Универзитет у Београду

Електротехнички факултет



**Софтверски систем за евиденцију туристичких услуга**

Дипломски рад

|  |  |
| --- | --- |
| Ментор: | Кандидат: |
| проф. др Бошко Николић | Антоније Ранковић  2016/0118 |

Београд, Мај 2024.

Садржај

[Садржај 2](#_Toc169974260)

[1. Увод 3](#_Toc169974261)

[2. Функционална спецификација система 4](#_Toc169974262)

[2.1. Угоститељски захтеви 4](#_Toc169974263)

[2.2. Администраторски захтеви 5](#_Toc169974264)

[2.3. Туристички захтеви 5](#_Toc169974265)

[3. Опис коришћених технологија 6](#_Toc169974266)

[3.1. Технологије на клијентској страни апликације 7](#_Toc169974267)

[3.1.1. XHTML 7](#_Toc169974268)

[3.1.2. CSS 8](#_Toc169974269)

[3.1.3. Primefaces 8](#_Toc169974270)

[3.2. Технологије на серверској страни апликације 9](#_Toc169974271)

[3.2.1. Wildfly 9](#_Toc169974272)

[3.2.2. Аpache Maven 10](#_Toc169974273)

[3.2.3. Hibernate ORM 10](#_Toc169974274)

[3.2.4. PostgreSQL 11](#_Toc169974275)

[4. Реализација система 12](#_Toc169974276)

[4.1. Опис базе података 12](#_Toc169974277)

[4.2. Филтер ауторизације и расподела улога 15](#_Toc169974278)

[4.3. Чување лозинке 17](#_Toc169974279)

[4.4. Валидација података у оквиру форме 18](#_Toc169974280)

[4.5. Повезивање хардвера и софтвера 20](#_Toc169974281)

[5. Опис рада система 22](#_Toc169974282)

[5.1. Опис рада интернет апликације 22](#_Toc169974283)

[5.1.1. Почетна страница 22](#_Toc169974284)

[5.1.2. Прављење налога - регистрација 25](#_Toc169974285)

[5.1.3. Страница за пријављивање - логин страница 28](#_Toc169974286)

[5.1.4. Објекти који припадају систему фитнес центра 30](#_Toc169974287)

[5.1.5. Страница за остављање коментара 31](#_Toc169974288)

[5.1.6. Страница са подацим о фитнес центру 32](#_Toc169974289)

[5.2. Корисник 32](#_Toc169974290)

[5.3. Администратор 33](#_Toc169974291)

[5.4. Тренер 35](#_Toc169974292)

[5.5. Радник 36](#_Toc169974293)

[Закључак 37](#_Toc169974294)

[Литература 38](#_Toc169974295)

[Списак слика 39](#_Toc169974296)

1. Увод

У данашњем дигиталном добу, управљање туристичким услугама захтева ефикасне и поуздане софтверске системе који омогућавају прецизну евиденцију, брзу обраду података и генерисање релевантних извештаја. Туризам је једна од кључних грана привреде која захтева висок ниво организације и ефикасности како би се пружила квалитетна услуга посетиоцима, а истовремено омогућило лакше управљање и праћење података надлежним институцијама.

Овај рад се фокусира на дизајн и имплементацију софтверског система за евиденцију туристичких услуга, чија је сврха олакшавање процеса управљања информацијама о пружаоцима туристичких услуга, угоститељским објектима, корисницима смештајних услуга, као и евиденцију боравишне таксе и генерисање извештаја на основу ових података. Софтверски систем се развија као Java веб апликација, користећи савремене технологије и приступе развоју софтвера.

У наставку рада ће се детаљно описати функционалности система, архитектура система, технологије коришћене при развоју, као и кораци имплементације кључних модула система. Такође, биће анализирани изазови у развоју софтверског система за евиденцију туристичких услуга и предлози за унапређење и даљи развој система како би се задовољиле потребе савременог туристичког тржишта.

1. Функционална спецификација система
   1. Угоститељски захтеви

Угоститељ има могућност креирања налога и регистрације у систем. Угоститељи могу бити један од следећих типова: правно лице, предузетник, физичко лице и установа. Сваки тип угоститеља садржи одређени скуп података специфичан за тај тип. Након креирања налога, угоститељ може изменити и ажурирати своје податке. Да би приступио интернет апликацији и управљао својим налогом, угоститељ се прво мора пријавити у систем. По пријави, све доступне опције постају му доступне, што представља суштину интернет апликације. Опције које угоститељ поседује су: евиденција угоститељских објеката, претрагу угоститеља, претрагу угоститељских објеката, управљање захтевима за категоризацију угоститељских објеката, управљање резервацијама својих угоститељских објеката. Угоститељу је омогућен лак преглед и измена података о угоститељским објектима, као и контрола резервација угоститељских објеката. Сваки угоститељ може поседовати више угоститељских објеката, док угоститељски објекат може да припада само једном угоститељу.

Све могућности угоститеља су:

* Креира налога
* Вршење измена над својим налогом
* Пријављивање на систем
* Регистрација угоститељских објеката
* Измена података о угоститељским објектима
* Претрага угоститеља
* Претрага угоститељских објеката
* Преглед података о угоститељском објекту
* Подношење захтева за категоризацију угоститељских објеката
* Креирање резервације туристима за угоститељске објекте за које има овлашћење
* Измене и брисање већ постојећих резервација угоститељских објеката за које има овлашћење
  1. Администраторски захтеви

Администратор система има могућност да у систем додаје угоститељске објекте, претражи угоститеље и угоститељске објекте, преглед захтева за категоризацију угоститељских објеката, одобравање или одбијање захтева за категоризацију угоститељских објеката. Као и угоститељ да би администратор приступио интернет апликацији прво се мора пријавити у систем. По пријави, све доступне опције постају му доступне.

Могућности које администратор поседује су:

* Вршење измена над својим налогом
* Креирање угоститељског објекта одређеном угоститељу
* Измене и брисање угоститељских објеката
* Претрага угоститеља
* Претрага угоститељских објеката
* Преглед захтева за категоризацију
* Одобравање или одбијање захтева за категоризацију
  1. Туристички захтеви

Туристи у систему имају најмање права и најједноставнији опсег могућности. Да би приступили систему туристи морају да региструју налог и да се након тога пријаве у систем да би приступили свим опцијама које туристи поседују. Главна сврха туристичких корисника је креирање и управљање резервацијама у одређеним угоститељским објектима.

Могућности које поседују су:

* Вршење измена над својим налогом
* Претрага угоститељских објеката
* Креирање резервације у угоститељском објекту
* Преглед сопствених резервација

1. Опис коришћених технологија

У овом поглављу биће описане технологије које су коришћене за израду веб информационог система.

Сама апликација је базирана на Java платформи, намењена је Windows оперативном систему. Као развојно окружење је коришћен *IntelliJ* 2023.2.3, уз коришћење *JavaServer Faces* (*JSF*) радног оквира (*framework*). За верзионисање апликације у току развоја коришћен је *Git* као и помоћни алат *GitKraken*. [2] [3]

*IntelliJ* је интегрисано развојно окружење (IDE) које се првенствено користи за развој Јава апликација, укључујући веб, десктоп и мобилне апликације. Поред Јаве *IntelliJ*  подржава широк спектар програмских језика као што су Kotlin, Groovy, Scala, JavaScript, TypeScript, SQL и многи други. *IntelliJ* је развио JetBrains, компанија основана 2000. године у Прагу, која се фокусира на развој професионалних алата за програмере. Прва верзија *IntelliJ* објављена је 2001. године и брзо је стекла популарност због своје напредне подршке за рефакторисање кода и паметне навигације кроз код. *IntelliJ* се истиче својим напредним функцијама као што су статичка анализа кода, алати за тестирање, подршка за интеграције са верзионим системима као што су Git, Svn и Mercurial, као и могућности за брзо и ефикасно уређивање кода. Од свог оснивања *IntelliJ*  је постао један од најпопуларнијих и најмоћнијих алата за развој софтвера, широко коришћен од стране појединаца и компанија широм света. JetBrains наставља да активно развија *IntelliJ* додајући нове функције и побољшања како би подржао најсавременије праксе у развоју софтвера.

*JavaServer Faces* (*JSF*) се дефинише као радни оквир (*framework*) базиран на Јava технологијама. Првенствена употреба је дефинисање модела компонената корисничког интерфејса интернет апликација. Као свој подразумевани систем шаблонирања *JSF* користи *JSP* – *JavaServer Pages*. Прва верзија *JSF-а* је објављена 2004. године, од тада па до данас су објављиване следеће верзије: *JSF 1.1* (2004), *JSF 1.2* (2006), *JSF 2.0* (2009), *JSF 2.1* (2010), *JSF 2.2* (2013) и *JSF 2.3* (2017). Сваку нову верзију су красили новитети као што су: уклањање багова, побољшња *API-ja*, све једноставнији начин употребе, побољшавање перформанси.. Кориснички интерфејс *JSF-а* базиран је на једноставности, састоји се од форме на којој се уносе захтеви који се шаљу систему на даљу обраду тј. на серверску страну апликације. За израду овог система коришћена је верзија JSF 2.2.

*Git* је систем за контролу верзија који се широко користи за праћење промена у изворном коду током развоја софтвера. Првобитно га је развио Линус Торвалдс 2005. године за развој Линукс кернела. *Git* омогућава да више програмера раде на истом пројекту истовремено, без страха од губитка података или међусобних конфликата у коду, као и лако праћење историје развоја софтвера и измена над њим. *Git* користи дистрибуирани модел, што значи да сваки програмер има копију целокупне историје пројекта на свом локалном систему. Овај модел обезбеђује већу флексибилност и брзину јер већина операција се може обавити локално, без потребе за константним повезивањем са централним сервером. Кључне карактеристике *Git-а* укључују гранање и спајање (branching and merging), што омогућава програмерима да експериментишу са новим функцијама и идејама без угрожавања стабилне верзије пројекта. Гране се могу лако креирати, модификовати и касније спојити назад у главну грану (master branch), чиме се олакшава управљање различитим верзијама кода.

* 1. Технологије на клијентској страни апликације

Од конструктора система се очекује да клијентска апликација буде визуелно привлачна и лака за коришћење. Корисник на својој страни апликације уноси захтеве који се обрађују у реалном времену, одговор на своје захтеве треба да добије у изузетно кратком временском интервалу. Што је апликација једноставнија и на поглед лепша то ће се боље позиционирати на тржишту, тако да се доста времена и труда улаже у сам дизајн клијентског дела апликције. Технологије које су коришћене на клијентској стани апликације су: *XHTML* (*Extensible Hyper Text Markup Language*), *CSS*, библиотека *Primefaces* са подршком *Ajax-a* за освежавање самих страница и делова страница апликације. [4] [5] [6]

* + 1. XHTML

*XTHML* је језик за означавање, базиран на *XML* (*Extensible Markup Language*) описном језику, представља измeну *HTML* језика који је намењен за опис (програмирање) веб страница. Дизајниран је са циљем да замени *HTML* језик и прошири типове података са којима је могуће радити. За израду овог система коришћена је верзија *XHTML 1.0*, док постоје и верзије *XHTML 1.1*, као и *XHTML 2.0*. У наставку је приказан код за део странице која служи за пријављивање корисника – логин страна.

<h:form class=**"loginContent"**>

<h:panelGrid columns=**"3"** cellpadding=**"5"**>

<p:outputLabel value=**"Korisničko ime:"**></p:outputLabel>

<p:inputText value=**"#{loginController.username}"** id=**"korisnickoime"** required=**"true"** requiredMessage=**"Polje ne sme ostati prazno!"**/>

<p:message for=**"korisnickoime"**></p:message>

<p:outputLabel value=**"Lozinka:"** ></p:outputLabel>

<p:password value=**"#{loginController.password}"** id=**"lozinka"** required=**"true"** requiredMessage=**"Polje ne sme ostati prazno!"**/>

<p:message for=**"lozinka"**></p:message>

<h:commandButton value=**"Login"** action=**"#{loginController.login()}"** styleClass=**"indexButton"**></h:commandButton>

</h:panelGrid>

<br/>

<p:outputLabel value=**"#{loginController.message}"** style=**"color: red"**></p:outputLabel>

</h:form>

Код 1.1

* + 1. CSS

*CSS* (*Cascading Style Sheets*) је језик помоћу којег се дефинише изглед елемената веб странице. Заменио је *HTML* који се првобитно користио за дефиницију копмлетног изгледа странице. Од верзије *HTML* 4.0, *CSS* је уведен као засебна целина, док је *HTML* задржао своју првобитну функцију и користи се за структуру и садржај страница. У наставку можемо видети део кода који се користи за дизајн и улепшавање изгледа елемента *button*. [7]

**.**indexButton **{**

**background:linear-gradient(to bottom, #c62d1f 5%, #f24437 100%);**

**background-color:#c62d1f;**

**border-radius:26px;**

**border:2px solid #d02718;**

**display:inline-block;**

**cursor:pointer;**

**color:#ffffff;**

**font-family:Arial;**

**font-size:17px;**

**padding:15px 62px;**

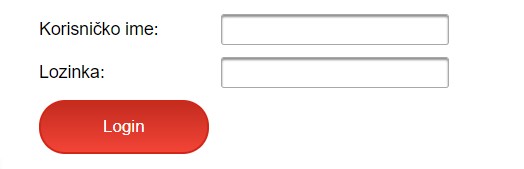
**text-decoration:none;**

**}**

Код 2.1

* + 1. Primefaces

Представља библиотеку са компонентама које служе за дизајн корисничког интерфејса интернет апликација које се користе у комбинацији са *JavaServer Faces* технологијом. Прва верзија ове библиотеке је објављена 2009. године од стране компаније *PrimeTek Informatics*. Последња верзија, као и верзија која је коришћена при изради овог система је верзија *Primefaces 8.0*. која је објављена у марту 2020. године.



Слика 1 Изглед Primefaces компоненти

* 1. Технологије на серверској страни апликације

Захтеви који су послати преко корисничког интерфејса се обрађују на серверској страни апликације. Очекује се да сервер одговори одмах по пријему захтева. Подаци који су важни за функционисање целог система се често налазе на серверу и морају бити правилно складиштени. Због тога сваки сервер има базу података у којој се чувају сви подаци који се користе и обрађују. Након што се захтеви обраде, резултати се враћају на клијентску страну апликације. Серверски део апликације написан је у програмском језику Java, а за повезивање са базом и размену података користи се алат *Hibernate*. За базу података коришћен је *PostgreSQL*, а алат који је коришћен за креирање, одржавање и ажурирање базе података је *Dbeaver*. Обрада захтева послатих преко HTTP-a се врши помоћу *Wildfly* Server-a 24.0.1. Као build tool коришћен је *Apache Maven*.

* + 1. Wildfly

*Wildfly* је апликативни сервер отвореног кода (open source) који пружа платформу за хостовање Јава апликација. Првобитно познат као Jboss Application Server, *Wildfly* је развијен од стране Red Hat-a и нуди снажне могућности за развој, имплементацију и управљање Јава ЕЕ (Java Enterprise Edition) апликацијама. *Wildfly* је дизајниран да буде лаган и брз, пружајући високе перформансе и скалабилност. Подржава најновије стандарде Јава ЕЕ и омогућава програмерима да лако креирају сложене, високо скалабилне апликације. *Wildfly* такође укључује напредне функције као што су кластеринг, управљање трансакцијама и подршка за веб сервисе, што га чини погодним за примену у предузећима. Једна од кључних предности *Wildfly-a* је његова модуларна архитектура, која омогућава корисницима да одаберу и укључе само оне делове система који су им потребни, чиме се смањује употреба ресурса и побољшава перформансе. *Wildfly* се може лако интегрисати са различитим алатима и технологијама за развој софтвера, као што су *Maven, Gradle* и *Jenkins*, чиме се побољшава продуктивност и убрзава развојни циклус.

* + 1. Аpache Maven

*Apache Maven* је алат за управљање пројектима и аутоматизацију изградње софтвера, веома популаран у Јава заједници. Развијен од стране Apache Software Foundation, *Maven* олакшава изградњу, управљање зависностима и конфигурацију пројеката кроз централизовану POM датотеку. Кључна предност *Maven*-а је управљање зависностима помоћу репозиторијума. Локални и удаљени репозиторијуми омогућавају аутоматско преузимање и ажурирање библиотека, елиминишући потребу за ручним управљањем зависностима. Ово значајно убрзава постављање нових пројеката и одржавање конзистентности између различитих развојних окружења. *Maven*-ова модуларна архитектура омогућава проширење функционалности путем додатака (plugins), који подржавају компилацију, тестирање, паковање и распоређивање апликација. Ова флексибилност чини га моћним алатом за аутоматизацију различитих аспеката софтверског развоја. *Maven* се лако интегрише са алатима као што су Jenkins, Eclipse и IntelliJ IDEA, побољшавајући продуктивност и убрзавајући развојни процес. Подршка за континуирану интеграцију и испоруку (CI/CD) омогућава брзу и сигурну имплементацију промена у производним окружењима. *Apache Maven* је постао кључни алат за Јава програмере због своје флексибилности и једноставности употребе. Apache Software Foundation наставља да активно развија и одржава *Maven*, обезбеђујући његову релевантност у савременом софтверском развоју.

* + 1. Hibernate ORM

*Hibernate* је популарни ORM (Object-Relational Mapping) алат за Јава који омогућава лако повезивање објеката у Јави са релационим базама података. Развијен од стране Red Hat-а, *Hibernate* аутоматизује многе задатке повезане са базама података, чиме поједностављује развој апликација. Једна од главних предности *Hibernate*-а је његова способност да мапира Јава класе на табеле у бази података и да аутоматски конвертује податке између формата који користи апликација и формата који користи база података. Ова аутоматизација смањује количину кода коју програмери морају да пишу и одржавају, побољшавајући продуктивност и смањујући могућност грешака. *Hibernate* подржава и напредне функције као што су кеширање, lazy loading, и управљање трансакцијама. *Hibernate* је високо конфигурабилан и може се лако интегрисати са другим Јава фрејмворцима и алатима, као што су Spring и Java EE. Ово олакшава изградњу сложених, скалабилних апликација које могу да задовоље различите потребе корисника и пословне захтеве. Подршка за различите базе података, укључујући MySQL, PostgreSQL, Oracle, и Microsoft SQL Server, чини *Hibernate* флексибилним решењем које се може користити у различитим окружењима. Ова флексибилност, заједно са снажним функцијама и једноставношћу употребе, учинила је *Hibernate* једним од најпопуларнијих ORM алата у Јава заједници. *Hibernate* је постао суштински алат за Јава програмере захваљујући својој способности да поједностави рад са базама података и побољша продуктивност развоја. Red Hat наставља да развија и одржава *Hibernate*, обезбеђујући његову релевантност и ефикасност у савременом развоју софтвера.

* + 1. PostgreSQL

*PostgreSQL* је моћан, релациони систем за управљање базама података (RDBMS). Развијен од стране PostgreSQL Global Development Group-а, *PostgreSQL* је познат по својој стабилности, скалабилности и подршци за напредне функције. Једна од главних предности *PostgreSQL*-а је његова компатибилност са стандардом SQL, као и подршка за ACID (атомичност, конзистентност, изолација, трајност) трансакције, што осигурава висок ниво интегритета података. *PostgreSQL* такође подржава напредне типове података, као што су JSON и XML, и омогућава креирање сопствених типова и функција. Флексибилност *PostgreSQL*-а огледа се и у његовој способности да подржава различите индексе и механизме за претрагу, као што су B-tree, Hash, GiST и GIN индекси. Ова флексибилност омогућава оптимизацију перформанси за различите врсте упита и апликација. *PostgreSQL* је високо конфигурабилан и проширив, омогућавајући корисницима да додају нове функције путем додатака (plugins) и екстензија. Подршка за репликацију и хеадерпад партиционисање чини га идеалним за велике и сложене системе. Због своје поузданости и богатства функција, *PostgreSQL* се користи у разним индустријама, од финансија и здравства до телекомуникација и владиних служби. Активна заједница корисника и развојних инжењера обезбеђује континуирани развој и подршку за овај робусни систем за управљање базама података.

1. Реализација система

Веб информациони систем описан у овом документу је подељен на две апликације:

* Интернет апликација
* Системска апликација
  1. Опис базе података

Целокупна база података је креирана у алату *MySQL Workbench*. База података садржи следеће табеле: *korisnik, ugostitelj, preduzetnik, fizickolice, pravnolice, ustanova, ugostiteljskiobjekat, zahtevzakategorizaciju, sanduce, poruka, rezervacija, turista, privilegovanikorisnik.*

A screenshot of a computer

Description automatically generated

Слика 2 Изглед базе података

Табела *korisnik* садржи информације о корисницима система, укључујући привилеговане кориснике, угоститеље и туристе. Колоне ове табеле укључују: *id, username, email* и *password*. Ове колоне представљају основне податке потребне за регистрацију сваког корисника у систему. Приликом регистрације, корисничко име (колона *username*) мора бити јединствена за сваког корисника, како би се обезбедила уникатност. Колона *password* чува лозинку коју је корисник унео, али ради заштите личних података, лозинка се не чува у изворном облику. Уместо тога, она се криптује помоћу Java класе BCrypt пре него што се унесе у базу података.

Табела *ugostitelj* представља табелу која садржи табелу која садржи додатне податке о угоститељима. Угоститељ представља један од три типа корисника у систему, поред туриста и привилегованих корисника. Сваки угоститељ има своје сандуче са порукама и са табелом *sanduce* повезан је на основу *id*-a угоститеља. Постоје четири врсте угоститеља: *preduzetnik, ustanova, pravno lice* и *fizicko lice*. Са тим табелама табела *ugostitelj* је повезана на основу *id*-a угоститеља.

Табела *preduzetnik* чува информације корисника угоститеља који су предузетници. Колоне *ime, prezime, adresa* и *delatnost* садрже личне податке предузетника.

Табела *ustanova* чува информације корисника угоститеља који представљају установе. Колоне *naziv, adresa* и *vrstaustanove* садрже податке о установама. У колони *vrstaustanove* чувају се подаци о врсти установе.

Табела *pravnolice* чува информације корисника угоститеља који представљају правна лица. Колоне *nazivfirme, adresafirme* и *pib* садрже податке о правним лицима. Колона *pib* представља порески идентификациони број правног лица, који идентификује правно лице и самим тим мора бити јединствена.

Табела *fizickolice* чува информације корисника угоститеља који представљају физичка лица. Колоне *ime, prezime, adresa* и *jmbg* садрже податке о физичким лицима. Колона *jmbg* представља јединствени матични број грађана, који идентификује физичко лице и самим тим мора бити јединствена.

Табела *turista* представља табелу која садржи табелу која садржи додатне податке о туристима. Туристи су корисници система чија је главна функционалност креирање/измена сопствених резервација. Са табелом *rezervacija* повезани су на основу *id-*а туристе. Колоне *ime, prezime, datumrodjenja* и *drzavljanstvo* садрже основне информације о туристима.

Табела *privilegovanikorisnik* представља табелу која садржи табелу која садржи додатне податке о привилегованим корисницима. Колоне *ime, prezime* и *datumrodjenja* садрже основне информације о привилегованим корисницима.

Табела *sanduce* садржи информације о сандучетима угоститеља. Свако сандуче има свог угоститеља и поруке које су у том сандучету. Са табелом угоститељ повезана је на основу *id-*a угоститеља, док је са порукама повезана на основу *id-*a сандучета.

Табела *poruka* садржи информације о порукама које стижу угоститељима. Порука мора припадати сандучету и са сандучетом је повезана по *id-*ју сандучета. Једно сандуче може да садржи више порука док једна порука може припадати само једном сандучету. Колона *sadrzaj* представља садржај поруке, док колона *procitana* представља информацију да ли је порука прочитана или не.

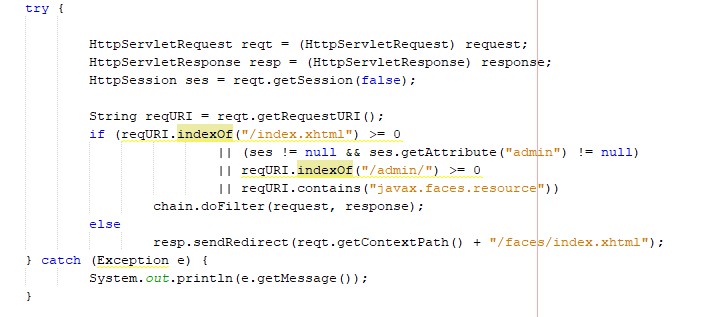
Табела *ugostiteljskiobjekat* садржи информације о угоститељским објектима. Угоститељски објекат мора припадати угоститељу и са табелом *ugostitelj* је повезана по *id-*ју угоститеља. Један угоститељ може поседовати више угоститељских објеката, док угоститељски објекат може припадати само једном угоститељу. Угоститељски објекти морају бити категоризовани и у колони *kategorizovan* се чува та информација. Датум истека категоризације угоститељског објекта се чува у колони *istekkategorizacije*, док се информација да ли је за угоститељски објекат послата порука о истеку категоризације чува у колони *notifikovanoistekukategorizacije*. Угоститељски објекат могу да креирају угоститељи, као и привилеговани корисници, али у том случају морају навести *username* угоститеља који ће бити власник угоститељског објекта.

Табела *zahtevzakategorizaciju* чува информације о захтевима за категоризацију угоститељских објеката. Захтев за категоризацију креира угоститељ, власник угоститељског објекта за који се подноси захтев за категоризацију. Захтев за категоризацију повезан је са угоститељем по *id*-ју угоститеља, а са угоститељским објектом по *id-*ју угоститељског објекта. Угоститељски објекат може имати више захтева за категоризацију, док захтев за категоризацију може припадати једном само једном угоститељском објекту. Иста веза постоји између угоститеља и захтева за категоризацију. У табели *zahtevzakategorizaciju* чувају се подаци о датуму подношења захтевa (колона *datumzahteva*), о томе да ли је захтев прегледан (колона *pregledan*), да ли је одобрен(колона *odobren*), као и о кориснику који га је прегледао(колона *korisnikpregledao*). Само привилеговани корисници имају право да доносе одлуку о захтеву за категоризацију.

* 1. Филтер ауторизације и расподела улога

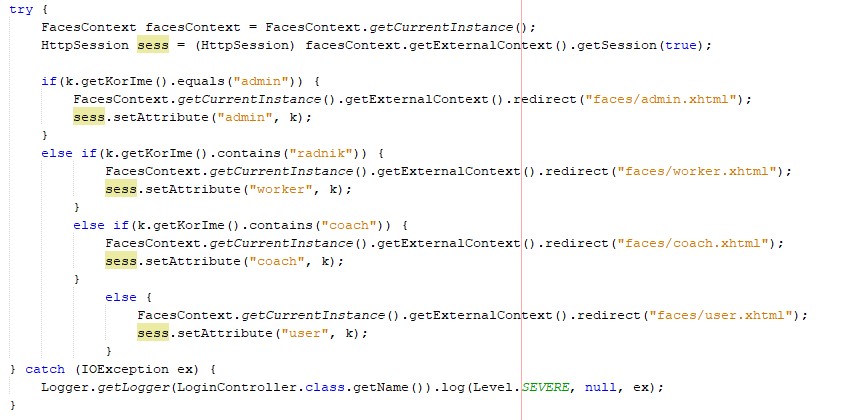
Контрола приступа заснована на атрибутима дефинише парадигму контроле приступа којом се права приступа додељују корисницима коришћењем смерница у комбинацији са атрибутима. Атрибути могу бити кориснички атрибути, атрибути ресурса, атрибути окружења, објекти. Логика атрибута се своди на логику “АКО, ОНДА” тј. АКО је подносилац захтева на пример-администратор ОНДА има могућност да врши радње за које је он овлашћен, читање и упис осетљивих података. [10]

Контрола приступа у овом систему је заснована на улогама. Улоге су унапред дефинисане и имају одређене скупове привилегија.



Слика 3 Филтер ауторизације

Проблем око додељивања улога је решен помоћу филтера ауторизације. Филтер ауторизације очекује да се током остваривања сесије у самој сесији нађе одређени атрибут на основу кога ће филтер одобрити приступ одређеним страницама или одбити приступ и пребацити корисника на почетну страницу. На слици 5. можемо видети филтер ауторизације везан за администратора. Где су администратору дата права приступа страницама које у себи имају префикс “/*admin*/”. Из HTTP (*HyperText Tansfer Protocol*) сесије се извлачи атрибут који је именован са “*admin*” у оквиру кога се налази објекат, уколико је сесија успешно остварена, корисник који је администратор ће имати приступ страницама које у оквиру своје путање (смернице – линка) поседују кључну реч “*admin*”.



Слика 4 Успешно оставаривање сесије

Приликом пријављивања на сервис корисник уноси корисничко име и лозинку. Улоге које постоје у систему су:

* Администратор
* Корисник
* Тренер
* Радник

У зависности од улога креирају се корисничка имена која се састоје од атрибута који представља кључну реч и од произвољног остатка који администратор, тренер и радник бирају приликом регистрације, на основу тога се администратору додељује кључна реч “*admin*”, тренеру “*coach*”, раднику “*worker*”, док се кориснику оставља могућност да сам дефинише своје јединствено корисничко име приликом регистрације.

Након уноса података, врши се провера улога у систему што је приказано на слици 6. На основу корисничког имена из базе података се дохвата јединствени корисник, за којег се проверава исправност лозинке. Након тога следи провера података везаних за улоге. Уколико су подаци исправни, на основу корисничког имена се одређује улога у систему и у оквиру HTTP сесије се додаје атрибут улоге који може бити: “*admin*”, “*coach*”, “*worker*” или “*user*”. Поред атрибута у сесију се додаје и објекат самог корисника са свим његовим подацима. Након успешног пријављивања на систем и остваривања сесије на ред долази филтер ауторизације који одобрава или одбија приступе одређеним деловима система које јесу или нису одобрене кориснику.

* 1. Чување лозинке

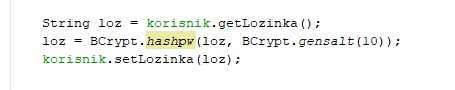
У претходном поглављу су омогућена ограничена права приступа, поставља се питање да ли лозинка корисника система може бити злоупотребљена, да ли се на неки начин може приступити бази, открити лозинка и тиме доћи до поверљивих података које корисник поседује.

У ту сврху се у овом систему користи *Java* класа *BCrypt*, која врши криптовање – хеширање лозинке, која се у криптованом облику чува у бази података. Уколико се на нелегалан начин приступи корисниковој лозинки тешко ће се моћи декриптовати и искористити зарад измене и злоупотребе корисникових података.



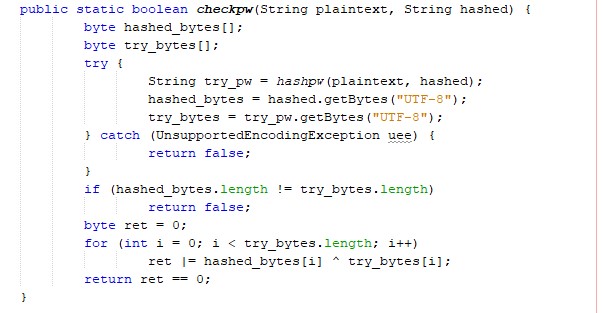
Слика 5 Метода за хеширање шифре

Метода која се користи приликом хеширања је метода *hashpw*, која као аргументе прима лозинку и *salt*. Лозинка је у изворном формату – онаква какву је корисник унео. *Salt* представља случајан податак на основу ког се врши хеширање и који представља додатну заштиту од напада као што је *Rainbow* table. Детаљнији опис метода је дат на следећем линку <https://en.wikipedia.org/wiki/Bcrypt> .



Слика 6 Коришћење методе за хеширање

Метода *hashpw* се у овом систему користи приликом регистровања корисника и то непосредно пре него што се сви подаци упишу у базу података. Корисникова лозинка пролази кроз процес хеширања и тако хеширана бива упамћена у базу података.



Слика 7 Метода checkpw

Метода *checkpw* служи да би се упоредила лозинка која је у изворном формату са хешираном лозинком. Изворни код методе *checkpw* се налази на слици 9.

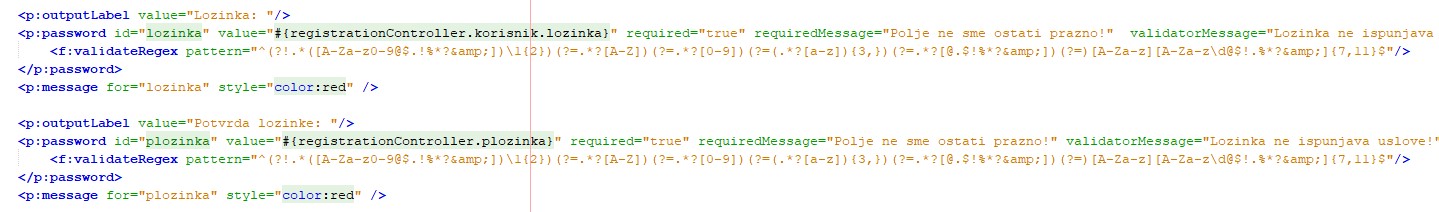


Слика 8 Коришћење методе checkpw

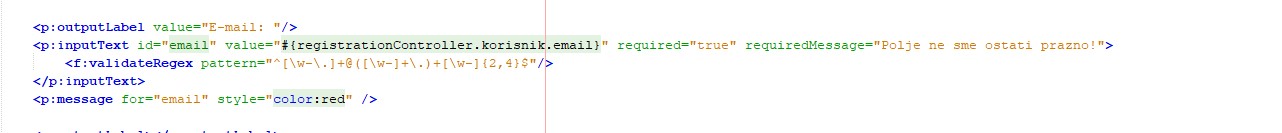
Приликом пријављивања на систем, пре него што се оствари сесија, проверава се унета лозинка са лозинком која се налази у бази података. Из базе података се као објекат на основу корисничког имена узимају сви подаци везани за корисника. Затим се из објекта извлачи лозинка у хешираном формату, позива се метода *checkpw* са аргументима лозинка и хеширана лозинка. Уколико се лозинке поклапају корисник ће бити успешно пријављен на систем.

* 1. Валидација података у оквиру форме

Пре него што се подаци унесу у базу података они морају проћи процес валидације. Процес валидације се састоји од испитивања података, тј. да ли су подаци унети у исправном формату. Приликом ових провера коришћени су регуларни изрази и валидаторске поруке. Форма која служи за унос података не може остати непопуњена, па се и о томе мора водити рачуна. [11]



**Слика 9 Коришћење регуларних израза приликом валидације лозинке**



**Слика 10 Коришћење регуларних израза за испитивање формата email-а**

Три најбитнија поља на форми за регистровање за која су коришћени регуларни изрази и валидација јесу поље за унос лозинке, потврду лозинке као и поље за унос email адресе. Ова поља морају испоштовати одређене формате који ће бити описани у наставку документа.

Регуларни израз који је коришћен за проверу валидности формата лозинке обезбеђује да се лозинка састоји од минимум осам, а максимум дванаест карактера. Лозинка треба да садржи минимум једно велико и три мала слова, да почиње великим или малим словом. Лозинка треба да садржи и минимум један нумерик, као и минимум један специјални знак. Максималан број узастопних карактера је два.

Валидација email адресе се састоји од регуларног израза који дефинише формат саме адресе. Очекивани формат је низ карактера и бројева, након којег следи специјални знак “@” након којег следи низ карактера, затим специјлани знак “.” И након тога поново низ карактера.

Осим ових валидација постоје и валидације да свако поље форме мора бити попуњено и садржати валидне податке. Кликом на дугме “Registracija” најпре се проверава попуњеност свих поља форме, након чега следи провера регуларних израза и испуњавање очекиваног формата. Уколико је све у реду корисник ће бити успешно регистрован, уколико је дошло до неке грешке корисник ће путем валидаторских порука бити обавештен о конкретној насталој грешци.

* 1. Повезивање хардвера и софтвера

Систем се састоји од Arduino хардвера са својим софтвером помоћу којег се програмира и системске и интернет апликације. Потребно је податке очитане помоћу Arduino плочице пренети у остатак система и обрадити.

Овај проблем је решен тако што су сву подаци који су очитани путем серијског улаза Arduino плочице пре одласка у систем, били унети у базу података. На овај начин је комуникација између апликативног и хардверског дела апликације остварена кроз базу података. Након уноса очитаних података у базу, апликативни део система преузима улогу и исте податке користи.

Начин на који је *Aruino* повезан са базом података се налази на слици 13. Најпре се креира конекција између базе података и *Arduino* софтвера, за пренос података се користи *localhost* и *GlassFish Server*. Након успешне конекције могуће је очитавање података са *RFID* картице. Прислањањем картице на сензор долази до очитавања података са картице. Специјални идентификатор картице се уноси у базу података. Даљи ток обраде података преузима апликативни део система.



Слика 11 Arduino код

1. Опис рада система

У наредним редовима документа ће бити представљен начин рада система кроз кратко корисничко упутство. Свака функционалност система биће описана, а уз опис ће бити приложена слика ради бољег разумевања.

* 1. Опис рада интернет апликације

У овом поглављу биће дат опис свих опција које се могу наћи на почетној страници апликације. Опције које су наведене у наставку може видети сваки корисник интернет апликације.

* + 1. Почетна страница

Након покретања апликације кориснику се приказује почетна страница веб апликације коју можемо видети на слици 14. Почетна страница представља такозвани *landing page*, без превише садржаја, корисник може да кликне на линк за пријаву и регистрацију.

A screenshot of a computer

Description automatically generated

**Слика 12 Почетна страница**

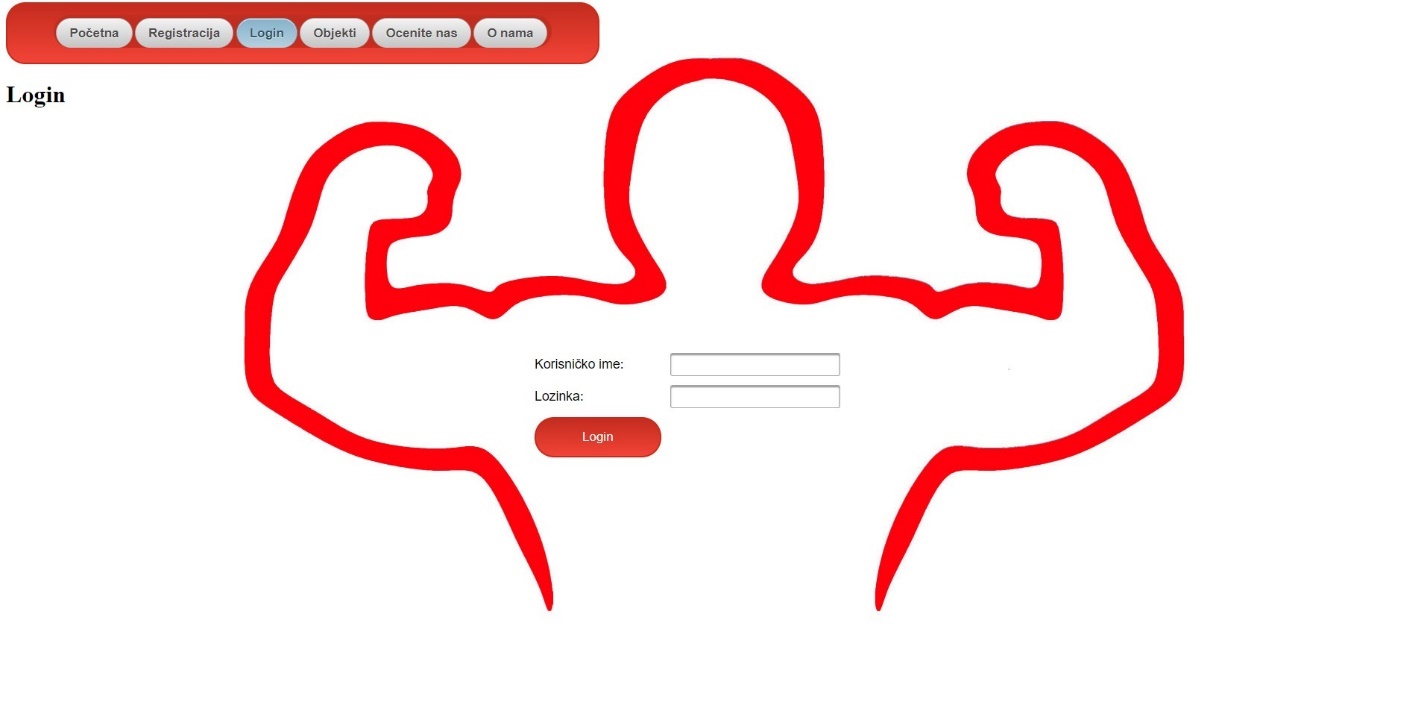
Кликом на линк *Registracija* корисник се преусмерава на страницу за регистрацију.

Променом странице мења се и URL, на пример:

* Са <http://localhost:8080/touristInfo/index.xhtml>,
* На <http://localhost:8080/touristInfo/registration.xhtml> )

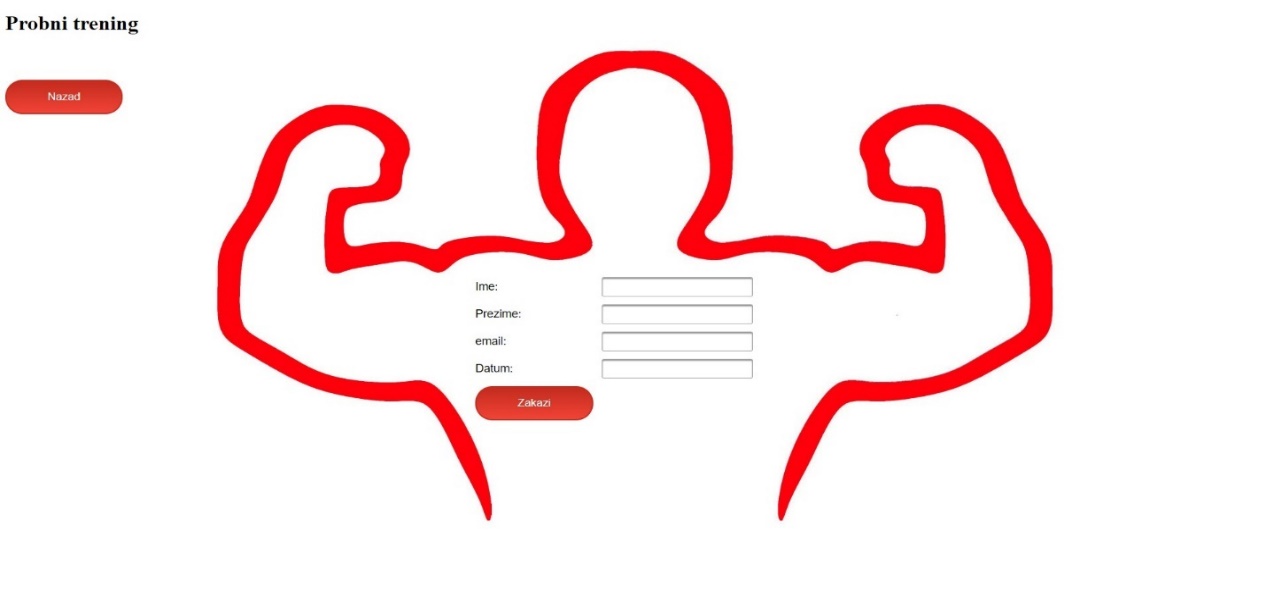
На почетној страници апликације се налази мени са опцијама које корисник бира кликом. Поред менија, на централном делу екрана се налазе три дугмета која представљају три различите опције: пробни тренинг, преглед свих тренинга и прављење налога.

Кликом на било коју опцију из менија отвара се нова страница која поред свог садржаја има и мени са почетне странице помоћу којег корисник може лакше да се креће кроз апликацију. Опција која је селектована означена је светло плавом бојом на основу које корисник зна на којој се страници трентуно налази.



**Слика 13 Скок са почетне странице на неку од опција (у овом случају логин страница)**

Кликом на дугме “Probni trening” отвара се нова страница. О страници на којој корисник заказује своје термине за пробни тренинг ће више речи бити касније.



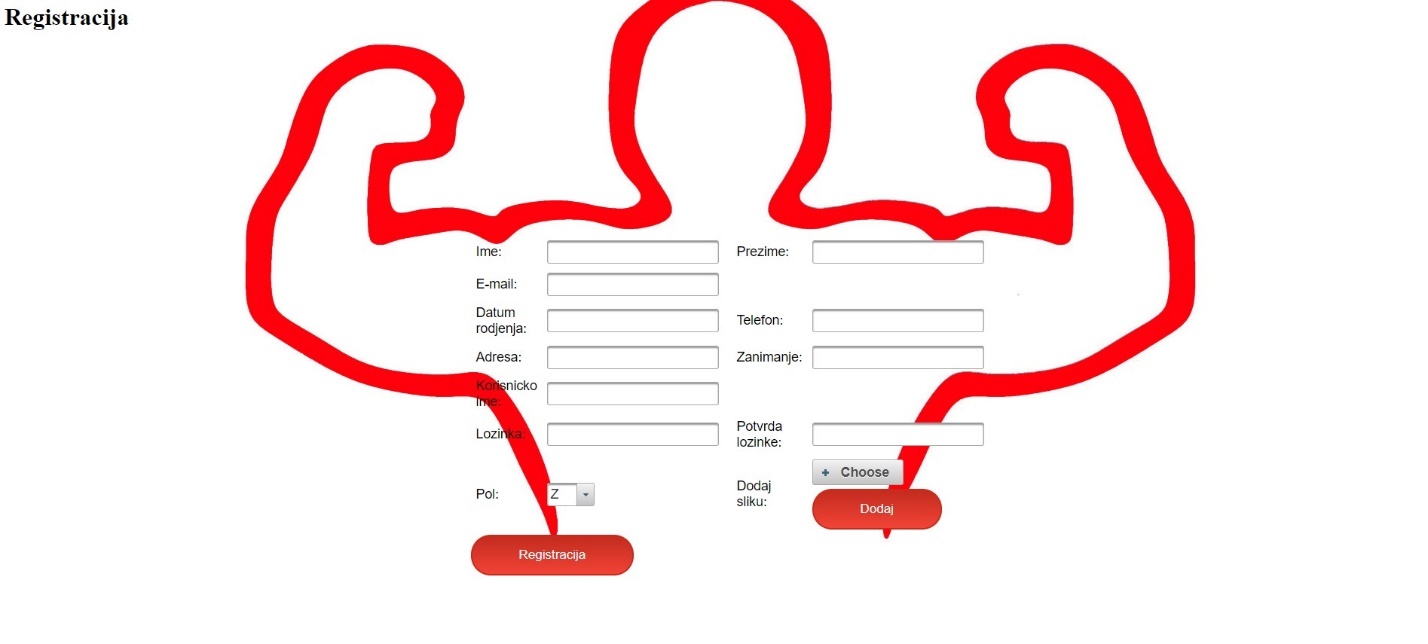
Слика 14 Изглед странице за заказивање пробног тренинга

Уколико корисник кликне на дугме “Pogledaj sve tipove treninga” на почетној страници се приказује табела у којој се налазе врсте тренинга заједно са ценама. Кликом на ову опцију корисник има увид у типове тренинга које фитнес центар нуди што је приказано на слици 17.



Слика 15 Тренинзи који су у понуди

Кликом на опцију “Napravite nalog” отвара се нова страница. Новоотворена страница представља регистрацију и прављење налога, детаљнији опис ове странице биће дат касније.



Слика 16 Страница за креирање налога

* + 1. Прављење налога - регистрација

Када корисник са почетне странице изабере опцију “*Registracija*”, отвориће се страница са формом за креирање корисничког налога. Форма је једноставна и лака за попуњавање.

A screenshot of a computer

Description automatically generated

Слика 17 Регистрација

Корисник првенствено треба да изабере који тип налога жели да региструје: туристу или угоститеља. Туриста и угоститељ имају различит скуп података који је потребан за унос при регистрацији.

A screenshot of a computer

Description automatically generated

И угоститељи и туристи морају унети основне податке за креирање налога, а сва поља која се налазе на форми морају бити попуњена и валидна пре клика на дугме *“Registruj”*. У случају да неко од поља остане празно, или поље које има додатне валидације није у коректном формату, регистрација неће бити могућа и адекватна порука ће бити приказана поред поља које није валидно.

A screenshot of a computer

Description automatically generated

Када је корисник унео сва поља и када сва поља пролазе валидације, корисник може успешно наставити даље. Кликом на дугме “*Registruj* ” корисник бива пребачен на страницу за пријављивање.

При регистрацији угоститеља, угоститељи морају изабрати тип угоститеља чији се налог региструје. Угоститељ може бити један од четири типа угоститеља а то су: Физичко лице, Правно лице, Установа или Предузетник. За сваки тип угоститеља постоји јединствен скуп података који корисник мора да унесе при регистрацији.

A screenshot of a computer

Description automatically generated

Слика 18 Изостављање садржаја поља

A screenshot of a computer

Description automatically generated

**Слика 19 Пример не испуњавања формата лозинке**

A screenshot of a computer

Description automatically generated

* + 1. Страница за пријављивање

До ове странице се може доћи на више начина: са почетне странице кликом на линк “*Prijava*”, након успешне регистрације корисника или након завршетка клијентове сесије и одјављивања са апликације. На овој страници корисник уноси своје креденцијале и у зависности од њих бива прослеђен на нову страницу. Пре уноса података корисник мора изабрати за ког типа корисника врши пријаву: туристу, угоститеља или привилегованог корисника.

A screenshot of a computer

Description automatically generated

**Слика 20 Логин страница**

На самој страници се налази мени у горњем левом углу, док се на средини налази форма са пољима за унос корисничког имена и лозинке са дугметом “*Prijava*”. Као и на страници за регистрацију поља се валидирају.

A screenshot of a computer

Description automatically generated

Уколико корисник унесе непостојеће корисничко име и валидну лозинку, кликне на дугме “*Prijava*”, апликација га обавештава на начин који можемо видети на слици 23.

A screenshot of a computer

Description automatically generated

**Слика 21 Непостојеће корисничко име**

Приликом уноса валидних података и кликом на дугме “*Prijava*”, отвара се нова страница. Нова страница садржи опције које су намењене одређеном типу корисника: туристи, угоститељу или привилегованом кориснику. Након пријаве, сваком кориснику се у горњем десном углу апликације приказује линк *“Odjava”* којим се корисници одјављују из апликације.

* 1. Угоститељ

Наредно поглавље ће се бавити описом функционалности које поседује корисник система - *user*. У пар реченица ће бити дат опис могућности које корисник поседује. На овај начин ће кориснику бити олакшано коришћење апликације.

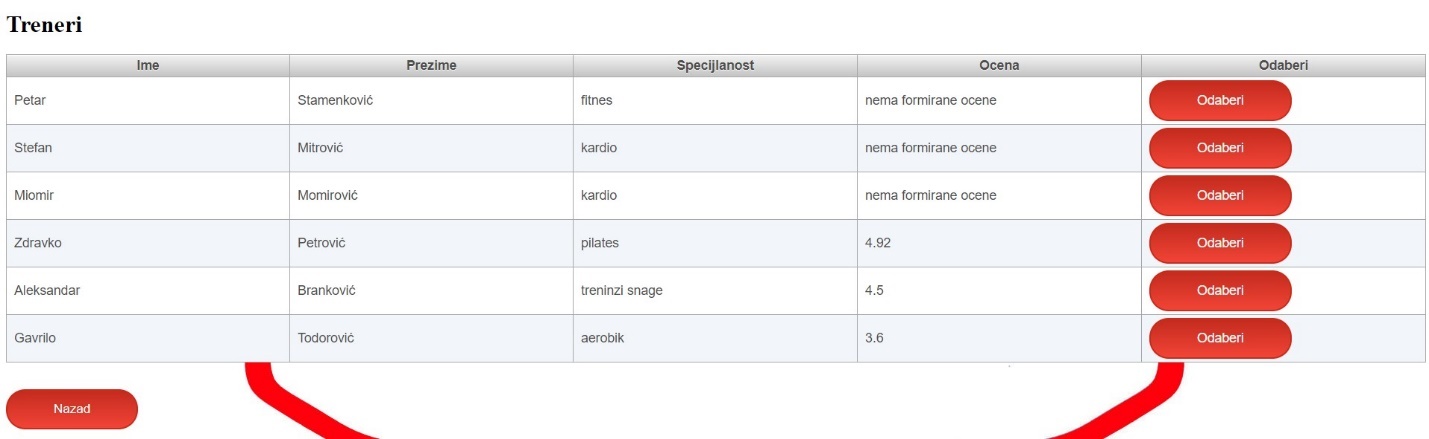
Након успешне пријаве на сервис угоститељ се преусмерава на своју страницу. Угоститељу се у горњем левом углу апликације приказује мени са свим страницама којим угоститељ има приступ у овој апликацији. Угоститељ У горњем десном углу екрана угоститељ има и линк са својим корисничким именом, као и иконицу поруке, која представља поштанско сандуче угоститеља.

A screenshot of a computer

Description automatically generated

**Слика 24 Кориснички део апликације**

Угоститељ у менију контрола има опцију Опција “Odaberi trenera” отвара нову страницу на којој је могуће бирање тренера На страници која служи за одабир тренера приказује се табела са колонама за име, презиме, специјалност и оцена тренера. Корисник на основу приказаних резултата може да изврши одабир тренера кликом на дугме “Odaberi”. Када је корисник завршио са одабиром кликом на дугме “Nazad” се враћа на корисничку страницу.



**Слика 25 . Одабир тренинга**

Кликом на опцију “Odaberi trening” кориснику се отвара нова страница на којој има могућност бира тренинг. На страници су у виду табеле представљени назив тренинга и цена. Кликом на дугме “Odaberi” у корисников налог се додаје одабрани тренинг и креира се ставка у рачуну.

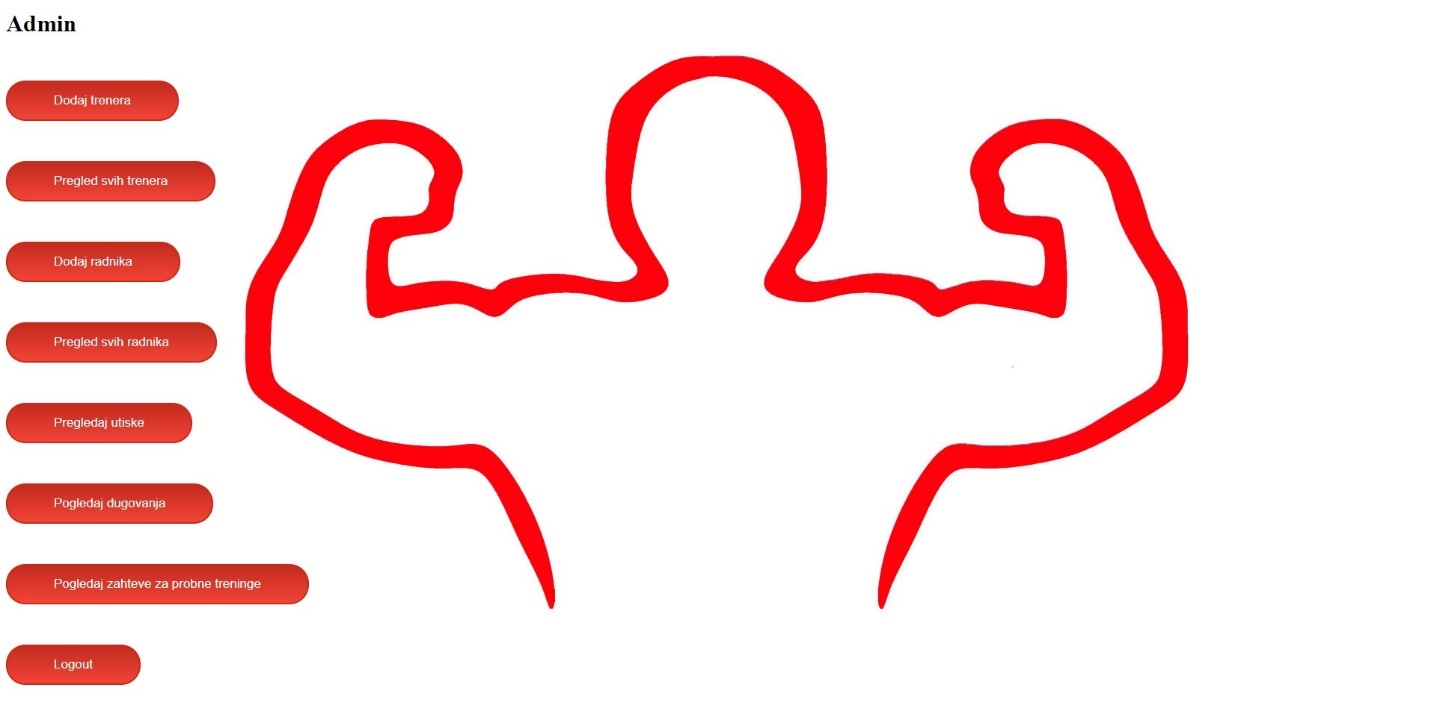
Опција “Pregled dugovanja” даје преглед корисникових дуговања. Корисник има увид о количини дуговања, опису дуга, месецу за који то дуговање важи. Кликом на дугме “Uplati” отвара се нови прозор на коме корисник бира начин плаћања и врши плаћање.

Корисник има могућност да прегледа тренере и тренинге које је изабрао кликом на дугмад “Pogledaj trenere” и “Pogledaj treninge” респективно. Корисник такође може да одустане од изабраног тренера или тренинга кликом на дугме “Odustani”.

Уколико корисник жели да промени лозинку то може урадити кликом на опцију “Promeni lozinku”. Да би корисник променио лозинку треба да унесе стару лозинку, нову лозинку и потврду нове лозинке, кликом на “Promeni” стара лозинка бива замењена новом.

* 1. Администратор

Све функционалности које поседује администратор система ће бити дате у облику теза. Свака теза ће бити објашњена са по једном реченицом.

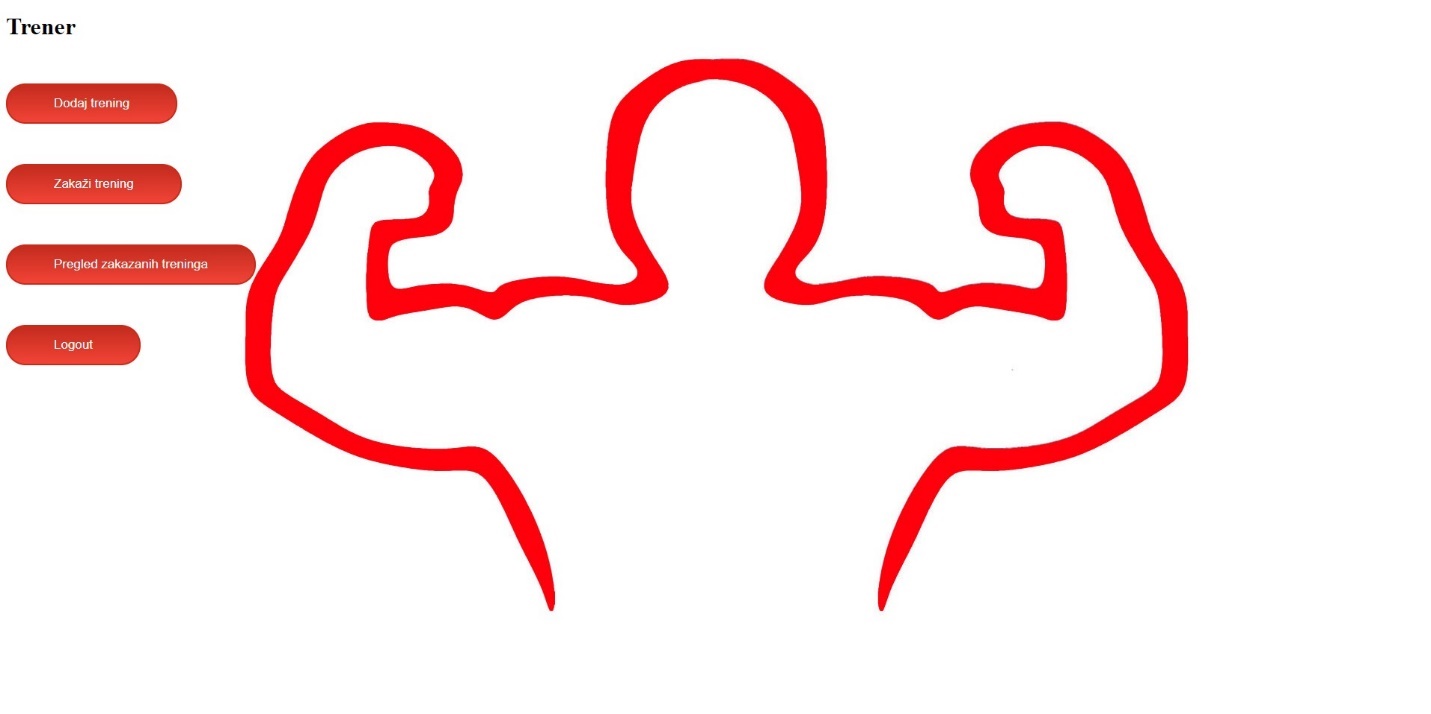


Слика 26 Страница администратора

Опције администратора система су:

* Додавање тренера – у базу података додаје тренера, тренер добија тренерска права, након овога тренер се може регистровати у систем
* Преглед свих тренера – администратор има увид у све тренере који су у систему, уколико неки тренер одустане од рада у фитнес центру, администратор га може уклонити кликом на дугме “Ukloni trenera”.
* Додавање радника – администратор има могућност да у базу података дода новог радника, да му да права која радник поседује, након додавања у базу, радник може креирати налог и регистровати се
* Преглед свих радника – дат је преглед свих особа које су у систему пријављене као радници, администратор има право да уколико неко од радника одустане од рада у фитнес центру истог уклони из базе, одузме му сва права и уклони налог
* Преглед утисака – администратор из базе података добија преглед свих утисака које су корисници оставили, утиске може да филтрира и уклони оне који су нерелевантни.
* Преглед дуговања – у виду табеле је дат преглед свих дуговања, колоне табеле су ИД корисника, укупно дуговање, опис дуговања, као и податак за који месец корисник дугује. Администратор води рачуна да сва дуговања буду измирена у одговарајућем временском периоду, па кликом на дугме “Obavesti korisnika” генерише поруку која се шаље кориснику, у поруци је кратак опис дуговања, месец за који дугује као и цифра дуга.
* Преглед захтева за пробне тренинге – администратор има увид у све захтеве које су корисници послали за пробне тренинге. Администратор система проверава податке које је корисник унео и на основу валидности података одобрава пробни тренинг.
* Последња опција јесте опција повратка на почетну страницу, приликом чега се сесија успешно завршава.
  1. Тренер

У овом поглављу биће наведене и описане све могућности које тренер поседује.

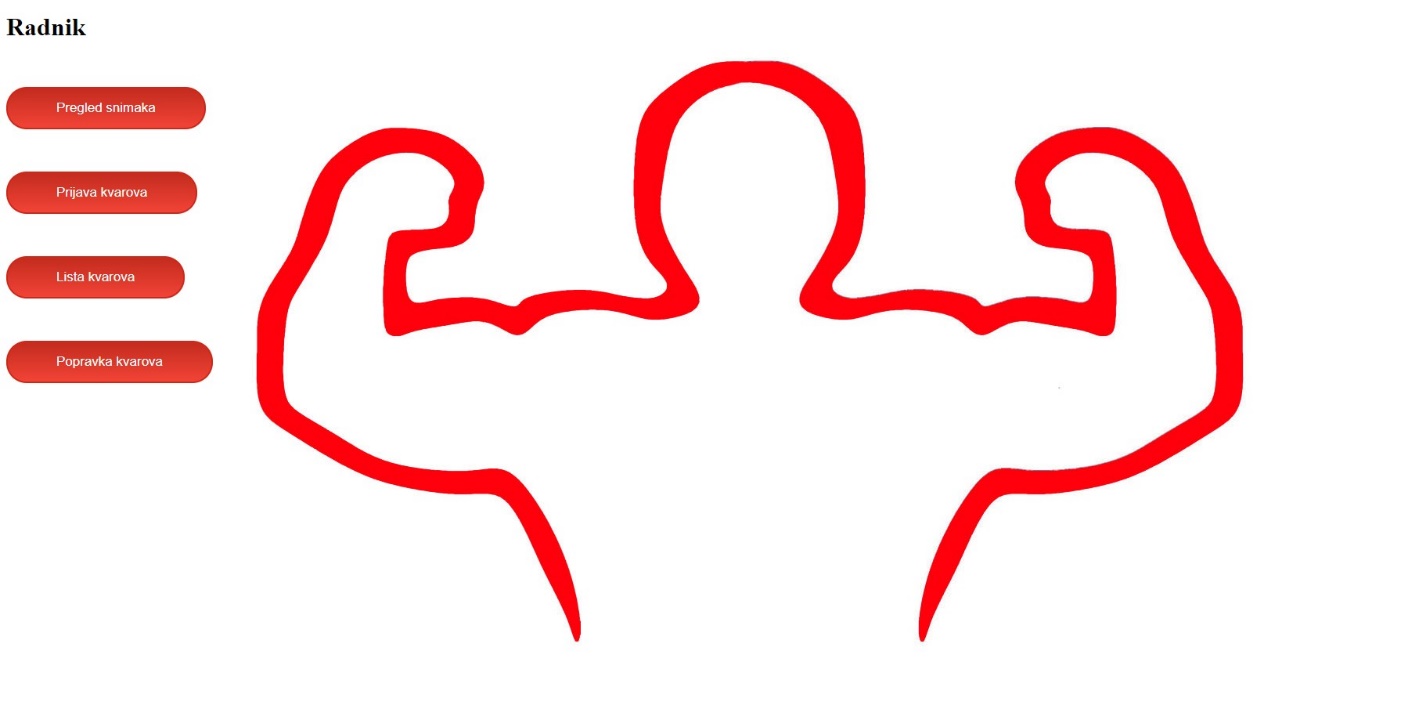


**Слика 27 Изглед странице тренера**

Тренер поседује опције:

* Додавања тренинга – тренер има ограничен приступ бази где може да унесе нови тип тренинга. За нови тренинг тренер уноси назив, цену и даје кратак опис.
* Заказивање тренинга – тренер има увид у све захтеве који су пристигли од корисника и на основу захтева бира датум и време и кликом на дугме “Obavesti korisnika” обавештава корисника за који датум и у које време је заказан тренинг.
* Преглед заказаних тренига – у оквиру ове опције тренеру се приказује списак свих тренинга које је он заказао. Тренер има право да кликом на дугме “Otkaži” откаже заказани тренинг и о томе обавести корисника.
* Последња опција јесте опција одјављивања из система и враћање на почетну страницу.
  1. Радник

Могућности које поседује радник у систему биће дате кроз тезе. Сва теза представља по једну могућност која ће бити описана са по једном реченицом.



Слика 28 Радник

Опције које радник поседује су:

* Преглед снимака – радник има могућност да прегледа снимке сигурносиних камера уколико је открио квар
* Пријава кварoва – уколико је радник открио квар може га пријавити и одредити новчану накнаду за исти
* Листа кварова – Радник има увид у листу кварова

Закључак

У овом раду је дат опис система аутоматизованог фитнес центра у коме се присуство људског фактора своди на минимум. Обухваћен је развој система од функционалне спецификације па све до имплементације сваког дела система појединачно. Овакав систем представља пример вишеслојне апликације.

Технологије које су коришћене за израду ове апликације знатно су учинили да сама израда буде олакшана, брза и једноставна. Широк спектар компоненти које нуди Primefaces је омогућио да за сваки проблем на који се наилазило постоји једноставно и брзо решење. Hibernate алат је дао могућност лаког манипулисања са базом података, где се помоћу једноставних метода може доћи до сваког податка унутар базе. И као овир свему JSF је допринео да сви алати и све библиотеке функционишу у складу и беспрекорно.

Сам систем је могуће надоградити што се хардверског дела тиче, додавањем нових сензора и уређаја који ће у знатном опсегу олакшати управљање системом и самим фитнес центром. Додавањем додатног хардвера и сама апликација би се проширила, тако да би свака новододата хардверска компонента била испраћена и апликативним делом.

Литература

1. Arduino - Arduino Home - <https://www.arduino.cc/> , приступано септембрa 2020.
2. NetBeans - Wikipedia, <https://en.wikipedia.org/wiki/NetBeans> , приступано септембра 2020.
3. JavaServer Faces - Wikipedia, <https://en.wikipedia.org/wiki/Jakarta_Server_Faces> , приступано септембра 2020.
4. XHTML - Wikipedia, <https://en.wikipedia.org/wiki/XHTM> , приступано септембра 2020.
5. CSS - Wikipedia, <https://en.wikipedia.org/wiki/Cascading_Style_Sheets> , приступано септембра 2020.
6. Primefaces - Showcase – Primefaces, <https://www.primefaces.org/showcase/ui/button/button.xhtml> , приступано септембра 2020.
7. W3Schools - W3schooles - <https://www.w3schools.com/> , приступано септембрa 2020.
8. Hibernate (framework) - Wikipedia, <https://en.wikipedia.org/wiki/Hibernate_(framework)> , приступано септембрa 2020.
9. MySQL Workbench - Wikipedia, <https://en.wikipedia.org/wiki/MySQL_Workbench> , приступано септембрa 2020.
10. Електротехнички факултет у Београду, Катедра за рачунарску технику и информатику, Програмирање интернет апликација - “Уџбеник са збирком задатака” - <http://rti.etf.bg.ac.rs/rti/ir4pia/materijali/knjige/PIA_knjiga_2017_v0.9_draft.pdf> , приступано септембрa 2020.
11. Електротехнички факултет у Београду, Катедра за рачунарску технику и информатику, Програмирање интернет апликација - “JavaServer Faces (JSF)” - <http://rti.etf.bg.ac.rs/rti/ir4pia/materijali/predavanja/PIA_Lekcija4_JSF.pdf> , приступано септембрa 2020.

Списак слика

[Слика 3 Изглед Primefaces компоненти 9](#_Toc52241482)

[Слика 5 Изглед базе података 12](#_Toc52241483)

[Слика 6 Филтер ауторизације 15](#_Toc52241484)

[Слика 7 Успешно оставаривање сесије 16](#_Toc52241485)

[Слика 8 Метода за хеширање шифре 17](#_Toc52241486)

[Слика 9 Коришћење методе за хеширање 17](#_Toc52241487)

[Слика 10 Метода checkpw 18](#_Toc52241488)

[Слика 11 Коришћење методе checkpw 18](#_Toc52241489)

[Слика 12 Коришћење регуларних израза приликом валидације лозинке 19](#_Toc52241490)

[Слика 13 Коришћење регуларних израза за испитивање формата email-а 19](#_Toc52241491)

[Слика 14 Arduino код 21](#_Toc52241492)

[Слика 15 Почетна страница 22](#_Toc52241493)

[Слика 16 Скок са почетне странице на неку од опција (у овом случају логин страница) 23](#_Toc52241494)

[Слика 17 Изглед странице за заказивање пробног тренинга 24](#_Toc52241495)

[Слика 18 Тренинзи који су у понуди 24](#_Toc52241496)

[Слика 19 Страница за креирање налога 25](#_Toc52241497)

[Слика 20 Регистрација 25](#_Toc52241498)

[Слика 21 Изостављање садржаја поља 27](#_Toc52241499)

[Слика 22 Пример не испуњавања формата лозинке 28](#_Toc52241500)

[Слика 23 Логин страница 29](#_Toc52241501)

[Слика 24 Непостојеће корисничко име 30](#_Toc52241502)

[Слика 25 Унос погрешне лозинке **Error! Bookmark not defined.**](#_Toc52241503)

[Слика 26 Објекти 31](#_Toc52241504)

[Слика 27 Остављање коментара 31](#_Toc52241505)

[Слика 28 Кориснички део апликације 32](#_Toc52241506)

[Слика 29 . Одабир тренинга 33](#_Toc52241507)

[Слика 30 Страница администратора 34](#_Toc52241508)

[Слика 31 Изглед странице тренера 35](#_Toc52241509)

[Слика 32 Радник 36](#_Toc52241510)