



UNIVERZITET U ZENICI
POLITEHNIČKI FAKULTET
Softversko inženjerstvo



PROJEKTNI ZADATAK

Predmet: Razvoj informacionih sistema
Tema: Virtual reality escape rooms

Studenti: Irma Kulačić, Adna Varupa
Broj indeksa: 339, 379

Profesor: Doc. Dr. Sc. Denis Čeke

Zenica, akademska 2024/2025. godina

SADRŽAJ

UVOD.....	3
OSNOVNE INFORMACIJE O KOMPANIJI.....	3
VR ESCAPE ROOM KONCEPT.....	3
CILJEVI PROJEKTA.....	3
OPSEG PROJEKTA.....	4
PLAN IZVEDBE PROJEKTA.....	6
ANALIZA I PROCJENA RIZIKA.....	6
SPECIFIKACIJA I ANALIZA KORISNIČKIH ZAHTJEVA.....	7
1. Sistemski Administrator.....	7
2. Programer.....	7
3. Korisnik (Igrač).....	7
PITANJA I ODGOVORI ZA INTERVJU.....	8
FUNKCIONALNI ZAHTJEVI U PROJEKTU INFORMACIONOG SISTEMA.....	9
REGISTRACIJA KORISNIKA.....	10
PRIJAVA KORISNIKA.....	10
PRETRAGA I ODABIR IGRE.....	11
REZERVACIJA IGRE.....	12
UPRAVLJANJE IGRAMA.....	12
PREGLED REZULTATA.....	13
SISTEM OCJENJIVANJA.....	13
IDENTIFICIRANJE MOGUĆIH PROBLEMA.....	14
ARHITEKTURA SISTEMA.....	15
ARHITEKTURA ASP.NET WEB APLIKACIJE.....	18
MODEL PODATAKA U C#.....	18
DbContext U C#.....	19
PUNJENJE BAZE PODATAKA PODACIMA.....	20
IMPLEMENTACIJA I UPRAVLJANJE CLOUD BAZOM PODATAKA.....	21
EXPORT I REIMPORT BAZE PODATAKA.....	21
CLOUD HOSTING I POVEZIVANJE BAZE PODATAKA.....	21
MODEL BAZE PODATAKA I RELACIJE.....	22
MIGRACIJE BAZE PODATAKA.....	22
VIZUALIZACIJA BAZE PODATAKA I ERD ANALIZA.....	23
BACKEND INFORMACIONOG SISTEMA.....	24
POČETNO STANJE NAKON MIGRACIJE.....	24
AUTOMATSKO GENERISANJE KONTROLERA.....	25
GENERISANJE KONTROLERA NA PRIMJERU IgraController.....	25
TESTIRANJE API-JA POMOĆU POSTMANA.....	25
ARHITEKTURA BACKENDA I MVC PRINCIP.....	26
DIZAJN KORISNIČKIH INTERFEJSA.....	27
FINALIZIRAN MVC DIJAGRAM.....	27
DODAVANJE FRONTEND KOMPONENTI SA SCAFFOLD.....	27
ISPRAVAN PRIKAZ ENUMERACIJSKIH TIPOVA.....	29

KORIŠTENJE PODATAKA SA BACKENDA U FRONTENDU.....	30
VALIDACIJA PODATAKA.....	30
FINALNI IZGLED WEB DIZAJNA.....	31
SWAGGER DOKUMENTACIJA.....	37
DEPLOYMENT I DOKERIZACIJA.....	38
DOCKERIZACIJA.....	38
DEPLOYMENT.....	39

UVOD

Sa razvojem modernih tehnologija, virtualna stvarnost (VR) postaje sve više prisutna u raznim sferama industrije i društva. Ovaj napredak omogućava ne samo nove načine zabave, već i interaktivno učenje, trening i simulacije koje pružaju korisnicima jedinstvena iskustva koja su ranije bila nezamisliva. Jedan od zanimljivih primjera upotrebe VR tehnologije jeste u kreiranju “escape room” igara koje su prvobitno nastale kao fizičke avanture, a sada su unaprijeđene korištenjem VR tehnologije.

OSNOVNE INFORMACIJE O KOMPANIJI

Hunt Escape je kompanija koja se bavi pružanjem uzbudljivih escape room iskustava gdje timovi rješavaju zagonetke kako bi pobjegli iz tematski uređenih soba unutar zadanog vremenskog roka. Trenutno nude igre uživo, ali planiraju proširenje na virtualne escape room igre, koristeći tehnologiju virtualne stvarnosti (VR), kako bi korisnicima omogućili novi nivo digitalnog uranjanja. Ovaj projekat je usmjeren na razvijanje informacionog sistema koji će podržati tu ekspanziju, omogućavajući efikasno upravljanje, nadzor i pružanje VR igara.

VR ESCAPE ROOM KONCEPT

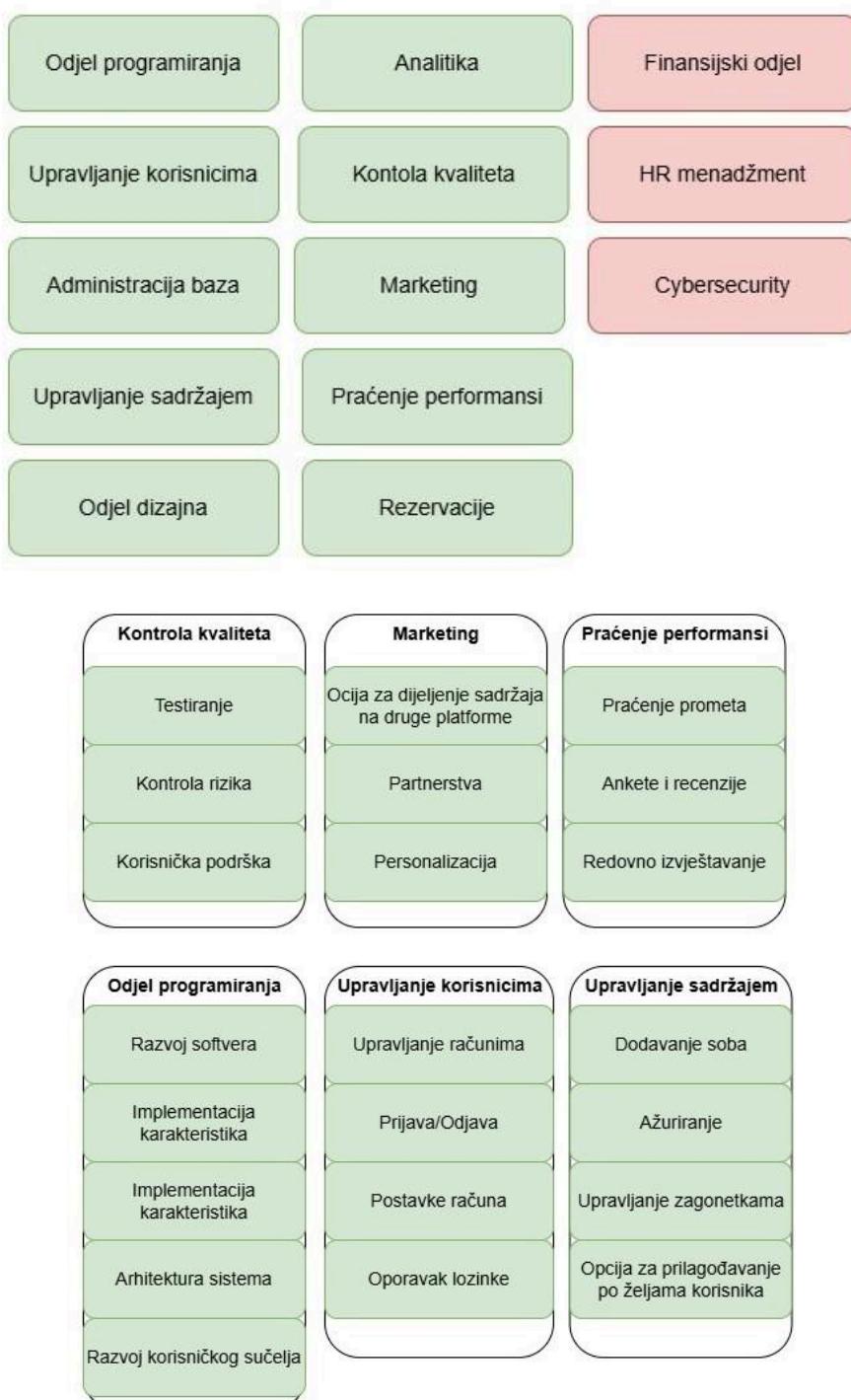
VR escape room je koncept koji omogućava korisnicima da urone u virtualni svijet i istraže različite scenarije, rješavajući zagonetke i zadatke kako bi došli do izlaza. Na ovaj način se kombinuje zabava sa izazovima logičkog razmišljanja, te pruža intenzivniji i interaktivniji doživljaj u odnosu na tradicionalne escape room igre. Koriste VR uređaje i senzore za praćenje pokreta, pružajući korisnicima mogućnost da se kreću i manipulišu objektima kao da su u stvarnom prostoru. Za razliku od klasičnih escape room soba, VR escape room omogućava stvaranje različitih okruženja i scenarija koji nisu ograničeni fizičkim prostorom, poput srednjovjekovnog dvorca, svemirske stanice ili fantastičnog svijeta čarolija, a sve to bez napuštanja prostorije. Pored toga, VR tehnologija omogućava i personalizaciju iskustva, gdje se težina zagonetki, dinamika igre, pa čak i specifični vizuelni elementi mogu prilagoditi potrebama i željama korisnika.

CILJEVI PROJEKTA

Cilj ovog projekta je razviti informacioni sistem za Hunt Escape koji centralizuje sve aktivnosti i omogućava upravljanje korisnicima, rezervacijama i sadržajem igara. Sistem će biti koncipiran tako da administratori lako ažuriraju igre i prilagođavaju ih potrebama korisnika, dok korisnici mogu jednostavno rezervisati igre, pratiti napredak i personalizovati svoje iskustvo. Sistem će omogućiti praćenje rezultata korisnika i analizu podataka o učinku, čime se dodatno unapređuje korisničko iskustvo i podstiče ponovna igra.

OPSEG PROJEKTA

Opseg projekta za informacioni sistem Hunt Escape obuhvata razvoj rešenja koje podržava VR escape room igre, centralizujući informacije i omogućavajući efikasno upravljanje korisnicima, rezervacijama i sadržajem. Sistem se fokusira na aktivnosti vezane za VR segment, a ne obuhvata sve poslovne procese kompanije.

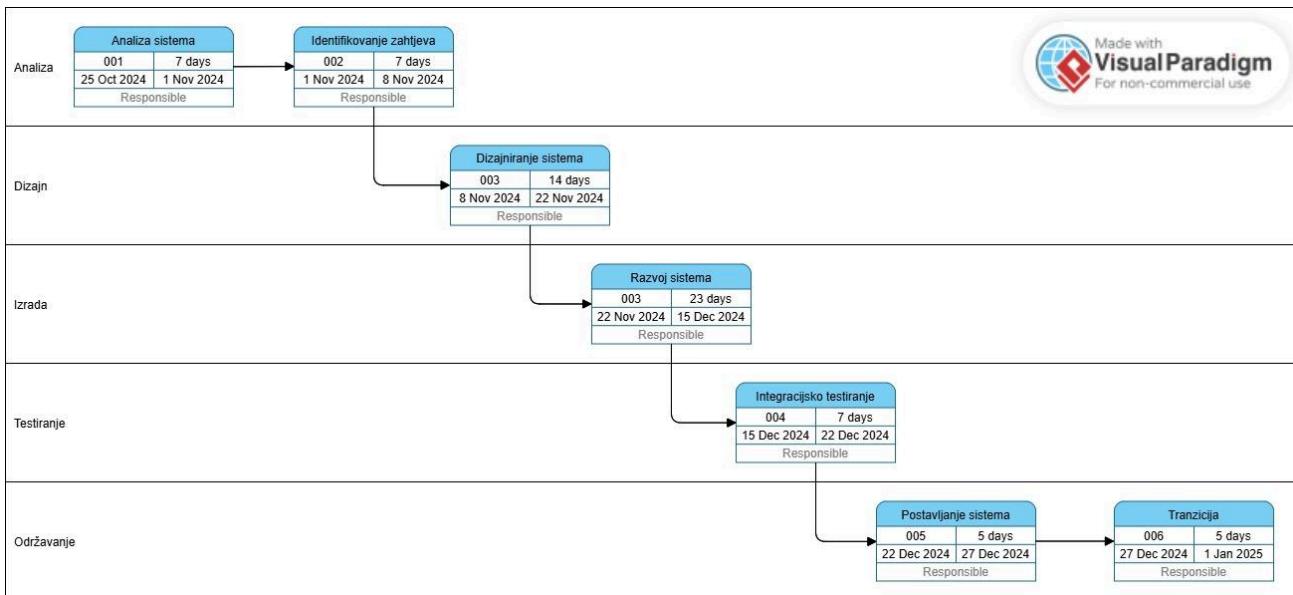




Biće razvijeni moduli kao što su: **upravljanje korisnicima**, koji omogućava registraciju i praćenje istorije igranja; **rezervacijski sistem**, koji olakšava rezervaciju VR igara; **upravljanje sadržajem**, koje omogućava administratorima da dodaju i ažuriraju igre; **praćenje performansi**, koje pruža analitiku o igramu i korisničkim aktivnostima; **odjel programiranja**, koji se fokusira na razvoj i implementaciju sistema; **administracija baza**, koja osigurava efikasno upravljanje podacima; **odjel dizajna**, koji se bavi vizualnim aspektima i korisničkim interfejsom; **analitika**, koja omogućava prikupljanje i analizu podataka; **marketing**, koji se fokusira na promociju i privlačenje korisnika; i **kontrola kvaliteta**, koja osigurava visok standard proizvoda i usluga. Moduli kao što su finansijsko upravljanje ili upravljanje ljudskim resursima neće biti dio ovog projekta.

Glavne funkcionalnosti uključuju registraciju korisnika, brzu rezervaciju igara i prikupljanje podataka o performansama, što pomaže u donošenju informisanih odluka i unapređenju ponude. Korištenje dijagrama za vizualizaciju obuhvata dijelova sistema biće korisno za predstavljanje modula i funkcionalnosti, olakšavajući razumevanje obima projekta i njegovih ključnih elemenata.

PLAN IZVEDBE PROJEKTA



ANALIZA I PROCJENA RIZIKA

1. Rizici u rasporedu

- Podcijenjeno vrijeme: Rokovi za razvoj mogu biti podcijenjeni, što dovodi do kašnjenja.
- Problemi s alokacijom resursa: Nepravilna raspodjela resursa ometa napredak.

2. Tehnički rizici

- Složenost integracije: Integracija komponenti sistema može biti izazovna.
- Promjene zahtjeva: Izmjene u dizajnu dovode do dodatnog rada.

3. Rizici korisničkog iskustva

- Negativne povratne informacije: Neispunjavanje očekivanja može rezultirati lošim recenzijama.
- Tehničke greške: Problemi tokom korištenja frustriraju korisnike.

4. Rizici usklađenosti

- Pitanja privatnosti podataka: Neusaglašenost s propisima o zaštiti podataka može uzrokovati pravne probleme.

SPECIFIKACIJA I ANALIZA KORISNIČKIH ZAHTJEVA

1. Sistemski Administrator

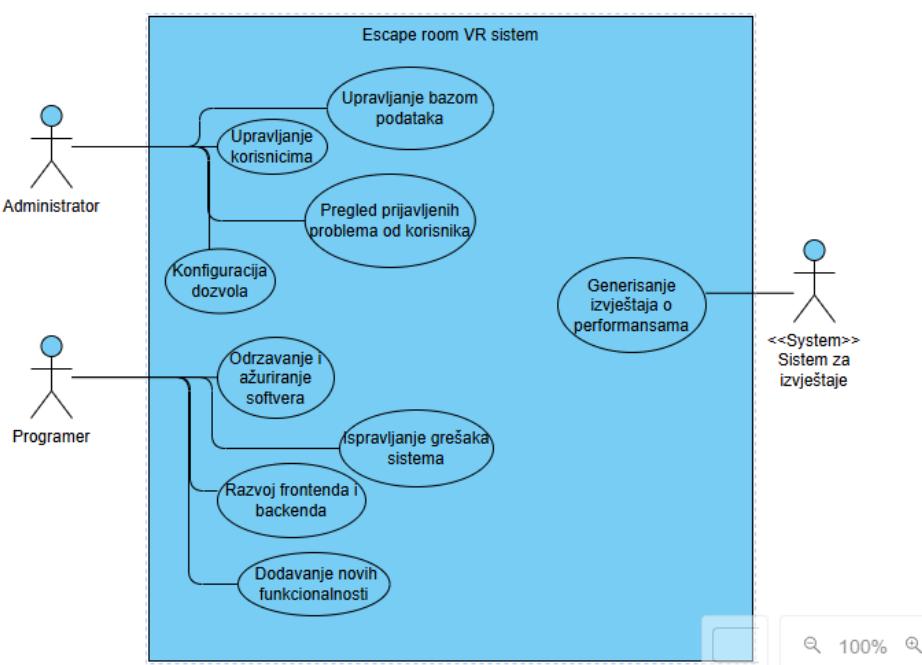
- Upravljanje korisnicima:** Dodavanje/uklanjanje programera i igrača, upravljanje ulogama.
- Konfiguracija sistema:** Postavke sistema, obavijesti, dostupnost igara.
- Upravljanje bazom igara:** Dodavanje, uređivanje ili uklanjanje igara.
- Izvještaji i analitika:** Generisanje izvještaja o korištenju sistema.
- Podrška i žalbe:** Rješavanje žalbi i pružanje podrške korisnicima.

2. Programer

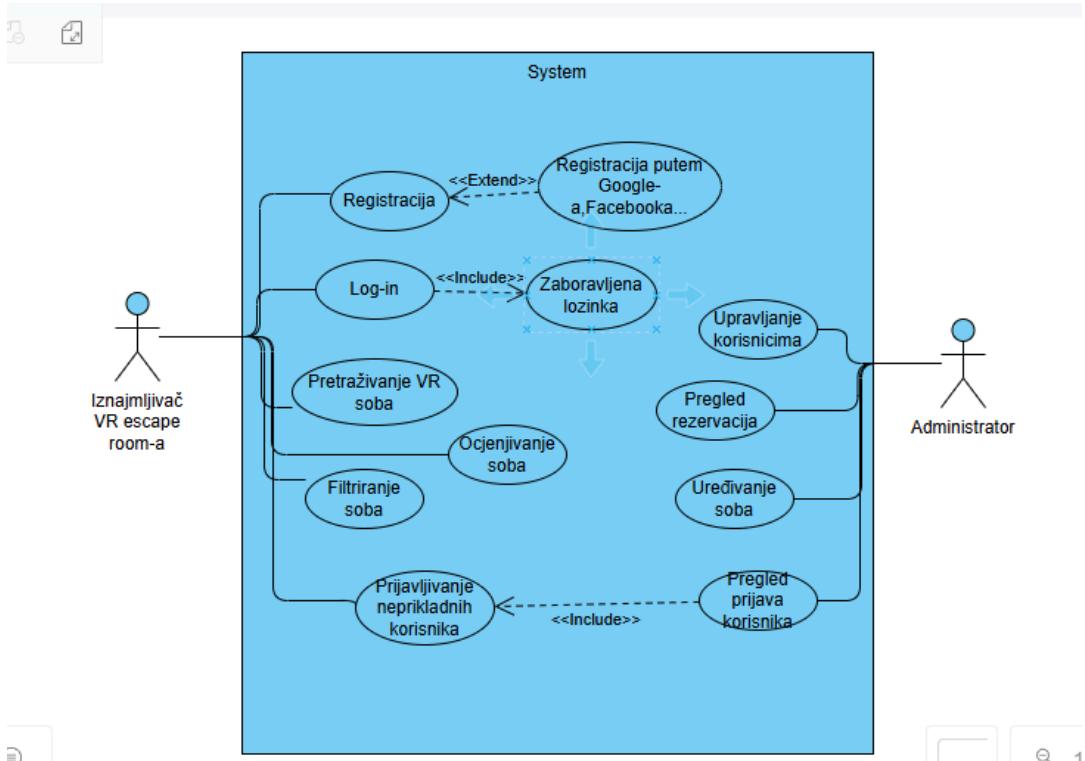
- Održavanje sistema:** Ažuriranje backend/frontend sistema, ispravljanje grešaka.
- Integracija novih igara:** Dodavanje novih VR igara (ne razvoj).
- API menadžment:** Upravljanje spoljnom integracijom sistema (npr. plaćanja).

3. Korisnik (Igrač)

- Upravljanje nalogom:** Registracija, prijava, ažuriranje profila.
- Rezervisanje igara:** Pregledanje i rezervacija VR igara.
- Istorijski igara i liderboard:** Pregled prošlih igara, pozicije na tabeli.
- Ocjene i povratne informacije:** Ocjenjivanje i komentarisani odigranih igara.



Slika 1: UML dijagram slučaja upotrebe poslovnih procesa



Slika 2: UML dijagram slučaja upotrebe poslovnih procesa 2

PITANJA I ODGOVORI ZA INTERVJU

Šta je glavni cilj ovog projekta i koji problem rješava?

- Na ovo pitanje ste već sami odgovorili u okviru početne faze projektovanja.

Ko su glavni korisnici sistema i kako će ga koristiti?

- Ovo je također nešto što je dogovoreno u početnoj fazi projekta. Kao što smo se dogovorili, očekujemo da postoje tačno tri različite vrste korisnika.

Koje funkcionalnosti su apsolutno neophodne, a koje bi bile „lijepo za imati“?

- O ovome smo također već pričali. Mi smo vam već naveli 10 funkcionalnosti koje su nam neophodne, a nema potrebe da se u ovom trenutku uvode nikakve dodatne sisteme.

Možete li opisati tipičan tok rada korisnika u sistemu?

- Pa, sve ovisi o tome na kog korisnika mislite. Svaki korisnik ima svoja zaduženja, slobodno sami odredite kako to treba izgledati pa nam dajte pregled funkcionalnih zahtjeva unutar projekta.

Treba li sistem omogućiti personalizaciju (npr. teme, težinu igara)?

- Da, to bi bilo jako lijepo. U prvoj verziji možete predvidjeti dvije različite teme i dvije različite težine, da se vidi kako to izgleda, a onda možemo u narednim verzijama dodavati po potrebi.

Koje podatke treba pratiti o korisnicima i igrama (npr. statistike, napredak)?

- Bilo bi dobro da postoji neki element gamifikacije koji pravi rang listu korisnika na osnovu broja osvojenih bodova, koji se akumulišu kroz više odigranih escape room igara. Na osnovu svojih performansi u svakoj igri, korisniku bi bili dodijeljeni bodovi koji bi se sabirali u

njegov ukupni high score, kao i max. score u jednoj igri, i imali bismo dvije rang liste svih korisnika, kako bi bili motivisani za postizanje što boljih ukupnih rezultata.

Postoje li specifični tehnički zahtjevi ili platforme koje treba koristiti?

- Jedan od ključnih parametara za nas je svakako brzina, jer želimo da sistem bude što više i lakše dostupan. Drugi parametar je intuitivnost, kako bi se korisnici brzo mogli ulogovati i koristiti razne dijelove sistema, povećavajući nam popularnost.

Treba li sistem podržavati više jezika?

- Za početak je dovoljno da bude na bosanskom jeziku, jer prvo ciljamo bh. tržište, a u budućnosti se možemo širiti na globalni nivo.

Kako će se rješavati sigurnost podataka i pristup korisnicima (npr. uloge, lozinke)?

- Pojedinosti svakako ostavljamo vama na slobodni odabir, ali naravno najznačajnije je da su te stvari sigurne od krađe, da nas niko ne može optužiti za malverzacije tuđim podacima i da se naravno koriste korisnička imena/emailovi i šifre.

Koliko često želite primati izvještaje o napretku?

- Bilo bi dobro da je to u vidu neke stranice koja svaki put kada se otvorí prikazuje najažurnije podatke koje onda možemo filtrirati po mjesecu, godini ili nekim drugim parametrima.

Ko će testirati sistem i kako će se prikupljati povratne informacije?

- Pa to bi trebao biti vaš razvojni tim, mi nemamo neke posebne testere.

Kako ćemo rješavati promjene u zahtjevima tokom razvoja?

- Kontaktirajte nas kako bismo mogli pomoći da se svi eventualni problemi što lakše riješe, jer nam je u interesu da se sistem što prije pusti u rad.

Da li planirate proširenje sistema u budućnosti (nove funkcionalnosti, mobilna verzija)?

- Svakako, ali s obzirom da smo ograničeni budžetom, to ipak ne planiramo u ovom trenutku, nego tek kada novi informacioni sistem zaživi.

Kako sistem treba raditi kod velikog broja korisnika ili slabije internet konekcije?

- Bilo bi idealno da se takve stvari ne dešavaju nikako, ali dovoljno je da imaju što manji utjecaj na korisničko iskustvo. Bilo bi dobro da se barem pojavi neka poruka koja govori da se treba čekati, ili da je sistem pao, da barem korisnici razumiju u čemu je problem.

Šta biste prioritetizirali ako ne možemo isporučiti sve funkcionalnosti na vrijeme?

- 10 funkcionalnosti su prag ispod kojeg se ne može prelaziti, a sve preko toga je stvar dogovora

FUNKCIONALNI ZAHTJEVI U PROJEKTU INFORMACIONOG SISTEMA

IDENTIFIKACIJA FUNKCIONALNIH ZAHTJAVA:

Registracija korisnika → Korisnik pravi nalog.

Prijava korisnika → Korisnik se loguje u sistem.

Pretraga i odabir igre → Korisnik bira dostupne VR igre.

Rezervacija igre → Korisnik rezerviše termin za igru.
 Upravljanje igrama (admin) → Administrator dodaje nove VR igre.
 Pregled rezultata → Korisnik vidi svoj učinak nakon igre.
 Sistem ocjenjivanja → Korisnici ocjenjuju igru i ostavljaju komentare.

REGISTRACIJA KORISNIKA

Registracija korisnika je osnovni zahtjev sistema koji omogućava novim korisnicima da kreiraju svoj nalog i dobiju pristup svim funkcionalnostima sistema. Korisnik otvara stranicu za registraciju gdje unosi osnovne podatke kao što su email, korisničko ime i lozinka. Nakon što korisnik klikne na dugme „Registruj se“, sistem provjerava validnost unesenih podataka. Ako su podaci ispravni, sistem kreira korisnički nalog i šalje potvrdu o uspješnoj registraciji. Ovaj proces osigurava da samo autentični korisnici mogu pristupiti sistemu.



Slika 3: UML dijagram aktivnosti za registraciju korisnika

PRIJAVA KORISNIKA

Prijava korisnika omogućava registriranim korisnicima da pristupe svom nalogu i koriste sistem. Korisnik otvara stranicu za prijavu gdje unosi svoj email i lozinku. Nakon klika na dugme „Prijava se“, sistem provjerava tačnost unesenih podataka. Ako su podaci tačni, korisnik dobija pristup sistemu i može koristiti sve dostupne funkcionalnosti. U slučaju netačnih podataka, sistem prikazuje poruku o grešci, što osigurava sigurnost i zaštitu korisničkih naloga.



Slika 4: UML dijagram aktivnosti za prijavu korisnika

PRETRAGA I ODABIR IGRE

Pretraga i odabir igre omogućava korisnicima da pregledaju dostupne VR escape room igre i odaberu one koje ih interesuju. Nakon prijave u sistem, korisnik otvara sekciju „Dostupne igre“ gdje može koristiti filtere kao što su težina, trajanje i tema kako bi suzio pretragu. Klikom na određenu igru, korisnik može vidjeti detaljne informacije o njoj, uključujući opis, težinu i trajanje. Ako je korisnik zainteresovan, može odabrati igru i nastaviti sa procesom rezervacije.



Slika 5: UML dijagram aktivnosti za pretragu i odabir igre

REZERVACIJA IGRE

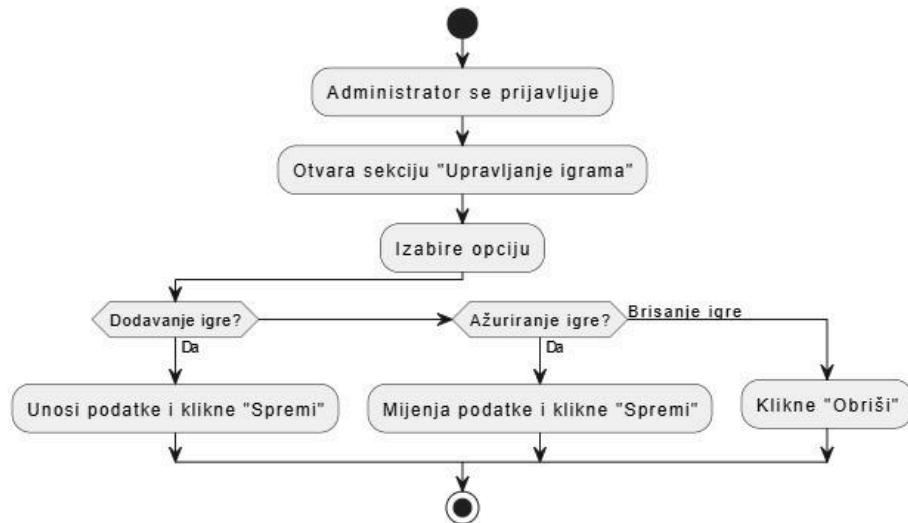
Rezervacija igre je ključna funkcionalnost sistema koja omogućava korisnicima da rezervišu termine za igranje VR igara. Nakon što korisnik odabere željenu igru, klikne na dugme „Rezerviši termin“ i odabire datum i vrijeme igranja. Sistem provjerava dostupnost odabranog termina i, ukoliko je slobodan, potvrđuje rezervaciju. Korisnik zatim dobija potvrdu o rezervaciji putem emaila ili notifikacije, što osigurava da su svi podaci o rezervaciji jasno i tačno preneseni.



Slika 6: UML dijagram aktivnosti za rezervaciju igre

UPRAVLJANJE IGRAMA

Upravljanje igrama je funkcionalnost namijenjena administratorima sistema. Omogućava im da dodaju nove VR igre, uređuju postojeće podatke ili brišu igre koje više nisu dostupne. Administrator se prijavljuje u sistem i otvara sekciju „Upravljanje igrama“ gdje može unijeti detalje o novoj igri, kao što su naslov, opis, težina i trajanje. Sve promjene se automatski ažuriraju u sistemu, čime se osigurava da su korisnici uvijek informirani o dostupnim igrama.



Slika 7: UML dijagram aktivnosti za upravljanje igrama

PREGLED REZULTATA

Pregled rezultata omogućava korisnicima da vide svoje performanse nakon završetka VR escape room igre. Sistem prikuplja podatke o vremenu završetka, tačnosti odgovora i ostalim relevantnim metrikama. Ovi rezultati se prikazuju korisniku u obliku poena, vremena i statusa (uspješan/neuspješan). Korisnik takođe može pregledati historiju svih prethodnih igranja, što pruža uvid u njihov napredak i postignuća.



Slika 8: UML dijagram aktivnosti za pregled rezultata

SISTEM OCJENJIVANJA

Sistem ocjenjivanja omogućava korisnicima da ocijene igru i ostave komentare nakon završetka igranja. Korisnik otvara sekciju „Ocjeni igru“ gdje unosi ocjenu (1-5 zvjezdica) i opcioni komentar. Klikom na dugme „Pošalji ocjenu“, sistem bilježi ocjenu i prikazuje je

drugim korisnicima. Ova funkcionalnost pomaže u poboljšanju kvaliteta igara i pruža korisnicima mogućnost da podijele svoja iskustva.



Slika 9: UML dijagram aktivnosti za sistem ocjenjivanja

IDENTIFICIRANJE MOGUĆIH PROBLEMA

1. Problem: Ponavljanje koda u klasama

U sistemu postoji više klase koje dijele slične funkcionalnosti, kao što su Korisnik, Igra, Rezervacija, Ocjena, i PrijavaKorisnika. Na primjer, sve ove klase imaju svojstvo ID i metode za upravljanje tim podacima. Ako se ove funkcionalnosti implementiraju zasebno u svakoj klasi, to može dovesti do ponavljanja koda (DRY princip kršenje).

Rješenje: Nasljeđivanje

Možemo kreirati apstraktну baznu klasu (npr., Entitet) koja će sadržavati zajednička svojstva i metode, kao što su ID i osnovne CRUD operacije. Ostale klase mogu naslijediti ovu baznu klasu, čime se smanjuje ponavljanje koda i povećava modularnost.

2. Problem: Različiti tipovi korisnika

U sistemu postoji potreba za razlikovanjem različitih tipova korisnika (npr., obični korisnici i administratori). Trenutno, ova funkcionalnost je implementirana kroz svojstvo TipKorisnika u klasi Korisnik. Međutim, ovo može postati problematično ako se dodaju novi tipovi korisnika ili ako svaki tip zahtijeva različite funkcionalnosti.

Rješenje: Interfejsi i nasljeđivanje

Možemo koristiti interfejs i nasljeđivanje za definisanje specifičnih funkcionalnosti za svaki tip korisnika. Na primjer, interfejs IAdmin može definirati metode za upravljanje igram, dok obični korisnici mogu koristiti interfejs IKorisnik.

3. Problem: Upravljanje različitim vrstama igara

U sistemu postoji potreba za upravljanjem različitim vrstama VR igara (npr., akcijske, avanturističke, horor). Trenutno, ovo je implementirano kroz svojstvo Zanr u klasi Igra. Međutim, ako svaka vrsta igre zahtijeva različite funkcionalnosti, ovo može postati problematično.

Rješenje: Polimorfizam i interfejsi

Možemo koristiti polimorfizam i interfejse za definisanje specifičnih funkcionalnosti za svaku vrstu igre. Na primjer, interfejs IIgra može definirati osnovne metode, dok svaka vrsta igre može implementirati svoje specifične funkcionalnosti.

4. Problem: Upravljanje ocjenama i komentarima

Ocjene i komentari su povezani sa igrami i korisnicima, ali trenutno su implementirani kao zasebne klase. Ovo može dovesti do problema u održavanju i proširivanju funkcionalnosti.

Rješenje: Kompozicija i interfejsi

Možemo koristiti kompoziciju za povezivanje ocjena i komentara sa igrami i korisnicima. Takođe, možemo koristiti interfejse za definisanje funkcionalnosti vezanih za ocjenjivanje.

ARHITEKTURA SISTEMA

Ovaj informacioni sistem koristi klijent-server arhitekturu za razdvajanje frontend i backend aplikacija, te MVC (Model-View-Controller) šablon za organizaciju server-side aplikacije. Ova kombinacija omogućava modularnost, skalabilnost i lakše održavanje sistema.

1. Klijent-Server Arhitektura

Klijent-server model dijeli sistem na tri glavne komponente: klijent, server, i baza podataka. Ova arhitektura omogućava jasno razdvajanje odgovornosti i efikasnu komunikaciju između korisnika i sistema.

Komponente Klijent-Server Arhitektura

Klijent (Frontend Aplikacija)

Klijentska aplikacija omogućava korisnicima interakciju sa sistemom. Implementirana je pomoću modernih web tehnologija kao što su React.js, HTML, CSS, i JavaScript. Klijent komunicira sa serverom putem HTTP zahtjeva, šaljući podatke i primajući odgovore u realnom vremenu.

Server (Backend Aplikacija)

Server je zadužen za obradu zahtjeva klijenata, izvršavanje poslovne logike i komunikaciju sa bazom podataka. Implementiran je pomoću ASP.NET Core, koji pruža visoku performans i skalabilnost. Server takođe rukuje autentifikacijom, autorizacijom i sigurnošću podataka, osiguravajući da su svi zahtjevi validni i zaštićeni.

Baza Podataka (SQL Server)

Baza podataka služi za čuvanje svih podataka sistema, uključujući informacije o korisnicima, igrama, rezervacijama, ocjenama i prijavama. Koristi se SQL Server, relacioni sistem za upravljanje bazom podataka, koji omogućava efikasno skladištenje i manipulaciju podacima putem SQL upita.

Kako Klijent-Server Arhitektura Funkcioniše u VR Escape Room Sistemu?

Klijent: Web aplikacija omogućava korisnicima da se registruju, prijave, pretražuju i rezervišu igre.

Server: Backend aplikacija prima zahteve od klijenta, obrađuje ih, komunicira sa bazom podataka i vraća odgovarajuće podatke.

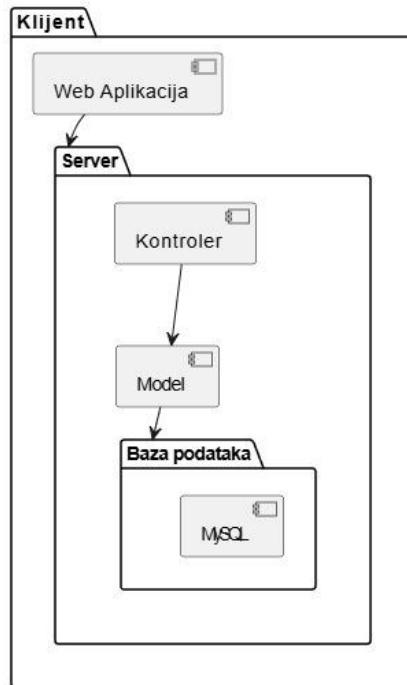
Baza Podataka: Čuva sve potrebne informacije, kao što su korisnički profili, detalji o igrama, rezervacije i ocjene.

Prednosti Klijent-Server Arhitekture

Odvajanje Odgovornosti: Frontend i backend se mogu razvijati i testirati odvojeno, što povećava efikasnost timova.

Skalabilnost: Sistem se lako može proširiti dodavanjem novih servera ili poboljšanjem postojećih resursa.

Sigurnost: Podaci su zaštićeni na serveru, a klijentska aplikacija prima samo potrebne informacije, smanjujući rizik od sigurnosnih propusta.



Slika 10: Klijent-server arhitektura

2. MVC (Model-View-Controller) Arhitektura

MVC je obrazac dizajna softvera koji se koristi za organizovanje server-side aplikacije. Dijeli aplikaciju na tri glavne komponente: Model, View, i Controller. Ova struktura omogućava jasno razdvajanje logike, podataka i korisničkog interfejsa.

Komponente MVC Arhitekture

Model (M)

Model predstavlja poslovnu logiku i podatke sistema. Komunicira direktno sa bazom podataka, upravlja podacima i obavlja operacije kao što su čitanje, pisanje i ažuriranje. U VR Escape Room sistemu, modeli uključuju klase kao što su Korisnik, Igra, Rezervacija, Ocjena, i PrijavaKorisnika.

View (V)

View je odgovoran za prikaz podataka korisnicima kroz korisnički interfejs. U ovom sistemu, View komponente su implementirane pomoću Razor Pages i HTML/CSS, omogućavajući dinamičko generisanje web stranica na osnovu podataka dobijenih od Modela.

Controller (C)

Controller upravlja zahtjevima korisnika, koordinira komunikaciju između Modela i View-a. Prima zahtjeve od klijenta, poziva odgovarajuće metode u Modelu za obradu podataka, a zatim vraća rezultate u View za prikaz. U VR Escape Room sistemu, kontroleri uključuju KorisniksController, IgrasController, RezervacijasController, i druge.

Kako MVC Funkcioniše u VR Escape Room Sistemu?

Korisnik izvršava akciju u frontend aplikaciji (npr., klikne na "Rezerviši igru").

Zahtjev se šalje odgovarajućem kontroleru (npr., RezervacijasController).

Kontroler komunicira sa Modelom (npr., Rezervacija) kako bi dohvatio ili ažurirao podatke u bazi.

Model vraća podatke kontroleru, koji zatim prosljeđuje te podatke View-u.

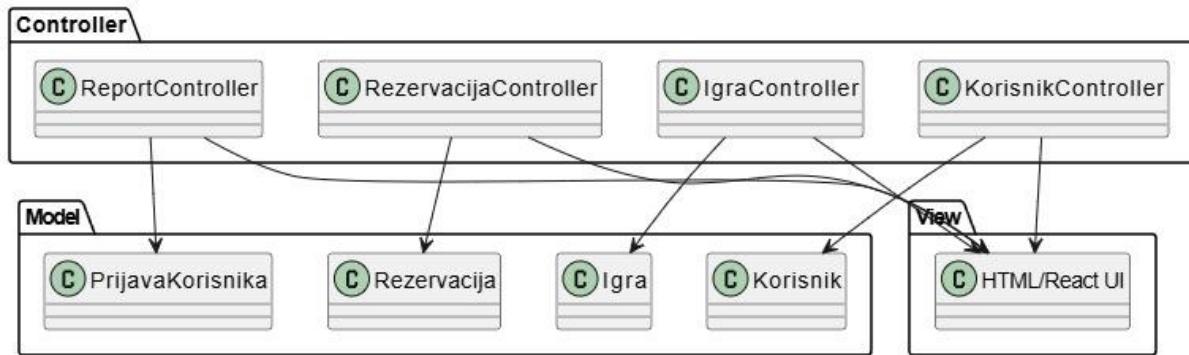
View generiše odgovarajući korisnički interfejs (npr., stranicu sa potvrdom rezervacije) i prikazuje ga korisniku.

Prednosti MVC Arhitekture

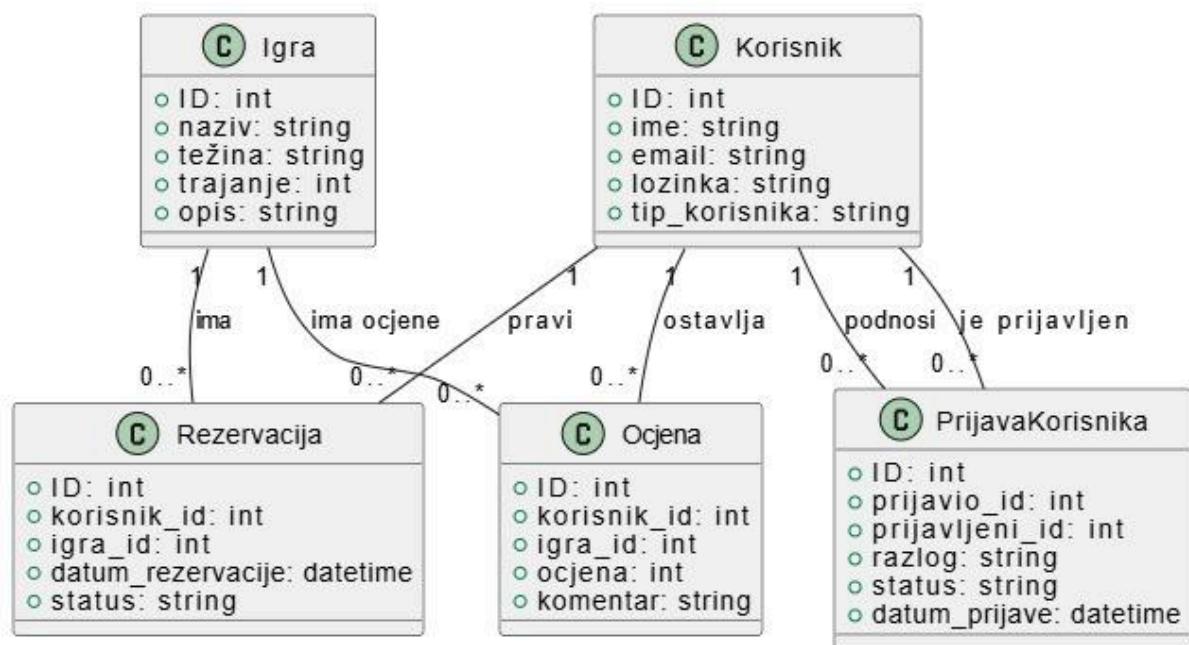
Modularnost: Svaka komponenta (Model, View, Controller) može se razvijati i mijenjati nezavisno, što olakšava razvoj i održavanje.

Lakše Održavanje: Logika aplikacije je jasno odvojena od korisničkog interfejsa, što smanjuje kompleksnost i olakšava oticanje grešaka.

Ponovna Upotreba Koda: Modeli i kontroleri mogu biti korišteni od strane više različitih View komponenti, smanjujući ponavljanje koda.



Slika 11: UML dijagram komponenti



Slika 12: Dijagram klasa prilagođen bazi podataka

ARHITEKTURA ASP.NET WEB APLIKACIJE

MODEL PODATAKA U C#

U ovom dijelu projekta korišten je Code-First pristup u Entity Frameworku, što omogućava da se modeli podataka definiraju direktno u kodu, a zatim generiraju odgovarajuće tabele u bazi podataka. Svi modeli podataka dodani su u direktorij: VirtualRealityEscapeRooms\VirtualRealityEscapeRooms\Models



Slika 13: Prikaz modela u solution exploreru

Jedan od ključnih modela u aplikaciji je Igra.cs, koji predstavlja entitet igre u sistemu.

Struktura modela uključuje:

Osnovne informacije (Naziv, Težina, Trajanje, Opis, Žanr)

Slika igre (SlikaPath – čuva putanju za sliku)

Relacije s drugim modelima (Veze s Ocjenama i Rezervacijama)

```
namespace VR_escape_rooms.modeli
{
    public class Igra
    {
        public int ID { get; set; }
        public string Naziv { get; set; }
        public Tezina Tezina { get; set; } // 'lako', 'srednje', 'tesko'
        public int Trajanje { get; set; }
        public string Opis { get; set; }
        public Zanr Zanr { get; set; }

        public string? SlikaPath { get; set; }

        [InverseProperty("Igra")]
        public ICollection<Rezervacija> Rezervacije { get; set; }
        public ICollection<Ocjena> Ocijene { get; set; } = new List<Ocjena>();

        public Igra()
        {
            Rezervacije = new List<Rezervacija>();
        }
    }
}
```

Slika 14: Igra.cs - primjer modela

DbContext U C#

Dodan predefinisani kontekst u Data/ApplicationDbContext.cs sa definicijama.

```

namespace VirtualRealityEscapeRooms.Data
{
    37 references
    public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
    {
        0 references
        public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
            : base(options)
        {
        }
    }

    39 references
    public DbSet<Korisnik> Korisnici { get; set; }
    28 references
    public DbSet<Igra> Igre { get; set; }
    15 references
    public DbSet<Rezervacija> Rezervacije { get; set; }
    13 references
    public DbSet<Ocjena> Ocjene { get; set; }
    14 references
    public DbSet<PrijavaKorisnika> PrijaveKorisnika { get; set; }
}

```

Slika 15: Definicije u ApplicationContext.cs

Dodana metoda OnModelCreating koja vrši mapiranje između atributa konteksta baze podataka i tabela baze podataka

```

base.OnModelCreating(modelBuilder);

modelBuilder.Entity<Korisnik>().ToTable("Korisnik");
modelBuilder.Entity<Igra>().ToTable("Igra");
modelBuilder.Entity<Rezervacija>().ToTable("Rezervacija");
modelBuilder.Entity<Ocjena>().ToTable("Ocjena");
modelBuilder.Entity<PrijavaKorisnika>().ToTable("PrijavaKorisnika");

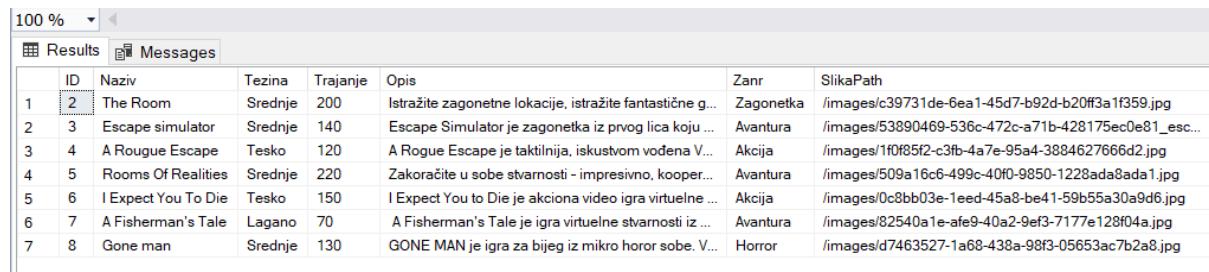
modelBuilder.Entity<Korisnik>().HasKey(k => k.ID);
modelBuilder.Entity<Igra>().HasKey(i => i.ID);
modelBuilder.Entity<Rezervacija>().HasKey(r => r.ID);
modelBuilder.Entity<Ocjena>().HasKey(o => o.ID);
modelBuilder.Entity<PrijavaKorisnika>().HasKey(p => p.ID);

```

Slika 16: Izgled metode OnModelCreating

PUNJENJE BAZE PODATAKA PODACIMA

Prikazana je Igra tabela unutar baze podataka, popunjena unosima koji predstavljaju različite VR escape room igre. Svaki red sadrži podatke o igri, uključujući njen naziv, težinu, trajanje, opis, žanr i putanju slike. Struktura tabele jasno prikazuje organizaciju podataka, omogućavajući lakšu manipulaciju i pretragu unutar sistema.



The screenshot shows a database query results table with the following columns: ID, Naziv, Tezina, Trajanje, Opis, Zanr, and SlikaPath. The data is as follows:

ID	Naziv	Tezina	Trajanje	Opis	Zanr	SlikaPath
1	2 The Room	Srednje	200	Istražite zagonetne lokacije, istražite fantastične g...	Zagonetka	/images/c39731de-6ea1-45d7-b92d-b20ff3a1f359.jpg
2	3 Escape simulator	Srednje	140	Escape Simulator je zagonečka iz prvog lica koju ...	Avantura	/images/53890469-536c-472c-a71b-428175ec0e81_esc...
3	4 A Rouge Escape	Tesko	120	A Rogue Escape je taktilnija, iskustvom vođena V...	Akcija	/images/1f0f85f2-c3fb-4a7e-95a4-3884627666d2.jpg
4	5 Rooms Of Realities	Srednje	220	Zakoračite u sobe stvarnosti - impresivno, kooper...	Avantura	/images/509a16c6-499c-40f0-9850-1228ada8ada1.jpg
5	6 I Expect You To Die	Tesko	150	I Expect You to Die je akciona video igra virtuelne ...	Akcija	/images/0c8bb03e-1eed-45a8-be41-59b55a30a9dd6.jpg
6	7 A Fisherman's Tale	Lagano	70	A Fisherman's Tale je igra virtuelne stvarnosti iz ...	Avantura	/images/82540a1e-afe9-40a2-9ef3-7177e128f04a.jpg
7	8 Gone man	Srednje	130	GONE MAN je igra za bijeg iz mikro horor sobe. V...	Horror	/images/d7463527-1a68-438a-98f3-05653ac7b2a8.jpg

Slika 17: Prikaz tabele Igra iz baze podataka

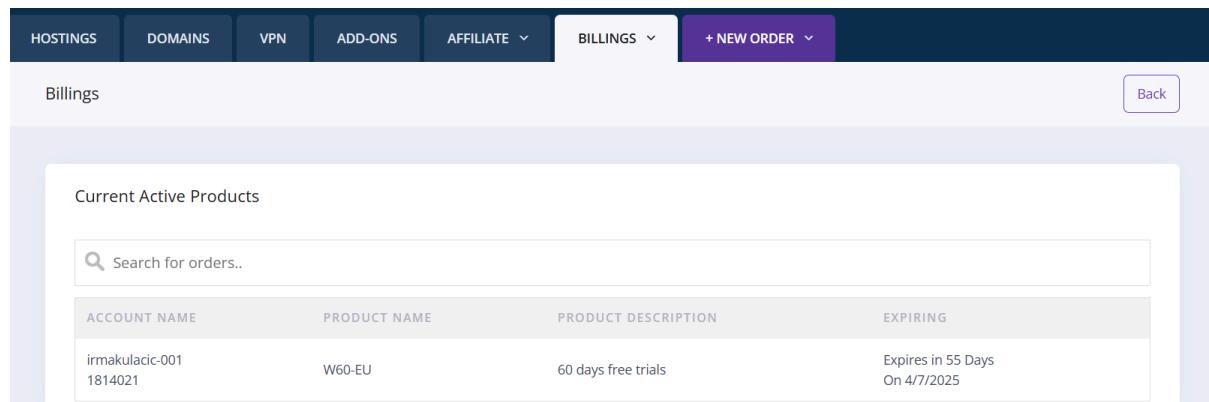
IMPLEMENTACIJA I UPRAVLJANJE CLOUD BAZOM PODATAKA

EXPORT I REIMPORT BAZE PODATAKA

U ovom projektu koristi se MSSQL baza podataka hostovana na SmarterASP.NET platformi, što omogućava jednostavan izvoz (export) i ponovni uvoz (reimport) cijelokupnog stanja baze podataka. Ova funkcionalnost osigurava očuvanje podataka i integriteta relacija, omogućavajući migraciju baze ili njeno vraćanje u slučaju kreiranja novog korisničkog računa ili premještanja aplikacije na drugi server.

CLOUD HOSTING I POVEZIVANJE BAZE PODATAKA

Projekt je hostovan koristeći SmarterASP.NET, dok se baza podataka upravlja putem Microsoft SQL Servera, a aplikacija se povezuje s bazom putem Entity Framework Core (EF Core) ORM-a.

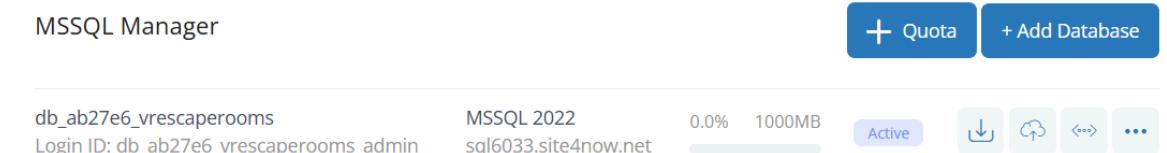


The screenshot shows an administrator dashboard with a navigation bar at the top containing links for HOSTINGS, DOMAINS, VPN, ADD-ONS, AFFILIATE, BILLINGS (selected), and + NEW ORDER. Below the navigation bar, there is a search bar labeled "Billings". The main content area is titled "Current Active Products" and contains a table with the following data:

ACCOUNT NAME	PRODUCT NAME	PRODUCT DESCRIPTION	EXPIRING
irmakulacic-001 1814021	W60-EU	60 days free trials	Expires in 55 Days On 4/7/2025

Slika 18: Kreiran administratorski račun

Konekcija između aplikacije i baze podataka ostvarena je pomoću konekcijskog stringa, koji je sigurno pohranjen u appsettings.json konfiguracijskoj datoteci. Implementirana je code-first metodologija, pri čemu se migracije koriste za upravljanje bazom podataka i njenim strukturama.



Slika 19: Konfiguracija cloud baze podataka

```
".ConnectionStrings": {  
    "DefaultConnection": "Data Source=SQL6033.site4now.net;Initial Catalog=db_ab27e6_vrescaperooms;  
    User Id=db_ab27e6_vrescaperooms_admin;Password=nesto_random_1$"  
},
```

Slika 20: Konekcioni string

Ovaj string omogućava sigurno povezivanje ASP.NET aplikacije s MSSQL bazom podataka hostovanom na SmarterASP.NET serveru.

MODEL BAZE PODATAKA I RELACIJE

Baza podataka je kreirana koristeći **relacijski model** i sastoji se od nekoliko ključnih tabela:

- **AspNetUsers** – Sadrži podatke o registrovanim korisnicima, uključujući autentifikaciju putem ASP.NET Identity sistema.
- **Korisnik** – Dodatni podaci o korisnicima, uključujući uloge, rezultate igara i identifikacione podatke.
- **Igra** – Informacije o dostupnim VR igrama, uključujući težinu, trajanje i žanr.
- **Rezervacija** – Podaci o rezervacijama igara, povezujući korisnike s igrama.
- **PrijavaKorisnika** – Evidencija prijava korisnika za administrativnu obradu.

Entity Framework Core je korišten za definiranje relacija između entiteta, uključujući primarne i strane ključeve.

MIGRACIJE BAZE PODATAKA

Za usklađivanje modela aplikacije s bazom podataka korištene su Entity Framework Core migracije.

Proces migracije

Inicijalna migracija – Kreiranje početne strukture baze podataka.

PM> Add-Migration InitialCreate

PM> Update-Database

Modifikacija tabele Korisnik – Uklanjanje polja Lozinka kako bi se podaci o autentifikaciji koristili samo kroz ASP.NET Identity.

PM> Add-Migration RemoveLozinkaFromKorisnik

PM> Update-Database

Validacija podataka – Nakon svake migracije, podaci su provjereni putem SSMS-a i DBeavera kako bi se osigurala ispravnost tabela i relacija.

Webbase MS SQL Server Manager



Slika 21: Konekcija sa bazom nakon izvršene migracije pristupljenoj putem Webconnect opcije

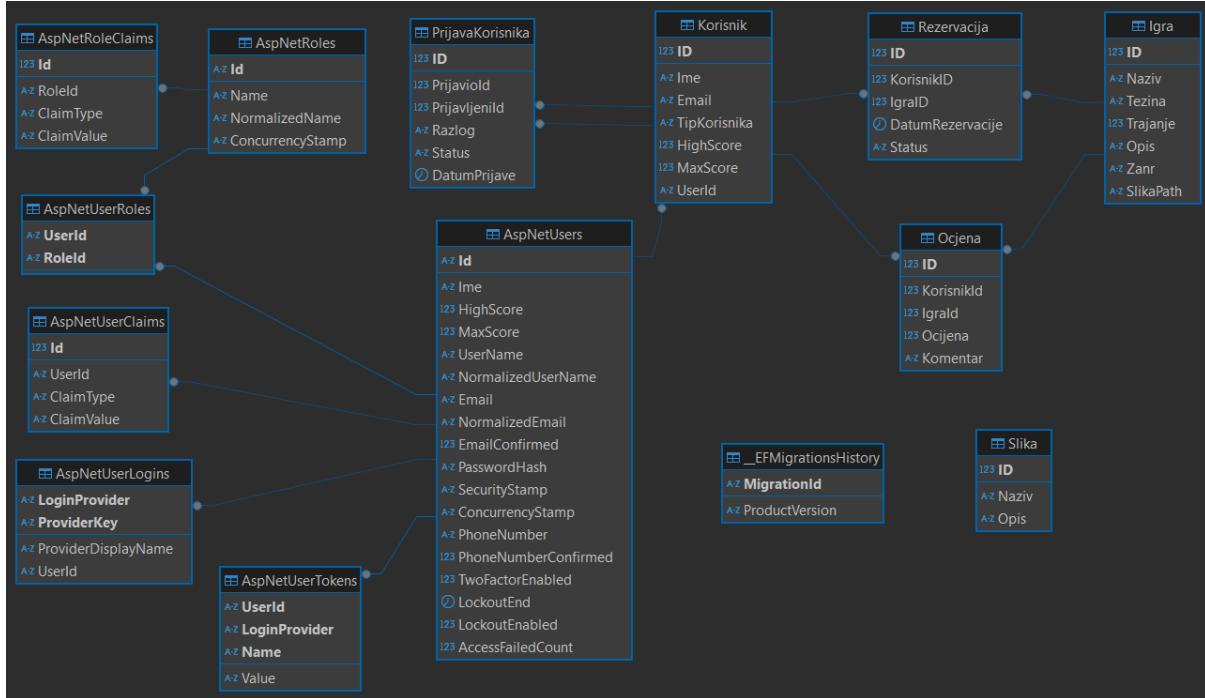
Slika 22: Pristup cloud bazi podataka

VIZUALIZACIJA BAZE PODATAKA I ERD ANALIZA

Za analizu i provjeru ispravnosti strukture baze podataka kreiran je ER dijagram (Entity-Relationship Diagram), koji omogućava:

- Pregled primarnih i stranih ključeva.
- Validaciju relacija između entiteta (Rezervacija povezana s Korisnik i Igra).
- Provjeru međutabela i njihove uloge u relacijama.

ERD je generisan pomoću DBeaver alata i eksportovan u dokumentaciju kao vizualni prikaz arhitekture baze podataka.



Slika 23: ERD koji odgovara modelu nakon izvršene migracije

BACKEND IFORMACIONOG SISTEMA

U sklopu razvoja aplikacije VirtualRealityEscapeRooms, implementirana je moderna MVC arhitektura (Model-View-Controller) koristeći ASP.NET Core i Entity Framework Core. Ovaj dokument opisuje način na koji su kontroleri aplikacije generisani, kako se vrši komunikacija između backenda i baze podataka te kako je backend organizovan kako bi omogućio efikasno upravljanje podacima i korisničkim interakcijama.

POČETNO STANJE NAKON MIGRACIJE

U inicijalnoj fazi razvoja kreirana je struktura projekta sa sljedećim ključnim direktorijima:

- Models/ – Definicija modela podataka i njihovih odnosa.
 - Data/ – Konfiguracija baze podataka i konekcija putem Entity Framework Core-a.
 - Controllers/ – Sadrži kontrolere koji obrađuju zahtjeve korisnika i omogućavaju interakciju s bazom podataka.

Uspješno je izvršena inicijalna migracija baze podataka, čime su kreirane sve potrebne tabele i odnosi u Microsoft SQL Server-u (MSSQL).

AUTOMATSKO GENERISANJE KONTROLERA

Kontroleri su osnovna komponenta MVC arhitekture, a njihova primarna funkcija je rukovanje korisničkim zahtjevima i interakcija s modelima podataka.

U ovom projektu, kontroleri su generisani automatski pomoću scaffolding tehnike u ASP.NET-u, što je omogućilo brzo kreiranje osnovnih CRUD operacija (kreiranje, čitanje, ažuriranje i brisanje podataka) nad modelima.

Funkcionalnosti koje kontroleri obavljaju:

- Pristup podacima iz baze putem GET metoda.
- Kreiranje novih zapisa u bazi putem POST metoda.
- Ažuriranje podataka pomoću PUT metoda.
- Brisanje zapisa iz baze pomoću DELETE metoda.

GENERISANJE KONTROLERA NA PRIMJERU IgraController

Kao primjer, prikazujemo način na koji je kreiran kontroler za upravljanje igrama (IgrasController), koristeći Entity Framework Core.

Koraci generisanja kontrolera:

- Desni klik na folder Controllers → Add → New Scaffolded Item...
- Odabir opcije API Controller with actions, using Entity Framework
- Povezivanje kontrolera s ApplicationDbContext kontekstom baze podataka
- Definisanje naziva kontrolera (IgrasController)
- Ovim postupkom kreiran je IgrasController, koji omogućava interakciju s bazom podataka putem RESTful API zahtjeva.

ASP.NET Web API koristi različite HTTP metode za rukovanje podacima:

- GET – Dohvaća podatke iz baze (GetIgre, GetIgra).
- POST – Kreira nove zapise (PostIgra).
- PUT – Ažurira postojeće podatke (PutIgra).
- DELETE – Briše zapis (DeleteIgra).

Svaka metoda vraća IActionResult, koji omogućava fleksibilno rukovanje odgovorima (npr. NotFound(), CreatedAtAction()).

TESTIRANJE API-JA POMOĆU POSTMANA

Za provjeru ispravnosti API-ja korišten je Postman, alat za testiranje HTTP zahtjeva.

Testiranje GET zahtjeva:

- Pokrenuta aplikacija (<https://localhost:7221/api/Igras>).
- Unesen GET zahtjev u Postman → vraća listu igara.

Testiranje POST zahtjeva:

- Odabran POST zahtjev.
- Uneseni podaci u JSON formatu
- Nakon slanja zahtjeva, igra je uspješno kreirana i dodana u bazu.

Testiranje DELETE zahtjeva:

- Pokrenut DELETE zahtjev (<https://localhost:7221/api/Igras/5>).
- Igra sa ID=5 uspješno je obrisana.

The screenshot shows the Postman application interface. On the left, there's a sidebar titled 'History' listing various API requests made. In the center, a request is being viewed for 'GET https://localhost:7015/api/Igra'. The 'Params' tab is selected. The 'Body' tab shows a JSON response:

```

1  {
2   "id": 2,
3   "naziv": "The Room",
4   "tezina": 2,
5   "trajanje": 200,
6   "opis": "Istražite zagonetne lokacije, istražite fantastične gadžete i otkrijte onostrano otkriće koje briše granicu između stvarnosti i iluzije.",
7   "zanr": 2,
8   "slikaPath": "/images/c39731de-6ea1-45d7-b92d-b20ff3a1f359.jpg",
9   "rezervacije": []

```

At the bottom right, the status bar indicates '200 OK 2.31 s 2.47 KB'. A 'Send' button is visible at the top right of the main request view.

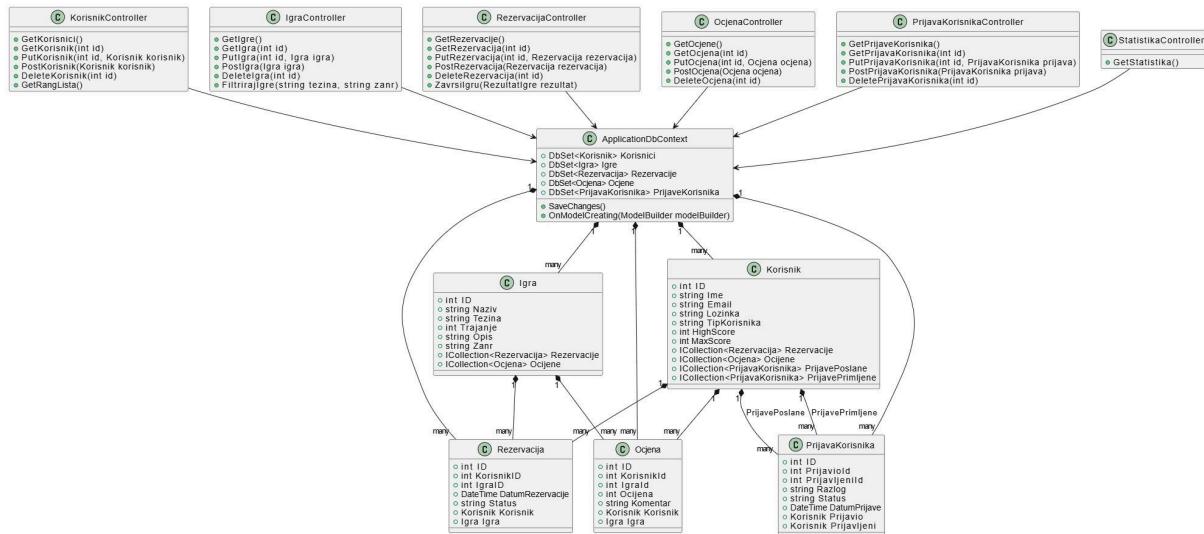
Slika 24: Testiranje u Postmanu

ARHITEKTURA BACKENDA I MVC PRINCIP

Aplikacija je organizovana prema MVC (Model-View-Controller) principu, što omogućava jasnu separaciju poslovne logike, podataka i korisničkog interfejsa.

Struktura backenda:

- Models/ – Definicija podataka (Igra, Korisnik, Rezervacija).
- Controllers/ – API kontroleri (IgrasController, RezervacijeController).
- Data/ – Konekcija s bazom (ApplicationDbContext).

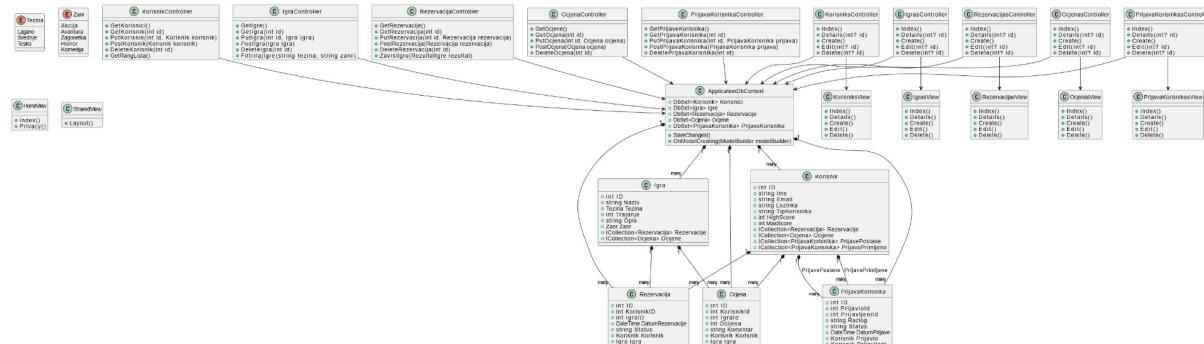


Slika 25: MVC dijagram klasa sa kontrolerima

DIZAJN KORISNIČKIH INTERFEJSA

Ovaj dio obuhvaća ključne korake u razvoju aplikacije: Scaffolding i dodavanje kontrolera kako bi se uspostavila osnovna struktura i logika aplikacije, prilagodba template-a pomoću Bootstrapa, te rješavanje problema s enumeracijama kako bi se održao intuitivan dizajn. Također, uključena je validacija podataka kako bi se osigurala ispravnost i sigurnost unosa, te korištenje podataka iz backenda za dinamičko prikazivanje informacija na frontenu.

FINALIZIRAN MVC DIJAGRAM



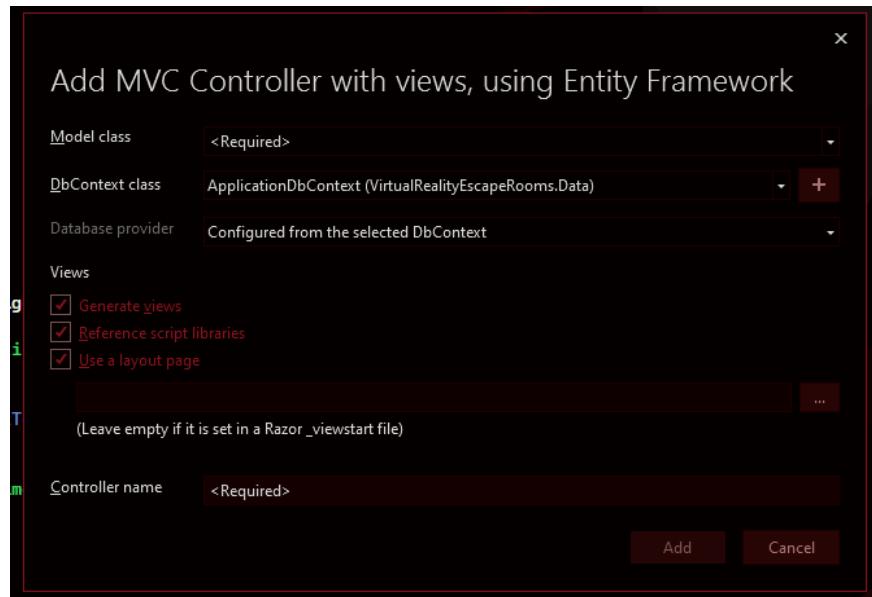
Slika 26: MVC dijagram sa svim promjenama

DODAVANJE FRONTEND KOMPONENTI SA SCAFFOLD

Kako bi se izbjegli potencijalni problemi sa rutama koje imaju isto ime (npr. /Korisnik), postojale su dvije mogućnosti:

1. Brisanje prethodno generisanog kontrolera
2. Zadržavanje prethodno generisanih kontrolera – uz imenovanje novih kontrolera na drugačiji način kako se ne bi poklapale rute.

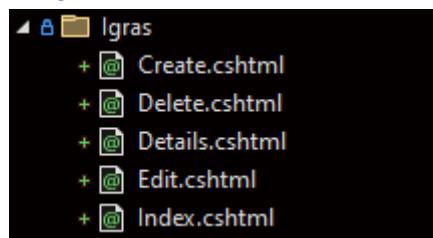
Odabrana je druga opcija, nakon čega je izvršeno generisanje novih kontrolera sa odgovarajućim pogledima korištenjem Scaffold items. Za svaki model, odabrana je opcija “Add MVC Controller with Views, using Entity Framework”, čime su automatski kreirani potrebni kontroleri i pogledi za dalji rad.



Slika 27: Dodavanje kontrolera pomoću scaffold-a

U folderu Views također su dodane nove komponente - novi folder istog imena kao generisani kontroler sa pet pogleda:

1. Create - forma za dodavanje nove igre
2. Delete - forma za brisanje postojeće igre
3. Details - forma za prikaz svih atributa postojeće igre
4. Edit - forma za promjenu atributa postojeće igre
5. Index - forma za prikaz liste svih igara



Slika 28: Generisani pogledi

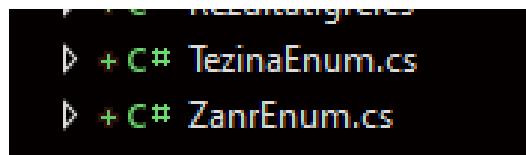
Na ruti /Igras se sada u tabelarnom prikazu vide svi unjeti podaci u bazu za igre.

ESCAPE ROOM IGRE					
Dodaj novu igru					
NAZIV	TEZINA	TRAJANJE	OPIS	ZANR	
The Room	Srednje	200	Istražite zagonetne lokacije, istražite fantastične gadžete i otkrijte onostrano otkriće koje briše granicu između stvarnosti i iluzije.	Zagonetka	Uredi Detalji Obrisи
Escape Simulator	Tesko	150	Escape Simulator je zagonetka iz prvog lica koju možete igrati solo ili u kooperaciji. Istražite sve veći skup visoko interaktivnih soba za bijeg. Premjestite namještaj, pokupite i pregledajte sve, razbijte lonce i razbijte brave!	Avantura	Uredi Detalji Obrisи
A Rogue Escape	Tesko	120	A Rogue Escape je taktilnija, iskustvom vodena VR re-imaginacija prvog naslova Spare Parts Oasis: Nauticrawl.	Akcija	Uredi Detalji Obrisи

Slika 29: Izgled /Igras

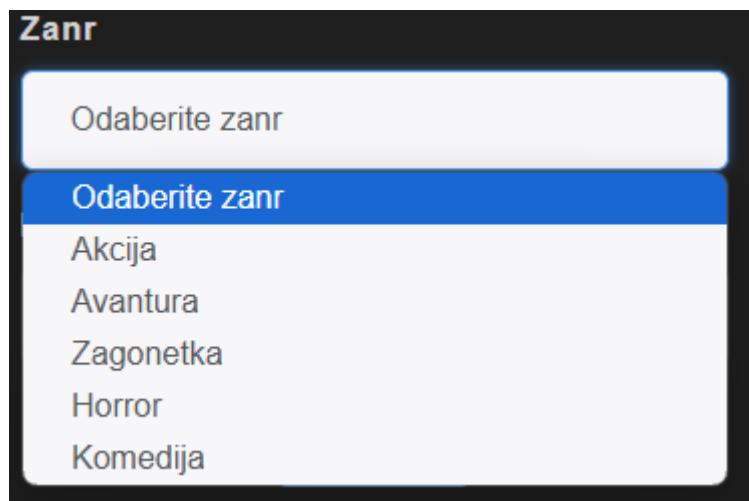
ISPRAVAN PRIKAZ ENUMERACIJSKIH TIPOVA

Kako bi se umjesto polja za unos teksta stavio dropdown meni sa izborima, prvo je potrebno dodati odgovarajuće klase za enumeracije. U ovom slučaju dodane su ZanrEnum.cs i TezinaEnum.cs. Također je potrebno updateati Igra model da tip podatka odgovara enumeracijama npr. String Zanr se zamjeni sa Zanr Zanr.



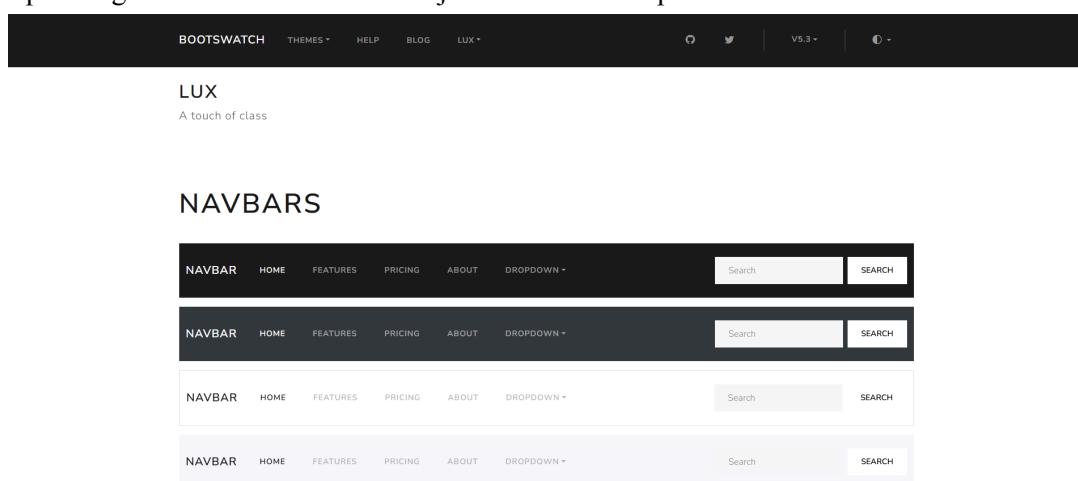
Slika 30: Enum klase

Nakon promjena u Create.cshtml da se za polje “Zanr” pojavi dropdown meni, to ovako izgleda:



Slika 31: Žanr dropdown meni

Za template izgleda web stranice korišten je Bootswatch template LUX



Slika 32: Izgled LUX template

KORIŠTENJE PODATAKA SA BACKENDA U FRONTENDU

Primjer može biti korištenje DisplayNameFor za prikazivanje naziva atributa i DisplayFor za prikazivanje sadržaja atributa.

```
<tbody>
    <tr>
        <th>@Html.DisplayNameFor(model => model.Naziv)</th>
        <td>@Html.DisplayFor(model => model.Naziv)</td>
    </tr>
    <tr>
        <th>@Html.DisplayNameFor(model => model.Tezina)</th>
        <td>@Html.DisplayFor(model => model.Tezina)</td>
    </tr>
    <tr>
        <th>@Html.DisplayNameFor(model => model.Trajanje)</th>
        <td>@Html.DisplayFor(model => model.Trajanje) min</td>
    </tr>
    <tr>
        <th>@Html.DisplayNameFor(model => model.Opis)</th>
        <td>@Html.DisplayFor(model => model.Opis)</td>
    </tr>
    <tr>
        <th>@Html.DisplayNameFor(model => model.Zanr)</th>
        <td>@Html.DisplayFor(model => model.Zanr)</td>
    </tr>
</tbody>
```

Slika 33: Prikazivanje detalja igrice

VALIDACIJA PODATAKA

Unos null podataka se može omogućiti dodavanjem upitnika pored deklaracije tipa atributa. U ovom slučaju to je zaobideno i na slici su prikazani errori koje korisnik dobije ako ne unese podatke. U ovom kodu to je postignuto koristeći pomoćnog tag-a “asp-validation-for”

The screenshot shows a form titled "IGRA" with four input fields: "Naziv", "Tezina", "Trajanje", and "Opis". Each field has a red border and a validation message below it:

- "Naziv": "The Naziv field is required."
- "Tezina": "The Tezina field is required."
- "Trajanje": "The Trajanje field is required."
- "Opis": "The Opis field is required."

The underlying ASP.NET code for the Trajanje field is as follows, with the validation part highlighted:

```
<!-- Polje trajanje -->


<label asp-for="Trajanje" class="form-label"></label>
    <input asp-for="Trajanje" class="form-control" />
    <span asp-validation-for="Trajanje" class="text-danger"></span>


```

Slika 34: Implementacija validacije

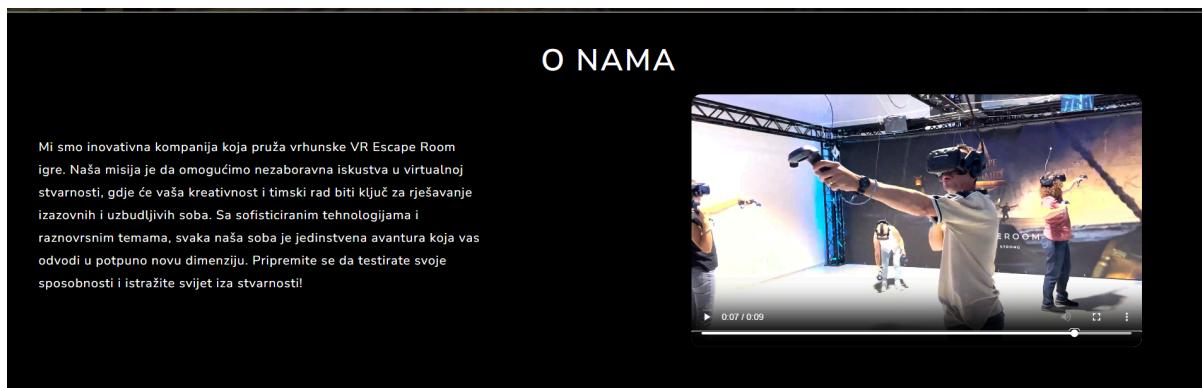
FINALNI IZGLED WEB DIZAJNA

Na početnoj stranici nalazi se poruka dobrodošlice, ispod toga 3 dijela koja na jednostavan način potiču korisnika da istraži opcije i izvrši rezervaciju. Ispod toga se nalaze utisci prethodnih korisnika. Iza svega se nalazi pokretna pozadina koja odgovara temi stranice. Na video je dodan tamni overlay kako bi kontrast i čitljivost bili veći.



Slika 35: Izgled početne stranice

Kada se scrolla, prikaže se dio koji daje više informacija o kompaniji, te video pored koji se može pustiti i pauzirati.



Slika 36: O nama sekcija stranice

I naravno u Layout.cshtml nalaze se navbar i footer koji su prisutni na svakoj stranici. Navbar sadrži dugmad koja vode na početnu, pravila privatnosti, ponuda VR igara te dugmad za login i register. Navbar se mijenja u skladu sa korisnikovim statusom(nije prijavljen, prijavljen kao obični korisnik, prijavljen kao admin). Status logged in dodaje dugme logout i adminu dodaje admin panel.



Slika 37: Razlika u izgledu navbar-a korisnika

RAZVOJ INFORMACIONIH SISTEMA - PROJEKTNI ZADATAK

The screenshot shows a registration form titled "REGISTER". It includes fields for Name (Adna), Email (adna@gmail.com), Password, Confirm Password, Role (User), and a "Register" button. Below the form is a link to log in.

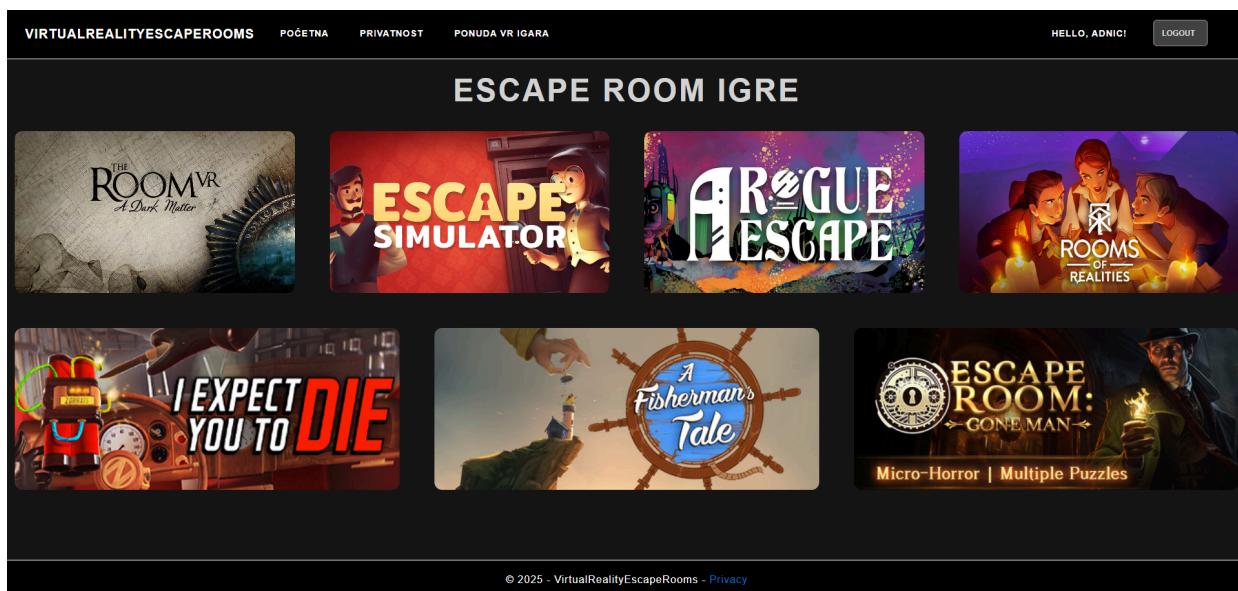
Slika 38: Register dizajn

The screenshot shows a login form titled "LOGIN". It includes fields for Email (adna@mail.com) and Password, a "Remember Me" checkbox, and a "Login" button. Below the form is a link to register.

Slika 39: Login dizajn

The screenshot shows the "POLITIKA PRIVATNOSTI" (Privacy Policy) page. It contains sections: "KOJE INFORMACIJE PRIKUPLJAMO?", "KAKO KORISTIMO VAŠE PODATKE?", "SIGURNOSNE MJERE", "VAŠA PRAVA", and "KONTAKT". The page also includes a "NAZAD NA POČETNU" (Back to Home) button at the bottom.

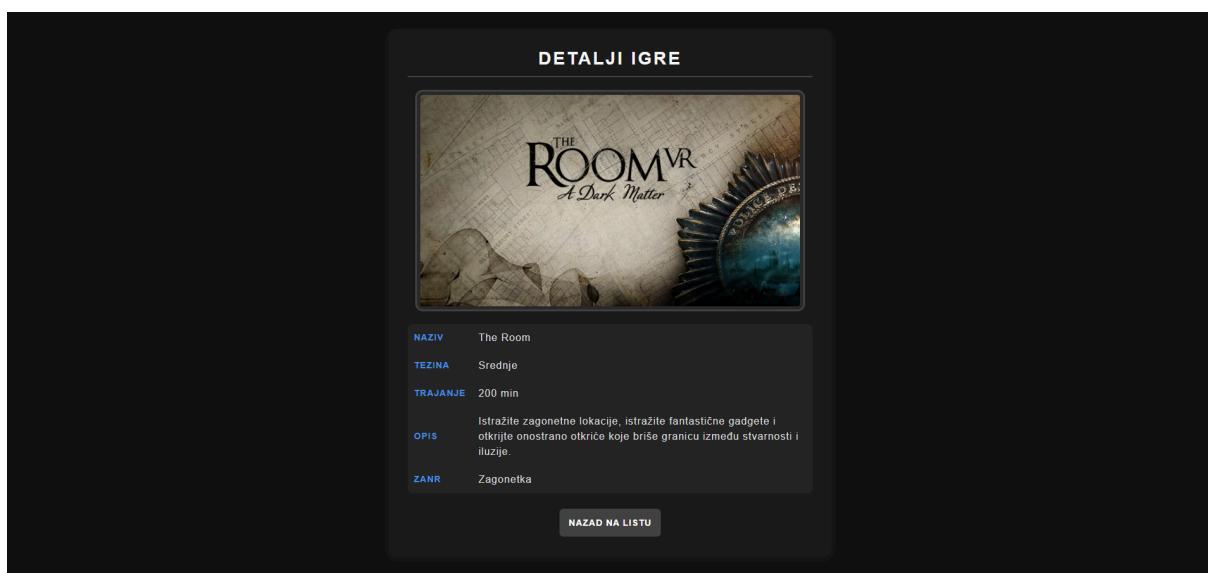
Slika 40: Politika privatnosti dizajn



Slika 41: Ponuda VR igara dizajn

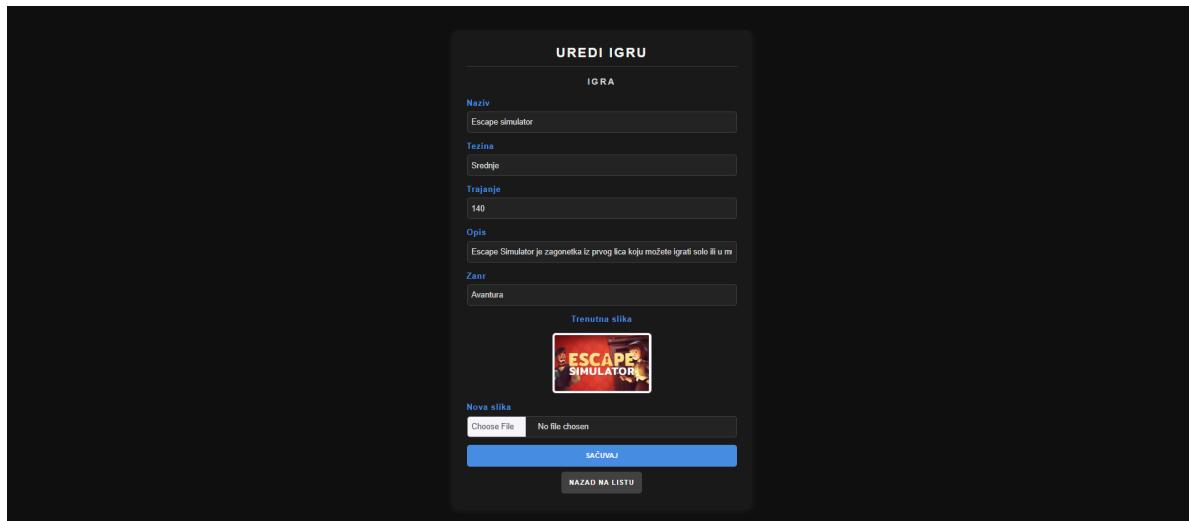


Slika 42: Razlika hover-a igre za korisnika i admina

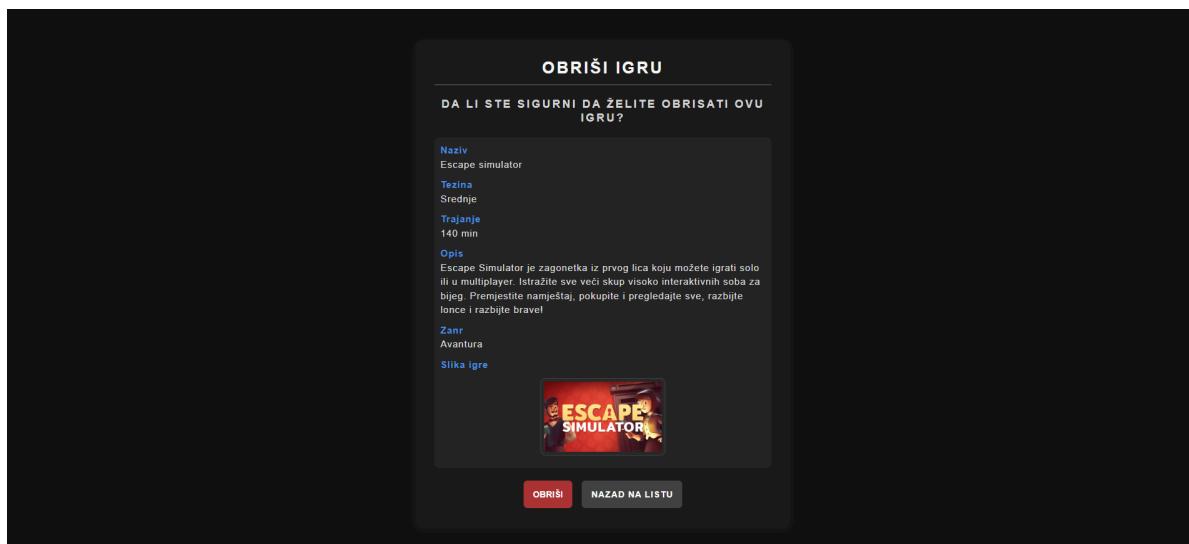


Slika 43: Admin/User - detalji igre

RAZVOJ INFORMACIONIH SISTEMA - PROJEKTNI ZADATAK



Slika 44: Admin - uredi igru



Slika 45: Admin - obriši igru

Slika 46: Admin - admin panel

RAZVOJ INFORMACIONIH SISTEMA - PROJEKTNI ZADATAK

The screenshot shows the 'REZERVACIJE' (Reservations) section of the admin panel. It displays a table with two rows of reservation data:

KORISNIK	IGRA	DATUM REZERVACIJE	STATUS	AKCIJE
nekaaa@mail.com	The Room	2/11/2025 3:14:26 PM	Confirmed	Uredi Detalji Obriši
adnicvarupa@gmail.com	The Room	2/11/2025 5:00:57 PM	Pending	Uredi Detalji Obriši

At the bottom of the page, there is a copyright notice: © 2025 - VirtualRealityEscapeRooms - [Privacy](#).

Slika 47: Admin - pregled rezervacija

The screenshot shows the 'STATISTIKA' (Statistics) section of the admin panel. It displays a table with five rows of statistical data:

STATISTIKA	VRIJEDNOST
Ukupno korisnika	7
Ukupno igara	7
Ukupno rezervacija	2
Ukupno prijava	0

At the bottom of the page, there is a copyright notice: © 2025 - VirtualRealityEscapeRooms - [Privacy](#).

Slika 48: Admin - pregled statistike

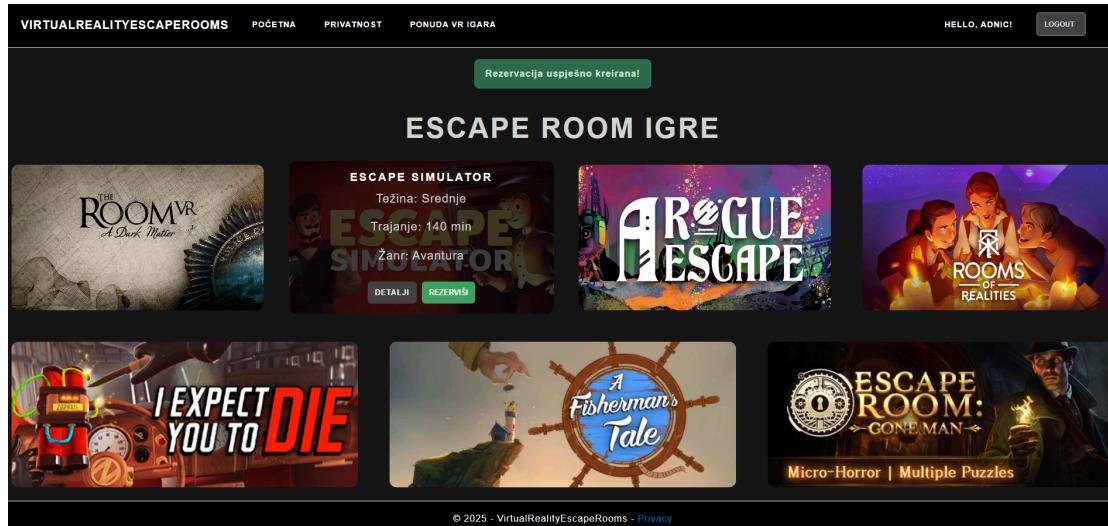
The screenshot shows the 'PRIJAVE KORISNIKA' (User Registrations) section of the admin panel. It displays a table with columns: RAZLOG PRIJAVE, STATUS, DATUM PRIJAVE, KORISNIK KOJI JE PRIJAVIO, PRIJAVLJENI KORISNIK, and AKCIJE.

RAZLOG PRIJAVE	STATUS	DATUM PRIJAVE	KORISNIK KOJI JE PRIJAVIO	PRIJAVLJENI KORISNIK	AKCIJE

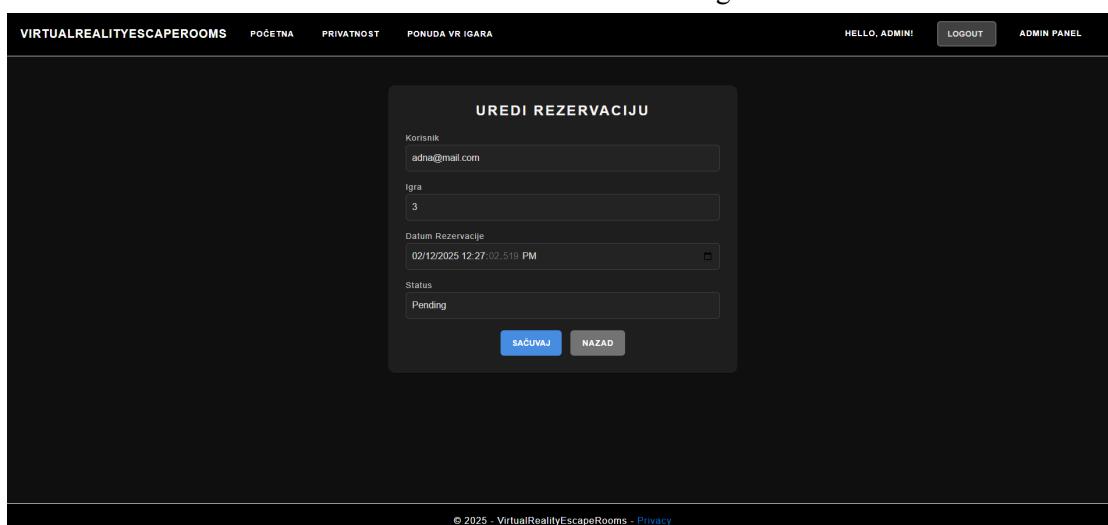
At the bottom of the page, there is a copyright notice: © 2025 - VirtualRealityEscapeRooms - [Privacy](#).

Slika 49: Admin - pregled prijava korisnika

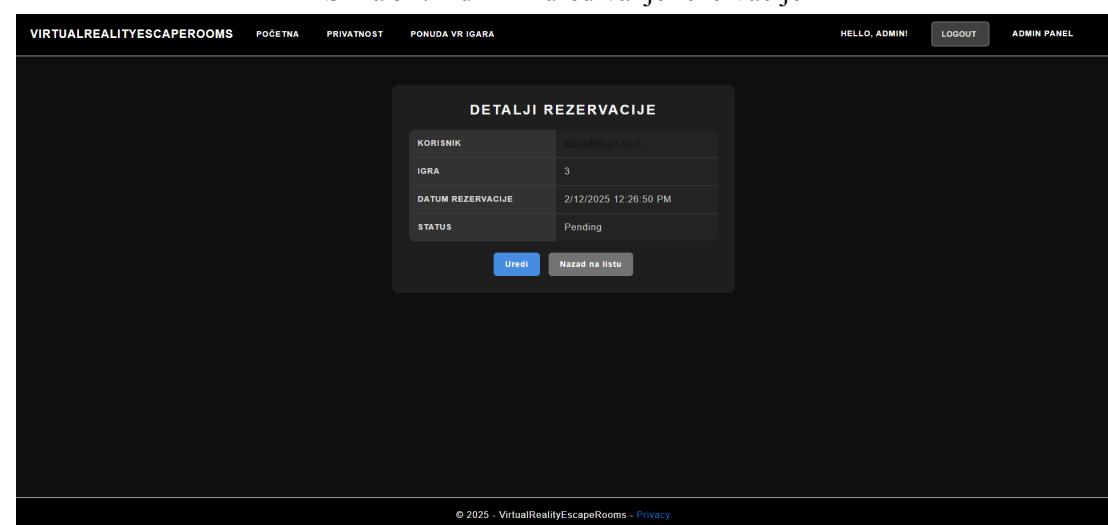
Kada je obični korisnik prijavljen, na hoveru igre ima dugme za rezervaciju i kada ga pritisne, podaci se spreme u bazu nakon čega administrator pregleda podatke i upravlja njima.



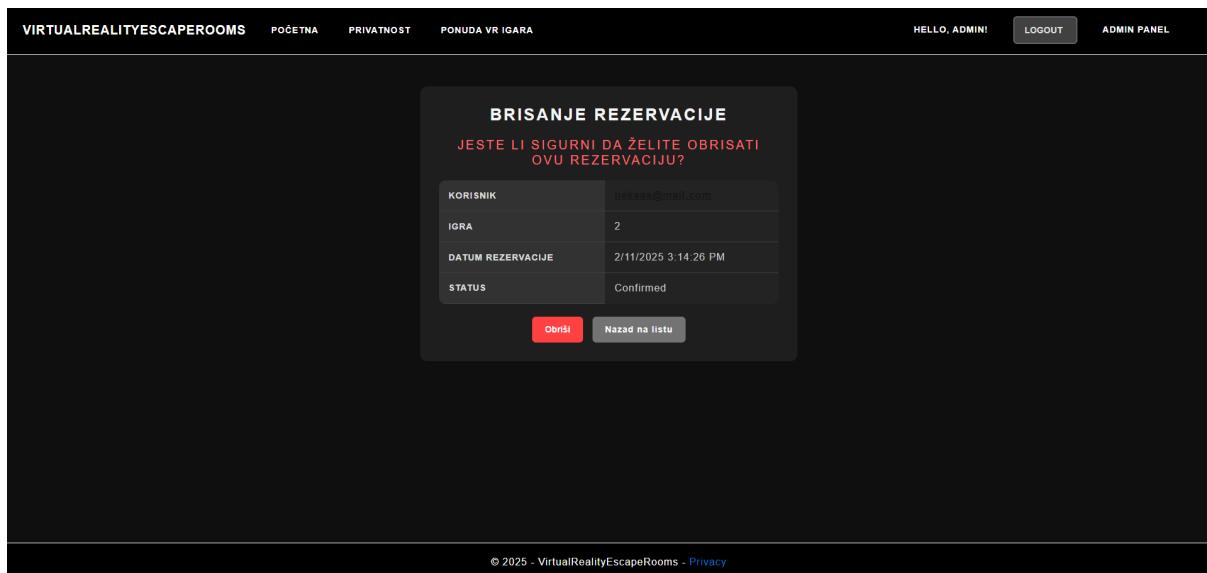
Slika 50: Korisnik rezerviše igru



Slika 51: Admin - uređivanje rezervacije



Slika 52: Admin - detalji rezervacije



Slika 53: Admin - brisanje rezervacije

SWAGGER DOKUMENTACIJA

Za naš projekat, Swagger dokumentacija nudi pregled svih API metoda. Za svaki model implementirani su osnovni HTTP zahtjevi sa dodatkom dodatnih po potrebi:

GET: Preuzimanje podataka o modelu.

POST: Kreiranje novih podataka u sistemu.

PUT: Ažuriranje postojećih podataka.

DELETE: Brisanje podataka iz sistema.

The screenshot shows the Swagger UI interface for the 'VirtualRealityEscapeRooms' API. At the top, there's a header with the Swagger logo and a dropdown menu 'Select a definition' set to 'VirtualRealityEscapeRooms v1'. Below the header, the API title 'VirtualRealityEscapeRooms' is shown along with its version '1.0' and 'OAS 3.0'. The URL 'https://localhost:7015/swagger/v1/swagger.json' is also listed.

The interface is organized into sections for different models:

- Igra** section:
 - Operations: GET /api/Igra, POST /api/Igra, GET /api/Igra/{id}, PUT /api/Igra/{id}, DELETE /api/Igra/{id}, GET /api/Igra/filteraj
- Korisnik** section:
 - Operations: GET /api/Korisnik, POST /api/Korisnik, GET /api/Korisnik/{id}, PUT /api/Korisnik/{id}, DELETE /api/Korisnik/{id}

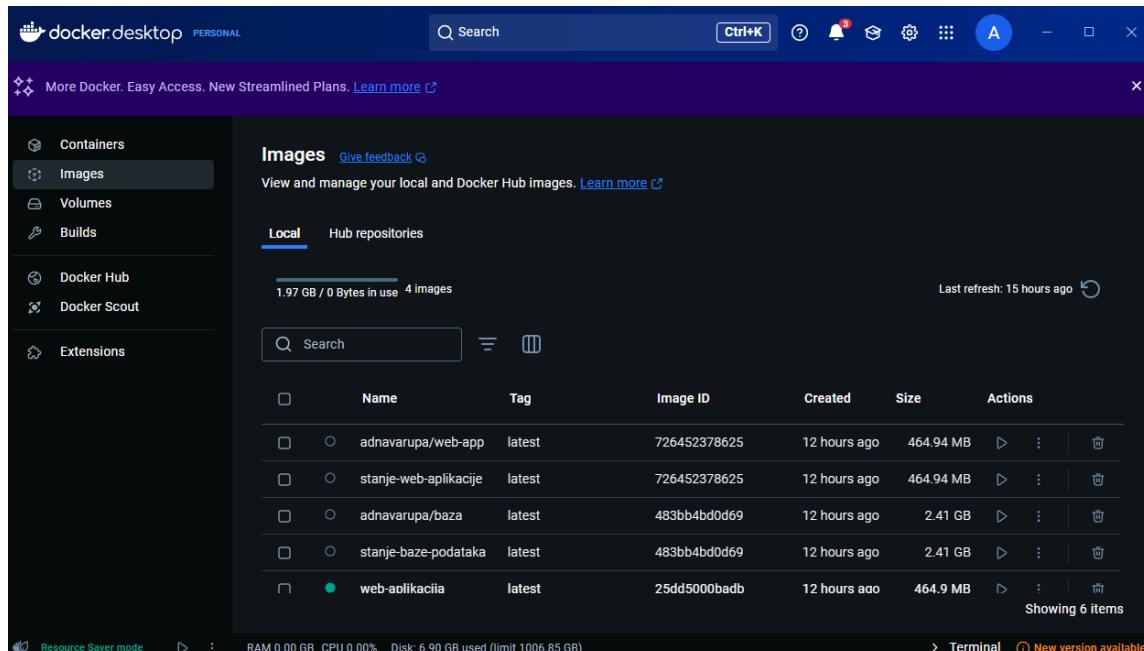
Slika 54: Swagger UI

DEPLOYMENT I DOKERIZACIJA

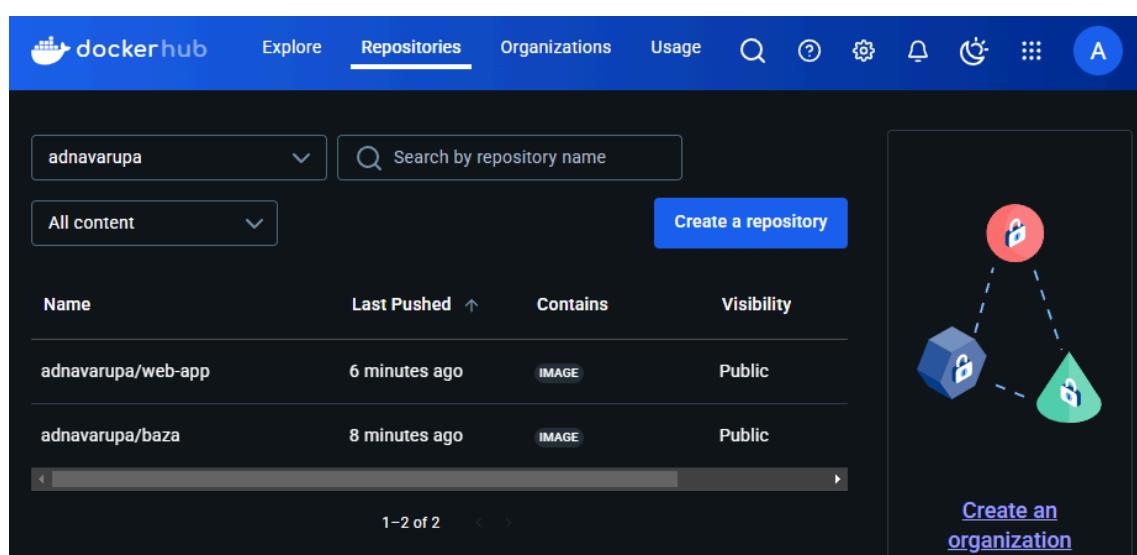
U ovom dijelu dokumentacije, opisujemo proces **dockerizacije** i **deploymenta** aplikacije na server putem **Docker** kontejnera i hosting platforme **SmarterASP**. Korišteni radi lakšeg upravljanja aplikacijom i jednostavnog pokretanja na različitim okruženjima.

DOCKERIZACIJA

Za potrebe razvoja i testiranja aplikacije, kreirane su Docker slike za bazu podataka i web aplikaciju. Ove slike omogućavaju dosljedno okruženje za razvoj, testiranje i implementaciju. Nakon kreiranja, Docker slike su pushane na Docker Hub radi lakšeg korištenja i dijeljenja s drugim članovima tima ili za potrebe implementacije na drugim sistemima. Slike su public na repozitorijima [web-app](#) i [baza](#).



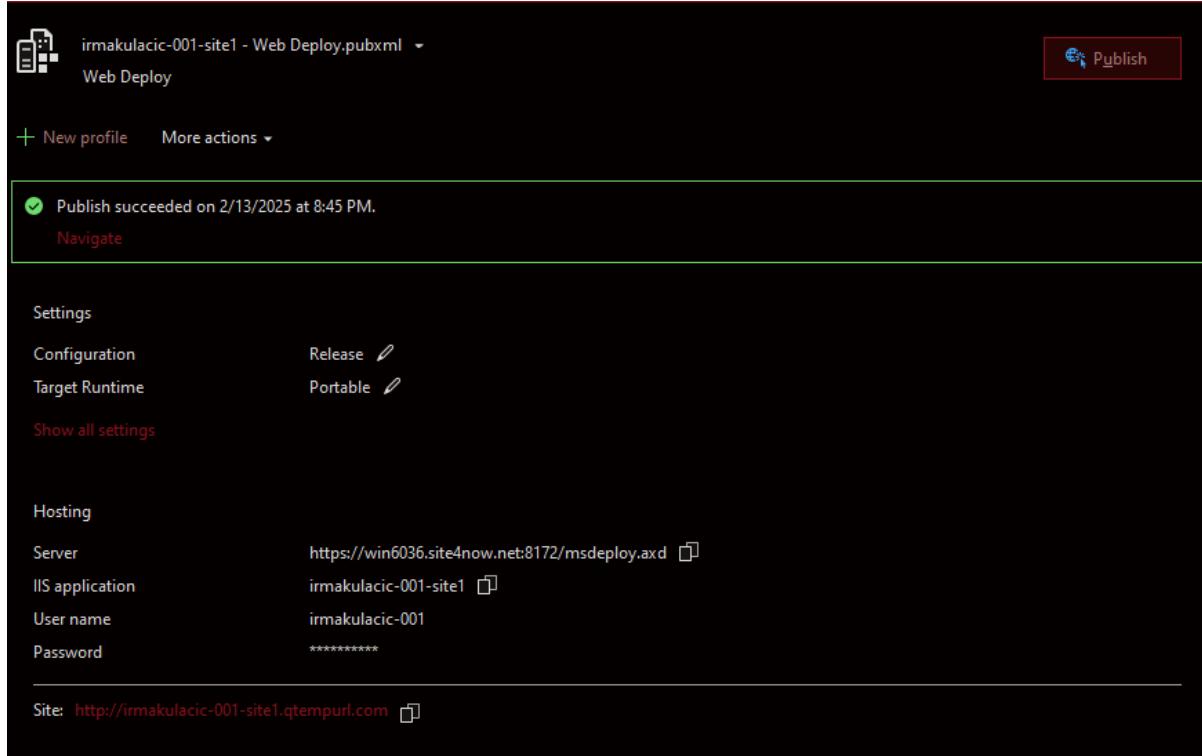
Slika 55: Kreirane slike u docker GUI



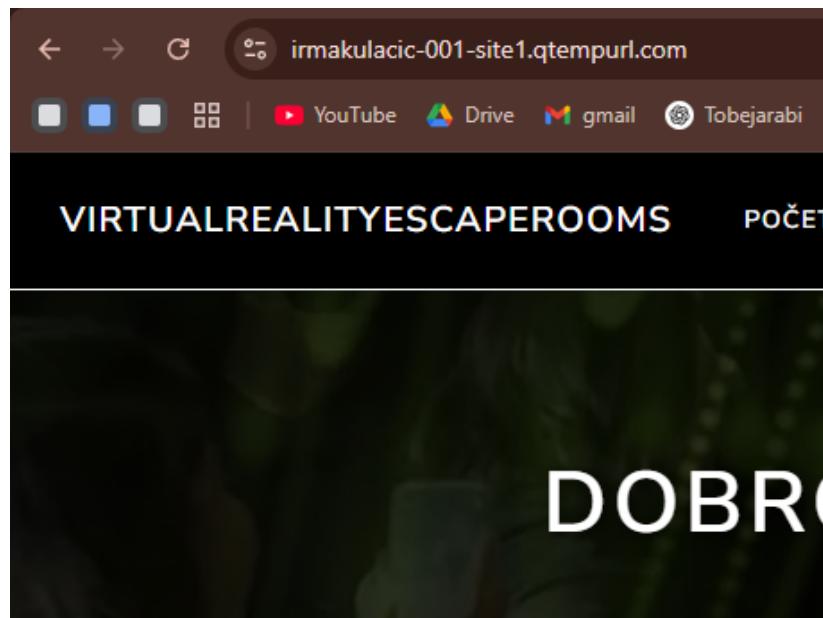
Slika 56: Uploadane slike na Docker hub

DEPLOYMENT

Koristile smo SmarterASP po uputstvu na vježbama za deployment. Nudi jednostavan i brz način za postavljanje aplikacije na server. Prvo smo uredile pubxml fajl kako bi svi podaci odgovarali za SmarterASP, a nakon toga publishali bez problema. Na slikama se može vidjeti rezultat publishanja u Visual Studiu i u browseru. Web app se sada može pristupiti



Slika 57: Rezultat deploymenta u Visual Studiu



Slika 58: Rezultat deploymenta u browseru