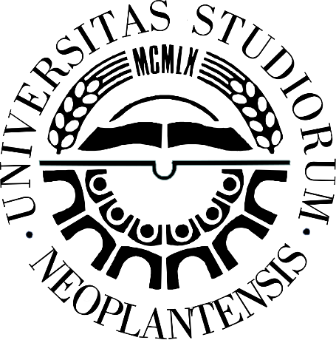
Univerzitet u Novom Sadu   
 Tehnički fakultet „Mihajlo Pupin“   
 Zrenjanin



SEMINARSKI RAD

Predmet: Programski prevodioci

**Upoređivanje gramatike C# i Jave" - SEMANTIKA DOMENA: Evidencija fudbalskih igrača - tip 5/A**

Predmetni nastavnik: Autor rada

doc. dr Ljubica Kazi Aleksa Cakić SI 23/17

Zrenjanin 2020.

1. Uvod

Microsoft je od 2014 godine menjajući kompajler u C# programski jezik; znači danas C# programski jezik kompajlira kompajler za C# programski jezik; omogućio programerima da kompajler kodnog imena „Roslyn“ kao kompajler nije više zatvorena crna kutija, već je open-source kod. Ali je isto tako time omogućio da se prave još bolje promene na samom programskom jeziku, pa tako danas imate male promene u kodu koje prave velike promene i umanjuju pisanje koda ali će ih biti sve više. Već sad možete izguglati neke promene koje Microsoft već najavljuje pored mnogobrojnih promena koje su ove godine uvedene.

Za razliku od C# programskog jezika, C++ programski jezik se smatra nešto bržim što danas i nije toliko relevantno s obzirom da su kompjuteri danas mnogo brži, osim možda u nekim segmentima kad su u pitanju programiranje igrica, ali ništa toliko zastrašujuće što se ne može nadomestiti. Sa C++ programskim jezikom možete da programirate i na Linux operativnom sistemu i na OS X platformi jer C++ programski jezik komunicira direktno sa operativnim sistemom i kompajlira odmah mašinski kod, dok je C# zavisan od Microsoft .Net Framework-a koji je posrednik između vaše aplikacije i operativnog sistema. Međutim vi u C++ ne možete da programirate na primer WPF – Windows Presentation Foundation aplikacije dok je danas programirati Windows Forms zastarela tehnologija i pored toga što se mora znati radi starijih projekata. Sa C# programskim jezikom možete definitivno da uradite više i da kodirate mnogo jednostavnije i lakše nego sa C++ programskim jezikom i to uvek imajte na umu.

## Malo o C#

Sada već davne 1995. godine je kompanija Sun Microsystems lansirala programski jezik Java, a koji se definiše kao objektno - orjentisani. Kasnije je kompanija Oracle otkupila pomenutu, te tako i postala zvaničan vlasnik ove tehnologije.

Princip na kome je ovaj progrmaski jezik kreiran je primarno usmeren na jezik Oberon, ali i na mnoge druge, dok mu je sintaksa nalik C i C++ programskim jezicima. Ipak, ovaj tip programskog jezika ima značajno strožiji princip prevođenja, a potpuno je nezavisan od platforme na kojoj se radi. Posebna pažnja prilikom njegovog kreiranja je usmerena na upravljanje memorijom, a koja je kod Java programskog jezika u velikoj meri jednostavnija. Smatra se da je razlog za to vezan za veliku popularnost C programskog jezika.

Svrha primene ovog jezika je na prvom mestu vezana za davanje konkretnih instrukcija određenom uređaju, a kako bi on bio u mogućnosti da izvrši određene komande.

Upravo se Java programski jezik smatra jednim od najpopularnijih, a veruje se da je osnovni razlog za to vezan za činjenicu da se njegovo korišćenje uopšte ne naplaćuje. Budući da se korišćenje mnogih drugih programskih jezika gotovo uvek naplaćuje, sasvim je jasno i zbog čega upravo Java uživa toliku popularnost.

Druga prednost primene ovog programskog jezika se odnosi na činjenicu da se komande koje navodi Java, praktično rečeno mogu izvršavati na gotovo bilo kom računaru, jer je ona kreirana tako da uopšte ne zavisi od platforme koja se koristi na konkretnom tipu uređaja.

Svakako je prednost i ta što se baš Java koristi kada je potrebno pristupiti razvoju različitih vrsta aplikacija. Uz primenu tog programskog jezika se mogu praviti takozvane GUI aplikacije (Graphical User Interface), kao i apleti aplikacije (Applets), ali isto tako i one koje su poznate kao konzolne. Kada se karakteristike ovog programskog jezika uporede sa drugima, stiče se jasan utisak da je Java izuzetno jednostavna za korišćenje, pa se i to svakako može svrstati u jednu od mnogobrojnih prednosti koje se na njenu primenu odnose.

Zanimljivo je navesti i podatak da naziv ovog programskog jezika vodi poreklo upravo od istoimenog ostrva, a koje se nalazi na teritoriji Indonezije. Iako je Java primarno programski jezik, činjenica je da se ovim pojmom označava isto tako i sotverska platforma, koja je zadužena da pokreće aplikacije koje su u ovom programskom jeziku izrađene.

Neretko se događa da se pojam Java i JavaScript mešaju, ali za to ne postoji razlog, sobzirom na to da su u pitanju dva potpuno drugačija programska jezika, čije karakteristike i funkcije se svakako razlikuju. Činjenica je da ovaj programski jezik ima brojne prednosti, ali naravno da su prisutne i određene mane, odnosno nedostaci kada je u pitanju njegova primena. Najznačajniji nedostatak se odnosi na brzinu njenog rada, uzevši u obzir da se ova tehnologija relativno sporo pokreće, a u poređenju sa drugima koje su joj na neki način slične.

## Malo o MSSQLU

Bez obzira koji programski jezik učite ili sa kojim programskim jezikom programirate iz hobija ili poslovno; od svakog programera se očekuje da poznaje rad sa bazama podataka kao i osnove strukturiranja relacionih baza podataka. Baza podataka vam je najjednostavnije rečeno kolekcija podataka smeštena u elektronskom formatu. Na engleskom jeziku se baza podataka kaže database ili skraćeno db. Inače prema Wikipedia-iji; baza podataka je organizovana kolekcija podataka za brzo pretraživanje i pristup; koja zajedno sa sistemom za održavanje i administraciju, organizovanje i memorisanje tih podataka čine sistem baze podataka. U školama će vam reći da je baza podataka kolekcija podataka koja se zapisuje u SQL server. Sve su ove definicije tačne ali je vama najvažnije da shvatite da je dobro organizovana baza podataka rešenje pola vašeg programerskog posla. Sve poznate baze podataka poput SQL, MySQL ili Oracle Database baza podataka; koriste isti standard za rad sa podacima i sve podržavaju SQL programski jezik. Kod Microsoft-a je to T-SQL i on je važan aspekat bez obzira na pojavu upitnog integrisanog jezika LINQ-a koji naveliko danas menja način programiranja ali definitivno nije zamena. T-SQL je jezik baza podataka i ne možete ga preskočiti već ga morate znati. Ali zato kad jednom savladate rad sa bazama podataka, lako će te moći koristiti sve vrste baza podataka jer funkcionišu na istim standardnim principima. Ne morate vi biti stručnjak za baze podataka ali neke osnovne stvari morate poznavati kako bi ste uopšte mogli da koristite baze podataka u vašim programima, sajtovima, servisima ili sistemima. U školama računara se uglavnom rad sa bazama podatka ne svrstava u C# programski jezik, ali se uči uporedo. Tako ću vam i ja uporedo sa postovima C# programskog jezika pisati i postove za rad sa Microsoft SQL Server 2016 Express serverom.

( Microsoft SQL Server 2016 )

Pre su programeri uglavnom koristili tekstualne ili binarne datoteke da skladište informacije koje koristi njihov program. Međutim na taj način podaci su se samo gomilali, ponavljali; zauzimali su dosta memorije; mnoge kolone su ostajale prazne, svaka datoteka je u suštini predstavljala samo jednu tabelu i zato se javila velika potreba za programima i serverima koji su pored skladištenja podataka mogle da uvedu pre svega organizaciju u podacima ali i mnoštvo tehnoloških robusnih novina koje danas serveri za baze podataka nude. Prva baza podataka sa kojom ste se možda susretali kroz Microsoft Office jeste Microsoft Access baza podataka. I ona takođe obuhvata tabele, poglede, uskladištene procedure, funkcije i druge objekte koje su neophodni za pravljenje sistema podataka ali navedena baza podataka je desktop aplikacija i može biti veoma problematična kad su u pitanju mnoštvo objekata, tabela, relacija i podataka i pristupanjem istih preko mreže. Zato je pametnije odmah koristiti neku od edicija Microsoft SQL Server baza podataka. Već više od deceniju postoje razne verzije i edicije Microsoft SQL Server-a , često se koriste starije verzije i one se stalno menjaju poput:

SQL Server Express Edition

SQL Server Workgroup Edition

SQL Server Developer Edition

SQL Server Standard Edition

SQL Server Enterprice Edition

SQL Server Mobile Edition

Naravno većina edicija se razlikuje pored tehničkih alata, ograničenja i mogućnosti; tako isto i po godinama izdanja. Microsoft SQL Server 2005 Express Edition i Microsoft SQL Server 2016 Express Edition se takođe razlikuju iako se te razlike često ne vide. Microsoft SQL Server 2016 Express je odličan izbor pre svega za početnike i zato što je besplatan dok su vam za druge edicije potrebne licence i mogu vam reći da one nisu ni malo jeftine. Zato moj izbor za vas je definitivno Microsoft SQL Server 2016 Express. Dobra vest je da Microsoft SQL Server 2016 Express pored 1 GB RAM-a, jednog procesora, nudi ograničenje skladištenja podataka na 10 GB. U ranijim verzijama ograničenje je bilo 4 GB. Sada dolazi sa Advanced Services i nećete imati komplikacije sa pravljenjem dijagrama.

Malo o Postgresu  
PostgreSQL ili jednostavnije Postgres je vrsta objektno-orijentisanih relacionih sistema za upravljanje bazama podataka (SUBP), pod open source licencom (otvoreni kod). Smatra se jednom od najpouzdanijih baza podataka. Najčešće koristi za web aplikacije i web baze podataka. Reklo bi se da je najveći konkurent MySQL-u.

PostgreSQL je bazično razvijen za rad na UNIX platformama, ali je portovan i na Linux, Windsows, macOS.

Prva zvanična verzija je objavljena 29. januara 1997. godine.

Inicijalno je razvijen u programskom jeziku C, ali takođe ima podršku za integraciju sa programskim jezicima kao što su Python, Perl, .NET, C++, Java, PHP, Ruby on Rails i drugi.

Postgres je naslednik Ingres-a koji je bio jedan od sistema baza podataka, razvijen između 1977. i 1985. godine. Zvanično, autor PostgreSQL-a je Michael Stonebraker, profesor na Kalifornijskom univerzitetu u Berkliju (UCB). Stonebrakerova ideja je bila da izgradi napredniju verziju Ingresa koja je robusnija, uz bolje performanse. Stonebraker i njegove kolege u UCB-u su osam godina (1986-1994) razvijali sistem Postgres baze podataka. Stonebrakerove kolege Andrev Iu i Jolli Chen su dodatno poboljšali razvijeni sistem zamjenom POSTQUEL upitnog jezika (query language) sa popularnijim i najčešće korišćenim SQL.

Ta poboljšana verzija nazvana je Postgres95. Nakon toga, 1996. godine, Postgres95 je po prvi put ušao u softversku industriju i postao jedan od najrobusnijih i najčešće korišćenih servera otvorenog programskog koda (open source).

Osobine i prednosti PostgreSQL-a

Open Source SUBP (DBMS – Database Management System) – Prva glavna prednost korišćenja Postgresa je to što je open source i može se prilagoditi prema zahtevima developera. Ova mogućnost prilagođavanja je izuzetno korisna u razvoju velikih aplikacija.

Velika razvojna zajednica (iliti velik community) – Postgres je na tržištu već više od 15 godina i njegova zajednica je u ovom trenutku izuzetno velika, što samim tim znači dobru podršku i pomoć pri rešavanju problema vezanih za rad i uporebu.

Isplativost – Postgres je izuzetno ekonomičan i ne zahteva mnogo obučavanja korisnika kako bi se naučilo kako koristiti i programirati za ovu bazu podataka. Takođe, zahtevi za održavanje i podešavanje Postgres baze podataka su relativno mali u odnosu na druge sisteme za upravljanje bazama podataka.

Portabilnost – Dobra stvar u Postgresu je to što je portabilan i prenosiv sa gotovo svim glavnim platformama i programskim jezicima. Ova baza podataka je idealna za aplikacije namenjene višestrukim platformama.

Alatke za razvoj i GUI – Server Postgres baze podataka ne zahteva obimne konfiguracije komandne linije. Razvijen je nekoliko alata i GUI interfejsa koji vam mogu pomoći u jednostavnoj instalaciji i upravljanju serverom baze podataka.

Pouzdanost i stabilnost – Postgres je svetski priznat kao najsigurnija i stabilnija baza podataka. Šanse da se uništi baza podataka su minimalne i čak i ako se baza podataka sruši, postoje načini i funkcije koje vam omogućavaju da obnovite i vratite podatke.

Nedostaci PostgreSQL-a

Performanse – za jednostavne operacije čitanja i brisanja, PostgreSQL postiže lošije rezultate u odnosu na, na primer, MySQL.

Popularnost – definitivno da nije toliko zastupljen, u odnosu na druge sisteme baza podataka i samim tim nema tako jaku podršku.

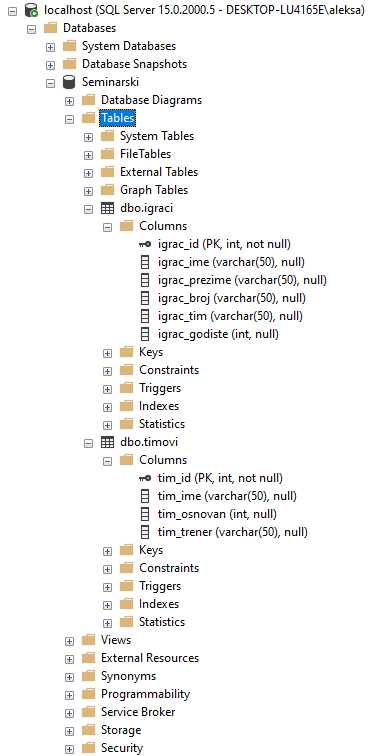
Hosting – nije toliko zastupljen kod provajdera i pružaoca hosting usluga.

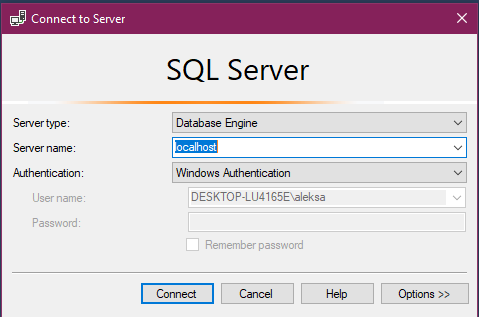
Instalacija I pripremanje projekta

Prvi primer projekta je u C#.   
Izrada projekta je vrlo laka I brza, I ne treba biti uopšte upoznat sa tehnologijom kako bi se uradio kompletan projekat, čime se implicira lakoća. Ovo je možda I najbolja tehnologija za apsolutno početnike da se upoznaju sa Web servisima, ali definitivno ne za dalji razvoj.

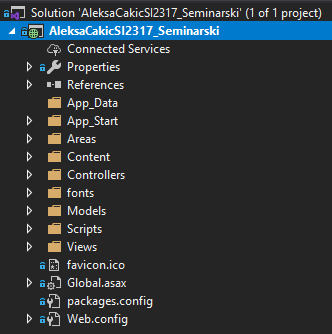
Prateći slike, projekat se dalje razvijao:

## Uputstvo

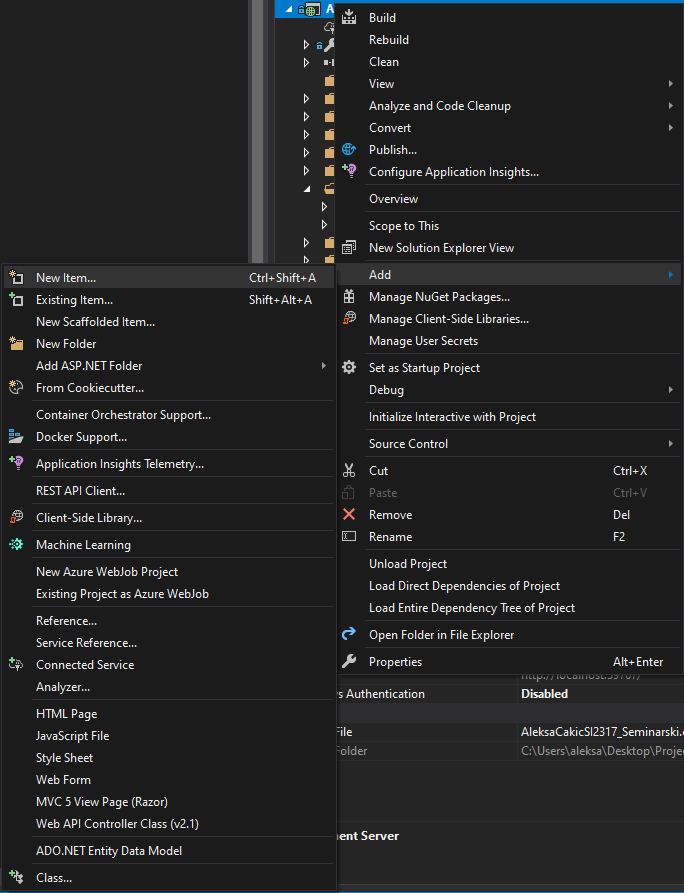




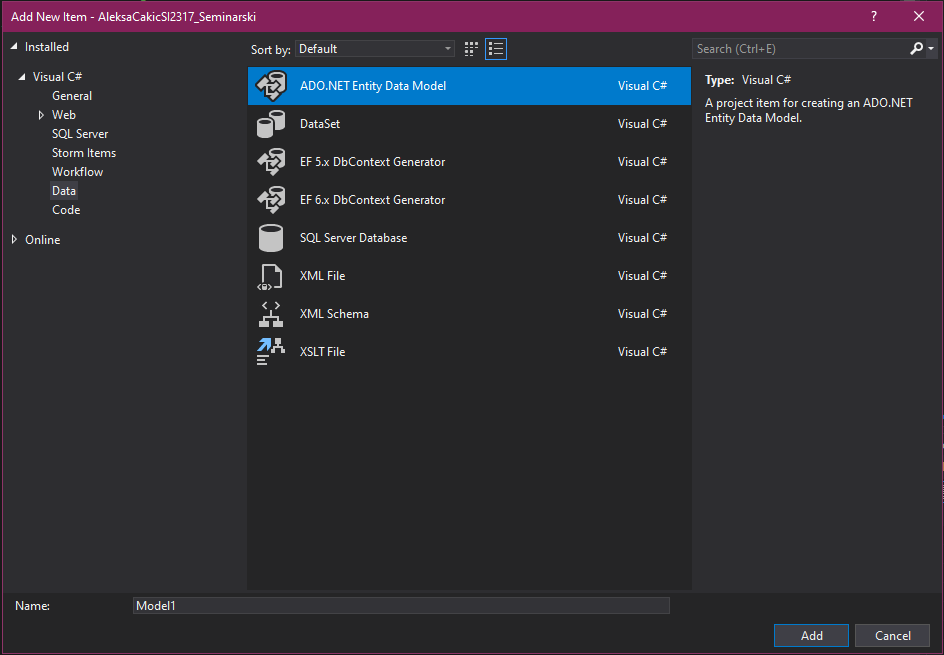
Povezivanje na server je prvo.

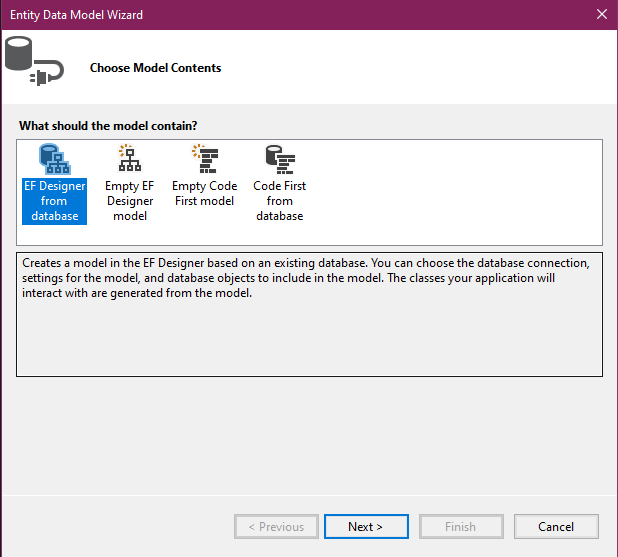


Struktura C# projekta kada se napravi.

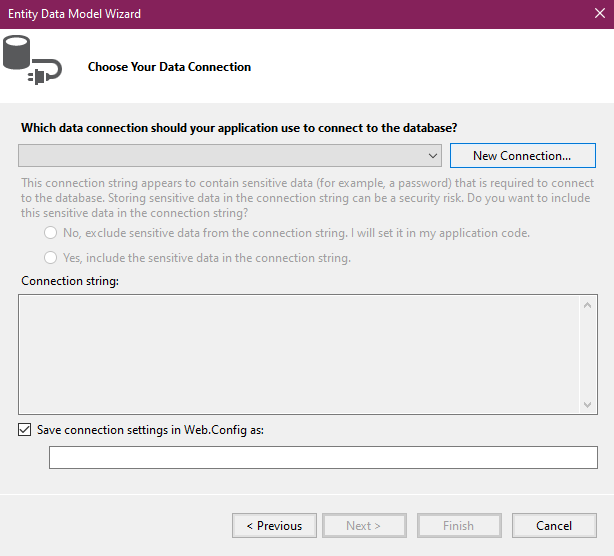


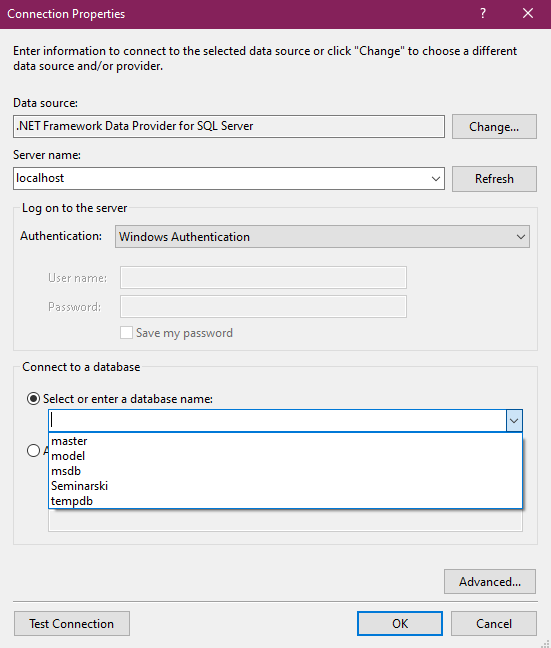
Kada u MSSQL serveru napravite tabele, tako što će se desnim klikom na šemu I odabirom da želite da napravite tabelu, u VS-u će te dodati novi item, koji će biti tipa data entity kao na slici ispod.





Posle toga će vas dočekati sledeći prozori, gde I da ne znate engleski, možete se snaći vrlo lako.

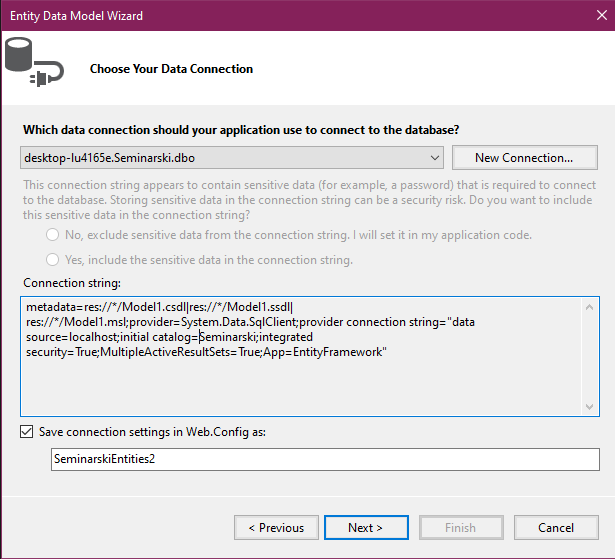




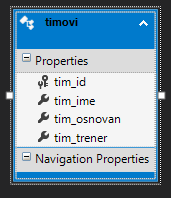
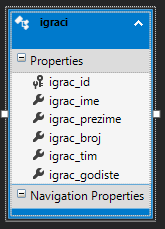
Posle nekoliko “Next” opcija, dočekaće Vas odabir baze podataka, sada je baza seminarski.



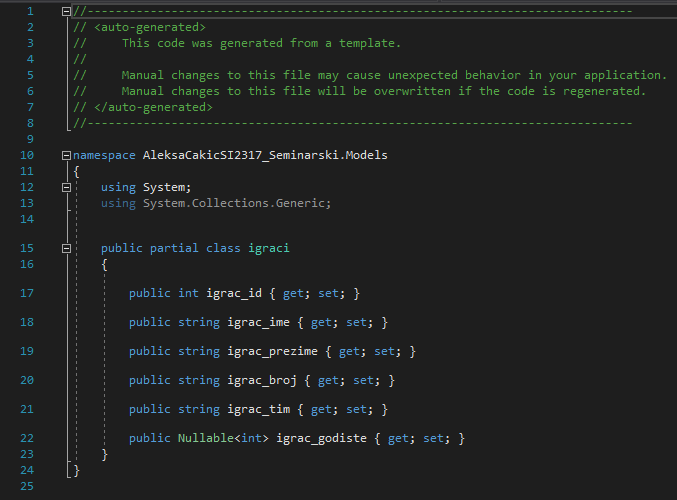
Odaberite jednu ili obe tabele, rezultat će biti isti.

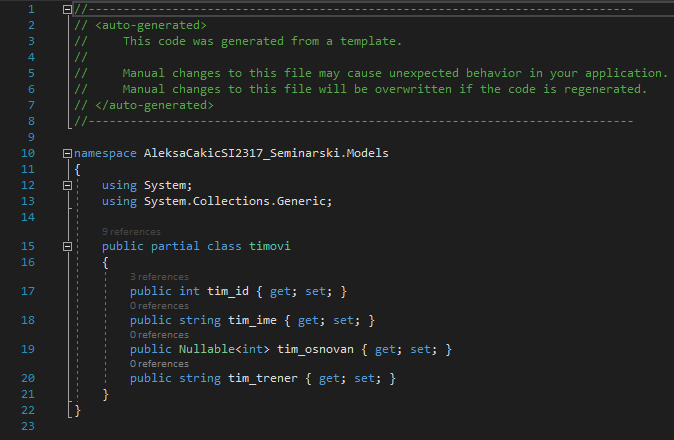


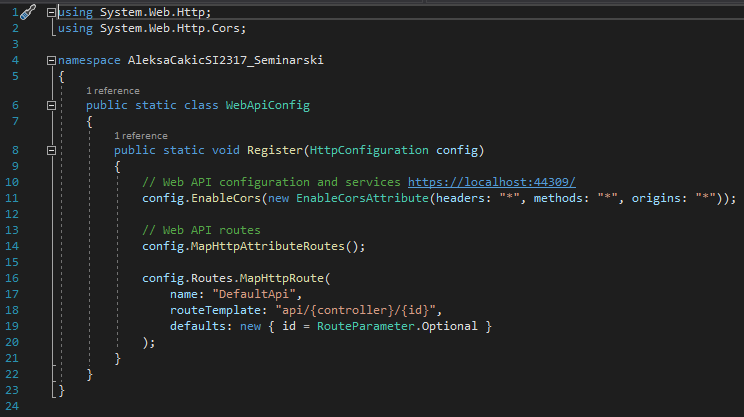
I samo pritiskati “Next” I čekati da system uradi svoj deo posla – kreiranje celog bekend projekta.

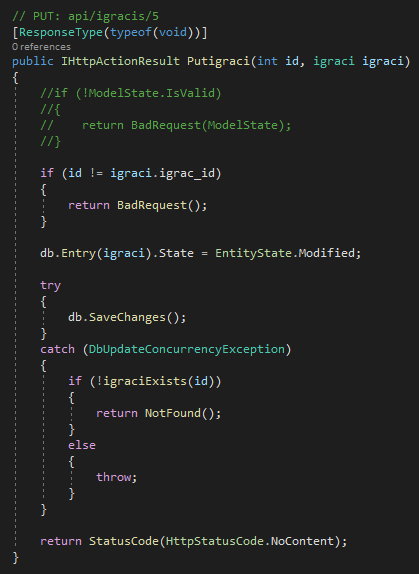
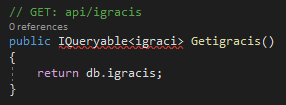
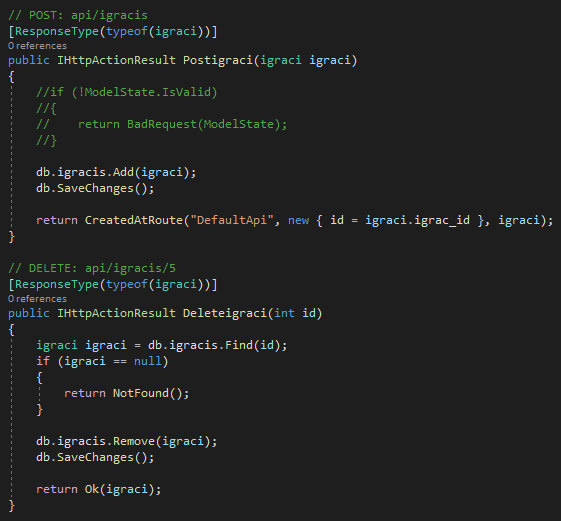


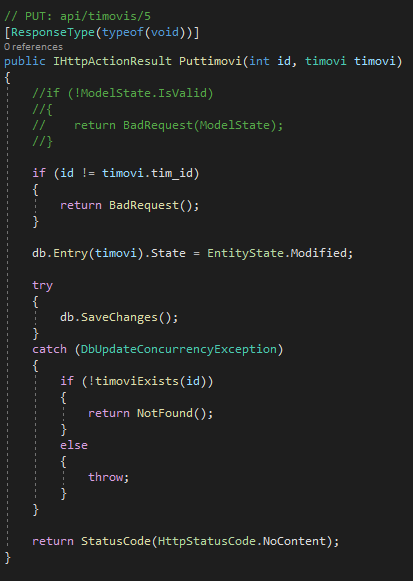
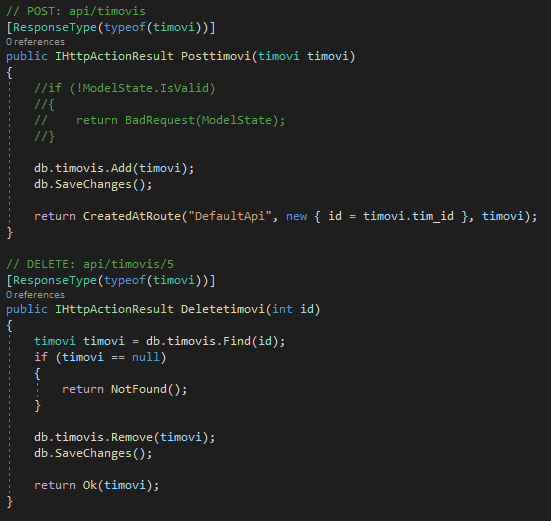
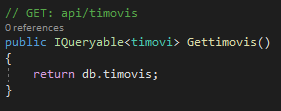
Na kraju će te imati modele kaon a slici iznad.

  
U folderu Models, moći ćete da vidite C# rendiciju tog istog modela koji je automatski generisan.



  
U webapiconfig hoćemo da podesimo CORS headere I port na kome će raditi aplikacija.

  
Nakon toga, možemo poglati koji se endpointovi nalaze u Kontroleru, kako bismo mogli da im pristupimo preko Frontenda.  
  




## JAVA SETUP

Za razliku od prostog načina autokonfiguracije, Java u stvari zahteva znanje I umeće. Uzimajući u obzir da je Java možda I najbolji način da se nauči nešto, autor ovog seminarskog rada je na najprostiji I bezbolniji način hteo da prikaže koliko je potrebno da se zna kako bi mogla da se uradi full stack aplikacija.

Iako je autor mogao prostim principom, linijom manjeg otpora, kao u C# projektu, da završi sve, ipak se odlučio da pokaže lepotu Java jezika, Docker virtuelizacije I build toolova.

Prvo ćete pokrenuti docker, sačekati da se podigne I krene sa radom. I želite da budete sigurni da docker radi sa Linux kontejnerima, inače neće raditi kako treba, ili uopšte neće raditi, baš kao Windows.

Kada uđete u java projekat, sačekajte da se indeksiraju fajlovi ako koristite Intellij.

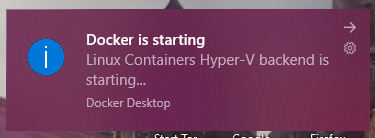
Zatim u build.gradle zelite pokrenete instalaciju I skidanje svih dependecija.

Kada se sve to završi, pokrenuti App main metodu za podizanje Spring aplikacije.

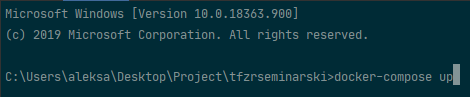
Onda ući u front end folder, gde sus vi Angular fajlovi I u terminal pokrenuti sledeće:  
- npm install  
- npm serve -o

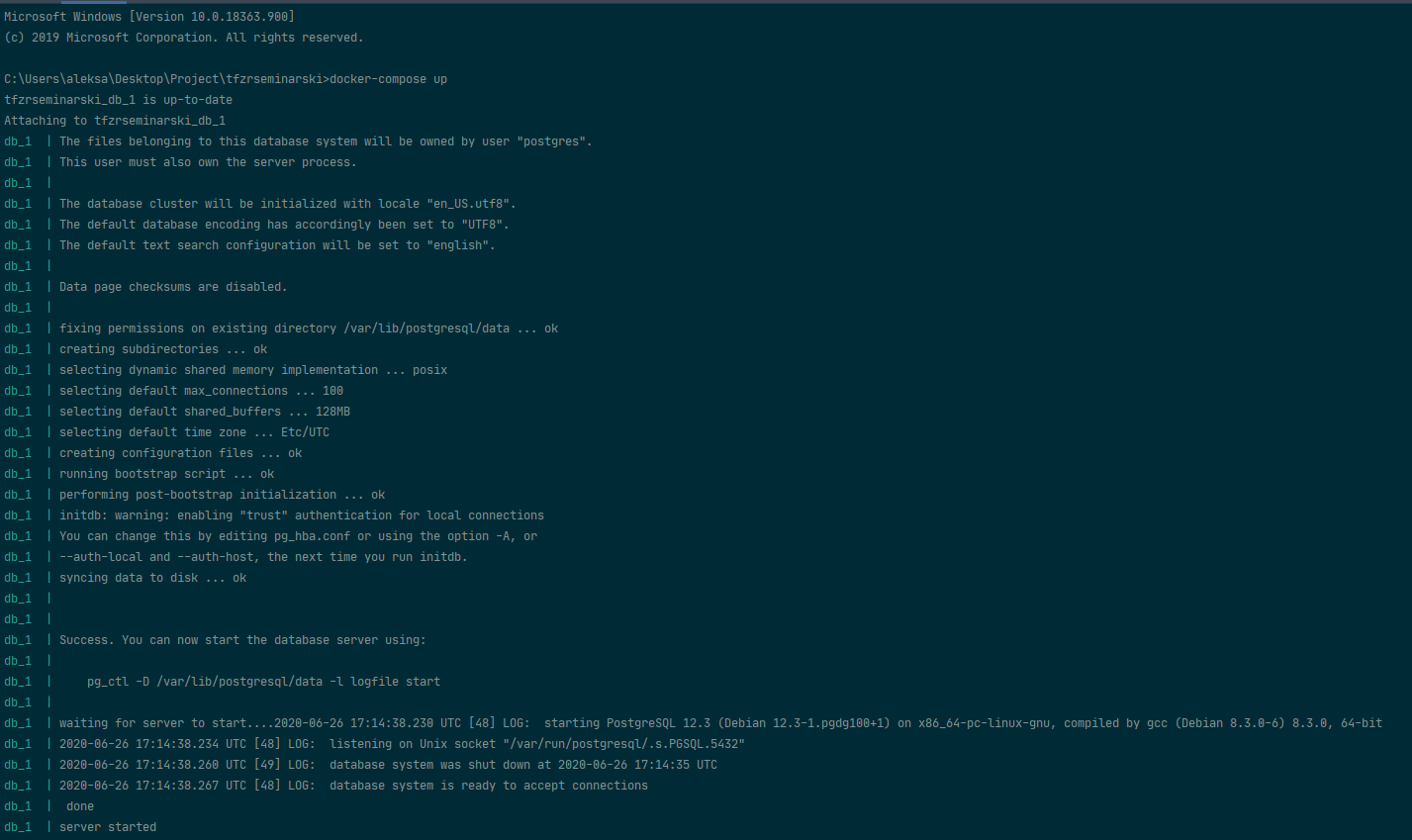
Pre poslednjeg koraka, ustanoviti da imate nodejs I Angular installirano.

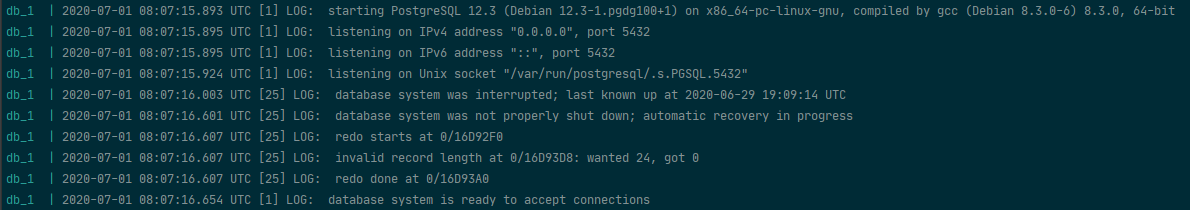


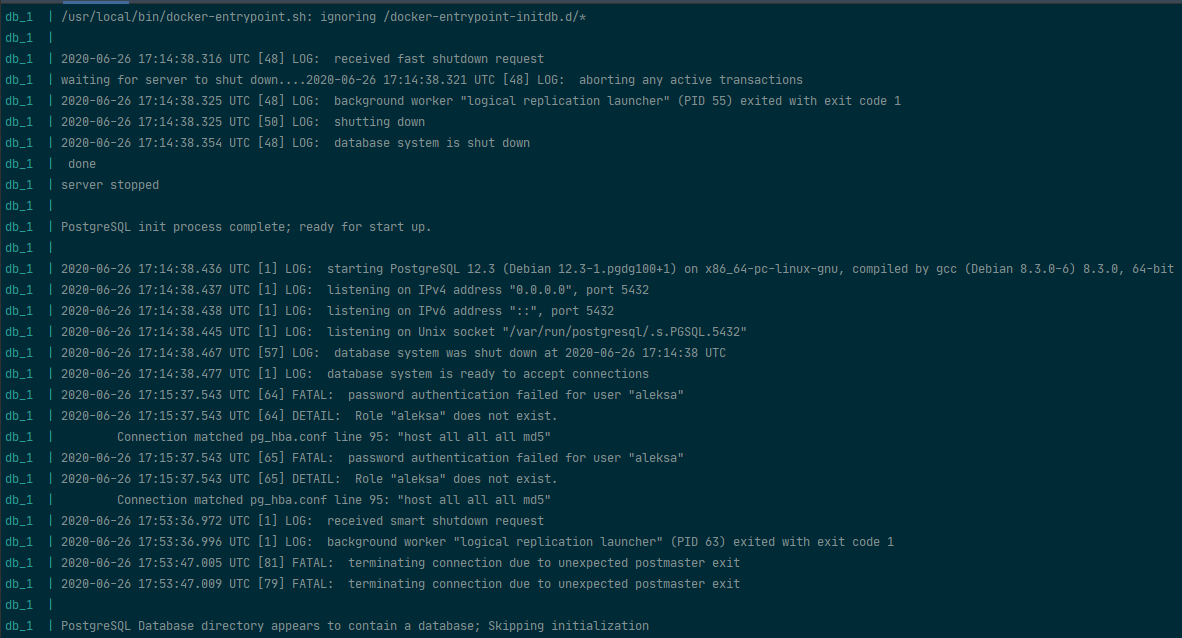


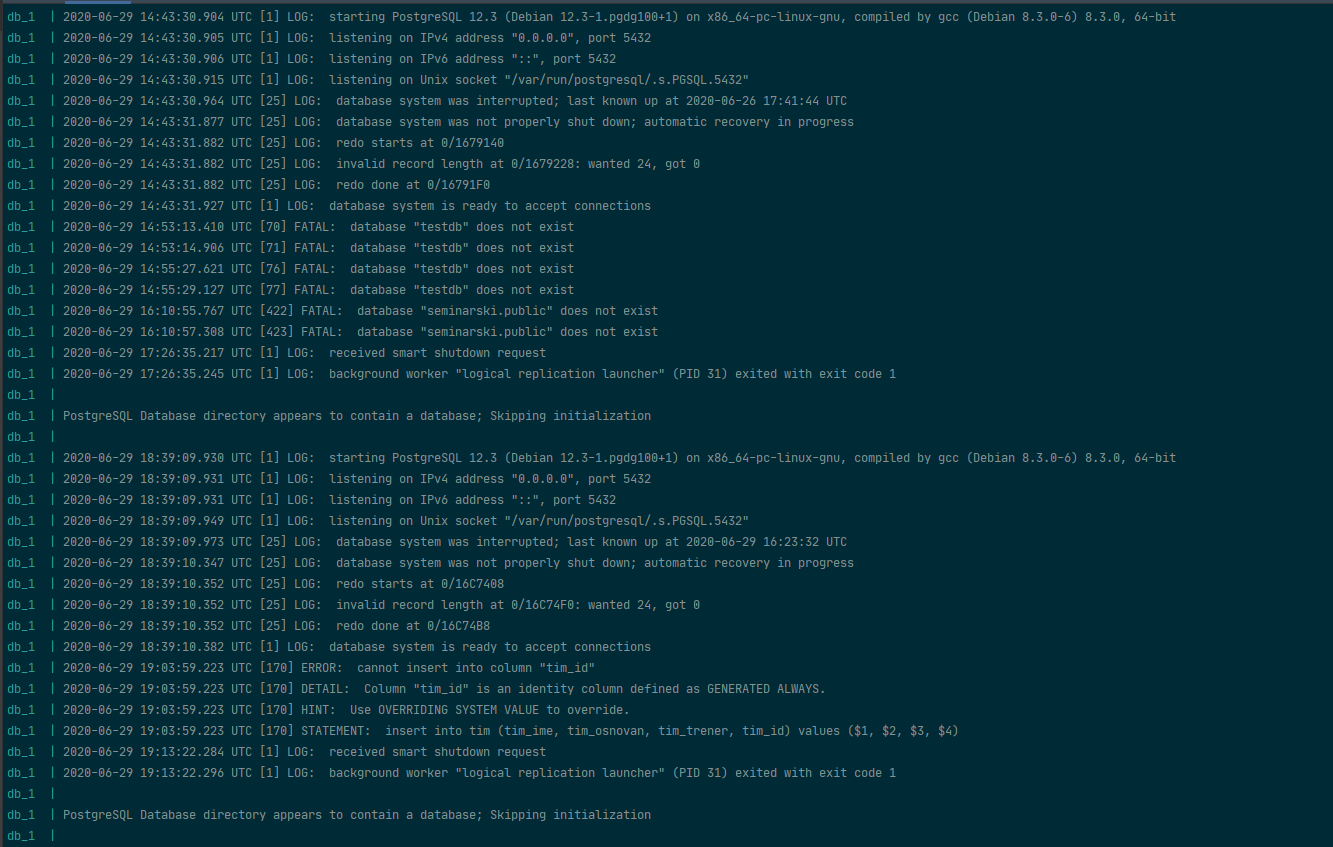


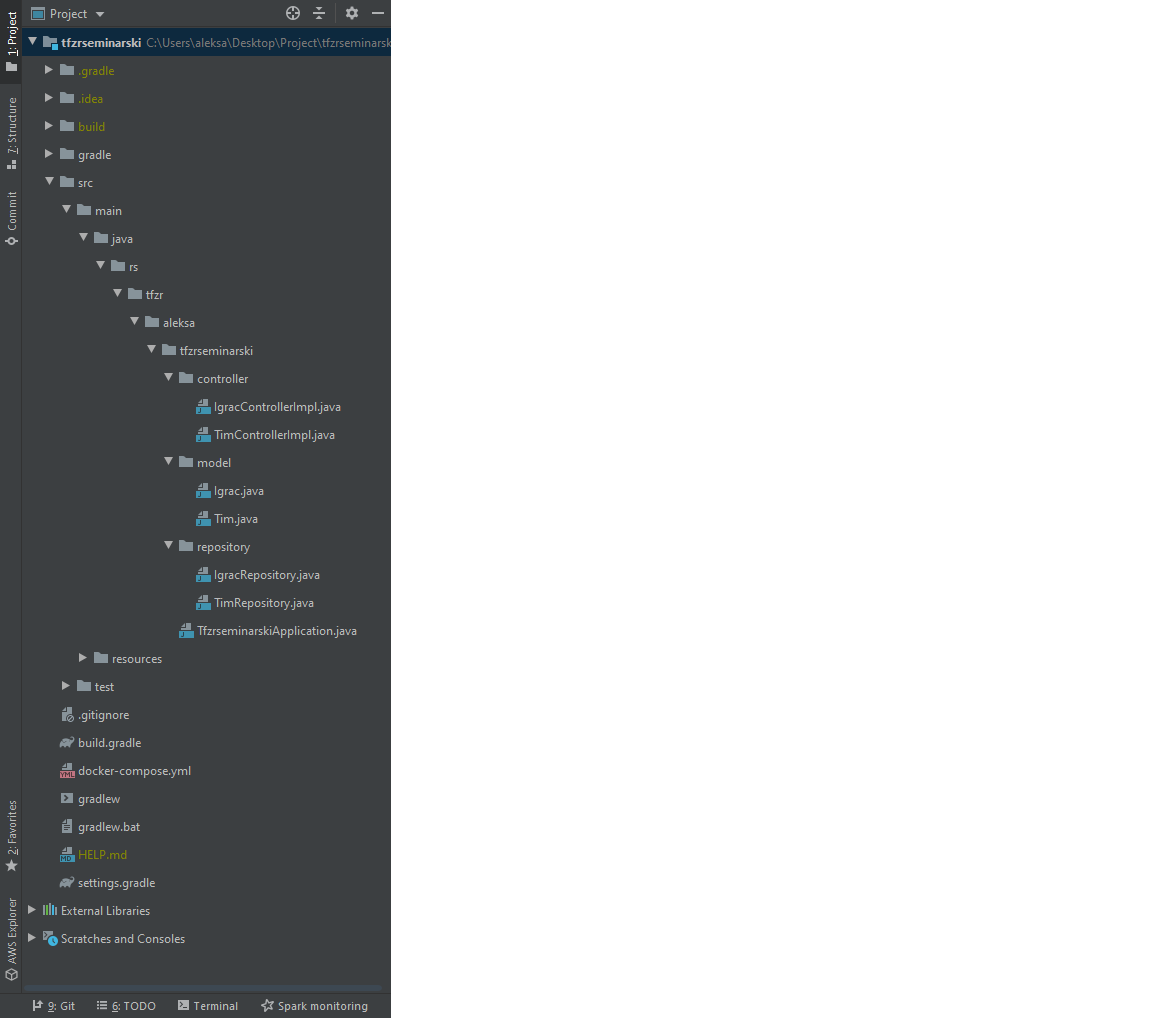
  
Docker se pokreće, I kada se virtuelne mašine podignu, preko docker compose fajla podižemo Postgres server definisan u fajlu bez potrebe da skidamo I insatliramo isti.



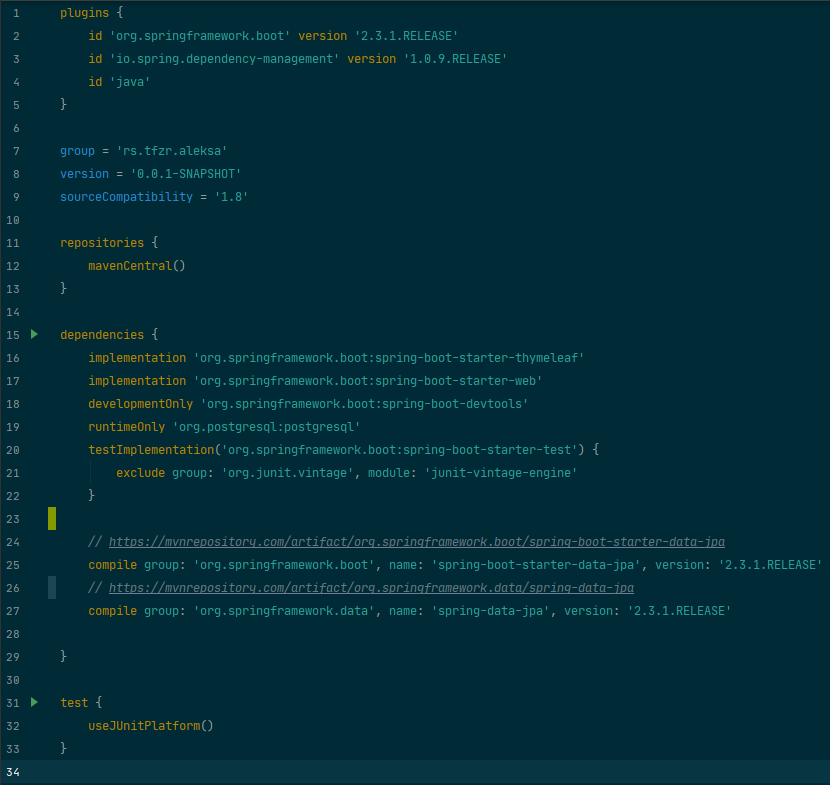
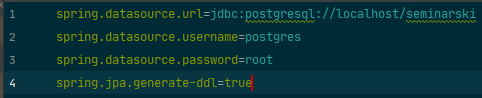
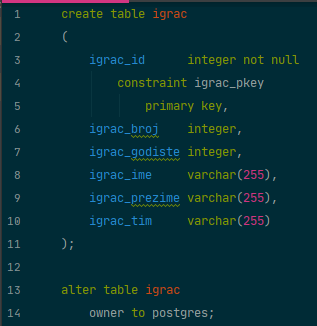
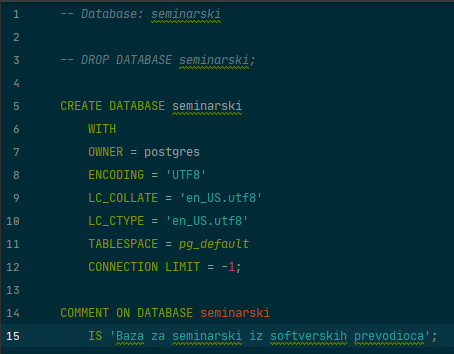
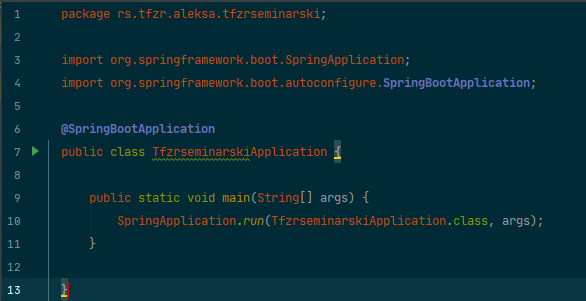


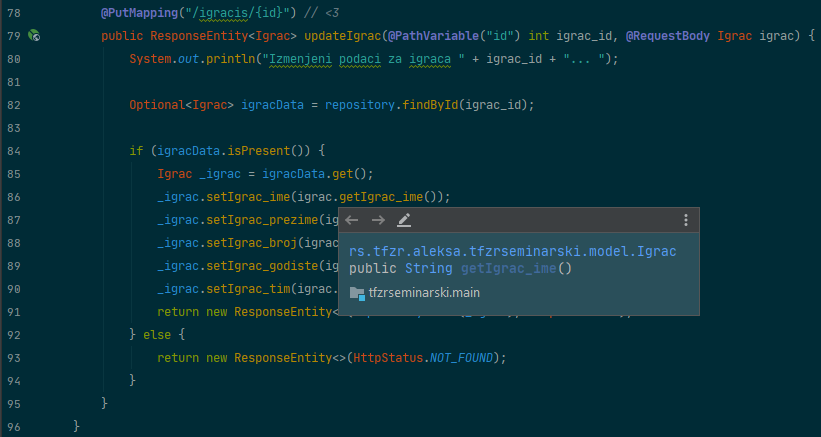
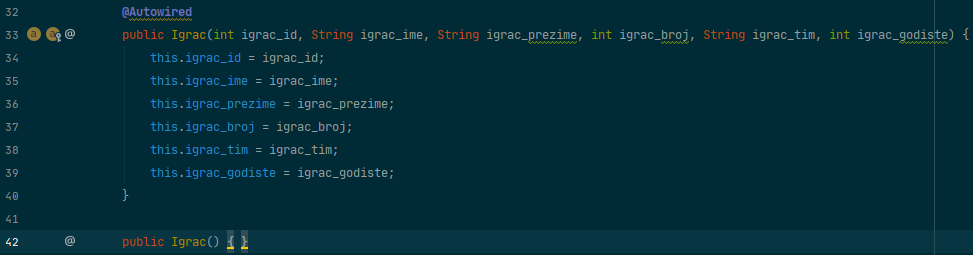


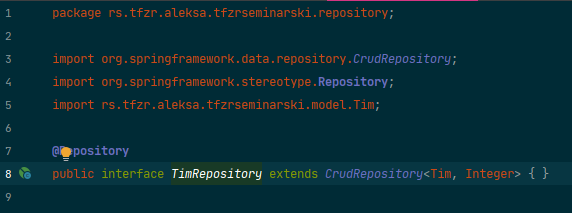


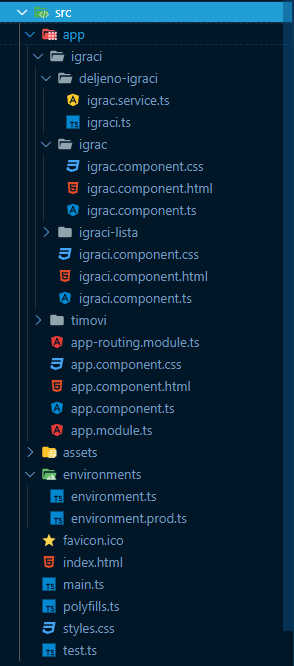


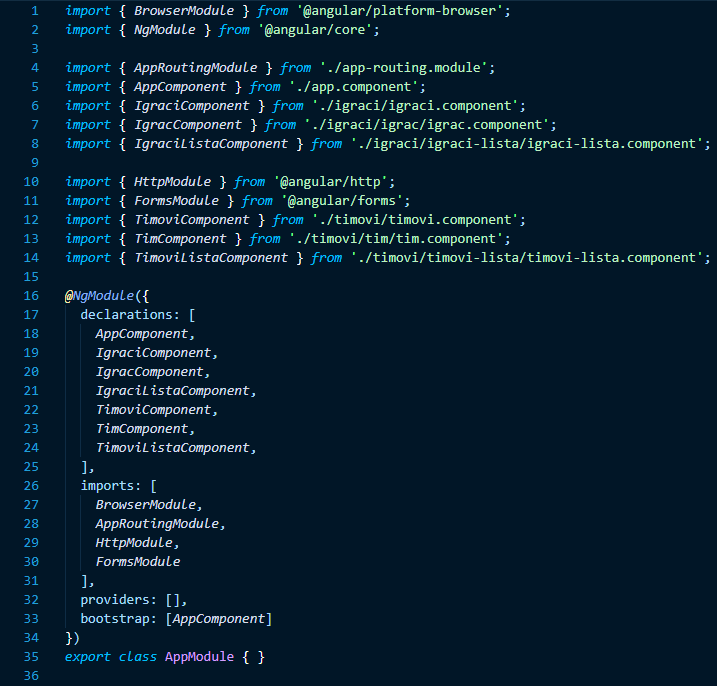
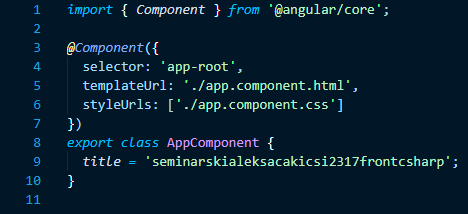
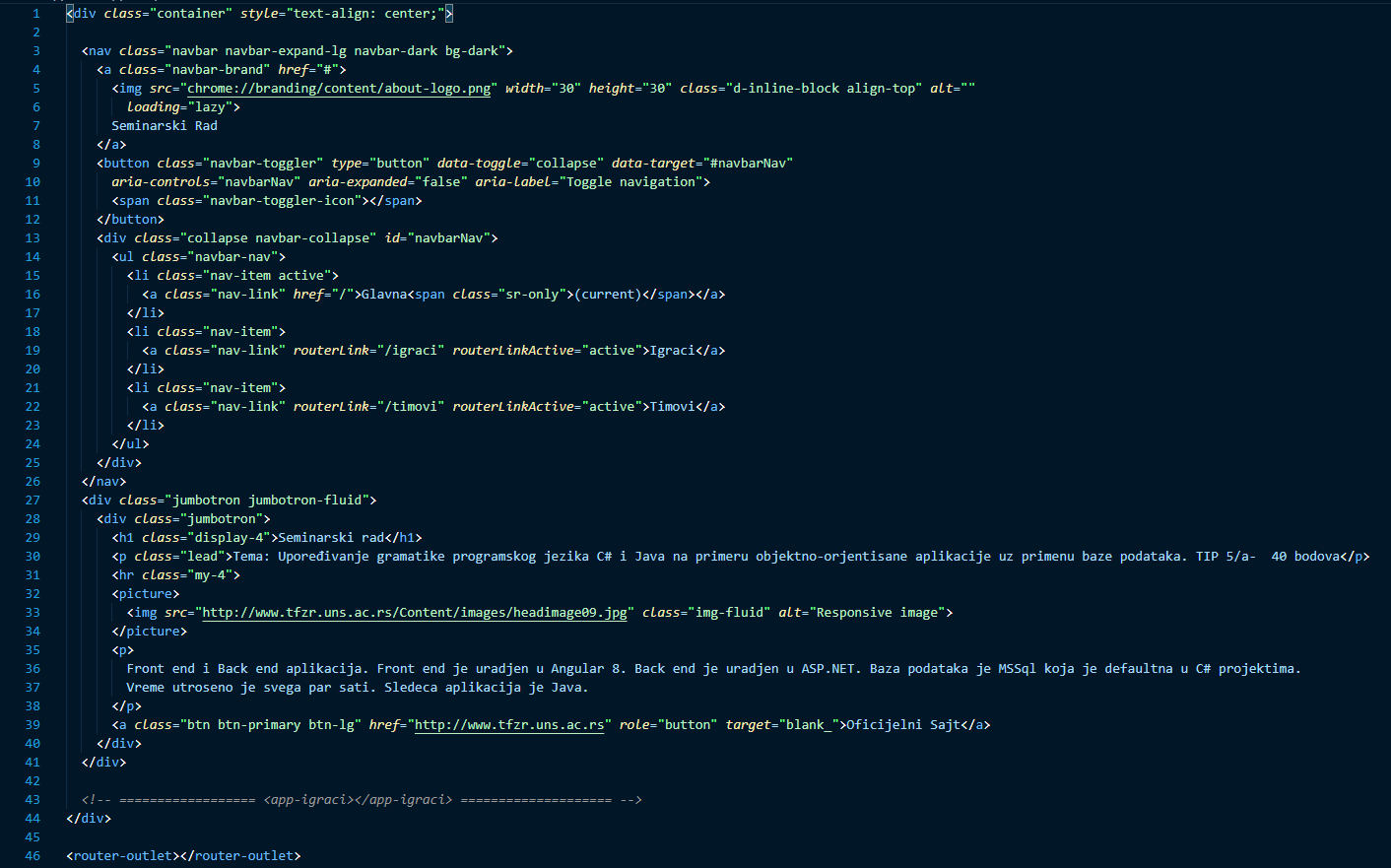
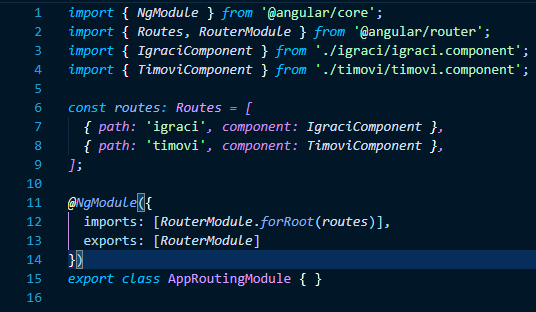
  
Struktura java projekta I Docker compose fajl.

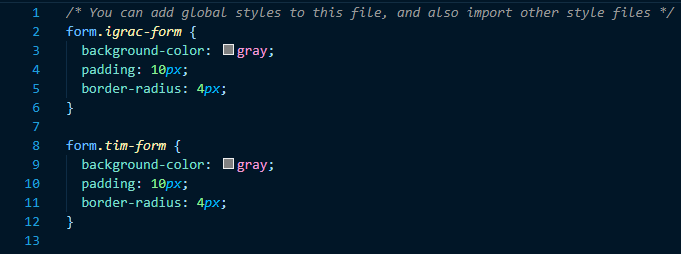
  
Gradle I Spring konfiguracija.  
  
Postgresql skripte

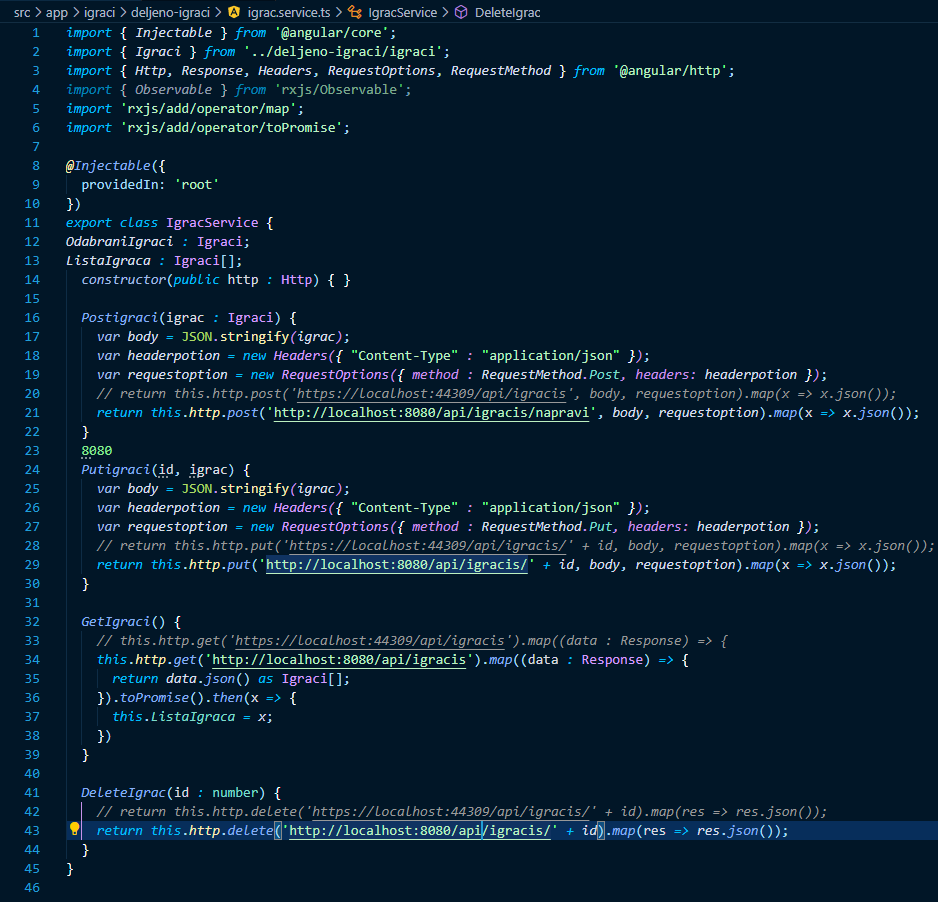
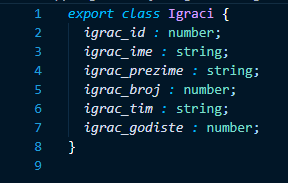
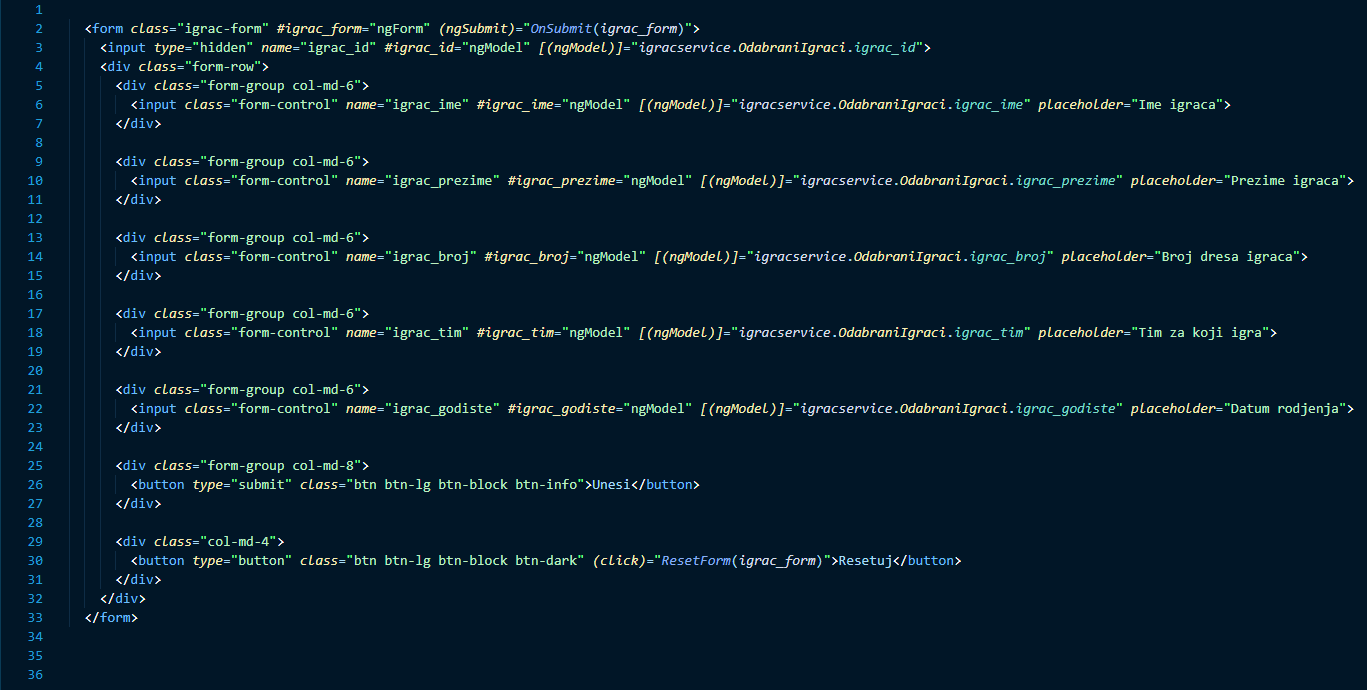
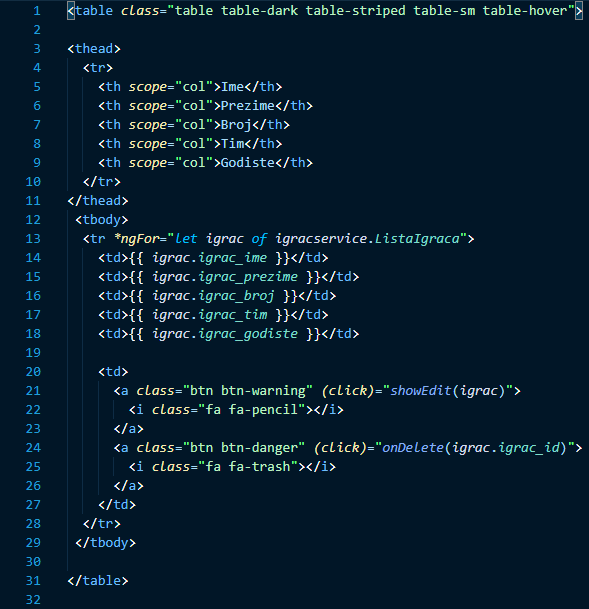


  
Modeli Igraca I Timova I kontroleri za iste.

  
Front end struktura projekta – prikaz zadatka



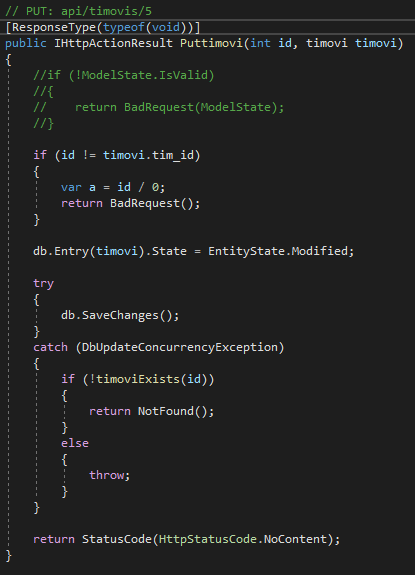


  
  
Projekat celokupno nije oduzeo više od dva sata.

## Greške

### RunTime

Greške koje nastaju za vreme rada programa (runtime errors) – jesu greške koje kompajleri/interpreteri ne mogu da uoče ali se one ipak ispoljavaju u toku rada tj. korišćenja programa. Ove greške, ukoliko se ne „uhvate“ posebnim programskim strukturama u kodu (najčešće try-catch metoda), mogu da izazovu prestanak rada programa uz gubitak svih podataka i međurezultata neke obrade koji su postojali do momenta pojave greške. Tipični primeri ovakvih grešaka u kodu se pojavljuju kada se jednoj promenjivoj numeričkog tipa pokuša da dodeli vrednost teksutalnog tipa koja ne može da podleže konvertovanju u broj (osim u slučaju ako sam tekst nije i broj). Tu je zatim tipična greška deljenja nekog broja sa nulom, pokušaj povezivanja na server baze podataka kada su parametri konekcije loše zadati ili kada je server neaktivan i sl. Runtime greške, ukoliko postoji dobra praksa programiranja, mogu da budu uhvaćene, ubeležene (log sistem) i preduhitrene na način koji neće obustaviti dalji rad programa.

Izgled RunTime greške u C#  


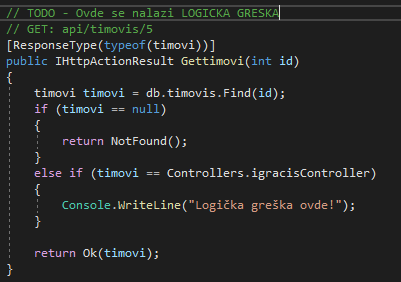
Ovde runtime greška u if upitu. Promenljiva a je količnik kojem je delilac 0. Time se stvara runtime greška.

Izgled RunTime greške u Javi  

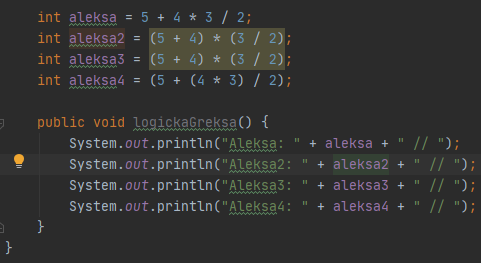

Kao I u prethodnom primeru, količnik je nepostojeći broj, jer je delilac broj 0.

## **Logičke**

Aplikacija je uspešno kompajlirana, prilikom rada ne prijavljuje nikakve greške niti prestaje sa radom. Sve je u redu, ali vaš program “samo” ne radi ono što treba da radi. Vraća pogrešne rezultate, ne upisuje podatke i tome slično. Ovo su verovatno greške koje je najteţe otkriti jer su posledice lošeg dizajna aplikacije.

Izgled logičke greške u C#  


Greška koja se ovde nalazi je u else if bloku, naime, timovi I igracisController nisu istog tipa, I time nema smisla ni da se proveravaju.

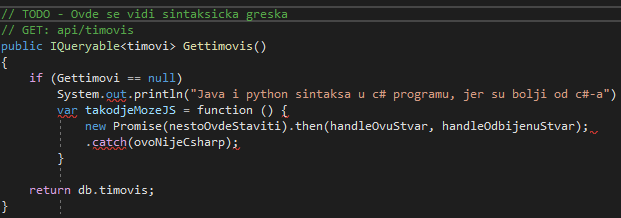
Izgled logičke greške u Javi  


Logičke greše koje se ovde dešavaju će rezultirati u različitim rezultatima.

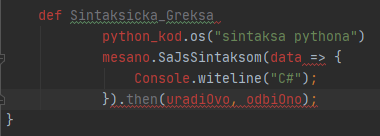
## **Sintaksičke**

Slično se dešava i u programiranju. Pravila za pisanje programa su vrlo precizna i za svaki deo programa je tačno određeno kako se može napisati i šta u tom slučaju znači. Skup pravila za pisanje programa na nekom jeziku naziva se sintaksa. Sintaksna greška nastaje kada kompjuter ne razume neki deo tvog programa, zato što nije napisan tačno po pravilima.

Kao što smo videli, može biti dovoljno da se u rečenici zameni samo jedno slovo, pa da ona dobije smisao. Ipak, kompjuteru je sasvim svejedno da li je rečenica (program) skoro ispravna ili potpuno besmislena, poput „Dobrome dva nežno krug“ ili “adf#fgh$ty!p o?e&c\*nm^,.%”. Kompjuter zahteva potpunu preciznost u izražavanju i uvek samo prijavljuje šta tačno ne razume, jer nije sposoban da pogodi šta je trebalo da bude napisano i da sam ispravi grešku, ma koliko mala bila.

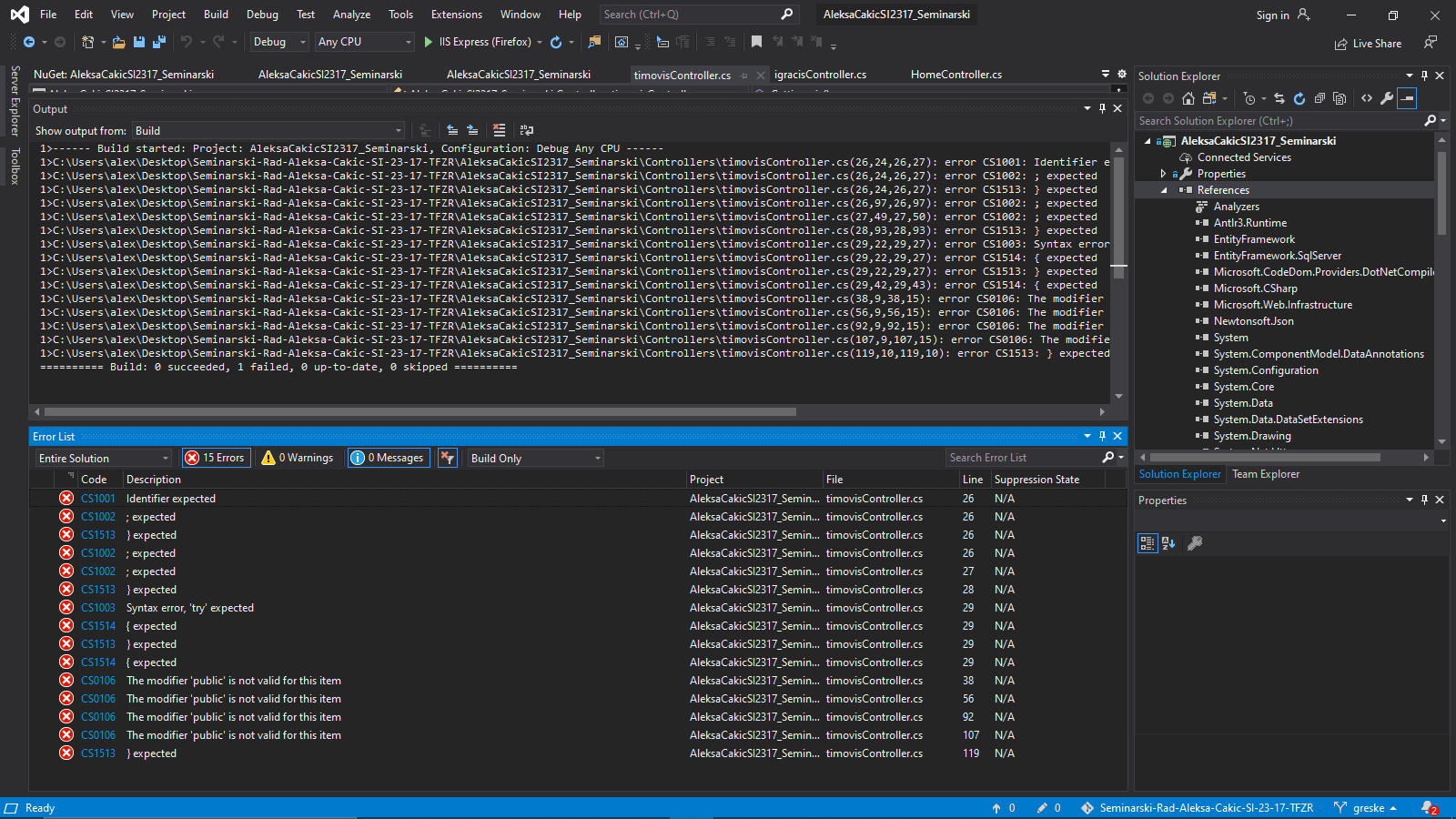
Izgled sintaksičke greške u C#  


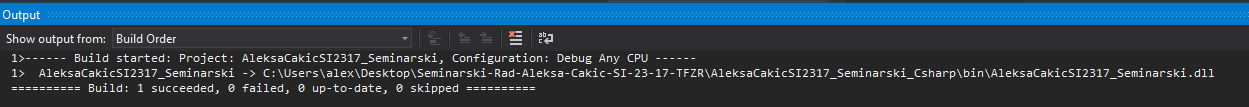
Sintaksička greša ovde je sto se u if bloku ne nalazi C# kod, već JavaScript jezik. Koliko god apsurdno zvučalo, ovo jeste sintaksička greška, isto koliko je I izostavljanje “;”.

Izgled sintaksičke greške u Javi  


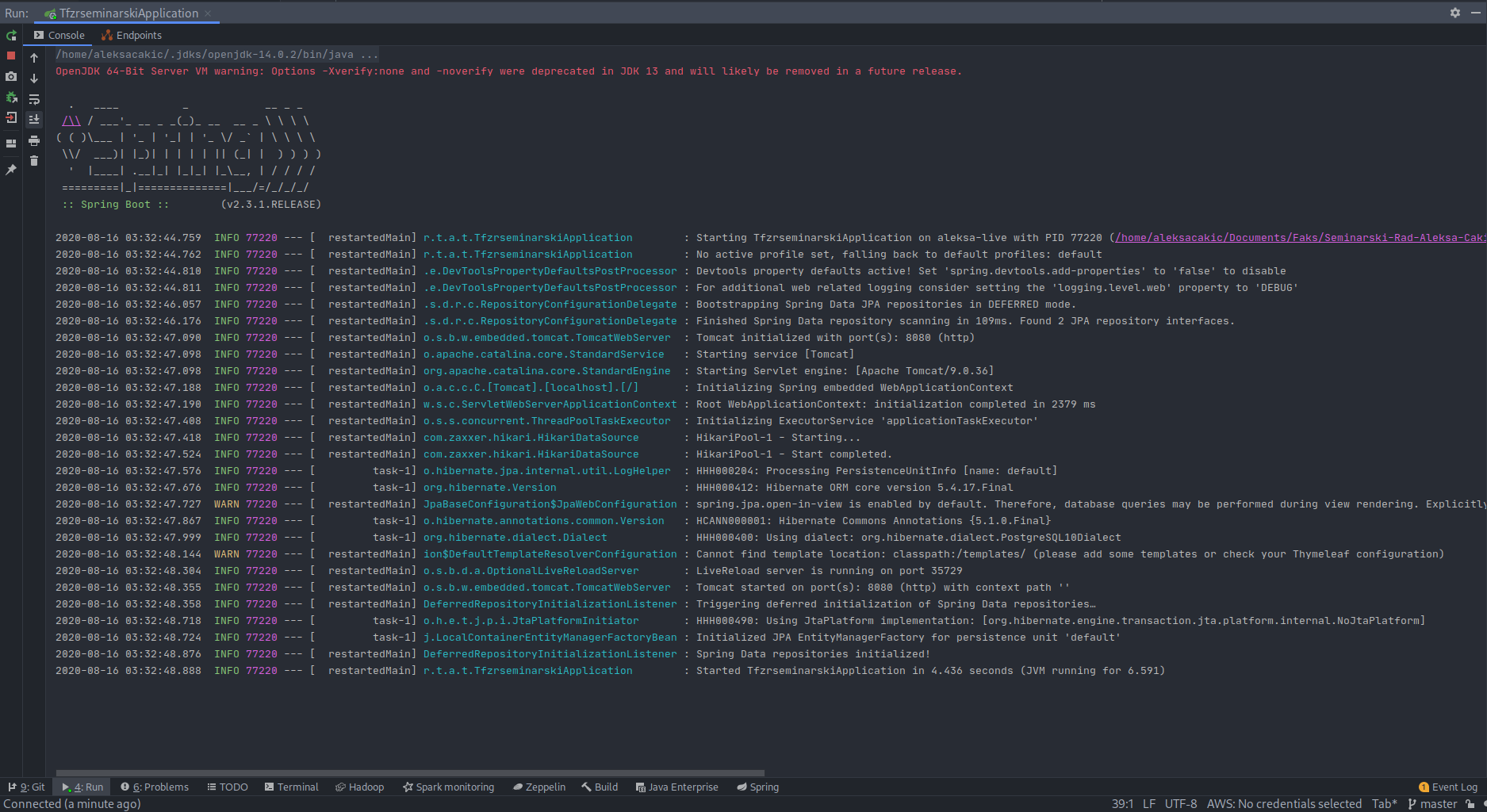
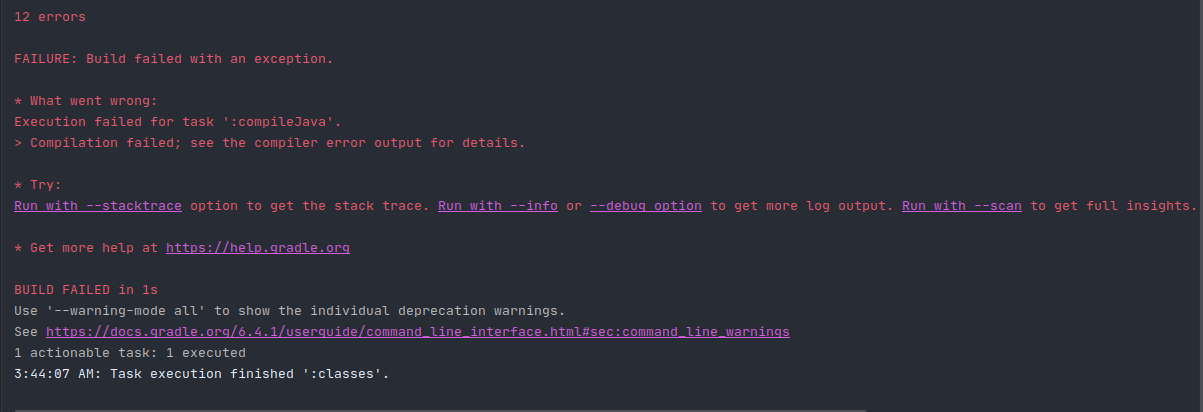
Isti slučaj kao I u prethodnom primeru. Samo što je ovde uključen I Python programski jezik.

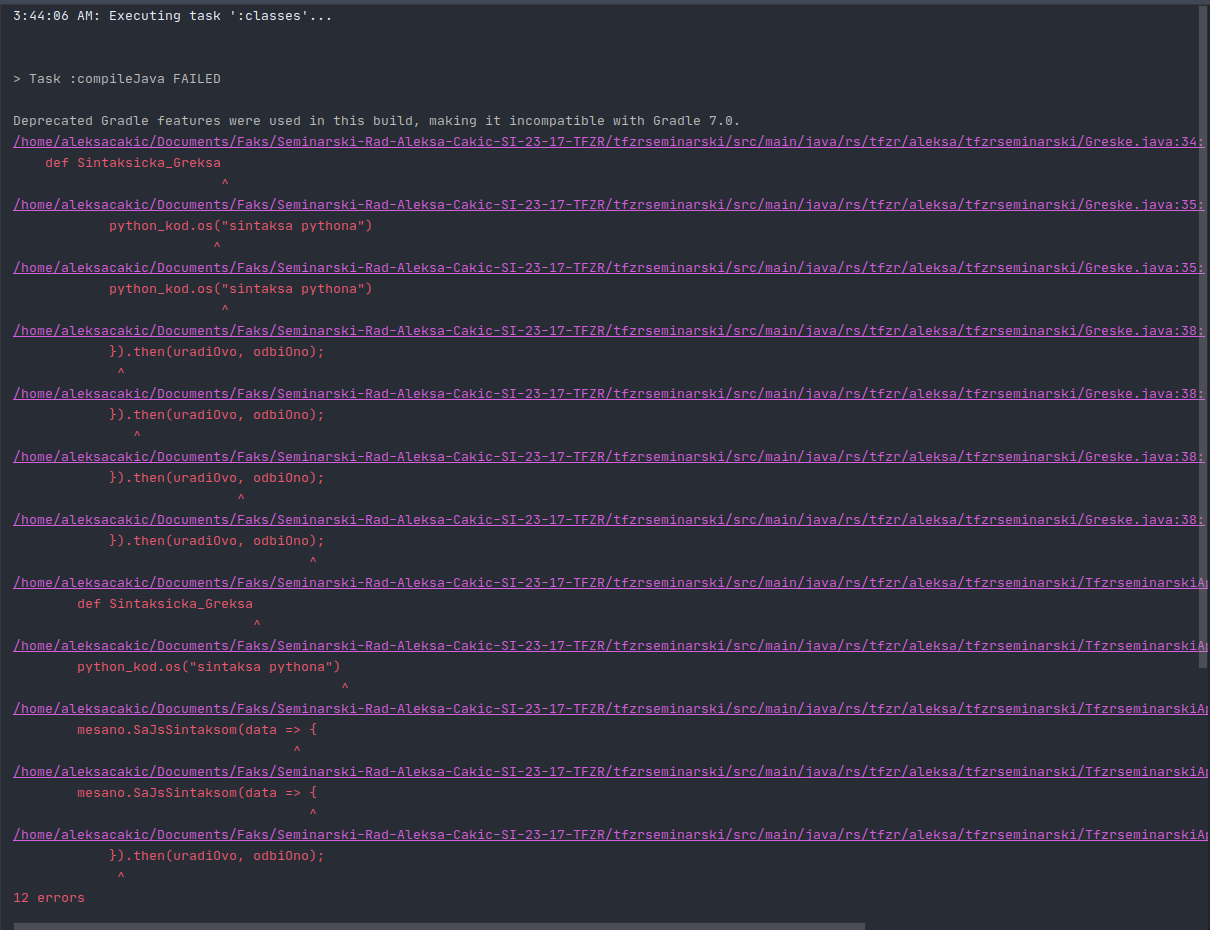
## Kako se kompajler ponaša

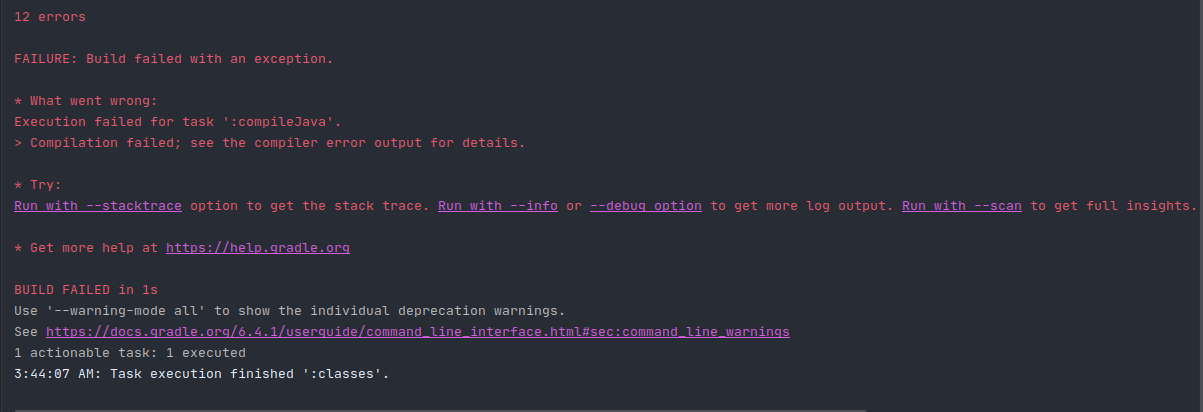
Ovde se vidi kako se ponaša VS sa errorima.

Ovo je poruka kada se uspešno kompajluje kod u VS-u.

Intellisense koji je razvio JetBrains, koji je posle postao najpopularniji IDE tool je bio adaptovan u IntelliJ idea IDE. Vremenom je postao sve bolji. Ovde se vidi kako upozorava korisnika.

Sa druge strane, ovde se vidi primer uspešnog kompajla sa porukama

Ovde se nalaze poruke sa svim greškama u Javi.



## EBNF prikaz

### Sintaksičke

<identifikator>::=<telo\_naredbe>”;”

<identifikator> ::=<telo\_naredbe>( [identifikator] )<separator>

<identifikator1> ::=<identifikator2><naziv\_metode>( [identifikator] )<separator>

<identifikator> ::=<identifikator2><naziv\_metode>( [identifikator]) || ()<separator>

<int> <identifikator> ::= cifra{cifra}||<string> <promenjiva\_tipa\_string> ::= <vrednost><cifra> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0

### Runtime

<int> <identifikator> ::= cifra{cifra} <cifra> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0

<int> <identifikator> ::= <mestoZaKastovanje><identifikator><separator>

Gledajući pravila koja jezici imaju, koliko mora biti tačno, zavisi od tipa jezika. Neki jezici su dinamički kucani, neki statično. Treba imati u vidu tačnu sintaksu. Ne samo što je važno imati tačnu sintaksu, I malo razuma kako se greške ne bi dešavale, takođe je važno imati znanje o jeziku u kome se piše aplikacija, OS, sub-sistem ili bilo šta drugo.

## Korisnički interfejs

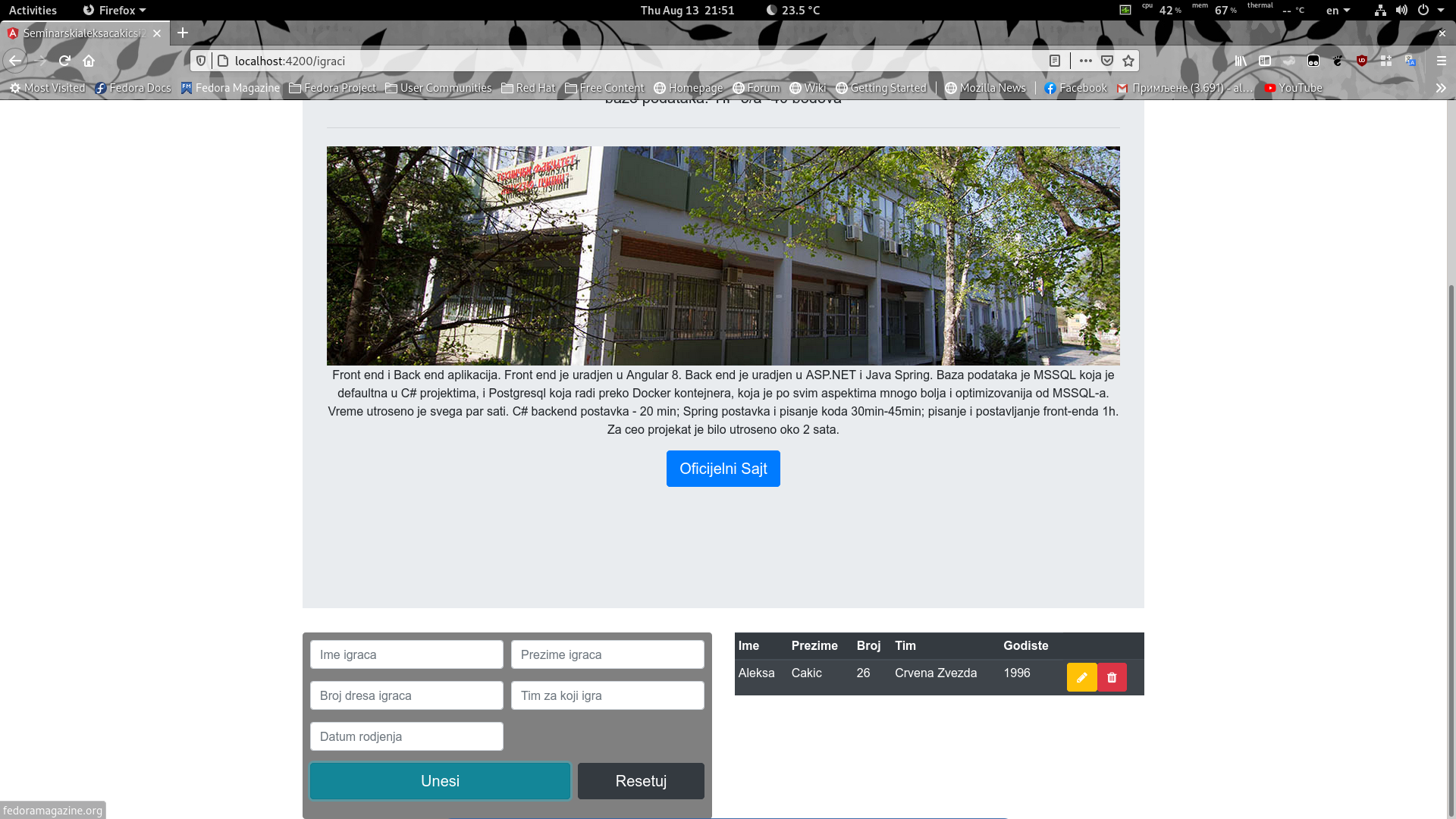
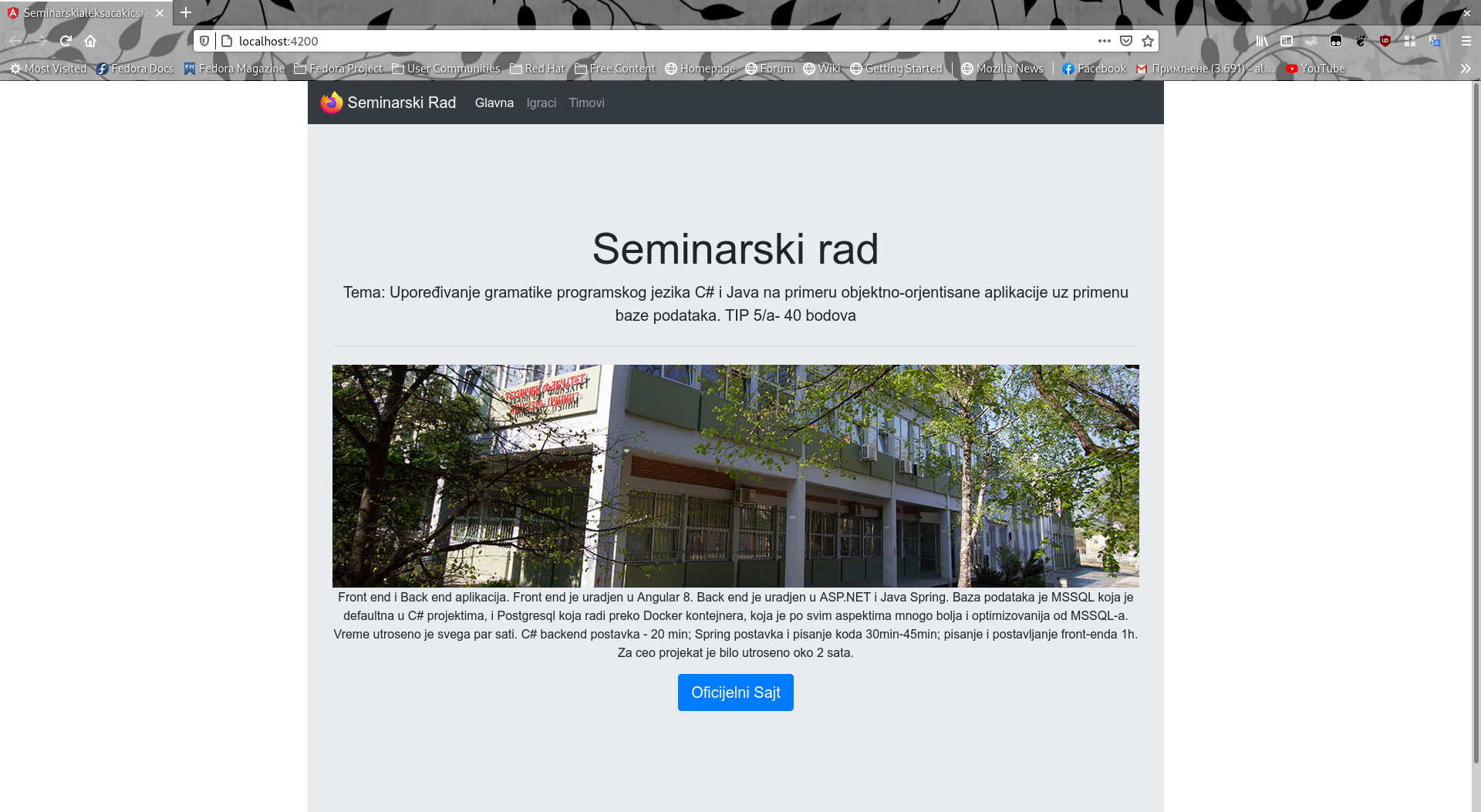
Pošto je front end SPA, korinik u navbaru može da odabere da li će ubaciti tim ili igrača. Postoje dva dugmeta kojim korinik može da unese ili resetuje sam unos. U tabeli pored, korisnik ima opciju da izmeni neki od unosa ili obriše.

Figure 1

Sadržaj

[1. Malo o C# 2](#__RefHeading___Toc3393_1116422282)

[2. Malo o MSSQLU 3](#__RefHeading___Toc3395_1116422282)

[3. Uputstvo 8](#__RefHeading___Toc3397_1116422282)

[4. JAVA SETUP 23](#__RefHeading___Toc3399_1116422282)

[5. Greške 47](#__RefHeading___Toc3401_1116422282)

[1. RunTime 47](#__RefHeading___Toc3403_1116422282)

[6. Logičke 49](#__RefHeading___Toc3405_1116422282)

[7. Sintaksičke 50](#__RefHeading___Toc3407_1116422282)

[8. Kako se kompajler ponaša 52](#__RefHeading___Toc3424_1116422282)

[9. EBNF prikaz 56](#__RefHeading___Toc3426_1116422282)

[Sintaksičke 56](#__RefHeading___Toc3428_1116422282)

[Runtime 56](#__RefHeading___Toc3430_1116422282)

[10. Korisnički interfejs 57](#__RefHeading___Toc3432_1116422282)