Visoka škola za menadžment i dizajn Aspira

Smjer: Računarstvo

Kolegij: IT sustavi u oblaku

SPRING BOOT - JAVA

Seminarski rad

Nastavnik: Ante Projić Student: Lucija Mikulić

Split, svibanj, 2022. godina

# 1. SADRŽAJ:

[1. SADRŽAJ: 2](#_Toc104560156)

[2. UVOD 3](#_Toc104560157)

[3. SPRING 3](#_Toc104560158)

[3.1 KAKO SPRING BOOT RADI? 4](#_Toc104560159)

[4. IZRADA APLIKACIJE SA SPRING BOOT-OM 5](#_Toc104560160)

[5. IZGLED SPRING BOOT APLIKACIJE 7](#_Toc104560161)

[6. KREIRANJE JEDNOSTAVNE WEB APLIKACIJE 10](#_Toc104560163)

[7. POKRETANJE APLIKACIJE 12](#_Toc104560164)

[8. DODAVANJE UNIT TESTOVA 15](#_Toc104560165)

[9. DODAVANJE PRODUCTION GRADE SERVISERA 17](#_Toc104560166)

[10. PODRŠKA ZA JAR I GROOVY PODRŠKA 19](#_Toc104560167)

[11. ZAKLJUČAK 22](#_Toc104560168)

[12. LITERATURA 23](#_Toc104560169)

# 2. UVOD

Spring Framework razvojno je okruženje otvorenog koda (engl. Open source) kreirano 2013. godine od strane Rod Johnsona kao poboljšanje dotad korištenog Java Enterprise Edition (JEE) standarda za izradu robusnih web aplikacija. Za razliku od JEE, Spring je pridobio naklonost korisnika dvjema karakteristikama: inverzija kontrole (engl. Inversion of Control) odnosno sam framework kontrolira izvršavanje programskog koda te injektiranje objekata o kojima ovisi rad koda (engl. Dependency Injection). Nastanak Spring-a, osim što je olakšao kreiranje robusnih Java aplikacija, i dalje je zahtijevao veliku količinu konfiguracijskih klasa i napora prilikom same inicijalizacije projekata.

Spring Boot jedan je od modula koji je nastao kako bi uz čim manje truda i konfiguriranja projekt bio inicijaliziran i spreman za daljnji razvoj. Neke od karakteristika Spring Boot-a su: unaprijed pripremljene funkcionalnosti (eng. Out of the box functionalities), nema generiranja klasa i koda već se koriste unaprijed definirane biblioteke te set nefunkcionalnih alata i klasa u pogledu konfiguracije i pokretanja servera, sigurnosti, metrike te ostalih pomoćnih poslova.

Spring Boot je samo proširenje već postojećih i ekspanzivnih Spring okvira, ali ima neke specifične značajke koje aplikaciju čine lakšom za rad unutar razvojnog ekosustava. To proširenje uključuje unaprijed konfigurabilne početne komplete za web koji pomažu olakšati odgovornosti aplikacijskog poslužitelja koje su potrebne za druge Spring projekte.

# 3. SPRING

Spring Boot pruža dobru platformu za Java programere za razvoj samostalne i proizvodne Spring aplikacije koju možemo jednostavno pokrenuti. Možemo započeti s minimalnim konfiguracijama bez potrebe za cijelom postavkom Spring konfiguracije.

Pruža fleksibilan način konfiguriranja Java Bean-ova, XML konfiguracija i transakcija baze podataka. Također pruža moćnu grupnu obradu i upravlja krajnjim točkama REST-a. U Spring Boot-u sve je automatski konfigurirano, nisu potrebne ručne konfiguracije. Nudi spring aplikaciju temeljenu na anotacijama. Olakšava upravljanje dependecy-ijima te uključuje ugrađeni spremnik servleta.

# 3.1 KAKO SPRING BOOT RADI?

Spring Boot automatski konfigurira aplikaciju na temelju dependency-ja koje smo dodali projektu pomoću napomene @EnableAutoConfiguration.

Na primjer, ako je MySQL baza podataka na classpath-u, ali nismo konfigurirali nijednu vezu s bazom podataka, tada Spring Boot automatski konfigurira bazu podataka u memoriji.

Ulazna točka spring aplikacije za pokretanje je klasa koja sadrži napomenu @SpringBootApplication i glavnu metodu. Spring Boot automatski skenira sve komponente uključene u projekt pomoću napomene @ComponentScan.

Ako klasi dodamo napomenu @SpringBootApplication, ne moramo dodati bilješku @EnableAutoConfiguration, @ComponentScan i @SpringBootConfiguration. Bilješka @SpringBootApplication uključuje sve ostale napomene.

Spring Boot aplikacija skenira sve bean-ove i deklaracije paketa kada se aplikacija inicijalizira. Moramo dodati napomenu @ComponentScan za datoteku klase da bismo skenirali svoje komponente dodane u projekt.

# 4. IZRADA APLIKACIJE SA SPRING BOOT-OM

Praktični dio ovog seminarskog rada daje prikaz izrade jednostavne web aplikacije koristeći Spring Boot, te prikaz usluga koje Spring Boot pruža.

Za izradu web aplikacije potreban je text editor ili IDE (Integrated Development Environment), u ovom slučaju Visual Studio Code editor. Potrebno je imati instaliran JDK (Java Development Kit) 1.8 ili noviju verziju, Maven 3.2+ i Spring Boot CLI naredbeni redak.

U Command Prompt-u (Naredbeni redak) vršimo provjere za Javu, Maven i Spring Boot CLI.

Slika 1: CMD, verzija Jave

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Slika 2: CMD, verzija Mavena

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Slika 3: CMD, verzija Spring Boot CLI



Ukoliko u nekoj od provjera, Command Prompt ne prepozna naredbu, potrebno je namjestiti putanje (path) sistemskih i korisničkih varijabli. Kod Windows 10 operativnog sustava, to radimo tako da desnim klikom otvorimo „Svojstva“ u „Ovaj PC“ direktoriju. Otvorimo „Dodatne postavke sustava“, pod „Dodatno“ izaberemo „Varijable okruženja“ i dodamo korisničke i sistemske varijable kao na slici.

Slika 4: Windows 10, Varijable okruženja

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Projekt započinjemo s Spring Initializr-om. Radi se o servisu koji povlači sve dependency-e koji nam trebaju u aplikaciji i konfigurira većinu postavki za nas.

U Spring Initializr-u izabiremo Maven umjesto Gradle-a, Java jezik te upisujemo osnovne podatke o projektu. Maven je popularni alat otvorenog koda za gradnju Java projekata, dizajniran da izvuče veći dio teškog rada iz procesa izgradnje. Budući da se radi o izradi web aplikacije, pod „Dependencies“ biramo Spring Web dependency. Spring Web daje podršku za web sadržaj, RESTful pozive i koristi Apache Tomcat server (web container). Klikom na „Generate“ gumb, skida se ZIP file koji je arhiva naše web aplikacije, konfigurirana točno kako smo odabrali u Spring Initializr izborniku.

Slika 5: Spring Initializr

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Izvor: https://start.spring.io/

Skinutu spring-boot.zip datoteku je potrebno izdvojiti i otvoriti u Visual Studio Code-u.

# 5. IZGLED SPRING BOOT APLIKACIJE

Slika 6: Izgled učitane Spring Boot aplikacije

# Slika na kojoj se prikazuje tekst Opis je automatski generiran

Slika 7: Spring Boot struktura aplikacije

Slika na kojoj se prikazuje tekst, monitor, snimka zaslona, zaslon

Opis je automatski generiran

.mvn folder sadrži Maven wrapper-e. Spring Boot koristi Maven wrapper umjesto samog Maven-a. To znači da nije potrebno skinuti i instalirati Maven, već on dolazi unutar Spring Boot projekta.

Slika 8: pom.xml datoteka

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Pom.xml file sadrži po default-u dva dependencija. Prvi je onaj kojeg smo dodali u Spring Initializr-u (spring-boot-starter-web), a drugi (spring-boot-starter-test) je Spring Boot dodao sam, jer pretpostavlja da ćemo raditi test za svoju aplikaciju. Unutar pom.xml file-a možemo dodavati dependencije i nakon stvaranja projekta u Spring Initializr-u.

Slika 9: Spring Boot main klasa

Slika na kojoj se prikazuje tekst, snimka zaslona, monitor, zaslon

Opis je automatski generiran

Spring Initializr kreira jednostavnu aplikacijsku klasu, main application class. Prva anotacija @SpringBootApplication govori Springu da se radi o Spring Boot projektu. To znači da će Spring Boot napraviti niz pretpostavki o izgledu koda ili o vremenu izvršavanja. Ovisno o tome koji je dependency potreban, kod na liniji 10 (SpringApplication.run(Application.class, args);) je glavna run metoda za našu Spring aplikaciju.

Slika 10: ApplicationTest.java klasa

Slika na kojoj se prikazuje tekst, snimka zaslona, zaslon

Opis je automatski generiran

Unutar Test direktorija se nalazi ApplicationTests klasa koja trenutno samo provjerava da li može učitati sadržaj spring aplikacije. Ovaj test je koristan početak za integraciju testova.

# 6. KREIRANJE JEDNOSTAVNE WEB APLIKACIJE

Potrebno je izraditi web kontroler HelloController.java kao na slici.

Slika 11: HelloController.java klasa

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Klasa ima anotaciju @RestController, što znači da je spremna za korištenje od strane Spring MVC-a za rukovanje web zahtjevima. @GetMapping mapira / sa index() metodom. Kada se pozove iz preglednika ili korištenjem curl-a u naredbenom retku, metoda vraća čisti tekst. To je zato što @RestController kombinira @Controller i @ResponseBody, dvije anotacije koje rezultiraju da web zahtjevi vraćaju podatke, a ne prikaz.

Potrebno je modificirati Application.java klasu kao na slici:

Slika 12: Modificirana main klasa

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

U Spring-u, objekti koji čine okosnicu aplikacije i kojima upravlja Spring IoC spremnik nazivaju se beans. Bean je objekt koji je instanciran, sastavljen i kojim na drugi način upravlja Spring IoC spremnik. Inverzija kontrole (IoC) je proces u kojem objekt definira svoje dependencije bez njihovog stvaranja. Ovaj objekt delegira posao izgradnje takvih ovisnosti na IoC spremnik.

@SpringBootApplication je praktična anotacija koja dodaje sve od sljedećeg:

@Configuration – označava klasu kao izvor bean definicija za kontekst aplikacije.

@EnableAutoConfiguration – govori Spring Boot-u da počne dodavati bean-ove na temelju postavki putanje klase. Npr., ako je spring-webmvc na putanji klase, anotacija označava aplikaciju kao web aplikaciju i aktivira ključna ponašanja, kao što je postavljanje DispatcherServlet-a.

@ComponentScan – govori Spring Boot-u da potraži druge komponente, konfiguracije i usluge u paketu com/example, dopuštajući mu da pronađe kontrolere.

Aplikacija je napisana u stopostotnoj Javi, te se nije potrebno baviti konfiguriranjem infrastrukture.

Postoji i metoda CommandLineRunner označena kao @Bean, a ona se pokreće pri pokretanju. Dohvaća sve bean-ove koje je stvorila aplikacija ili ih je automatski dodao Spring Boot. Razvrstava ih i ispisuje.

# 7. POKRETANJE APLIKACIJE

Da bi pokrenuli aplikaciju, potrebno je upisati ./mvnw spring-boot:run naredbu u terminal (u potpunom direktoriju).

Slika 13: Pokretanje aplikacije

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Output:

Slika 14: Output 1

Slika na kojoj se prikazuje tekst, snimka zaslona, monitor, zaslon

Opis je automatski generiran

Slika 15: Output 2

Slika na kojoj se prikazuje tekst, snimka zaslona, monitor

Opis je automatski generiran

Slika 16: Output 3

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Iz output-a se mogu vidjeti org.springframework.boot.autoconfigure beans, kao i tomcatEmbeddedServletContainerFactory.

U odvojenom terminalu je potrebno pokrenuti curl servis s naredbom: curl localhost:8080

Slika 17: curl naredba/prikaz localhost-a

Slika na kojoj se prikazuje tekst, snimka zaslona, monitor

Opis je automatski generiran

U web pregledniku, na portu 8080 se ispisuje poruka „Greetings from Spring Boot!“.

Slika 18: prikaz /goodbye endpoint-a

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Dodavanjem jos jednog endpointa, /goodbye, u web pregledniku nam se ispisuje „Goodbye from Spring Boot!“.

# 8. DODAVANJE UNIT TESTOVA

Potrebno je napraviti jednostavan unit test HelloControllerTest.java koji oponaša servlet zahtjev i odgovor kroz naš endpoint. Spring Test nam daje mehanizam za to.

Slika 19: HelloControllerTest.java

Slika na kojoj se prikazuje tekst, snimka zaslona, monitor, srebrno

Opis je automatski generiran

MockMvc dolazi iz Spring Testa i omogućuje kroz skup zgodnih graditelja klasa, pošaljemo HTTP zahtjeve u DispatcherServlet i damo tvrdnje o rezultatu.

Zadatak DispatcherServleta je uzeti dolazni URI i pronaći pravu kombinaciju rukovatelja (obično metoda na klasama Controllera) i pogleda (općenito JSP-ova) koji se kombiniraju kako bi tvorili stranicu ili resurs koji bi se trebao naći na tom mjestu.

Obratimo pažnju na upotrebu @AutoConfigureMockMvc i @SpringBootTest za ubacivanje MockMvc instance. Nakon što smo koristili @SpringBootTest, tražimo da se izradi cijeli kontekst aplikacije. Alternativa bi bila zamoliti Spring Boot da stvori samo web slojeve konteksta pomoću @WebMvcTest. U oba slučaja, Spring Boot automatski pokušava locirati glavnu klasu aplikacije aplikacije, ali možemo je nadjačati ili suziti ako želimo izgraditi nešto drugačije.

Osim oponašanja ciklusa HTTP zahtjeva, također možete koristiti Spring Boot za pisanje jednostavnog testa full-stack integracije. Na primjer, umjesto prethodno prikazanog lažnog testa, mogli bismo kreirati sljedeći test.

Slika 20: HelloControllerIT.java

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Ugrađeni poslužitelj pokreće se na slučajnom portu zbog webEnvironment = SpringBootTest.WebEnvironment.RANDOM\_PORT, a stvarni port se automatski konfigurira u osnovnom URL-u za TestRestTemplate.

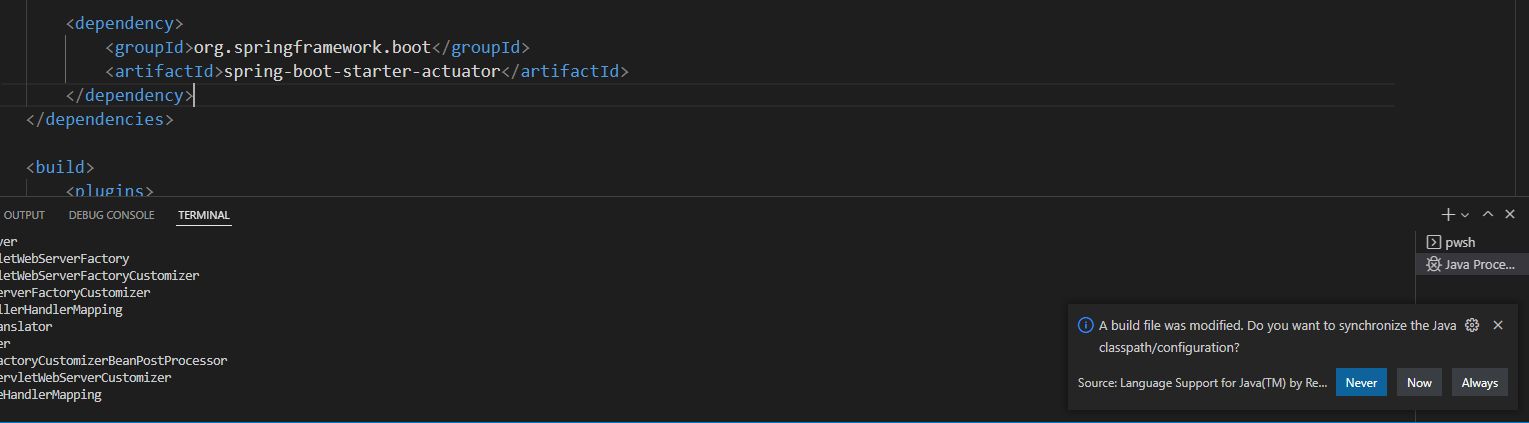
# 9. DODAVANJE PRODUCTION GRADE SERVISERA

U računalnoj i tehnološkoj terminologiji, "Production grade" izraz je koji se koristi za opisivanje robusnijeg i robusnijeg hardvera i softvera koji je dizajniran za intenzivna poslovna i poslovna računalna okruženja. To je u suprotnosti s potrošnom opremom, koja bi se koristila rjeđe i za manje intenzivne zadatke.

Za izgradnju web stranice za tvrtku, vjerojatno trebamo dodati neke usluge upravljanja. Spring Boot pruža nekoliko takvih usluga (kao što su zdravlje, revizije (audits), beans i još mnogo toga) sa svojim modulom pokretača (actuator module).

Potrebano je dodati u pom.xml file novi dependency. (<artifactId>spring-boot-starter-actuator</artifactId>)

Slika 21: pom.xml, spring-boot-starter-actuator depdendency



Nakon dodavanja novog dependencija, moramo ponovno pokrenuti aplikaciju iz potpunog direktorija. Prilikom ponovnog pokretanja aplikacije, možemo vidjeti da je dodan novi set RESTful endpoint-ova. To su usluge upravljanja koje pruža Spring Boot. Sljedeći popis prikazuje tipičan output:

Slika 22: novi endpoint-ovi output

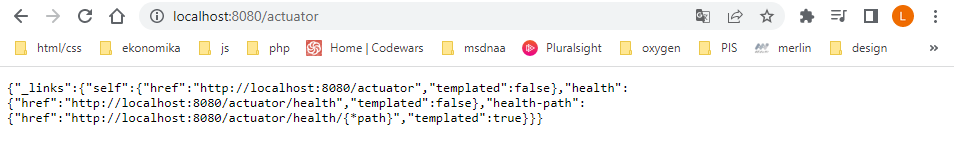
Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Actuator otkriva sljedeće:

* Actuator

Slika 23: Actuator



* actuator/health

Slika 24: Actuator/health

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Također zdravlje (health) aplikacije možemo pratiti unošenjem curl naredbe unutar terminala.

Slika 25: curl naredba za pokretanje health endpointa

Slika na kojoj se prikazuje tekst, monitor, snimka zaslona, srebrno

Opis je automatski generiran

Ukoliko pokušamo pozvati endopoint koji ne postoji, kao npr. endpoint shutdown, prikaže nam se ovakva poruka:

Slika 26: prikaz nepostojećeg endpoint-a

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

# 10. PODRŠKA ZA JAR I GROOVY PODRŠKA

JAR je skraćenica od Java ARchive. To je format datoteke koji se temelji na popularnom formatu ZIP datoteke i koristi se za združivanje mnogih datoteka u jednu. Iako se JAR može koristiti kao opći alat za arhiviranje, primarna motivacija za njegov razvoj bila je da se Java apleti i njihove potrebne komponente (.class datoteke, slike i zvukovi) mogu preuzeti u preglednik u jednoj HTTP transakciji, umjesto otvaranja nove veza za svaki komad. To uvelike poboljšava brzinu kojom se aplet može učitati na web stranicu i početi funkcionirati. JAR format također podržava kompresiju, što smanjuje veličinu datoteke i dodatno poboljšava vrijeme preuzimanja. Osim toga, pojedini unosi u JAR datoteci mogu biti digitalno potpisani od strane autora apleta kako bi se potvrdilo njihovo porijeklo. Applet je vrlo mala aplikacija, posebno uslužni program koji izvodi jednu ili nekoliko jednostavnih funkcija.

Spring Boot omogućuje sastavljanje izvršnih JAR-ova, zahvaljujući modulu za učitavanje Spring Boot-a. Različiti vodiči pokazuju ovu dvostruku podršku kroz spring-boot-gradle-plugin i spring-boot-maven-plugin.

Spring Boot također ima Groovy podršku, što omogućuje izradu Spring MVC web aplikacija sa samo jednom datotekom. Groovy je dinamički objektno orijentirani programski jezik za Java virtualni stroj (JVM) koji se može koristiti gdje god se koristi Java. Jezik se može koristiti za kombiniranje Java modula, proširenje postojećih Java aplikacija i pisanje novih aplikacija.

Potrebno je stvoriti novu datoteku app.groovy koja izgleda kao na slici. Nije bitno gdje se datoteka nalazi jer je jako mala.

Slika 27: app.groovy

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Za pokretanje Groovy aplikacije moramo imati instaliran Spring Boot CLI.

Prije pokretanja aplikacije moramo ugasiti prethodnu aplikaciju kako nebi došlo do kolizije na portu.

U terminalu upisujemo naredbu spring run app.groovy.

Slika 28: terminal, spring naredba za pokretane groovy aplikacije

Slika na kojoj se prikazuje tekst, snimka zaslona, monitor, zaslon

Opis je automatski generiran

Nova aplikacija je pokrenuta na portu 8080, i ispisuje poruku „Luce loves Spring Boot!“.

Slika 29: Ispis „Luce loves Spring Boot!“

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Spring Boot to čini dinamičkim dodavanjem anotacija našem kodu i korištenjem Groovy Grapea za povlačenje library-a koje su potrebne za pokretanje aplikacije.

# 11. ZAKLJUČAK

Spring Boot postao je sastavni dio Java ekosustava, nudeći učinkovit i skalabilan alatni okvir za izgradnju Spring aplikacija s arhitekturom mikroservisa. Ubrzava procese razvoja i implementacije korištenjem intuitivnih zadanih postavki za testove jedinica i integracije. Štoviše, Spring Boot pomaže programerima da izgrade robusne aplikacije s jasnim i sigurnim konfiguracijama bez trošenja puno vremena i truda na otkrivanje zamršenosti Springa. Unatoč mnogim prednostima Spring Boot-a, on još uvijek ima nekoliko nedostataka koje treba imati na umu, npr. nedostatak kontrole. Spring Boot stvara mnogo neiskorištenih ovisnosti, što rezultira velikom datotekom za implementaciju. Složen je i dugotrajan proces pretvaranja naslijeđenog ili postojećeg Spring projekta u Spring Boot aplikaciju. Nije prikladan za velike projekte. Iako je odličan za rad s mikrouslugama, mnogi programeri tvrde da Spring Boot nije prikladan za izgradnju monolitnih (slojevitih) aplikacija.

# 12. LITERATURA

1. Stackify, <https://stackify.com/what-is-spring-boot/>
2. TutorialsPoint, <https://www.tutorialspoint.com/spring_boot/spring_boot_introduction.htm>
3. IBM, <https://www.ibm.com/cloud/learn/java-spring-boot>
4. Spring Initializr, <https://start.spring.io/>
5. PhoenixNap, <https://phoenixnap.com/kb/install-maven-windows>
6. Online Tutorials Point, <https://www.onlinetutorialspoint.com/gradle/how-to-install-gradle-on-windows-10.html>
7. Baeldung, <https://www.baeldung.com/spring-bean>
8. Spring, <https://spring.io/guides/gs/spring-boot/>
9. Youtube, <https://www.youtube.com/watch?v=RfCWg5ay5B0>