

Bioskop

**Predmet: Klijent Server Sistemi**

# Profesor: Student:

**dr Mirko Kosanović Mladen Ranđelović**

**Miloš Kosanović REr 51/17**

**Januar, 2021.**

**SADRŽAJ**

1. Uvod

2. Instalacija i podešavanje projekta

2.1. Instaliranje modula

3. Arhitektura aplikacije

3.1 Serverski deo

3.2 Klijentski deo

3.3 Baza

3.4 Komunikacija

4. Rad aplikacije

4.1 Opis implementacije

4.2 Opis funkcionalnosti – korisničko uputstvo

5. Literatura

**1. Uvod**

U ovom projektu obrađena je izrada veb aplikacije za rezervaciju mesta u bioskopu. Tehnologije koje su korišćene na klijentskoj strani su HTML5, CSS sa Bootstrap framework-om dok su na serverskoj strani korišćeni NodeJS sa Express framework- om Alati koji su korišćeni

prilikom izrade aplikacije su Google Chrome pretraživač̌ i Visual Studio Code kao text editor.

Aplikacija omogućava pregled filmova koji se trenutno prikazuju u bioskopu sa terminima njihovog prikazivanja, pregled arhive filmova(filmova koji se više ne prikazuju), pregled filmova koji će se prikazivati u bliskoj budućnosti.Takođe registrovani korisnici mogu da izvrše rezervaciju mesta u bioskopu za određeni film.Bioskopski radnici imaju mogućnost dodavanja novih filmova,sala i termina za prikaz filmova, kao i pregled rezervisanih mesta za određeni termin.Neregistrovani korisnici imaju mogućnost registracije popunjavanjem forme.

**2. Instalacija i podešavanje projekta**

Da bismo pokrenuli nodejs aplikaciju potrebno je da instaliramo NodeJs, a za ovaj projekat

koristimo i MySql bazu podataka , pa je potrebno i nju instalirati u ovom slučaju smo koristili XAMPP server.

**2.1. Instaliranje modula**

U ovom projektu nalazi se package.json fajl, to je fajl koji se inicijalizuje sa projektom, i ukoliko popunimo sve informacije ispravno biće kreiran fajl.Kako bi na nekom drugom računaru pokrenuli aplikaciju potrebno je da imamo insalirane zavisne module.Moduli se instaliraju kompandom npm install.U ovom projektu imamo sledeće module:

* Express
* Express-session
* MySql
* Nodemon
* Pug

**3. Arhitektura aplikacije**

Aplikacija sadrži direktorijum node\_modules koji sadrži module koji se koriste u aplikaciji.Direktorijum public koji sadrži javno dostupne fajlove.Direktorijum rute koji sadrži rute(putanje ka stranicama web-sajta).Rute su grupisani u fajlove prema njihovoj svrsi rute sa filmove se nalaze u posebnom fajlu,rute za termine u posebnom itd.Direktorijum views koji sadrži stranice kreirane u pug template enginu.Direktorijum views sadrži poddirektorijume u kojima su razvrstane stranice.

Poddirektorijum filmovi sadrži stranicu za dodavanje filma,za prikaz trenutnih,arhivranih i filmova u najavi,stranicu za dodavanje termina za određeni film.Poddirektorijum korisinici sadrži stranicu za prijavu i registraciju.Poddirektorijum sale sadrži stranicu za dodavanje bioskopske sale.Poddirektorijum stranice sadrži kontakt stranicu i stranicu koja prikazuje informacije o bioskopu.Poddirektorijum termini sadrži stranicu za prikaz današnjih termina i stranicu za dodavanje termina.Poddirektorijum rezervacije sadrži stranicu za prikaz lista rezeravacija za određenog korisnika,stranicu za prikaz informacija o određenoj rezervaciji(ukupnu cenu karata,koja su mesta rezervisana),stranicu za rezervaciju mesta za određeni termin,stranicu za prikaz svih rezervacija za određeni termin.

Direktorijum zahtevi sadrži SQL upite,upiti su grupisani po svrsi,posebno za filmove,posebno za rezervacije itd.

Rute,zahtevi i stranice su grupisane u posebnim poddirektorijumima radi preglednosti samog koda.

**3.1 Serverski deo**

Na serverskom delu je korišnjen node.js sa Express frameworkom.

Node.js je open-source platforma za izvršavanje serverskih aplikacija.Kod u Node.js se piše u Javascript programskom jeziku,Node.js omogućava upotrebu asinhronih funkcije.

Express framework je framwork za Node.JS aplikacije.On sadrži metode za rutiranje,prikaz statičkih datoteka itd.

Pug je template engine koji omogućava bržu izradu dinamičkih web stranica.

Takođe koristimo sledeće module:

Express-session omogućava čuvanje podataka o sesiji

MySql -omogućava komunikaciju sa bazom podataka‚

Nodemon- omogućava automatsko restartovanje servera prilikom svake promene

Pug- template engine koji omogućava brže kreiranje stranice takođe umogućava ubacivanje dinamičikog sadžaja u HTML kod,korišćenje foreach petlji,uslova,promenjivih.

**3.2 Klijentski deo**

Na klijentskom delu je korišćen HTML,CSS i Bootstrap framwork.Boostrap framework je dizajniran tako da podrži razvnoj dinamičkih web sajtova.On omogućava brži razvnoj web sajtova.

**3.3 Baza podataka**

Podaci se čuvaju u MySQL bazi podataka.MySQL je relaciona baza podataka.Podaci su smešteni u 6 tabela:

* filmovi – sadrži informacije o filmovima
* korisnici-sadrži informacije o registrovanim korisnicima
* rezervacije-sadrži informacije o rezervacijama
* rezervacije\_sedista-sadrži informacije o rezervisanimsedištima koja pripadaju određenoj rezervaciji
* sale-sadrži informacije o bioskopskim salama
* termini-sadrži informacije o terminima u kojima se prikazuju filmovi

Diagram

Description automatically generated

**Text

Description automatically generated with low confidence**

**3.4 Komunikacija**

Lista i opis ruta i formata podataka koji se razmenjuju.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Adresa rute | HTTP  metod | Parameteri | OPIS | PRAVO PRISTUPA | | |
| neprijavljen | prijavljen | admin |
| / | GET | / | Glavna strana.Vrši preusmeravanje korisnika na rutu /filmovi/trenutni | DA | DA | DA |
| /onama | GET | / | Prikazuje stranicu sa informacijiama o bioskopu. | DA | DA | DA |
| /registracija | GET | / | Prikazuje formu za registraciju korisnika. | DA | NE | NE |
| /registracija | POST | ime,prezime,email,lozinka,br\_telefona | Vrši dodavanje novog korisnika. | DA | NE | NE |
| /prijava | GET | / | Prikazuje formu za prijavu korisnika. | DA | NE | NE |
| /prijava | POST | email,lozinika | Vrši prijavu korisnika. | DA | NE | NE |
| /odjava | POST | / | Izvršava odjavu korisnika. | NE | DA | DA |
| /rezervacije/moje | GET | / | Prikazuje sve rezervacije za korisnika koji je prijavljen | NE | DA | DA |
| /rezervacije/obrisi/  :rezervacija\_id | POST | rezervacija\_id-integer | Briše određenu rezervaciju | NE | DA(samo vlasnik rezervacije) | DA |
| rezervacije/:rezervacija\_id | GET | rezervacija\_id-integer | Prikazuje informacije za određenu rezervaciju. | NE | DA(samo vlasnik rezervacije | DA |
| /rezervacije/termin/:termin\_id | GET | termin\_id-integer | Prikazuje sve rezervacije za određeni termin.Takođe omogućava korisniku kreiranje rezervacije za izabrani termin. | NE | DA | DA |
| /dodaj/:termin\_id |  | termin\_id-integer,  sediste(POST)-niz koji sadrži rezervisana mesta u formatu:  REDxSEDIŠTE | Omogućava rezervaciju mesta za određeni termin. | NE | DA | DA |
| /sale/dodaj | GET | / | Prikazuje formu za dodavanje nove sale za bioskop. | NE | NE | DA |
| /sale/dodaj | POST | naziv,broj\_redova,broj\_mesta\_u\_redu | Vrši dodavanje nove sale u bazu. | NE | NE | DA |
| filmovi/dodaj | GET | / | Prikazuje obrazac za dodavanje filma. | NE | NE | DA |
| filmovi/dodaj | POST | Naziv,izvorni\_naziv,trajanje,drzava,pocetak\_prikazivanja,kraj\_prikazivanja,slika,imdb\_link | Vrši dodavanje novog filma u bazu. | NE | NE | DA |
| filmovi/arhiva | GET | / | Prikazuje filmove koji se više neprikazuju. | DA | DA | DA |
| filmovi/najava | GET | / | Prikazuje filmove koji tek treba da se prikazuju. | DA | DA | DA |
| /filmovi/nearhivirani | GET | / | Prikazuje filmove koji se trenutno prikazuju ili koji će se tek prikazivati.Koristi se kada želimo da dodamo termin nekom filmu. | NE | NE | DA |
| filmovi/trenutni | GET | / | Prikazuje filmove koji se trenutno prikazuju u bioskopu. | DA | DA | DA |
| termini/dodaj/:film\_id | GET | film\_id-integer | Prikazuje formu za dodavanje novog termina za određeni film. | NE | NE | DA |
| termini/dodaj/:film\_id | POST | film\_id-integer | Dodaje termin za određeni film u bazu. | NE | NE | DA |
| /termini/obrisi/:id | POST | id-integer | Briše određeni termin iz baze. | NE | NE | DA |
| /termini/danasnji | GET | / | Prikazuje sve današenje projekcije filmova.Koristi se prilikom pregleda rezervisanih mesta od strane radnika u bioskopu. | NE | NE | DA |

**4. Rad aplikacija**

**4.1 Opis implementacije**

**4.1.1 Opis nekih SQL upita**

function query(sql,parametri)

{

    return new Promise((resolve)=>

    {

        app.pool.query(sql,parametri,(err,result)=>

        {

            //console.log(err);

            resolve(result)

        }

        )

    })

}

Ova funkcija se nalazi u fajlu zahtev.js njen zadatak je da na osnovu prosleđenog SQL querija i parametara za taj query izvrši upit.Ona koristi pool konekcija koji je kreiran u glavnom delu aplikacije(fajlu app.js),koristeći mysql modul(koji je dodatno instaliran i dodat kao dependency) ona komunicira sa bazom podataka.Pošto je funckija query(funkcija iz mysql modula) asinhorana onda moramo da vratimo Promise.Funkcija ne vraća vrednost izvršavanja neke funkcije pošto ta vrednost nije odmah dostupna jer se radi o asinhronoj funkciji,već vraća Promise tj. obećanje da će vrednost izvršenja funkcije biti dostupna čim se ta funckija izvrši.Na taj način program može da nastavi da izvršava neki drugi deo koda dok se čeka izvršenje funkcije.

async function prikazi\_za\_korisnika(korisnik\_id)

{

let sql="SELECT rezervacije.id,filmovi.naziv as naziv\_filma,rezervacije.sifra\_rezervacije,

sale.naziv as naziv\_sale,DATE\_FORMAT(termini.vreme\_prikazivanja, '%d.%m.%Y. %H:%i') as vreme\_prikazivanja,rezervacije.racun as cena\_karata,DATE\_FORMAT(rezervacije.vreme\_rezervacije, '%d.%m.%Y. %H:%i')

as vreme\_rezervacije FROM rezervacije INNER JOIN termini ON termini.id=rezervacije.termin\_id

INNER JOIN filmovi ON filmovi.id=termini.film\_id INNER JOIN sale ON sale.id=termini.sala\_id WHERE reze rvacije.korisnik\_id=? ORDER BY vreme\_rezervacije DESC";

   return zahtev.query(sql,[korisnik\_id]);

}

Prikazuje sve rezervacije za korisnika sortirane po vremenu rezervacije od najnovije rezervacije do najstraji.Takođe je pored samih podataka o rezervaciji potrebno da prikažemo za koji film je ta rezervacija i za koji termin.Zbog toga vršimo “spajanje” (join) sa tebelom filmovi i tabelom termini.Pošto mysql vraća vreme u formatu koji nije pogodat za čitanje vrši se transformacija u drugi format koji je pogodniji( dan.mesec.godina. sat:minuti).Takođe vršimo bindovanje parametra korišnjenjem ? u SQL queriju i prosleđivanjem vrednosti za te parametre u posebnom nizu.Bindovanjem parametara izbegavamo da korisnik izvrši SQL injekciju tj. da izvrši dodavanje malicionoznog SQL koda u query.Na primer u ovom slučaju bi korisnik mogao da izvrši dodavanje kod dela koda WHERE rezervacije.korisnik\_id=1 OR 1=1 i na taj način prikaže rezervacije svih korisnika.Ovaj način bindovanja parametara je primenjen u svim SQL upitima.

async function prikazi\_rezervisana\_mesta(termin\_id)

{

    //let sql1="SELECT rezervacije\_sedista.\* FROM rezervacije,rezervacije\_sedista WHERE rezervacije.termin\_id=? AND rezervacije\_sedista.rezervacija\_id=rezervacije.id";//prvi nacin za join 2 tabele

    let sql="SELECT rezervacije\_sedista.\* FROM rezervacije INNER JOIN rezervacije\_sedista ON rezervacije\_sedista.rezervacija\_id=rezervacije.id WHERE rezervacije.termin\_id=? ";//2 nacin za join 2 tabele

   return zahtev.query(sql,[termin\_id]);

}

Ovaj sql upit prikazuje sva mesta koja su rezervisana za određeni termin.Ovaj podatak nam je bitan kako bi smo korisniku prilikom rezervacije označili i kako bi smo ga sprečili da rezerviše mesta koja su već zauzeta.Prvo vršimo selekciju svih rezervacija za određeni termin,a ona spajamo tu tabelu sa tabelom rezervacije\_sedista koja nam sadrži podatke o sedištima koja pripadaju određenoj rezervaciji.

**4.1.2 Opis implementacije sesije**

Sesije se koriste kako korisnik ne bi morao da unosi šifru svaki put kada bi promenio stranicu. Korisnik na stranici za prijavu unese svoj email i šifru,zatim server proveri da li korisnik sa tim podacima postoji. U slučaju da korisnik sa tim podacima postoji kreira se sesija. Sesija u slučaju ove aplikacije čuva podatke o ID-u(iz baze) korisnika i o tipu naloga.Ti podaci se čuvaju na serverskoj strani.Na korisničkoj strani se u kolačiću čuva ID sesije.Web browser prilikom svake posete nekoj stranici na sajtu šalje podatak o ID-u sesije.Na osnovu ID-a sesije server pronalazi podatke o sesiji,koji se zatim dalje mogu koristiti u aplikaciji.

**4.1.3 Opis načina čuvanja šifri u bazi podataka**

Šifre u bazi podataka nije bezbedno čuvati kao običan tekst.Haker u slučaju pristupa bazi može videti šifre svih korisnika.Zbog toga je te šifre potrebno na neki način obezbediti.Jedan od načina je putanjem hešovanja.Šifra se hešuje pomoću kriptografske funkcije za sažimanje(one-way funkcija).Npr. imamo šifru “sifra123“ kada šifru “provučemo” kroz heš funkciju SHA1 mi dobijamo 678d658eb40259754d7346b6d2eba579f677d17d.Međutim to nije dovoljno zaštićeno zato što ako više korisnika ima istu šifru u bazi će se nalaziti iste vrednosti hešovane šifre,tako da će haker ipak imati neke podatke koje može da koristi..Zbog toga se dodaje salt,nasumično generisan tekst,npr. sct4df7t9g tako da prilikom hešovanja prethodne lozinke mi dobijamo sledeći izlaz 6957ea677e4cb2afc462fd5e4a1ce30208a49126. Takođe haker može da već odradi hešovanje često korišćenih šifri (rainbow table) i da te rezultati koristi više puta time ubrzava proces probijanja šifri,dok u slučaju da je šifri dodat salt njegova tabela je beskorisna.Međutim ni to nije dovoljno bezbedno,zbog toga je potrebno da šifru hešujemo više puta.Šifra se u bazi čuva u sledećem formatu.

sha1$322e3e42$1$3c43cc283419345173bdcfa421022bb26f627b08

| | | |

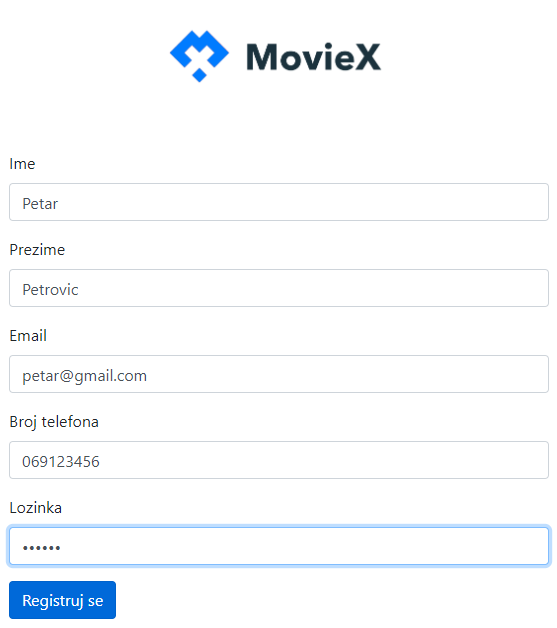
| | hešovana vrednost lozinke=3c43cc283419345173bdcfa421022bb26f627b08

| | broj iteracija=1

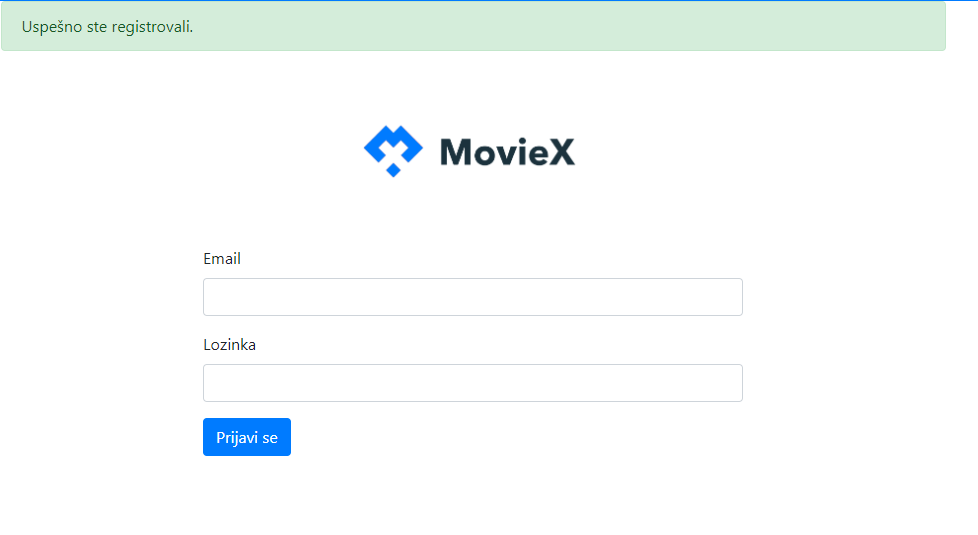
| salt = 322e3e42(8 karaktera)

naziv algoritma koji je korišćen za hešovanje = sha1

**4.2 Opis funkcionalnosti – korisničko uputstvo**



Forma za registraciju korisnik ispunjuje formu kako bi se registrovao na sajt.

Posle registracije korisnik može da se prijavi na svoj nalog.

Zavisno od toga da li je korisnik prijavljen ili ne i da li je admin.Postoje tri tipa navigacionog menija.

1. Navigacioni meni za neprijavljenog korisnika

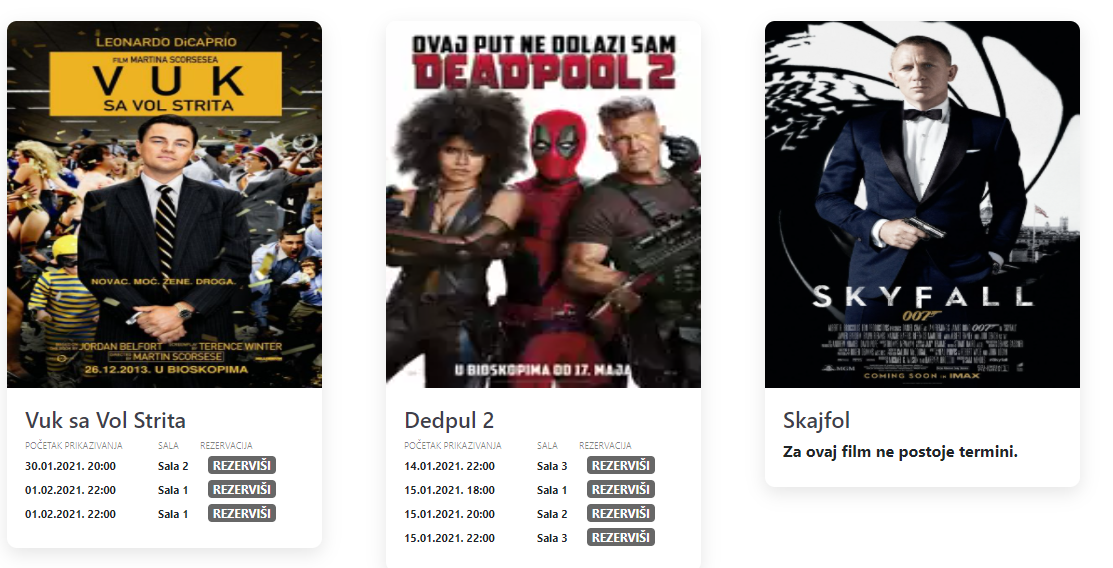


1. Navigacioni meni prijavljenog korisnika (običnog)

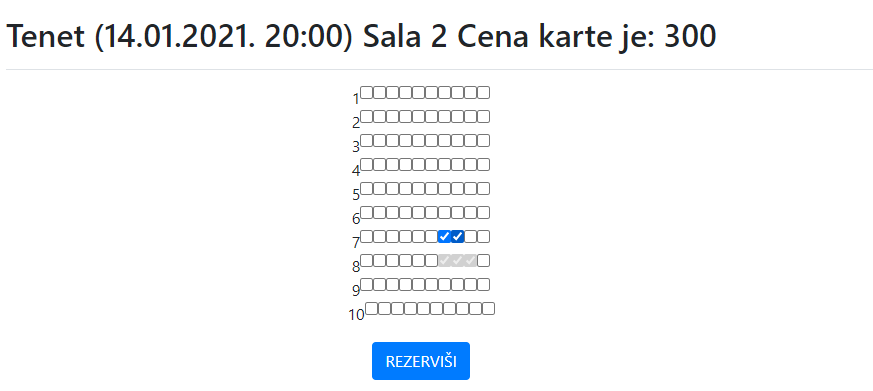


1. Navigacioni meni prijavljenog korisnika (admina)

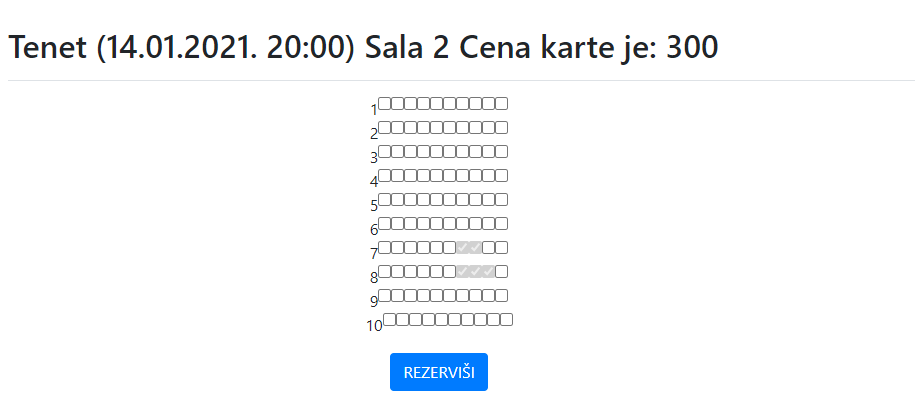




Lista filmova koji se trenutno prikazuju.Korisnik koji je prijavljen može da izvrši rezervaciju za izabrani termin.

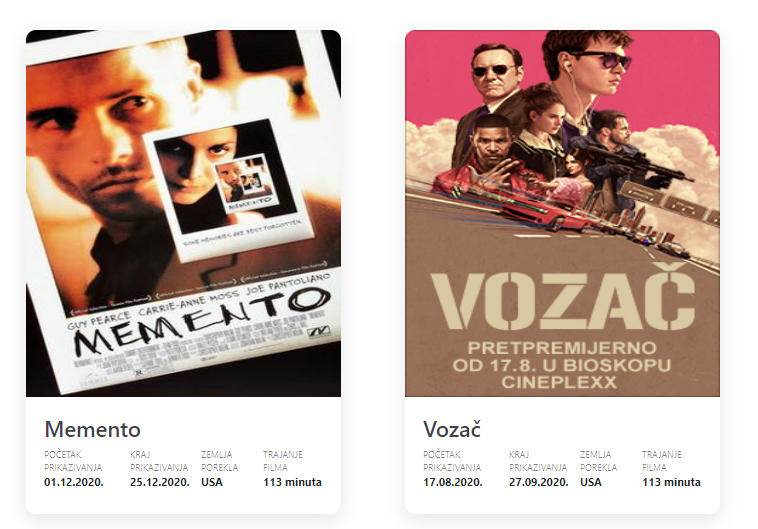


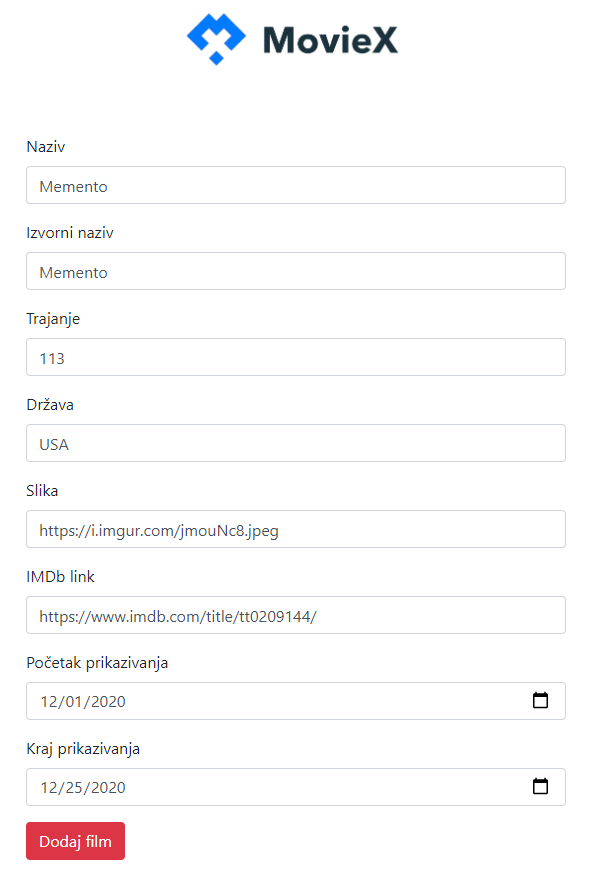
Korisniku se posle klika na dugme ''REZERVIŠI'' u listi filmova koji su trenutno na repertoaru prikazuje dijalog za rezervaciju mesta.Mesta koja su rezervisana su označena sivom bojom,a mesta koja je on odabrao plavom.

 Posle rezervacije ta mesta su prikazana kao rezervisana.   


Korisnik klikom na opciju “Moje rezervacije” u navigacionom meniju dobija hronološki prikaz svih svojih rezervacija od najnovije do najstarije.Broj rezervacije i šifru rezervacije koristi prilikom preuzimanja karata na blagajni bioskopa.

Klikom na dugme “Info” u pregledu rezervacija korisnik može da vidi više detalja o izabranoj rezervaciji.

Korisnik takođe ima mogučnost da vidi listu filmova koji se više ne prikazuju u bioskopu.



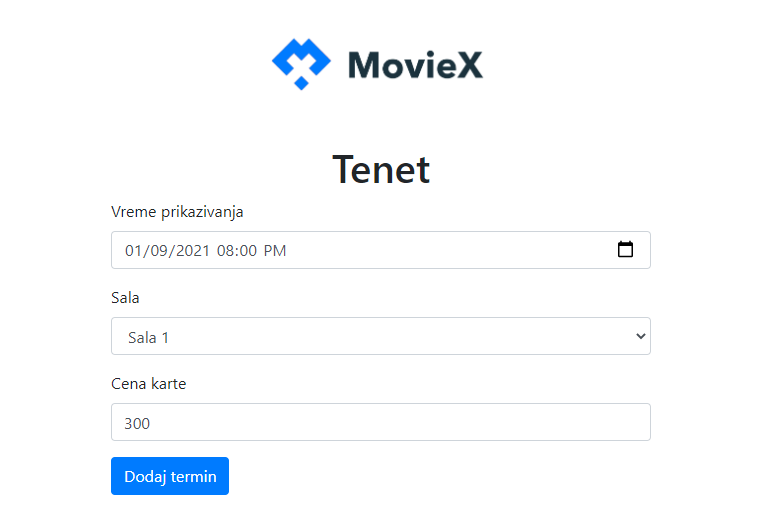
Forma za dodavanje filma.Samo admin može dodati film popunjavanjem podataka.

Početak i kraj prikazivanja su bitni kako bi zavisno od tih parametara film bio rasporđen u arhivu filmova,u filmove koji se trenutno prikazuju ili u listu filmova u najavi.

Admin ima mogućnost da doda novu bioskopsku salu.Od podataka upisuje naziv,broj redova i broj mesta u redu.

Admin ima mogućnost da doda termine za filmove koji

se trenutno prikazuju ili koji će se prikazivati.



Admin odabere film za koji želi da doda termin,zatim dobija formu u kojoj izabere salu u kojoj će se film prikazivati,termin i cenu karte.

5. Literatura

1.Klijent Server Sistemi - Predavanja, Mirko Kosanović

2.Klijent Server Sistemi - Praktikum, Miloš Kosanović

3. <http://expressjs.com/en/5x/api.html>

4. <https://www.w3schools.com/>

5. <https://stackoverflow.com/>

6. https://github.com/mysqljs/mysql