

Универзитет у Београду
Електротехнички факултет



ДИПЛОМСКИ РАД

**Имплементација веб апликације за
продају фото артикала**

Ментор:

Доц. др Марија Пунт

Студент:

Магдалена Станковић 2016/0065

Београд, 2020.

Садржај

1. Увод.....	2
2. Преглед коришћених технологија.....	3
2.1. Програмски језик Јава	3
2.2. <i>Java</i> платформа	4
2.3. Apache Maven алат.....	5
2.4. Model view controller (MVC).....	5
2.5. JavaServer Faces технологија	6
2.6. HTML и CSS.....	7
2.7. Bootstrap.....	8
2.8. Hibernate.....	8
2.9. База података - MySQL Workbench	9
2.10. Развојно окружење NetBeans.....	10
3. Опис задатка и имплементација решења	11
3.1. Опис задатка.....	11
3.2. Имплементација решења	13
3.2.1. Пакет DB	13
3.2.2. Пакет controllers.....	14
3.2.3. Пакет entiteti.....	17
4. Опис корисничког интерфејса	22
4.1. Главни мени и почетна страна	22
4.2. Пријављивање и регистравање корисника.....	24
4.3. Заборављена лозинка и промена лозинке	27
4.4. Администратор	29
4.5. Уметник	31
4.6. Купац.....	32
5. Закључак.....	38
6. Литература	39

1. Увод

Технолошки напредак и развој интернета утицао је на све области пословања. Због брзог развоја технологије, компаније прелазе са традиционалне методе продаје робе на електронску. Као главно средство за обављање трансакција користи се интернет. Развој веб апликација је допринео развоју удаљене продаје, јер је омогућио корисницима да у пар корака наруче жељени производ.

Тема овог дипломског рада је развој и имплементација веб апликације за продају фото артикала. Један од циљева ове апликације је да омогући купцу да на што једноставнији начин направи своју поруџбину. Такође, могући корисник система је уметник, који путем ове веб апликације, приказује своје слике. Ова апликација даје могућност новој генерацији младих уметника, да буду примећени. Рад је организован у пет целина.

У другом поглављу су детаљно описане технологије које су коришћене за развијање веб апликације. За развојно окружење је коришћен *NetBeans IDE 8.2*. У овом раду коришћене су неке готове компоненте из библиотеке *PrimeFaces*.

У трећем поглављу је дат опис самог задатка са захтевима за реализацијом веб апликације. Такође, у овом поглављу дат је преглед класа и кратак опис имплементираних метода.

У четвртном поглављу је приказана апликација из угла крајњег корисника. Приказане су све функционалности веб апликације у зависности од улоге корисника система.

У петом поглављу је приказан закључак, док се у шестом налази списак коришћене литературе.

2. Преглед коришћених технологија

У овом поглављу дат је преглед технологија које су коришћене за реализацију веб апликације. Клијентски део апликације (*front end*) развијен је коришћењем *HTML*-а и *CSS*-а, док је серверски део (*back end*) базиран на *Java* програмском језику.

2.1. Програмски језик Јава

Програмски језик *Java* виши је програмски језик опште намене који у потпуности подржава парадигму објектно-оријентисаног програмирања. Настао је као резултат компаније *Sun Microsystems, Inc.* на челу са Џејмсом Газлингом (*James Gosling*) почетком 1990-их година.

Нови програмски језик је требало да буде лак за коришћење, поуздан и независан од платформе на нивоу преведеног облика. У почетку се звао *Oak* (храст), а име *Java* добио је 1995. године. Језик је још у интензивном развоју и публикован је већи број издања, али уз задржавање могућности да се сви програми писани у претходним верзијама могу извршавати у најновијој.

Језик *Java* је дизајниран да буде независан од платформе. Са развојем језика *Java* почело је интензивно ширење *Interneta* и посебно његове услуге *World Wide Web*, организованих у такозване веб странице, са специјалних сервера до произвољних клијената широм света. Пошто би се програми извршавали на рачунару клијента, морали су да буду независни од платформе и безбедни да не направе штету клијентском рачунару. *Java* се користи и за израду самосталних програма који се називају апликације.

Програмски језик *Java* је објектно-оријентисан програмски језик, програм се састоји од објеката (*objects*) који имају нека могућа стања и понашања. Објекти са сличним особинама чине класе (*classes*).

Такође, *Java* је вишенитни програмски језик, што значи један програм може обављати више нити истовремено. То омогућава бољи одзив и понашање у реалном времену.

Још једна особина овог програмског језика је дистрибуираност. Поседује исцрпну библиотеку рутина за рад са *TCP/IP* протоколима, као што су *HTTP* и *FTP*. Јава апликације могу да приступају објектима преко мреже и преко *URL*-а, са подједнаком лакоћом као да приступају локалном систему датотека.

Језик *Java* је веома безбедан и робустан. То је неопходно пошто се класе апликације која се извршава на локалном рачунару често довлаче са удаљених серверских рачунара.

2.2. *Java* платформа

Java платформа је стандардизовано окружење за извршавање програма написаних у *Java* програмском језику. Извршавање програма на језику *Java* постиже се интерпретирањем бајткода (превод програма, ускладиштава се као низ бајтова у одговарајућу датотеку) од стране *JVM*.

Хардвер рачунара не извршава корисников програм непосредно, већ извршава програм *JVM* (Јавина виртуелна машина) за који датотека са преведеним обликом (бајткодом) корисниковог програма представља улазне податке које треба да обрађује. На овакав начин програм написан на језику *Java*, када се преведе на неком рачунару, може да се извршава на било ком рачунару на којем постоји *Јавина виртуелна машина*.

Интерпретирање бајткода је знатно спорије, од непосредног извршавања машинских наредби, због тога су у новије време *Јавине виртуелне машине* оспособљене за динамичко превођење бајткода у машински језик тренутно коришћеног рачунара у моментима пуњења класа у меморију (*Just In Time Compiler – JIT Compiler*). На овај начин покретање програма је спорије него без динамичког превођења, али је зато извршавање брже.

Java Development Kit (JDK) садржи *Java* компајлер и потпуну копију *Java Runtime Environment (JRE)*. *Java Runtime Environment (JRE)* је платформа за извршавање *Java* програма. Састоји се од *JVM* и помоћних класа које омогућавају успешно извршавање програма.

Постоје три издања *Java* платформе:

- *Java Platform Standard Edition (Java SE)* је развојно окружење које нуди *Java Application Programming Interface (API)* и алатке потребне за креирање, тестирање и извршавање *Java* програма.
- *Java Platform Enterprise Edition (Java EE)* је развојно окружење слично *Java SE*, али за разлику од *Јаве SE* поседује *API* који нуди могућност развоја софтвера коришћењем технологија попут *Java Servlet*, *JSF*, *JSP*, *EJB*... У веб апликацији смо користили *Java EE* (*Java Enterprise Edition*) платформу.

- *Java Platform Micro Edition (Java ME)* је развојно окружење које омогућује израду апликација за не тољко комплексне уређаје као што су мобилни телефони, телевизори, *PDA* уређаји.

2.3. Apache Maven алат

Систем је креиран уз помоћ *Apache Maven* алата којим се аутоматизује процес израде софтвера. Овај алат развијен је од стране *Apache Software Foundation*.

Maven алат аутоматизује процес компајлирања, дефинисања зависности (*dependencies*) које дати софтвер користи. Конфигурација се налазу у *XML* фајлу у коме описује софтверски пројекат који се гради, његове зависности од спољних модула и компонената, редослед израде и потребне додатке.

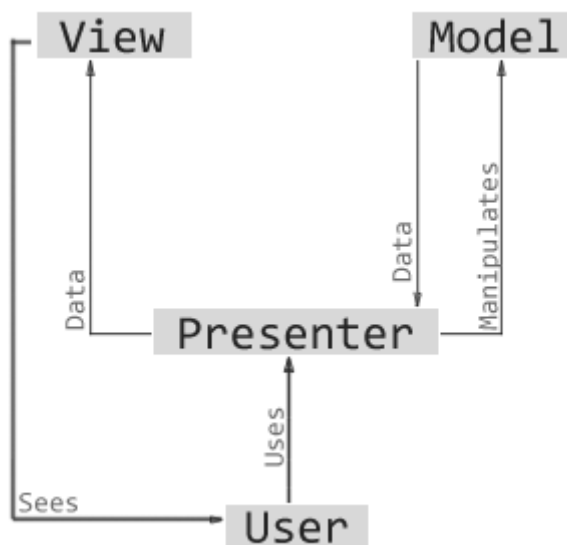
Maven омогућује да у изградњи увек учествују праве верзије *Java* библиотека, тако што их аутоматски скида из једног или више репозиторијума, као што је *Maven 2 Central Repository* и смешта их у локални кеш. *Maven* креира локални репозиторијум на диску са потребним *jar*-овима тако да приликом покретања пројекат, све библиотеке које су наведене у *XML* фајлу тражи локално, а ако их не пронађе преузима их са централног репозиторијума на интернету.

2.4. Model view controller (MVC)

MVC архитектура је пројектни узорак (*patern*) који се обично користи за развој корисничких интерфејса. Заснива се на поновној употреби већ постојећег софтверског кода, олакшавању развоја и каснијем одржавању апликационог софтвера раздвајањем на посебне компоненте. Свака компонента је задужена за обављање специфичних функција. *MVC* архитектура је приказана на слици 1.

Разликујемо три компоненте, које су међусобно повезане:

- Модел – централни део апликације. Обухвата променљиву структуру података, директно управљање подацима, логиком и правилима апликације.
- Поглед података (*view*) – приказ података у корисничком окружењу као што су дијаграми. Исти подаци се могу приказати на више начина.
- Контролор (*controller*) – улазне податке претвара у команде које управљају моделом.



Слика 1: MVC архитектура

Основна предност *MVC* архитектуре је што раздвајањем у посебне целине, код великих пројеката, на коме ради више особа, омогућава се лака измена неког елемента, као и поновно коришћење већ направљених елемената. [4]

2.5. JavaServer Faces технологија

JavaServer Faces је технологија која се користи у оквиру *Java* интернет апликација. Циљ ове технологије је аутоматизација и једноставност извршавања најкоришћенијих операција израде ове врсте апликација.

Главна одлика ове технологије је да се састоји од следећих делова:

- скуп уграђених улазно-излазних компоненти
- догађајима вођен (*event driven*) програмски модел, са опцијама за израду и реаговање на догађаје
- модел компоненти који омогућава програмерима серверске стране развој додатних компоненти [1]

JSF садржи сав потребан код за обраду догађаја и организацију компоненти. Апликативни програмери могу да занемаре све непотребне детаље и да се усредсреде на сам развој апликационе логике.

JSF је данас део *Java EE* стандарда, што значи да је укључена у сваки *Java EE* апликациони сервер, и да се може једноставно додати на Web сервере. [2]

2.6. HTML и CSS

Hyper Text Markup Language је описни језик специјално намењен опису веб страница. Помоћу њега се једноставно могу одвојити елементи као што су наслови, параграфи, цитати и слично. HTML5 је последња верзија језика HTML и овом верзијом HTML је добиона функционалности, с обзиром да су претходне верзије користиле помоћ других скрипт језика, што су били њихови главни недостаци. [9]

HTML је уведен у касним 80-им годинама. Данас је садржан у бројним стандардним описима од стране *World Wide Web* конзорцијума (W3C). Задња спецификација довршена је 1999. године. XHTML је нова, побољшана верзија HTML-а базирана на језику XML (*eXtensible Markup Language*). Оригинална формулација HTML-а има неке нерегуларности које могу узроковати проблеме код читања HTML докумената. XHTML с друге стране, користи прилично регуларну и предвидиву синтаксу.

HTML користи ознаке (тагове) да опише различите елементе интернет странице. HTML тагови се пишу унутар знакова < и >, постоје одговарајући и затварајући тагови. Отварајући таг има улогу да означи почетак неког елемента, док затварајући таг означава крај истог елемента.

HTML није довољан за креирање комплетног веб сајта и данас је нераскидиво везан за CSS (*Cascading Style Sheets*). *Cascading Style Sheets* је језик форматирања помоћу ког се дефинише изглед елемената веб страница, фонтова, боја, размака између параграфа, уређивање табела. Од верзије 4.0 HTML уведен је CSS који би дефинисао конкретан изглед, док је HTML остао у функцији дефинисања структуре и садржаја. Синтакса у CSS-у се састоји од описа изгледа елемената у документу. Опис може да дефинише изглед више елемената, и више описа може да дефинише један елемент. На тај начин се описи слажу један преко другог да би дефинисали коначни изглед одређеног елемента. CSS се може наводити на три стандардна места: директно у тагу, користећи аргумент *style*, у заглављу документа унутар тага *style* и у екстерној датотеци, која се у документ укључује тагом *link*. Трећи приступ има предност у односу на преостала два, а то је што на тај начин можемо да утичемо на елементе више докумената.

2.7. Bootstrap

Bootstrap је бесплатни веб оквир отвореног кода, за креирање веб сајтова и веб апликација. Циљ *Bootstrap* фрејмворка је олакшавање веб програмирања. Базиран је на HTML и CSS шаблонима за типографију, креирању формулара, дугмади, навигационим и осталим компонентама интерфејса, као и опционим *JavaScript* додацима.

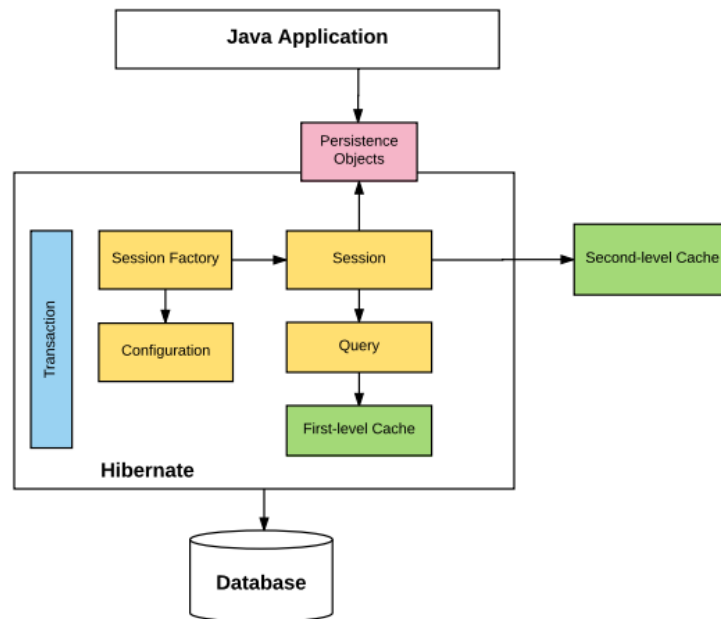
Поред основних HTML елемената, *Bootstrap* садржи и неке често коришћене елементе за формирање интерфејса. Они укључују дугмад са посебним функцијама, ознаке, напредне типографске могућности, поруке са упозорењима, прогресивне траке, и др. Компоненте су имплементиране као CSS класе, које морају бити везане за одређене елементе на веб страни.

2.8. Hibernate

Hibernate је објектно-релациони алат за пресликавање објектно оријентисаног модела домена у релациону базу података. Овај алат решава проблеме неусклађености релационих објеката, замењујући директне приступе бази података функцијама управљања објектима на високом нивоу. [6]

Његова архитектура се састоји од неколико блокова приказана је на слици 2:

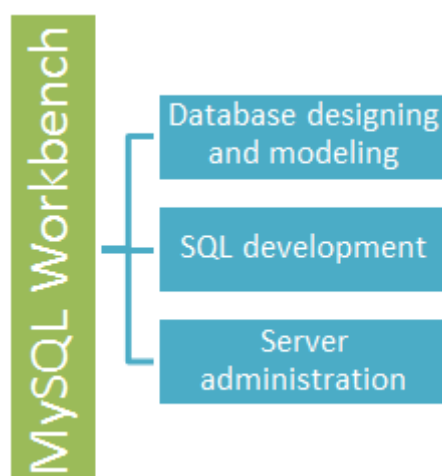
- *Configuration* – налази се у `hibernate.cfg.xml` фајлу. За Java конфигурацију, класу можете пронаћи означену са `@Configuration`. *Session Factory* за рад са Java апликацијом и базом података. [7]
- *Session Factory* – доставља објекте сесије до апликације.
- *Session* – служи за креирање и управљање сесијом.
- *Transaction* – омогућава да апликације буду портабилне међу различитим окружењима.
- *Query* – пише се помоћу SQL језика, *Criteria* упит се креира помоћу објектно-оријентисаних упита.
- *Persistent objects* – Java објекти који могу бити конфигурисани у `hibernate.cfg.xml` фајлу или означени са `@Entity`.



Слика 2: Hibernate архитектура [8]

2.9. База података - MySQL Workbench

MySQL Workbench је алатка која се користи користи се за дизајнирање и моделирање релационих база података. Омогућава стварање нових физичких модела и модификацију постојећих база података. [10] Сврха *MySQL Workbench* је пружање интерфејса за лакши и структуриранији рад са базама података. На слици 3 приказане су истакнуте карактеристике *MySQL Workbench*.



Слика 3: *MySQL Workbench* систем

Java апликација се повезује на базу помоћу *JDBC* (Java Database Connectivity) интерфејса. *JDBC* (Java Database Connectivity) је интерфејс који дефинише како клијент може приступити бази података. То је *Java* приступна технологија за приступ подацима која се користи за повезивање базе података.

2.10. Развојно окружење NetBeans

NetBeans је почео да развија 1996. године студент Roman Stanek. Од 2010. године компанија Oracle Corporation развија ово окружење. Верзија која је актуелна је верзија 12.0.

NetBeans је интегрисано развојно окружење (IDE) првенствено намењено развоју *Java* технологија, али се једнако ефикасно може користити за развој рачунарских програма и у осталим програмским језицима као што су C, C++, PHP, Фортран, Пајтон, Руби и други. Једнако добро ради на различитим платформама Windows, macOS, Linux, Solaris. Подржава различите технологије и алате који побољшавају развојни процес апликације. Намењено је за развој различитих програма. Ради на јава платформи, потребно је инсталирати JRE да би програм функционисао.

3. Опис задатка и имплементација решења

Ово поглавље се бави спецификацијом захтева за израду веб апликације која се бави онлајн продајом фото артикала. У наставку је дат преглед свих функционалности које апликација нуди.

3.1. Опис задатка

Потребно је креирати веб апликацију за онлајн продају фото артикала. За сваког корисника чувамо податке као што су име, презиме, корисничко име, лозинка, имејл, безбедоносно питање, одговор, тип корисника (администратор, уметник, купац). Администратори додају, бришу и ажурирају податке о артиклима и о тим артиклима се памти назив, цена, тренутна количина, категорија. О сликама које додају уметници памти се назив и корисничко име уметника. Купци врше поруџбине и оне садрже податке о шифри поруџбине, датум поруџбине, корисничко име купца, начин доставе (преузимање у радњи или достава на кућну адресу), адреса и списак наручених артикала.

Администратор система има увид у тек регистроване кориснике, он може да прихвати или одбије нове кориснике. Нови корисник не може да се пријави док администратор не прихвати његов захтев. Администратор има могућност да додаје нове артикле, брише постојеће и ажурира цену и количину већ додатих артикала. Приликом додавања он дефинише назив артикла, цену и количину.

Уметник може да додаје слике (дефинише назив слике), брише постојеће. Уметник може да види и да обрише само слике које је он додао.

Купац може да изабере артикал који жели да купи, након тога он бира да ли жели да дода неку слику коју је убацио уметник, да сам дода неки натпис или без принта. Уколико изабере слику или натпис има могућност да одлучи да ли жели у боји или црно-бело.

Битно је нагласити да администратор и уметник могу да купују артикле, док купац нема приступ додатним улогама које имају администратор и уметник.

Табеларни приказ могућих функција сваког корисника апликације:

- Купац

Табела 1. дужности купца

1. Пријава корисника
2. Регистрација корисника

3. Куповина производа
4. Преглед производа и брисање производа из корпе
5. Избор начина доставе
6. Преглед својих података
7. Промена лозинке
8. Заборављена лозинка

- Уметник

Табела 2. дужности уметника

1. Унос слика
2. Брисање слика
3. Пријава корисника
4. Регистрација корисника
5. Промена лозинке
6. Заборављена лозинка
7. Куповина производа
8. Преглед производа и брисање производа из корпе
9. Избор начина доставе
10. Промена лозинке
11. Заборављена лозинка

- Администратор

Табела 3. дужности администратора

1. Прихватање или одбијање нових корисничких захтева
2. Унос артикла
3. Брисање артикала
4. Ажурирање цене и количине већ датог артикла
5. Пријава корисника
6. Регистрација корисника
7. Промена лозинке
8. Заборављена лозинка

9. Куповина производа
10. Преглед производа и брисање производа из корпе
11. Избор начина доставе
12. Промена лозинке
13. Заборављена лозинка

3.2. Имплементација решења

Решење је имплементирано на програмском језику *Java*, за резвојно окружење је коришћен *NetBeans IDE 8.2*. Веб апликација је функционална уколико је развијен и обједињен и предњи део (front end) и задњи део (back end). Реализација задњег дела (back end) је подељена на три пакета:

- DB
- controllers
- entiteti

3.2.1. Пакет DB

У пакету DB налази се само једна класа *HibernateUtil.java*. Ова класа служи за добијање инстанце *SessionFactory*, која се даље користи за дохватање сесија. Имплементација класе *HibernateUtil.java* приказана је на слици 4.

```

public class HibernateUtil {

    private static final SessionFactory sessionFactory;

    static {
        try {
            Configuration cfg = new Configuration().configure("hibernate.cfg.xml");

            StandardServiceRegistryBuilder ssrb = new StandardServiceRegistryBuilder();
            ssrb.applySettings(cfg.getProperties());

            StandardServiceRegistry ssr = ssrb.build();
            sessionFactory = cfg.buildSessionFactory(ssr);
        } catch (Throwable ex) {
            // Log the exception.
            System.err.println("Initial SessionFactory creation failed." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
}

```

Слика 4: класа *HibernateUtil.java*

3.2.2. Пакет controllers

У пакету controllers се налазе четири класе:

- LoginController
- LozinkaController
- RegistrationController
- SupervizorController

Свака од ових класа представља бин (ManagedBean). Бинови се користи да би се одвојио презентациони слој од слоја логике. Многе JSF компоненте корисничког интерфејса садрже атрибут *value* којим је могуће директно специфицирати вредност или везати компоненту за вредност која се добија из поља неког бин-а.[3]

У класи **LoginController** налазе се две методе, једна метода за пријављивање корисника (public String login()), друга за одјављивање корисника система (public void logout()). У овој класи имамо и методу *kriptovanjePass(String password)* помоћу које шифрујемо лозинке наших корисника. Имплементација је приказана на слици 5. Шифровање се заснива на хеш функцији MD5.

```
public String kriptovanjePass(String password) {
    String generatedPassword = null;
    try {
        // Create MessageDigest instance for MD5
        MessageDigest md = MessageDigest.getInstance("MD5");
        //Add password bytes to digest
        md.update(password.getBytes());
        //Get the hash's bytes
        byte[] bytes = md.digest();
        //This bytes[] has bytes in decimal format;
        //Convert it to hexadecimal format
        StringBuilder sb = new StringBuilder();
        for(int i=0; i< bytes.length ;i++)
        {
            sb.append(Integer.toString((bytes[i] & 0xff) + 0x100, 16).substring(1));
        }
        //Get complete hashed password in hex format
        generatedPassword = sb.toString();
    }
    catch (NoSuchAlgorithmException e)
    {
        e.printStackTrace();
    }
}
```

Слика 5: метода за шифровање лозинке корисника

У овој класи налазе се налазе следећи атрибути:

- private String username;

- `private String password;`
- `private String tip;`

Сваки од ових атрибута мора имати своје функције за дохватање (*getter*) и постављање (*setter*), како би се компоненте корисничког интерфејса преко атрибута *value* могле везати за само поље bean-а. У зависности од тога да ли је улазна или излазна компонента, могуће је читање или упис вредности референцираног поља. У оквиру витичастих заграда и тарабе („#{ }“) у фајлу *prijava.xhtml*, директно се поставља вредност ових атрибута.

У класи **RegistrationController** налази се метода која се позива прилоком регистрације корисника (`public String registrujse()`). Такође, као и у класи `LoginController` врши се шифровање лозинке помоћу методе `kriptovanjePass(String password)` и криптована лозинка се чува у бази података. Ова класа садржи следеће атрибуте:

- `private String username;`
- `private String password;`
- `private String ime;`
- `private String prezime;`
- `private String email;`
- `private String pol;`
- `private String pitanje;`
- `private String odgovor;`
- `private String tip;`
- `private String passwordAg; //поље које се користи за потврду лозинке`
- `private String prihvacen;`

Сваки од ових атрибута има своје функције за дохватање (*getter*) и постављање (*setter*) вредности, на тај начин се у фајлу *registracija.xhtml* спецификују вредности ових атрибута. Притиском на дугме "*Registruj se*" позива се метода `registrujse()` из ове класе и уколико су сва поља прошла неопходну валидацију, вредности ових атрибута чувају се у табели *korisnik* базе података *diplomski*.

Класа **LozinkaController** садржи методе које се позивају уколико корисник жели да промени лозинку или уколико је заборавио лозинку. У овој класи налазе се следеће методе:

- `public String promeniPass();` *//позива је корисник када жели да промени лозинку*
- `public String zaboravljenaLozinka();` *//позива је корисник уколико је заборавио лозинку*
- `public String novaLozinka();` *//ова метода проверава идентитет корисника*
- `public String dodajLozinku();` *//овом методом се поставља нова лозинка*

Класа **SupervizorController** садржи све методе које позива корисник система пријављен као администратор, уметник или купац. У овој класи налазе се следеће методе:

- `public String prihvati(Korisnik kor);` *//метода којом администратор може да прихвати захтев корисника приликом регистрације*
- `public String odbij (Korisnik kor);` *//метода којом администратор може да одбије захтев корисника приликом регистрације*
- `public String dodajArtikal();` *//метода којом администратор додаје нове артикле и ажурира табелу artikal у бази података diplomski*
- `public void obrisiArtikalAdmin (Artikal artikal)` *//метода којом администратор брише означени артикал и ажурира табелу artikal у бази података diplomski*
- `public String azurirajKolicinuAdmin(Artikal artikal)` *//метода којом администратор мења цену и количину постојећег артикла и ажурира табелу artikal у бази података diplomski*
- `public String dodajSliku()` *//метода којом уметник додаје нове слике и ажурира табелу слика у бази података diplomski*
- `public String obrisiSlikuUmetnik (Slika slika)` *//метода којом уметник брише означену слику и ажурира табелу slika у бази података diplomski*
- `public void izaberiSliku(Artikal artikal)` *//метода којом купац додаје жељени артикал у корпу и прелази на страницу за избор слике, ажурира табелу korpa у бази података diplomski*
- `public void zavrshiKupvinu(Slika slika)` *//метода којом купац бира слику, додаје у корпу и завршава куповину, ажурира табелу korpa у бази података diplomski*
- `public void dodajSlikuKorpa(Slika slika)` *//метода којом купац бира слику, додаје у корпу и наставља куповину, ажурира табелу korpa у бази података diplomski*
- `public void dodajArtikalBezPrint()` *//метода којом купац додаје жељени артикал у корпу без принта и наставља куповину, ажурира табелу korpa у бази података diplomski*
- `public void dodajNatpisArtikal (Artikal artikal)` *//метода којом купац додаје артикал у корпу са изабраним натписом, ажурира табелу korpa у бази података diplomski*
- `public void pregledKorpe()` *//метода која се позива сваки пут када купац жели да погледа своју корпу*

- `public void potvrdi()` //метода коју позива купац када жели да заврши куповину, ажурира табелу *narudzbina* у бази података *diplomski*
- `public void obrisi(Korpa korpa)` //метода коју позива купац када жели да обрише селектовани артикал из своје корпе, ажурира табелу *artikal* у бази података *diplomski*
- `public void dostavaNacin(String nacinDostave)` //метода коју позива купац када бира начин преузимања своје поруџбине, ажурира табелу *dostava* у бази података *diplomski*
- `public void dodajAdresu()` //метода коју позива купац уколико је изабран као начин преузимања своје поруџбине доставу на кућну адресу, ажурира табелу *dostava* у бази података *diplomski*
- `public void pregledStavki()` //метода коју позивају претходне две методе ради прегледа изабраних ставки
- `public void kraj()` //метода којом се потврђује поруџбина

3.2.3. Пакет **entiteti**

У пакету *entiteti* налазе се следеће класе:

- *Artikal*
- *Dostava*
- *Korisnik*
- *Korpa*
- *Narudzbina*
- *Slika*

Класе у овом пакету су анотиране као ентитет и мапирани на одговарајућу табелу у бази. Ове класе не садрже никакве методе за манипулисање њеним атрибутима. Мапирање самих класа се врши помоћу атрибута *@Entity*. На сликама 6, 7, 8, 9, 10 и 11 приказани су атрибути класа *Artikal*, *Slika*, *Dostava*, *Korisnik*, *Korpa* и *Narudzbina*.

Атрибути класе **Artikal**:

```
@Entity
public class Artikal {
    @Id
    @Column(name = "slika")
    private String slika;

    @Column(name = "kategorija")
    private String kategorija;

    @Column(name = "kolicina")
    private int kolicina;

    @Column(name = "naziv")
    private String naziv;

    @Column(name = "cena")
    private float cena;
```

Слика 6: Атрибути класе Artikal

Атрибути класе **Slika**:

```
@Entity
public class Slika {
    @Id
    @Column(name = "slika")
    private String slika;

    @Column(name = "naziv")
    private String naziv;

    @Column(name = "umetnik")
    private String umetnik;
```

Слика 7: Атрибути класе Slika

Атрибути класе **Dostava**:

```
@Entity
public class Dostava {
    @Id
    @Column(name = "sifra")
    private int sifra;

    @Column(name = "nacin")
    private String nacin;

    @Column(name = "ulica")
    private String ulica;

    @Column(name = "broj")
    private String broj;

    @Column(name = "grad")
    private String grad;

    @Column(name = "telefon")
    private int telefon;

    @Column(name = "cena")
    private float cena;
```

Слика 8: Атрибути класе *Dostava*

Атрибути класе **Korisnik**:

```
@Entity
public class Korisnik {
    @Id
    @Column(name = "username")
    private String username;

    @Column(name = "password")
    private String password;

    @Column(name = "ime")
    private String ime;

    @Column(name = "prezime")
    private String prezime;

    @Column(name = "email")
    private String email;

    @Column(name = "pol")
    private String pol;

    @Column(name = "pitanje")
    private String pitanje;

    @Column(name = "odgovor")
    private String odgovor;

    @Column(name = "tip")
    private String tip;
```

Слика 9: Атрибути класе *Korisnik*

Атрибути класе **Korpa**:

```
@Entity
public class Korpa implements Serializable {

    @Id
    @Column(name = "id")
    private int id;

    @Column(name = "artikal")
    private String artikal;

    @Column(name = "slika")
    private String slika;

    @Column(name = "kupac")
    private String kupac;

    @Column(name = "stanje")
    private String stanje;

    @Column(name = "obradjeno")
    private String obradjeno;

    @Column(name = "cena")
    private float cena;

    @Column(name = "natpis")
    private String natpis;

    @Column(name = "boja")
    private String boja;
```

Слика 10: Атрибути класе Korpa

Атрибути класе **Narudzbina**:

```
@Entity
public class Narudzbina {
    @Id
    @Column(name = "id")
    private int id;

    @Column(name = "sifra")
    private int sifra;

    @Column(name = "korisnik")
    private String korisnik;

    @Column(name = "suma")
    private float suma;

    @Column(name = "artikal")
    private String artikal;

    @Column(name = "slika")
    private String slika;

    @Column(name = "natpis")
    private String natpis;

    @Column(name = "datum")
    private Date datum;
```

Слика 11: Атрибути класе *Narudzbina*

4. Опис корисничког интерфејса

У овом поглављу је описан рад веб апликације. Које функционалности су омогућене, какве могућности има корисник система у зависности од његове улоге. Веб апликација биће приказана из угла корисника система.

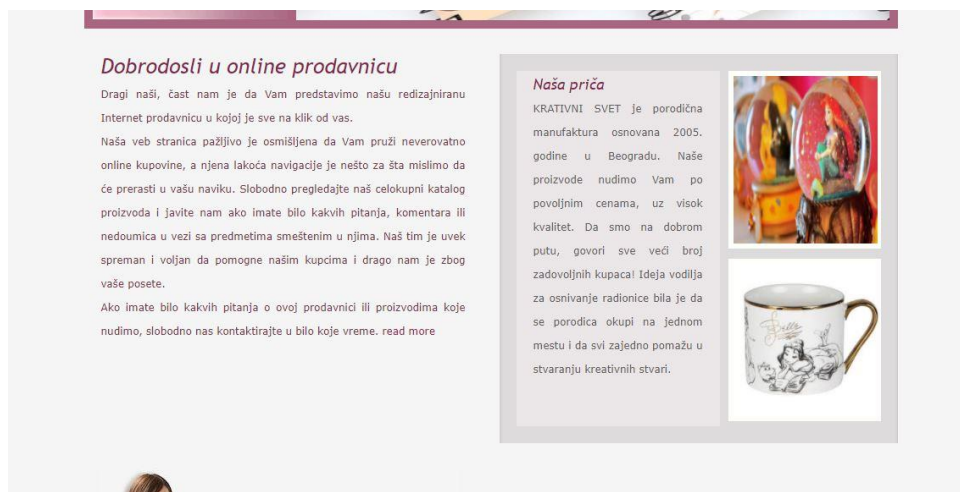
4.1. Главни мени и почетна страна

Главни мени налази се на свакој веб страници, омогућује сваком кориснику апликације независно од његове улоге да приступи страницама које су видљиве свим корисницима. Мени садржи линкове ка другим страницама, налази се на врху странице. На слици 12 налази се изглед главног менија. Притиском на неко дугме из менија, корисник прелази на истоимену страницу.



Слика 12: Главни мени веб апликације

Почетна страна веб апликације приказана је на слици 13.



Слика 13: Почетна страна веб апликације

У одељку *O Nama* налазе се информације о самој веб продавници, када је настала, како се развијала, шта је то што је разликује од других веб продавница. На веб страни *Česta Pitanja* корисник може пронаћи најчешће постављана питања и одговоре на њих. На сликама 14 и 16 приказане су веб странице *O Nama* и *Česta Pitanja*.



Слика 14: Веб страница *O Nama*



Слика 15: Веб страница *Česta Pitanja*

Веб странице *Kontakt*, *Isporuка* и *Povраћај* приказане су на сликама 16 и 17.



Слика 16: Веб страница Kontakt



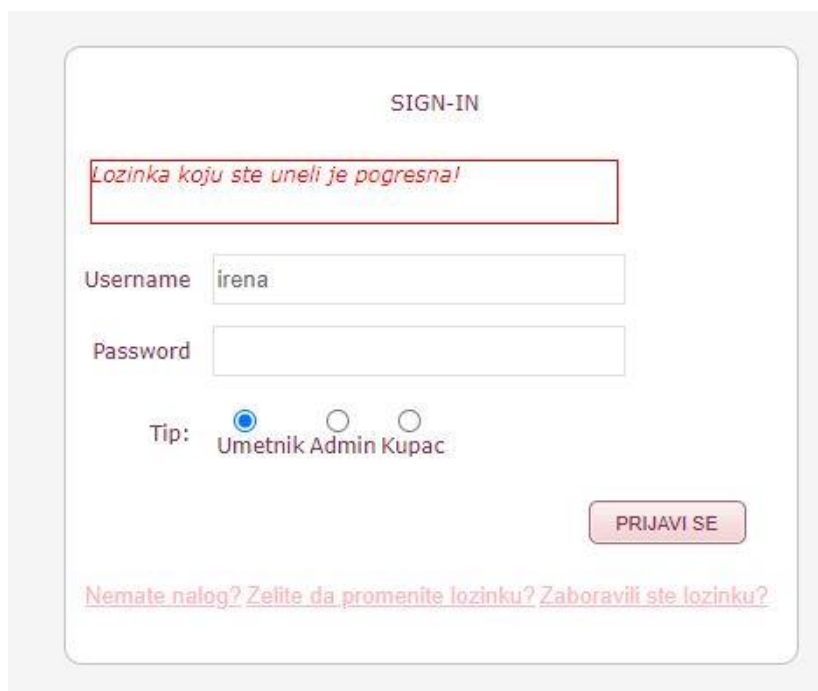
Слика 17: Веб страница Isporka I Povracaj

4.2. Пријављивање и регистровање корисника

Када корисник на почетној страни притисне линк *Sign In* (горњи десни угао), биће пребачен на страну за пријаву. На слици 18 приказана је форма за пријављивање корисника. Сваки корисник има одређену улогу, коју захтева приликом регистровања. У зависности од тога како се корисник пријави, биће приказана одговарајућа страна. Приликом пријављивања корисник уноси податке како би проверили да ли је корисник регистрован и да ли је његов захтев прихваћен. Корисник уноси своје корисничко име, лозинку и тип. Након уноса података корисник кликне на дугме за пријављивање. Уколико неко поље није попуњено приказаће се порука са одговарајућим садржајем.

Слика 18: Форма за пријаву корисника

У случају да нису унети адекватни подаци (корисник са тим корисничким именом не постоји или није исправна лозинка) и систем није пронашао корисника приказати се одговарајућа порука, то је представљено на слици 19.



The image shows a web form titled "SIGN-IN". At the top, there is a red-bordered box containing the error message: "Lozinka koju ste uneli je pogresna!". Below this, there are two input fields: "Username" with the value "irena" and "Password" which is empty. Under the password field, there is a "Tip:" label followed by three radio buttons. The first radio button is selected and labeled "Umetnik", the second is labeled "Admin", and the third is labeled "Kupac". To the right of the form is a pink button labeled "PRIJAVI SE". At the bottom of the form, there is a red link: "Nemate nalog? Zelite da promenite lozinku? Zaboravili ste lozinku?".

Слика 19: Порука за неуспешно пријављивање

Форма за регистровање корисника садржи име, презиме, корисничко име, лозинку, потврду лозинке, имејл, пол, безбедоносно питање, одговор, тип. Форма је приказана на слици 20. Као и код форме за пријављивање, уколико неко поље није попуњено приказати се порука са одговарајућим садржајем. Регистрација се завршава кликом на дугме *REGISTRUJ SE*.

REGISTRUJ SE

Ime:

Prezime:

Username:

Email:

Lozinka:

Potvrdi lozinku:

Pol: ☐ M ☐ Z

Pitanje:

Odgovor:

Tip: ☒ Umetnik ☐ Admin ☐ Kupac

REGISTRUJ SE

Слика 20: Форма за регистровање корисника

Уколико корисник са наведеним корисничким именом већ постоји у систему, лозинка није одговарајуће дужине, или се не поклапају лозинке приказаше се одговарајућа порука, то је приказано на слици 21.

REGISTRUJ SE

Korisnik sa ovim korisnickim imenom vec postoji.

Ime: Mila

Prezime: Milic

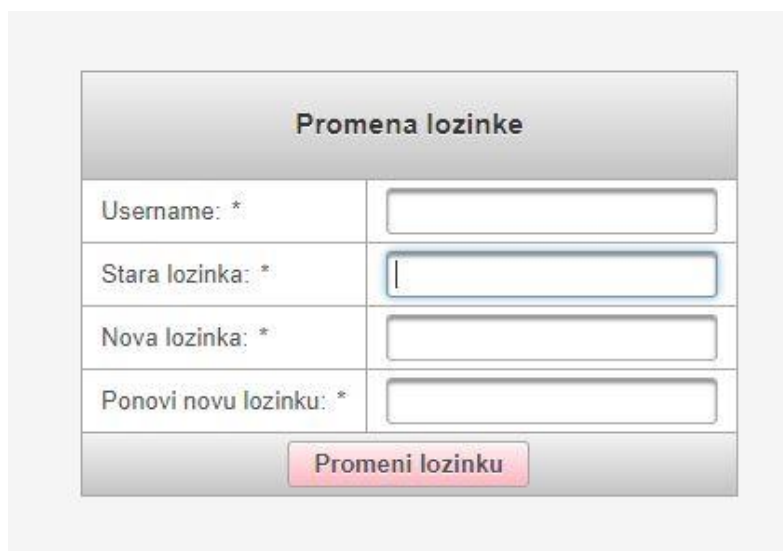
Username: mila

Email: mila@gmail.com

Слика 21: Порука за неуспешно регистровање

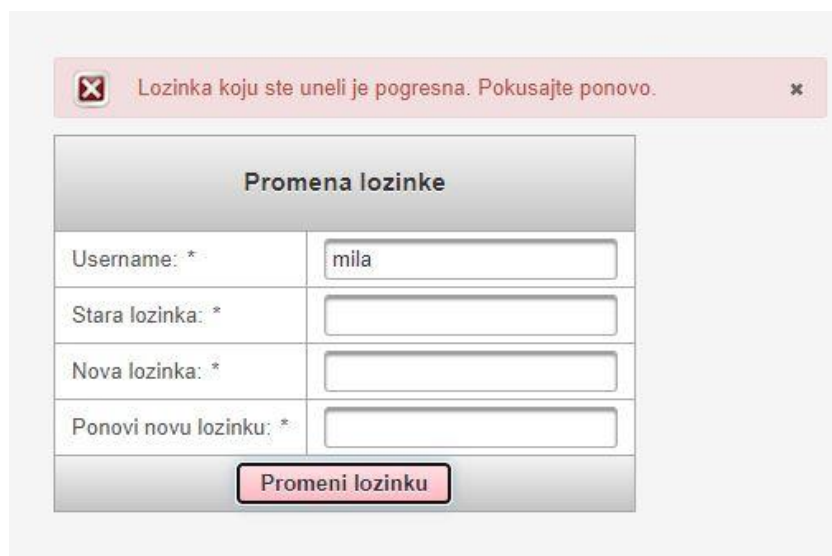
4.3. Заборављена лозинка и промена лозинке

Испод форме за пријаву, налази се и линк за промену лозинке и линк за добијање нове лозинке, ако је корисник заборавио лозинку. Уколико корисник жели да промени лозинку, кликом на линк *Želite da promenite lozinku?* приказаше се страна са формом представљеној на слици 22.



Слика 22: Форма за промену лозинке [5]

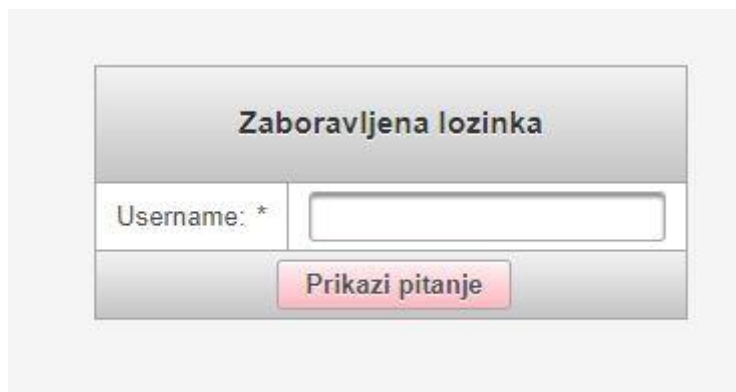
Ако неко поље није попуњено или систем не може да пронађе корисника са унетим корисничким именом и лозинком, систем ће приказати поруку, као на слици 23.



Слика 23: Порука за неуспешно мењање лозинке [5]

Добијање нове лозинке, ако је корисник заборавио лозинку, састоји се из неколико корака. Кликком на линк *Zaboravili ste lozinku?* приказује се форма за унос корисничког

имена, како би систем пронашао у бази безбедоносно питање за корисника са овим корисничким именом. Форма је приказана на слици 24.



Слика 24: Форма за унос корисничког имена
за добијања нове лозинке [5]

Када систем пронађе корисника са унетим корисничким именом, приказује се форма са безбедоносним питањем, као што је представљено на слици 25, где корисник треба да унесе одговор. Корисник има два покушаја.



Слика 25: Форма за идентификовање корисника [5]

Ако је корисник тачно одговорио на питање које је унео приликом регистравања, приказује се форма за упис нове лозинке. Форма за упис нове лозинке приказана је на слици 26.

Unesite novu lozinku

Unesite novu lozinku:

Promeni

Слика 26: Форма за упис нове лозинке

4.4. Администратор

Приликом пријављивања, када се корисник улогује као администратор прелази на нову страницу која је видљива само администраторима. Ако у систему постоје необрађени захтеви за регистрацију корисника, администратору се приказује садржај као на слици 27.

Azuriranje novih korisnika							
Ime	Prezime	Email	Pitanje	Odgovor	Tip	Prihvati	Odbij
Dusan	Peric	dusan@gmail.com	Ime oca?	Stefan	A	<input type="button" value="Prihvati"/>	<input type="button" value="Odbij"/>
Nemanja	Ilic	nemanja@gmail.com	Ime uciteljice?	Nevena	U	<input type="button" value="Prihvati"/>	<input type="button" value="Odbij"/>

Слика 27: Захтеви за регистрацију корисника

Администратор има могућност да додаје нове артикле, додајући слику артикла, назив, категорију, количину и цену. Ако изабрани артикал већ постоји у систему приказује се одговарајућа порука. На слици 28 представљена је форма за унос артикла.

Dodaj novi artikal

Dodaj sliku:
+ Choose

Naziv:

Kategorija:

Kolicina:

Cena:

Dodaj novi artikal

Слика 28: Форма за унос артикла

Администратор може да брише постојеће и ажурира цену и количину већ додатих артикала. Форма која се појављује кориснику када жели да промени количину или цену приказана је на слици 30. Приликом додавања он дефинише назив артикла, цену и количину. Слика 29 представља изглед веб странице *Azuriraj artikle*.

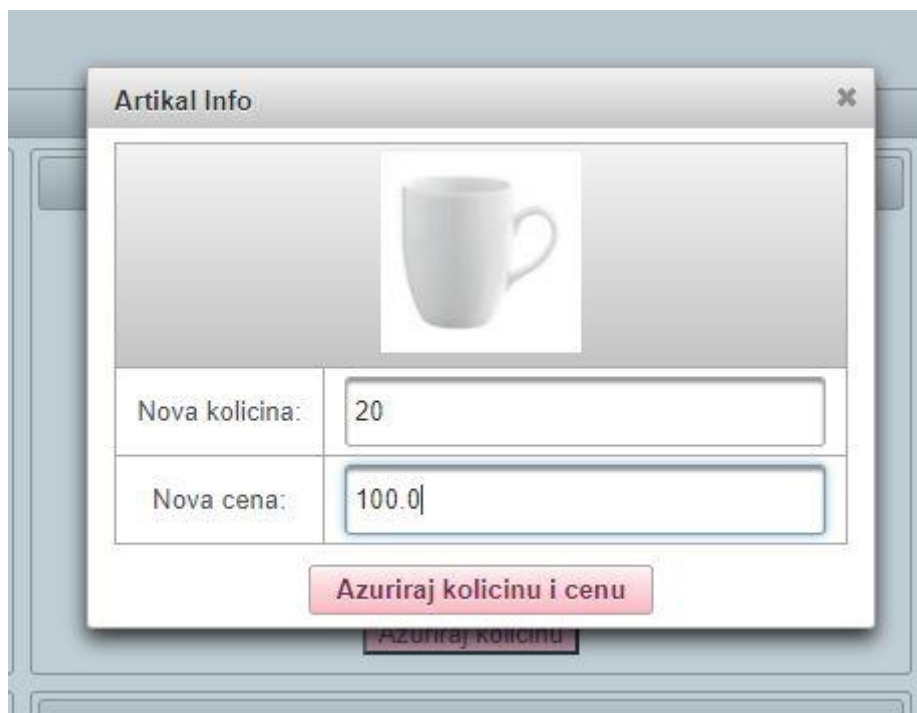
Naslovna
Azuriraj Artikle
O Nama
Kontakt
Isporuka I Povraćaj
Česta Pitanja

Foto šolja - savršen poklon za zaljubljene

Azuriranje artikala

<div style="border: 1px solid #ccc; padding: 5px; text-align: center;"> Solja3 <div style="margin-top: 5px;"> Obrisi artikal </div> <div style="display: flex; justify-content: space-between; font-size: 0.8em;"> Trenutno na stanju: 18 Trenutno na stanju: 10 </div> <div style="display: flex; justify-content: space-between; font-size: 0.8em;"> Trenutno cena: 100.0 Trenutno cena: 100.0 </div> <div style="text-align: center; margin-top: 5px;"> Azuriraj kolicinu </div> </div>	<div style="border: 1px solid #ccc; padding: 5px; text-align: center;"> Snezna kugla <div style="margin-top: 5px;"> Obrisi artikal </div> </div>	<div style="border: 1px solid #ccc; padding: 5px; text-align: center;"> Solja3 <div style="margin-top: 5px;"> Obrisi artikal </div> <div style="display: flex; justify-content: space-between; font-size: 0.8em;"> Trenutno na stanju: 18 Trenutno na stanju: 10 </div> <div style="display: flex; justify-content: space-between; font-size: 0.8em;"> Trenutno cena: 100.0 Trenutno cena: 100.0 </div> <div style="text-align: center; margin-top: 5px;"> Azuriraj kolicinu </div> </div>
--	--	--

Слика 29: Веб страница *Azuriraj artikle*



Слика 30: Форма за промену количине и цене изабраног артикла

Поред овог, администратор има исте функције као и купац.

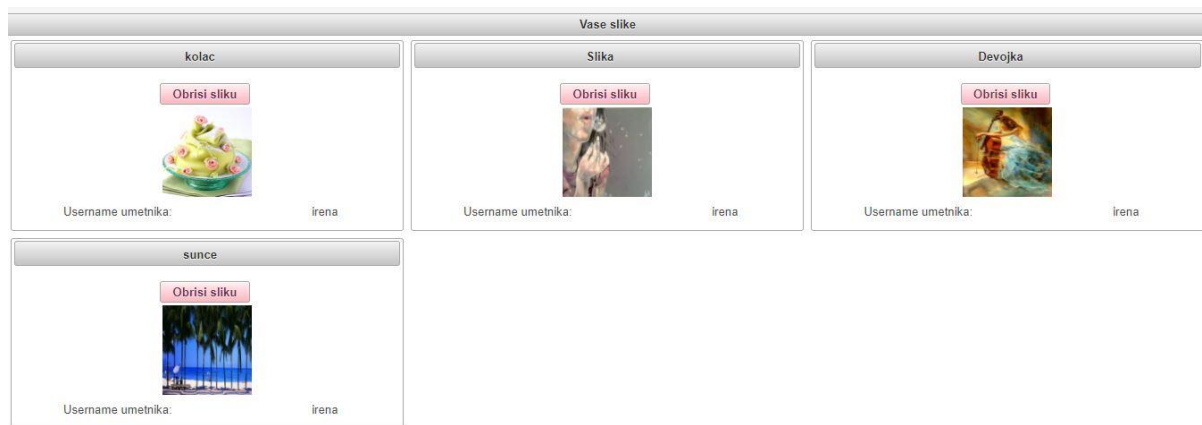
4.5. Уметник

Када се корисник система пријави као уметник, њему се приказује форма за постављање нове слике. Он уноси саму слику и назив слике, то је приказано на слици 31.



Слика 31: Форма за унос нове слике

Уметник на својој веб страници поред ове форме, види и листу слика које је он додао, приказано је на слици 32. Он може да обрише само своје слике.

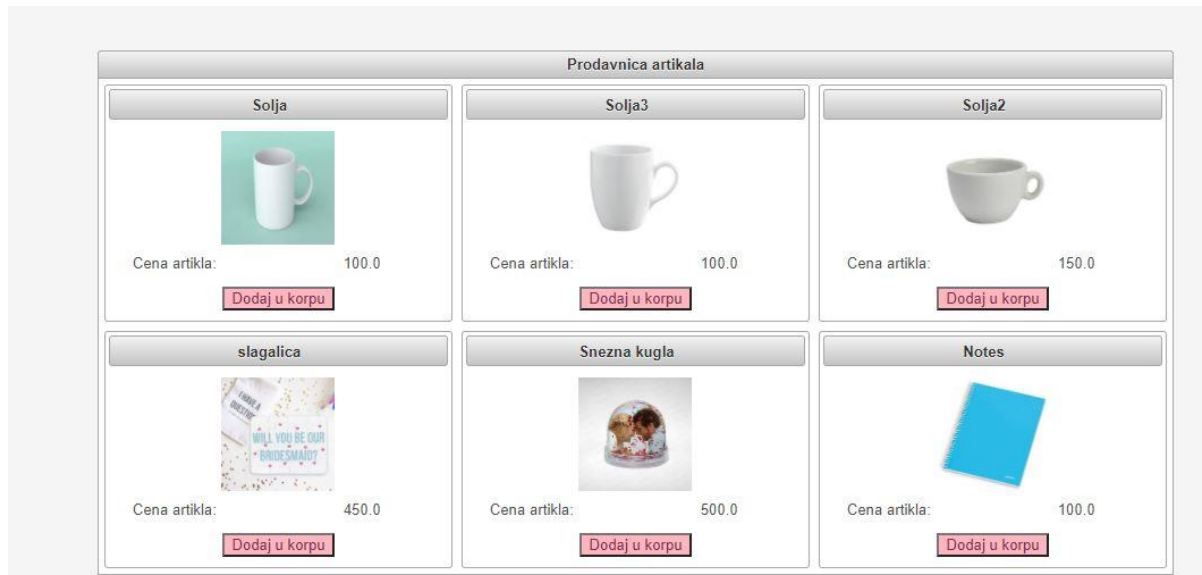


Слика 32: Приказ слика које је додао уметник

Такође, уметник има исте функције као и купац.

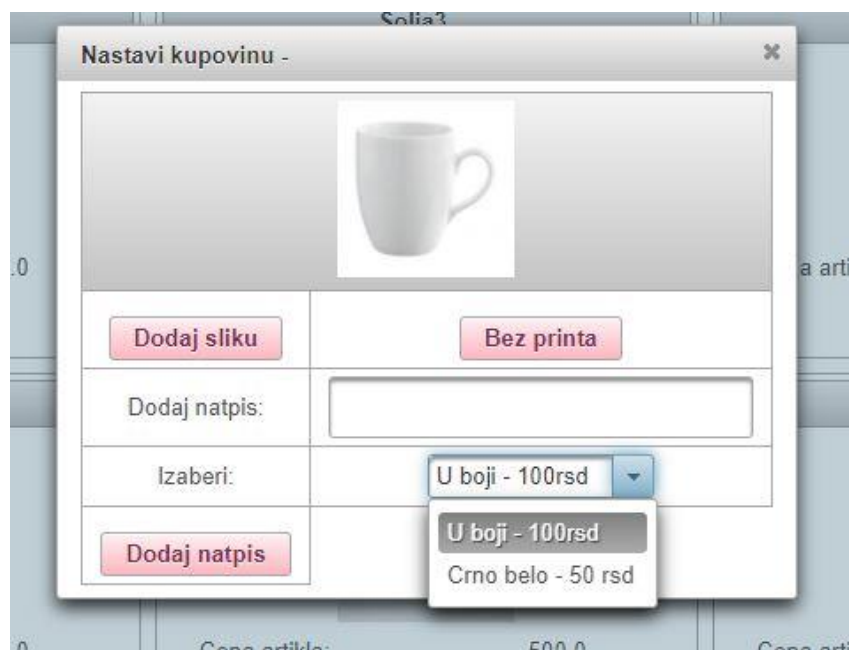
4.6. Купац

Када се корисник пријави као купац, у одељку *Online Prodavnica* може да види тренутно дотупне артикле. *Online Prodavnica* је представљена на слици 33.



Слика 33: Веб страница *Online Prodavnica* када је корисник пријављен

Купац може да изабере артикал који жели да купи кликом на дугме *Dodaj u korpu*, након тога он бира да ли жели да дода неку слику коју је убацио уметник, да сам дода неки натпис или без принта. На слици 34 приказана је форма за избор артикла.



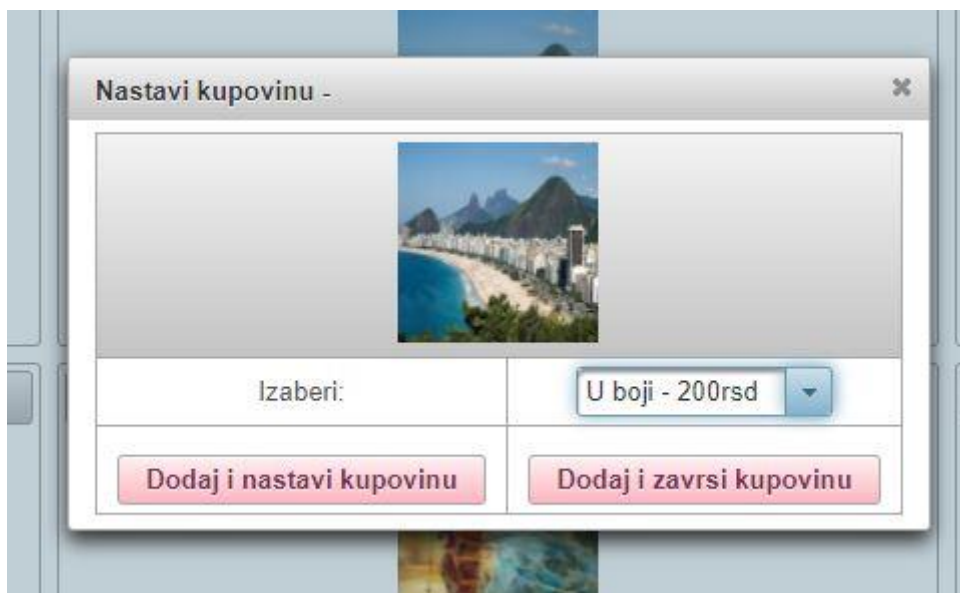
Слика 34: Приказ форме за избор артикла

Уколико купац изабере слику прелази на следећу страницу, која је приказана на слици 35.



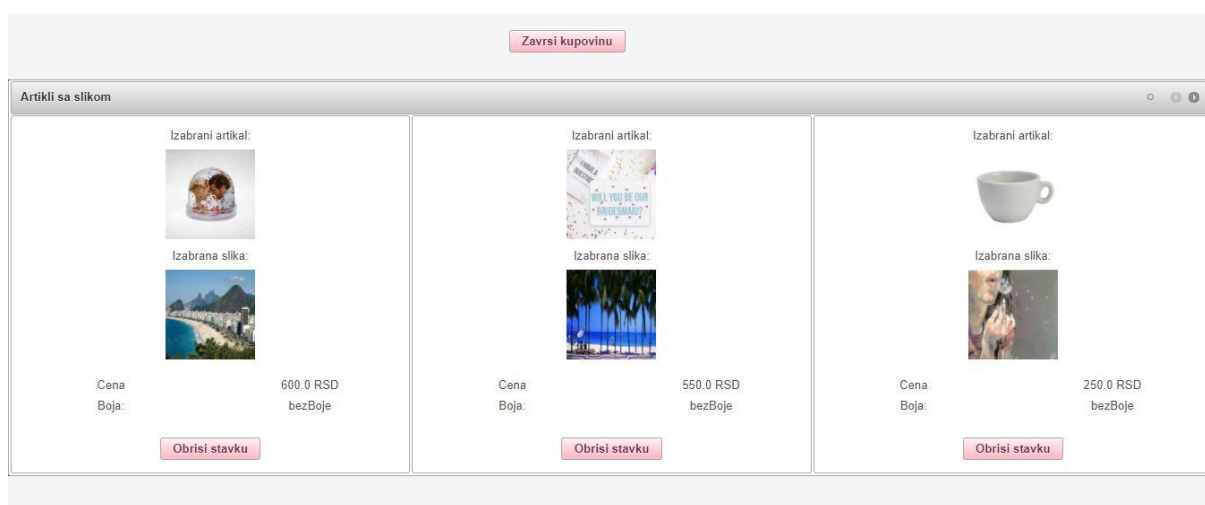
Слика 35: Приказ слика које купац може да изабере

Купац бира слику кликом на дугме *Dodaj u korpu*, након тога појављује се форма која је представљена на слици 36, где бира да ли жели да нстави куповину или да заврши куповину.



Слика 36: Приказ форме за избор слике

У горњем десном углу сваке странице налази се поље корпа, купац у сваком тренутку може да погледа артикле које је изабрао. На сликама 37 и 38 приказана је веб страна *Korpa* и артикли који се ту налазе.



Слика 37: Корпа

Слика 38: Изабрани артикал није доступан

Купац може да заврши куповину кликом на дугме *Završi kupovinu*. Купцу се тада појављује форма за избор начина преузимања купљених артикала, ова форма представљена је на слици 39.

Слика 39: Форма за избор начина доставе

Нова форма за унос жељене адресе се појављује, ако је купац изабрао доставу на кућну адресу. Форма за унос адресе представљена је на слици 40.

Dodajte adresu

Ulica:

Broj:



Grad:

Telefon:

Dodaj adresu

Слика 40: Форма за упис адресе

Купац може још једном да види производе које је изабрао и провери начин доставе и адресу коју је унео. На сликама 41 и 42 представљене су информације о поруџбини.

Naruceni artikal	
Artikal: 	
Cena	150.0 RSD
Naruceni artikal	
Artikal: 	
Cena	200.0 RSD
Ukupno: 1750.0	

Potvrdi kupovinu

Слика 41: Информације о наруџбини

Informacije o narudzbini

Narucene stavke	Nacin dostave
-----------------	---------------

Nacin dostave: Dostava na kucnu adresu

Ulica:	Vranjska
Broj:	5
Grad	Beograd
Telefon:	641233456

Potvrdi kupovinu

Слика 42: Информације о наруџбини – начин доставе

Кликом на дугме *Potvrdi kupovinu* кориснику се отвара страница приказана на слици 43.



Слика 43: Куповина је успешно извршена

5. Закључак

У овом раду приказана је реализације једне онлајн продавнице за продају фото артикала. Приказан је рад *JSF* технологије, коришћен је *Hibernate* за пресликавање података у релациону базу података. За развој предњег дела (*front end*) апликације коришћен је *HTML*, *CSS* и неке готове компоненте из библиотеке *PrimeFaces*.

Предност овако имплементиране апликације је њена преносивост и прилагодљивост независно од претраживача. Захваљујући једноставном корисничком интерфејсу, корисници система могу без потешкоћа да користе веб апликацију.

Имплементирана веб апликација може се надограђивати, то укључује интернационализацију, плаћање путем *PayPal* платформе, прослеђивање порука корисницима система из саме апликације, омогућавање корисницима система промену свих података, а не само лозинке.

6. Литература

1. Николић Бошко, Драшковић Дражен, “Програмирање интернет апликација ”, ЕТФ, Београд, 2017.
2. David Geary, Cay S. Horstmann - "Core JavaServer Faces", четврто издање, Prentice Hall
3. Програмирање Интернет апликација JavaServer Faces (JSF), приступано: септембар 2020, http://rti.etf.bg.ac.rs/rti/ir4pia/materijali/predavanja/PIA_Lekcija4_JSF.pdf
4. MVC Framework, приступано: септембар 2020, https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm
5. PrimeFaces, приступано: септембар 2020, <https://www.primefaces.org/showcase/index.xhtml>
6. Hibernate JavaDoc, приступано: септембар 2020, <https://docs.jboss.org/hibernate/stable/core/javadocs/index.html?overview-summary.html>
7. Using Hibernate in a Web Application, приступано: септембар 2020, <https://netbeans.org/kb/docs/web/hibernate-webapp.html>
8. Hibernate Arhitecture, приступано: septembar 2020, <http://www.javasafari.com/hibernate/hibernate-architecture.php>
9. Jon Duckett – “*Web & CSS Design and Build Web Sites*”, John Wiley & Sons, Inc
10. MySQL 5.7 Reference Manual, приступано: septembar 2020, <https://dev.mysql.com/doc/refman/5.7/en/datetime.html>