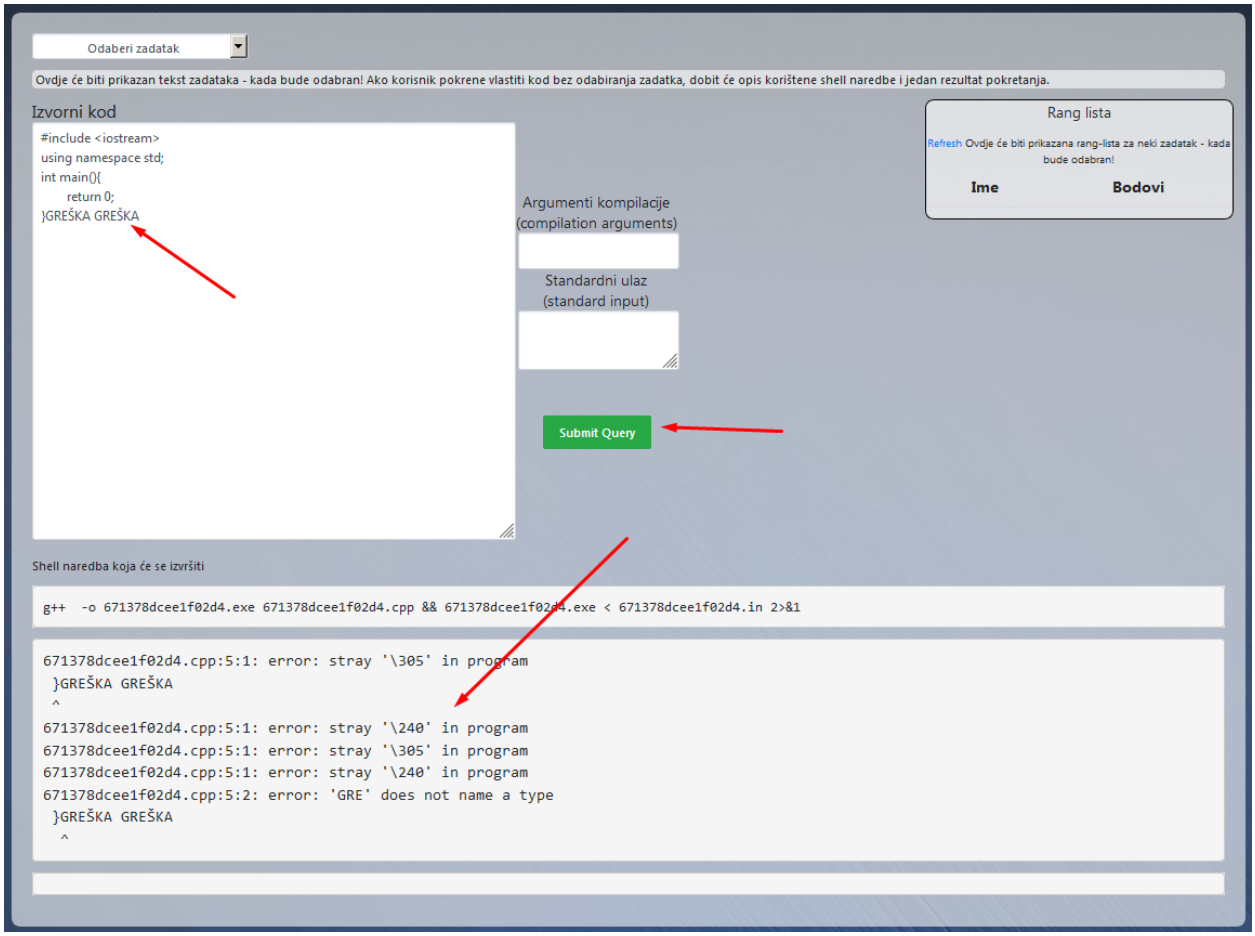
Ukoliko korisnik odabere zadatak koristeći padajući izbornik na gornjem dijelu središnjeg prozora, pali se složena funkcionalnost ovog sustava.



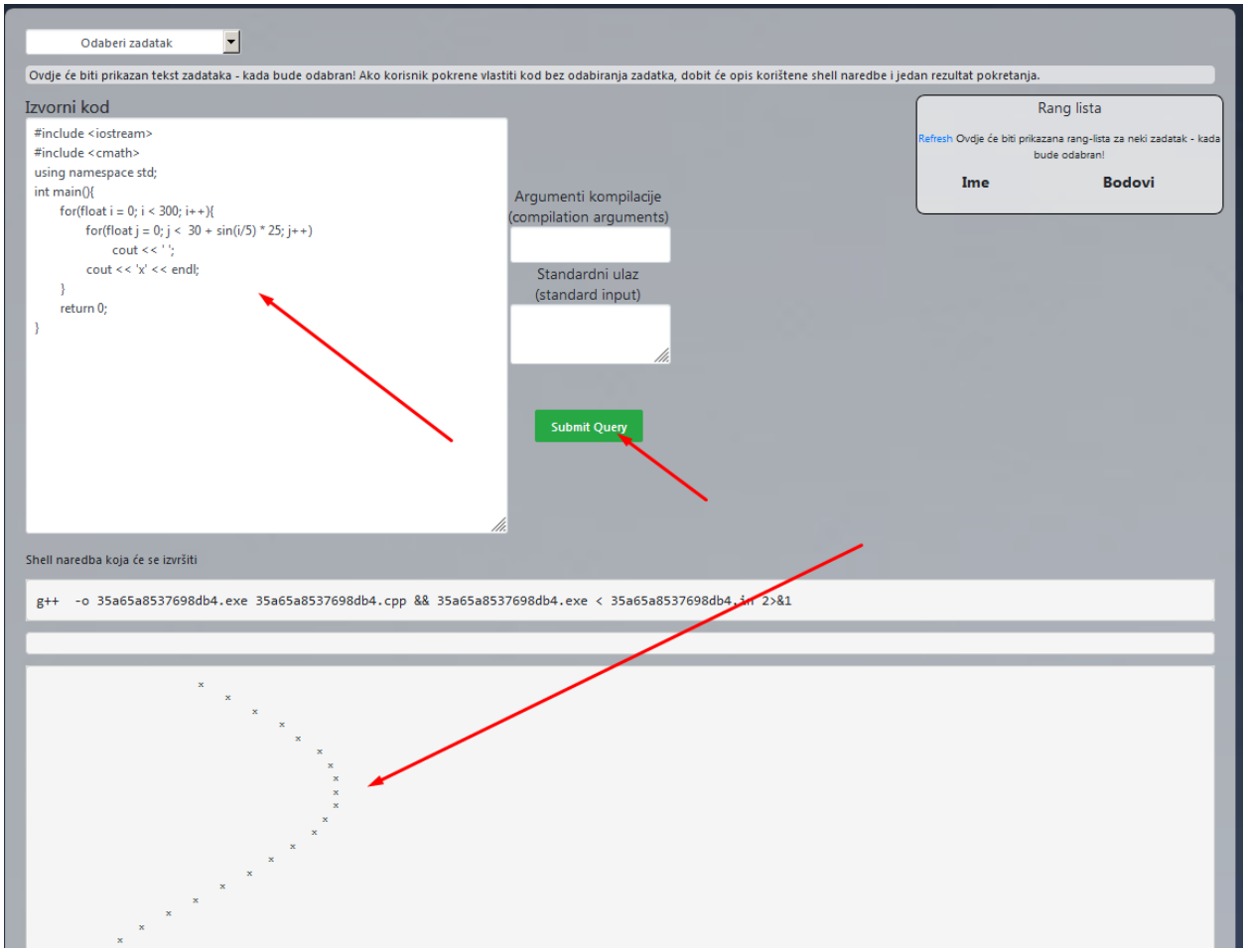
Slika 3.5.3. Sadržaj slike 3.5.2. adekvatno uvećan radi praktičnosti pogleda.



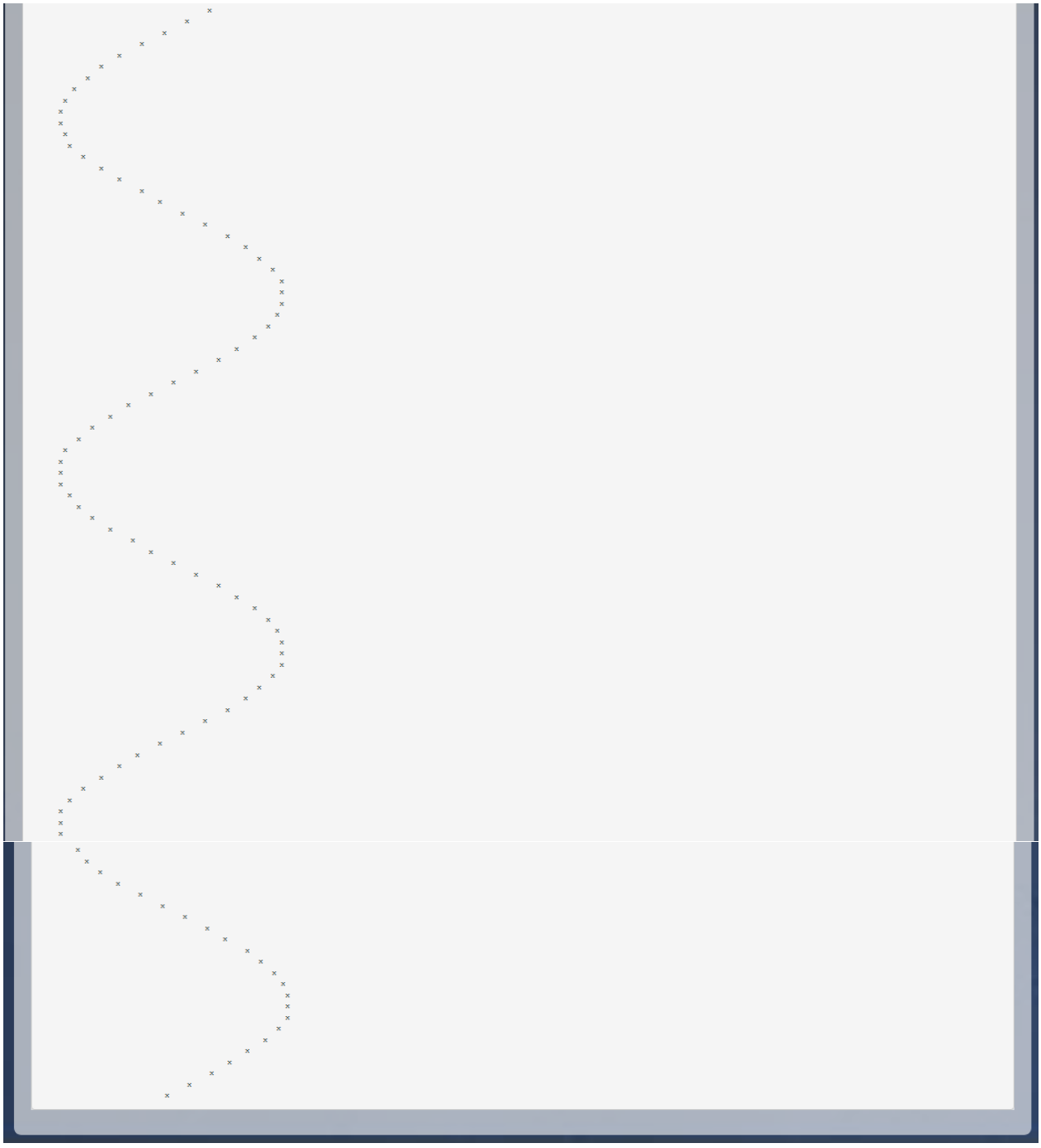
Slika 3.5.4. Kada se odmah prilikom učitavanja ovog pogleda klikne tipka za predaju obrasca, sustavu će se poslati početni, standardni programski kod kako je vidljivo u obrascu „Izvorni kod“. Taj je programski kod ispravan, za njega sustav ne vraća nikakvu poruku greške ili upozorenja i za njega sustav vraća rezultat rada. Rezultat rada igrom slučaja je prazan.



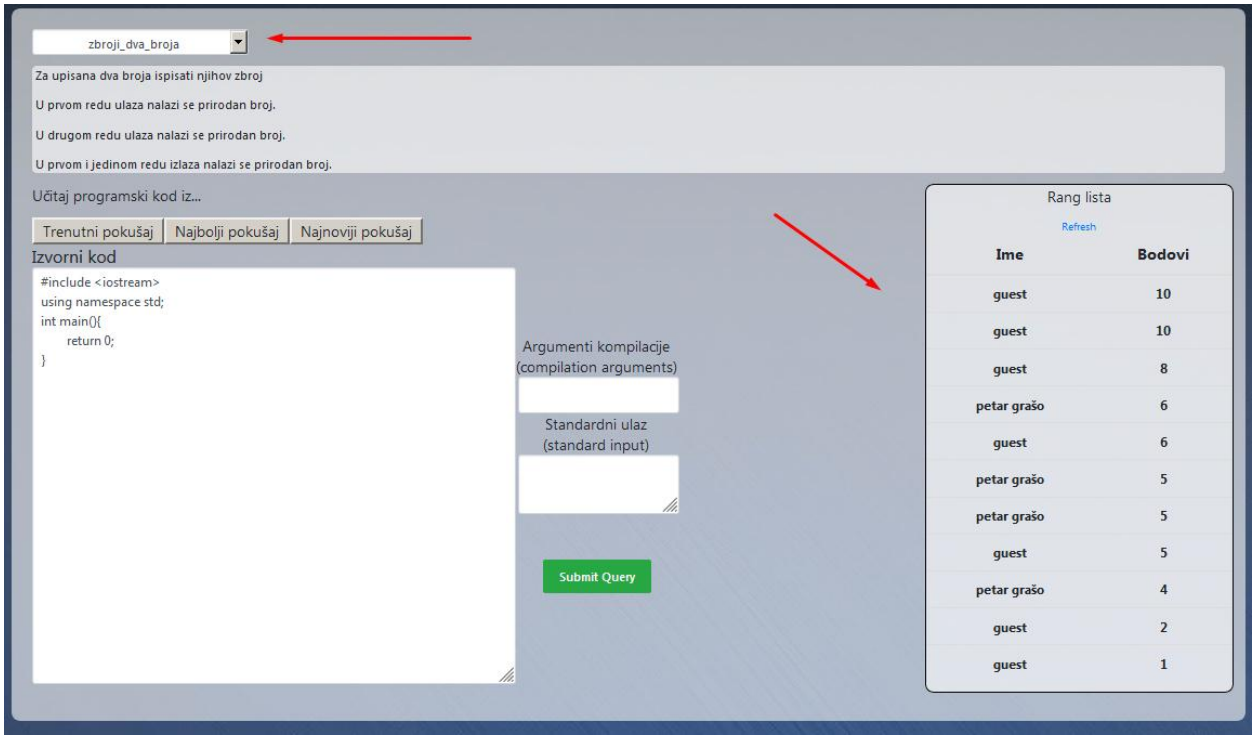
Slika 3.5.5. Ukoliko se preda neispravan C++ kod, sustav će to prepoznati i kao tok podataka odvojen od standardnog izlaza programa, vratiti podatke o neispravnosti programa, tipične za prijavljivanje grešaka u programskom kodu za jezik C++.



Slika 3.5.6. ostatak je na idućoj stranici.

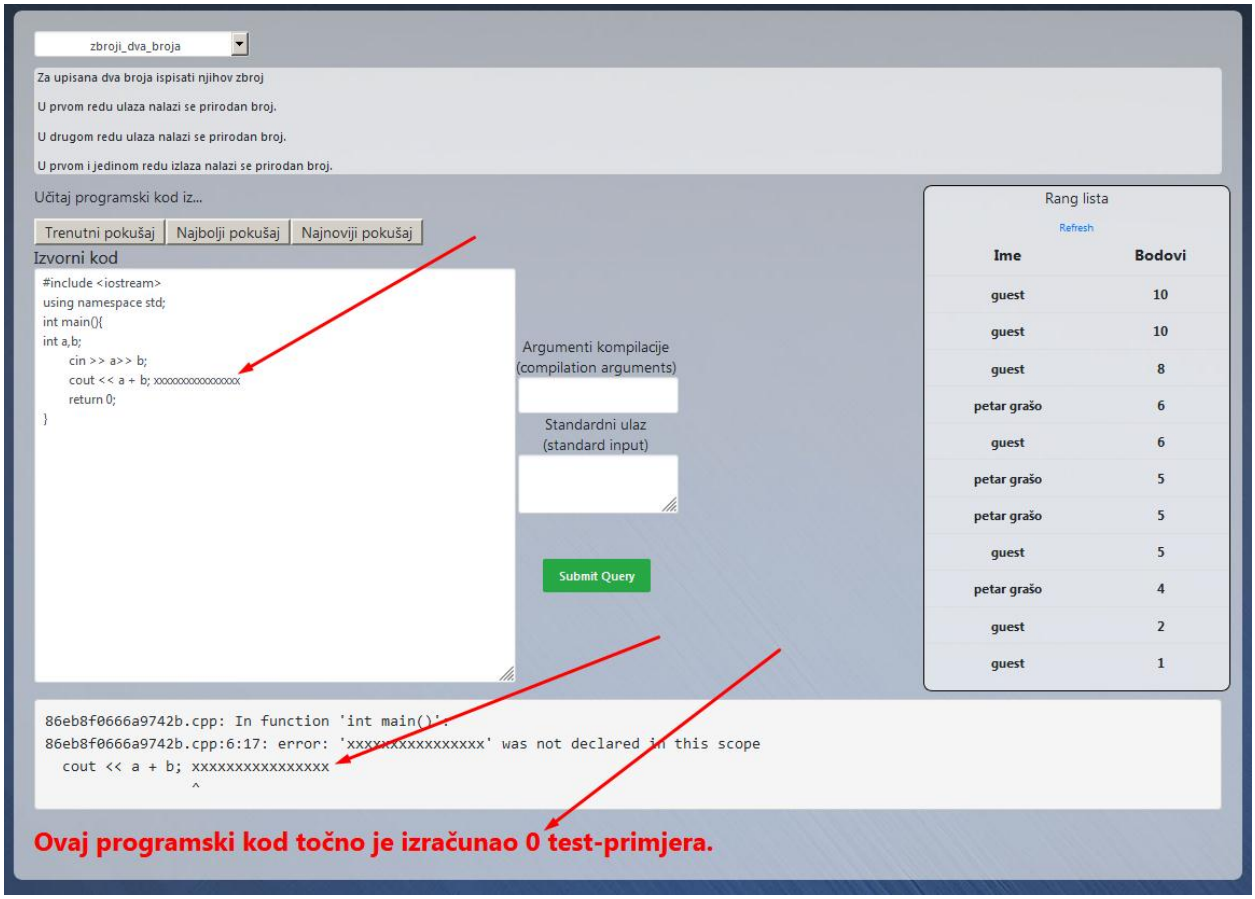


Slika 3.5.6. U ovom je slučaju predan nekakav ispravan programski kod koji koristi svojstva mjesta gdje se njegov rezultat ispisuje kako bi ispisao nekakvu vertikalnu sinusoidu.



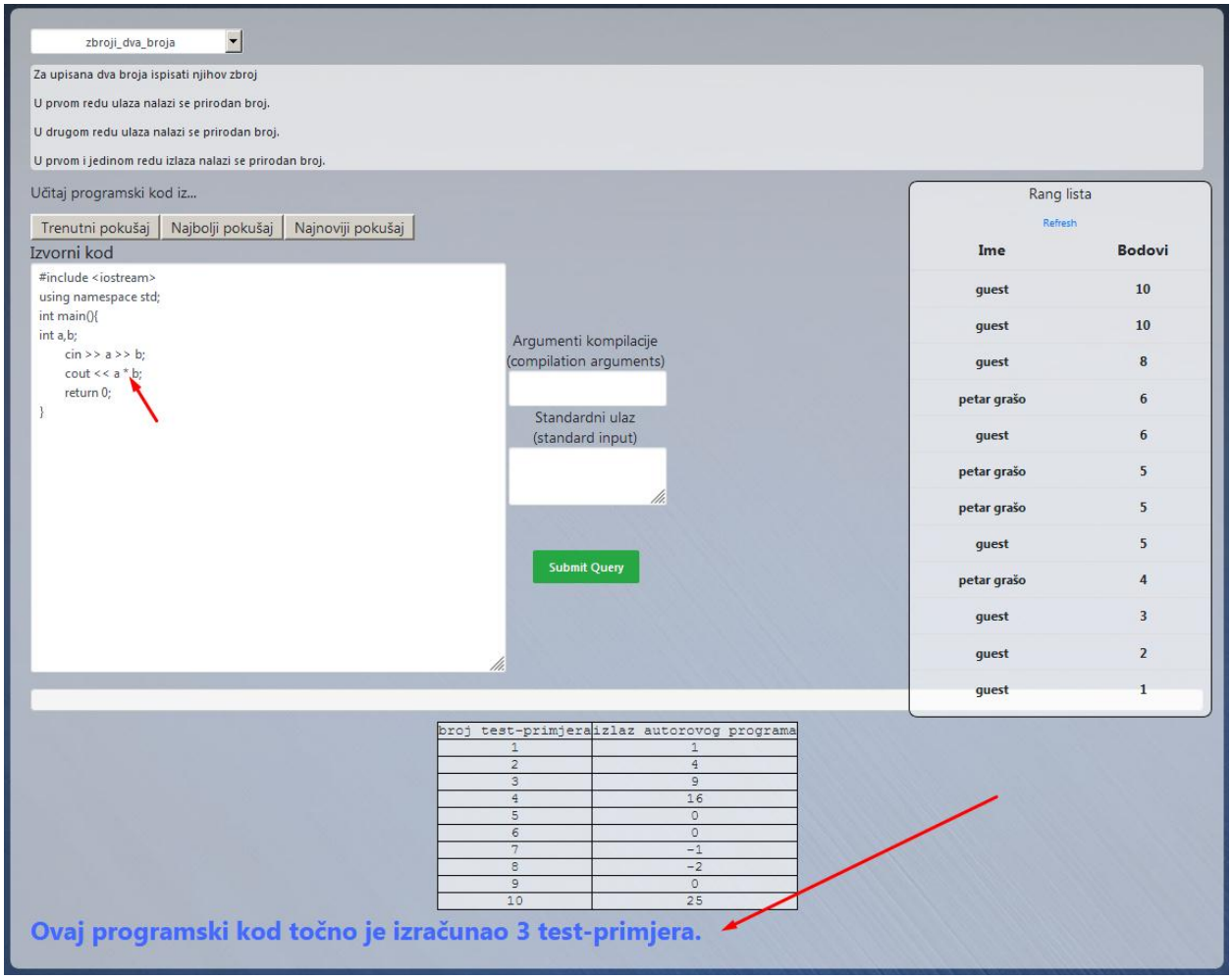
Slika 3.5.7. Na ovoj slici vidljiv je razmještaj i vanjština elemenata sučelja kada se odabere neki zadatak.

Kada se korištenjem padajućeg izbornika pri vrhu sučelja odabere zadatak, automatski će biti učitan tekst zadatka kao i tri tipke pomoću kojih je lako brzo učitati programski kod iz pokušaja trenutne sjednice, pokušaja koji je zavrijedio najviše bodova ili pokušaja koji je bio poslan u prethodnoj sjednici. Ovi programski kodovi bit će spremani u bazu podataka samo kada zavrijede barem jedan bod (barem jedan njihov izlaz odgovara očekivanom izlazu). Također, kako je vidljivo na slici 3.5.7., bit će učitana i rang lista za odabrani zadatak, koja navodi korisnike koji su predali programski kod koji je ostvario barem jedan bod, padajuće prema bodovima toga pokušaja. Ovo se sve događa prije nego se preda nekakav programski kod.



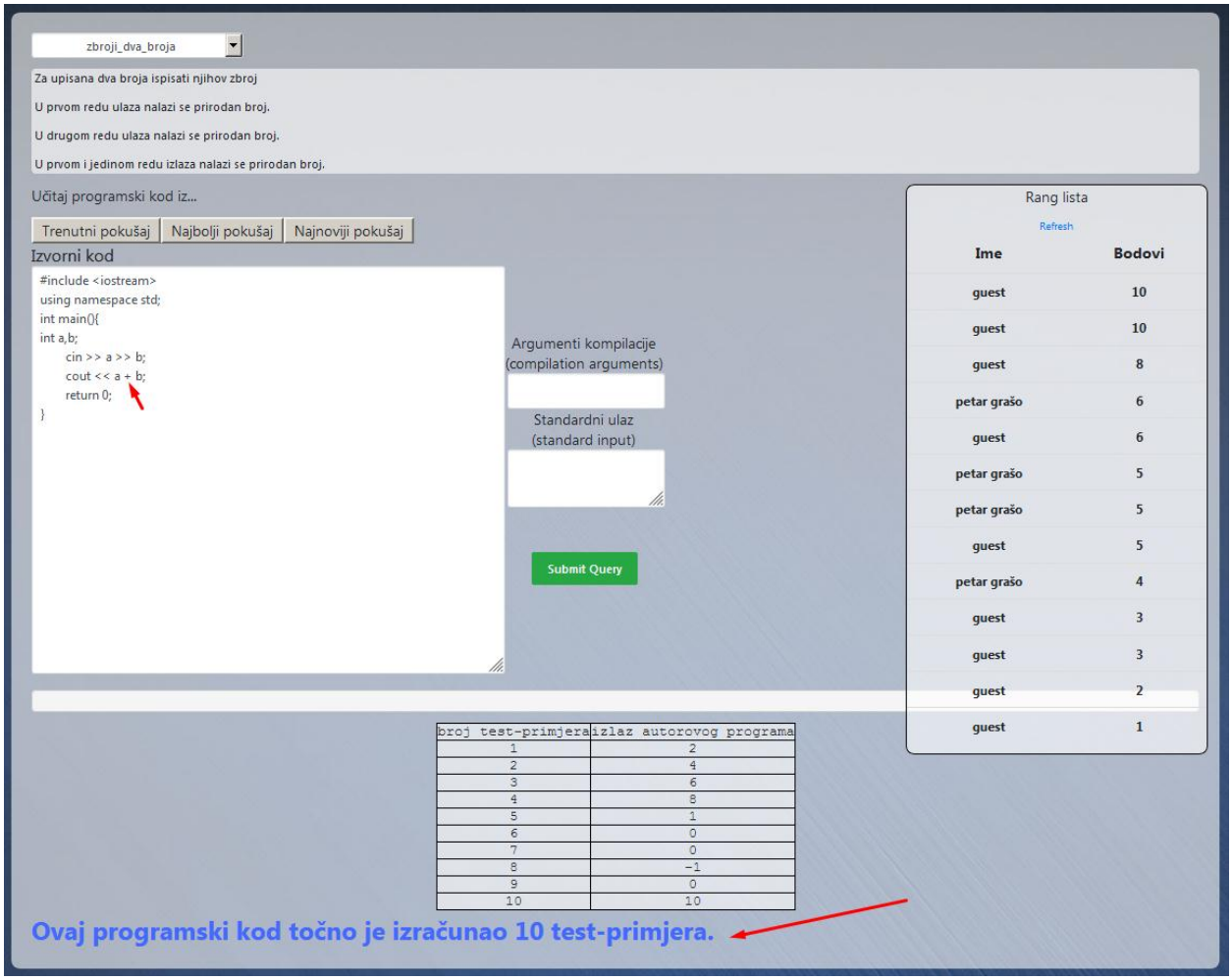
Slika 3.5.8. Pokušaj predaje neispravnog programskog koda kada je selektiran zadatak.

Ukoliko se pokuša predati sintaktički neispravan programski kod kada je selektiran zadatak, on će zavrijediti nula bodova i bit će prijavljena greška u kodu kao što je vidljivo na slici 3.5.8.



Slika 3.5.9. Pokušaj predaje djelomično ispravnog programskog koda kada je selektiran zadatak.

Kod sa slike 3.5.9. predstavlja sintaktički ispravan, ali logički neispravan programski kod. Igrom slučaja, izlaz ovog programskog koda odgovarao je očekivanom u slučaju tri test-primjera. S obzirom na to, ovaj kod zavrijedio je 3 boda i njegov autor biva stavljen na rang listu na odgovarajuće mjesto, a programski kod ovog pokušaja bit će spremljen u bazu podataka, zajedno s imenom autora, vremenom spremanja i brojem bodova.



Slika 3.5.10. Pokušaj predaje djelomično ispravnog programskog koda kada je selektiran zadatak.

Kod sa slike 3.5.10. predstavlja sintaktički ispravan i logički također ispravan programski kod. Ovaj programski kod očekivano je zavrijedio sve bodove. Njegov autor biva stavljen na rang listu na odgovarajuće mjesto, a programski kod ovog pokušaja bit će spremljen u bazu podataka, zajedno s imenom autora, vremenom spremanja i brojem bodova.

Ovo sučelje predstavljalo je najkompleksniji dio projekta. Bilo je potrebno realizirati mnogo provjera i procedura čitanja i pisanja podataka kako bi sve ispravno radilo. Slijedi detaljno objašnjenje važnijih dijelova koda.

```php
<?php
    $DB_URI = "localhost";
    $DB_USERNAME = "root";
    $DB_PASSWORD = '';
    $DB_NAME = "cpp_programming_competition";
?>
<html>
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
    <link href="vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
    <link href="css/simple-sidebar.css" rel="stylesheet">
    <script src="vendor/jquery/jquery.min.js"></script>
    <script src="vendor/bootstrap/js/bootstrap.bundle.min.js"></script>
</head>
    <?php
        //include("session.php");
        session_start();
```

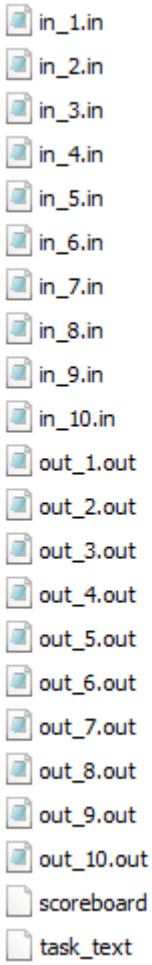
Slika 3.5.11. Dio programskog koda ovog sučelja

Na slici 3.5.11. vidljivo je kako programski kod ovog sučelja započinje definiranjem varijabli potrebnih za kvalitetnu vezu na bazu podataka, navođenjem CSS biblioteka koje se koriste i pozivom PHP funkcije *session_start()*. Ovaj poziv osigurava da kontrola programa prepozna *username* zapamćen u početnoj PHP skripti, radi potrebe sjednice. Dalje, u kodu, naveden je i vanjski CSS dizajn koji se koristi.

```php
<?php
    $scandir_result = scandir("./tasks");
    for($i = 2; $i < count($scandir_result); $i++){//$i=0 and $i=1 are . and ..
        if(isset($_POST["task_selection"])){
            if($_POST["task_selection"] == $scandir_result[$i])
                echo '<option id="select_select0" selected>'.$scandir_result[$i]."</option>";
            else
                echo '<option id="select_select1">'.$scandir_result[$i]."</option>";
        }
        else
            echo '<option id="select_select2">'.$scandir_result[$i]."</option>";
    }
?>
```

Slika 3.5.12. Isječak programskog koda koji osigurava popunjenost padajućeg izbornika za izbor zadatka.

Zadaci se spremaju u obliku datoteka u datotečnoj strukturi projekta. To je realizirano na način da se sprema tekst zadatka, ulazni i izlazni test primjeri. U direktoriju *tasks* u korijenskoj visini projekta nalaze se direktoriji od kojih svaki predstavlja jedan zadatak i nosi navedene stvari. Ovako izgleda unutarnja struktura direktorija */tasks/mnozenje_dva_broja/*:



Slika 3.5.13. Izgled unutarnje strukture direktorija */tasks/zbroji_dva_broja/*.

datoteka *task_text* nosi tekst zadatka, datoteka *scoreboard* nosila je podatke o rang listi, ali je ta funkcionalnost bila prebačena na bazu podataka, a ostale datoteke su ili ulazne ili izlazne datoteke od kojih svaka nosi točno jedan test podatak, ulazni ili izlazni.

Na slici 3.5.14. vidljiv je isječak programskog koda zadužen za popunjavanje rang liste u sklopu svakog od zadataka.

```php
if(isset($_POST["task_selection"])){
    $mysqli = new mysqli($DB_URI, $DB_USERNAME, $DB_PASSWORD, $DB_NAME);
    if (mysqli_connect_errno()){
        echo "Failed to connect to MySQL: " . mysqli_connect_error();
    }
    $scoreboard = "SELECT * FROM runs WHERE task = '" . $_POST["task_selection"] . "' ORDER BY
        points DESC, date_time DESC;";
    if (!($scoreboard = $mysqli->query($scoreboard))){
        echo "scoreboard insert query failed: (" . $mysqli->errno . ") " . $mysqli->error;
    }
    while($row = $scoreboard->fetch_assoc()){
        echo '<tr><th id="score_unutar_ime">'.$row["user"].'</th>';
        echo '<th id="score_unutar_bodovi">'.$row["points"].'</th></tr>';
    }
}
else
    echo "Ovdje će biti prikazana rang-lista za neki zadatak - kada bude odabran!";
```

Slika 3.5.14. Isječak programskog koda zaduženog za popunjavanje rang liste u sklopu svakog od zadataka.

```php
226                 if(isset($standard_input))
227                     file_put_contents($filenames.".in" , $standard_input);
228                 if(isset($code_to_run))
229                     file_put_contents($filenames.".cpp" , $code_to_run);
230                 if(!isset($shell_command_args))
231                     $shell_command_args = "";
232                 if(!isset($_POST["task_selection"]))
233                     echo '<p id = "shell_naredba">Shell naredba koja će se izvršiti</p>';
234                 $shell_command = "g++ " . $shell_command_args . " -o " . $filenames . ".exe " . $filenames . ".cpp && " . $filenames
                        . ".exe < " . $filenames . ".in 2>&1";
235                 if(!isset($_POST["task_selection"]))
236                     echo "<div id='shell_command'><pre>$shell_command</pre></div>";
237                 if(file_exists($filenames . ".cpp"))
238                     $output = cmd_exec($shell_command, $stdout, $stderr);// or die("shell command execution fail");
239                 if(isset($output)){
240                     echo '<div id="stderr"><pre id="stderr_pre">';
241                     /*for($i = 0; $i < count($stdout); $i++)
242                         echo $stdout[$i];*/
243                     for($i = 0; $i < count($stderr); $i++)
244                         echo $stderr[$i];
245                     echo "</pre></div>";
246                     //echo "<pre>" . $stdout . $stderr . "</pre>";
247                 }
```

Slika 3.5.15. Ključni dio programskog koda ovog projekta.

Na slici 3.5.15. vidljiv je ključni dio programskog koda ovog projekta. Programski kod predan od strane korisnika, kao i eventualni argumenti komandne linije, spremaju se u datoteke u korijensku razinu projekta. Definira se g++ *shell* naredba i u redu 238 ona se izvršava. PHP poslužitelj doslovno započinje novi proces na operativnom sustavu i u njemu izvršava sistemski poziv prema *shell* naredbi. *Shell* naredba bit će implementirana preko CMD naredbe i prvo će prevesti, a zatim pokrenuti predani programski kod. Kada ovo završi, nadležna funkcija prenijet će rezultat rada programa u PHP varijablu *$output*.

Kod na slici 3.5.15. kod je za scenarij neodabranog zadatka. Ukoliko je odabran zadatak, izvodi se programski kod nešto niže u programu, na slici 3.5.16.

```
248                 $points = 0;
249                 if(isset($_POST["task_selection"])){
250                     if(isset($code_to_run) && !(strpos(implode($stderr), 'error') !== false)){//assumes that there are compile-time
                        errors in the code only if 'error' is present in compiler return, $stderr
251                         echo '<table id="run_results" >';
252                         echo "<tr><td>broj test-primjera</td>";
253                         echo "<td>izlaz autorovog programa</td></tr>";
254                         for($i = 1; $i <= 10; $i++){
255                             $shell_command = $filenames . ".exe < ./tasks/".$_POST["task_selection"]."/in_" . $i . ".in 2>&1";
256                             //echo "<script type='text/javascript'>alert('$shell_command');</script>";
257                             cmd_exec($shell_command, $stdout, $stderr);// or die("shell command execution fail");
258                             //echo "<script type='text/javascript'>alert('".trim(Implode($stdout))."');</script>";
259                             //echo "<script type='text/javascript'>alert('".trim(file_get_contents("./tasks/".$_POST[
                                "task_selection"]."/out_" . $i . ".out"))."');</script>";
260                             if(trim(Implode($stdout)) == trim(file_get_contents("./tasks/".$_POST["task_selection"]."/out_" . $i . "
                                .out")))
261                                 $points++;
262                             if(count($stdout) == 0){
263                                 if(isset($stdout[$i]))
264                                     echo "<tr><td>".$i."</td><td>".$stdout[$j]."</td></tr>";
265                                 else
266                                     echo "<tr><td>".$i."</td><td></td></tr>";
267                             }
268                             else{
269                                 echo "<tr><td>".$i."</td><td>";
270                             for($j = 0; $j < count($stdout); $j++)
271                                 if(isset($stdout[$j]))
272                                     echo $stdout[$j]."<br />";
273                             echo "</td></tr>";
274                             }
275                         }
276                         echo "</table>";
277                     }
```

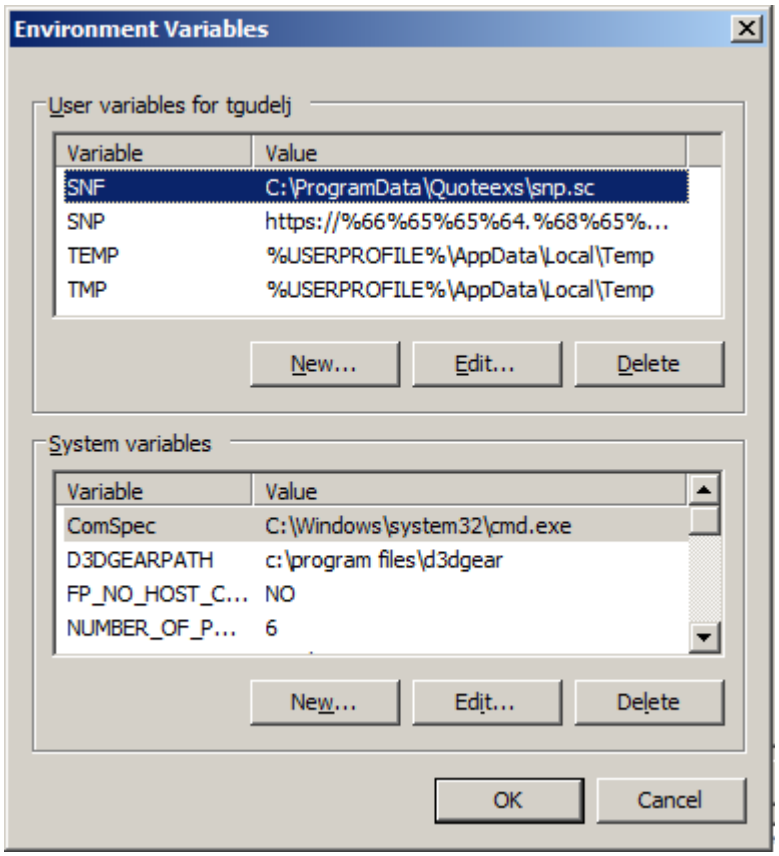
Slika 3.5.16. Ključni dio programskog koda ovog projekta ukoliko je selektiran zadatak.

Ovaj kod je vidljivo sličan kodu sa slike 3.5.15, ali je razlika u tome što ovaj kod, s obzirom da je selektiran zadatak, mora usporediti izlazne podatke programa s onima zapisanima u pripadajućoj .out (izlaznoj) datoteci. Ukoliko su oni jednaki, autorov program nagrađen je jednim bodom. Ovo se ponavlja deset puta, jer ima deset test primjera. Ulazni se podaci programu prosljeđuju kroz *shell* naredbu pomoću znaka <, odnosno preusmjerenja podataka iz onoga što je na desnoj strani naredbe u ono što je na lijevoj strani naredbe, kako je vidljivo u redu 257.

Nadalje, u programskom kodu ovog sučelja još ima funkcionalnost spremanja podataka o ovom pokušaju pokretanja programskog koda u bazu podataka, u relaciju *runs*. Iz te relacije se pune rang liste, na temelju imena zadatka i poretka autora prema broju bodova. Programski kod se u bazu podataka sprema kao *Base64* reprezentacija istog kako bi se spriječili namjerni ili nenamjerni slučajevi narušavanja sintaktičke ispravnosti SQL naredbe za spremanje u bazu podataka.

Bitno je napomenuti kako je osposobljen prevoditelj za C++, g++. Prvo je bilo potrebno instalirati programski paket MinGW, ali rezultat te operacije jest da je aplikacija g++ dodana u direktorij prema strukturi *C:\MinGW\bin*. Korišteni PHP server, Apache, nalazi se u strukturi *C:\xampp\Apache\bin*. U ovakvim okolnostima, sustav iz ovog projekta uporno bi vraćao grešku kako aplikacija g++ nije mogla biti pronađena. Kako bi se ovo saniralo, potrebno je lokaciju navedene aplikacije dodati u globalnu varijablu putanje sustava, *Path*. To se postiže različito na

različitim inačicama operativnog sustava Windows, ali u slučaju ovog sustava, postiže se klikom na tipku Start, zatim desnim klikom na „Computer“ i odabiranjem opcije „Properties“. Na novootvorenom prozoru (naime, radi se o slici 2.3.) kliknuti *Advanced System Settings*, a na novootvorenom prozoru, u kartici *Advanced* kliknuti tipku *Environment Variables*. Na novootvorenom prozoru, koji izgleda kao na slici 3.5.17., potrebno je pronaći varijablu Path u skupu varijabli pod rubrikom *System variables*.



Slika 3.5.17. Varijable okruženja sustava.

Varijablu *Path* potrebno je urediti i na kraj polja *Variable value* dodati sljedeći niz znakova:

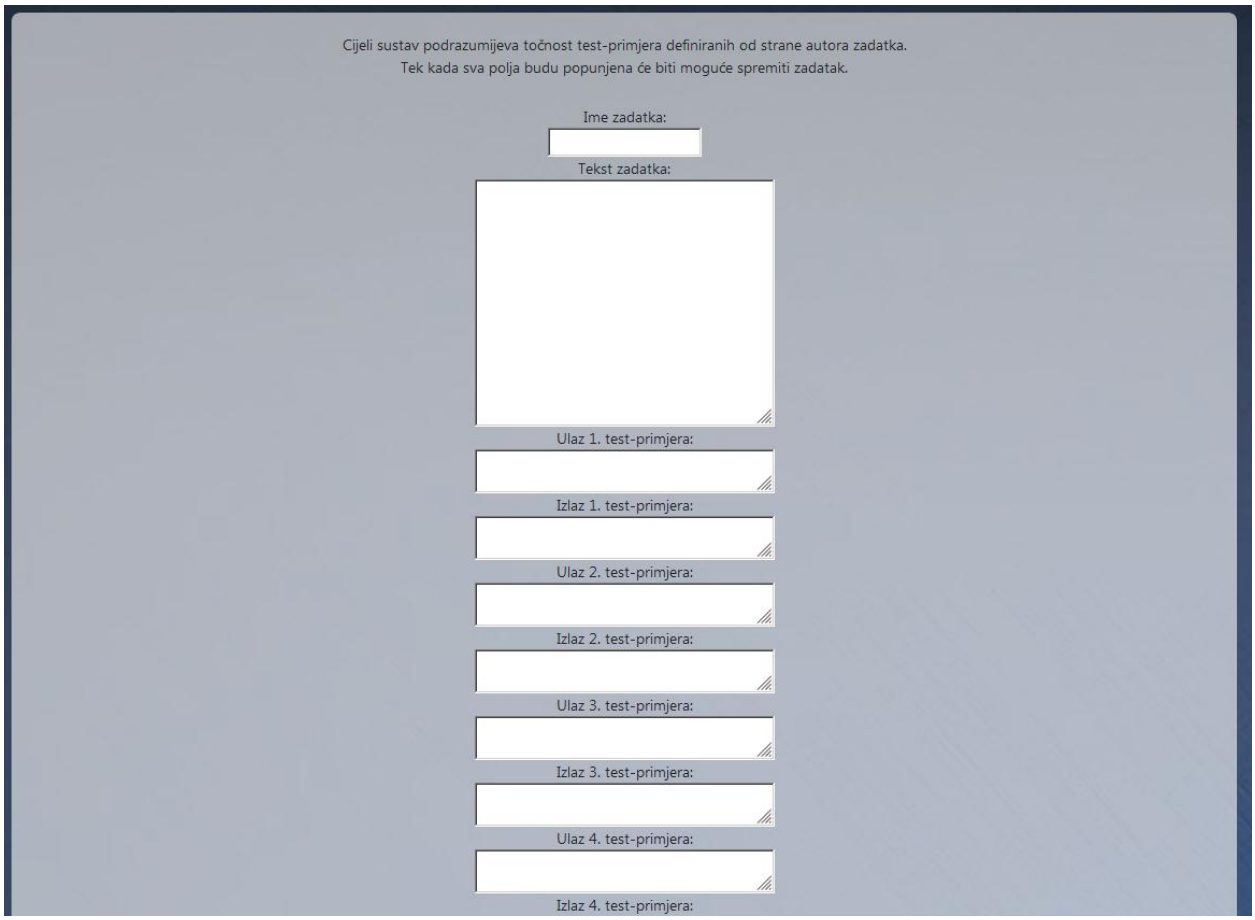
;C:\MinGW\bin;

Sada, nakon ponovnog pokretanja Apache poslužitelja, Apache poslužitelj će prepoznavati aplikaciju g++ neovisno iz kojeg direktorija se promatra.

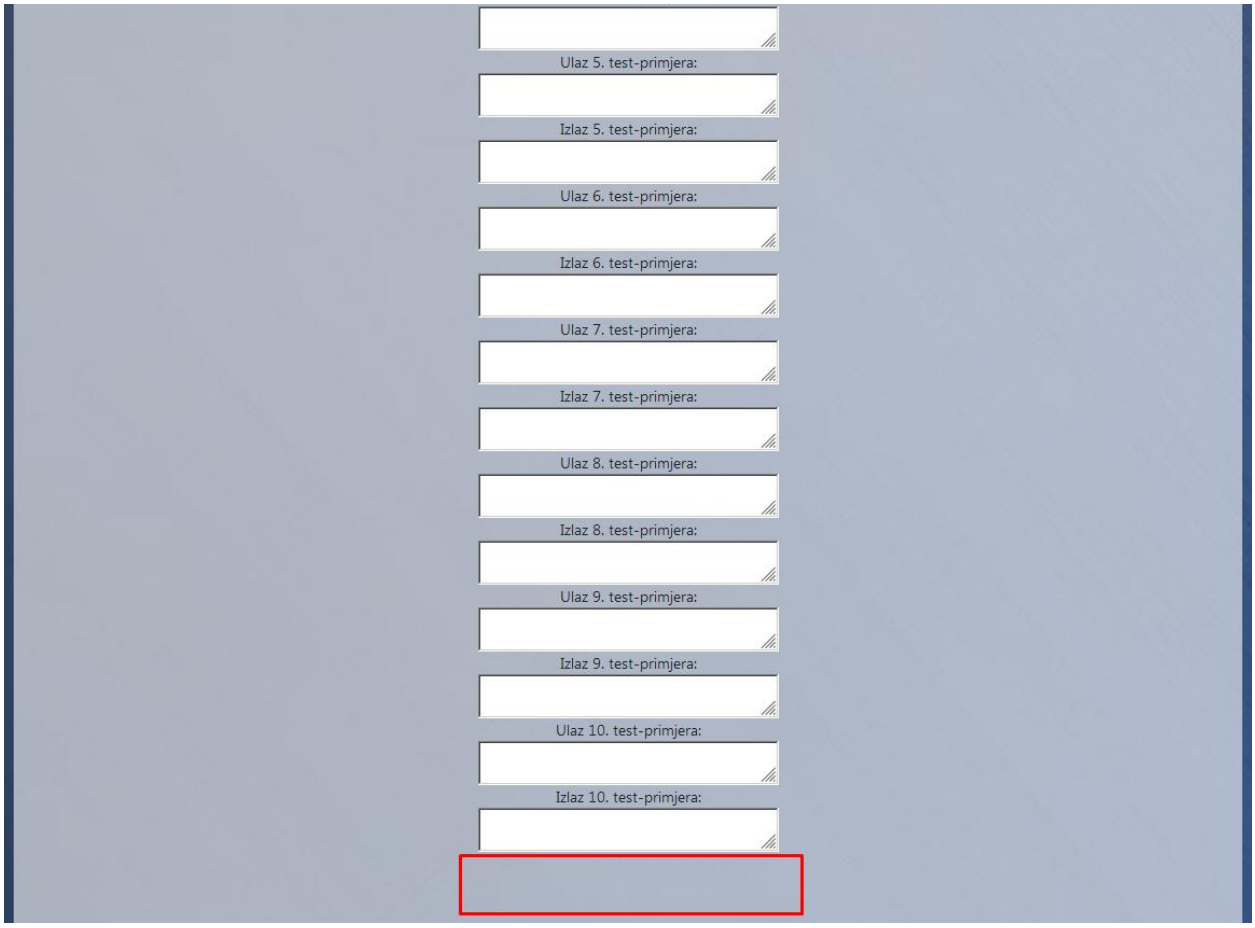
3.3. Sučelje za dizajniranje vlastitog programskog problema – zadatka

Sučelje za dizajniranje vlastitog programskog problema odnosno zadatka je također bitan dio ovog projekta. Kako bi sustav uopće mogao zaživjeti, potrebno je omogućiti korisnicima da

samostalno i donekle slobodno dodaju vlastite zadatke. Zadaci koji budu dodani i u konačnici nemaju smisla mogu jednostavno biti obrisani s datotečnog sustava (čime ih se efektivno briše i s cijelog sustava) ili samo zanemareni od strane korisnika.



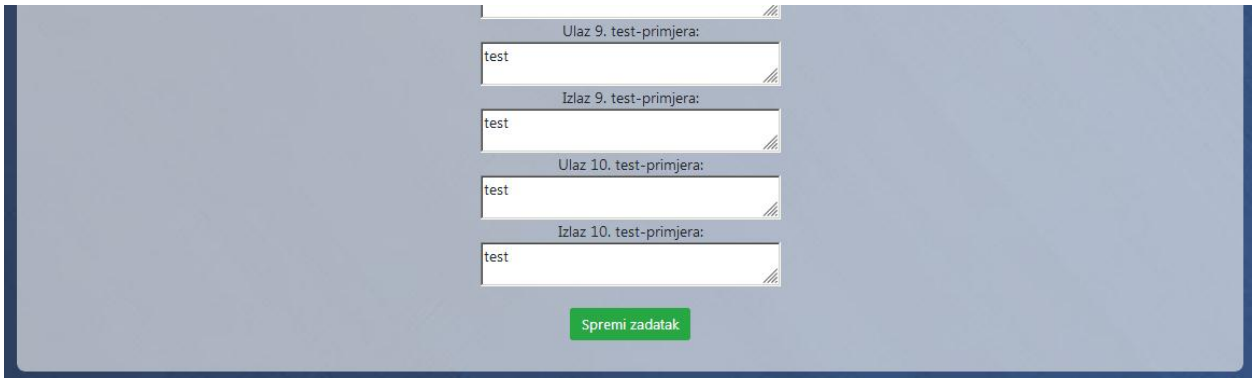
Slika 3.6.1. nastavak na idućoj stranici.



Slika 3.6.1. Vanjština sučelja za dodavanje vlastitog zadatka.

U kontekstu ovog sustava za zadavanje i rješavanje programskih problema, zadatak se sastoji od imena zadatka, teksta zadatka, deset ulaznih skupina podataka i deset izlaznih skupina podataka. Prema tome, postoji obveza popunjenja svih polja koja su prikazana na ovom sučelju. Ukoliko nisu svi podaci popunjeni, tipka za pokretanje procedure kreiranja zadatka neće biti raspoloživa.

U trenutku kada su ispunjena sva polja, navedena tipka bit će prikazana:



Slika 3.6.2. Vanjština sučelja za dodavanje vlastitog zadatka u trenutku popunjavanja svih polja.

Pritiskom na tipku sustav će kreirati novi direktorij i u njemu kreirati strukturu nalik na onu na slici 3.5.13.. Od tog trenutka, na glavnom sučelju sustava bit će moguće selektirati novododani zadatak, a njegova će rang lista biti prazna.

Programski kod ovog sučelja trivijalniji je u odnosu na programski kod glavnog sučelja i kao takav neće biti posebno navođen ovdje.

Dodatno obrazloženje kako kvalitetno kreirati zadatke pomoću ovog sučelja (takve zadatke koji su zapravo rješivi s punih 10 bodova) ponuđeno je u poglavlju 3.6..

3.4. Međuslojna programska funkcionalnost za odjavu

TODO: TGUDELJ: Ovdje će doći obrazloženje uz isječke koda

3.5. Međuslojna programska funkcionalnost za održavanje sjednice

TODO: TGUDELJ: Ovdje će doći obrazloženje uz isječke koda

3.6. Primjer jednog složenijeg test-primjera

Promotrimo zadatak *rimski_u_arapski*. Tekst ovog zadatka glasi, kako je zapisano u datoteci:

„Za dani broj zapisan u rimskom zapisu ispisati njegov arapski ekvivalent.

U prvom i jedinom redu ulaza nalazi se niz velikih slova engleske abecede bez razmaka.

U prvom i jedinom redu izlaza nalazi se prirodni broj.“

Ovaj tekst zadatka, kao i svakog zadatka, uređuje pravila pisanja ulaznih i izlaznih podataka. Ako u ovom tekstu piše da se u prvom i jedinom redu ulaza nalazi niz znakova bez razmaka, onda se ne smije među njima pojaviti razmak, u kojem bi slučaju najvjerojatniji ishod bio da će svi natjecatelji na tom zadatku uvijek dobiti nula bodova, jer će napisati program koji očekuje podatke drukčije nego kako će ih stvarno program dobiti. Jedan od test primjera iz ovog zadatka glasi

Ulaz:

MMXIV

Izlaz:

2014

Ukoliko je postojao razmak nakon znakova MM, to bi vjerojatno potaklo program nekog od korisnika da ni ne pročita preostala tri znaka, pa bi rezultat uvijek bio neispravan, za taj test primjer. Pravila pisanja test-primjera zadanih u tekstu zadatka treba se držati, a navedena pravila svakako treba napisati u tekstu zadatka. Bez ovih uvjeta, teško je natjecati se u ovom modelu natjecanja. I sami HONI i Z-trening su imali ovakav model dogovaranja svojevrsnih pravila s korisnikom sustava.

4. ZAKLJUČAK

U ovom diplomskom radu razvijeno je programsko rješenje koje korisnicima nudi mogućnost prevođenja i izvršenja programskog koda napisanog prema pravilima C++ jezika. Korisnici mogu vidjeti rezultate rada svog programa i dodatno doraditi program ako tako prohtiju. Osim ove osnovne funkcionalnosti, ovo softversko rješenje nudi funkcionalnost kreiranja i rješenja programskih problema i zadataka. Korisnici se mogu međusobno natjecati, s obzirom da postoji implementirana rang lista u sklopu svakog zadatka. Zadatak jest konkretan programski problem radi kojeg korisnik treba razviti C++ program koji na temelju unosa računa rezultate i ispisuje svojevrsna rješenja prema pravilima koja su zadana u zadatku. Programska rješenja korisnika nagrađena su brojem bodova srazmjernim točnosti podataka koje ispisuju. Ova funkcionalnost inspirirana je bivšim servisom Z-trening, koji je nudio slične usluge. Hrvatsko natjecanje HONI također funkcionira prema istom modelu. Ovaj softver predstavlja svojevrsno uzajamno djejstvo tehnologija za razvoj internetskog softvera i tehnologija za razvoj softvera za lokalno računalo. Tijekom implementacije ovog sustava otkriveno je mnogo situacija neodređenog ishoda koje je bilo potrebno razraditi kako bi sam proces predaje programskog koda, prevođenja, izvršenja, prikupljanja rezultata rada i usporedbe istih s očekivanima tekao glatko i stabilno. Bilo je potrebno razumjeti rad operativnih sustava „ispod haube“, kako bi se procedure prevođenja i izvršenja programa mogle ispravno i sigurno koristiti. Tijekom izrade ovog projekta, razvijen je detaljniji uvid u koliko zapravo razvoj softvera može biti ovisan o detaljima, a pogotovo o temeljima koje inženjer često postavi rano u procesu razvoja kada nekih implikacija takvog određenog načina postavljanja temelja ne može niti biti svjestan. Razvoj vlastitih vještina programiranja teče projekt po projekt i ovaj je projekt ostvario svoj učinak osobnog razvoja autora.

5. LITERATURA

[1] - Hrvatsko otvoreno natjecanje u informatici 2006-2016, dostupno na: http://hsin.hr/honi/arhiva/ [21.09.2018]

[2] - O HSIN-u, dostupno na: http://www.hsin.hr/o_hsinu/ [21.09.2018]

[3] - I. Šeparović: O HSIN-u, Zagreb, 2012. dostupno na: http://www.hsin.hr/o_hsinu/hsin1985-2011.pdf [21.09.2018]

[4] - Zadaci. dostupno na: http://hsin.hr/honi/arhiva/2006_2007/kolo1_zadaci.pdf [21.09.2018]

[5] - Plain Text Offenders, dostupno na: http://plaintextoffenders.com/ [21.09.2018]

[6] - Operacijski sustav, dostupno na: https://hr.wikipedia.org/wiki/Operacijski_sustav [21.09.2018]

[7] - Navigating the HTML, dostupno na: https://developer.mozilla.org/en-US/docs/Tools/Page_Inspector/How_to/Examine_and_edit_HTML [21.09.2018]

[8] - A. Malhotra, HTML Tags, dostupno na: https://tutorial.techaltum.com/htmlTags.html [21.09.2018]

[9] - Custom Design: Css Basics, dostupno na: https://en.support.wordpress.com/custom-design/css-basics/ [21.09.2018]

[10] - JavaScript, dostupno na: https://hr.wikipedia.org/wiki/JavaScript [21.09.2018]

[11] - JavaScript in Visual Studio Code, dostupno na: https://code.visualstudio.com/docs/languages/javascript [21.09.2018]

[12] - JQuery, dostupno na: https://jquery.com/ [21.09.2018]

[13] - Apache: HTTP Server Project, dostupno na: https://httpd.apache.org/ [21.09.2018]

6. SAŽETAK

Naslov: Web platforma za zadavanje i rješavanje programskih problema

Web platforma za zadavanje i rješavanje programskih problema je web platforma koja korisniku pruža funkcionalnost prevođenja i izvršenja programskog koda napisanog u C++ programskom jeziku. Osim te osnovne funkcionalnosti, ova web platforma pruža podršku za dizajn programskih problema i zadataka koje zatim svi korisnici platforme mogu interaktivno riješiti svojim programskim kodom. Navedeni programski kod prvo biva testiran na sintaktičku ispravnost, a zatim, prema modelu ulaz-izlaz, na točnost, preciznost i ispravnost rada programa. Što je korisnikov program točnije rješenje (prema modelu usporedbe ulaza i izlaza podataka u kontekstu tog programa), to je korisnik nagrađen većim brojem bodova. Sustav nudi kvalitetnu i sigurnu implementaciju registracije i prijave, kao i rang listu svih korisnika u sustavu prema njihovoj sumi osvojenih bodova.

Ključne riječi: web platforma, programski problem, zadatak, prevođenje, izvršavanje, C++

7. ABSTRACT

Title: Web platform for designing and overcoming program problems

The Web platform for designing and overcoming program problems is a web platform that presents its user with the functionality of compiling and executing of program code written in C++ programming language. Aside from this basic functionality, the web platform offers support for designing one's own programming problems and tasks that can ultimately be overcome by all of the platform's users in an interactive fashion, by having said users provide program code that would offer a solution to the problem. The provided program code is firstly tested for syntax compliance, followed by testing for solution accuracy and fidelity based on the input-output model. The more accurate the user's solution (based on comparing input and output data in terms of the solution), the more points the user is awarded. The system provides quality and secure login and registration implementation, as well as a scoreboard listing of all users within the system ordered by their total awarded points.

Keywords: web platform, program problem, task, compilation, executing, C++

8. ŽIVOTOPIS

Tomislav Gudelj rođen je 18. kolovoza 1994. godine u Osijeku. Po završetku osnovne škole Bilje u Bilju, upisuje Treću gimnaziju u Osijeku i maturira 2013. godine, a zatim upisuje preddiplomski studij računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija. Tijekom akademske godine 2015./2016. završava preddiplomski studij i upisuje diplomski studij računarstva na istom fakultetu. Uz položene prve i druge godine diplomskog studija i obranu diplomskog rada, diplomirat će s titulom magistra računarstva. 2015. godine sudjeluje na europskom natjecanju u programiranju CERC 2015, predvođenom od strane udruge ACM ICPC.

PRILOZI

Cjelokupan izvorni kod aplikacije, programski resursi, sama digitalna verzija ovog dokumenta u .docx i .pdf formatu nalazi se na CD-u priloženom uz diplomski rad.