УНИВЕРЗИТЕТ У БЕОГРАДУ

ФАКУЛТЕТ ОРГАНИЗАЦИОНИХ НАУКА

ЗАВРШНИ РАД

Тема: Развој веб апликације електронске књижаре применом Јава технологије

Ментор: Студент:

доц. др. Душан Савић Растко Митровић 172/2015

Београд, 2021. Године

Садржај

[1. Увод 5](#_Toc67845193)

[2. Преглед коришћених технологија 6](#_Toc67845194)

[2.1. Java 7](#_Toc67845195)

[2.2. Kotlin 9](#_Toc67845196)

[2.3. Spring 10](#_Toc67845197)

[2.4. Spring Boot 12](#_Toc67845198)

[2.5. Java Persistence API (JPA) 13](#_Toc67845199)

[2.5.1. Анотације 14](#_Toc67845200)

[2.6. Hibernate 14](#_Toc67845201)

[2.7. JSP стране и JSTL библиотека 15](#_Toc67845202)

[2.8. Maven 16](#_Toc67845203)

[3. Студијски пример 17](#_Toc67845204)

[4. Вербални опис 18](#_Toc67845205)

[5. Случајеви коришћења 19](#_Toc67845206)

Попис слика

[Слика 1: Јава Технологија 7](#_Toc67845229)

[Слика 2: Јава платформа 8](#_Toc67845230)

[Слика 3: MVC патерн применом Spring-а 11](#_Toc67845231)

[Слика 4:JPA 13](#_Toc67845232)

Попис табела

**No table of figures entries found.**

# Увод

Са развојем технологије и све већом употребом исте расла је и потреба за развојем квалитетног и употребљивог софтвера. У почетку су најзаступљеније биле десктоп апликације које су се користиле у локалном окружењу без могућности контакта са другим корисницима. Са развојем и популаризацијом интернета у фокус улазе потребе као што су умрежавање са другим корисницима и пружање различитих услуга путем веба. Све ово довело је до експанзије развоја веб апликација и алата који се користе за пројектовање истих. Овај тип апликација се у великом броју случаја развија применом трослојне архитектуре у којој претпостављамо да је наша апликација састављена од три „слоја“ а то су:

1. Слој података – овде се налазе сви подаци које апликација користи.
2. Апликациони слој – овде се извршава читава логика веб апликације.
3. Презентациони слој – суштински представља веб претраживач и оно што корисник види.

Сваки слој апликације има специфичне технологије и алате који олакшавају његову имплементацију. Временом су се од ова три слоја и раздвојила два типа софтверског инжењерства а то су инжењерство база података, „Backend“ инжењерство и „Frontend“ инжењерство.

Замисао веб апликације књижаре је да омогући корисницима овог система лако поручивање књига које им стижу на адресу као и администраторима система лако управљање садржајем система.

У овом раду прећићемо све слојеве описане веб књижаре и описати технологије и алате који су коришћењи за њену имплементацију. За конкретно документовање самог софтвера коришћена је упрошћена Ларманова метода.

# Преглед коришћених технологија

У овом поглављу дат је опис технологија и алата који су коришћени у имплементацији описане веб књижаре.

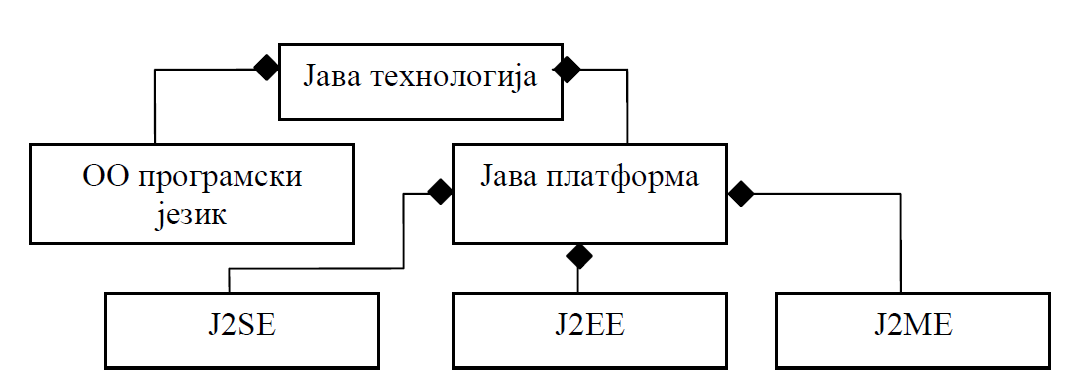
## Java

Под Јавом се може сматрати програмски језик или платформа, то јест рачунарско окружење на ком се извршавају програми написани у Јава програмском језику. Ову платформу је развио Џејмс Гослинг са колегама из „Sun Microsystems“ фирме почетком деведесетих година.

Јава се за разлику од других програмских језика не компајлира у машински код или интерпретира од изворног кода током извршавања већ се компајлира у Јава бајткод који се потом извршава на Јава Виртуелној Машини(JVM). Ово омогућава да се код испише и компајлира једном а након тога извршава билогде без потребе прилагођавања специфичном оперативном систему или хардверу на ком се извршава.

Јава платформа постоји у три верзије развојног окружења:

1. J2SE(Standard Edition)
2. J2EE(Enterprise Edition)
3. J2ME(Micro Edition)



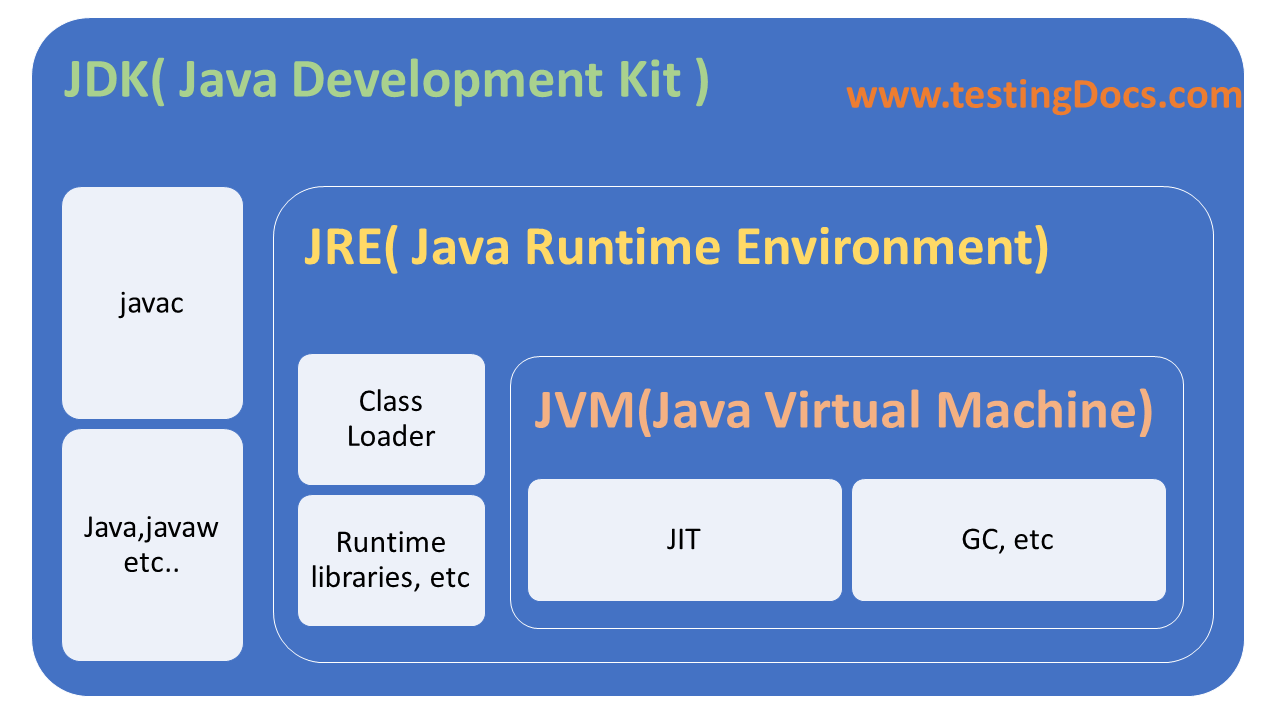
Слика 1: Јава Технологија

У оквиру овог рада коришћена је J2EE едиција која се састоји од више технологија које се класификују у три групе:

1. Веб технологије:
   1. Java Servlet
   2. Java Server Pages(JSP)
   3. Java Server Pages Standard Tag Library(JSTL)
   4. Java Server Faces (JSF)
   5. Интернационализација и локализација Веб апликација
2. Enterprise JavaBeans(EJB) технологија:
   1. Session beans
   2. Entity beans
   3. Message-driven beans
3. Java XML технологија[[1]](#footnote-1)

Јава платформа се такође састоји од три компоненте:

1. Java Virtual Machine (JVM) - Виртуелна машина која претвара Јава бајткод у машински код за извршавање.
2. Java Runtime Environment (JRE) – Садржи минималне конфигурације за извршавање Јава апликације. Сасјтоји се од Јава виртуелне машине, основних класа и библиотека.
3. JDK (Java Development Kit) – Састоји се од JRE окружења, компајлера и свих осталих неопходних компонента за изградњу и покретања Јава апликација.



Слика 2: Јава платформа[[2]](#footnote-2)

## Kotlin

Котлин је програмски језик развијен у оквиру JetBrains корпорације и у употреби је од 2016. године. Намењен је за употребу као објектно оријентисани и као функционални програмски језик. Представља званично подржани програмски језик од стране компаније Google за развој Android апликација. Котлин код се компајлира у Јава бајткод и због тога се може извршавати на Јава виртуелној машини. Употребом овог језика убрзава се развој апликација и смањује број грешака у коду зато што омогућава да се код који је написан у Јави напише у значајно краћој нотацији.

## Spring

Spring је апликациони оквир који олакшава креирање Java Enterprise апликација. Поред Јаве подржава и Kotlin и Groovy програмске језике. Настао је 2003. године као одговор на велику комплексност J2EE спецификације.

Он се интегрише само са одређеним спецификацијама J2EE платформе као што су на пример:

* Servlet API
* WebSocket API
* Concurrency Utilities
* JSON Binding API
* Bean Validation
* JPA
* JMS

Неке од основних технологија Spring оквира који су коришћени у оквиру овог рад су:

* Inversion of Control (IoC) – Представља дефиницију Java Bean–ова (Јава класа које су потребне) и других Bean–ova који су им неопходни за извршавање својих операција. Овим путем се креира Application Context у којем ће се налазити сви објекти који су неопходни апликацији за извршавање. [[3]](#footnote-3)
* Dependency Injection – Процес повезивања објеката са другим објектима који су им неопходни.
* Spring Data JPA – Олакшава вршење упита над базом података применом JPA API – ја.[[4]](#footnote-4)
* Spring Security – Пружа неопходне имплементација за валидацију корисника и обезбеђивање приступа одређеним деловима система.[[5]](#footnote-5)

Овај оквир омогућава имплементацију MVC патерна креирањем контролера који служе за дефинисање понашања система и генерисање приказа за одговарајућу URL путању.



Слика 3: MVC патерн применом Spring-а[[6]](#footnote-6)

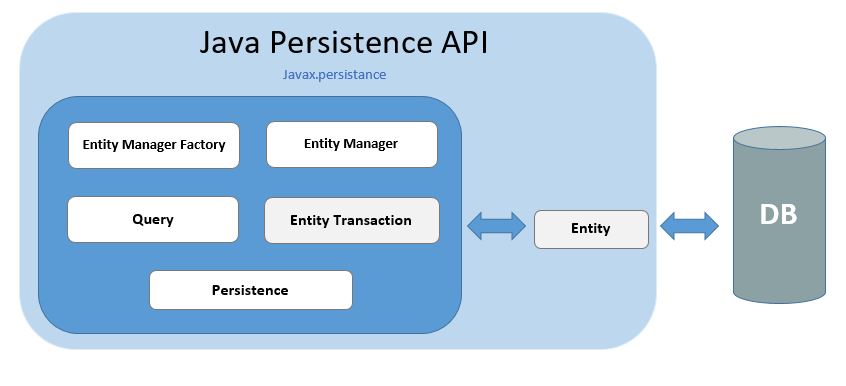
У оквиру апликације дефинишу се контролери који дефинишу понашање система које се окида позивањем одговарајућег URL-а. Довољно је да се на класу контролера стави Spring анотација @Controller и потом да се одговарајуће методе анотирају са @RequestMapping, @GetMapping, @PutMapping, @PostMapping или неком другом антоацијом како би дефинисали понашање за одређену HTTP методу на URL-у. У оквиру ове методе, употребом Model објекта, могу се убацивати подаци који су неопходни на приказу. Front Controller је контролер који се не дефинише већ постоји у самом Spring оквиру. Његова је улога да приликом пријема HTTP захтева нађе одговарајући контролер који има имплементацију понашања тог захтева. Контролер на крају треба да врати приказ који може бити обична HTML страна или се могу користити друге технологије као што су JSP, Apache Freemarker Template. Уколико су контролери имплементирани као REST веб сервис код њих би тај приказ представљао неки JSON, XML или документ неког другог типа.

## Spring Boot

Spring Boot олакшава креирање Spring апликација. Настао је након увиђања да велики део развоја апликације се троши на основна подешавања апликације. Омогућава нам да бирамо које све технологије желимо да користимо у оквиру апликације. Све ове и остале опције бирају се у оквиру сајта <https://start.spring.io/> који потом креира већ конфигурисани Spring пројекат. У оквиру овог пројекта налази се и application.properties фајл у који се могу уносити додатна подешавања која су потребна.[[7]](#footnote-7)

## Java Persistence API (JPA)

JPA представља спецификацију која омогућава да се табеле из базе података мапирају на класе у оквиру Јава програмског језика ко. Ове класе се често називају ентитетима. Он такође даје спецификацију интерфејса и основних операција над базом података и тако скраћује време потребно за имплементацију операција над базом податка.



Слика 4:JPA[[8]](#footnote-8)

Са слике се уочавају следећи елементи:

* Persistence – Садржи три статичке методе, две за добијање EntityManagerFactory објекта и једна за добијање PersistanceUtil интерфејса.
* EntityManagerFactory – Интерфејс који може да креира и одржава EntityManager објекте.
* EntityManager – Интерфејс намењен за извршавање свих операција над ентитетом. Такође креира и одржава Query објекте.
* EntityTransaction – Интерфејс који управља трансакцијама.
* Query – Интерфејс намењен контроли извршавања упита.
* Entity – Објекат перзистенције који се чува у табели базе података.

Ентитет може бити у једном од четири стања:

* New (Transient) – Ентитет је креиран ал није још увек сачуван у оквиру базе података.
* Persistent (Managed) – Стање након чувања у бази.
* Detached – Ентитет више није асоциран са перзистентним контекстом.
* Removed – Ентитет је обрисан из перзистентног контекста и базе података.

Постоји више имплементација ове спецификације а најпопуларнија, која је коришћена у овом раду, је Hibernate.[[9]](#footnote-9)

### Анотације

Све горе наведене функционалности JPA оквира постижемо употребом одговарајућих анотација које се потом интерпретирају и на основу њих се генеришу сви неопходни упити над базом података.

Неке од анотација коришћених у овом раду су:

* @Entity – Означава да ће обележеха класа представљати ентитет мапиран на одговарајућу табелу базе података.
* @Table – Спецификује назив табеле података ентитета.
* @Id – Обележава параметар класе као примарни кључ табеле и ентитета
* @Column – Мапира параметар класе на одговарајућу колону у оквиру табеле базе података.
* @ManyToMany, @ManyToOne, @OneToMany, @OneToOne – Користи се како би се успоставила асоцијација са другим ентитетима то јест табелама базе података.
* @JoinColumn – Представља колону по којој се одговарајући ентитет то јест табела спаја са другим ентитетом то јест табелом путем спољнег кључа.

## Hibernate

Hibernate представља ORM(Object-Relational Mapper) имлементацију Java Persistence API-ја који олакшава пресликавање табела базе података на Јава класе које се користе у раду апликације, конкретно у манипулацији базе података. Основне карактеристике ове имплементације су:

* Перформансе – Значајно боље перформансе од коришћења чистог JDBC кода, имплементација Fetch стратегија, не захтева додатне табеле или поља у оквиру базе и генерише већи део SQL упита приликом покретања система а не током извршавања.
* Скалабилност – Дизајниран је за рад у серверским кластерима како би омогућио висок ниво скалабилности.
* Поузданост – Стабилност и квалитет су му основне особине које су доказане употребом од стране хиљаде Јава програмера.
* Конфигурабилност – Лако се надограђује и конфигурише у скоро свим пословним процесима.[[10]](#footnote-10)

## JSP стране и JSTL библиотека

JSP представља екстензију Servlet-а и састоји се од HTML кода и JSTL тагова. Пошто се ове стране у позадини претварају у Servlet-е постоји могућност уметања чистог Јава кода, што није препоручиво пошто JSTL тагови решавају већину проблема на које се наилази у креирању ових страна. Значајно олакшавају приступ подацима који су смештени у Model објекту. Употребом JSTL тагова олакшавамо приступ и операције над овим подацима као и манипулацију самих HTML елемената на страници као што су на пример креирање форми, итерација елемената у листи и др.

## Maven

Са растом развоја софтвера расла је и комплекснот праћена неопходних пакета који су потребни у компајлирању и извршавању софтвера. Maven олакшава праћење свих ових неопходних пакета а такође и пружа додатне могућности креирања различитих конфигурација покретања апликације и праћења њене верзије. Све неопходне пакете и конфигурације дефинишу се у оквиру POM.xml фајла.

# Студијски пример

# Вербални опис

# Случајеви коришћења

1. Синиша Влајић, Душан Савић, Војислав Станојевић, Илија Антоновић, Милош Милић. Пројектовање софтвера – напредне јава технологије. Београд: Факултет организационих наука; 2008. [↑](#footnote-ref-1)
2. Преузето са сајта testingdocs.com: <https://www.testingdocs.com/java-platform-overview/>, посећен 27.03.2021 [↑](#footnote-ref-2)
3. Spring: https://docs.spring.io/spring-framework/docs/5.1.0.RELEASE/spring-framework-reference/ [↑](#footnote-ref-3)
4. Spring Data JPA: https://docs.spring.io/spring-data/jpa/docs/current/reference/html/#repositories.core-concepts [↑](#footnote-ref-4)
5. Spring Security: https://docs.spring.io/spring-security/site/docs/current/reference/html5/ [↑](#footnote-ref-5)
6. Преузето са: <https://www.javatpoint.com/spring-mvc-tutorial>, посећен 28.03.2021 [↑](#footnote-ref-6)
7. Spring Boot: https://spring.io/projects/spring-boot [↑](#footnote-ref-7)
8. Преузето са: <https://isd-soft.com/nl/tech_blog/behold-almighty-orm/>, посећено 28.03.2021 [↑](#footnote-ref-8)
9. JPA: https://isd-soft.com/nl/tech\_blog/behold-almighty-orm/ [↑](#footnote-ref-9)
10. Hibernate: https://hibernate.org/orm/ [↑](#footnote-ref-10)