Развој апликације за руковођење научним семинаром

*Дипломски рад*

Студент Ментор

Јована Мика Драган Матић

Бања Лука, 2023.

Садржај:

[1. Увод 2](#_Toc152511290)

[2. Постојеће апликације на сличну или исту тему 3](#_Toc152511291)

[3. Развој веб апликације 7](#_Toc152511292)

[3.1. Веб апликација 7](#_Toc152511293)

[3.2. Динамичке веб странице 7](#_Toc152511294)

[3.3. Front-end и Back-end развој 8](#_Toc152511295)

[3.3.1. Креирање корисничког интерфејса 8](#_Toc152511296)

[3.3.2. Развој функционалности у позадини 8](#_Toc152511297)

[3.4. Кориштене техологије и алати 8](#_Toc152511298)

[3.4.1. HTML 8](#_Toc152511299)

[3.4.2. CSS 9](#_Toc152511300)

[3.4.4. React JS 10](#_Toc152511301)

[3.4.4. Visual Studio Code 11](#_Toc152511302)

[3.4.5. Java 12](#_Toc152511303)

[3.4.6. Spring Boot 12](#_Toc152511304)

[3.4.7. Maven 13](#_Toc152511305)

[3.4.8. Eclipse IDE 13](#_Toc152511306)

[4. Кориснички захтјеви 15](#_Toc152511307)

[4.1. Дефиниције 15](#_Toc152511308)

[4.2 Расподијела захтјева 16](#_Toc152511309)

[3. Модел базе 17](#_Toc152511310)

[5. Развој (изворни код и интерфејс) alati? 18](#_Toc152511311)

[4.1.Početna stranica 19](#_Toc152511312)

[4.2.Modul za registraciju 20](#_Toc152511313)

[4.3. Modul za prijavu 21](#_Toc152511314)

[4.4 Modul za prikaz korisničkog profila 22](#_Toc152511315)

[4.5 Modul za prikaz seminara 24](#_Toc152511316)

[4.6 Modul za prikaz objava 25](#_Toc152511317)

[4.7. Modul za prikaz pojedinačne objave 27](#_Toc152511318)

[4.8. Administratorski panel 28](#_Toc152511319)

[6. Закључак 31](#_Toc152511320)

[7. Литература 32](#_Toc152511321)

# 1. Увод

Друштво све више препознаје значај науке и образовања као кључних елемената напретка и развоја. Научни семинари постају важан дио академске сфере, а напредак технологије омогућава нове приступе организацији и праћењу научних семинара путем апликација које пружају ефикасније алате за управљањем садржајем.

Учествовање у научном семинару не само да доприноси академском развоју студената, већ такође омогућава истраживачима и професорима да дијеле своје знање и искуство. Ипак, организација и вођење семинара могу представљати изазов, посебно у окружењима гдје се технолошки алати не користе довољно или нису прилагођени одређеним потребама ове врсте образовне активности.

Циљ овог дипломског рада је развој апликације која ће омогућити лако и ефикасно руковођење научним семинаром. Веб апликација за руковођење научним семинаром креирана је тако да корисницима омогући преглед информација везаних за семинаре који су се одржали или тек треба да се одрже, корисне објаве у области науке као и обавјештења о другим догађајима научног карактера. Поред информативног караткера, апликација омогућава и резервацију мјеста, односно пријаву на семинар.

# 2. Постојеће апликације на сличну или исту тему

Проучавање већ постојећих апликација са сличном или истом тематиком помаже нам да боље разумијемо основне функционалности коју су важне за кориштење апликација. Анализом њихових предности и недостатака можемо доћи до идеје о томе како унаприједити постојеће апликације и створити нову која ће задовољити одређене потребе циљне групе.

У наставку ће бити приказани кориснички интерфејси двије апликације које садрже неки вид управљања научним семинаром. Оно што је заједничко овим апликацијама јесте што садрже све релевантне информације везане за рад факултета. С обзиром на количину информација које пружају и сложене структуре, проналажење одређених секција посвећених вођењу семинара може представљати изазов.



**Слика 2.1.** *Приказ одјељка који садржи информције о научним скуповима, семинарима и исраживачким радовима.*



**Слика 2.2.** *Приказ објаве везане за мјесто и вријеме одржавање семинара.*

**Слика 2.3.** *Примјер објава везаних за научна истраживања.*

# ScIMI.png

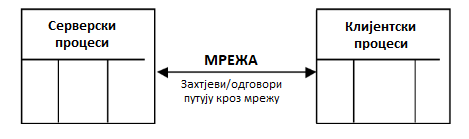
**Слика 2.4.** *Примјер апликације Института за математику и информатику.*

# 3. Развој веб апликације

У овом поглављу ће бити описане технологије које су кориштене у изради веб апликације. Прије него што се опишу конкретне технологије, размотрићемо основне појмове везане за развој веб апликација.

## 3.1. Веб апликација

Веб апликација је врста софтверске апликације којој се приступа и са којом се остварује интеракција путем веб претраживача [1] . Ове апликације се базирају на клијент-сервер моделу. Клијентски дио извршава се у веб прегледачу и представља особу или организацију која користи одређену услугу, док је сервер удаљени рачунар који пружа информације или омогућава приступ одређеним услугама.



***Слика 3.1.1.*** Приказ основног клијент- сервер модела

На почетку развоја интернета, већина веб страница служила је за приказ статичких података написаних уз помоћ ХТМЛ- а (енгл. *HyperText Markup Language* ) . Клијент- сервер модел у овом случају функционисао је тако што би корисници слали захтјев серверу за одређеном страницом, а сервер би им као одговор враћао предефинисану страницу која се налази у фајл систему сервера.

## 3.2. Динамичке веб странице

Развојем технологије, јавила се потреба за већом интеракцијом између корисника и сервера. Како би се ово постигло, развијене су динамичке веб репрезентације чије су странице генерисане динамички од стране сервера приликом обраде корисничког захтјева.

Динамичка веб страница пружа различит садржај корисницима, при чему задржава исти изглед и дизајн. У поређењу са једноставним статичким страницама, чији се садржај не мијења, за учитавање динамичке странице потребно је више времена.

Динамичке странице често користе апликативне програме за различите врсте услуга и захтјевају ресурсе на страни послужитеља, укључујући и приступ базама података.

## 3.3. Front-end и Back-end развој

Развој веб апликације обухвата 2 важна аспекта: креирање корисничког интерфејса (енгл. *front-end*) и развоја функционалности у позадини (енгл. *back-end*) .

### 3.3.1. Креирање корисничког интерфејса

Развој корисничког интерфејса обухвата креирање и имплементацију видљивих елемената сајта, са циљем да се кориснику пружи пријатно и функционално искуство. Важно је да кориснички интерфејс буде прилагођен различитим типовима уређаја и претраживача. Поред основих технологија као што су *HTML*, *CSS* и *JavaScript* , за развој корисничког интерфејса кориштена је *React* библиотека, која ће бити детаљније описана у наставку.

### 3.3.2. Развој функционалности у позадини

Развој функционалности у позадини односи се на дио апликације који се бави логиком и обрадом података, а којем корисник не може директно да приступи. Овај дио укључује серверску страну апликације, базу података као и све процесе који се одвијају како би се омогућиле одређене функционалности корисницима и осигурао интерфејс са корисничком страном.

## 3.4. Кориштене техологије и алати

### 3.4.1. HTML

HTML (*HypetText Markup Language*) је мета језик за означавање који се користи за креирање структуралног оквира веб страница. Састоји се од низа елемената, односто тагова, на осниву којих веб прегледач приказује садржај одређеног веб документа. Неки од примјера кориштења тагова су: форматирање садржаја, приказивање табеле, уметање слике и многи други [4].

На слици 3.4.1.1. приказана је осовна структура *HTML* документа.



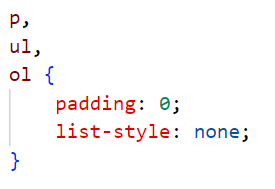
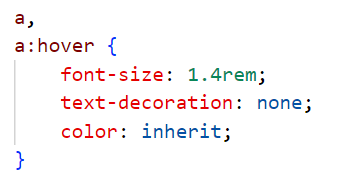
**Слика 3.4.1.1.** *Основна структура html документа.*

### 3.4.2. CSS

CSS (*Cascading Style Sheets*) је језик који се користи за дефинисање правила приказа елемената у документима као што су на примјер HTML и XML. CSS правила обично се дефинишу унутар датотеке чија је екстензија .css, .scss, или могу бити интегрисана унутар HTML или XML докумената [4].

Помоћу CSS-а можемо прецизно управљати изгледом елемената у прегледачу. Да бисмо стилизовали одређени елемент, наводимо назив елемента, а затим у витичастим заградама додајемо CSS код како бисмо постигли жељени изглед. Такође, правила се могу примјењивати на одређене класе додавањем тачке испред имена класе, или на одређене идентификаторе кориштењем знака „#“ испред имена идентификатора елемента.

Могуће je стилизовати више елемената истовремено тако што се називи класа, идентификатора или елемената дефинишу један за другим, одвојених зарезом (*Слика 3.4.2.1.*).

**Слика 3.4.2.2.** *Примјер постављања „hover“ догађаја на елемент.*

**Слика 3.4.2.1.** *Примјер стилизовања више елемената истовремено.*

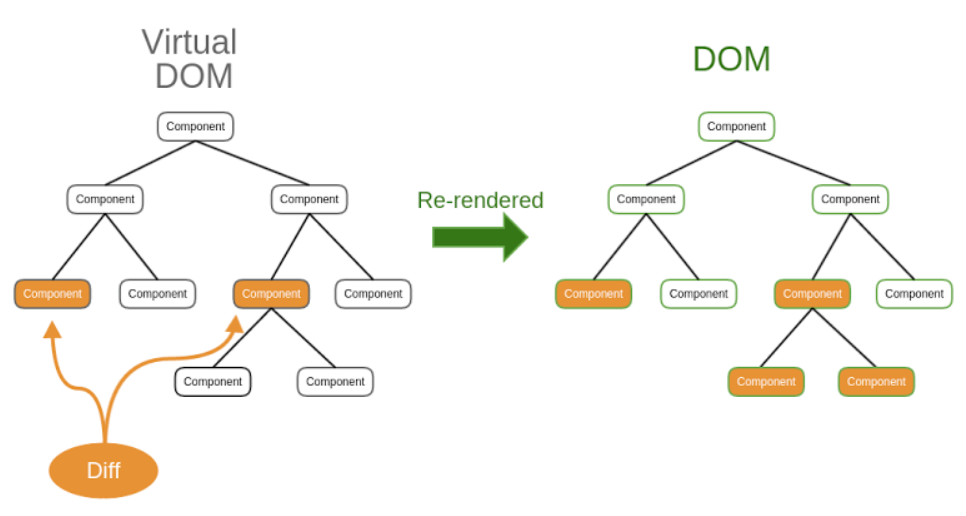
За свако правило можемо додати догађаје, као што су *„hover“*, „*focus“*, „*submit“* и други, који дефинишу када се правило примјењује на одређени елемент (*Слика 3.4.2.2.*). Уколико желимо одређени стил примијенити на све елементе то можемо урадити кориштењем ознаке „\*“ умјесто назива елемента.

### 3.4.4. React JS

React JS је JavaScript библиотека, која је креирана како би се изградили кориснички интерфејси који ефикасно реагују на догађаје корисничког уноса и управљање стањем апликације [7]. Главни циљ у креирању ове библиотеке био је превазилажење изазова који су повезани са развојем сложених апликација које садрже велику количину података. За разлику од других популарних окружења за развој веб апликација React представља самосталну библиотеку чија је основна намјена дефинисање изгледа и приказа података на екрану, док остале функционалности програмер мора да напише самостално или уз помоћ библиотека које су компатибилне са React библиотеком.

*React* подржава кориштење више екстензија као што су *. js* или *.jsx*. *JSX* ( *JavaScript XML*) је синтаксна екстензија која омогућава писање структура корисничког интерфејса на начин сличан *HTML*-у. Кориштење *JSX* синтаксе олакшава визуелизацију и писање *React* компоненти комбиновањем ознака сличних *HTML*-у са *JavaScript* логиком [9].

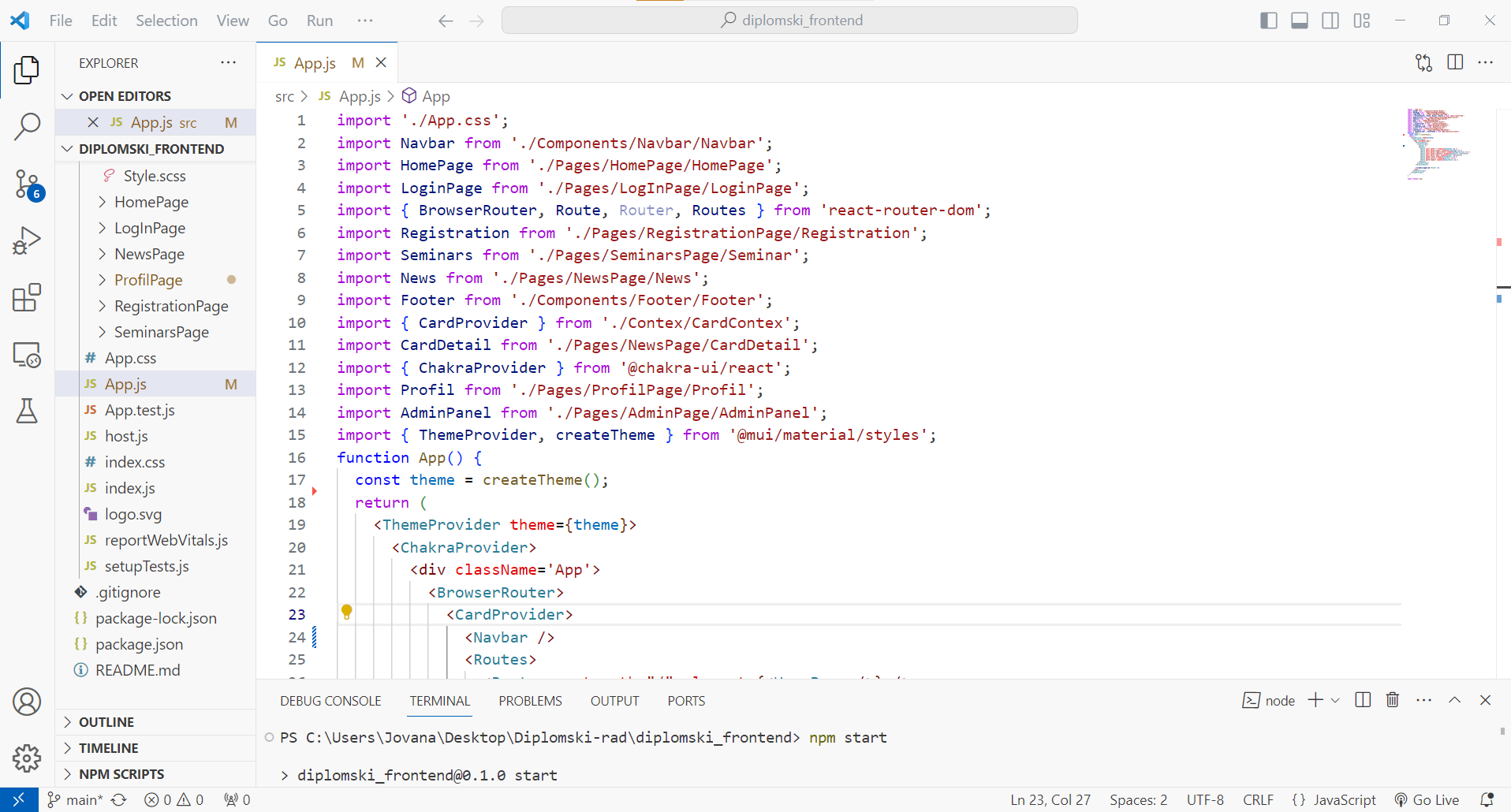
Основна карактеристика React-a је употреба виртуалног DOM-а ( *Слика 3.4.3.1.*).  
DOM ( *енгл. Document Object Model*) представља интерфејс за програмиране апликацији и омогућава програмима да читају и мијењају садржај било које веб странице. За разлику од *DOM*-a код кога се промјене приказује тек након поновног учитавања странице, виртуални *DOM* омогућава тренутно приказивање промијена у стварном времену без потребе за поновним учитавањем.



**Слика 3.4.3.1.** *React Virtual DOM*

### 3.4.4. Visual Studio Code

*Visual studio Code*, развијен од стране Microsofta, предстаља уређивач изворног кода који је доступан за *Windows*, *MacOS* и *Linux* (*Слика3.4.4.1*)[11]. Долази са уграђеном подршском за *JavaScript*, *TypeScript* и *Node.js* али има богат екосистем екстензија за друге језике као што су: *C++*, *Java*, *Python* и други [11].



**Слика 3.4.4.1.** *Изглед Visual Studio Code-a.*

### 3.4.5. Java

Java је објектно оријентисан, платформски независан програмски језик [12]. Карактеристике овог програмског језика су једностава употреба, сигурност извршавања и могућност преносивости кода. Развијен је од стране компаније *Sun Microsystems* 1995. године [12].

*Java* програмски језик користи специфичан начин компилације. Након писања изворног кода, свака класа чува се у однојеној датотеци која има *.java* екстензију. Кориштењем *Java* компајлера, изворне датотеке се преводе у бајткод и на тај начин се генеришу *.class* датотеке. *Class* датотеке садрже инструкције за извршавање које виртуелнa машинa (*JVM*) користи приликом покретања програма.

Програмски језик Java може се користити за развој различитих врста софтвера, укључујући:

* Веб апликације
* Десктоп апликације
* Мобилне апликације
* Уграђене системе
* Игрице

### 3.4.6. Spring Boot

*Spring Boot* представља оквир отвореног типа за развој апликација у програмском језику *Javа*. Уграђени веб сервери, у нашем случају *Tomcat* омогућава једноставно покретање *Java* апликације. Довољно је апликацију покренути као *Java* апликацију.

При иницијализацији пројекта, *Spring Boot* поставља иницијална *pom.xml* датитека који представља конфигурацију *Maven* пројекта и омогућава *Maven* алату да аутоматски управља зависностима, изградњом и експортовањем *Spring Boot* апликације. Кључни елементи .xml датотеке су [14] :

1. ***<modelVersion>*:** Означава верзију POM датотеке.
2. ***<parent>* :** Дефинише родитељски ПОМ.
3. ***<groupId>***, ***<artifactId>***, ***<version>***: Идентификатори пројекта.
4. ***<packaging>* :**Дефинише врсту архивирања пројекта, на примјер .*war*.
5. ***<properties>*:** Дефинишу својства пројекта.
6. ***<dependencies>*:**  Листа зависности које пројекат користи. У зависности спадају *Spring Boot* стартери, *Tomcat* и други, о којима ће бити ријечи у наставку рада.
7. ***<build>*** ,***<plugins>*:** Конфигурација за Maven процесе.
8. ***<exclude>*:**Искључује *Lombok* библиотеку из *Spring Boot Maven Plugin-a* како би се избјегао конфликт са генерисаним кодом.

### 3.4.7. Maven

*Apache Maven* је за управљање пројектима који се користи за изградњу, управљање завистима и аутоматизацију процеса развоја софтвера [11].

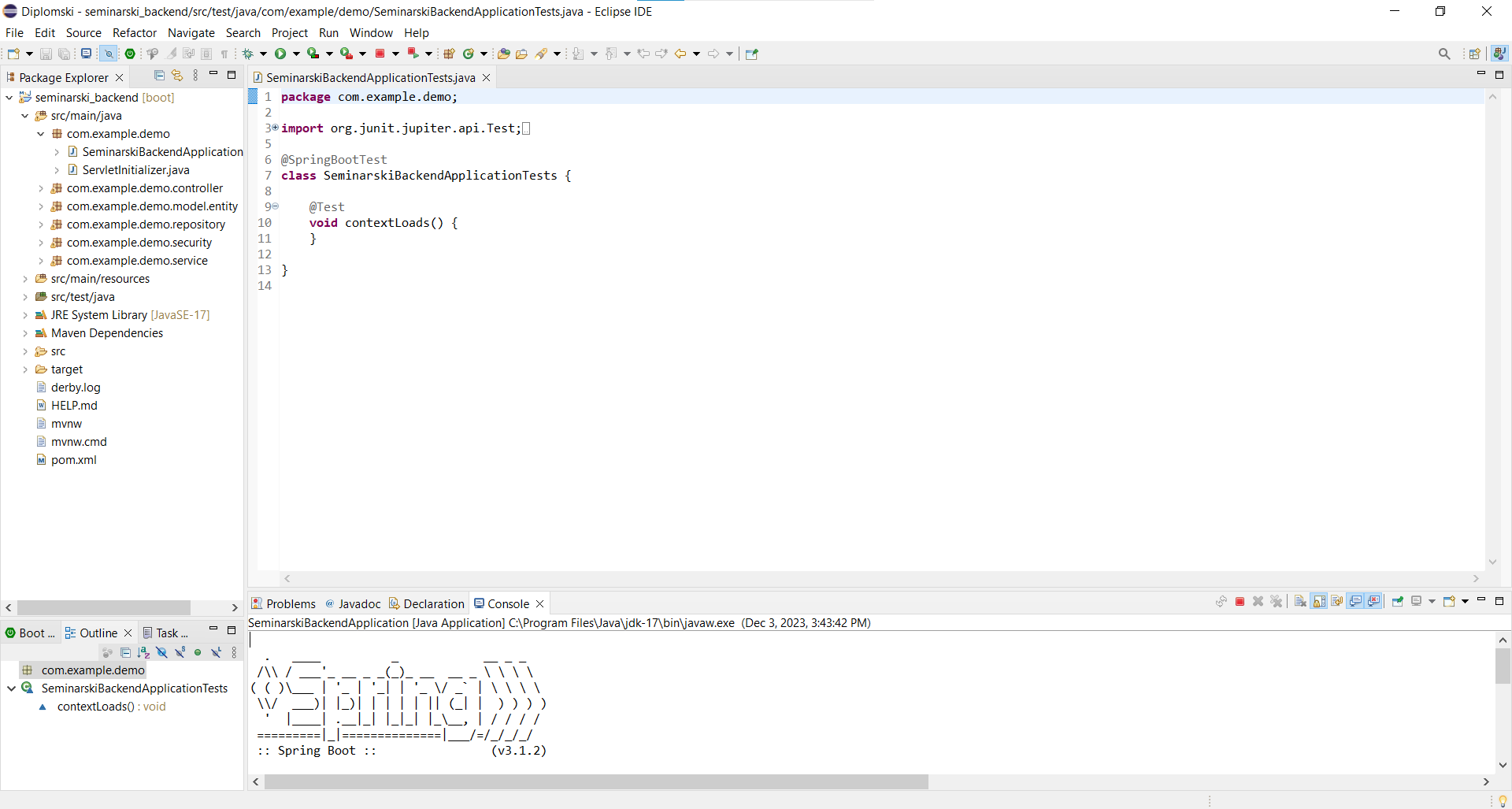
За изградњу датотека користи декларативни приступ који користи конфигурацијске датотеке како би програмерима омогућио ефикасно управљање свим аспектима животног циклуса програма. Једна од основних карактеристика је употреба Објектног модела пројекта (енгл. *Project Object Model*), који дефинише све потребне зависности.

*Maven* аутоматски обавља компилацију, тестирање и друге активности које су саставни дио развоја софтвера.

### 3.4.8. Eclipse IDE

*Eclipse Integrated Development Enviroment* је популаран алат, орвореног кода, за развој софтвера (*Слика 3.4.8.1*) [15]. Развијен је у програмском језику Java, али подржава рад у различитим програмским језицима, укључујући C++, PHP, Python и друге [15].

Доступан је на различитим оперативним системима (Windows, macOS, Linux), што омогућава кориштење истих алата на различитим платформама. Са подршком за развој мобилних апликација, прилагодљивости додатака (енг.plugins), алатима за дебаговање и разним другим могућностима*, Eclipse* пружа окружење за ефикасно окружење за развој софтвера.



**Слика 3.4.8.1.** *Изглед Eclipse развојног окружења.*

### 3.4.9. PostgreSQL

PostgreSQL је систем за управљање базама података објектно-релационог типа [16]. Својим објектно-релационим приступом омогућава дефинисање властитих типова података и функција.

Неке од карактеристика су висока поузданост и подршка трансакцијама, проширивост која омогућава да се база података пришири према потребама пројекта. Поред наведених карактеристика Постгресљл подржава стандардни SQL уз могућност проширења и додавање одређених функционалности. Нуди подршку за JSON податке и на тај начин омогућава рад са веб апликацијама и сервисима који користе JSON формат за размјену података. Припада групи „open source“ софтвера, односно слободно је доступан за преузимање, кориштење и прилагољавање потребама корисника.

Примјену је нашао између осталог за развој веб апликација, пословних система, геопросторних информативних система, аналитичких база података и других.

# 4. Спецификација апликације

Веб аппликација за руковођење научним семинаром намијењена је свим корисницима који желе да се пријаве на семинар, прате дешавања у свијету науке или активно учествују у креирању истих.

Апликацију можемо подијелити на три дијела. Први дио који не захтјева пријаву у систем. Он садржи преглед новости, и листу семинара који су се одржали или тек треба да се одрже, али корисник нема могућност пријаве на семинар.

Други дио, који захтјева регистрацију корисника, омогућава пријаву на семинар и слање захтјева за објавњивање научног поста у случају да је пријавњени корисник професор. Пријавњени корисник има могућност прегледа и измијене приступних података на личном профилу, као и могућност одјаве.

Трећи дио, односно администраторки панел, намијењен је искључиво администраторима апликација. Може да постоји више од једног администратора. Администраторски панел садржи информације о свим објавама, семинарима, корисницима као и могућности измјене и брисања.

Апликација садржи 3 типа корисника: администратор, корисник и предавач. Улоге корисника, као и наведене функционалности биће детаљније описане у наставку рада.

## 4.1. Дефиниције

Администратор Особа која додаје нове кориснике и предаваче, брише их из система и мијења њихове податке. Управља објављивањем, измијеном и брисањем постова и семинара.

Корисник Особа која може да прегледа садржајј веб сајта без да је пријављена. Уколико је корисник пријављен на сајт има могућност пријаве на семинар.

Предавач Особа која може да се пријави на семинар или да активно учестује као предавач, али и да пошаље захтјев администратору за креирање новог семинара.

Регистрација Активност коју корисник извршава попуњавањем одређених поља личним подацима како би могао да приступи сајту.

Пријава Активност коју корисник извршава попуњавањем одређених форми како би потврдио свој идентитет и приступио садржају веб сајта.

Приступни подаци Подаци којима се корисик прјављује на сајт. Приступни подаци су маил и лозинка.

## 4.2 Кориснички захтјеви

***Табела 4.2.1.*** *Приказ учесника и њихових захтјева који се извршавају унутар система.*

|  |  |
| --- | --- |
| **Учесник** | **Захтјев** |
| Администратор | * Пријава * Преглед свих корисника   + Унос новог корисика   + Брисање корисика   + Измјена корисника * Преглед свих семинара   + Унос новог семинара   + Преглед семинара   + Измјена семинара   + Брисање семинара * Преглед свих захтјева за додавање објаве   + Одобравање и објава   + Одбацивање захтјева * Преглед свих објава   + Унос нове објаве   + Измјена објаве   + Брисање објаве * Одјава |
| Предавач | * Пријава * Преглед свих семинара   + Избор појединачног семинара     - Пријава на семинар     - Поништавање пријаве на семинар * Преглед свих објава   + Преглед појединачне објаве * Слање захтјева администратору за објављивање новог поста * Преглед корисничких података на профилу * Измјена корисничких података * Oдјава |
| Корисник | * Преглед садржаја веб странице * Пријава   + Преглед свих семинара     - Избор појединачног семинара       * Пријава на семинар       * Поиштавање пријаве на семинар   + Преглед свих објава     - Преглед појединачне објаве * Преглед корисничких података * Измјена корисничких података * Одјава |

# 5. Развој апликације

У оквиру практичног дијела дипломског рада развијана је апликација за руковођење научним семинаром. Развој апликације може да се подијели на два дијела:

1. Развој серверског дијела апликације (енг. Back-end)
2. Развој корисничког интерфејса (енрг. Front-end)

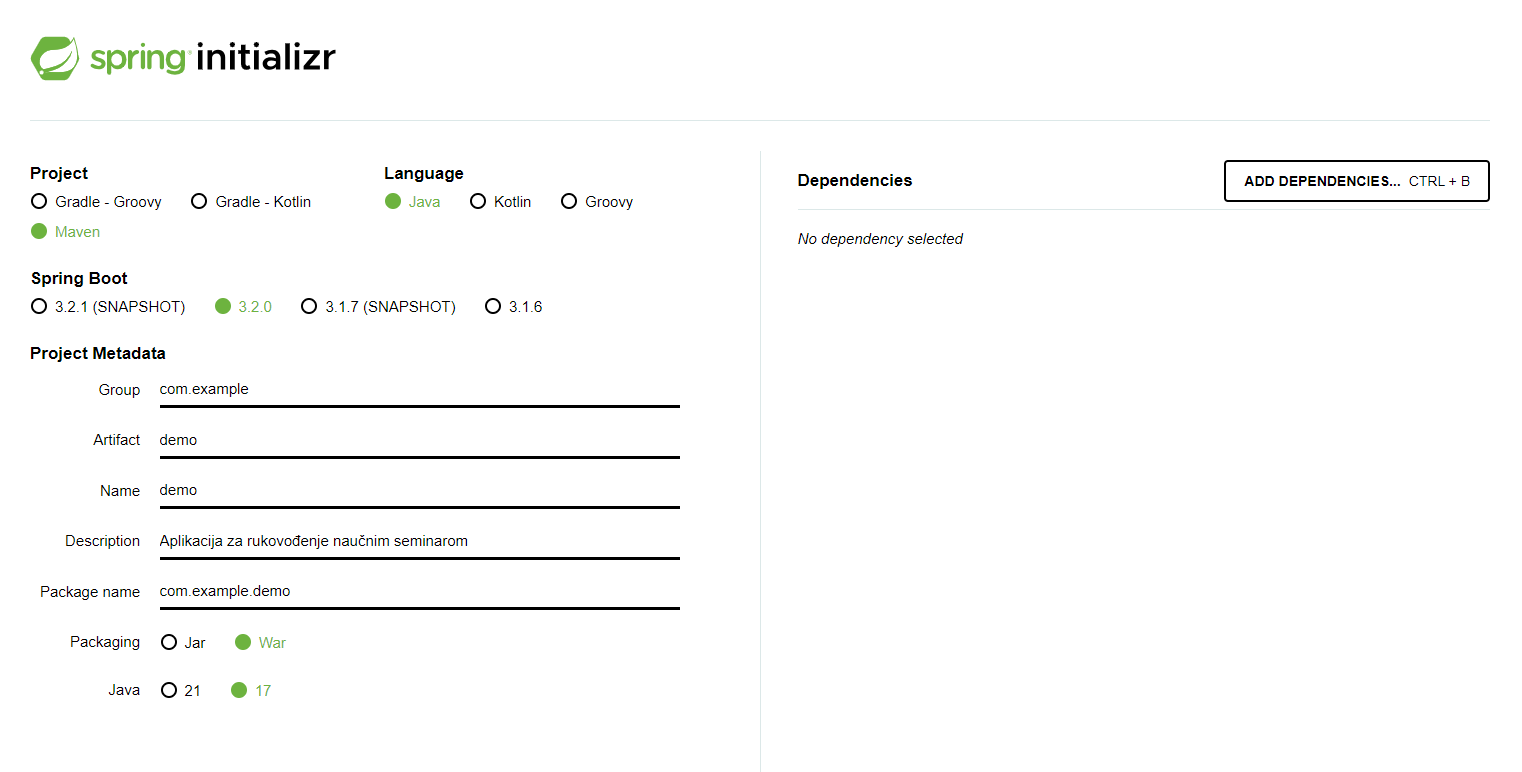
## 5.1. Развој серверског дијела апликације

Серверски дио апликације писан је у *Јava* програмском језику, кориштењем Spring развојног оквира.

Први корак у креирању *Spring Boot* апликације је инсталирање следећих алата:

* *Java SE Development Kit (JDK) 17*
* *Maven 3.2+*
* *Eclipse IDE развојно окружење*

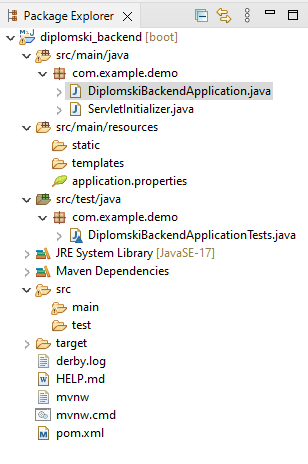
Помоћу *Spring Initializr*-a креиран је *Maven* пројекат (*Слика 5.1.1.*).



**Слика 5.1.1.** *Spring Initializr кориснички интерфејс.*

Генерисани пројекат садржи неколико директоријума и датотека (*Слика 5.1.2.)*:

* ***pom.xml*** - Садржи *Maven build* спецификацију.
* ***application.properties*** - Конфигурацијска даторека.
* ***DiplomskiBackendApplication.java*** - Класа која садржи *main()* методy и служи за покретање апликације.
* ***DiplomskiBackendApplicationTests.java*** – Тест класа која учитава контекст *Spring* апликације помоћу аутоматске конфигурације.

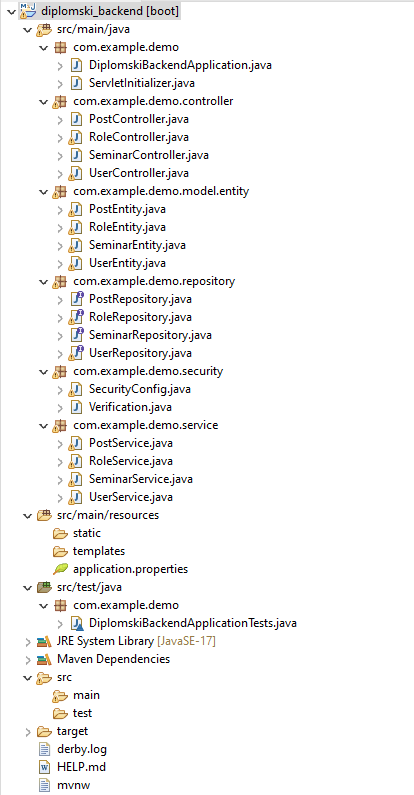


**Слика 5.1.2.** *Структура Maven генерисаног пројекта*.

### 5.1.1. Структура апликације

Структура апликације прати *Model-View-Controller* (MVC) архитектонски образac, при чему поглед није експлицитно дефинисан унутар *Spring* оквира. Кориштењем MVC образца апликација се дијели на три основне компоненте [referenca]:

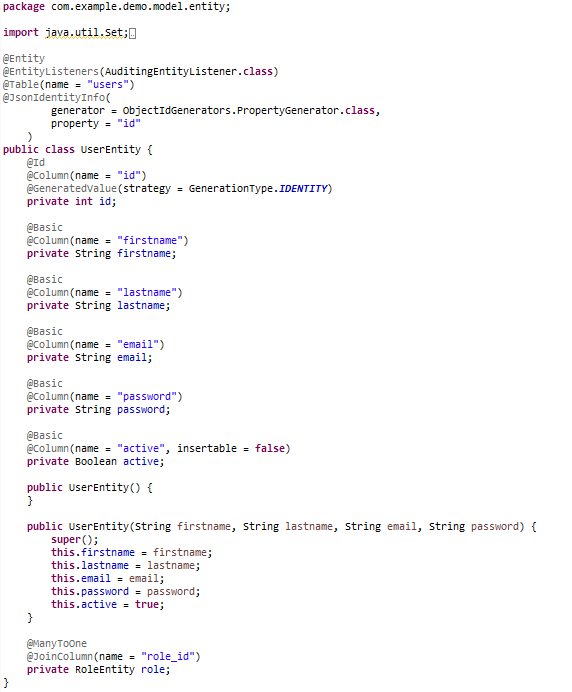
1. ***Модел*** (енгл. *Model*) – Представља податке и пословну логику, обезбијеђује читање и ажурирање података али не зна ништа о корисничком интерфејсу.
2. ***Поглед*** (енгл. *View*) – Обезбијеђује кориснички интерфејс за приказ података и може да прикаже информације из модела.
3. ***Контролер*** (енгл. *Controller*) – Представља повезницу између модела и погледа. Реагује на интеракције корисника.



**Слика 5.1.1.1.** *Структура пројекта након креирања свих потребних пакета*.

Модели апликације креирани су унутар пакета *com.example.demo.model.entity*. Модели су *PostEntity*, *UserEntity*, *RoleEntity*, *SeminarEntity* и они представњају табеле у бази података.

На *слици 5.1.1.2*. приказан је програмски код којим је дефинисана класа UserEntity.



**Слика 5.1.1.2.** *UserEntity класа*.

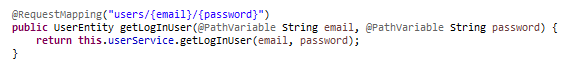
Анотацијe које су кориштене у класама које се налазе у овом пакету:

* *@Entity* – Означава да класа UserEntity представља JPA ентитет који користи ову анотацију како би препознао класу као ентитет и мапирао је на базу података.
* *@EntityListeners(AuditingEntityListener.class)* - Указује на кориштење *AuditingEntityListener*-a који аутоматски поставља вриједности за временске ознаке као што су датум креирања и ажурирања ентитета.
* *@Table(name = 'users')* – Користи се за прилагођавање детаља у бази података. У овом случају мапира ентитет на табелу са именом „users“.
* *@JsonIndentityInfo* - Управља идентитетом приликом серијализације у JSON формат.
* *@Id* - Означава да је поље 'id' кључ ентитета.
* *@GeneratedValue(strategy = GenerationType.IDENTITY)* –Користи се за аутоматско генерисање вриједности за поље које се наведе испод анотације, у овом примјеру то је поље 'id'.
* *@Column(name = 'id')* - Дефинише име колоне у табели базе података која се мапира на одговарајуће атрибуте у класи.
* *@ManyToOne*, *@JoinColumn* - Користе за мапирање асоцијација. Поред ове постоје и *@OneToMany*, *@OneToOne*, *@ManyToMany* анотације.

Унутар класе *com.example.demo.controller* налазе се контролери за сваки од креираних ентитета (*Слика 5.1.1.1*) . Примјер контролера за *UserEntity* класу приказан је на *слици 5.1.1.4.*

Кориштене анотације:

* *@RestController* - Помоћу ове ознаке класе се означава као контролер, а сама анотација комбиује двије анотације *@Controller* и *@ResponseBody* што значи да се резултати метода директно конвертују у HTTP одговор као JSON или XML.
* @CrossOrigin(origins = '\*') - Омогућава захтјевима са свим извора да приступају ресурсима на серверу.
* @RequestMapping - Користи се за мапирање HTTP захтјева на одређени метод (*Слика 5.1.1.3.*) .
* @PathVariable - Користи се за екстракцију вриједности из URL путање (*Слика 5.1.1.3.).*
* @RequestBody – Мапира тијело HTTP захтјева на овјекат.



**Слика 5.1.1.3.** Примјер кориштења @RequestMappig @PathVariable анотација.



**Слика 5.1.1.4.** *UserController класа*.

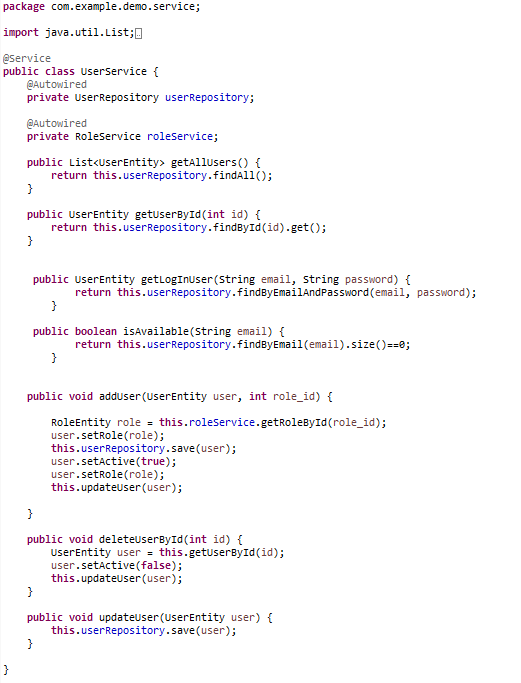
У пакету *com.example.demo.repository* налазе се интерфејси који се користе за приступ и манипулацију података у бази подазака. Користи се у комбинацији са Spring Data JPA модулом. Да би се користио овај модул потребно је додати зависност у pom.xml датотеку (*Слика 5.1.1.5.*).



**Слика 5.1.1.5.** *Дио pom.xml датотеке у којој се дефинише зависност Spring Data JPA модула*.

Service класе налазе се у пакету *com.example.demo.service* и представњају слој пословне логике у апликацији. Све Service класе означене су анотацијом @Service.

На *слици 5.1.1.6.* је приказана UserService класа која управља пословном логиком везаном за кориснике. Користи UserRepositorz интерфејс како би приступила подацима о корисницима из базе података и RoleService за добијање података о улогама. Користећи методе ове класе можемо добити листу свих корисника, провјерити да ли је могућа регистрација новог корисника, додавање новог корисник и друго.



**Слика 5.1.1.6.** *UserService класа.*

### 5.1.2. Модел базе података

Модел базе података креиран је помоћу алата *MySQLWorkbench.*

База се састоји од 4 табеле (*Слика 5.1.2.1):*

1. ***User*:** Представља све кориснике апликације.
2. ***Role*:** Представља улоге које корисник може да има (корисник, админ или предавач) .
3. ***Seminar*:** Представља семинар који се одржавао или се тек треба одржати, садржи колоне у којима се чувају информације о пријављеним учесницима, вријеме одржавања.
4. ***Post*:** Користи се за чување информација о објавама које нису семианри, а мофу да буду објавјњштења, научни чланци и слично.

Поред четири основне табеле постоје и двије међутабеле које се користе при креирању м:н веза.



***Слика 5.1.2.1.*** *Модел базе података*

## 5.2. Развој корисничког интерфејса

U nastavku će biti prikazani moduli koji odgovaraju dizajnu budućeg softverskog rješenja.

Moduli koji su obrađeni u nastavku ovog dokumenta su sljedeći:

1. Početna strana
2. Modul za registraciju korisnika
3. Modul za prijavu korisnika
4. Modul za prikaz korisničkog profila
5. Modul za prikaz svih seminara
6. Modul za prikaz svih objava
7. Modul za prikazivanje pojedinačne objave
8. Administratorski panel

## 4.1.Početna stranica

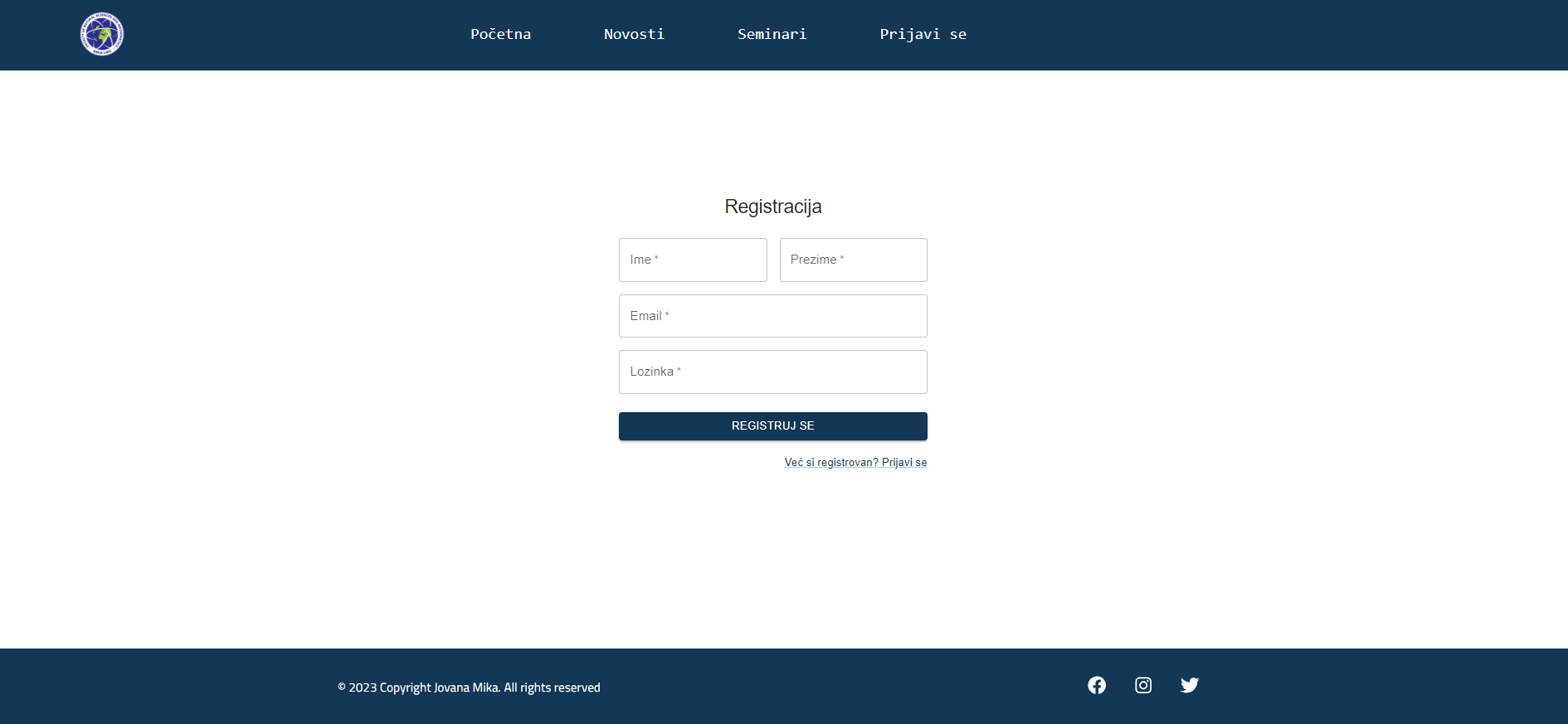
Početna strana predstavlja početni modul koji se prikaže korisniku kada se kroz browser otvori web lokacija aplikacije za potrebe upravljanja naučnim seminarom. Na početnoj strani linkovane forme ka modulima za registraciju korisnika, prikaz svih seminara i slider u kome se prikazuju novosti odnosno postovi.



***Slika 4.1.1.*** *Korisnički prikaz početne strane*

## 4.2.Modul za registraciju

Sadrži formu za registraciju u koju korisnik unosi ime, prezime, email, lozinku. Korištenjem ovog modula nije moguća registracija predavača (predavače dodaje administrator).



***Slika 4.2.1.*** *Prikaz modula za registraciju*

## 4.3. Modul za prijavu

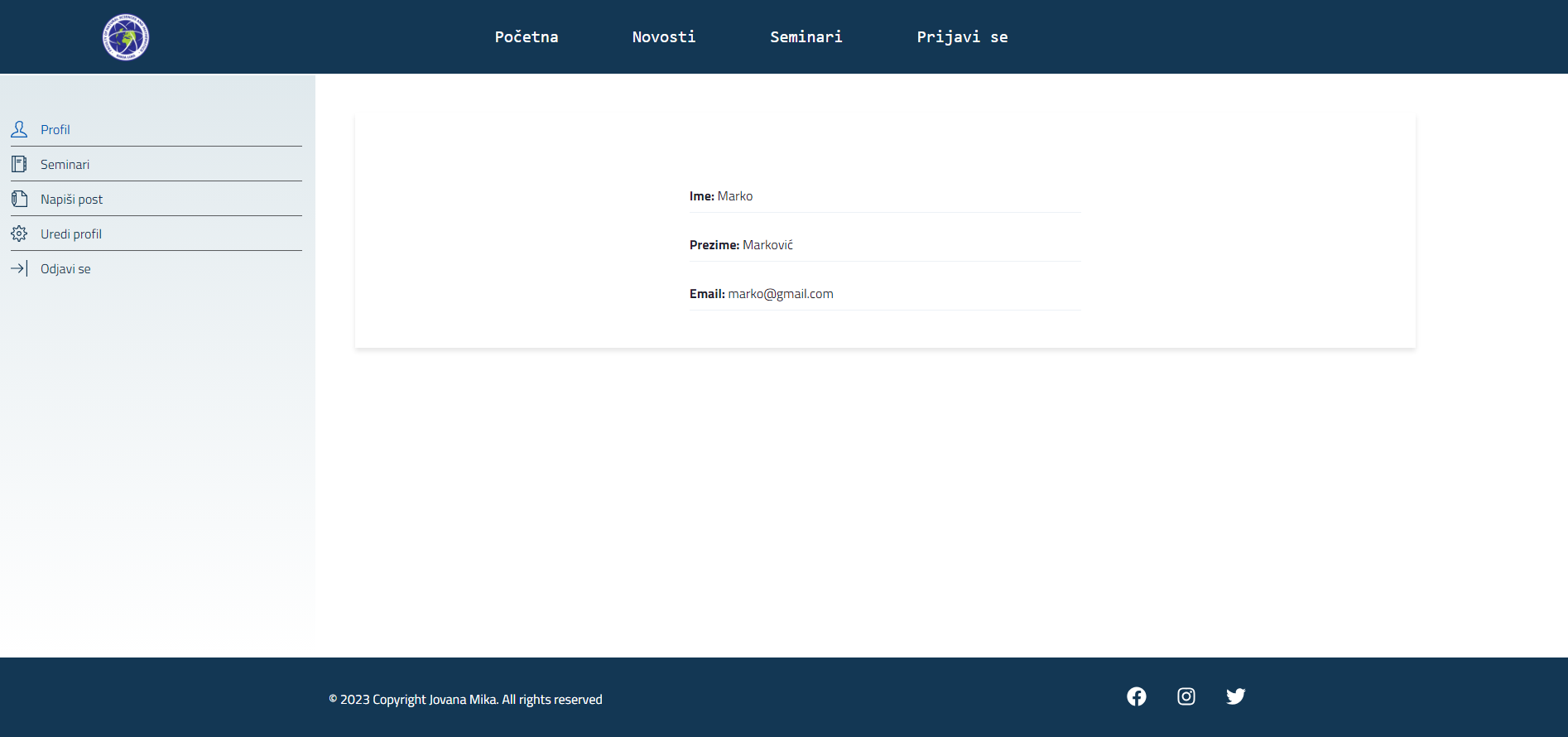
## Login.png

***Slika 4.3.1.*** *Prikaz modula za prijavu*

## 4.4 Modul za prikaz korisničkog profila

Modul za prikaz korisničkog profila sadrži:

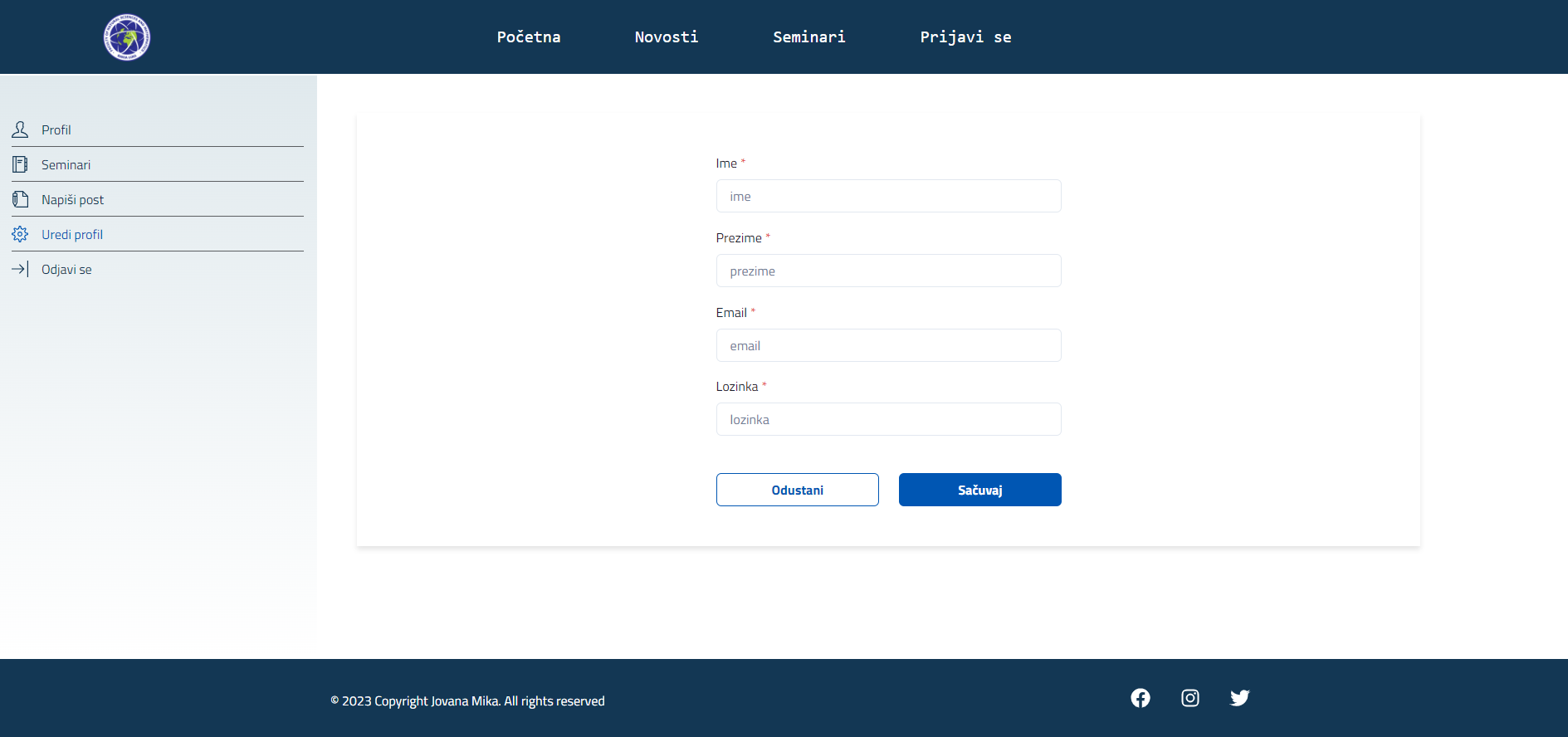
1. Prikaz osnovnih informacija o korisniku.
2. Mogućnost izmjene korisničkih podataka.
3. Opcija za slanje zahjeva za dodavanje novog posta, u slučaju da je prijavljeni korisnik predavač.
4. Opcija za odjavu.



***Slika 4.4.1.*** *Prikaz osnovnih informacija o korisniku u slučaju da je prijavljeni korisnik profesor*

## Profil add post.png

***Slika 4.4.2.*** *Prikaz opcije za slanje zahtjeva za dodavanje nove objave u slučaju da je prijavljeni korisnik profesor*

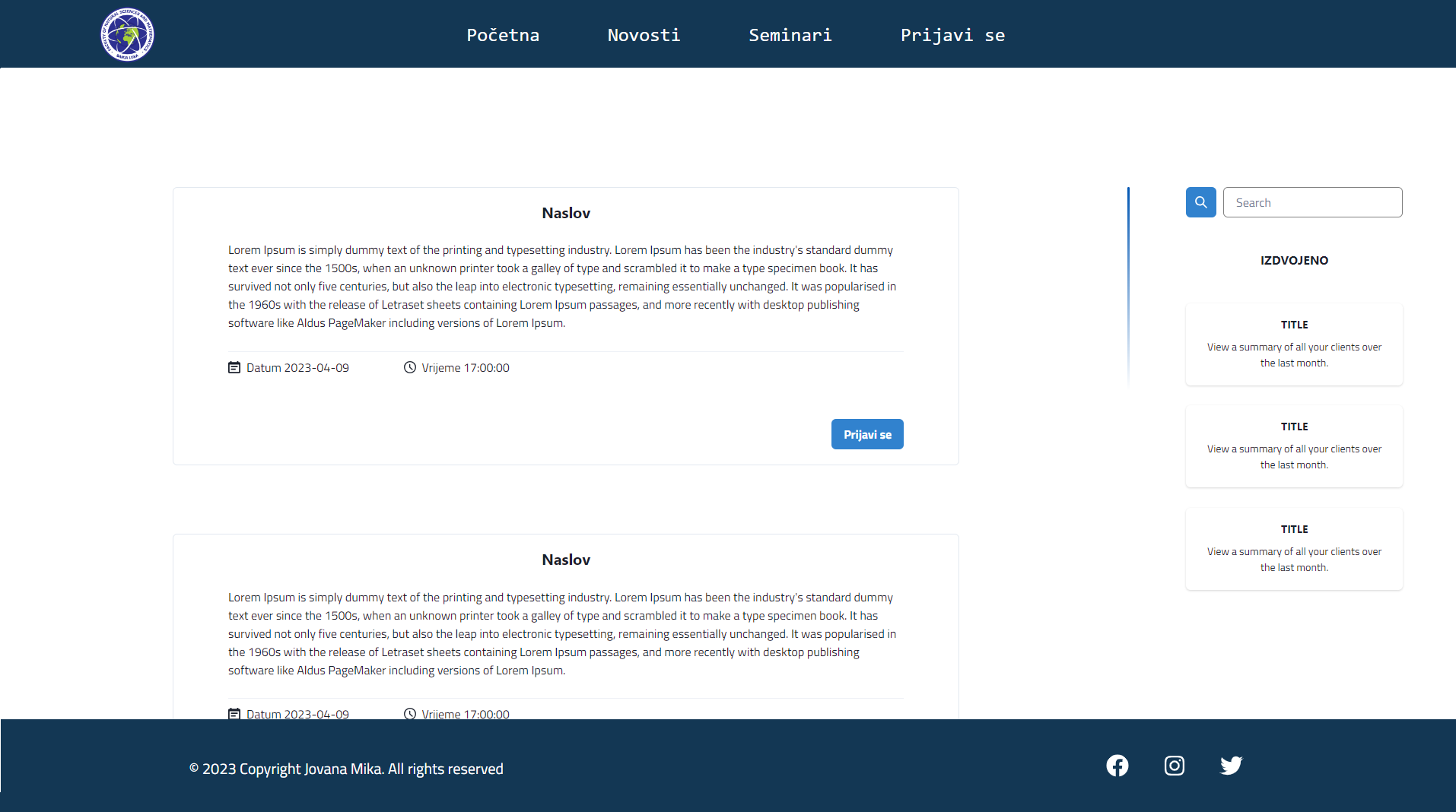


***Slika 4.4.3.*** *Prikaz opcije za uređivanje profila*

## 4.5 Modul za prikaz seminara

Modul za prikaz seminara sastoji se iz lijevog i desnog panela. U lijevom panelu prikazani su svi seminari koje učitavamo iz baze. Svaki seminar prikazan je u vidu kartice na kojoj se nalaze sve potrebne informacije kao i opcija za prijavu na seminar.

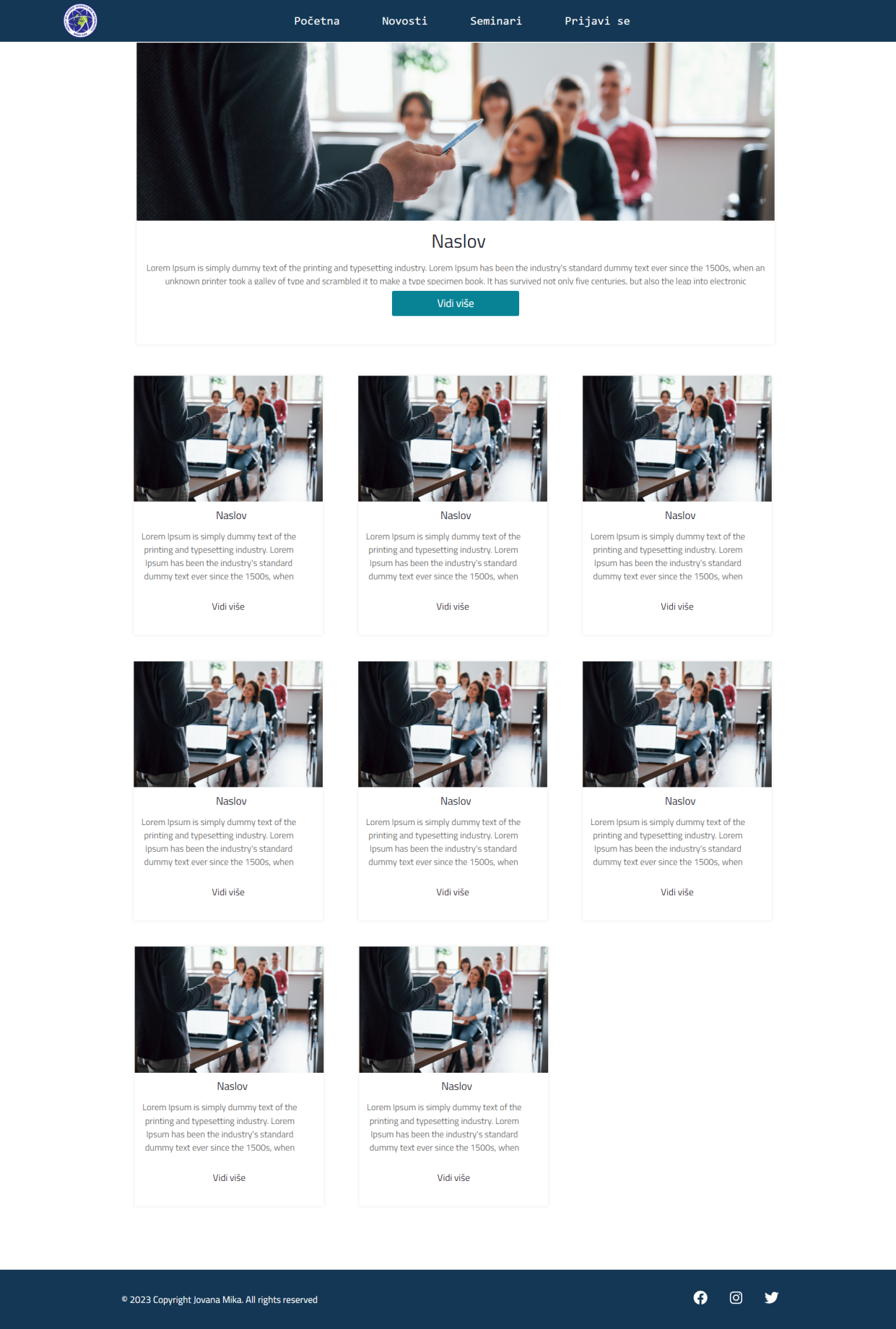
U desnom panelu nalazi se opcija za pretragu seminara po nazivu. Takođe na ovom modulu izdvojeno je i par objava koje se biraju iz baze po datumu objave.



***Slika 4.5.1.*** *Prikaz modula za seminare*

## 4.6 Modul za prikaz objava

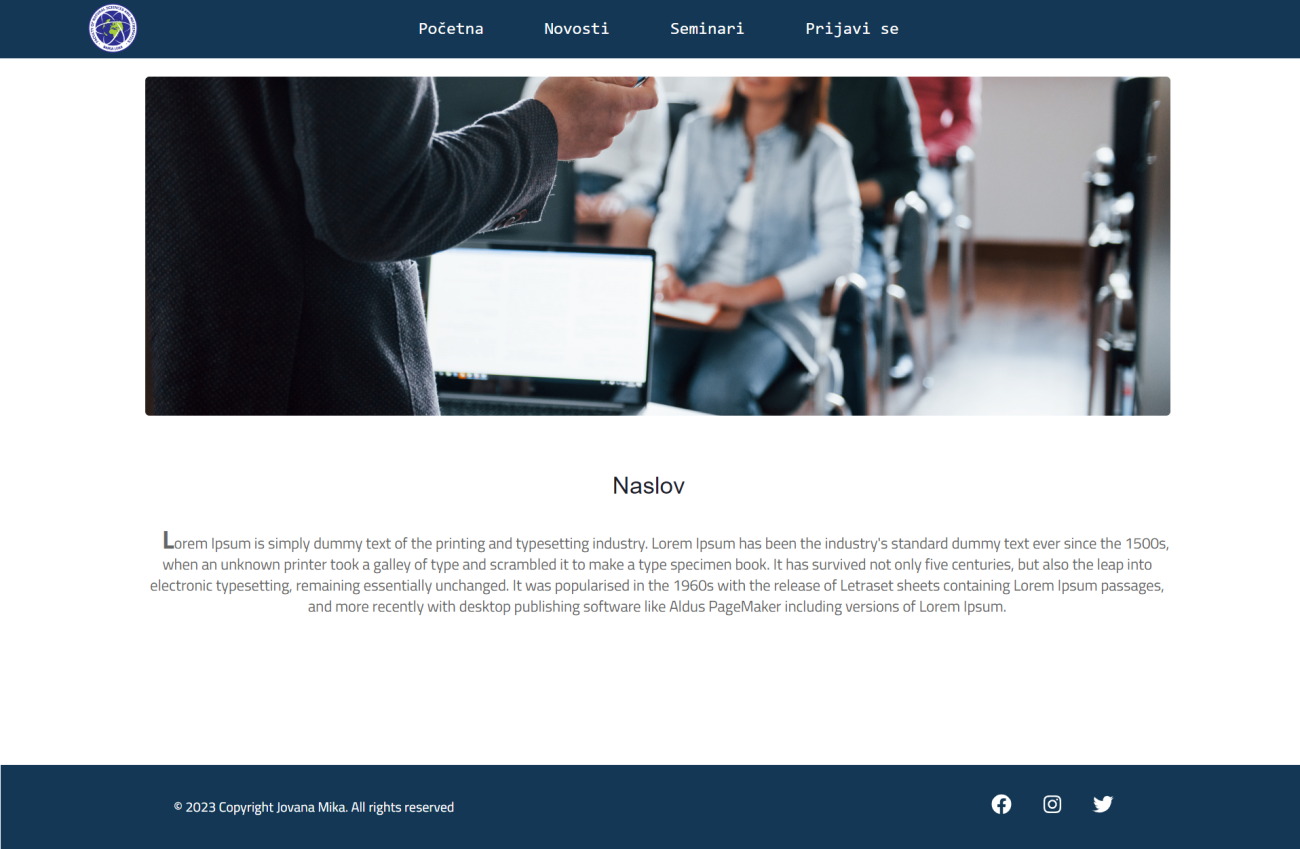
Modul za prikaz objava sadrži kartice na kojima se nalaze određeni podaci vezani za neku objavu. Klikom na dugme „Vidi više“ otvara se novi modul za prikaz pojedinačne objave.



***Slika 4.6.1.*** *Prikaz modula za objave*

## 4.7. Modul za prikaz pojedinačne objave

Modul za prikaz pojedinačne objave sadrži sve informacije vezane za izabranu objavu.



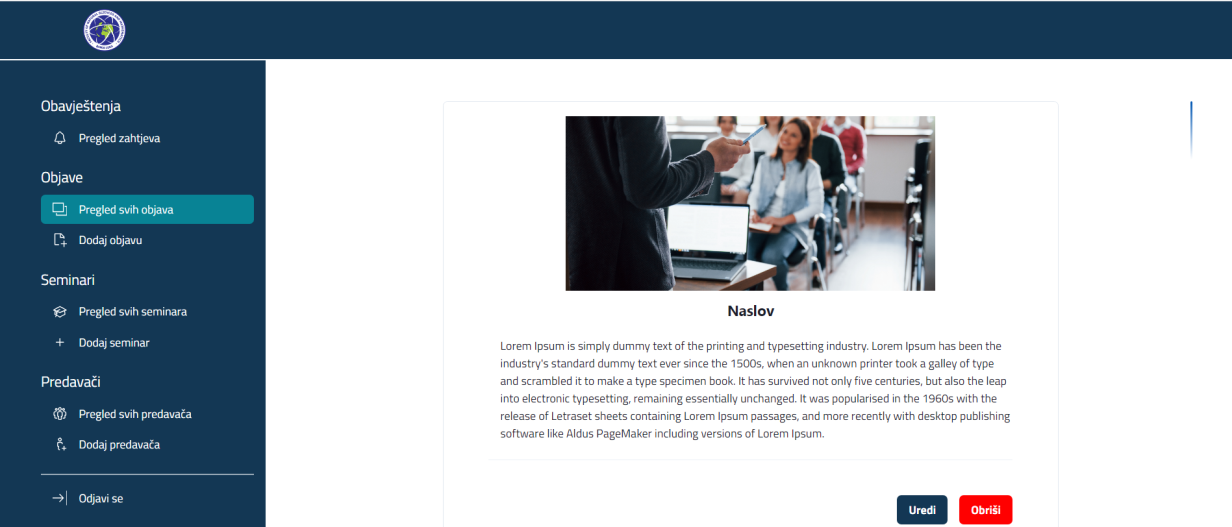
***Slika 4.7.1.*** *Prikaz modula za pojedinačnu objavu*

## 

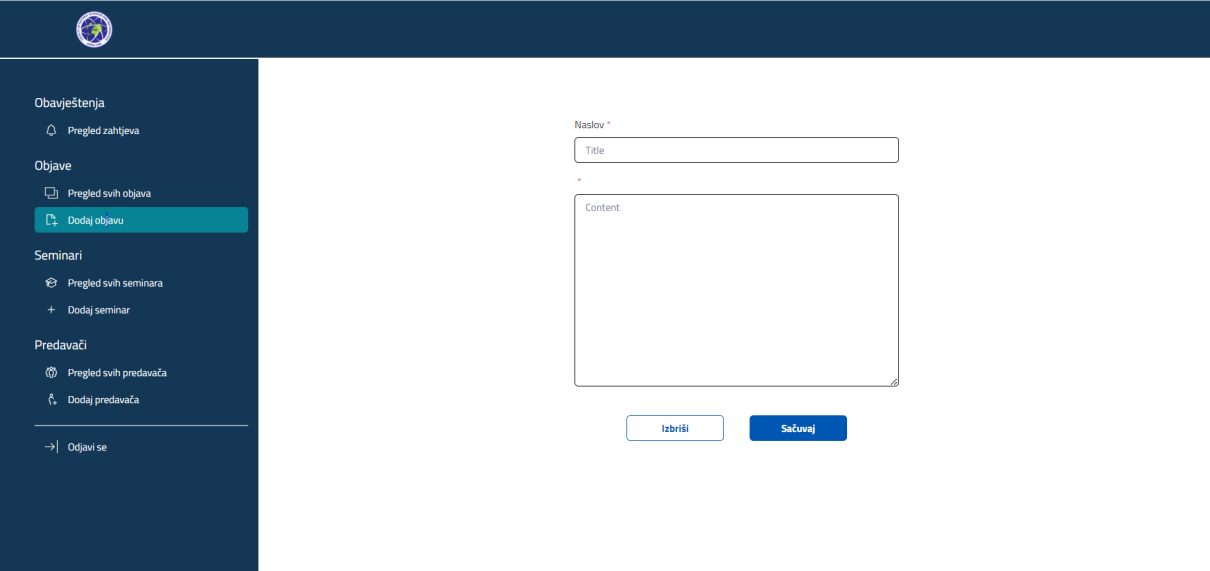
## 4.8. Administratorski panel

Prilikom ulaska na administratorski panel nudi se opcija za prijavu, te se nakon uspješne prijave omogućava:

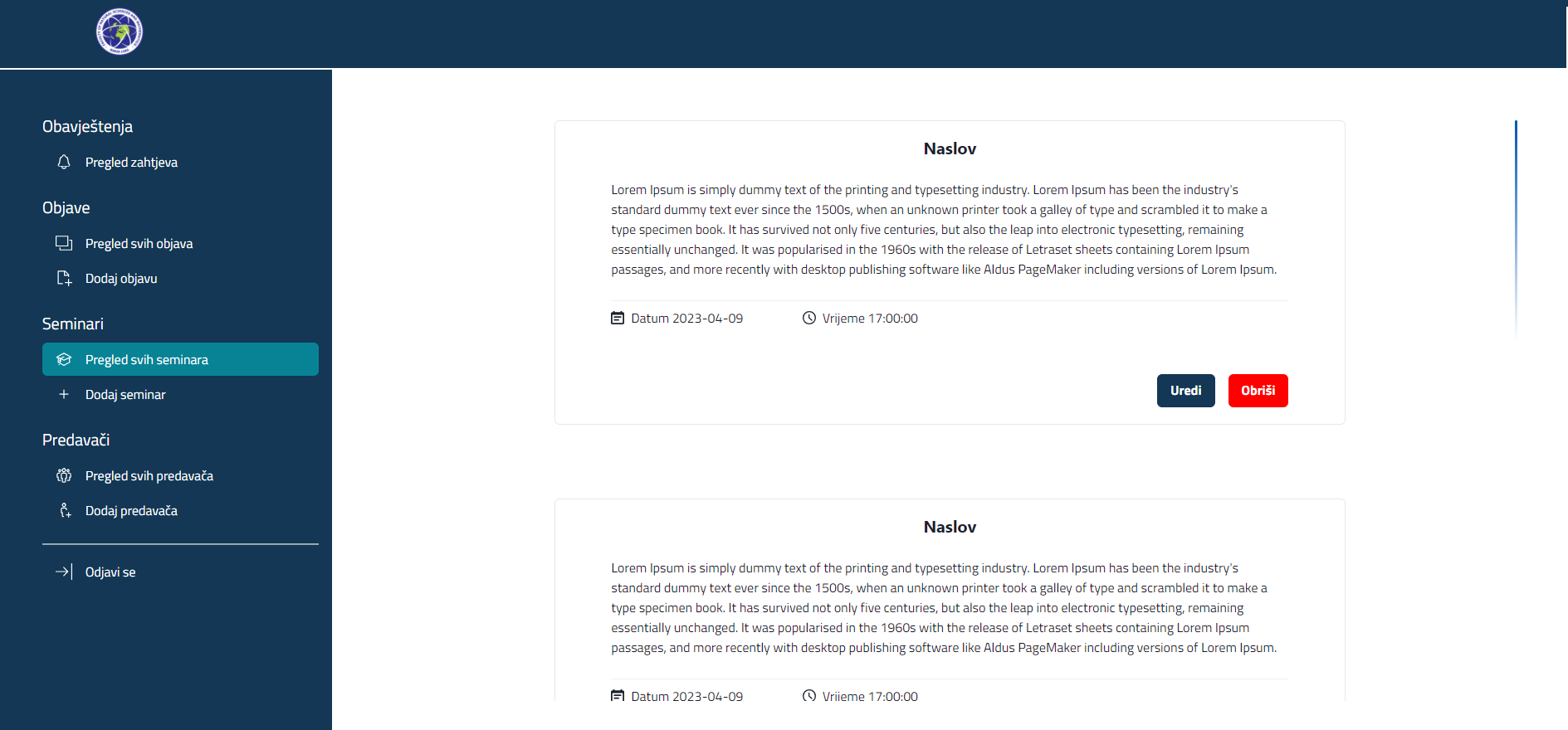
1. Prikaz svih zahtjeva za kreiranje novog seminara ili postavljanje objave kao i mogućnost prihvatanja ili odbacivanje pojedinačnog zahjteva.
2. Pregled svih objava, brisanje ili izmjena objave.
3. Forma za unos nove objave.
4. Pregled svih seminara, brisanje ili izmjena pojedinačnog seminara.
5. Forma za kreiranje novog seminara.
6. Pregled svih predavača, brisanje ili izmjena predavača.
7. Forma za dodavanje novog profesora.
8. Filtliranje po vremenu održavanja seminara i mogućnost štampanja izvještaja.



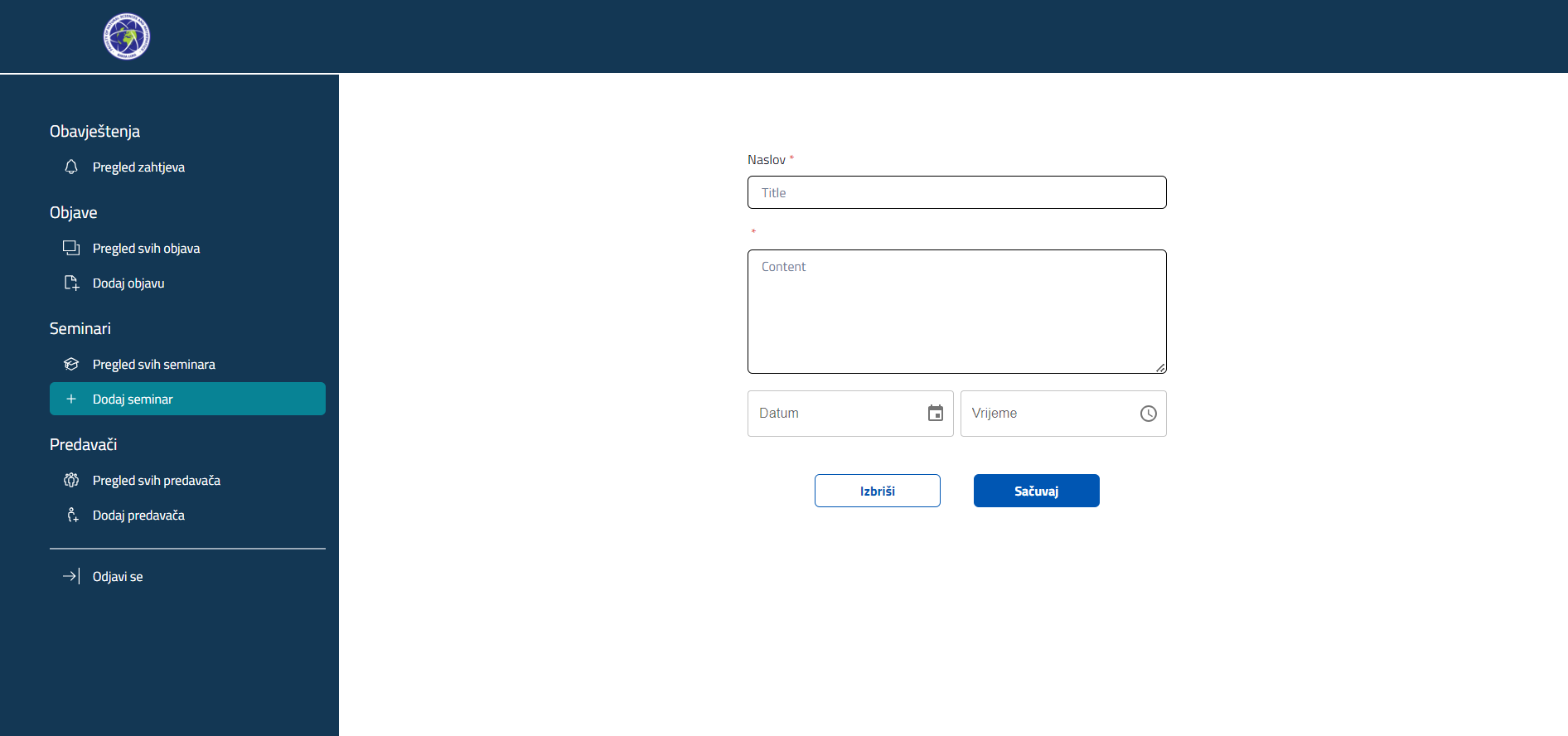
***Slika 4.8.1.*** *Prikaz adminske strane – pregled svih objava*



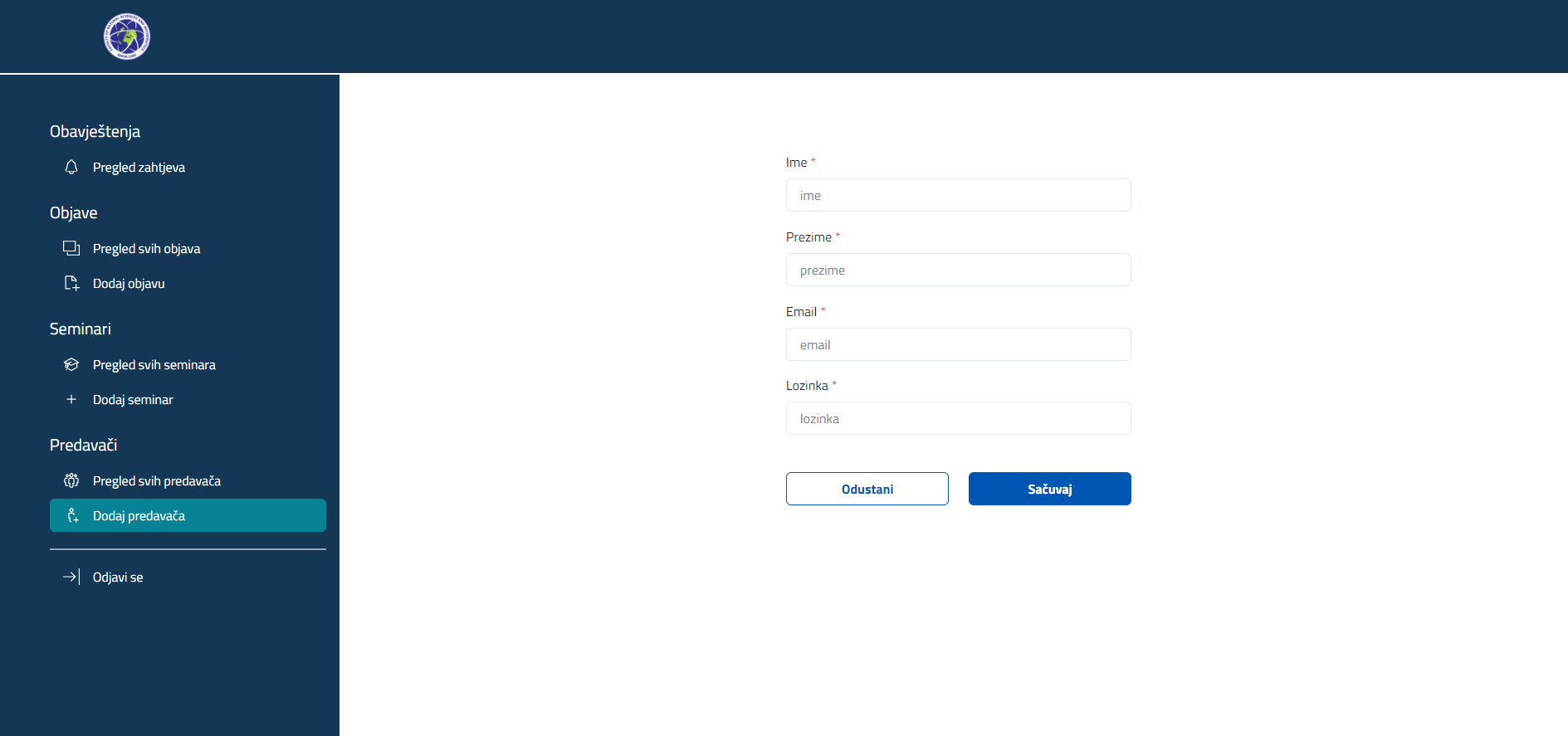
***Slika 4.8.2.*** *Prikaz adminske strane- dodavanje nove objave*



***Slika 4.8.4.*** *Prikaz adminske strane – pregled svih seminara*



***Slika 4.8.5.*** *Prikaz adminske strane – opcija za dodavanje novog seminara*



***Slika 4.8.6.*** *Prikaz adminske strane – opcija za dodavanje predavača*

# 6. Закључак

# 7. Литература

[1] Leon Shklar, Richard Rosen: *Web Application Arhitecture*.  
[2] Cory Gackenheimer: *Introduction to React*.  
[3] Naimul Islam Naim: *React JS: An Open Source JavaScript Library for Front-end Developement.*  
[4] MDN contributors, “What is CSS?”: [*https://developer.mozilla.org/en-US/docs/Learn/CSS/First\_steps/What\_is\_CSS*](https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/What_is_CSS) [приступано 16.11.2023.].  
[5] Eric A. Meyer: *CSS:The Definitive Guide, Third Edition*.  
[6] ReactJS.org, ‘ReactJS official’: *http://www.ReactJs.org* [приступано 16.11.2023.].  
[7] Doel Sengupta, Manu Sighal, Danillo Corvalan: *Getting Started with React.*  
[8] Alex Banks, Eve Porcello: *Learning React – Functional Web Development with React and Redux.*  
[9] ReactJS.org, 'Introducing JSX': [*https://legacy.reactjs.org/docs/introducing-jsx.html*](https://legacy.reactjs.org/docs/introducing-jsx.html)[ приступано 29.11.2023.].  
[10] K. Siva Prasad Reddy: *Beginning Spring Boot 2: Applications and Microservices with the Spring Framework.*[11] Visual Studio  
[12] Apache Maven Project, „What is Appache Maven?“ : [*https://maven.apache.org/what-is-maven.html*](https://maven.apache.org/what-is-maven.html) [приступано 22.11.2023.].  
[13] Oracle Corporation, „The Java Programming Language“: [*https://docs.oracle.com/javase/specs/jls/se8/html/*](https://docs.oracle.com/javase/specs/jls/se8/html/) [приступано 22.11.2023].  
[14] Spring Boot Official Documentation: [*https://docs.spring.io/spring-boot/docs/1.1.4.RELEASE/reference/html/getting-started-first-application.html*](https://docs.spring.io/spring-boot/docs/1.1.4.RELEASE/reference/html/getting-started-first-application.html) [приступано 30.11.2023.].  
[15] Eclipse Foundatio, „Eclipse IDE“ : [*https://www.eclipse.org/ide/*](https://www.eclipse.org/ide/)[приступано 30.11.2023.].  
[16] Joshua D. Drake, John C. Worslay: *Practical PostgreSQL*.  
[17] Spring Framework, Getting szarted with Spring Boot: [*https://spring.io/guides/gs/spring-boot/*](https://spring.io/guides/gs/spring-boot/)[приступано 01.12.2023.]