Aplikacija za predviđanje Raspored ispitnih rokova

Dundara, Darko

Sveučilište Jurja Dobrile u Puli
Tehnički fakultet u Puli

DARKO DUNDARA

Aplikacija za predviđanje rasporeda ispitnih rokova

Završni rad

Pula, rujan, 2023. godine

Sveučilište Jurja Dobrile u Puli
Tehnički fakultet u Puli

DARKO DUNDARA

Aplikacija za predviđanje rasporeda ispitnih rokova

Završni rad

JMB:0303098838, redoviti studen

Studijski smjer: Sveučilišni preddiplomski studij Računarstva

Predmet: Inženjerska grafika i konstruiranje

Znanstveno područje: Tehničke znanosti

Znanstveno polje: Računarstvo

Mentor: Doc. dr. Sc. tech. Marko Kršulja

Pula, rujan, 2023. godine

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani Darko Dundara, kandidat za prvostupnika, smjera Računarstva ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student

U Puli, 28.09.2023 godine Darko Dundara

IZJAVA

o korištenju autorskog djela

Ja, Darko Dundara dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj završni rad pod nazivom_Aplikacija za predviđanje rasporeda ispitnih rokova, koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

Potpis

U Puli, 28.09.2023 godine

Darko Dundara

Sadržaj

1. Uvod

Aplikacija za predviđanje rasporeda ispitnih rokova zamišljena je da prati unos i registraciju novih članova sustava, unos ispitnih rokova, dvorana, kolegija, te omogućuje detaljan uvid u sve te aktivnosti. Naglasak sustava je da prati i neke dodatne mogućnosti kao što su: nedozvoljavanje unosa ispitnog roka na datum koji je već zauzet nekim drugim ispitnim rokom, nedozvoljavanje unosa ispitnih rokova na praznike (blagdane) i sl. Aplikacija za predviđanje rasporeda ispitnih rokova je korisna aplikacija ili alat koji omogućuje učenicima, nastavnicima i školama olakšano upravljanje rasporedom školskih aktivnosti. Ova platforma omogućuje kreiranje, ažuriranje i praćenje školskih rasporeda na jednostavan i efikasan način. Jedna od glavnih prednosti Aplikacije za predviđanje rasporeda ispitnih rokova je mogućnost prilagodbe rasporeda prema individualnim potrebama i zahtjevima svake škole ili čak svakog učenika. Korisnici mogu lako unositi informacije o predmetima, učiteljima, učionicama i vremenima nastave te generirati rasporede koji su optimalno prilagođeni školskom okruženju. Osim toga, Aplikacija za predviđanje ispitnih rokova često nudi dodatne funkcionalnosti kao što su obavijesti o promjenama u rasporedu, mogućnost sinkronizacije s drugim kalendarima (kao što su Google Calendar ili iCal) te izvještaji o prisutnosti i izostancima učenika. Sve to doprinosi boljoj organizaciji školskih aktivnosti i smanjenju stresa povezanog s upravljanjem rasporedom. Kako biste iskoristili sve prednosti Aplikacije za predviđanje rasporeda ispitnih rokova, važno je da se detaljno upoznate s njegovim funkcionalnostima i prilagodite ga specifičnim potrebama vaše škole ili obrazovne ustanove. Uz ovakav alat, olakšavate procese planiranja i omogućujete bolje iskustvo kako za učenike tako i za nastavnike.

1.1. Hipoteza

Osnovna hipoteza je da će se istražiti mogućnost izrade softverskog rješenja za rezervaciju prostorija za potrebe nastave i ispitnih rokova na sveučilištu Jurja Dobrile u Puli.

Aplikacija trenutačno nije na razini profesionalnih web stranica, što omogućava niže cijene mojih usluga. Očekuje se poboljšanje mojih vještina i kvalitete usluge tijekom vremena rezultirati povećanjem cijena. Smatram da će analizom i rješavanjem nedostataka profesionalnih web stranica moje usluge postati konkurentnije i privući više klijenata. Ova hipoteza bit će testirana kroz stvarna iskustva i povratne informacije klijenata kako bi se utvrdio je li ta strategija održiva i učinkovita. "Programi koje sam koristio u razvoju ovog projekta, kao što su C#, .NET Core, HTML, CSS i MSSQL, su besplatni alati koji omogućuju pristup mnogim funkcionalnostima potrebnim za izradu sofisticiranih web aplikacija.

Osim ovih besplatnih alata, na tržištu postoje i plaćeni alati kao što su Adobe Dreamweaver, WebStorm i Microsoft Visual Studio, koji također nude niz prednosti i nedostataka.

Prednosti besplatnih alata poput C# i .NET Core smanjuju troškove razvoja, posebno za manje timove i pojedince. Velika zajednica: Besplatni alati često imaju velike zajednice korisnika i razvojnih zajednica koje pružaju podršku i obilje resursa za učenje. Otvoreni izvor (open source): Otvoreni izvor omogućava pristup izvornom kodu, što omogućava prilagodbu i prilagođavanje specifičnim potrebama. Međutim, plaćeni alati poput Adobe Dreamweaver i Microsoft Visual Studio također imaju svoje prednosti. Plaćeni alati često nude napredne značajke i integraciju s drugim profesionalnim alatima. Brži razvoj: Integrirane razvojne okoline (IDE) često olakšavaju brži razvoj aplikacija. Podrška i sigurnost: Plaćeni alati obično dolaze s profesionalnom podrškom i ažuriranjima za sigurnost. Vizualni uređivači: Alati poput Adobe Dreamweaver nude vizualne uređivače koji olakšavaju dizajniranje web stranica bez potrebe za kodiranjem. Nedostaci ovih plaćenih alata uključuju visoke troškove licenci, posebno za komercijalnu upotrebu, i moguću manju zajednicu korisnika u usporedbi s besplatnim alatima.

1.2. Problem istraživanja

Selekcija sudionika: Odabir pravilnog uzorka sudionika može biti izazov, jer će trebati uključiti učitelje, učenike i administrativno osoblje škole. Osiguranje da uzorak bude reprezentativan i adekvatan može biti teško, a nedostatak raznolikosti može utjecati na opću primjenjivost rezultata. Mjerenje učinka: Definiranje i mjerenje učinka ili poboljšanja koja se očekuju od Aplikacije za predviđanje rasporeda ispitnih rokova može biti složeno. Na primjer, kako ćete kvantificirati bolju organizaciju ili smanjenje stresa za nastavnike i osoblje? Precizno definiranje varijabli može biti izazov. Vrijeme istraživanja: Trajanje istraživanja može biti dugotrajno jer će trebati pratiti učinke Aplikacije za predviđanje rasporeda ispitnih rokova tijekom dužeg razdoblja. Ovo može biti logistički izazovno i zahtijevati kontinuiranu suradnju sa školama. Utjecaj drugih faktora: Školsko okruženje je kompleksno, a mnogi faktori mogu utjecati na organizaciju i učinkovitost. Teško je izolirati utjecaj Aplikacije za predviđanje rasporeda ispitnih rokova od drugih faktora koji mogu igrati ulogu u školskom uspjehu. Mjerenje zadovoljstva korisnika: Ispitivanje zadovoljstva korisnika može biti subjektivno i oslanjati se na njihove osobne percepcije i iskustva. Ovo može biti sklono pristranostima i ne uvijek odražavati objektivne promjene. Financijski resursi: Provoditi istraživanje može zahtijevati financijske resurse za prikupljanje podataka, analizu podataka i suradnju sa školama. Nedostatak financija može ograničiti opseg istraživanja. Etika i privatnost:
Uključivanje učitelja, učenika i osoblja u istraživanje zahtijeva poštivanje etičkih smjernica i zaštite privatnosti. Potrebno je osigurati suglasnost i povjerljivost podataka. Vremenski okvir implementacije: Implementacija Aplikacije za predviđanje rasporeda ispitnih rokova može potrajati i biti promjenjiva u različitim školama.

To može utjecati na dosljednost i usporedivost rezultata između različitih institucija. Rješavanje ovih problema zahtijeva pažljivo planiranje, metodologiju istraživanja i suradnju sa školama kako bi se osigurala valjanost i relevantnost rezultata istraživanja Aplikacije za predviđanje rasporeda ispitnih rokova. Te probleme uspio sam riješiti pomoću C#, .NET Core, HTML, CSS i MSSQL te njihovih dokumentacija, i knjižnica koje su mi znatno pomogli pronalaženjem rješavanja problema.

1.3. Predmet istraživanja

Zakazivanje Algoritama: Ovo je osnovno područje istraživanja za Aplikaciju za predviđanje rasporeda ispitnih rokova. Treba istražiti različite algoritme za zakazivanje kako biste pronašli onaj koji najbolje odgovara potrebama škole. To uključuje istraživanje algoritama kao što su genetski algoritmi, algoritmi sažimanja, algoritmi pohlepnog pristupa itd. Školske Potrebe i Ograničenja: Važno je razumjeti specifične potrebe i ograničenja škole. To uključuje informacije o broju učenika, nastavnicima, razredima, predmetima, dostupnim resursima, vremenima kad su resursi dostupni itd. Optimizacija Rasporeda: Istraživanje optimizacijskih tehnika kako bi se postigao najbolji raspored, uzimajući u obzir različite faktore kao što su minimalno preklapanje vremena, preferencije nastavnika, maksimizacija resursa i slično. Koristan Softver: Proučavanje postojećih softverskih alata za raspoređivanje u obrazovanju. Analiza njihovih prednosti i nedostataka kako biste identificirati moguće poboljšanja i razlike u odnosu na vašu aplikaciju. Korisničko Iskustvo (UX): Istraživanje korisničkog iskustva i dizajna korisničkog sučelja kako biste osigurali da aplikacija bude jednostavna za korištenje i efikasna za korisnike. Podaci i Sigurnost: Proučavanje kako najbolje upravljati podacima u vezi sa rasporedom, uključujući zaštitu podataka i privatnost korisnika. Implementacijske Tehnologije: Istraživanje tehnologija i okvira za razvoj aplikacije, kao što su programski jezici, baze podataka, serveri, i slično. Evaluacija Performansi: Definiranje metrika za mjerenje performansi aplikacije, kao što su brzina generiranja rasporeda, responzivnost aplikacije, skalabilnost i druge. Regulatorni Aspekti: Ako se koristi u formalnom obrazovanju, potrebno je istražiti zakone i regulative koje se odnose na raspoređivanje školskih aktivnosti i sigurnost podataka. Obuka i Implementacija: Razmislite o strategijama obuke za korisnike i kako ćete implementirati aplikaciju u školskom okruženju.

2. Korištena arhitektura

MVC je poznat kao arhitektonski obrazac, koji utjelovljuje tri dijela Model, View i Controller. Korišten je za grafička korisnička sučelja stolnih računala, ali danas se koristi u dizajniranju mobilnih i web aplikacija.

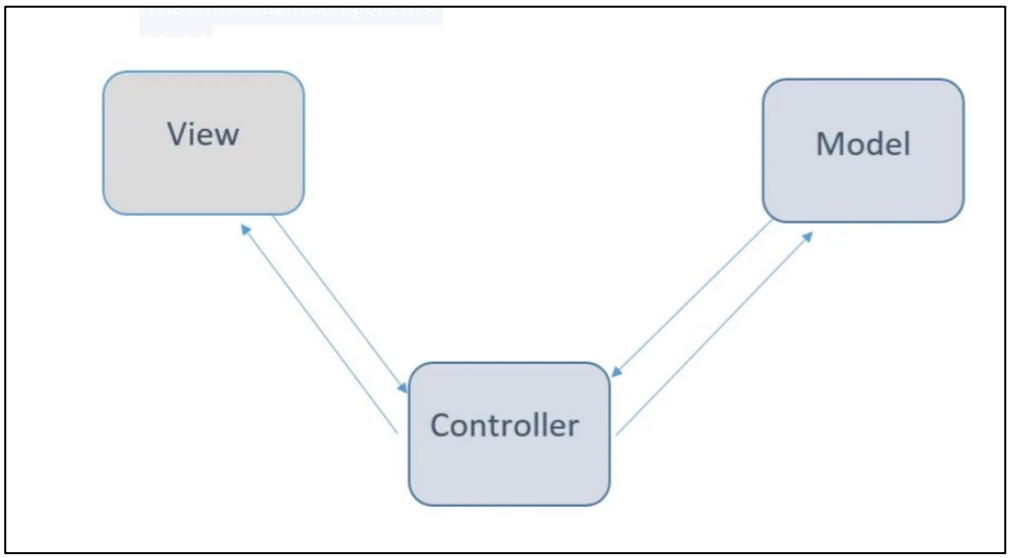
MVC je arhitektonski obrazac što znači da upravlja cijelom arhitekturom aplikacija. Iako je često poznat kao uzorak dizajna, možda griješimo ako ga nazivamo samo uzorkom dizajna jer se obrasci dizajna koriste za rješavanje specifičnog tehničkog problema, dok se arhitektonski uzorak koristi za rješavanje arhitektonskih problema, pa utječe na cijeli arhitektura naše aplikacije.

Ima tri glavne komponente:

- **Model**
- **Pogled**
- **Kontroler**

a svaki od njih ima specifične odgovornosti.

Slika 1 MVC Arhitektura



Izvor: https://www.kodeco.com/

2.1. Što je to MVC arhitektura?

MVC je komponenta koja čuva podatke i sadrži logiku za njihovu obradu. Ovo uključuje: Manipulaciju podacima: Dodavanje, čitanje, ažuriranje i brisanje podataka. Poslovnu logiku: Izvođenje operacija i proračuna nad podacima. Komunikaciju sa skladištima podataka: Povezivanje s bazama podataka ili eksternim servisima za dohvat ili pohranu podataka. View je komponenta koja je odgovorna za prikazivanje podataka korisnicima. Ovo uključuje, grafički korisnički interfejs (GUI) ili HTML šablone. Prikazivanje podataka na način koji je razumljiv i privlačan korisnicima.
Osvježavanje prikaza kada se podaci promjene. Controller je posrednička komponenta između Modela i View-a. Njegove odgovornosti uključuju, primanje korisničkih zahtijeva. Upućivanje zahtjeva Modelu za dohvat ili ažuriranje podataka. Ažuriranje View-a s novim podacima.

Validaciju korisničkog unosa i upravljanje greškama. Donošenje odluka o tome kako će aplikacija reagirati na korisničke akcije.

Prednosti MVC Arhitekture su Razdvajanje odgovornosti: Razdvaja logiku za obradu podataka od prikazivanja podataka, što olakšava održavanje i skaliranje aplikacije. Fleksibilnost i ponovna upotreba koda: Omogućava različitim timovima da rade nezavisno na Modelu, View-u i Controller-u.Testiranje: Omogućava lakše testiranje svake komponente nezavisno. Bolje upravljanje kompleksnošću: Ograničava kompleksnost svake komponente tako da je lakše razumjeti i održavati. Dakle MVC arhitektura je moćan obrazac za razvoj softverskih aplikacija koji pomaže u organizaciji koda i olakšava održavanje i razvoj. Razumijevanje uloga Modela, View-a i Controller-a ključno je za efikasno korištenje ovog obrasca u vašim projektima.

2.2. Korištene tehnologije

U ovom projektu korištene su sljedeće tehnologije:

- **C#** programski jezik
- **.NET core** – open source platforma za razvoj
- **HTML** kao jezik za prikaz korisničkog sučelja
- **CSS** kao stil za uređivanje
- **MSSQL** – Microsoft baza za pohranu svih podataka na sustavu

2.3. Povijest C#

C# je programski jezik razvijen od strane Microsofta i predstavljen javnosti 2000. godine. Ovaj jezik je osmišljen kako bi omogućio razvoj modernih aplikacija za Windows i web okruženja. U ovom dokumentu ćemo istražiti ključne točke u povijesti C# jezika i njegov utjecaj na svijet razvoja softvera. Razvoj C# jezika započeo je u drugoj polovini 1990-ih pod nazivom "Cool" (C-like Object-Oriented Language). Tim Microsoftovih inženjera predvođen Andersom Hejlsbergom bio je odgovoran za razvoj jezika.C# je prvi put predstavljen javnosti 2000. godine kao ključni dio Microsoftove .NET platforme. Ovaj jezik je osmišljen kako bi objedinio najbolje aspekte drugih popularnih programskih jezika poput C++, Jave i Visual Basic. To je rezultiralo jezikom koji je objektno usmjeren, snažan, siguran i jednostavan za razumijevanje i koristiti. Godine 2003., C# je postao standardiziran putem Ecma Internationala i ISO/IEC (International Organization for Standardization/International Electrotechnical Commission).

Ovo je povećalo prihvaćenost jezika i omogućilo njegovu upotrebu izvan Microsoftovih ekosustava. C# je brzo stekao popularnost i postao ključni jezik za razvoj Windows aplikacija, web aplikacija (putem ASP.NET-a) i igara (putem razvojnog okvira Unity3D).

Također je postao popularan u korporativnom okruženju. Tijekom godina, C# je doživio brojna izdanja i nadogradnje, svaka donoseći nove značajke i poboljšanja jezika kako bi se olakšao razvoj suvremenih aplikacija. Unatoč tome što je razvijen od strane Microsofta, C# je postao jezik s velikom otvorenom zajednicom programera. Otvoreni izvorni kodni projekti poput Mono i .NET Core omogućili su C# razvoj na različitim platformama osim Windowsa. Danas, C# je jedan od najpopularnijih programskih jezika na svijetu. I dalje se aktivno razvija i koristi za razvoj raznolikih aplikacija, uključujući Windows aplikacije, web aplikacije, mobilne aplikacije i igre. C# programski jezik ima bogatu povijest i ostvario je dubok utjecaj na svijet razvoja softvera. Njegova sposobnost za razvoj modernih i sigurnih aplikacija ga čini neprocjenjivim alatom za programere širom svijeta. Microsoft i otvorena zajednica programera nastavljaju raditi na njegovom razvoju kako bi ga učinili još snažnijim i prilagodljivijim za buduće izazove u svijetu razvoja softvera.

2.4. Povijest .NET core-a

NET Core je moderna i open-source platforma za razvoj i izvođenje aplikacija razvijena od strane Microsofta. Ova platforma je postala ključna komponenta razvoja softvera, omogućujući razvoj aplikacija koje su Inter operabilne, skalabilne i prijenosne na različite operativne sustave. U ovom dokumentu istražit ćemo ključne trenutke u povijesti .NET Core platforme i njezin utjecaj na razvoj softvera, .NET Core platforma ima svoje korijene u .NET Frameworku, koji je prvi put predstavljen krajem 1990-ih. Međutim, potreba za modernijom i otvorenijom platformom postala je očita u drugoj polovini 2000-ih, .NET Core 1.0 Prva verzija .NET Core-a, poznata kao .NET Core 1.0, predstavljena je 2016. godine. Ovo izdanje označilo je značajan korak prema otvorenom izvoru i podršci za različite operativne sustave. Razvojni inženjeri mogli su sada razvijati aplikacije koje su prijenosne na Windows, Linux i macOS. Nakon prvog izdanja, .NET Core je brzo rastao i evoluirao. Microsoft i otvorena zajednica programera surađivali su na razvoju platforme, donoseći brojne nadogradnje i poboljšanja. Otvoreni izvorni kod (.NET Core je postao .NET 5 i .NET 6. Ova izdanja donose poboljšanja u performansama, skalabilnosti i kompatibilnosti s postojećim kodom. Izdani 2020. i 2021. godine, .NET 5 i .NET 6 predstavljaju moderna i unificirana izdanja platforme. Ova izdanja objedinjuju razvoj za različite platforme, uključujući desktop, web i mobilne aplikacije, što čini .NET Core (sada samo .NET) sveprisutnom platformom za razvoj aplikacija. Jedna od ključnih karakteristika .NET Core-a je snažna otvorena zajednica programera koja je doprinijela njegovom uspjehu. Otvoreni izvorni kodni projekti, poput Entity Framework Core i ASP.NET Core, omogućili su programerima stvaranje bogatih i funkcionalnih aplikacija. Danas, .NET platforma (uključujući .NET 6) ostaje jedna od najpopularnijih i najkorištenijih platformi za razvoj softvera širom svijeta.

Ona je temelj za razvoj aplikacija na različitim platformama, uključujući Windows, Linux i macOS, te podržava različite jezike, uključujući C#, F# i Visual Basic.
.NET Core platforma je evoluirala iz pionirskih koraka u modernu, inter operabilnu i otvorenu platformu za razvoj softvera. Njezin utjecaj na razvoj aplikacija i tehnologiju je dubok i neprocjenjiv. Microsoft i zajednica programera nastavljaju raditi na njenom razvoju kako bi je učinili još snažnijom i prilagodljivijom za buduće izazove u svijetu razvoja softvera.

2.5. Povijest Html-a

HTML je jezik koji služi za strukturiranje i oblikovanje web stranica. Ovaj jezik igra ključnu ulogu u World Wide Webu (WWW), omogućujući stvaranje interaktivnih i povezanih sadržaja na internetu. U ovom dokumentu ćemo istražiti ključne trenutke u povijesti HTML-a i njegovu evoluciju kao osnovnog jezika web razvoja. Povijest HTML-a počinje s SGML-om (Standard Generalized Markup Language) kao pretečom. Tim Berners-Lee, tvorac WWW-a, razvio je prvu verziju HTML-a, poznatu kao HTML 1.0, 1991. godine. Ova verzija omogućila je povezivanje dokumenata putem hiperlinka i označavanje sadržaja putem osnovnih HTML tagova. HTML 2.0, objavljen 1995. godine, uveo je nove elemente i tagove za bolju kontrolu sadržaja. Međutim, standardizacija je bila problematična jer su različiti preglednici interpretirali HTML različito. Slijedio je razdoblje "ratova preglednika" (Browser Wars) kada su Microsoftov Internet Explorer i Netscape Navigator konkurirali za dominaciju na webu. To je dovelo do fragmentacije standarda. Tek 1997. godine HTML 3.2 postao je zvanični standard, ali problemi s kompatibilnošću nisu nestali. HTML 4.01, objavljen 1999. godine, donio je napredne mogućnosti za oblikovanje i stilizaciju web stranica. Paralelno, XHTML (Extensible HyperText Markup Language) je razvijen kao prvi korak prema strožoj sintaksi i XML-u (eXtensible Markup Language). Najznačajnija prekretnica u povijesti HTML-a je dolazak HTML5, objavljen 2014. godine. Ovaj standard donosi mnoge nove elemente, API-je, i mogućnosti, uključujući video i audio integraciju, lokalnu pohranu podataka (localStorage), i mnoge druge. HTML5 je postao temelj za moderni web razvoj i omogućio razvoj bogatih web aplikacija. Iako HTML igra ključnu ulogu u strukturiranju web sadržaja, razvoj web stranica često uključuje i CSS (Cascading Style Sheets) za stilizaciju i JavaScript za interaktivnost. Ovi tehnološki komplementi zajedno čine temelj modernog weba. HTML ostaje osnovni jezik web razvoja, a HTML5 i dalje služi kao standard za izradu web aplikacija. Buduće verzije HTML-a i dalje će se razvijati kako bi podržale napredne potrebe modernog weba. HTML je jezik koji je oblikovao način na koji koristimo internet. Njegova evolucija od skromnih početaka do modernih standarda omogućila je razvoj raznovrsnih web aplikacija i interaktivnih sadržaja koji oblikuju naše online iskustvo.

2.6. Povijest CSS-a

CSS je jezik za stilizaciju web stranica koji omogućuje programerima i dizajnerima da definiraju izgled i prezentaciju web sadržaja. Ovaj jezik igra ključnu ulogu u oblikovanju modernih web stranica i pruža mogućnosti za stvaranje atraktivnih i funkcionalnih web iskustava. U ovom dokumentu ćemo istražiti ključne trenutke u povijesti CSS-a i njegovu evoluciju kao temeljnog dijela web razvoja.
Prije pojave CSS-a, web stranice su oblikovane isključivo pomoću HTML-a (HyperText Markup Language). Međutim, održavanje i ažuriranje dizajna web stranica postajalo je sve teže jer je izgled bio čvrsto povezan s HTML strukturama. To je stvorilo potrebu za odvojenim jezikom koji bi omogućio bolju kontrolu nad stilizacijom. Prva specifikacija CSS-a, poznata kao CSS 1, objavljena je 1996. godine. Ova specifikacija uvela je osnovne mogućnosti za definiranje boja, fontova, margina i drugih stilskih svojstava elemenata na web stranici. Prve verzije preglednika počele su podržavati CSS, ali s ograničenim mogućnostima i često s nekompatibilnim implementacijama. CSS 2, objavljen 1998. godine, donio je mnogo naprednije mogućnosti za stilizaciju, uključujući pozicioniranje elemenata, slojevite pozadine i animaciju. Ovaj standard također je uveo pojmove "kaskadnosti" i "nasljeđivanja", koji su definirali kako se stilovi primjenjuju i nasljeđuju iznad strukture HTML-a.
Unatoč standardizaciji, razlike u implementacijama preglednika stvorile su probleme s kompatibilnošću. Web dizajneri su morali koristiti različite "hake" kako bi se nosili s različitim preglednicima, što je otežalo razvoj i održavanje web stranica. CSS3 je započeo modularni pristup, što znači da je specifikacija podijeljena u module, svaki s vlastitim funkcionalnostima. Ovo je omogućilo postupno uvođenje novih značajki i bolju kontrolu nad stilizacijom. CSS3 uključuje brojne dodatne mogućnosti poput gradijenata, okvira, sjena i transformacija. Danas, CSS ostaje ključni dio web razvoja i pruža programerima i dizajnerima alate za oblikovanje web stranica koje odražavaju njihovu kreativnost. Buduće verzije CSS-a nastavit će donositi nove značajke i tehnike za stvaranje bogatih web iskustava. Povijest CSS-a svjedoči o kontinuiranom razvoju i napretku u oblikovanju web sadržaja. Od svojih skromnih početaka do današnjeg statusa ključnog alata za web dizajn, CSS je postao neprocjenjiv dio stvaranja privlačnih i funkcionalnih web stranica.

2.7. Povijest MSSQL

MSSQL je sustav za upravljanje bazama podataka (DBMS) razvijen od strane Microsofta. Ovaj DBMS igra ključnu ulogu u pohrani i upravljanju podacima u mnogim organizacijama i aplikacijama diljem svijeta. U ovom dokumentu ćemo istražiti ključne trenutke u povijesti MSSQL-a i njegovu evoluciju kao moćnog i pouzdanog sustava za upravljanje podacima. Povijest MSSQL-a seže unatrag u 1980-te godine kada je Microsoft surađivao s tvrtkom Sybase kako bi stvorili svoju prvu verziju SQL Servera.

Tada se zvao Sybase SQL Server i bio je dostupan za operacijski sustav OS/2. Prva samostalna verzija Microsoft SQL Servera, SQL Server 4.2, objavljena je 1992. godine.

Ovo izdanje donijelo je podršku za Windows i omogućilo je kreiranje klijent-server aplikacija temeljenih na SQL Serveru. Tijekom 1990-ih i 2000-ih godina, SQL Server je prošao kroz brojne izmjene i nadogradnje koje su poboljšale njegovu pouzdanost, performanse i sigurnost. Izdanja poput SQL Server 7.0, SQL Server 2000 i SQL Server 2005 donijela su napredne značajke za baze podataka i poslovne aplikacije. SQL Server 2008, objavljen 2008. godine, donio je značajna poboljšanja u upravljanju poslovnim inteligencijama, poslovnom analitikom i skalabilnošću.

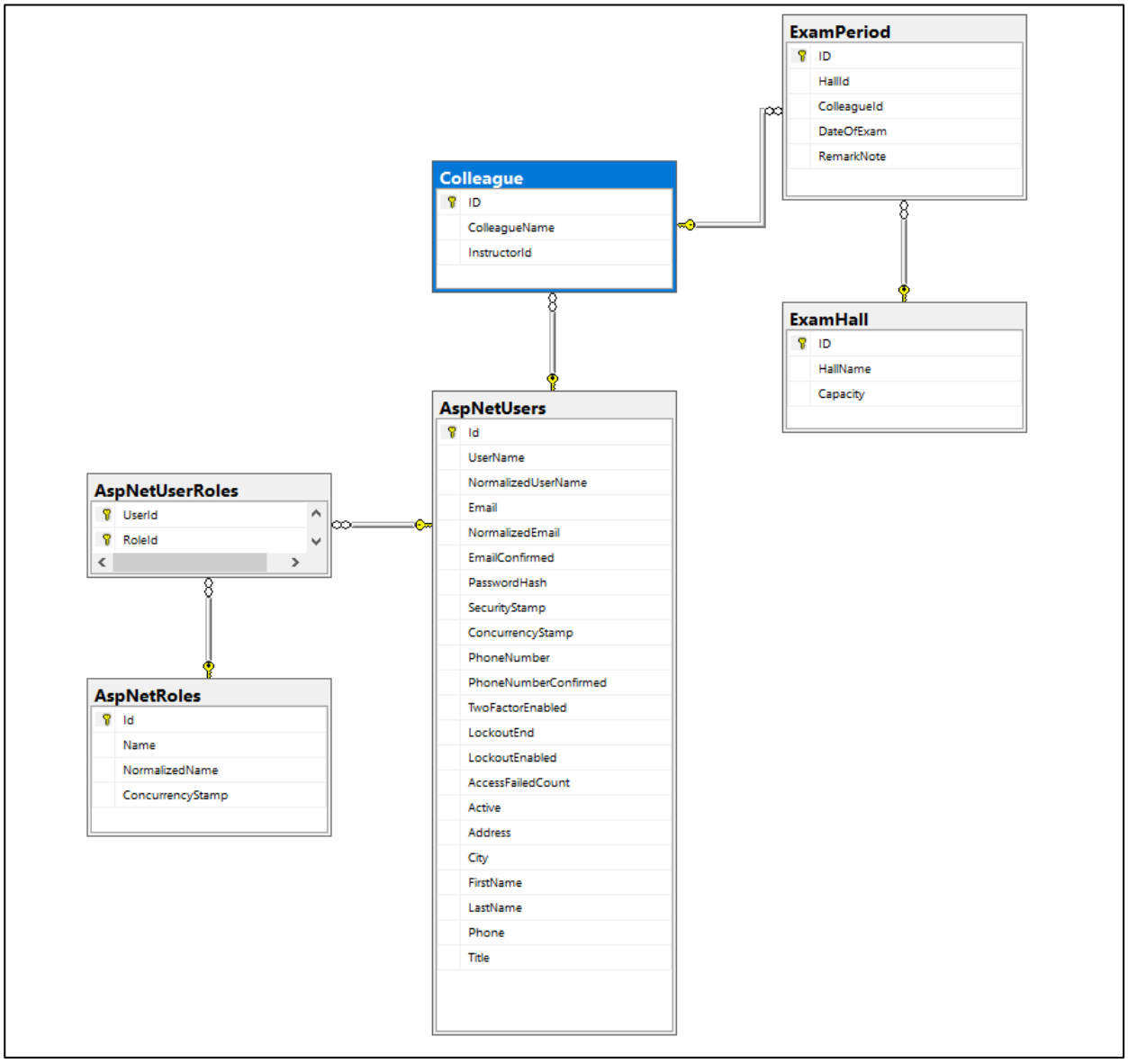
Ovo izdanje pomoglo je MSSQL-u, u uspostavljanju tržišne dominacije u segmentu baza podataka. SQL Server 2012, objavljen 2012. godine, uveo je inovacije poput AlwaysOn Availability Groups za visoku dostupnost i Power View za napredno izvještavanje.

Daljnja izdanja, uključujući SQL Server 2014, 2016 i 2017, donijela su poboljšanja u performansama, sigurnosti i analitici. SQL Server 2019, objavljen 2019. godine, donosi značajne promjene, uključujući integraciju s Microsoft Azure oblakom i podršku za Linux operativni sustav. Ovo izdanje omogućuje organizacijama da kombiniraju snagu lokalnih i oblak resursa za pohranu i analizu podataka. MSSQL je danas jedan od najpopularnijih i najkorištenijih DBMS-ova na svijetu. Microsoft i dalje aktivno razvija i nadograđuje SQL Server kako bi zadovoljio potrebe organizacija u digitalnoj eri. Buduće verzije MSSQL-a nastavit će donositi inovacije u upravljanju podacima i analitici. Povijest Microsoft SQL Servera svjedoči o njegovom kontinuiranom razvoju i evoluciji u moćan alat za pohranu, upravljanje i analizu podataka. Ovaj DBMS igra ključnu ulogu u podršci organizacijama širom svijeta u njihovim naporima za upravljanje i iskorištavanje podataka u današnjem digitalnom dobu.

2.8. Model podataka za bazu

ER model za ovaj sustav izgleda na sljedeći način:

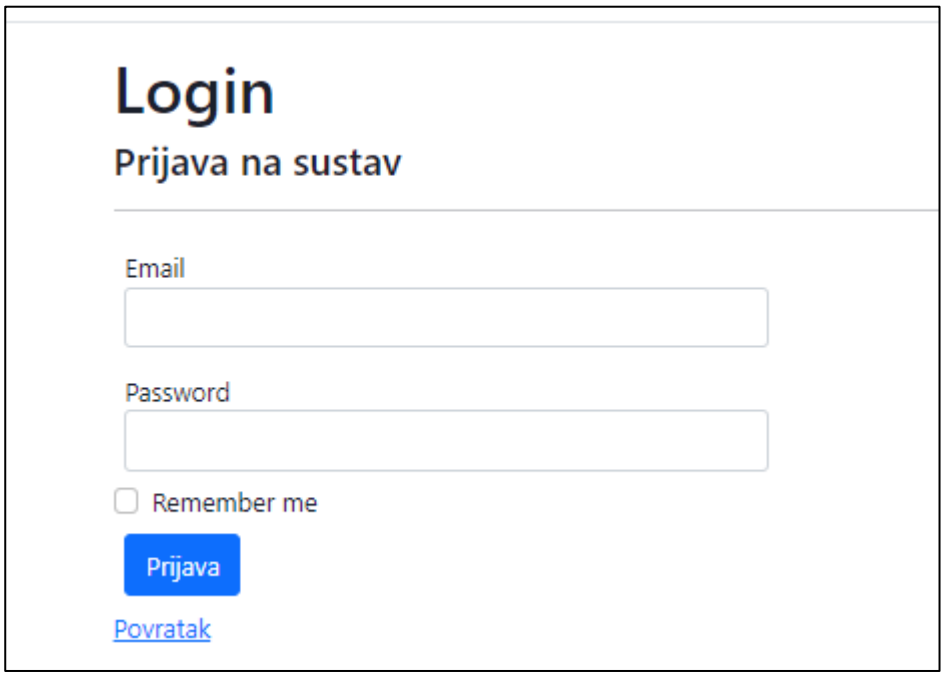
Slika 2 ER model



2.9. Prijava na sustav

Prijava na sustav dešava se klikom na Prijava na sustav u gornjem desnom kutu:

Slika 3 Prijava na sustav

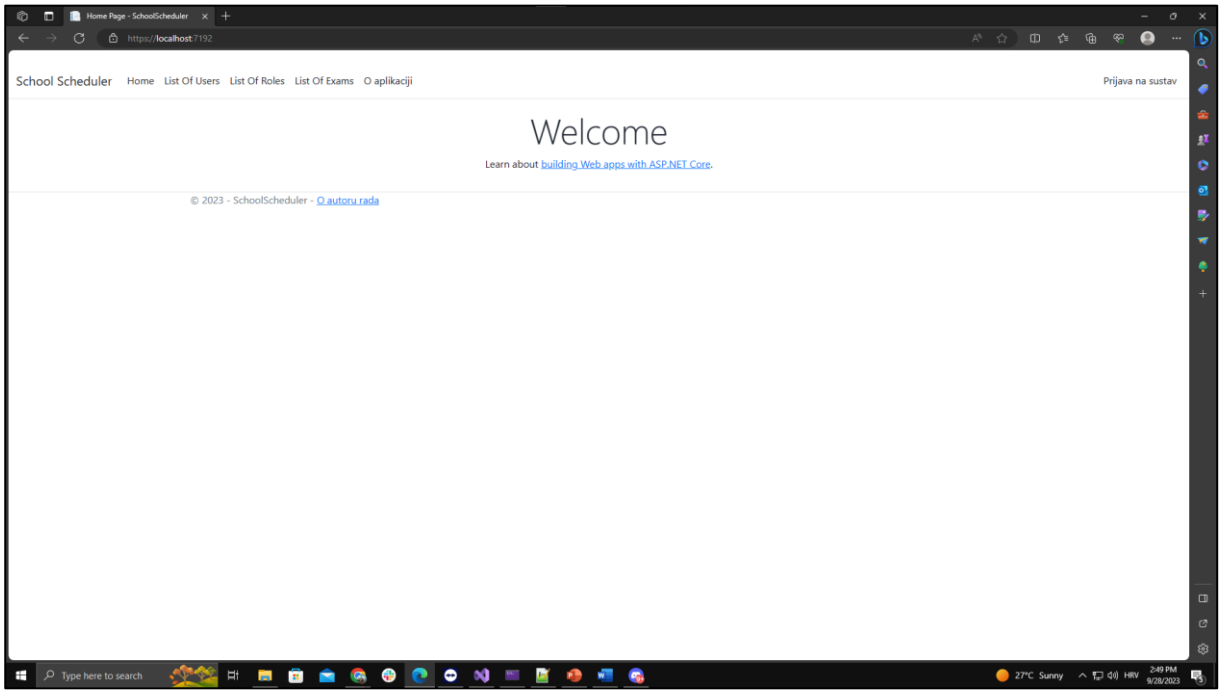


Slika 4 Source code Login metode

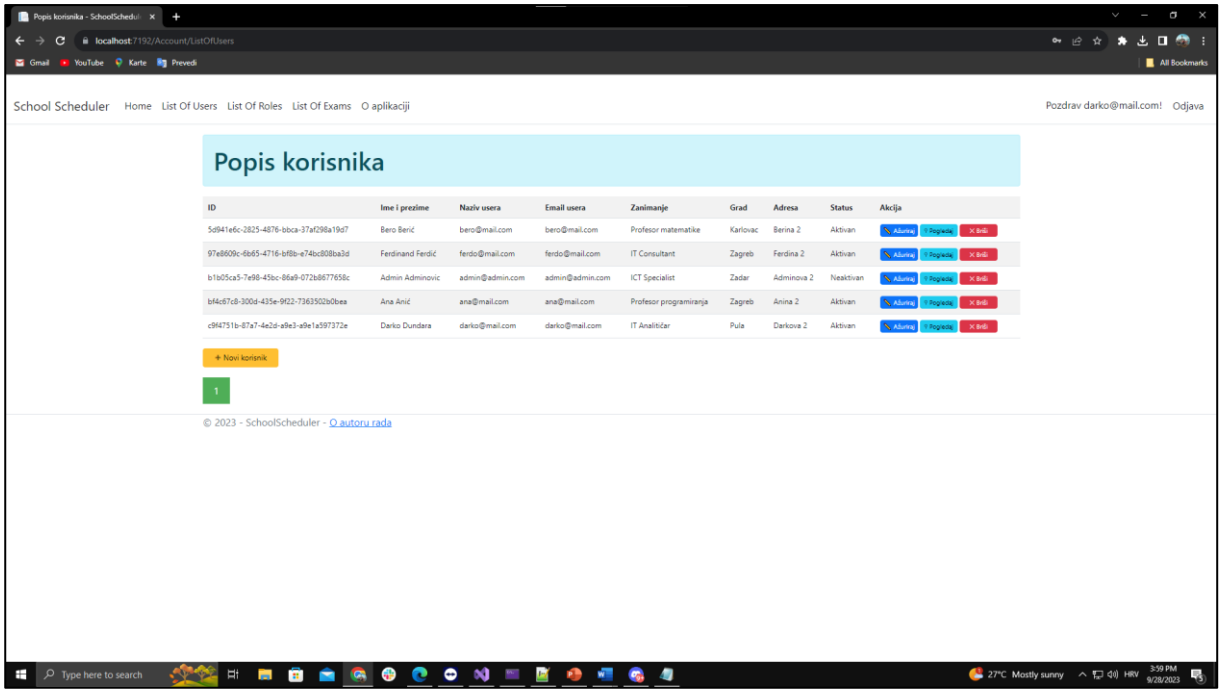
```
public IActionResult Login()
{
    return View();
}

[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Login(LoginViewModel model)
{
    if (ModelState.IsValid)
    {

        var result = await signInManager.PasswordSignInAsync(model.Email, model.Password, model.RememberMe, false);

        if (result.Succeeded)
        {
            var user = await userManager.FindByNameAsync(model.Email);
            await signInManager.SignInAsync(user, isPersistent: false);
            return RedirectToAction("Index", "Home");
        }

        ModelState.AddModelError(string.Empty, "Invalid login attempt");
    }
    return View(model);
}
```

2.10. Pregled ekrana

Slika 5 Pregled ekrana

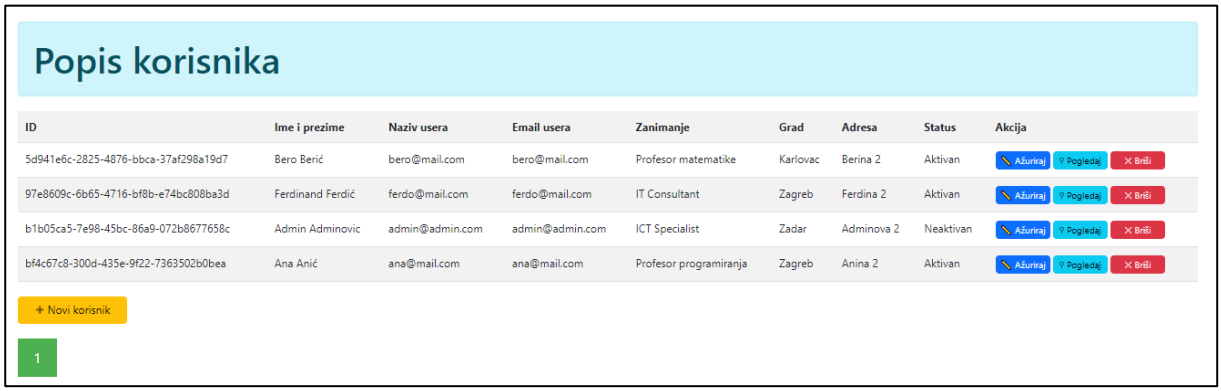


Slika 6 Pregled ekrana sa korisnicima



2.11. Pregled svih korisnika na sustavu

Slika 7 Pregled korisnika

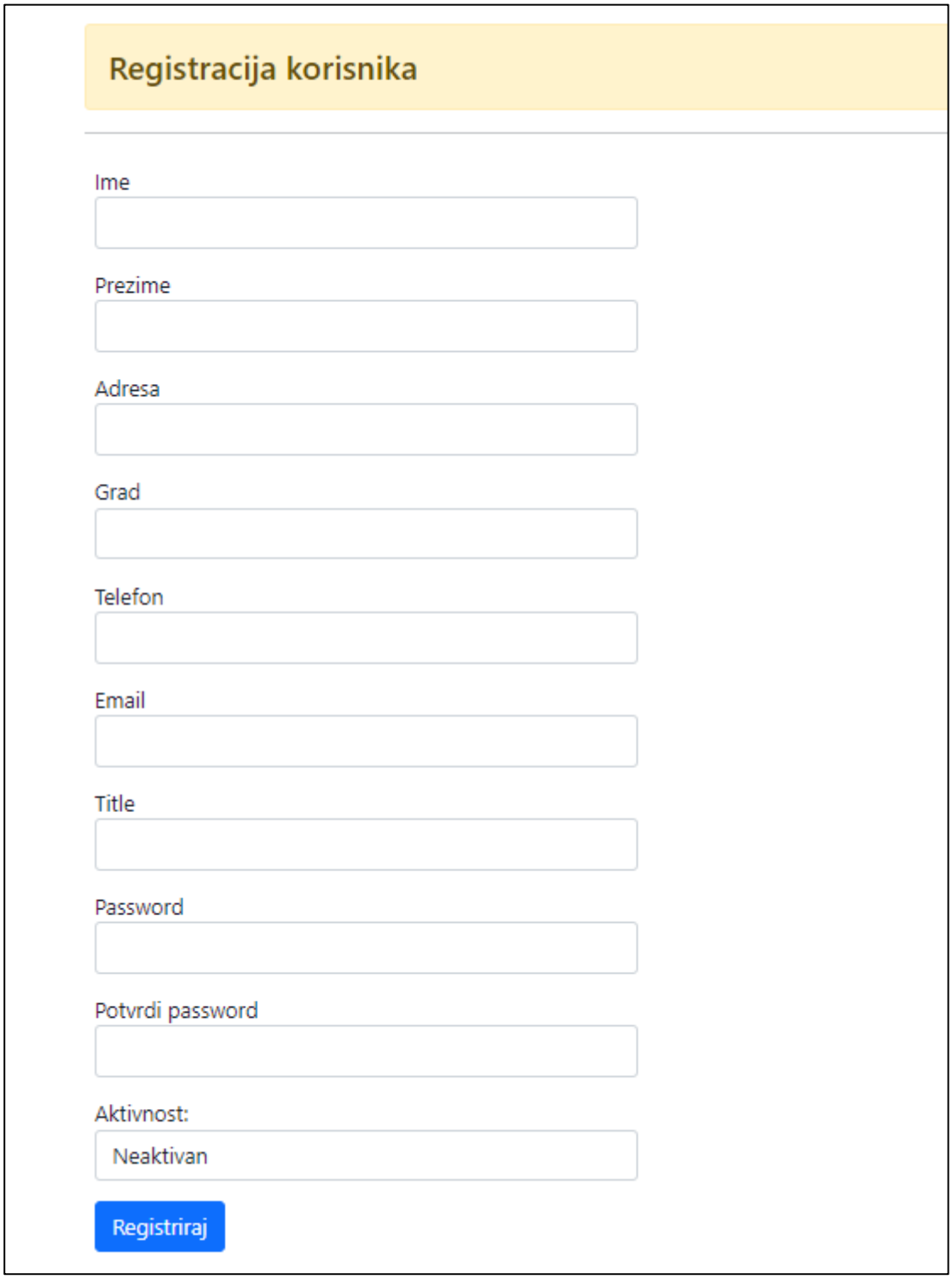


Slika 8 Source code za Popis korisnika

```
public IActionResult ListOfUsers(int page)
{
    var users = userManager.Users;

    double BrojRedaka = users.ToList().Count;
    int BrojStranica = (int)Math.Ceiling(BrojRedaka / 5);

    ViewBag.BrojStranica = BrojStranica;
    int aktivna;

    if (page == 0 || page == 1)
    {
        users = users.Take(5);
        aktivna = 1;
    }
    else
    {
        users = users.Skip((page - 1) * 5).Take(5);
        aktivna = page;
    }
    ViewBag.Aktivna = aktivna;


    return View(users);
}
```

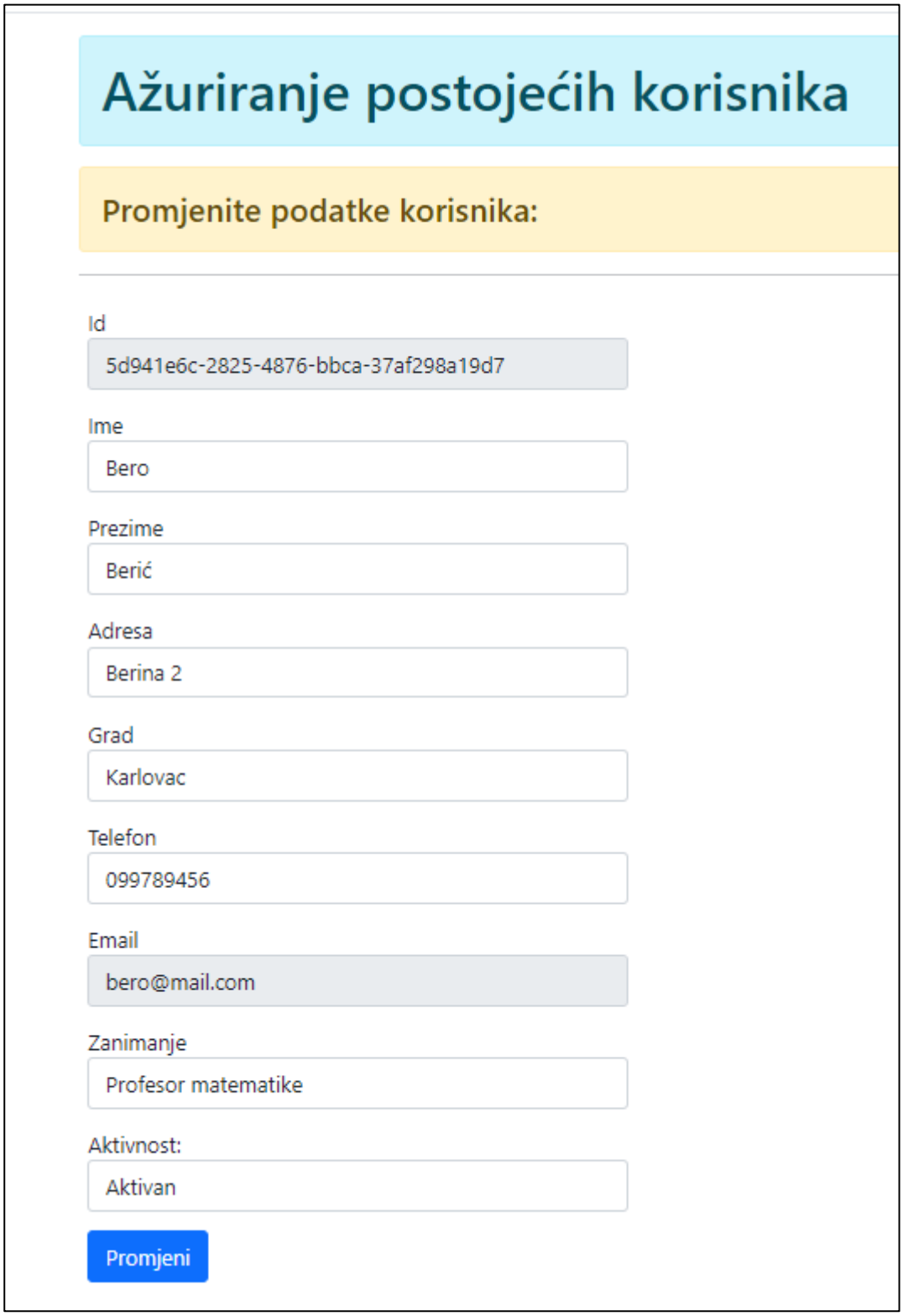
2.12. Novi korisnik

Slika 9 Novi korisnik



2.13. Ažuriranje korisnika

Slika 10 Ažuriranje korisnika

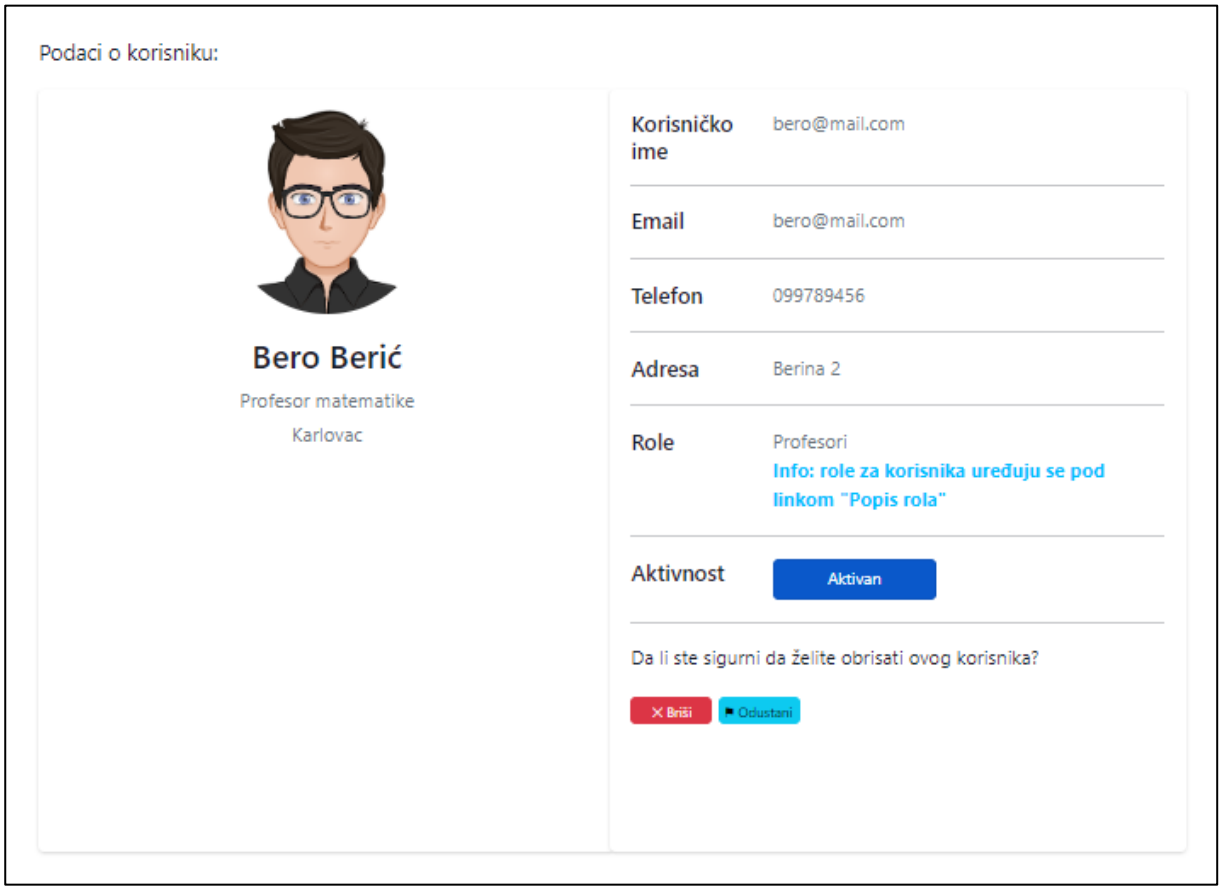


Slika 11 Source code za ažuriranje korisnika

```
0 references
public async Task<IActionResult> EditUser(string id)
{
    var user = await userManager.FindByIdAsync(id);
    EditUserViewModel model = new EditUserViewModel();
    model.Id = id;
    model.Email = user.Email;
    model.Title = user.Title;
    model.FirstName = user.FirstName;
    model.LastName = user.LastName;
    model.Address = user.Address;
    model.City = user.City;
    model.Phone = user.Phone;
    model.Active = user.Active;
    return View(model);
}

[HttpPost]
0 references
public async Task<IActionResult> EditUser(EditUserViewModel model)
{
    var user = await userManager.FindByIdAsync(model.Id);
    user.FirstName = model.FirstName;
    user.LastName = model.LastName;
    user.Address = model.Address;
    user.City = model.City;
    user.Phone = model.Phone;
    user.Active = model.Active;
    user.Title = model.Title;

    await userManager.UpdateAsync(user);
    return RedirectToAction("ListOfUsers");
}
```

2.14. Brisanje korisnika

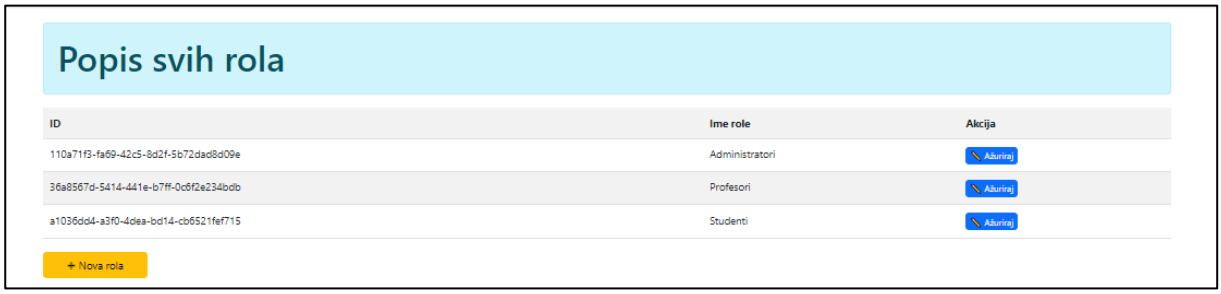
Slika 12 Brisanje korisnika



Ovaj preglednik korisnika predstavlja ključan alat za upravljanje korisničkim računima i praćenje njihovih aktivnosti unutar sustava. Kao administrator, imate moć brisanja korisnika kad god to postane potrebno. Ova funkcionalnost osigurava da imate kontrolu nad popisom korisnika i možete ukloniti pristup onima koji više nisu ovlašteni ili aktivni. Prikaz aktivnosti korisnika je također izuzetno važan. Možete jednostavno pregledavati informacije o tome je li određeni korisnik aktivan ili nije. To vam pomaže u praćenju sudjelovanja korisnika u sustavu i identificiranju onih koji možda nisu redovito koristili svoje račune. Ova analitika aktivnosti korisnika može biti ključna za održavanje sigurnosti i produktivnosti sustava. Osim toga, preglednik korisnika vam omogućuje pristup podacima o korisnicima, uključujući njihove osnovne informacije i postavke računa. Ova funkcionalnost vam pomaže da brzo provjerite i ažurirate podatke o korisnicima kad god je to potrebno, čime se osigurava točnost i ažurnost informacija u sustavu.

2.15. Pregled svih rola na sustavu

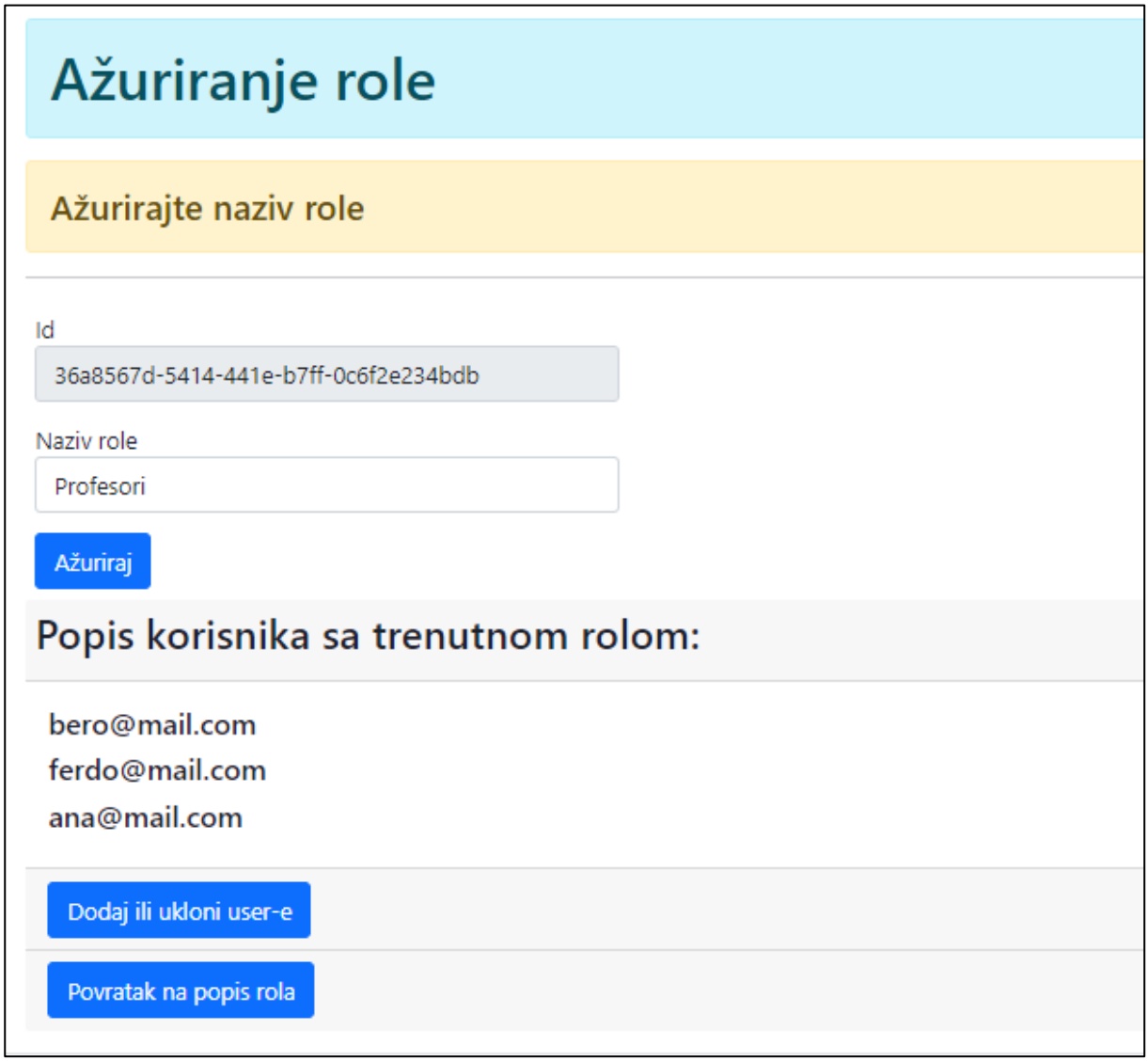
Slika 13 Pregled svih rola



Ovaj pregled rola i uloga predstavlja iznimno korisno i moćno sredstvo za upravljanje pristupom i ovlastima unutar sustava. Kao administrator, imate potpunu kontrolu nad tim procesima. Ovdje imate mogućnost ne samo da pregledavate postojeće uloge, već i da dodajete nove uloge prema potrebama sustava i korisnika. Mogućnost dodavanja novih uloga otvara vrata za prilagodbu sustava specifičnim potrebama vaše organizacije. Na primjer, možete stvoriti posebne uloge za različite odjele ili timove, osiguravajući da svaka grupa korisnika ima točno one ovlasti i pristupe koji su im potrebni za obavljanje svojih zadataka. To omogućava bolju organizaciju i efikasnost u radu. Ažuriranje postojećih uloga također je ključna funkcija. Možete brzo prilagoditi ovlasti postojećih uloga kako biste odgovarali promjenjivim potrebama. Na primjer, ako se zahtjevi za pristupom mijenjaju tijekom vremena, možete brzo prilagoditi postojeće uloge kako biste odražavali te promjene. Ovo osigurava da sustav bude dinamičan i prilagodljiv, što je ključno za njegovu dugoročnu funkcionalnost.

2.16. Administracija rola

Slika 14 Administracija rola

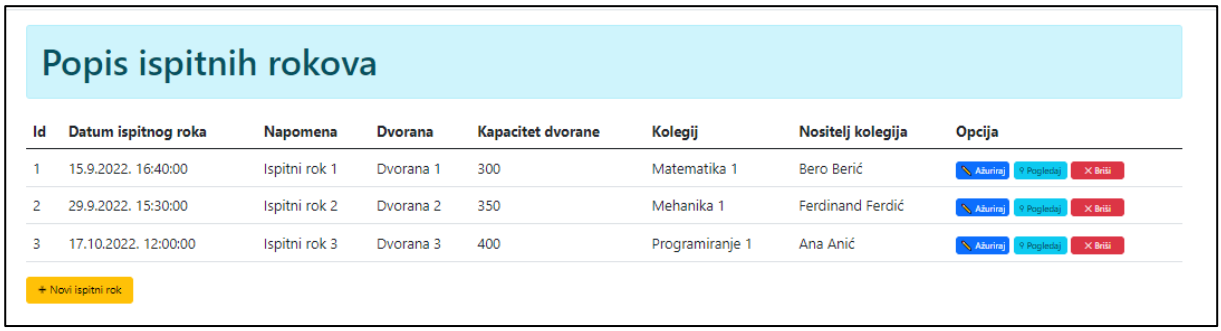


Proces ažuriranja korisničkih rola predstavlja jednu od ključnih ovlasti i odgovornosti administratora u sustavu. Samo administrator ima privilegiju da pregleda i mijenja uloge korisnika kako bi osigurao da svaki korisnik ima odgovarajuće pristupne ovlasti u skladu s njihovim trenutnim zadacima i odgovornostima. Prvo, administrator ima uvid u identifikacijske brojeve (ID) korisnika i njihove trenutne uloge. Ova informacija omogućava administratoru da brzo prepozna svakog korisnika u sustavu i razumije njihov trenutni status i ovlasti. Ažuriranje rola omogućuje administratoru da prilagodi ovlasti svakog korisnika prema potrebama. Na primjer, ako se neki korisnik promovira ili premješta unutar organizacije, administrator može ažurirati njihovu ulogu kako bi odražavala njihove nove odgovornosti.

Ovo osigurava da korisnici ne dobivaju pristup ili ovlasti koje više ne trebaju ili ne smiju imati.

2.17. Pregled ispitnih rokova

Slika 15 Pregled ispitnih rokova



Slika 16 Source code za popis ispitnih rokova

```
0 references
public IActionResult ListOfExamPeriods()
{
    List<ListOfExamPeriod> modelDb = (from e in _db.ListOfExamPeriods select e).ToList();

    List<ListOfExamPeriodViewModel> model = new List<ListOfExamPeriodViewModel>();

    foreach (var examPeriod in modelDb)
    {
        ListOfExamPeriodViewModel examItem = new ListOfExamPeriodViewModel();
        examItem.Id = examPeriod.Id;
        examItem.DateOfExam = examPeriod.DateOfExam;
        examItem.RemarkNote = examPeriod.RemarkNote;
        examItem.HallName = examPeriod.HallName;
        examItem.Capacity = examPeriod.Capacity;
        examItem.ColleagueName = examPeriod.ColleagueName;
        examItem.FirstName = examPeriod.FirstName;
        examItem.LastName = examPeriod.LastName;
        model.Add(examItem);

    }

    return View(model);
}
```

2.19. Skraćenice

Skraćenica HTML označava HyperText Markup Language ili, prevedeno na hrvatski jezik, Hipertekstualni označni jezik.

Skraćenica C# označava programski jezik nazvan C Sharp (čita se kao C sharp).

Skraćenica .NET Core označava dotNET Core ili jednostavno NET Core

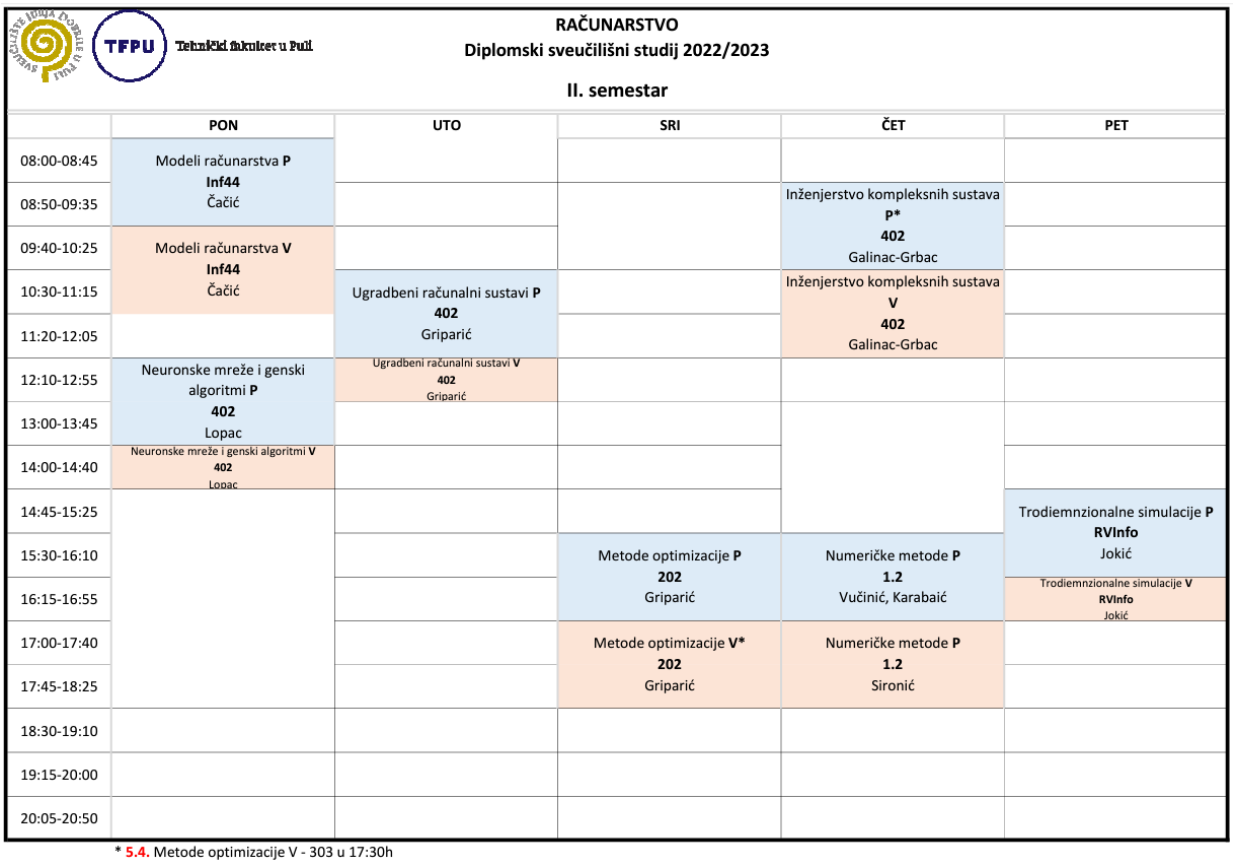
Skraćenica CSS označava Cascading Style Sheets.

Skraćenica MSSQL označava Microsoft SQL.

Skraćenica SQL označava Structured Query Language ili na hrvatski jezik strukturirani upitni jezik Skraćenica ER model označava

2.20. Usporedba ostalih stranica

Slika 17 Primjer rasporeda Računarstva TFPU



TFPU Tehnički fakultet u Puli

RAČUNARSTVO
Diplomski sveučilišni studij 2022/2023
II. semestar

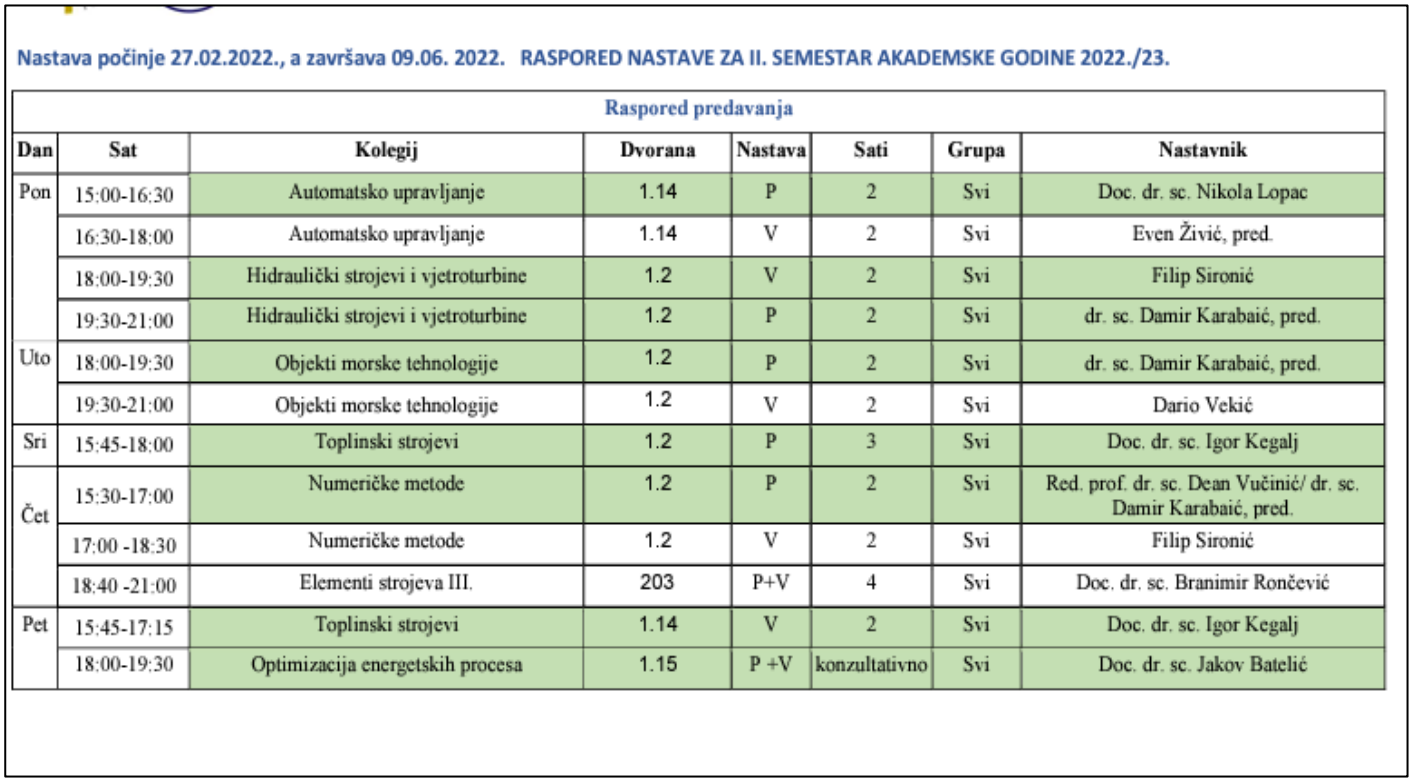
	PON	UTO	SRI	ČET	PET
08:00-08:45	Modeli računarstva **P** **Inf44** Čačić				
08:50-09:35				Inženjerstvo kompleksnih sustava **P*** **402** Galinac-Grbac	
09:40-10:25	Modeli računarstva **V** **Inf44** Čačić				
10:30-11:15		Ugradbeni računalni sustavi **P** **402** Griparić		Inženjerstvo kompleksnih sustava **V** **402** Galinac-Grbac	
11:20-12:05					
12:10-12:55	Neuronske mreže i genski algoritmi **P** **402** Lopac	Ugradbeni računalni sustavi **V** **402** Griparić			
13:00-13:45					
14:00-14:40	Neuronske mreže i genski algoritmi **V** **402** Lopac				
14:45-15:25					Trodiemnzionalne simulacije **P** **RVInfo** Jokić
15:30-16:10			Metode optimizacije **P** **202** Griparić	Numeričke metode **P** **1.2** Vučinić, Karabaić	
16:15-16:55					Trodiemnzionalne simulacije **V** **RVInfo** Jokić
17:00-17:40			Metode optimizacije **V*** **202** Griparić	Numeričke metode **P** **1.2** Sironić	
17:45-18:25					
18:30-19:10					
19:15-20:00					
20:05-20:50					

* 5.4. Metode optimizacije V - 303 u 17:30h

Izvor: https://tfpu.unipu.hr/

Na priloženom rasporedu jasno su vidljive mnoge mane. Prvenstveno, raspored ima značajne praznine između predavanja, što može biti frustrirajuće za studente jer gubi vrijeme koje bi mogli koristiti produktivnije. Osim toga, raspored navodi samo prezimena profesora, a nedostaje im titula, što može stvarati konfuziju i nedostatak poštovanja prema njihovoj akademskoj poziciji. Kako bismo poboljšali ovaj raspored, prvo bismo mogli razmotriti uklanjanje nepotrebnih tablica i informacija koje ne doprinose jasnoći rasporeda. To bi moglo učiniti raspored čišćim i lakšim za čitanje. Također, trebali bismo ozbiljno razmotriti kako popuniti rupe između predavanja. Možda bi se mogli dodati dodatni predmeti, radionice ili druge oblike aktivnosti kako bi se iskoristilo vrijeme između nastave na produktivan način. Također, trebali bismo razmotriti dodavanje titula profesora uz njihova prezimena kako bi se stvorila veća jasnoća i poštovanje prema njihovoj stručnosti. Titule imaju važnu ulogu u akademskoj zajednici i trebale bi biti vidljive kako bi se studentima omogućilo bolje razumijevanje njihovih predavača. U konačnici, poboljšanje rasporeda može značiti veću produktivnost i zadovoljstvo studenata te bolju komunikaciju u akademskom okruženju

Slika 18 Primjer rasporeda Strojarstva TFPU



Nastava počinje 27.02.2022., a završava 09.06. 2022. RASPORED NASTAVE ZA II. SEMESTAR AKADEMSKE GODINE 2022./23.

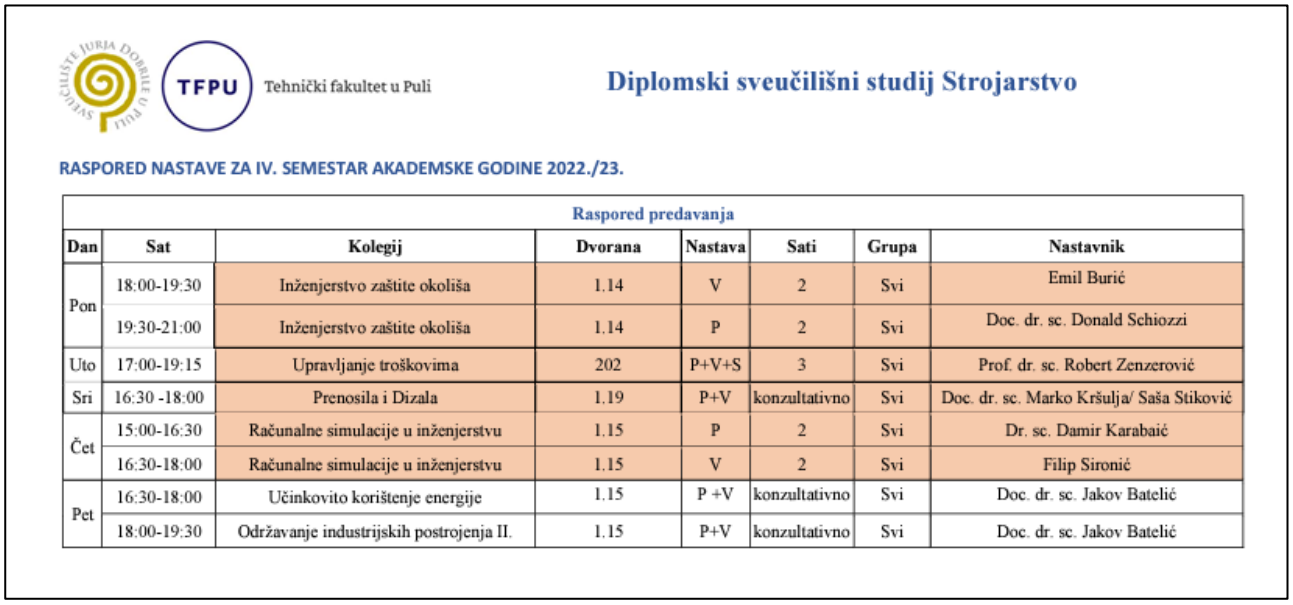
Raspored predavanja

Dan	Sat	Kolegij	Dvorana	Nastava	Sati	Grupa	Nastavnik
Pon	15:00-16:30	Automatsko upravljanje	1.14	P	2	Svi	Doc. dr. sc. Nikola Lopac
	16:30-18:00	Automatsko upravljanje	1.14	V	2	Svi	Even Živić, pred.
	18:00-19:30	Hidraulički strojevi i vjetroturbine	1.2	V	2	Svi	Filip Sironić
	19:30-21:00	Hidraulički strojevi i vjetroturbine	1.2	P	2	Svi	dr. sc. Damir Karabaić, pred.
Uto	18:00-19:30	Objekti morske tehnologije	1.2	P	2	Svi	dr. sc. Damir Karabaić, pred.
	19:30-21:00	Objekti morske tehnologije	1.2	V	2	Svi	Dario Vekić
Sri	15:45-18:00	Toplinski strojevi	1.2	P	3	Svi	Doc. dr. sc. Igor Kegalj
Čet	15:30-17:00	Numeričke metode	1.2	P	2	Svi	Red. prof. dr. sc. Dean Vučinić/ dr. sc. Damir Karabaić, pred.
	17:00 -18:30	Numeričke metode	1.2	V	2	Svi	Filip Sironić
	18:40 -21:00	Elementi strojeva III.	203	P+V	4	Svi	Doc. dr. sc. Branimir Rončević
Pet	15:45-17:15	Toplinski strojevi	1.14	V	2	Svi	Doc. dr. sc. Igor Kegalj
	18:00-19:30	Optimizacija energetskih procesa	1.15	P +V	konzultativno	Svi	Doc. dr. sc. Jakov Batelić

Izvor: https://tfpu.unipu.hr

Raspored strojarstva ozbiljno je poboljšao način na koji su organizirane informacije. Vidimo da su praznine između predavanja uvelike smanjene, što znači da studenti sada mogu iskoristiti svoje vrijeme na mnogo produktivniji način. Osim toga, ovaj novi raspored pruža detaljnije informacije o profesorima, uključujući njihova puna imena s titulama, što pomaže u stvaranju jasnije slike o njihovoj stručnosti i akademskom statusu. Jedna od značajnih prednosti ovog rasporeda je što pruža informacije o trajanju predavanja ili vježbi. Ovo je izuzetno korisno za studente koji žele bolje planirati svoje vrijeme između različitih nastavnih aktivnosti. Sada studenti znaju koliko će vremena provesti na svakom predavanju ili vježbama i mogu bolje uskladiti svoje obaveze. Još jedan izvanredan aspekt ovog rasporeda je to što jasno prikazuje koje grupe studenata trebaju prisustvovati pojedinim nastavnim aktivnostima. Ovo je od vitalnog značaja za organizaciju i koordinaciju studenata, omogućavajući im da se usmjere prema točnim predavanjima i vježbama koje su im potrebne. Sve u svemu, novi raspored strojarstva pruža više transparentnosti, preciznosti i praktičnosti za studente, čineći njihov akademski život puno olakšanim i bolje organiziranim. Ova poboljšanja omogućuju bolje iskorištavanje vremena i resursa, što je ključno za uspješno studiranje.

Slika 19 Diplomski studij Strojarstvo



Tehnički fakultet u Puli

Diplomski sveučilišni studij Strojarstvo

RASPORED NASTAVE ZA IV. SEMESTAR AKADEMSKE GODINE 2022./23.

Raspored predavanja

Dan	Sat	Kolegij	Dvorana	Nastava	Sati	Grupa	Nastavnik
Pon	18:00-19:30	Inženjerstvo zaštite okoliša	1.14	V	2	Svi	Emil Burić
	19:30-21:00	Inženjerstvo zaštite okoliša	1.14	P	2	Svi	Doc. dr. sc. Donald Schiozzi
Uto	17:00-19:15	Upravljanje troškovima	202	P+V+S	3	Svi	Prof. dr. sc. Robert Zenzerović
Sri	16:30 -18:00	Prenosila i Dizala	1.19	P+V	konzultativno	Svi	Doc. dr. sc. Marko Kršulja/ Saša Stiković
Čet	15:00-16:30	Računalne simulacije u inženjerstvu	1.15	P	2	Svi	Dr. sc. Damir Karabaić
	16:30-18:00	Računalne simulacije u inženjerstvu	1.15	V	2	Svi	Filip Sironić
Pet	16:30-18:00	Učinkovito korištenje energije	1.15	P +V	konzultativno	Svi	Doc. dr. sc. Jakov Batelić
	18:00-19:30	Održavanje industrijskih postrojenja II.	1.15	P+V	konzultativno	Svi	Doc. dr. sc. Jakov Batelić

Izvor: https://tfpu.unipu.hr

Razmotrimo kako ovaj primjer rasporeda s punim imenima i prezimenima profesora te njihovim titulama predstavlja značajno poboljšanje u akademskoj organizaciji. Ovaj primjer rasporeda jasno prikazuje kako treba izgledati dobro organiziran raspored. Imamo pristup punim imenima i prezimenima profesora, zajedno s njihovim akademskim titulama. Ovo je izuzetno korisno jer omogućava studentima da identificiraju svoje predavače s većom sigurnošću i poštuju njihovu stručnost. Titule profesora nisu samo formalnost; one ukazuju na njihovu razinu obrazovanja i iskustva, što je od vitalnog značaja za studente koji žele razumjeti pozadinu i kvalifikacije svojih profesora. Osim toga, ovaj raspored detaljno prikazuje koliko dugo traju pojedini nastavni sati. Ova informacija je ključna za studente koji planiraju svoj dan i žele znati koliko vremena će provesti na svakom predavanju ili vježbama. To omogućava bolje upravljanje vremenom i optimizaciju rasporeda. Slično tome, ovaj raspored jasno označava koje grupe studenata trebaju prisustvovati pojedinim nastavnim aktivnostima. To je izvanredno za organizaciju i koordinaciju studenata, što smanjuje zbunjenost i olakšava planiranje. U cijelosti, ovaj primjer rasporeda postavlja visoke standarde u akademskoj organizaciji. Puni identitet profesora, titule, trajanje nastave i informacije o grupama studenta čine ovaj raspored iznimno informativnim i korisnim alatom za sve studente i osoblje.

Slika 20 Kalendar prvog mjeseca

LISTOPAD

PON	UTO	SRI	ČET	PET	SUB	NED
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24/31	25	26	27	28	29	30

Izvor: https://www.unipu.hr

Ovaj kalendar označava izuzetno važan mjesec za sve studente. Tamno zelena boja kojom su označeni događaji poput prijema studenata prve godine studija i redovite sjednice senata jasno ukazuje na njihovu ključnu ulogu u akademskom kalendaru. Prijem studenata prve godine predstavlja početak njihovog visokog obrazovanja i uvodi ih u svijet sveučilišta, dok redovita sjednica senata donosi važne odluke koje utječu na sve studente i fakultet u cjelini. Ove događaje studenti ne bi smjeli propustiti jer su od suštinskog značaja za njihovu budućnost i iskustvo na sveučilištu. Plava boja koja označava početak nastavne godine u zimskom semestru također je od velike važnosti. To je trenutak kada se studenti vraćaju u učionice i laboratorije, spremni za nove izazove i akademske uspjehe. Početak semestra označava svjež početak, priliku za učenje i rast, te predstavlja trenutak kada se studenti ponovno posvećuju svojim akademskim ciljevima. Rozom bojom koja označava "Evolve Unipu" ili moderni studij, vidimo da se naglašava važnost inovacija i razvoja na sveučilištu.

Ovo je prilika za studente da se upuste u suvremene pristupe učenju i istraživanju, stvarajući tako bogatije i naprednije iskustvo tijekom svog studija. Evolucija i modernizacija sveučilišta donose brojne prednosti studentima, a njihova participacija u ovim inicijativama može im otvoriti vrata ka budućim prilikama i karijernom uspjehu.

Slika 21 Najvažniji mjesec

RUJAN

PON	UTO	SRI	ČET	PET	SUB	NED
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

4.9. Početak jesenskog ispitnog roka
4.-30.9. Upisi u više godine
5.9. Odbor za nastavu i studente
12.9. Redovita sjednica Senata
12.9. Prošireni Rektorski kolegij
19.-20.9. Upisi u 1. godinu preddiplomskog i integriranog preddiplomskog i diplomskog Studija
28.9. Dan Rektorata UNIPU
29.9. Završetak jesenskog ispitnog roka - Završetak akademske godine
do 29.9. Upisi u 1. godinu diplomskog studija

Izvor: https://www.unipu.hr

Ovaj kalendar je vrlo koristan alat za sve studente i članove fakultetskog osoblja jer pruža jasne vizualne oznake za ključne događaje u akademskom kalendaru. Plava boja kojom su označeni početak i kraj jesenskog ispitnog roka omogućava studentima da brzo i lako prepoznaju ove važne periode. To je od izuzetne važnosti jer su ispitni rokovi vremenski kritični trenuci za sve studente, kada su fokusirani na svoje ispite i pripreme. Narančasta boja koja označava datume upisa u više godine, sastanaka Odbora za nastavu studenata i upisa u prvu godinu preddiplomskih i diplomskih studija pruža dodatnu transparentnost. Ovi događaji su od suštinskog značaja za studente koji napreduju kroz svoje studije i trebaju znati kada i gdje se odvijaju.

Prikazivanje tih datuma narančastom bojom omogućava im da brzo identificiraju ključne datume u kalendaru. Ovaj kalendar kombinira vizualnu jasnoću s preciznim informacijama ispod njega, stvarajući tako kompletnu sliku akademskih događanja. To olakšava planiranje i organizaciju studenata te im pomaže da ostanu informirani o ključnim datumima tijekom cijele akademske godine.

2.21. Jednostavan primjer frontenda za Raspored

Slika 22 Primjer jednostavnog frontend koda

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Školski Raspored</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <header>
        <h1>Školski Raspored</h1>
    </header>
    <main>
        <table>
            <thead>
                <tr>
                    <th>Dan</th>
                    <th>Sat</th>
                    <th>Predmet</th>
                </tr>
            </thead>
            <tbody>
                <!-- Ovdje će se generirati redovi rasporeda iz baze podataka -->
            </tbody>
        </table>
    </main>
</body>
</html>
```

Izvor: https://developer.mozilla.org

Kratko objašnjenje koda:

TML dio definira strukturu web stranice. Imamo zaglavlje <header>, glavni sadržaj <main>, i tablicu <table> za prikaz rasporeda. U <thead> definiramo zaglavlje tablice s nazivima kolona (Dan, Sat, Predmet), a u<tbody> će se generirati redovi rasporeda iz baze podataka.

CSS stilizira izgled stranice. Postavljamo stilove za tijelo (body), zaglavlje (header), tablicu (table) i njezine ćelije (th i td). Također, postavljamo boje, fontove i razmak između elemenata

Slika 23 Jednostavan primjer backenda u .NET core

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Configuration;
using Microsoft.Data.SqlClient;
using System;
using System.Collections.Generic;
using System.Data;

namespace SchoolScheduleApi.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    public class ScheduleController : ControllerBase
    {
        private readonly IConfiguration _configuration;

        public ScheduleController(IConfiguration configuration)
        {
            _configuration = configuration;
        }

        [HttpGet]
        public IEnumerable<ScheduleItem> Get()
        {
            var connectionString = _configuration.GetConnectionString("DefaultConnection");
            var scheduleItems = new List<ScheduleItem>();

            using (var connection = new SqlConnection(connectionString))
            {
                connection.Open();

                using (var command = new SqlCommand("SELECT * FROM Schedule", connection))
                {
                    using (var reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            var item = new ScheduleItem
                            {
                                Day = reader["Day"].ToString(),
                                Hour = Convert.ToInt32(reader["Hour"]),
                                Subject = reader["Subject"].ToString()
                            };
                            scheduleItems.Add(item);
                        }
                    }
                }
            }

            return scheduleItems;
        }
    }
}
```

Izvor: https://developer.mozilla.org

2.22. Kratko objašnjenje koda

U ovom rasporedu definiramo ASP.NET Core kontroler za upravljanje rasporedom.

Konstruktor prima IConfiguration kako bismo mogli dohvatiti konfiguraciju iz appsettings.json datoteke. Metoda Get je HTTP GET metoda koja dohvaća raspored iz baze podataka.

U ovoj metodi koristimo SqlConnection za povezivanje s bazom podataka. Veza se otvara, izvršava SQL upit za dohvat rasporeda, a zatim se podaci čitaju iz rezultata i spremljeni u listu scheduleItems. Na kraju, vraćamo listu scheduleItems kao odgovor iz API-ja.

Slika 24 Jednostavan primjer SQL baze

```
{
    "ConnectionStrings": {
        "DefaultConnection": "Server=localhost;Database=SchoolSchedule;Trusted_Connection=True;"
    },
    // ostatak konfiguracije...
}
```

```
CREATE TABLE Students (
    StudentID INT PRIMARY KEY,
    Ime NVARCHAR(50),
    Prezime NVARCHAR(50),
    DatumRodjenja DATE,
    Adresa NVARCHAR(100),
    Email NVARCHAR(100)
);
```

```
{
    "ConnectionStrings": {
        "DefaultConnection": "Server=localhost;Database=SchoolSchedule;Trusted_Connection=True;"
    },
    // ostatak konfiguracije...
}
```

```
CREATE TABLE Students (
    StudentID INT PRIMARY KEY,
    Ime NVARCHAR(50),
    Prezime NVARCHAR(50),
    DatumRodjenja DATE,
    Adresa NVARCHAR(100),
    Email NVARCHAR(100)
);
```

```
CREATE TABLE Teachers (
    TeacherID INT PRIMARY KEY,
    Ime NVARCHAR(50),
    Prezime NVARCHAR(50),
    DatumZaposlenja DATE,
    Predmeti NVARCHAR(255) -- N
);
```

Izvor: https://developer.mozilla.org

Kratko objašnjenje koda
Tablica "Students" (Učenici):
StudentID je primarni ključ i jedinstveni identifikator svakog učenika.
Ime i Prezime pohranjuju ime i prezime učenika.
DatumRodjenja pohranjuje datum rođenja učenika.
Adresa pohranjuje adresu učenika.

Email pohranjuje email adresu učenika.
Tablica "Teachers" (Nastavnici):
TeacherID je primarni ključ i jedinstveni identifikator svakog nastavnika.
Ime i Prezime pohranjuju ime i prezime nastavnika.
DatumZaposlenja pohranjuje datum kada je nastavnik zaposlen.

2.23. Usporedba MSSQL i MongoDB baza

Usporedba Microsoft SQL Server (MSSQL) i MongoDB. Prednosti MSSQL-a

Kada razmatramo izbor između različitih sustava za upravljanje bazama podataka (DBMS), često se suočavamo s odlukom između relacijskih i NoSQL baza podataka. Microsoft SQL Server (MSSQL) predstavlja snažan relacijski DBMS koji donosi niz prednosti nad NoSQL rješenjima poput MongoDB. Evo nekoliko ključnih razloga zbog kojih biste mogli preferirati MSSQL u određenim situacijama: MSSQL je poznat po svojoj čvrstoj shemi podataka i podršci za ACID (Atomicity, Consistency, Isolation, Durability) transakcije. Ovo je ključno za aplikacije koje zahtijevaju potpunu konzistenciju podataka i jamčenje integriteta. Ako radite s financijskim transakcijama, medicinskim podacima ili drugim osjetljivim informacijama, MSSQL će vam pružiti sigurnost i dosljednost koje su ključne. MSSQL je izvrsno rješenje za složene upite i dubinsku analizu podataka. Integracija s alatima poput SQL Server Reporting Services omogućava generiranje bogatih izvještaja i vizualizacija. Ako vaša organizacija zahtijeva temeljite analize podataka, MSSQL pruža robusne mogućnosti za poslovno izvještavanje. Razvoj aplikacija na platformi Microsoft, ako razvijate aplikacije na platformi Microsoft (npr., ASP.NET), MSSQL će biti prirodan izbor zbog duboke integracije s Microsoft tehnologijama. Entity Framework i druge ORM (Object-Relational Mapping) tehnologije pojednostavljuju rad s MSSQL-om, čineći ga preferiranim izborom za razvoj aplikacija na Windows platformi. MSSQL nudi visoku razinu sigurnosti i autentikacije. Kroz mogućnosti kao što su Windows autentikacija, integracija s Active Directory-jem i mogućnosti enkripcije podataka, MSSQL osigurava pouzdanu zaštitu osjetljivih informacija. Ako već koristite relacijski DBMS i razmišljate o migraciji, MSSQL omogućava relativno glatku tranziciju. Alati poput SQL Server Integration Services (SSIS) olakšavaju prijenos podataka iz drugih relacijskih baza podataka u MSSQL.

Unatoč ovim prednostima, važno je napomenuti da nema univerzalnog DBMS-a koji odgovara svakoj situaciji. Odabir između MSSQL-a i MongoDB-a, ili drugog DBMS-a, treba se temeljiti na specifičnim zahtjevima projekta i tipu aplikacije koju razvijate. MSSQL ostaje snažan izbor za aplikacije koje zahtijevaju strogu kontrolu nad strukturom podataka i visoku razinu sigurnosti.

3. Zaključak

Cilj ovog projekta bio je uspješno razviti web aplikaciju koja omogućava profesorima i studentima da se prijavljuju na sustav. Studenti mogu prijaviti ispite, pregledavati dostupne termine za ispite i odjaviti se s njih ako je potrebno. S druge strane, profesori imaju uvid u prijave studenata i obavještavaju se o tome koji studenti su se prijavili za ispit. Osim toga, aplikacija omogućava i administratorskom korisniku potpunu kontrolu nad sustavom. Administrator može dodavati i brisati korisnike, ažurirati podatke i postavke, te upravljati aktivnostima korisnika, odnosno označavati ih kao aktivne ili neaktivne prema potrebi. Ova administrativna kontrola pruža fleksibilnost i omogućava efikasno upravljanje korisnicima i njihovim pristupom sustavu. Projekt se temeljio na istraživanju tehnologija kao što su .NET Core, HTML, CSS i MS SQL. Istraživanje je bilo usmjereno na pronalaženje grešaka i mana u postojećim rasporedima i razvoju rješenja koje bi bilo preglednije i učinkovitije od konkurencije. Kao rezultat ovog istraživanja i razvoja, stvorena je web aplikacija koja olakšava procese prijava ispita, pruža jasan pregled dostupnih termina i poboljšava komunikaciju između profesora i studenata. Ovo rješenje donosi transparentnost, olakšava administraciju i pridonosi učinkovitijem upravljanju ispitima, a istovremeno omogućava administratorskom korisniku potpunu kontrolu nad sustavom.

4. Literatura

[1] https://en.wikipedia.org/wiki/C_Sharp_(programming_language) (datum 14.09.2023)
[2] https://www.w3schools.com/cs/index.php (datum 14.09.2023)
[3] https://learn.microsoft.com/en-us/dotnet/csharp/ (datum 14.09.2023)
[4] https://en.wikipedia.org/wiki/HTML (datum 14.09.2023)
[5] https://html.com/ (datum 14.09.2023)
[6] https://www.tutorialsteacher.com/core/dotnet-core (datum 14.09.2023)
[7] https://www.c-sharpcorner.com/article/what-is-dot-net-core/ (datum 14.09.2023)
[8] https://hr.wikipedia.org/wiki/CSS (datum 14.09.2023)
[9] https://web.dev/learn/css/ (datum 14.09.2023)
[10] https://www.npmjs.com/package/mssql (datum 14.09.2023)
[11] https://hasura.io/learn/database/microsoft-sql-server/what-is-mssql/ (datum 14.09.2023)
[12] https://repository.ffri.uniri.hr/islandora/object/ffri%3A614/datastream/PDF/view (datum 14.09.2023)
[13] https://www.quackit.com/html/codes/html_frames_code.cfm (datum 14.09.2023)
[14] https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/language-specification/standard-library (datum 14.09.2023)
[15] https://learn.microsoft.com/en-us/aspnet/core/fundamentals/target-aspnetcore?view=aspnetcore-7.0&tabs=visual-studio (datum 14.09.2023)
[16] https://getbootstrap.com/docs/5.3/examples/ (datum 14.09.2023)
[17] https://animate.style/ (datum 14.09.2023)
[18] https://tailwindcss.com/ (datum 14.09.2023)
[19] https://ucenici.com/kalendar-skolske-godine-2022-2023/ (datum 14.09.2023)

5. Popis slika