Web aplikacija za provjeru programskog koda

Miletić, Bruno

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

Sveučilišni studij

Web aplikacija za provjeru programskog koda

Završni rad

Bruno Miletić

Osijek, 2022.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 19.09.2022.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime Pristupnika:	Bruno Miletić
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. Pristupnika, godina upisa:	R 4402, 22.07.2019.
OIB Pristupnika:	99710774715
Mentor:	Doc. dr. sc. Tomislav Galba
Sumentor:	Izv. prof. dr. sc. Alfonzo Baumgartner
Sumentor iz tvrtke:	
Naslov završnog rada:	Web aplikacija za provjeru programskog koda
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak završnog rad:	Napisati web aplikaciju koristeći Spring programski okvir koja će imati mogućnost unosa programskog kod-a i provjere koristeći odgovarajući kompajler (podržati minimalno jedan programski jezik npr. Java). Uz to, potrebno je opisati Spring programski okvir.
Prijedlog ocjene završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	19.09.2022.
Datum potvrde ocjene od strane Odbora:	21.09.2022.

Potvrda mentora o predaji konačne verzije rada:	Mentor elektronički potpisao predaju konačne verzije.
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 21.09.2022.

Ime i prezime studenta:	Bruno Miletić
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R 4402, 22.07.2019.
Turnitin podudaranje [%]:	7

Ovom izjavom izjavljujem da je rad pod nazivom: **Web aplikacija za provjeru programskog koda**

izrađen pod vodstvom mentora Doc. dr. sc. Tomislav Galba

i sumentora Izv. prof. dr. sc. Alfonzo Baumgartner

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:

Sadržaj

1. Uvod

Rad u programerskom okružju često iziskuje kreativnost te preciznost kako bi se došlo do optimalnog te svima zadovoljavajućeg rješenja. Profesija je to koja nije tradicionalna te nerijetko se programer pronalazi u situacijama kad mu je potrebno više vremena da razmisli o samom rješenju nego što mu je potrebno da svoj koncept prevede u kod. Također, programerski posao najčešće nije ograničen mjestom izvođenja te nudi prednosti i otvara prilike kao što je i rad izvan ureda. Prema tome, uvijek je korisno programerima pružiti razvojno okruženje koje ne mora nužno raditi na računalu, već mu se pristupa preko interneta u formi internetske stranice ili web aplikacije. U ovom završnom radu razmotrit će se područje rada, zahtjevi na rješenje s programske strane samog zadatka, dizajn te implementacija konačne aplikacije. Unutar programskog rješenja bit će prikazane sve rabljene tehnologije te dizajn i njihova konkretna uporaba.

U drugom dijelu rada bit će pregledano područje rada. Pet alternativnih rješenja koja postižu isti željeni rezultat bit će prikazani te opisane. Treće poglavlje će govoriti o zahtjevima na programsko rješenje koje će se temeljiti na prethodno navedenim postojećim rješenjima, njihove prednosti i kvalitete te nedostatci. Nadalje, bit će opisani pojmovi čije je shvaćanje ključno za dobru realizaciju samog zadataka predstavljenog ovim radom što uključuje pojmove kao prevoditelj, Spring i web aplikacija. Posljednje poglavlje prikazat će i opisati konačnu implementaciju rješenja te njegovo korisničko iskustvo sa potencijalnim nadogradnjama i nedostatcima.

1.1 Zadatak završnog rada

Zadatak završnog rada je napisati web aplikaciju koristeći Spring programski okvir koja će imati mogućnost unosa programskog kod-a i provjere koristeći odgovarajući kompajler (podržati minimalno jedan programski jezik npr. Java). Uz to, potrebno je opisati Spring programski okvir.

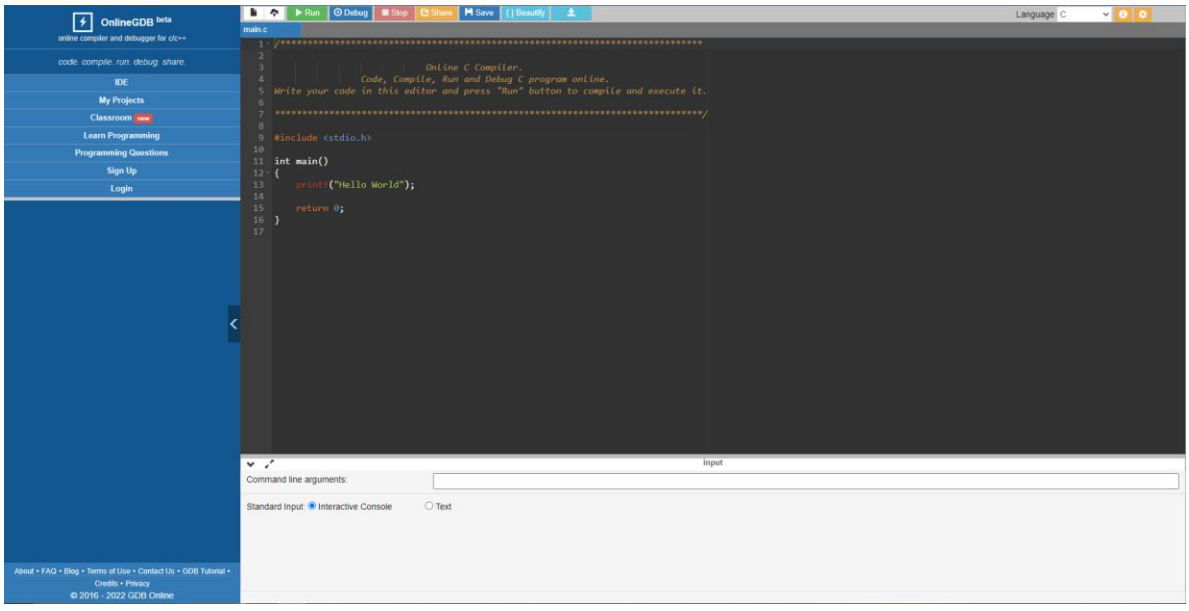
2. Pregled područja rada

Unutar ovog dijela rada bit će prikazana postojeća rješenja koja su služila kao primjer konačnog rješenja ovog zadatka. Odabranim rješenjima također će biti opisane njihove karakteristike te prednosti na osnovu postavljenog zadatka.

2.1. OnlineGDB

OnlineGDB je web aplikacija koja omogućuje provjeru programskog koda unutar bilo kojeg internetskog preglednika. Najveća prednost ovog rješenja je što je potpuno besplatno te ne zahtjeva nikakvu formu registracije od korisnika. Aplikaciju od drugih rješenja ističu minimalistički ,ali i pomalo zastarjeli dizajn te pouzdanost. Neke od mogućnosti koje OnlineGDB pruža su:

- Provjera napisanog programskog koda
- Mogućnost režima rada otkrivanja grešaka
- Izbor od gotovo trideset podržanih programskih jezika
- Automatska usluga isticanja koda i grešaka sintakse
- Maksimalno dopušteno izvođenje do 10 sekundi



Slika 2.1.1 – Prikaz korisničkog sučelja aplikacije OnlineGDB

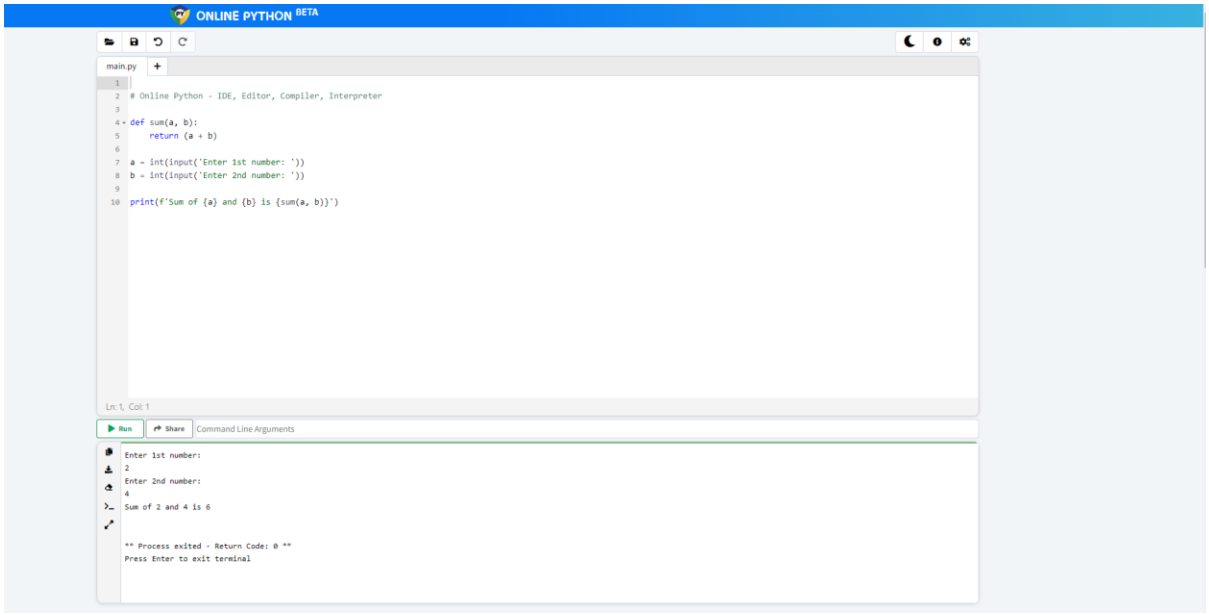
Unatoč svim prednostima, OnlineGDB također ima svoje nedostatke, od kojih su najveći nemogućnost korištenja korisnički napisanih biblioteka pristup internetu unutar samog napisanog koda je ograničen.

OnlineGDB prevoditelj odlikuje čist dizajn sa minimalističkim pristupom. Pristupačan te besplatan uvijek je dobar odabir za provjeru programskog koda zadovoljavajući osnovne potrebe svakoga korisnika. Najviše koristan amaterskim te rekreativnim zadatcima testiranja i provjere jednostavnijih isječaka koda.

2.2. Online Python

Online Python je web aplikacija namijenjena provjeri programskog koda napisanog u Python programskom jeziku. Pruža jednostavno i moderno sučelje te nudi nekolicinu prednosti:

- Mogućnost lokalnog spremanja koda
- Velik broj postavki za vizualne izmjene stranice kao što su tamni način te promjena fonta
- Ugrađene teme
- Terminal
- Ugrađene kratice koje automatski pokreću napisani kod
- Besplatno korištenje
- Više podržanih programskih jezika

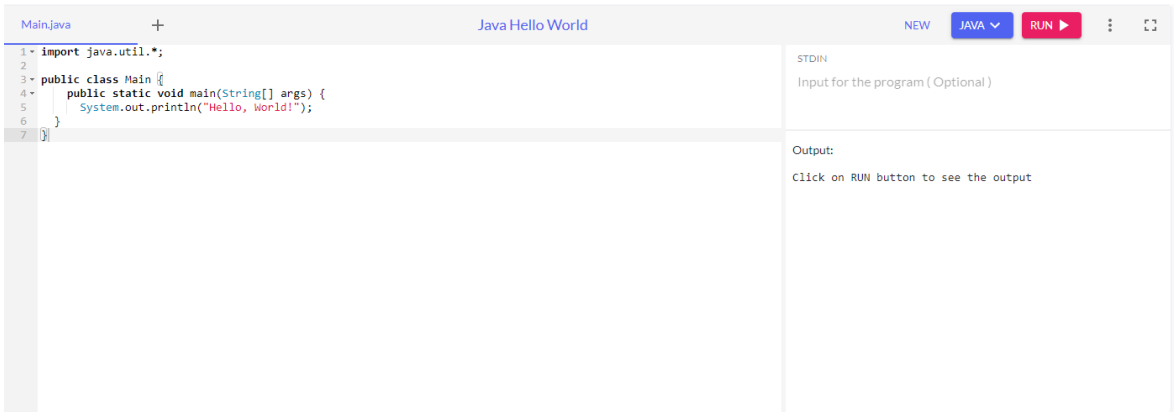


Slika 2.2.1 – Korisničko sučelje Online Python

Python online nudi minimalističko moderno sučelje ugodno oku korisnika. Nudi sve osnovne funkcionalnosti koje prevoditelj mora ispunjavati, ne zahtjeva registraciju niti prijavu te je potpuno besplatan. Također nudi i podršku nekim vanjskim bibliotekama.

2.3. OneCompiler

OneCompiler je Internet prevoditelj koji kao i dosadašnja rješenja nudi prevoditelje za više od 30 programskih jezika. Nešto što ga ističe od dosadašnjih rješenja je opcija ugrađivanja unutar bilo koje web stranice pomoću HTML-a, štedeći tako i vrijeme i novac potreban za razvoj ovakvog rješenja. Također, nudi i API onim korisnicima koji nemaju poslužitelja na raspolaganju za rad prevoditelja.



Slika 2.3.1 – Korisničko sučelje OneCompiler-a

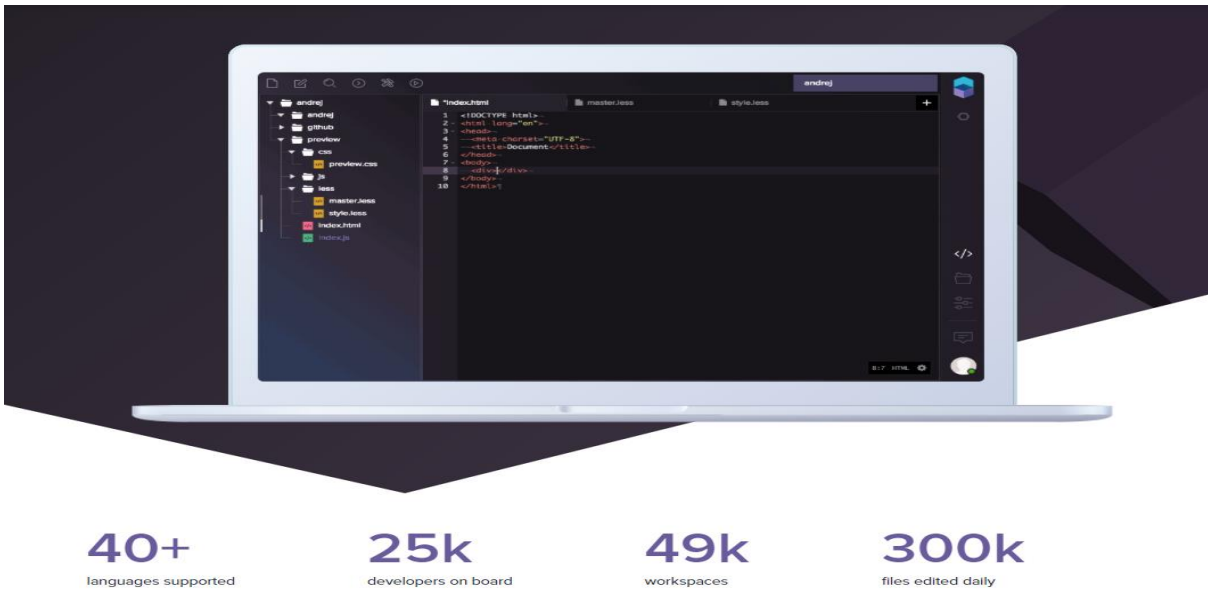
Također, OneCompiler nudi korisnicima opciju registracije kako bi njihov kod i povijest uređivanja ostali zapamćeni. Usluga korištenja i prije i poslije registracije ostaje besplatna što nije isto i sa API pozivima koji se naplaćuju ovisno o tome koliko poziva API dobije unutar razdoblja od mjesec dana.

2.4. CodeTasty

CodeTasty je rješenje sa najviše mogućnosti od svih dosad nabrojanih alternativa te je jedino rješenje koje se koristi tehnologijom oblaka. Nudi veliki broj prednosti:

- 4 različita plana plaćanja od kojih je jedan besplatan

- Početak korištenja bez instalacije za razliku od lokalnih prevoditelja
- Brzina koja odgovara onoj lokalnog korisnika
- Nema ograničenja na broj instaliranih ekstenzija
- Gotovo 40 jezika
- Terminal
- Opcija grupiranja korisnika u timove kako bi mogli pristupiti dijeljenim podatcima



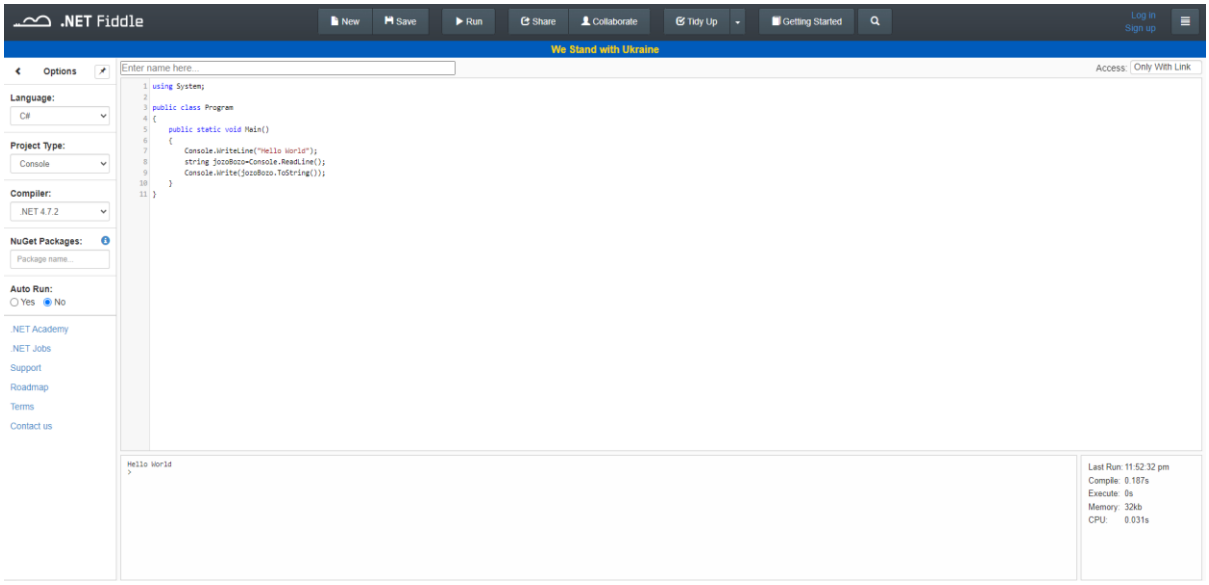
Slika 2.4.1. – CodeTasty sučelje

Jedino ograničenje koje CodeTasty rješenje postavlja na korisnika jest izbor platforme sa koje je moguć pristup. Zasada, pristup je moguć samo sa tri internet pretraživača: Google Chrome, Safari te Mozilla Firefox.

2.5. Dotnetfiddle

Dotnetfiddle je posljednje postojeće rješenje obrađeno u ovom radu. Kao i svi ostali, nudi funkcionalnost prevođenja korisnički definiranog programa, sposobnost lokalnog spremanja napisanog koda, dijeljenje istog te njegovo izvršavanje. Također, omogućuje automatsko „uljepšavanje“ koda popravljajući njegovu strukturu, poznaje i radi sa vanjskim bibliotekama te daje korisniku na odabir koja verzija prevoditelja se zahtjeva. Ograničen je isključivo na C# programski jezik te nema funkciju rada sa terminalom.

Ono što izdvaja Dotnetfiddle od ostalih prevoditelja je što nudi mogućnost uvoza bilo koje biblioteke koja bi se mogla pronaći i u lokalnom prevoditelju kao što je Visual Studio. Nadalje, prikazuje podatke koji opisuju proces izvođenja koda: vrijeme prevođenja, vrijeme izvođenja, korištena količina radne memorije te procesorsko vrijeme.



Slika 2.5.1. – Sučelje Dotnetfiddle

2.6. Općeniti zahtjevi

Iz prethodnih primjera vidljivo je kako se zadatak izrade ove web aplikacije lako može izdvojiti od konkurencije odabirom pravih značajki koje će sadržavati. Neke od tih prednosti su:

- Podržavanje više programskih jezika

Najveća prednost koju jedan ovakav projekt može ponuditi, a ujedino je i ona koja je najtraženija od strane korisnika, je opcija podržavanja više programskih jezika. Postojanje više od 700 programskih jezika iziskuje od ovakvih usluga da podrže barem nekolicinu onih najpopularnijih, od kojih su neki: Java, C, C++, Python te mnogi drugi. Većina ljudi koji se bavi ovakvih poslovima nije stručna u svakom od prethodno izlistanih jezika, pa funkcionalnost podržavanja više od jednog dojmi se kao osnovan zahtjev na ovakav tip web aplikacije.

- Funkcionalnost lokalnog spremanja napisanog koda

Funkcionalnost spremanja koda lokalno na uređaj putem kojeg korisnik pristupa ovakvoj usluzi također je poželjna funkcionalnost, jer eliminira potrebu za ponovnim pisanjem već postignutog. Nadalje, nudi dobru alternativu pohrani na poslužitelju, jer su lokalni podatci uvijek dostupni korisniku bez oslanjanja na samog poslužitelja na kojeg se može postaviti niz ograničenja: ograničeno vremensko izvođenje programa, ograničena količina memorije za pohranu te ograničena količina ostalih resursa koji se koriste prilikom prevođenja i izvođenja koda.

- Mogućnost registracije te spremanje napisanog na prostor poslužitelja

Ukoliko postoji potreba da korisnik podatke koje je napisao u sučelju ove web aplikacije spremi na prostor poslužitelja te ih dohvati sa poslužitelja u bilo kojem trenutku vremena, funkcionalnost registracije te organizacije datoteka unutar prostora poslužitelja razvrstana po korisnicima mora biti implementirana. Međutim, metode pohrane, registracije te autentifikacije korisnika moraju biti pažljivo implementirane kako bi se moglo pouzdano tvrditi da neće doći do neovlaštenih upada u sustav poslužitelja.

- Podržavanje vanjskih biblioteka

Rad sa korisnički definiranim bibliotekama je koristan alat jer je svijet programskih jezika dinamičan. Programski jezici su kao projekti koji se još razvijaju: ažuriraju se, definiraju se nove

biblioteke koje olakšavaju rad. Podržavanje vanjskih korisnički definiranih biblioteka pruža korisnicima priliku pisati kod punog potencijala, bez potrebe ponovnog pisanja već postignutog.

- Mogućnost pristupa neovisan o pristupnoj platformi

U idealnome slučaju, rad sa aplikacijom ovog tipa ne ovisi pristupa li joj se sa platforme kao što je to Windows ili Linux, ili pak operacijskog sustava nekog mobilnog uređaja kao što je Android. Za aplikaciju je poželjno da obavlja iste funkcionalnosti te pruža iste usluge neovisno o platformi i/ili internetskom pretraživaču korisnika.

- Opcije prilagodbe vizualnog dijela sučelja stranice

Svaki korisnik ima sklonosti prema načinu na koji se kod prikazuje unutar same web aplikacije, isticanja sintakse danog programskog jezika ili općenito izgledu korisničkog sučelja same web aplikacije. Poželjno je omogućiti korisniku barem male izmjene sučelja kako bi svatko mogao optimizirati svoj rad u aplikaciji.

3. Korištene tehnologije

U ovom poglavlju rada biti će opisane tehnologije i alati koji su korišteni prilikom izrade samog rada. Posebno će biti opisani Spring okvir te sami programski jezik Java.

3.1. Java

Java je objektno usmjereni programski jezik razvijen početkom 90-ih godina od strane američke tvrtke Sun Microsystems. Baziran je na programskom jeziku C++ sa pojednostavljenom sintaksom te mogućnošću izvođenja neovisno o platformi, što ga čini prikladnim za stvaranje programa za primjenu na internetu.

Java je postala popularan i koristan jezik zbog svojih izvrsnih značajki, koje imaju važnu ulogu u prethodno spomenutoj popularnosti ovog jezika. Značajke jave se još popularno nazivaju „Java BuzzWords“ ,a stvaratelji ga povezuju sa sljedećim izrazima:

- Jednostavan i prepoznatljiv

Stil pisanja koda je „čist“ i izrazito lagan za shvatiti. Uklonjeni su složeni koncepti jer Java ne koristi složene koncepte i značajke koje koriste jezici kao C i C++ od kojih su neki: eksplicitni pokazivači, predprocesorske datoteke, višestruko nasljeđivanje, go-to naredbe i jos drugih.

- Preveden i interpretiran

Uobičajeno, programski jezik može biti interpretiran ili preveden. Java integrira značajke prevedenih jezika sa fleksibilnošću interpretiranih. Javin prevoditelj – javac – prevodi izvorni kod u byte kod koji se potom izvodi na Java virtualnom stroju (JVM) što omogućuje izvođenje koda neovisno o korisničkoj platformi.

- Arhitekturalno neovisan

Java je arhitekturalno neovisan u smislu da program napisan na jednoj platformi ili operacijskom sustavu je neovisan o platformi i može se izvršiti u bilo kojem drugom okruženju. Drugim riječima, temeljen je na principu WORA („Write-once-run-anywhere“).

- Objektno usmjeren

Java je temeljena na objektno usmjerenim načelima pa se za Javu kaže da je čisti objektno usmjereni jezik. Omogućuje značajke objektno usmjerenih načela kao što su enkapsulacija, apstrakcija te nasljeđivanje. Gotovo sve unutar Jave je objekt, te su svi podatci spremljeni unutar objekta.

- Robustan

Java omogućuje rukovanje greškama prilikom izvođenja, pruža podršku automatskog sakupljača otpada (AGC) te rukovanje iznimkama. Izbjegava eksplicitnu uporabu pokazivača. Također, ima dobro razvijen sustav rukovanja memorijom što pomaže u eliminaciji grešaka jer provjerava kod i prilikom izvođenja i prevođenja. Rukovanje memorijom je također odrađeno automatski uz pomoć već spomenutog AGC-a.

- Siguran

Sigurnost je izrazito bitna u svakom programskom jeziku jer razvojem tehnologije razvijaju se i načini na koji se sloj sigurnosti može zaobići. Java podržava modifikatore pristupa, pokreće program unutar virtualnog stroja tako da daje mogućnost korisniku izvođenja aplikacije bez utjecaja na cjelokupan sustav.

- Višenitna

Višenitnost predstavlja rukovanje različitim zadatcima istovremeno ili izvršavanje više funkcionalnosti paralelno što omogućuje maksimalnu iskorištenost resursa. Također, samim time smanjuje cijenu održavanja i vrijeme te poboljšava performanse kompleksnih aplikacija.

- Dinamičan i proširiv

Java je dinamična te proširiva jer uz pomoć objektno usmjerenih principa možemo dodati klase te metode tim klasama. Pruža podršku metodama i funkcijama napisanim u drugom programskom jeziku kao što su C i C++. Takve metode nazivaju se „urođene“ metode i dinamički su povezane prilikom izvođenja.

3.2. Programski okvir Spring

Spring je aplikacijski okvir i alat za inverziju kontrole programskog jezika Java. Temeljna svojstva sustava mogu se koristiti od strane bilo koje Java aplikacije ,a daje i ekstenzije za izradu web aplikacija služeći se Java Enterprise platformom. Iako striktno ne nalaže programski model, postao je popularan unutar zajednice kao dodatak Enterprise JavaBeans modelu. Neke od prednosti koje pruža su:

- Spring je posvuda

Biblioteke unutar Spring okvira koriste se na projektima posvuda na internetu. Spring dostavlja ugodno iskustvo krajnjim korisnicima na bilo kojoj platformi u raznim sektorima života – od usluga internetske kupovine do drugih inovativnih rješenja. Također, Spring ima suradnike u velikim tvrtkama diljem svijeta od kojih su neke Amazon, Google i Microsoft.

- Spring je fleksibilan

Spring-ov fleksibilan set ekstenzija te korisnički definiranih biblioteka dopušta izgradnju bilo kakve aplikacije. U svojoj jezgri, Spring-ov način inverzije kontrole (IOC) i ubrizgavanja ovisnosti (DI) pruža temelje za širok set funkcionalnosti. Bilo da se radi o izgradnji sigurne, reaktivne, usluge na oblaku ili kompleksne stranice za strujanje podataka, Spring ima alate koji to omogućuju.

- Spring je produktivan

Spring Boot mijenja način pristupa programiranju radikalno mjenjajući iskustvo rada. Spring Boot kombinira nužnosti kao što su aplikacijski kontekst te automatska konfiguracija i ugrađeni web poslužitelj kako bi se mikro usluge razvile u kratkom periodu vremena. Da razvoj bude još brži, Spring Boot se može koristiti sa Spring Cloud tehnologijom oblaka koja podržava veliki broj biblioteka, poslužitelja, uzoraka i predložaka kako bi se konačan proizvod postavio na oblak u kratkom vremenu.

- Spring je brz

Spring pruža prednosti kao što su brzo pokretanje i gašenje te optimizirano izvođenje. Spring projekti također podržavaju reaktivan model programiranja u svrhu još veće efikasnosti.

- Spring je siguran

Spring-ov razvojni tim radi sa profesionalcima koji u kratkom roku pouzdano otklanjaju prijavljene greške. Također, vanjske poveznice se promatraju kako propust ne bi nasta u njima, održavajući tako nivo sigurnosti na visokoj razini.

3.3. IntelliJ IDEA

IntelliJ IDEA je razvojno okruženje napisano u Java programskom jeziku. Koristi se za razvoj računalnog softvera te podržava Javu, Kotlin, Groovy te ostale Java arhiv bazirane programske jezike. Prednosti korištenja IntelliJ razvojnog okruženja su :

- Asistent u kodiranju

Razvojno okruženje pruža niz pogodnosti kao što su automatska nadopuna koda analiziranjem konteksta, refaktoriranje koda, opcije popravljanja inkonzistentnosti prijedlozima, usluge traženja grešaka u kodu postepenim izvođenjem te mnoge druge.

- Ugrađeni alati

Podržava vanjske alate kao što su Grunt, Bower, Gradle i SBT. Omogućuje alate za upravljanje inačicama programa kao što su Git, Mercurial, Perforce i SVN. Baze podataka kao što su Microsoft SQL Server, Oracle, Postgre SQL, SQLite i MySQL mogu se pristupiti izravno iz razvojnog okruženja u Ultimate izdanju istog pomoću ugrađenog alata DataGrip.

- Dodatci

IntelliJ-u moguće je dodati vanjske dodatke koji nadodavaju ili poboljšavaju funkcionalnost razvojnog okruženja. Dodatci se mogu preuzeti direktno iz njihovog repozitorija ili sa njihove web stranice.

4. Implementacija rješenja

4.1. Zahtjevi na programsko rješenje

Unutar ovog poglavlja završnog rada na osnovu već postojećih prethodno opisanih rješenja objašnjeno je što će programsko rješenje imati od funkcionalnosti. Zahtjevi će također biti potkrepljeni dijagramima slučajeva korištenja.

Dijagram slučaja korištenja daje način objašnjenja detalja sustava te način na koji korisnici rukuju sustavom. Uglavnom je predstavljen grafički definirajući interakcije među različitim dijelovima sustava. Dijagram slučaja korištenja će odrediti radnje unutar sustava te kako te radnje teku, ali ne će odrediti na koji način će one biti implementirane.

Metodologija slučaja korištenja primjenjuje se u analizama sustava kako bi se identificirali, objasnili i organizirali zahtjevi na sustav. U ovom kontekstu riječ sustav odnosi se na nešto što je u procesu razvoj, konkretno ovaj završni rad. Ovakav način prikaza definiran je u UML-u, standardnoj notaciji za modeliranje objekata i sustava. Generalno je smatran boljim načinom predstavljanja sustava od organigrama te dijagrama toka. Razlozi uporabe ovog načina notacije jesu:

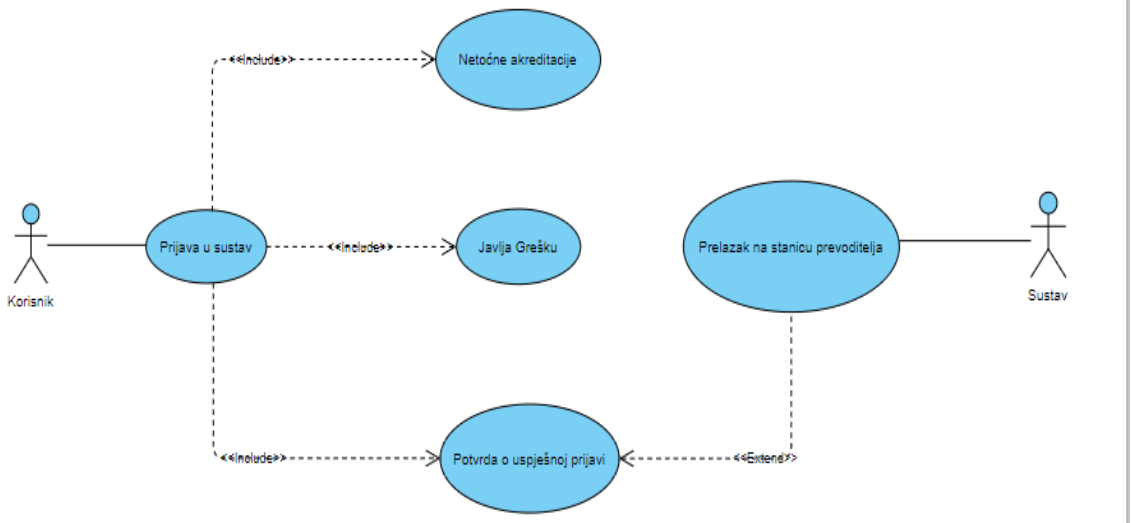
- Predstavljaju cilj sustava i korisnika
- Određuju kontekst u kojem treba rukovati sustavom
- Određuju zahtjeve na sustav
- Daju model za tijek ispravnog korištenja sustava
- Pružaju vanjski pogled na sustav
- Prikazuju vanjske i unutarnje utjecaje na sustav

Obično, UML dijagrami slučaja korištenja definiraju se prilikom razvoja ideje projekta te tijekom razvoja se referencija na njih. [1]

4.2. Prijava u sustav

Stranica za prijavu treba biti istaknuta unutar početnog korisničkog sučelja web aplikacije. Također, stranica za prijavu u sustav prva je stranica sa kojom se korisnik susreće prilikom pokretanja rada. Trebala bi se sastojati od dva tekstualna polja – jedno od njih za unos imena korisničkog računa ,a drugo za unos zaporke. Osim toga, pokretač radnje provjere unesenih

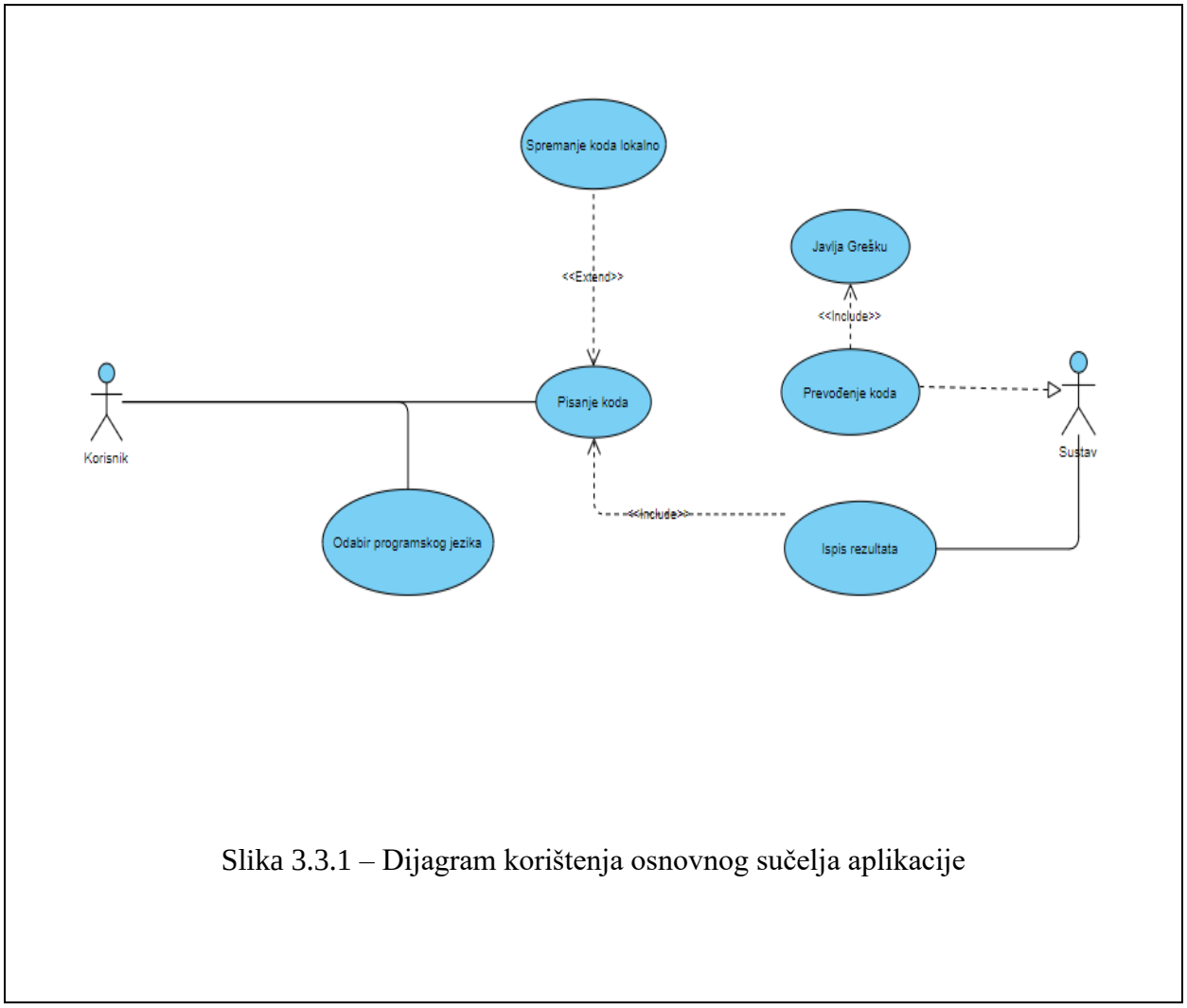
podataka mora biti dan u formi gumba. Ako je jedno od polja prazno, greška mora biti javljena korisniku, a ako se dano korisničko ime ne poklapa sa zaporkom vezanom uz to korisničko ime, sustav mora javiti grešku.



Slika 3.1.1 – Dijagram slučaja korištenja prijave

4.3. Korištenje web aplikacije

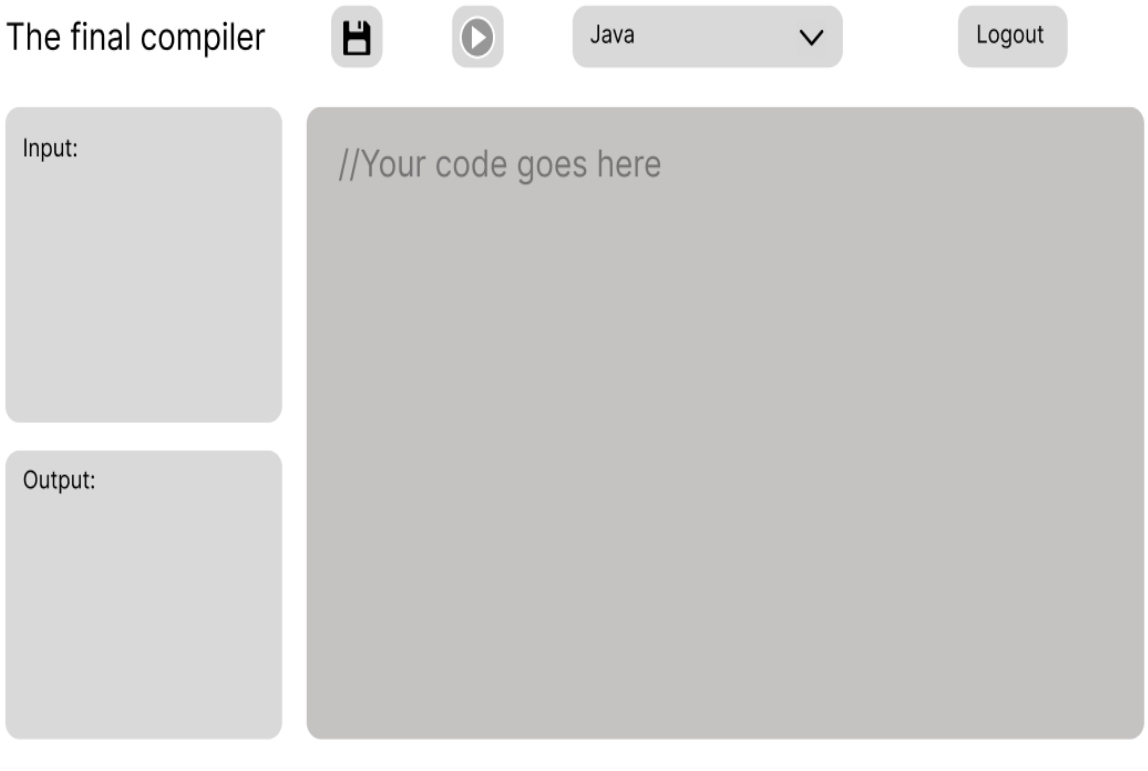
Korisniku, bez obzira na to prijavljen ili ne, mora biti moguće korištenje ostatka sučelja web aplikacije. Pod to ulazi, pisanje korisničkog koda, izvršavanje napisanog koda te njegovo lokalno spremanje. Na ovom zaslonu moraju se isticati gumbi koji su odgovorni za prevođenje koda te njegovo spremanje. Također, bitan je i padajući izbornik koji omogućuje postavljanje jezika koji se želi prevesti.



Slika 3.3.1 – Dijagram korištenja osnovnog sučelja aplikacije

4.4. Figma dizajn

Dizajn na Figmi predlaže izgled korisničkog sučelja te pomaže odrediti zahtjeve prije nego izrada aplikacije krene. Najčešće, dizajn ostvaren na Figmi dijeli se sa klijentima te izvođačima rada te se raspravljaju detalji izrade i implementacije projekta. Ovi modeli predstavljaju samo statičan sustav koji nalaže dizajn sustava, ali nema nikakvu funkcionalnost.



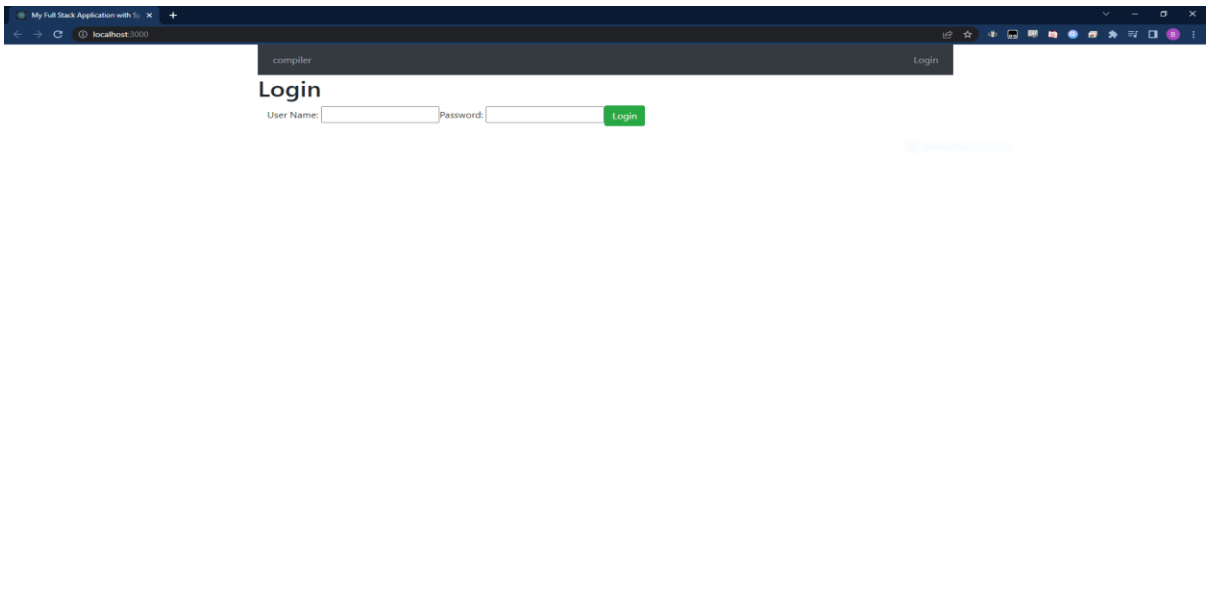
Slika 5.1.1 – Figma mockup – početni zaslon

4.5. Izrada aplikacije

Izrada aplikacije odvojena je u dva velika dijela: jedan projekt koji odrađuje sav posao dohvaćanja te slanja podataka na aplikacijsko programsko sučelje, čeka njihovu obradu te ih priprema za prikazivanje. Napisan je u java programskom jeziku koristeći Spring programsko sučelje. Drugi projekt pak se odnosi na izgled web aplikacije te sadrži logiku potrebnu za uobičajen rad web aplikacije. Napisan je u React programskom okviru.

Za potrebe funkcionalnosti web aplikacija se izvodi lokalno na portu 8080 pomoću Apache Tomcat programskog sučelja koje pruža HTTP poslužitelja na kojem se može izvršavati Java kod.

Prva stranica sa kojom se korisnik susreće je stranica za prijavu. Korisnik mora znati ispravno korisničko ime i zaporku kako bi mogao započeti sa radom sustava. Za potrebe rada zaporka i korisničko ime bit će postavljeni na „admin“. Ukoliko korisnik ne zna ispravnu zaporku ili korisničko ime rad aplikacije je onemogućen, a aplikacija javlja grešku.

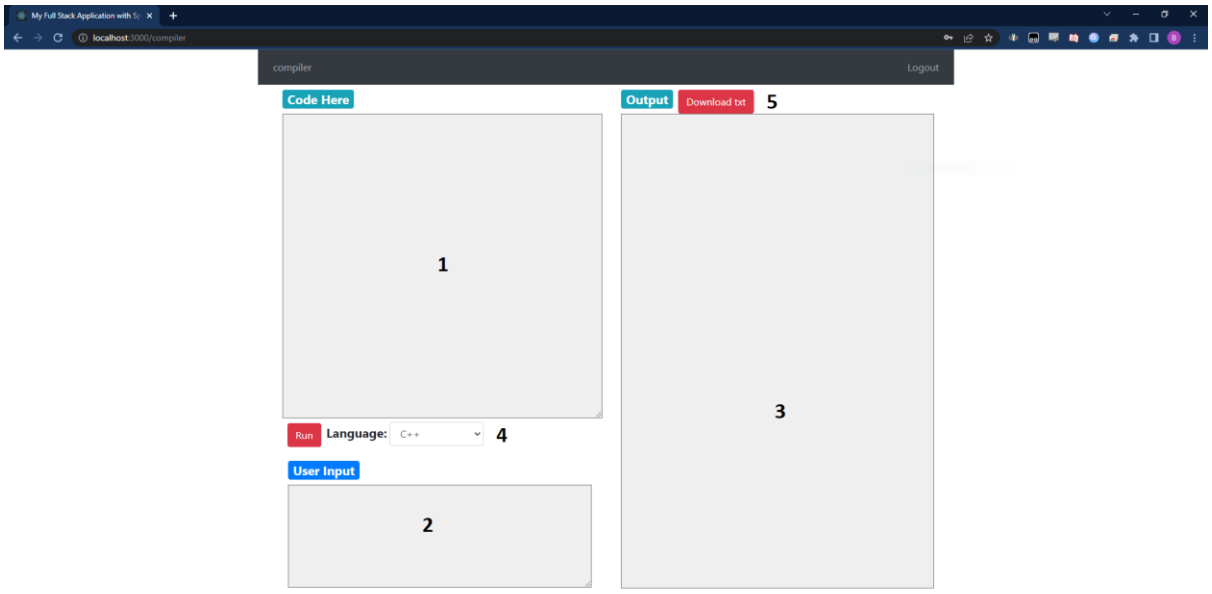


Slika 5.2.1 – Početni zaslon web aplikacije

Funkcionalnost rada osnovana je na Judge0 CE aplikacijskom programskom sučelju. Kod napisan u okviru web aplikacije zajedno sa korisničkim unosima potrebnim za funkcionalnost programa

šalju se na prethodno spomenuto aplikacijsko programsko sučelje. Sučelje za povrat daje odgovor koji je ujedino i rezultat izvršavanja programskog isječka. Četiri programska jezika su podržana unutar rada web aplikacije: Java, C++,C te Python.

Na slici 5.2.2. prikazan je osnovni zaslon web aplikacije. Oznaka 1 označava polje koje se koristi za unos isječka koda koji se testira. Oznaka 2 prikazuje polje koje služi za unos eventualnih ulaznih podataka koji su potrebni za ispravnu funkcionalnost programskog isječka. Padajući izbornik pod oznakom 3 omogućuje odabir prethodno spomenutih jezika. Rezultat programskog isječka nakon obrade bit će prikazan u polju sa oznakom 4. Ukoliko kod nije ispravno napisan, u istom polju bit će prikazana greška. Gumb pod oznakom 5 omogućava lokalno spremanje rezultata programskog isječka u tekstualnu datoteku.



Slika 5.2.2 – Sučelje glavne stranice projekta

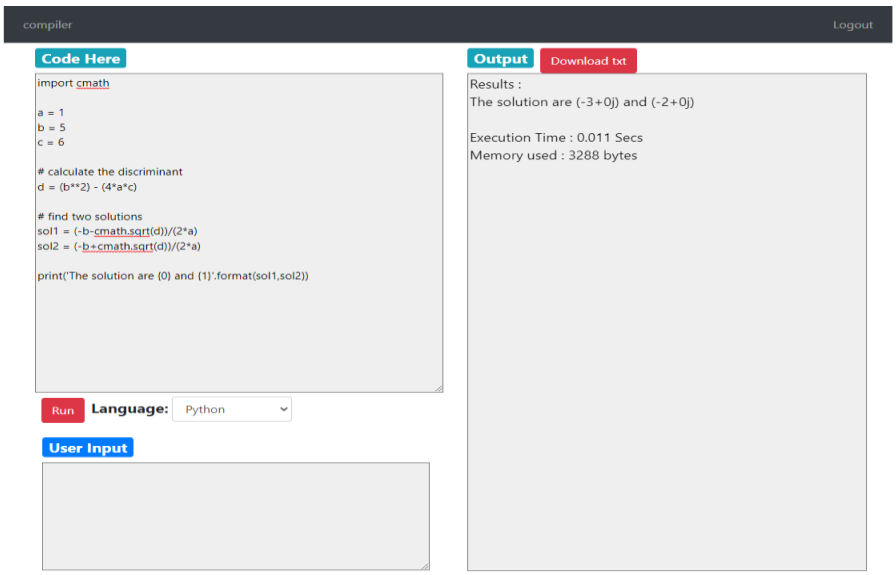
Pritisak na gumb „Run“ poziva izvršenje Post metode koja šalje sve podatke potrebne za prevođenje upisanog programskog koda u obliku JSON objekta. Zahtjev se obrađuje na poslužitelju te se odgovor potom prikazuje, ovisno o uspješnosti izvršavanja istog.

```
@RestController
@CrossOrigin(origins = {"http://localhost:3000", "http://localhost:3002",
"http://localhost:3003"})
public class OnlineCompilerController {

	private static final String API_KEY
="27c6bc68b6msh98173d4927e8500p1b3caejsn3719b5fd0d36";

	@PostMapping( value = "/submissions", consumes =
MediaType.APPLICATION_JSON_VALUE)
	public Object submitCode(@RequestBody HashMap<String, String> codebody)
throws IOException, InterruptedException {

		JSONObject jo = new JSONObject(codebody);

		HttpRequest request = HttpRequest.newBuilder()
				.uri(URI.create("https://judge0-
ce.p.rapidapi.com/submissions?fields=*"))
				.header("content-type", "application/json")
				.header("Content-Type", "application/json")
				.header("X-RapidAPI-Key", API_KEY)
				.header("X-RapidAPI-Host", "judge0-ce.p.rapidapi.com")
				.method("POST",
HttpRequest.BodyPublishers.ofString(jo.toString()))
				.build();
		HttpResponse<String> response = null;
		try {
			response = HttpClient.newHttpClient().send(request,
HttpResponse.BodyHandlers.ofString());
		} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
			return null;
		} catch (InterruptedException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
			return null;
		}
		return response.body();
	}

}
```

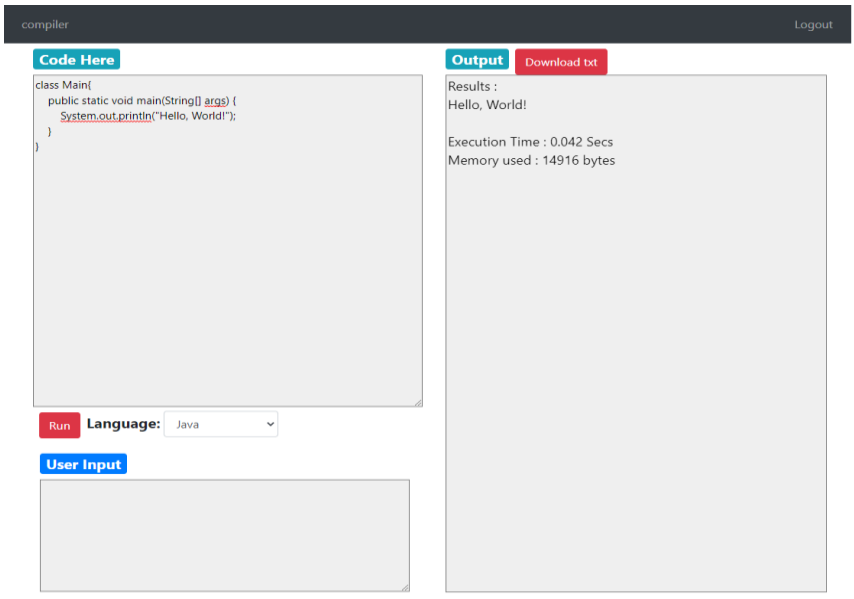
Programski kod 5.2.1 – POST zahtjev

4.6. Testiranje aplikacije

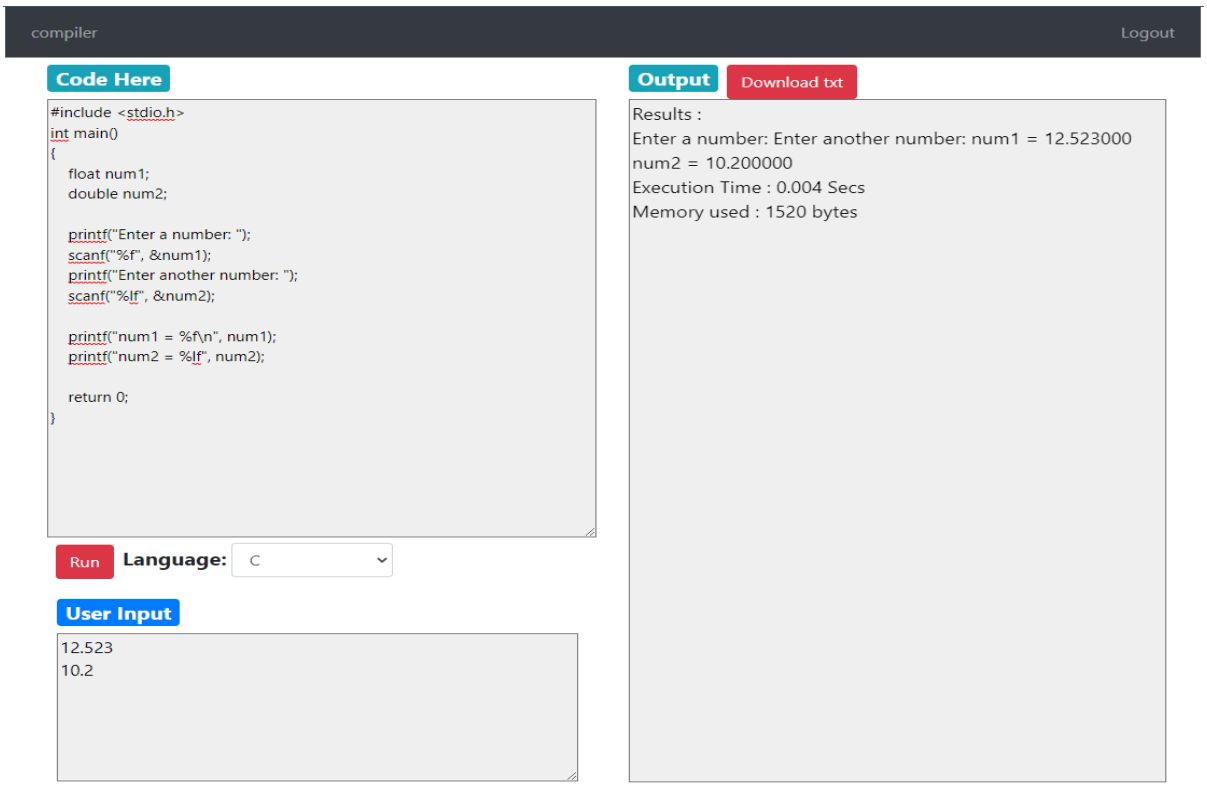
Testiranje aplikacije uključivalo je testiranje uspješnosti autentifikacije sustava te kontrolu točnosti prijenosa koda do i od poslužitelja. Točnost prevođenja koda nije uključena u odgovornosti ovog projekta jer se kod prevodi i izvršava na poslužitelju. Aplikacija je testirana za sva 4 podržana programska jezika sa različitim kombinacijama uključenih elemenata unosa. Testiranja su dana slikama 5.3.1. do 5.3.4. i bila su uspješna za sve jezike.



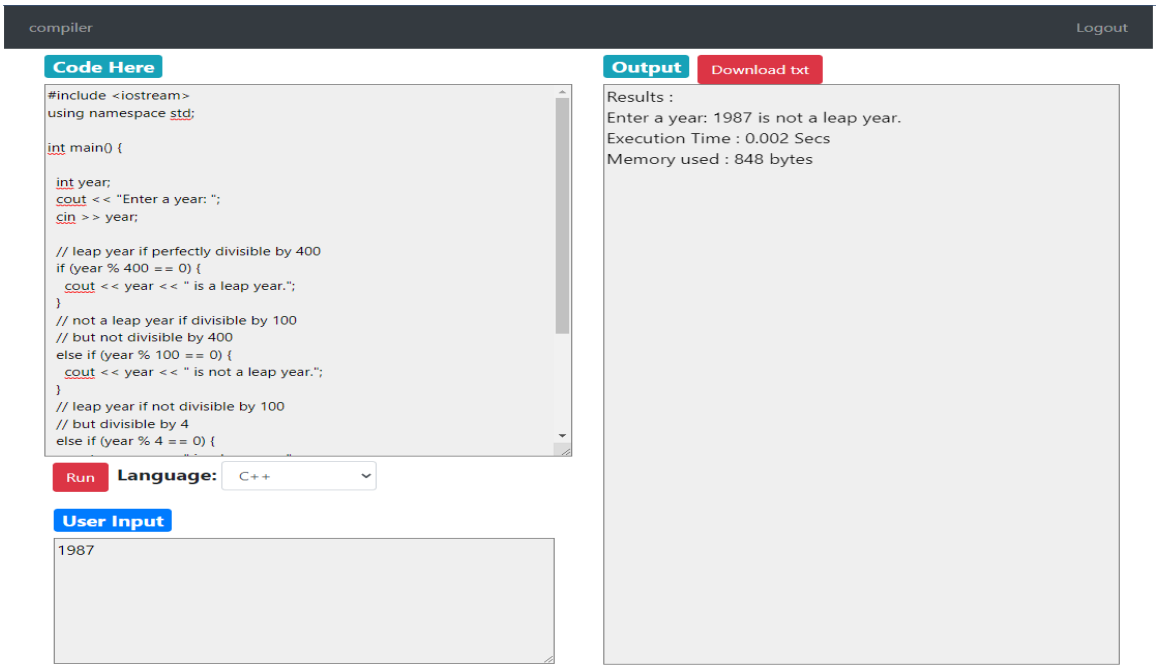
Slika 5.2.1. – Testiranje isječka u Pythonu



Slika 5.2.2 – Testiranje isječka u Javi

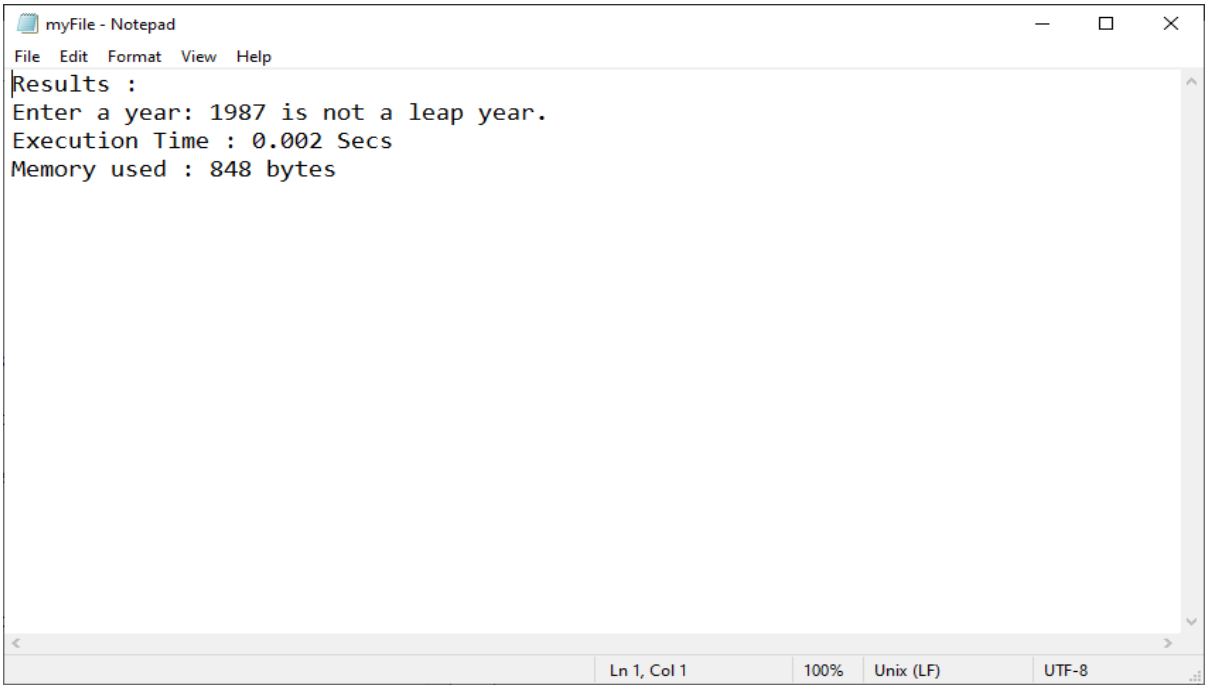


Slika 5.2.3. – Testiranje rada u C jeziku



Slika 5.2.4. – Testiranje rada u C++ jeziku

Funkcionalnost spremanja rezultata programskog isječka u tekstualnu datoteku također radi kao očekivano. Datoteka se sprema u mapu preuzimanja sa imenom „myFile“.



Slika 5.2.5 – Isječak koda spremljen lokalno u tekstualnu datoteku

```
downloadTxtFile = () => {

        const element = document.createElement("a");

        console.log(document.getElementById('output'));

              const file = new
      Blob([document.getElementById('output').value], {type: 'text/plain'});

        element.href = URL.createObjectURL(file);

        element.download = "myFile.txt";

        document.body.appendChild(element);

        element.click();

      }
```

Programski kod 5.2.6 – Lokalno spremanje rezultata programskog isječka

4.7. Usporedba sa postojećim rješenjima

Promatrano programsko rješenje nudi odabir između 4 jezika te mogućnost spremanja rezultata lokalno na tvrdu memoriju. Ograničenje poslužitelja odobrava samo 50 zahtjeva na dan, sve više od toga je izvan besplatne tarife i vlasnik poslužitelja naplaćuje. Podržavanje vanjskih datoteka određeno je poslužiteljem.

Najveća prednost koje ostala spomenuta programska rješenja pružaju nad rješenjem obrađenim u ovom radu je isticanje sintakse. Također, ne postoji broj zahtjeva koji predstavlja ograničenje. Nadalje, web aplikacija ne podržava mijenjanje vizualnog aspekta ni u kojem smislu, teme nisu podržane. Terminal također nije uključen o ovo programsko rješenje.

Područja poboljšanja ovog projekta uključuju dodavanje svih mogućnosti koje su prethodno spomenute kao prednosti alternativnih rješenja: dodavanje terminala, sposobnosti mijenjanja boja odnosno vizualnih postavka web aplikacije, povećanje broja dozvoljenih zahtjeva te poboljšanje sustava sigurnosti podataka.

5. Zaključak

Unutar ovog rada obrađena je tematika izrade web aplikacije za provjeru programskog koda te problematika prilikom izrade istog. Izrađena je aplikacija koja pruža sposobnost prevođenja i izvršavanja programskog koda napisanog u nekom od podržanih jezika. Također, aplikacija pruža osnovan sustav autentifikacije korisnika.

Pregledom već postojećih rješenja utvrđene su kvalitete i mogućnosti koje bi aplikacija trebala nuditi kako bi bila kompetentna, a neke od njih su: podržavanje više programskih jezika, osnovan sigurnosni sustav te mogućnost spremanja podataka, bilo to lokalno ili na poslužitelju.

Zahtjevi za osnovni rad su ispunjeni iako potpuni potencijal ovog rada nije dosegnut; aplikacija trenutno omogućava korištenje samo jednom korisniku ,a ne više kao što je prvobitno zamišljeno. Nadalje, iako postoji mogućnost korištenja četiri jezika, API kojim se aplikacija koristi pruža podršku više od 60 jezika pa je moguće dodatno proširiti ponudu jezika ovisno o potrebi korisnika.

Literatura

[1] https://www.techtarget.com/whatis/definition/use-case-diagram - use case diagram (UML use case diagram), srpanj 2020.

[2] https://techvidvan.com/tutorials/features-of-java-programming-language/ - Features of Java Programming Language that justifies its Popularity

[3] https://spring.io/why-spring - Why Spring?

[4] https://www.jetbrains.com/idea/

Sažetak

Razvoj dobrog i sigurnog razvojnog okruženja koje je uvijek dostupno ključno je za povećavanje produktivnosti. U ovom radu izrađena je web aplikacija koja omogućuje provjeru i izvođenje programskog koda. Neki od zahtjeva koji su implementirani u ovom rješenju su: osnovna sigurnost u obliku prijave u sustav, slanje i prikaz obrađenih rezultata te spremanje isječaka koda lokalno. Programsko rješenje obuhvaća Java Spring, React te JavaScript tehnologije koje ostvaruju pravilan rad projekta. Testiranja rješenje su prošla pozitivno jer rješenje ispunjava sve uvijete za pravilan rad.

Ključne riječi: izvođenje programskog koda, mrežna aplikacija, prevoditelj

Abstract

CODE RUNNER WEB APPLICATION

Development of a reliable and secure developing environment has become a key factor in productivity increase. In this paper, a web app was developed that allows compilation of code snippets. Some of the requirements implemented in this solution are: basic security in the form of a login form, sending and displaying of processed results and saving the code snippet locally. Technologies used in the development of the are Java Spring, React and JavaScript make the project work properly. The testing of the solution were successful because it meets every requirement for precise work.

Keywords: code execution, compiler, web app

Životopis

Bruno Miletić rođen je 24.01.2001. godine u Slavonskom Brodu. Završava Osnovnu školu Bogoslav Šulek u Slavonskom Brodu 2015.godine te upisuje Tehničku školu Slavonski Brod, smjer tehničar za elektroniku. Nakon završetka srednje škole i položene mature upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku, preddiplomski sveučilišni studij računarstva.

_____

Potpis autora