Kauno technologijos universitetas

Informatikos fakultetas

Informacinė filmų sekimo bei įspūdžių dalinimosi sistema

Baigiamasis bakalauro studijų projektas

Marius Arlauskas

Projekto autorius

lekt. Darius Matulis

Vadovas

Kaunas, 2020

Kauno technologijos universitetas

Informatikos fakultetas

Informacinė filmų sekimo bei įspūdžių dalinimosi sistema

Baigiamasis bakalauro studijų projektas

Programų sistemos (612I30002)

Marius Arlauskas

Projekto autorius

lekt. Darius Matulis

Vadovas

Dr. Mikas Binkis

Recenzentas

Kaunas, 2020

Kauno technologijos universitetas

Informatikos fakultetas

Marius Arlauskas

Informacinė filmų sekimo bei įspūdžių dalinimosi sistema

Akademinio sąžiningumo deklaracija

Patvirtinu, kad mano, **Mariaus Arlausko**, baigiamasis projektas tema „Informacinė filmų sekimo bei įspūdžių dalinimosi sistema“ yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

(vardą ir pavardę įrašyti ranka)	(parašas)

Arlauskas Marius. Informacinė filmų sekimo bei įspūdžių dalinimosi sistema. Bakalauro studijų baigiamasis projektas vadovas lekt. Darius Matulis; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatikos mokslai, Programų sistemos.

Reikšminiai žodžiai: API, filmų sekimas, socialinis tinklas.

Kaunas, 2020. 62 p.

Santrauka

Darbe pristatoma informacinė filmų sekimo sistema su dalimi socialinio tinklo elementų. Įvade apžvelgiamas darbo aktualumas, iškeliamas, darbu siekiamas pasiekti, tikslas bei uždaviniai. Aktualumo tema plėtojama analizės dalyje, kurioje taip pat nusakomas bendras sistemos veiklos tikslas ir nagrinėjami rinkoje esantys konkurentai.

Projektavimo dalyje pateikiami sistemos funkciniai ir nefunkciniai reikalavimai ir reikalavimai techninei įrangai. Vėliau taip pat apžvelgiamos technologijos ir metodikos, naudotos projekto projektavimo etape.

Testavimo etape pateikiamas testavimo planas bei pasiekti rezultatai.

Dokumentacijos dalyje pateikiamas sistemos naudotojo vadovas, diegimo ir priežiūros instrukcijos.

Darbo pabaigoje aprašomi sistemos kūrimo rezultatai ir išvados.

Arlauskas, Marius. Movie Tracking and Socialising System. Bachelor's Final Degree Project / supervisor lect. Darius Matulis; Informatics Faculty, Kaunas University of Technology.

Study field and area (study field group): Computer Sciences, Software Systems.

Keywords: API, movie tracking, socialising.

Kaunas, 2020. 62 p.

Summary

This thesis presents movie tracking information system with parts of social network. The introduction reviews the relevance of the work, raises the goal and objectives to achieve. The topic of relevance is developed in the part of the analysis, which also defines the general purpose of the system's operation and examines the competitors in the market.

System’s modelling section presents the functional and non-functional requirements of the system, system limitations and hardware requirements. After, technologies and methodologies used in the design phase of the project are reviewed.

In the testing section, the test plan, examples and achieved results are presented.

The documentation section provides a system user guide, installation and maintenance instructions.

At the end of the paper are described the results and conclusions of the system development.

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Santrumpų ir terminų sąrašas

Santrumpos:

- API (*angl. Application Programming Interface*) – aplikacijų programavimo sąsaja.
- HTTP (*angl. HyperText Transfer Protocol*) – užklausos – atsakymo protokolas, naudojamas duomenims perduoti pasauliniame tinkle.
- JWT (*angl. JSON Web Token*) – duomenų perdavimo standartas, naudojamas kuriant vartotojų prieigai skirtus žetonus.
- UML (*angl. Unified Modeling Language*) – Vieninga modeliavimo kalba.
- REST (*angl. Representational state transfer*) – API kūrimo architektūrinis stilius, kurio pagrindinis bruožas būsenos nesaugojantis veikimas.

Terminai:

- Aplikacijų programavimo sąsaja – Sąsaja, suteikiama programos, kad programuotojas galėtų naudotis jos teikiamu funkcionalumu.
- Vieninga modeliavimo kalba - modeliavimo ir specifikacijų kūrimo kalba, skirta objektiškai orientuotų programų dokumentams atvaizduoti.
- RESTful API – API atitinkantis REST architektūrinį stilių.

Įvadas

Bėgant metams susidomėjimas filmais vis auga, taip pat didėja ir noras pasidalinti savo įspūdžiais bei gauti rekomendacijų, apie panašius, o gal net ir visiškai niekuo nesusijusius filmus. Didžioji dalis filmų žiūrovų, kaip pavaizduota *statistika.com* statistikoje [1], juos žiūri namuose, todėl po filmo peržiūrėjimo neturi kur pasidalinti patirtais įspūdžiais, gauti rekomendacijų ar padiskutuoti su bendraminčiais neieškant tam specialiai skirtų grupių įvairiose svetainėse, o netgi ir radus grupę, vėliau reikia ieškoti kitos svetainės, kurios pagalba būtų galima rasti informacijos apie pasiūlytus filmus. Šias bėdas išspręsti ir siekia šiuo darbu kuriama filmų sekimo bei įspūdžių dalinimosi sistema.

Darbo tikslas – sukurti lengvą būdą pasidalinti peržiūrėtų filmų sukeltais įspūdžiais, bendraujant gauti ar pačiam pateikti pasiūlymų vėliau planuojamiems filmams bei greitai rasti filmų informaciją toje pačioje vietoje. Keliami uždaviniai:

1. Ištirti panašias rinkoje esančias sistemas ir konkurentus.
2. Sudaryti kuriamos sistemos projektą.
3. Realizuoti tokią sistemos dalį, kuri leistų jei atlikti savo pagrindines funkcijas, kadangi pilnos sistemos realizacijai reikėtų skirti daug daugiau laiko.
4. Ištestuoti realizuotą sistemą.
5. Paruošti sistemos dokumentaciją.

Pagrindiniai darbui atlikti skiriami etapai: analizė, projektavimas, testavimas ir dokumentacija. Analizės skyriuje apibrėžiamas projekto aktualumas, išanalizuojamos panašios rinkoje egzistuojančios sistemos - galimi konkurentai, nusistatomas tikslas bei išanalizuojami jam pasiekti užsibrėžti uždaviniai. Projektavimo skyriuje pristatoma projekto sudėtis, nusakomi sistemos funkciniai ir nefunkciniai reikalavimai, panaudotos metodikos bei technologijos. Testavimo skyriuje nusakomas testavimo planas, testavimo pavyzdžiai bei pasiekti rezultatai. Dokumentacijos skyrius susideda iš sukurto internetinio puslapio naudotojo vadovo, diegimo bei priežiūros dokumentacijų.

Dokumento pabaigoje pateikiamos darbą bei rezultatus apibendrinančios išvados.

1. Analizė

1.1. Techninis pasiūlymas

1.1.1. Sistemos apibrėžimas

Informacinė filmų sekimo bei įspūdžių dalinimosi sistema – tai internetinis puslapis, leidžiantis filmais besidomintiems žmonėms lengvai bei greitai išsisaugoti filmus, bendraujant su kitais dalintis savo patirtais įspūdžiais juos žiūrint bei rasti svarbiausią informaciją apie filmus toje pačioje vietoje.

1.1.2. Bendras veiklos tikslas

Filmų sekimo bei įspūdžių dalinimosi sistemos tikslas efektyviai ir lengvai leisti žmonėms išsisaugoti dominančius filmus ir bendraujant pasidalinti patirtais įspūdžiais vienoje vietoje taip sutaupant laiko bei supaprastinant dabartinį procesą.

1.1.3. Sistemos pagrįstumas

Didžioji dalis filmų žiūrovų , kaip 2018 metų statistikoje teigta *statistika.com* [1], juos žiūrėjo namuose (šis žmonių polinkis dabar yra dar daugiau išaugęs dėl *COVID-19* pandemijos, tai taip pat buvo tirta *statistika.com* [2]). Po filmo peržiūros žmonės neturi kur pasidalinti patirtais įspūdžiais, gauti rekomendacijų ar padiskutuoti su bendraminčiais neieškant tam specialiai skirtų grupių įvairiose svetainėse, o netgi ir radus grupę, vėliau ieško kitos svetainės, kurios pagalba būtų galima rasti informacijos apie pasiūlytus filmus. Šis projektas siekia supaprastinti šį procesą ir sutaupyti žmonių laiko bei suteikti patogesnį bendravimo būdą sukuriant tokią svetainę, kuri leistų žmonėms viską, įskaitant naujų filmų paiešką, jų informacijos peržiūrą, įspūdžių ar rekomendacijų dalinimąsi bendraujant, įvairių statistikų peržiūras ir dar daugiau, atlikti vienoje vietoje vietoj bereikšmiškai gaištant laiką keliaujant tarp skirtingų svetainių ar ieškant ankščiau minėtų grupių.

1.1.4. Konkurencija rinkoje

Kuriamas projekto idėja yra sudarytas iš dviejų tipų sistemų: **filmų sekimo** ir **socialinio tinklo**, bet sistemos pagrindinis tikslas yra sutelktas į filmų sekimą, todėl konkurentais laikomos tik filmų sekimo ar panašaus tipo svetainės.

Šiuo metu labiausiai populiarios ir į kuriamą sistemą panašios internetinės svetainės, galinčios būti laikomos konkurentais, yra *IMDb*, *iCheckMovies*, *Filmsomniac*. Šios svetainės kaip ir kuriamas produktas siekia padėti žmonėms išsisaugoti ar atrasti naujus filmus, TV serialus. Toliau trumpai apie kiekvieną iš šių svetainių

IMDb – svetainė save apibūdina kaip „populiariausias ir autoritetingiausias pasaulyje filmų, TV ir įžymybių turinio šaltinis, skirtas padėti gerbėjams tyrinėti filmų ir laidų pasaulį bei nuspręsti, ką žiūrėti“ [3].

iCheckMovies – svetainė siūlo mažiau informacijos nei *IMDb* ir dėmesį skiria savo rezultatų pasidalinimui. Toks svetainės tikslas ir aprašoma pačioje svetainėje:

1. „Create your profile“ [4];
2. „Watch lots of movies“ [4];
3. „Brag about it“ [4].

Filmsomniac – filmų sekimo svetainė savo funkcionalumu nesmarkiai tesiskirianti nuo *IMDb* [5].

Pagrindinis kuriamos sistemos privalumas yra didelis socialinio tinklo elementų panaudojimas, leidžiantis sistemos naudotojams pajausti nuolatinio bendravimo galimybes. Šis svetainės funkcionalumas ir siekia ją išskirti iš konkurentų.

Panašiausiais socialinio tinklo elementais konkurentų svetainėse galima laikyti:

- Visų išvardintų konkurentų svetainėse esanti galimybė peržiūrėti kitų vartotojų filmų sąrašus;
- *IMDbPro* – *IMDb* Pro versija leidžia sekti žmones bei jų veiklas;
- *iCheckMovies* galimybė nusiųsti vartotojui pakvietimą draugauti ir pridėti jį į draugų sąrašą. Taip pat rašyti asmenines žinutes kitiems vartotojams;
- *Filmsomniac* - forumas diskusijoms.

Kuriamos sistemos galutinė versija siekia realizuoti visus ką tik paminėtus konkurentų elementus bei dar daugiau, tačiau ataskaitos kūrimo metu sistemoje visko įgyvendinti nepavyko.

Kiti skirtumai atvaizduojami 1.1 lentelėje:

- Galimybė pasitobulinti į Pro versiją. *IMDb* riboja dalį svetainės funkcionalumo padarydama jį mokamą.
- Šiuo metu kuriamoje sistemoje realizuotas vartotojų žinučių paskelbimas, kuris gali būti atliekamas bendroje naujienų juostoje arba kaip filmo peržiūra filmų puslapiuose, tačiau kaip ir buvo minėta planuojama išsiskirti ir pridėti daug daugiau funkcionalumo šioje srityje.
- *IMDb* ir *Filmsomniac* leidžia kurti daugybę skirtingų filmų sąrašų ir taip pat kaip *iCheckMovies* suteikia galimybę pasižymėti filmus kaip planuojamus arba peržiūrėtus, tačiau *iCheckMovies* nesuteikia galimybės kurti filmų sąrašų. Kuriamoje sistemoje leidžiamas vienas filmų sąrašas, tačiau filmams gali būti priskiriami skirtingi statusai.
- *IMDb* ir *Filmsomniac* svetainėje teikia informaciją ir apie filmus ir TV serialus o iCheckMovies taip pat kaip ir kuriama sistema teikia informaciją tik apie filmus, tačiau ateityje sistemoje planuojama pridėti ir TV serialus.
- Nė vienas konkurentas neturi tokio funkcionalumo kaip asmeninių filmų rekomendacijų, siūlančių filmus kiekvienam pagal jų turimą filmų sąrašą. Kuriamoje sistemoje šis funkcionalumas taip pat kol kas nebuvo padarytas, tačiau jis planuojamas artimiausioje ateityje.

1.1 lentelė. Konkurentų apžvalga

Lyginimo kriterijai	**IMDb**	**iCheckMovies**	**Filmsomniac**	**Kuriama sistema**
Svetainės Pro versija	$19.99 per mėnesį	-	-	-
Socialinio tinklo	Vartotojų sekimas	Draugų sąrašai ir	Forumas	Naujienų juosta bei

Lyginimo kriterijai	IMDb	iCheckMovies	Filmsomniac	Kuriama sistema
galimybės	(Tik Pro versijoje)	asmeninės žinutės		žinučių skelbimai, tačiau planuojama daug daugiau
Filmų sąrašų kūrimas	Galima kurti keletą sąrašų	Galima tik pažymėti filmą kaip planuojamą arbą peržiūrėtą	Galima kurti keletą sąrašų	Vienas sąrašas bet leidžiama filmams priskirti skirtingus statusus
Filmai ar TV serialai	Filmai ir TV serialai	Filmai	Filmai ir TV serialai	Filmai, bet ateityje planuojami ir TV serialai
Asmeninės filmų rekomendacijos	-	-	-	Planuojama ateityje

1.1.5. Prototipai ir pagalbinė informacija

Kadangi pati projekto idėja kilo iš šiuo metu egzistuojančių sistemų bei ryšio tarp jų trūkumo, tai projekto tikslas, reikalingi funkciniai bei nefunkciniai reikalavimai, išsikelti uždaviniai ir buvo kuriami atsižvelgiant į dabar egzistuojančių, panašių svetainių bei konkurentų teikiamas paslaugas bei jų reikalingumą., tačiau pati sistema – ir vartotojo sąsaja ir API buvo kurti nenaudojant jokių prototipų.

1.1.6. Ištekliai, reikalingi sistemai sukurti

Sistema sudaryta iš dviejų pagrindinių dalių: API ir vartotojo sąsajos. Kadangi tokio tipo svetainės, dėl didelio laiko sunaudojimo žiūrint į pačia svetainę, reikalauja geresnio dizaino, kuris sudomintų vartotoją, tai atvaizdavimo dalies kūrimui buvo skirta kone daugiau laiko nei jos API. Projektas buvo atliktas vieno žmogaus ir jam buvo skirta apie 400 darbo valandų. Atsižvelgiant į tai, kad projektą dar planuojama smarkiai plėsti jam dar bus skiriama tiek pat ir net daugiau darbo valandų.

1.2. Galimybių analizė

1.2.1. Techninės galimybės

Sistema buvo kuriama naudojant rinkoje paplitusias ir patikrintas technologijas: *Symfony 4, VueJs* karkasus bei įvairias jų bibliotekas. Kadangi šie karkasai yra plačiai išplitę - populiarūs bei daug naudojami, problemų ieškant informacijos ar pagalbos kuriant projektą nebuvo.

1.2.2. Vartotojų pasiruošimo analizė

Projektas buvo kurtas siekiant paprasto ir efektyvaus svetainės naudojimosi, todėl ja naudotis nesudėtinga ir tai galėtų atlikti bet koks, naršykle sugebantis naudotis ir prieigą prie interneto turintis vartotojas.

2. Projektas

2.1. Reikalavimų specifikacija

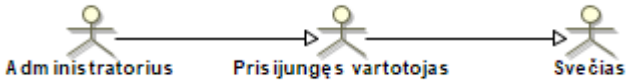
2.1.1. Komercinė specifikacija

Šis projektas yra universitetinis, bakalauro baigiamajam darbui skirtas projektas ir realaus užsakovo neturėjo. Projektą kūriau aš - Mariaus Arlauskas, o naudotojai – visi, filmais besidomintys žmonės, kurie norėtų turėti lengvą būdą pasidalinti savo įspūdžiais ar / ir susidaryti dominančių ar jau peržiūrėtų filmų sąrašą.

Kadangi projektas buvo asmeninis ir kurtas bakalauro baigiamajam darbui jokio biudžeto jis neturėjo, o numatyta baigimo data – 2020 metų gegužės 17 diena.

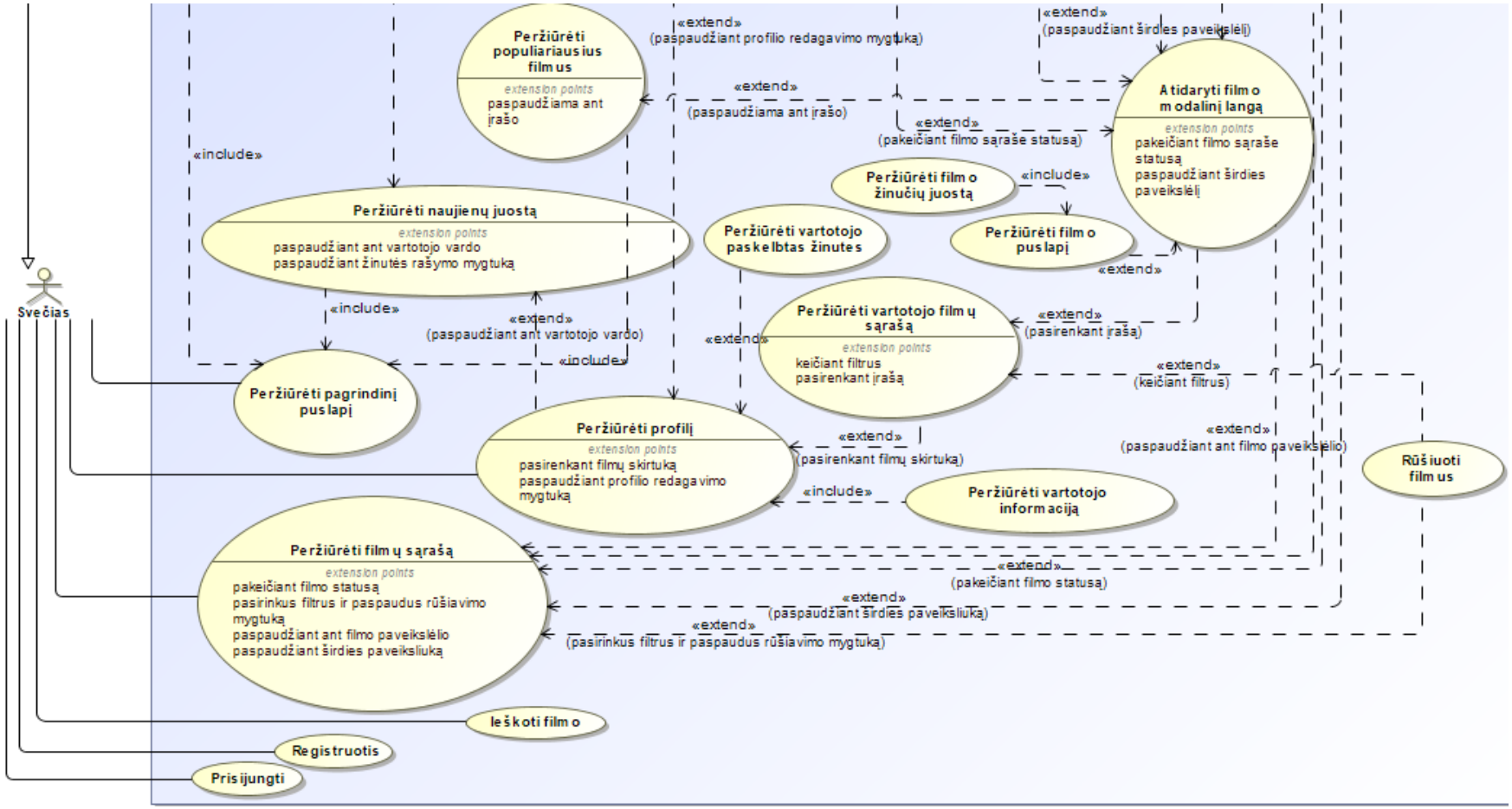
2.1.2. Sistemos funkcijos

Sistemos funkciniai reikalavimai atvaizduojami *UML* panaudojimo atvejų diagramomis, dėl didesnio aiškumo jas skaidant pagal vartotojų tipus (2.1 pav.).



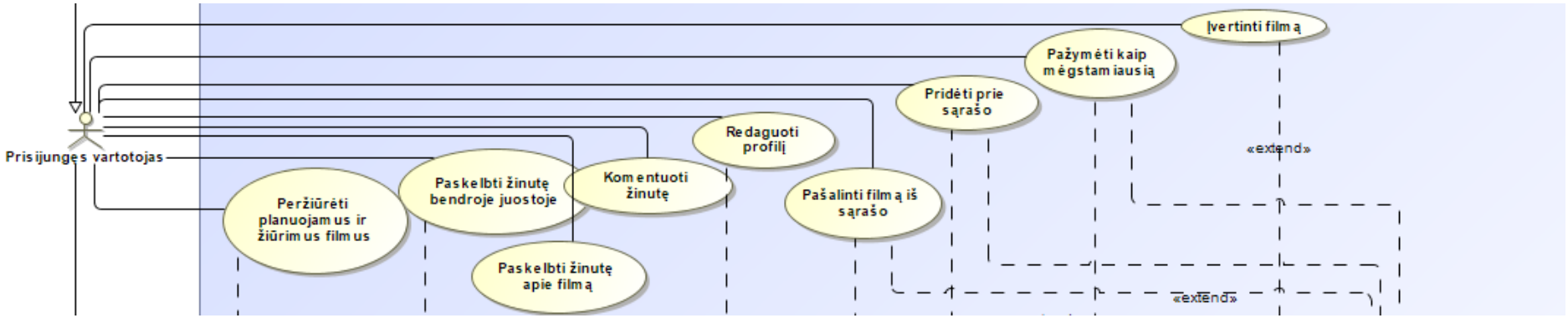
2.1 pav. Vartotojų tipai

Sistemoje egzistuoja trijų rūšių vartotojai, kur kiekvienas vartotojas turi tik sau galimus atlikti veiksmus. Žemiausiame lygyje yra *Svečias* – tai prie sistemos neprisijungęs vartotojas, kuris gali naudotis tik pagrindiniais sistemos teikiamais privalumais. Aukščiau yra *Prisijungęs vartotojas* – registruotas sistemoje bei prie jos prisijungęs vartotojas, kuris gali atlikti svečio veiksmus ir dar papildomus, prisijungusiam vartotojui galimus veiksmus. Visas sistemos galimybes pilnai gali išnaudoti *Administratorius* – tai sistemos savininkas, kuris gali atlikti visus prisijungusio vartotojo veiksmus bei papildomai juos valdyti.



2.2 pav. Neregistruoto vartotojo - *Svečio* panaudojimo atvejai

Svečias, tai neprisijungęs ir sistemoje neregistruotas vartotojas, jo panaudojimo atvejai atvaizduojami 2.2 lentelėje. *Svečias* gali registruotis arba prisijungti, peržiūrėti filmų sąrašą, kuriame yra galimybė atsirūšiuoti filmus pagal įvairius filtrus ar paspaudus ant filmo paveikslėlio atsidaryti modalinį langą, leidžiantį peržiūrėti filmo duomenis šiek tiek plačiau. Taip pat modalinis langas leidžia vartotojams peršokti į pagrindinį filmo puslapį ir pamatyti žmonių nuomones apie filmą. *Svečias* taip pat gali peržiūrėti kitų vartotojų profilius, kuriuose gali matyti vartotojų paskelbtas žinutes ir jų, taip pat galimus atsirūšiuoti, filmų sąrašus, bei peržiūrėti pagrindinį svetainės puslapį, suskaidytą į tris dalis: naujienų juostą, populiariausių filmų juostą ir, tik prisijungusiems vartotojams matomą, planuojamų bei šiuo metu žiūrimų filmų juostą.



2.3 pav. Prisijungusio vartotojo panaudojimo atvejai

Prisijungęs vartotojas – sistemoje registruotas ir prisijungęs vartotojas. *Prisijungęs vartotojas* gali atlikti svečio veiksmus ir taip pat pagrindiniame puslapyje matyti savo planuojamus bei šiuo metu žiūrimus filmus, paskelbti žinutę bendroje naujienų juostoje ar filmo puslapyje, redaguoti savo profilį, pašalinti ar pridėti filmą prie savo sąrašo, pažymėti filmus mėgstamiausiais bei juos įvertinti (2.3 pav.).

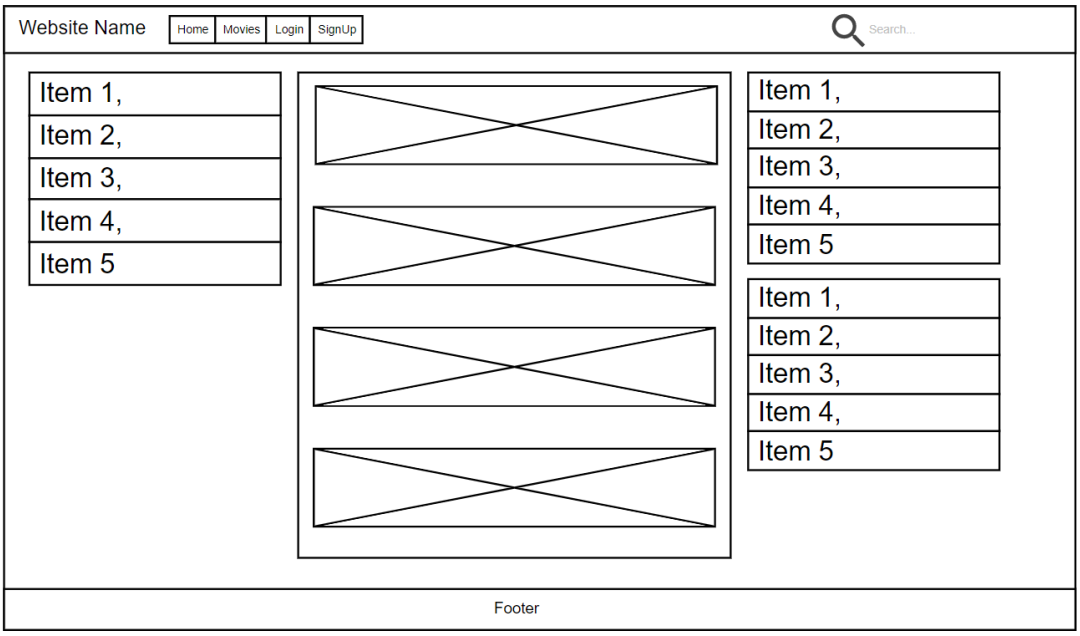
2.4 pav. pateikta *administratoriaus* panaudojimo atvejų diagramos dalį

2.4 pav. Sistemos administratoriaus panaudojimo atvejai

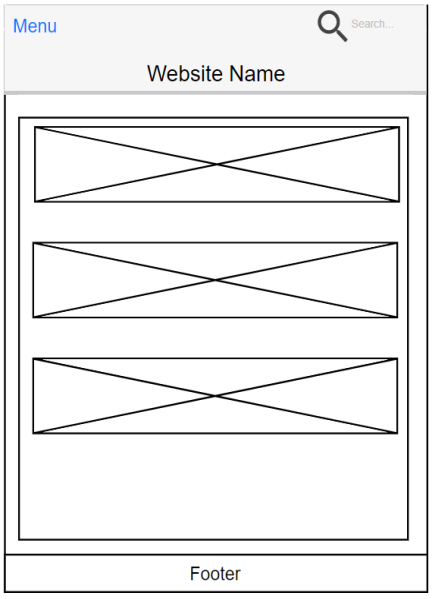
Administratorius – sistemos valdytojas, turintis visišką priėjimą prie visų jos teikiamų privalumų. *Administratorius* gali atlikti visus kitų vartotojų veiksmus ir taip pat valdyti sistemoje registruotus vartotojus jiems atimdamas ar grąžindamas galimybę rašyti žinutes (2.4 pav.).

2.1.3. Vartotojo sąsajos specifikacija

Kuriant vartotojo sąsają, jai naudojant svetainę moqups.com [6] buvo kuriami langų maketai, pagal kuriuos ir buvo formuojama visa sistema. Kadangi sistemos reikalavimai apima langų reaktyvumą, maketai taip pat buvo daromi dviem langų dydžiams. Paveikslėlyje 2.5 atvaizduotas pagrindinio puslapio maketas, o paveikslėlyje 2.6 jo telefonams skirta versija.



2.5 pav. Pagrindinio puslapio maketas kompiuteriui skirtam ekranui



2.6 pav. Pagrindinio puslapio maketas telefonui skirtam ekranui

2.1.4. Realizacijai keliami reikalavimai

Sistemai iškelti nefunkciniai reikalavimai:

1. Bendri projekto kūrimo reikalavimai:
    a. Projekto kodo saugojimui naudojama *Git* [7] versijų kontrolės sistema;
2. API reikalavimai:

    a. Kuriamas API privalo būti RESTful;
    b. Turi būti naudojama JWT autentifikacija;
3. Vartotojo sąsajos reikalavimai:
    a. Lengvai suprantama ir efektyviai leidžianti atlikti visas funkcijas;
    b. Parengta anglų kalba;
    c. Dizainas turi būti reaktyvus.

2.1.5. Techninė specifikacija

Projektas padalintas į dvi dalis: **API** ir **vartotojo sąsają**. Vartotojo sąsaja daryta su *VueJs* ir neturi jokių papildomų reikalavimų serveriui, tačiau jam reikalingų failų sukūrimui iškyla pora reikalavimų:

- *Node.js* ir *npm* instaliacijos.

API darytas su *Symfony 4* karkasu, kuris reikalauja:

- *PHP 7.1.3* arba didesnės versijos;
- Įrašytų plėtinių *Ctype, iconv, JSON, PCRE, Session, SimpleXML, Tokenizer*, kurie yra didžiojoje dalyje PHP 7;
- Įrašyto *composer* [8].

2.2. Projektavimo metodai

2.2.1. Projektavimo valdymas ir eiga

Kuriant projektą buvo naudotas iteracinis projekto kūrimo modelis. Iteracijos buvo skirstomos sistemos puslapiais, kur kiekviena iteracija apėmė skirtingą sistemos puslapį. Iš viso buvo atliktos 9 iteracijos ir kiekviena iteracija susidėjo iš puslapio projektavimo, realizavimo bei testavimo.

Atliktos iteracijos / sukurti svetainės puslapiai:

1. Visuose puslapiuose skirtos navigacijos juostos (poraštės juosta, antraštės juosta, vartotojo profilio juosta);
2. Prisijungimo puslapis;
3. Registracijos puslapis;
4. Pagrindinis svetainės puslapis;
5. Vartotojo profilio puslapis;
6. Vartotojo nustatymų puslapis;
7. Filmų sąrašo puslapis;
8. Filmo informacijos puslapis;
9. Administratoriaus nustatymų puslapis.

2.2.2. Projektavimo technologija

Sistemos projektas buvo sukurtas naudojant *UML* grafinius elementus. Projektas susideda iš sistemos duomenų bazės, panaudojimo atvejų ir jų veiklų diagramų, kur kiekviena diagrama turi savo aprašymą, nusakantį jos veikimo principą sistemoje. Visos diagramos buvo kurtos naudojant projektavimo įrankį *MagicDraw*.

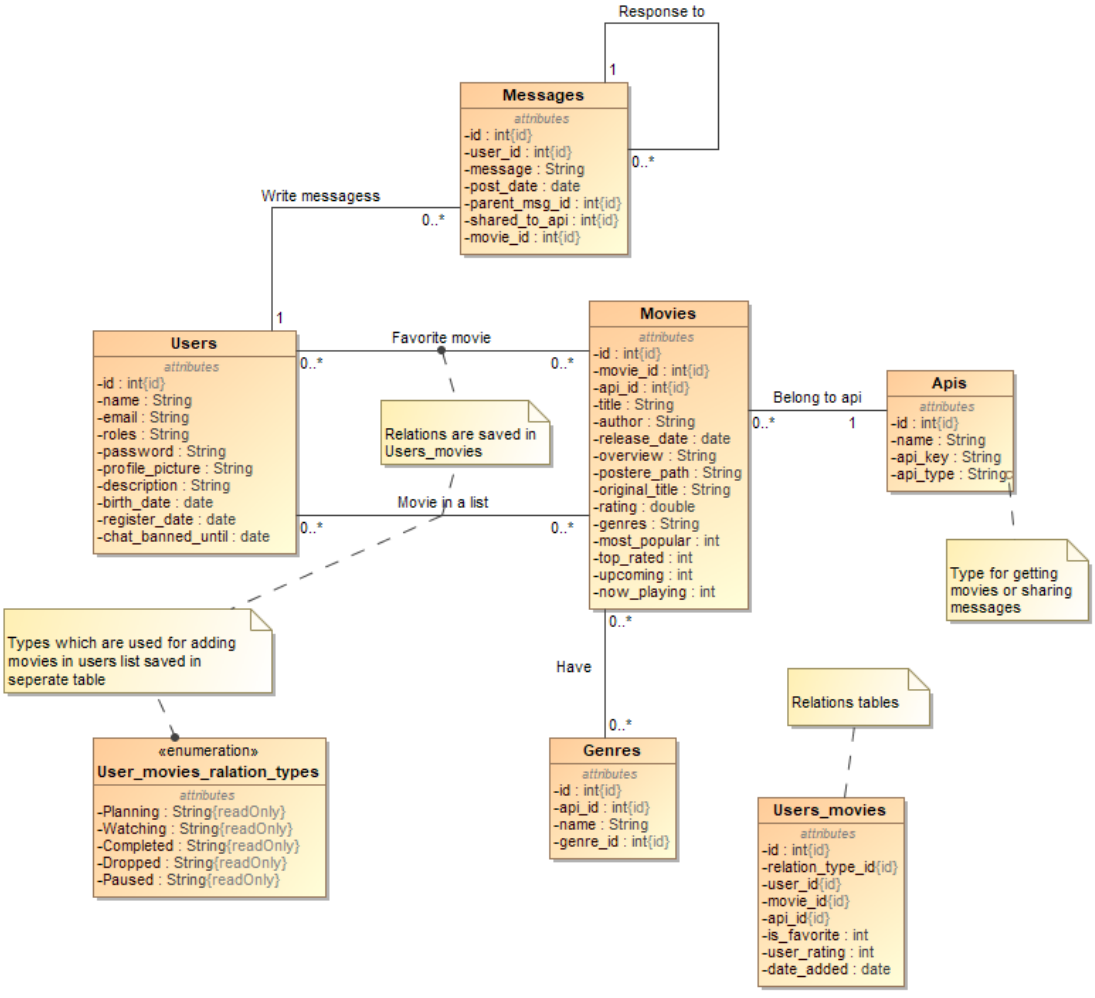
2.2.3. Programavimo kalbos, derinimo, automatizavimo priemonės, operacinė sistemos

Projektas buvo suskaidytas į dvi dalis, kurios naudoja skirtingas programavimo kalbas ir karkasus:

1. API:
    a. *PHP 7* pagrįstą *Symfony 4* karkasą.
    b. *MySQL* duomenų bazė.
2. Vartotojo sąsaja:
    a. *HTML* bei *JavaScript* pagrįstą *VueJs* karkasą.

2.3. Sistemos projektas

2.3.1. Statinis sistemos vaizdas



2.7 pav. Sistemos duomenų bazės schema

Sistemos duomenų bazė sudaryta iš septynių duomenų bazės lentelių, kurias galima suskaidyti į

pagrindines ir antraeiles.

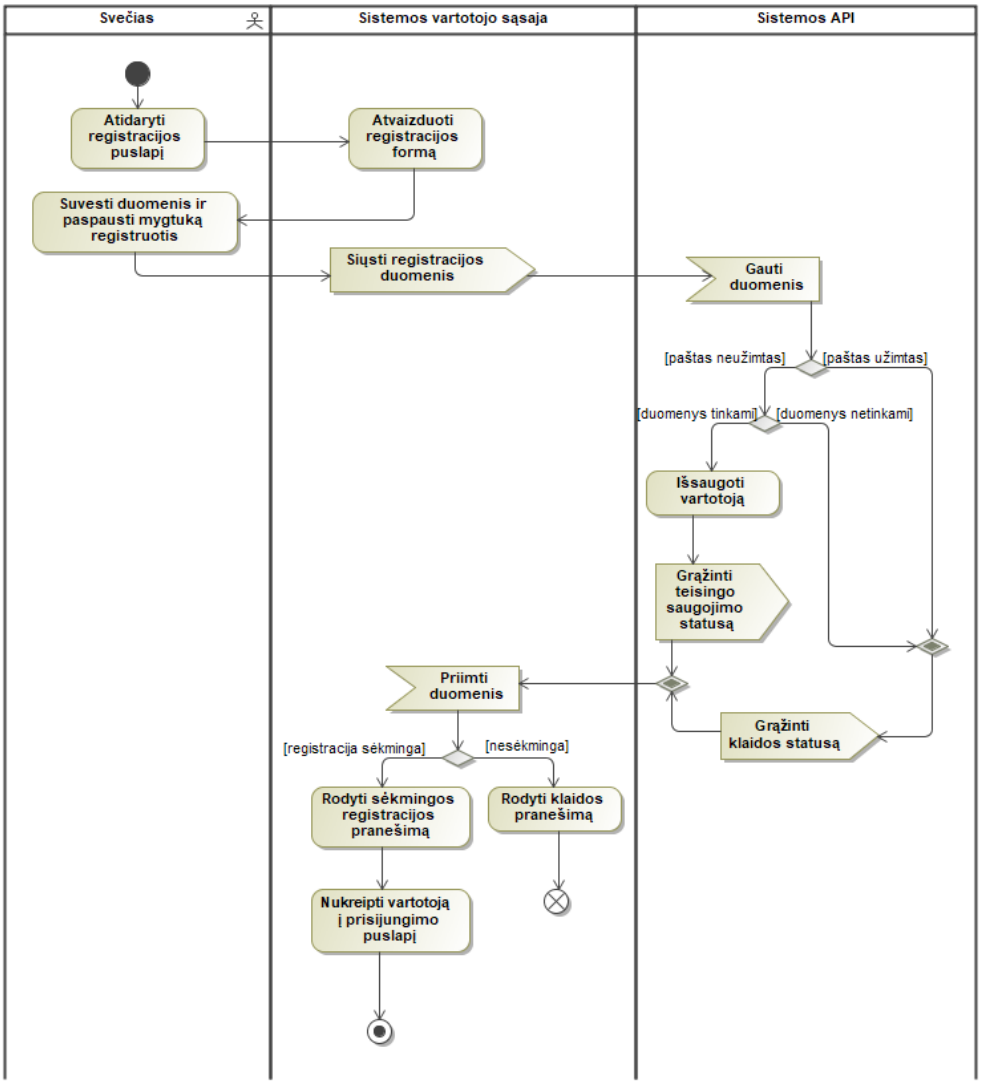
- Pagrindinės lentelės:
    - *Users* – vartotojų duomenų saugojimo lentelė;
    - *Messages* – visų sistemoje paskelbtų žinučių lentelė;
    - *Movies* – Filmų duomenų saugojimo lentelė. Šioje lentelėje duomenys pildosi automatiškai partraukiant duomenis iš viešo API.
    - *Users_movies* – visų vartotojų išsaugotų filmų statistika. Šioje lentelėje saugomas ir filmo tipas vartotojo sąraše ir vartotojo vertinimas bei tai, ar jis yra jo mėgstamiausias filmas.

- Antraeilės - mažiau svarbios lentelės:
    - *User_movies_relation_types* lentelėje saugomi vartotojų sąraše esančių filmų tipai.
    - *Genres* – saugomi filmų žanrai. Ši lentelė pildosi automatiškai iš sistemos gaunant filmų sąrašą.
    - *Apis* - naudojamų API sąrašas. Ši lentelė buvo sukurta planuojant naudoti keletą skirtingų API filmų siuntimui ir taip pat žinučių dalinimuisi.

Pagal naudojamo viešo API naudojimo taisykles duomenys negali būti saugomi ilgiau nei reikalinga svetainės funkcionavimui, todėl lentelės *Movies* ir *Genres* yra periodiškai kas 24 valandas ištrinamos.

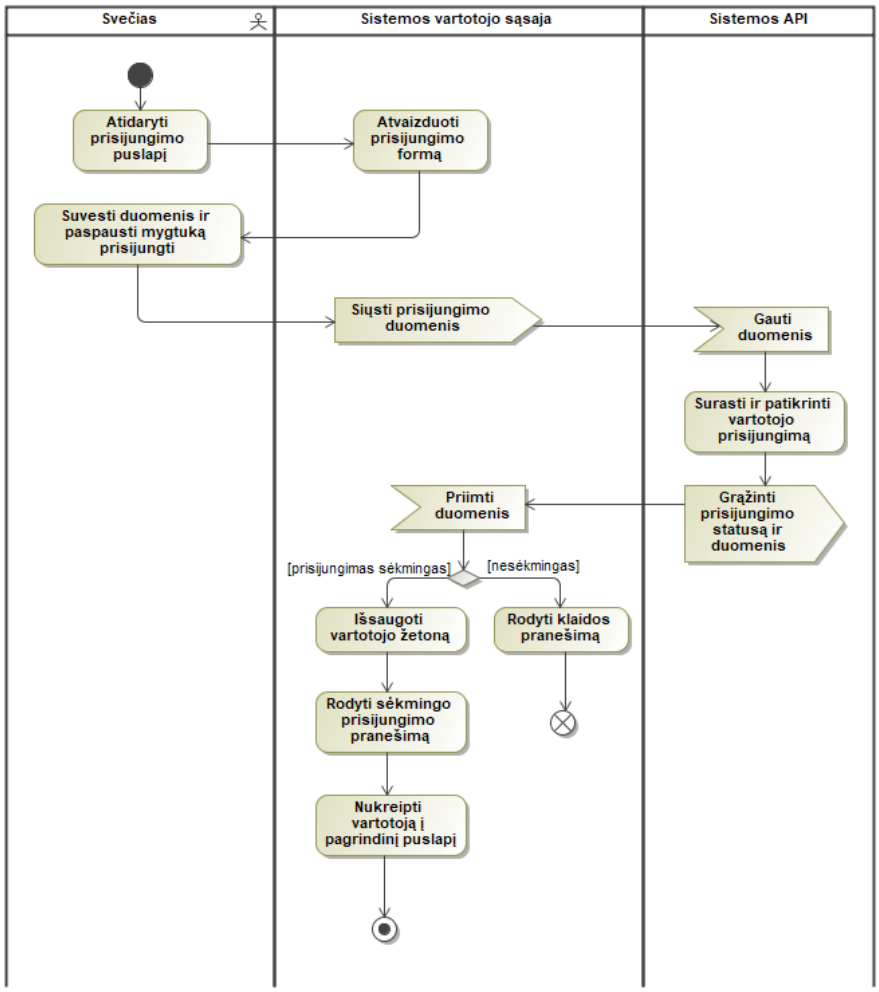
2.3.2. Dinaminis sistemos vaizdas

Dinaminis sistemos vaizdas atvaizduojamas toliau dokumentuotomis *UML* veiklos diagramomis. Šios diagramos atspindi visus galimus panaudojimų atvejus. *Svečių* panaudojimo atvejai vaizduojami 2.6 – 2.17 diagramose, *prisijungusių vartotojų* 2.18 – 2.20 diagramose, o *administratorių* 2.21 diagramoje.



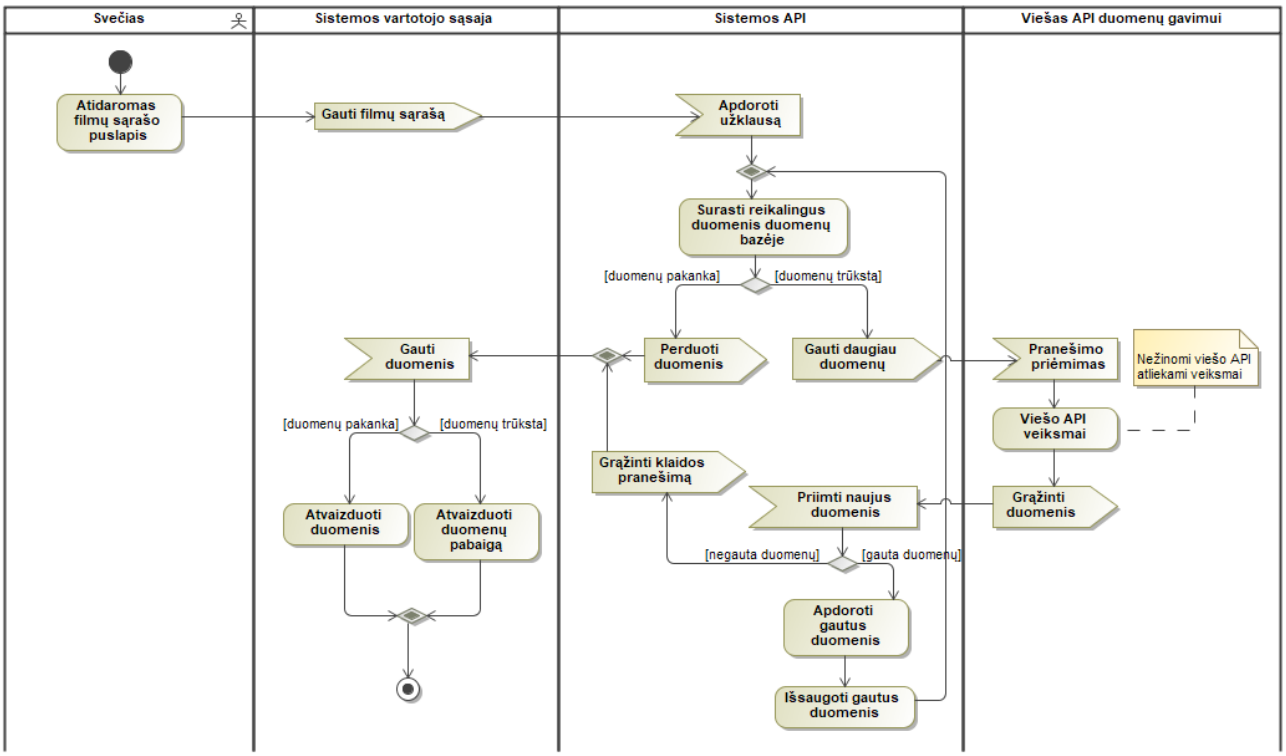
2.8 pav. *UML* veiklos diagrama: Registruotis

Vartotojui *Svečiui* atidarant registracijos puslapį parodoma registracijos forma, kurioje suvedus duomenis ir paspaudus registracijos mygtuką duomenys bus siunčiami į sistemos API. API patikrina ar vartotojo el. paštas neužimtas ir ar kiti duomenys atitinka taisykles. Jei el. paštas neužimtas ir duomenys atitinka, registracija laikoma sėkminga - vartotojas išsaugomas ir grąžinamas teigiamas rezultatas. Priešingu atveju – jei el. paštas užimtas arba kiti duomenys neatitinka, grąžinamas klaidos pranešimas su klaidingu statusu. Vartotojo sąsajai gavus duomenis nesėkmės atveju parodomas klaidos pranešimas, nusakantis klaidos priežastį, o sėkmingos registracijos atveju vartotojas nukreipiamas į prisijungimo puslapį (2.8 pav.).



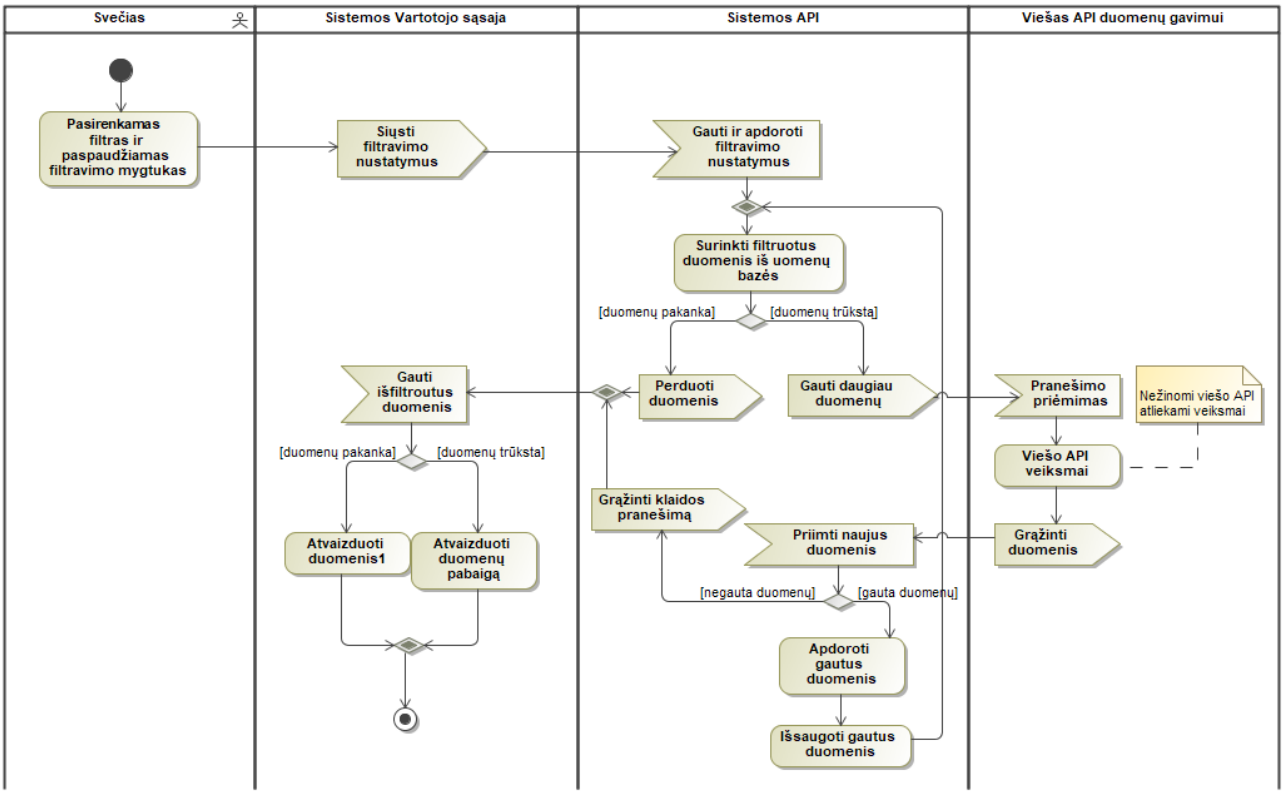
2.9 pav. *UML* veiklos diagrama: Prisijungti

Vartotojui *Svečiui* atidarant prisijungimo puslapį parodoma prisijungimo forma, kurioje suvedus duomenis ir paspaudus prisijungimo mygtuką duomenys bus siunčiami į sistemos API. API patikrina ar vartotojo prisijungimo duomenys teisingi ir grąžina prisijungimo rezultatus, kuriuose teisingo prisijungimo atveju yra ir vartotojo žetonas. Vartotojo sąsajai gavus duomenis nesėkmės atveju parodomas klaidos pranešimas, o sėkmingo prisijungimo atveju vartotojas nukreipiamas į pagrindinį puslapį bei naršyklėje išsaugomas jo žetonas (2.9 pav.).



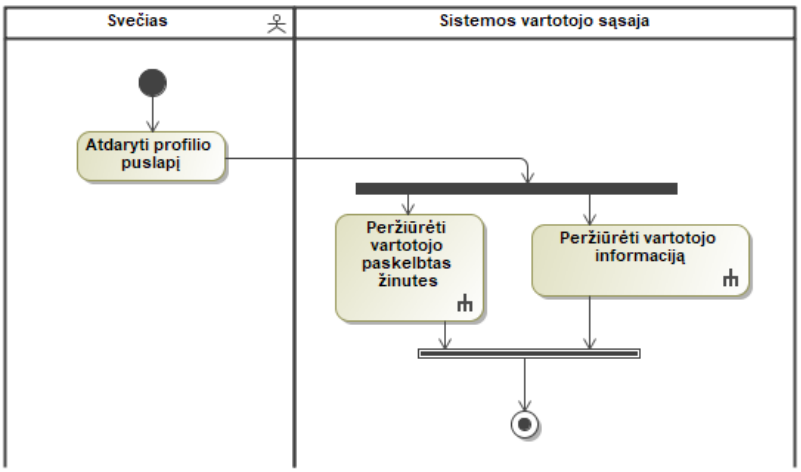
2.10 pav. *UML* veiklos diagrama: Peržiūrėti filmų sąrašą

Vartotojui *Svečiui* atidarant filmų puslapį yra kreipiamasi į sistemos API prašant filmų sąrašo. API atlieka reikalingų filmų paiešką duomenų bazėje ir, jei duomenų pakanka, grąžina duomenis, tačiau jei jų trūksta yra kreipiamasi į naudojamą viešą API duomenų papildymui. Gauti duomenys iš viešo API yra apdorojami pagal sistemos standartą ir išsaugojami duomenų bazėje iš kurios vėl atrenkamas reikalingas duomenų kiekis siuntimui. Jei viešas API negrąžina duomenų yra siunčiama klaida. Vartotojo sąsaja gavusi duomenis juos atvaizduoja ir jei duomenų negauta, atvaizduojama filmų sąrašo pabaiga (2.10 pav.).



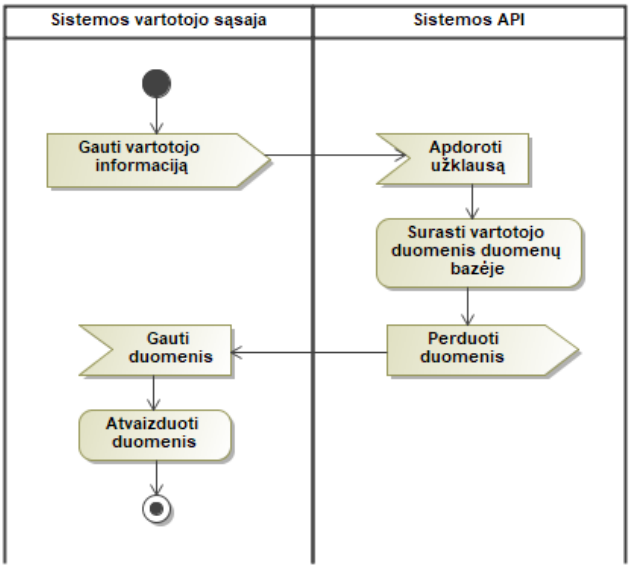
2.11 pav. *UML* veiklos diagrama: Rūšiuoti filmus

Vartotojas *Svečias* būdamas puslapyje, kuris turi filtrą, gali atsirūšiuoti rodomus filtrus pagal įvairius filmų parametrus. Pasirinkus paspaudžiamas filtravimo mygtukas kuris nusiunčia filtro nustatymus į sistemos API, kuris toliau dirba niekuo nesiskiriančiu principu nuo *peržiūrėti filmų sąrašą* UML veiklos diagramos, todėl tolimesnių veiksmų nebeaprašinėju (2.11 pav.).



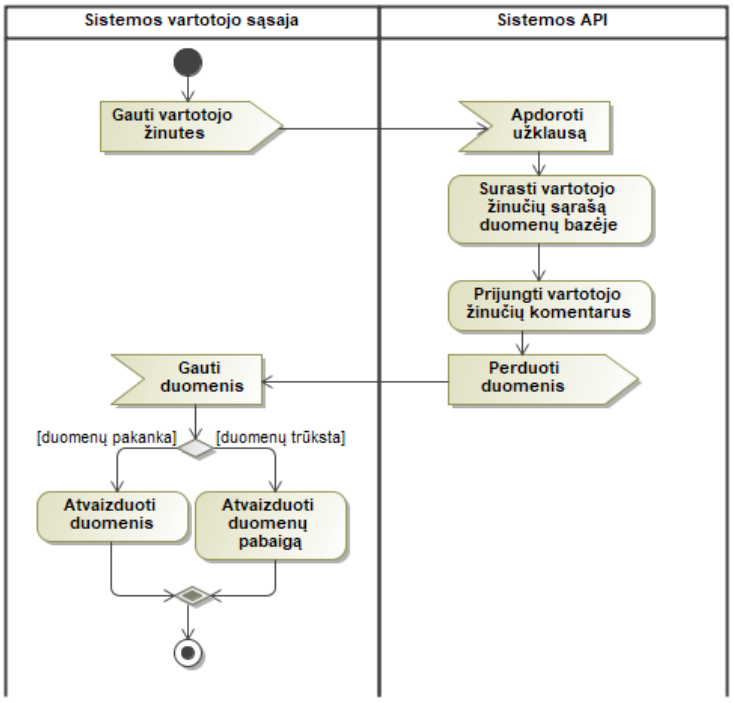
2.12 pav. *UML* veiklos diagrama: Peržiūrėti profilį

Vartotojui *Svečiui* atidarant sistemoje registruoto vartotojo profilio puslapį, viršutinėje jo dalyje yra atvaizduojama vartotojo informacija kurios veiksmai atvaizduojami *Peržiūrėti vartotojo informaciją* UML veiklos diagramoje, o žemiau, skirtukais sudalinti, pasirinkimai, iš kurių numatytasis skirtukas atvaizduoja vartotojo paskelbtas žinutes (tai atvaizduojama *Peržiūrėti vartotojo paskelbtas žinutes* UML veiklos diagramoje).



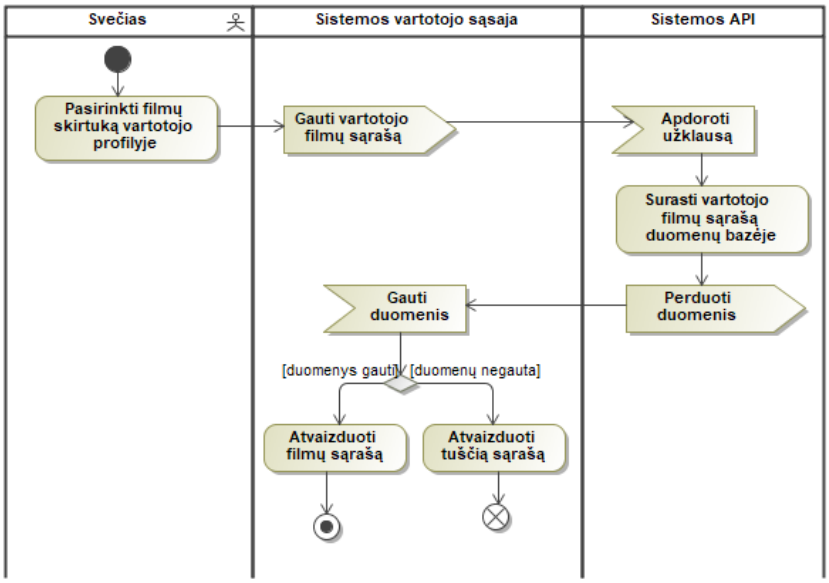
2.13 pav. *UML* veiklos diagrama: Peržiūrėti vartotojo informaciją

Šis veiklos procesas vykdomas tik vartotojui atidarant profilio puslapį. Tuo metu vartotojo sąsaja kreipiasi į API prašydama vartotojo duomenų atvaizdavimui. Gavusi duomenis apie vartotoją juos atvaizduoja (2.13 pav.).



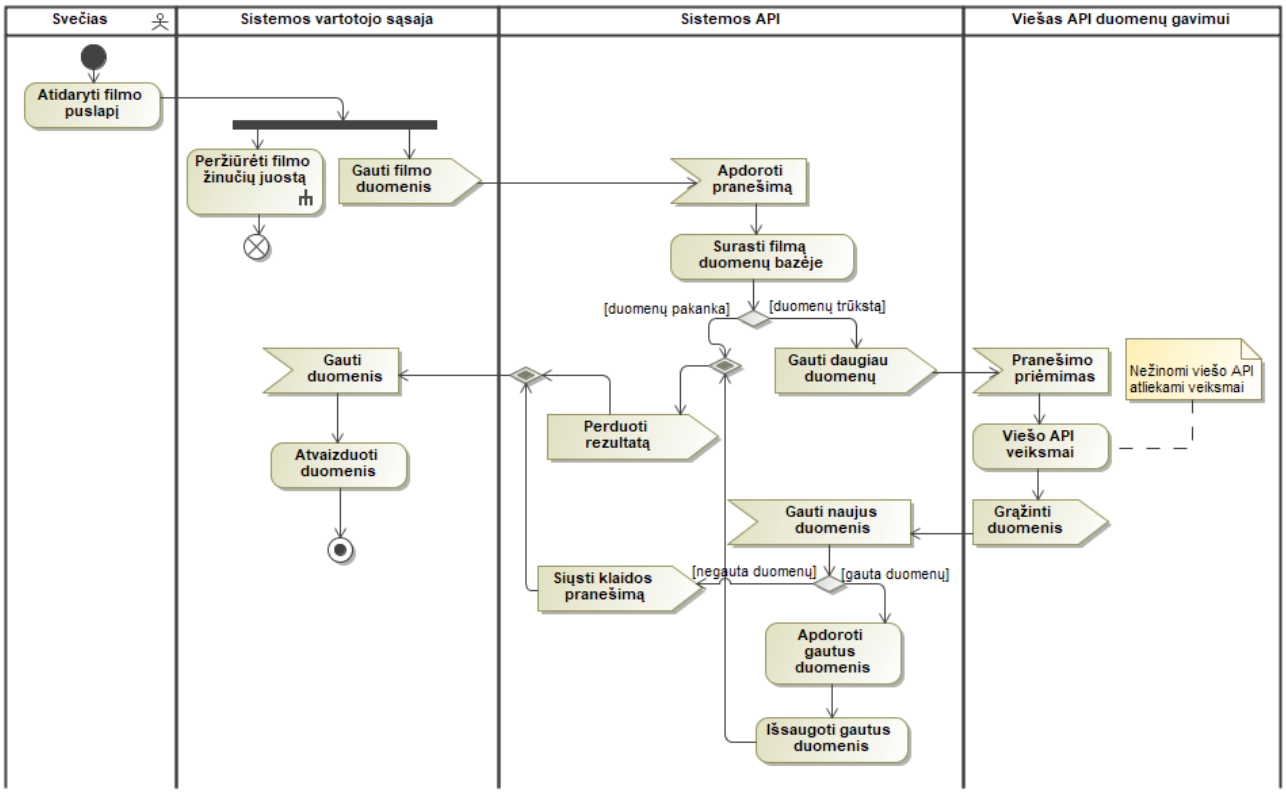
2.14 pav. *UML* veiklos diagrama: Peržiūrėti vartotojo paskelbtas žinutes

Vartotojui atsidarant profilį puslapio apačioje bandoma atvaizduoti jo paskelbtas žinutes – kreipiamasi į sistemos API prašant vartotojo žinučių. API randa visus vartotojo paskelbimus ir prie jų prijungia komentarus po ko seka duomenų grąžinimas. Gauti duomenys atvaizduojami ir jei jų nėra atvaizduojama puslapio pabaiga (2.14 pav.).



2.15 pav. *UML* veiklos diagrama: Peržiūrėti vartotojo filmų sąrašą

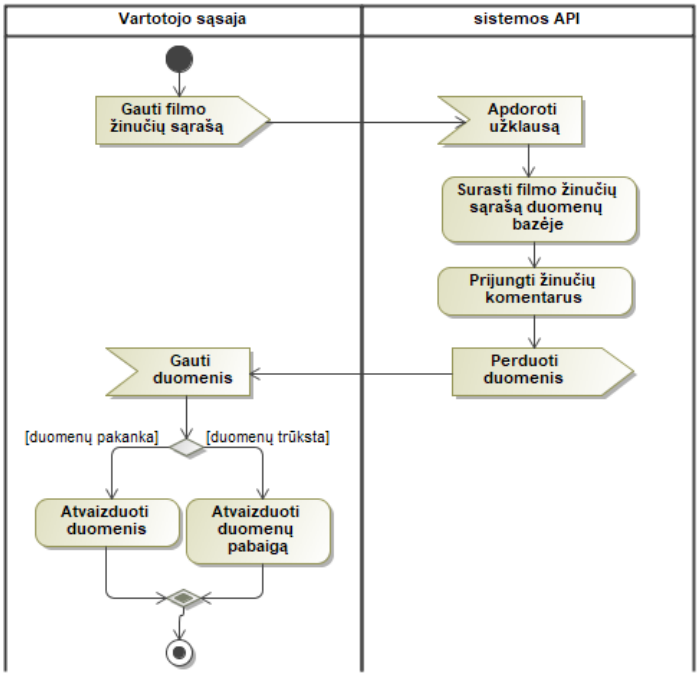
Vartotojas *Svečias* profilyje pasirinkęs filmų skirtuką gali peržiūrėti vartotojo sąraše esančius filmus. Pasirinkus skirtuką siunčiamas duomenų prašymas į sistemos API, kuris randa duomenis duomenų bazėje ir grąžina vartotojo sąsajai. Grįžus duomenims, jei filmų nėra atvaizduojamas tuščias sąrašas, priešingu atveju rodomi vartotojo filmai (2.15 pav.).



2.16 pav. *UML* veiklos diagrama: Peržiūrėti filmo puslapį

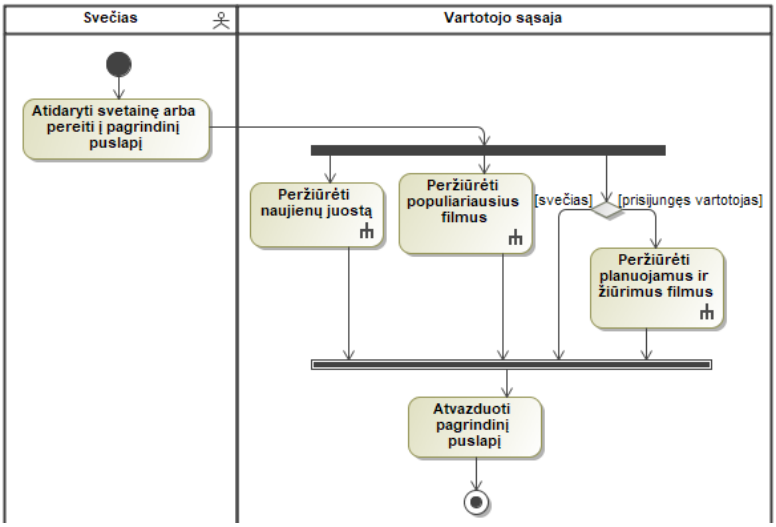
Vartotojui atidarius filmo puslapį viršutinėje puslapio dalyje yra atvaizduojama filmo informacija, o apatinėje – vartotojų rašytos filmo peržiūros, apie kurias daugiau *Peržiūrėti filmo žinučių juostą* UML veiklos diagramoje. Bandant gauti filmo duomenis į API yra nusiunčiamas prašymas duomenims, tada API ieško filmo duomenų bazėje. Jei filmas nerastas arba jei jo duomenų nepakanka, prašymas

siunčiamas į naudojamą viešą API. Grįžus duomenims, jei filmas nebuvo rastas grąžinamas klaidos pranešimas, priešingu – sėkmės atveju grąžinami filmo duomenys. Vartotojo sąsaja klaidos atveju atvaizduoja klaidą arba sėkmės atveju filmo duomenis (2.16 pav.).



2.17 pav. *UML* veiklos diagrama: Peržiūrėti filmo žinučių juostą

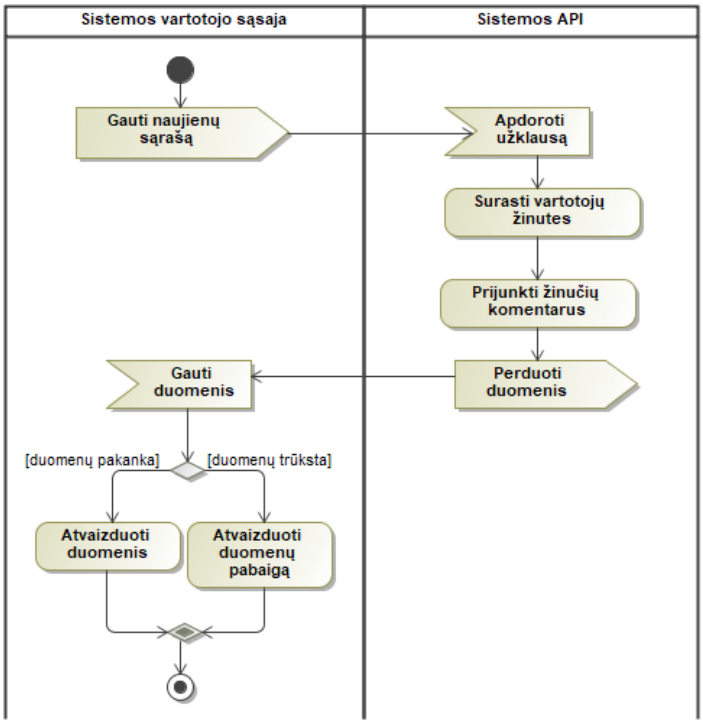
Šis veiklos atvejis vykdomas tik vartotojui atsidarius filmo puslapį. Atsidarymo metu į API siunčiamas duomenų prašymas, o API radęs žinutes apie filmą savo duomenų bazėje ir prijungęs žinučių komentarus jas grąžina. Grįžus duomenys atvaizduojami, tačiau jei duomenų negauta atvaizduojamas ir pabaigos puslapis (2.17 pav.).



2.18 pav. *UML* veiklos diagrama: Peržiūrėti pagrindinį puslapį

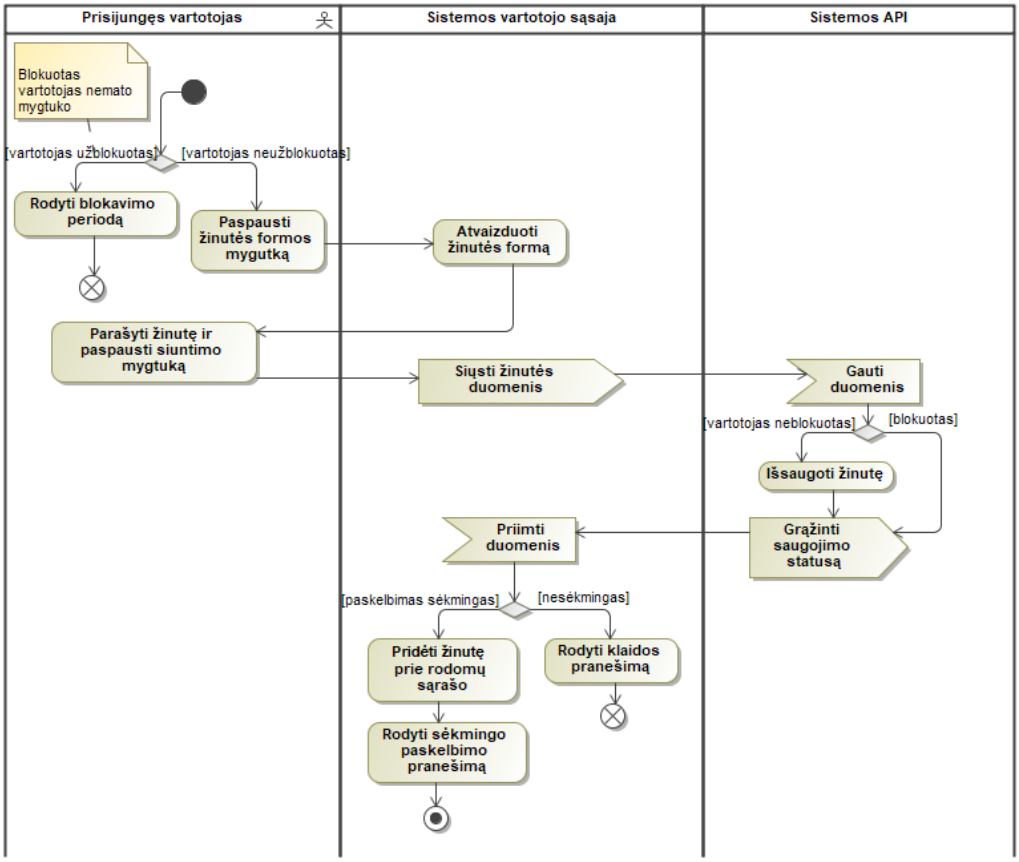
Vartotojas atsidaręs svetainę pradžioje bus nukreipiamas į pagrindinį puslapį, kuris padalintas į tris dalis apie kurias daugiau informacijos jų UML veiklos atvejų diagramose. Naujienų juostos bei populiariausių filmų dalys atvaizduojamos svečiui, tačiau planuojamų ir žiūrimų filmų dalis rodoma

tik prisijungusiam vartotojui, kadangi šioje dalyje duomenys imami iš vartotojo filmų sąrašo (2.18 pav.).



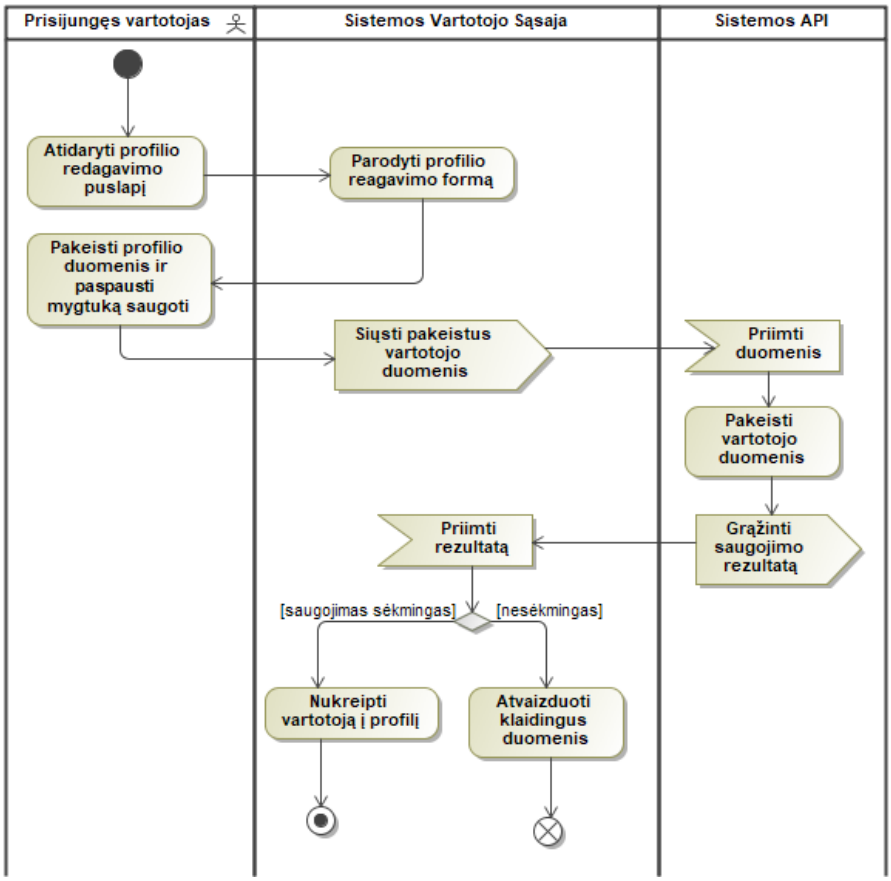
2.19 pav. *UML* veiklos diagrama: Peržiūrėti naujienų juostą

Naujienų juosta rodoma pagrindiniame puslapyje. Jame apsilankius Kreipiamasi į sistemos API prašant naujienų sąrašo. API randa visų vartotojų žinutes, įskaitant ir žinutes apie filmus, prijungia prie jų komentarus bei grąžina gautus duomenis. Duomenys atvaizduojami ir, jei jų negrąžinta, atvaizduojama rezultatų pabaiga. Populiariausių bei planuojamų ir šiuo metu žiūrimų filmų juostos skiriasi minimaliai: jie API pusėje imami iš vartotojų filmų sąrašo vietoj žinučių sąrašo, todėl jų veiklos diagramų nevaizduosiu (2.19 pav.).



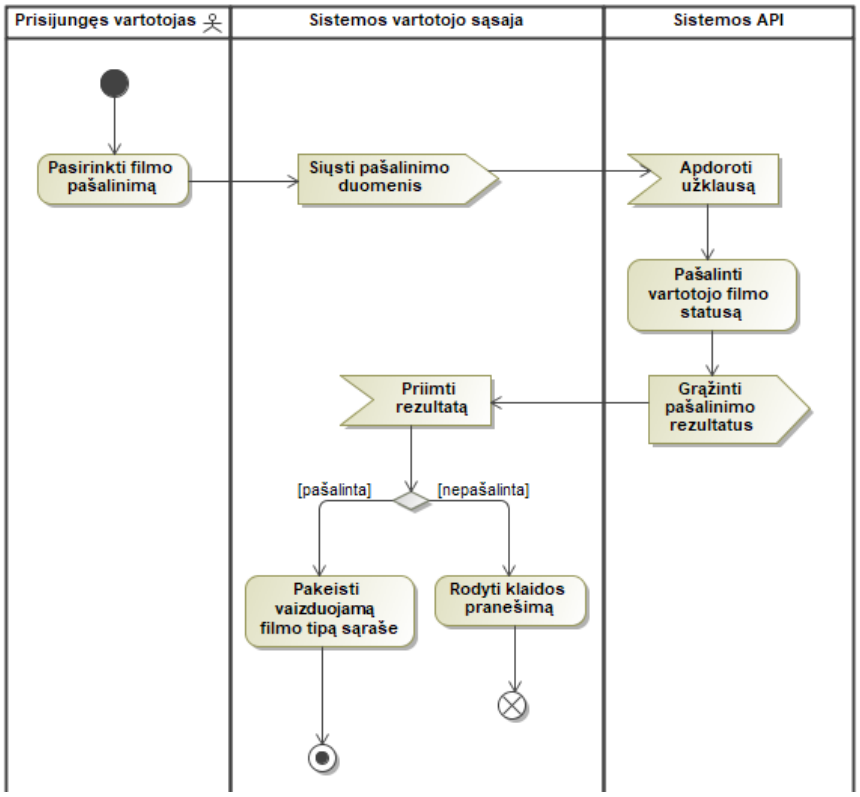
2.20 pav. *UML* veiklos diagrama: Paskelbti žinutę bendroje juostoje

Registruotas sistemoje ir prisijungęs vartotojas turi galimybę rašyti žinutes naujienų juostoje. Jei vartotojas nėra užblokuotas jis pagrindiniame puslapyje matys žinutės rašymo mygtuką, priešingu atveju, vietoj mygtuko bus parašytas vartotojo blokavimo terminas – data iki kada vartotojas užblokuotas. Paspaudus mygtuką atvaizduojama žinutės rašymo forma, gyvai rodanti ir žinutės rezultatą. Parašius žinutę ir paspaudus paskelbimo mygtuką žinutės duomenys nusiunčiami į API, kuris dar kartą patikrina ar vartotojas blokuotas ir jei ne žinutė išsaugoma. Saugojimo rezultatas grąžinamas vartotojo sąsajai ir jei rezultatas teigiamas – žinutė išsaugota, ji pridedama prie žinučių sąrašo vaizdavimo kartu atvaizduojant sėkmingo paskelbimo žinutę, jei ne atvaizduojama klaida. Komentaro rašymo ir žinutės paskelbimo apie filmą panaudojimo atvejai beveik nesiskiria, išskyrus tai, kad komentaras rašomas kitoje žinutėje, o žinutės apie filmą paskelbimas filmo puslapyje, todėl jų diagramų nevaizduosiu (2.20 pav.).



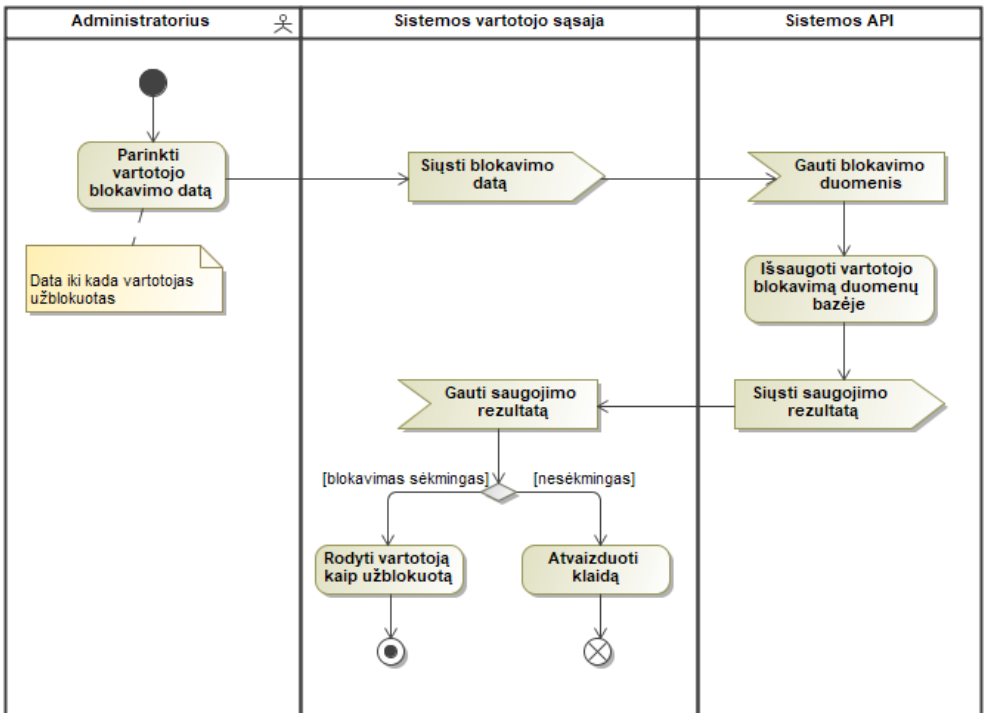
2.21 pav. *UML* veiklos diagrama: Redaguoti profilį

Prisijungęs vartotojas turi galimybę redaguoti savo profilį. Atsidarius profilio ragavimo puslapį jam atvaizduojama forma su dabartiniais vartotojo profilio duomenimis. Atidarant nėra siunčiamas duomenų prašymas į sistemos API, kadangi vartotojo profiliai šiuo metu neturi didelio kiekio duomenų ir prisijungus jie išsaugomi naršyklėje. Pakeitus duomenis ir paspaudus saugojimo mygtuką jie išsiunčiami į sistemos API, kuriame išsaugomi. Kadangi vartotojai gali turėti paveikslėlį, saugant paveikslėlis išsaugomas API serveryje, o jo pavadinimas įrašomas duomenų bazėje. Išsaugojus grąžinamas saugojimo rezultatas ir pagal jį atvaizduojama klaida arba sėkmingo veiksmo atveju vartotojas nukreipiamas į profilio puslapį (2.21 pav.).



2.22 pav. *UML* veiklos diagrama: Pašalinti filmą iš sąrašo

Prisijungęs vartotojas svetainės filmų sąraše arba savo profilio filmų sąraše atsidaręs modalinį langą gali pašalinti filmą iš savo filmų sąrašo. Veikimo principas labai panašus į filmo pridėjimą prie sąrašo, tačiau vietoj pridėjimo filmas iš sąrašo panaikinamas. Taip pat API naikindamas filmą iš sąrašo patikrina ar filmas dar yra pažymėtas kaip mėgstamiausias, jei ne ištrinamas visas duomenų bazės įrašas, priešingu atveju tik pažymima, kad filmo sąraše nėra ir jis tik mėgstamiausias. Toliau einančių panaudojimo atvejų veiklos diagramų (filmo pridėjimo arba išėmimo iš sąrašo kaip mėgstamiausio ir filmo įvertinimo) veikimo principai labai panašūs, todėl jų toliau nedemonstruosiu (2.22 pav.).



2.23 pav. *UML* veiklos diagrama: Uždrausti vartotojo žinučių rašymą

Administratorius turi galimybę blokuoti vartotojus taip neleisdamas jiems skelbti naujų žinučių visoje svetainėje. Vartotojo blokavimo procesas labai paprastas – *administratorius* būdamas vartotojų valdymo puslapyje gali kiekvienam vartotojui suvesti datą, iki kada vartotojas bus užblokuotas. Suvedus datą iškart yra siunčiami blokavimo nurodymai į API, kuris gavęs vartotojo profilį prideda jam blokavimo datą ir grąžina operacijos rezultatą. Jei blokavimas sėkmingas vartotojas atvaizduojamas kaip užblokuotas, priešingu atveju prašoma pakartoti operaciją. Kadangi vartotojų atblokavimo panaudojimo atvejo veiklos diagramos veikimas skiriasi minimaliai, jos neatvaizduosiu (2.23 pav.).

3. Testavimas

3.1. Testavimo planas

Sistemai testuoti projektavimo metu buvo numatyti šie testavimo būdai, priemonės:

1. Statinė kodo analizė;
2. Automatinis vienetų testavimas;
3. Funkcinis testavimas;
4. Rankinis testavimas.

Kuriant projekto API, statinei kodo analizei buvo nuolat taikomas *PhpStorm* esantis kodo redaktorius, kuris performatuoja kodą pagal nustatytus standartus ir taip pat kodo analizei bei galimų bėdų radimui buvo naudotas *PHPMD – „PHP Mess Detector“* [9] įrankis.

Kuriant vartotojo sąsają, kūrimo metu taip pat buvo nuolat taikoma statinė kodo analizė su *Visual studio code* plėtiniu *Vetur* [8], nuolat pažyminčiu bei performatuojančiu kodo bei stiliaus klaidas.

Taip pat projekto pabaigoje buvo atliekama abiejų sistemos dalių programos kodo peržiūra, siekiant kuo labiau sumažinti galimai likusias klaidas, kurios nepastebimos ankščiau minėtų įrankių.

3.2. Testavimo kriterijai

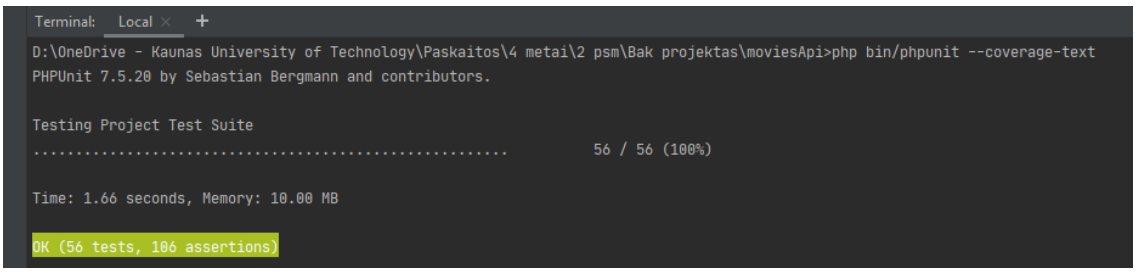
Sistemos testavimai buvo atlikti taikant šiuos kriterijus:

1. Programoje neturi būti likusio jokio nenaudojamo, užkomentuoto ar nevykdomo kodo, kuris tik užima vietą. Čia priskiriami ir nenaudojami kintamieji ar funkcijos.
2. Kode neturi būti „kietai“ užkoduotų reikšmių, kurios nekinta skirtingų įvykių metu (ši dalis testuojant nubuvo 100% įvykdyta).
3. Kode negali būti palikta tikrinimams naudojamų kodo išrašymo funkcijų tokių kaip *dump* ar *print_r*.
4. Automatiniai vienetų testai turi padengti bent 90% objektų klasių kodo.
5. Funkciniu testavimu automatiškai ištestuoti bent svarbiausius sistemos funkcionalumus.
6. Vartotojo sąsaja turi duomenis atvaizduoti korektiškai, po nesėkmingų veiksmų rodyti klaidos pranešimus bei būti reaktyvi.

3.3. Komponentų testavimas

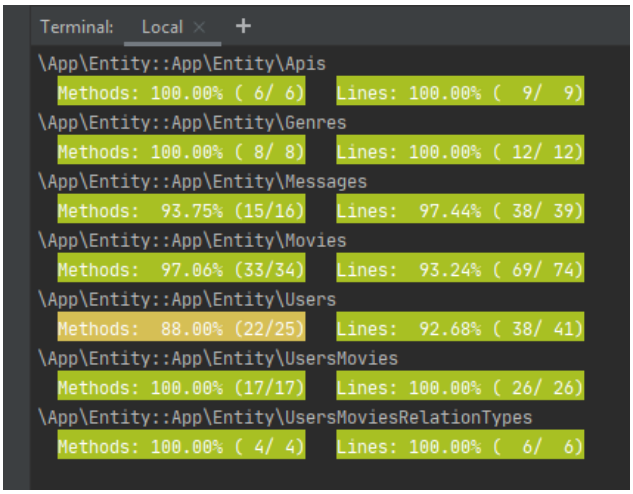
Sistemos API vienetų testavimas buvo atliekamas naudojant *PHPUnit* karkasą [11], kurio pagalba buvo ištestuotos visos sistemoje naudojamų objektų klasės bei taip pat funkcinio testavimo metu buvo ištestuoti pagrindiniai sistemos API metodai.

Testuojant naudojamų objektų klases buvo atliki 56 testai ir 106 patikrinimai (3.1 pav.).



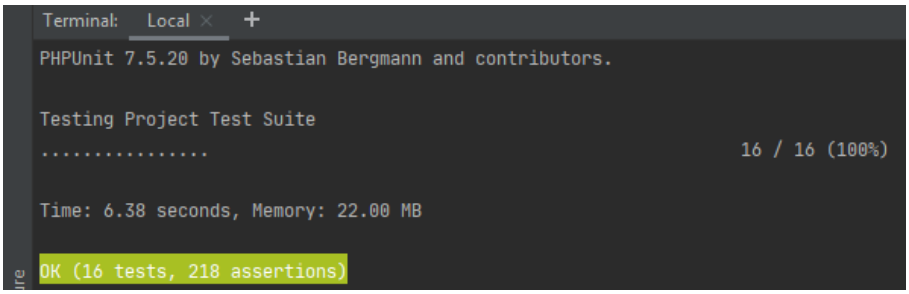
3.1 pav. Objektų klasių galutiniai testavimo rezultatai

Taip pat padengta bent 90% klasės kodo (3.2 pav.).



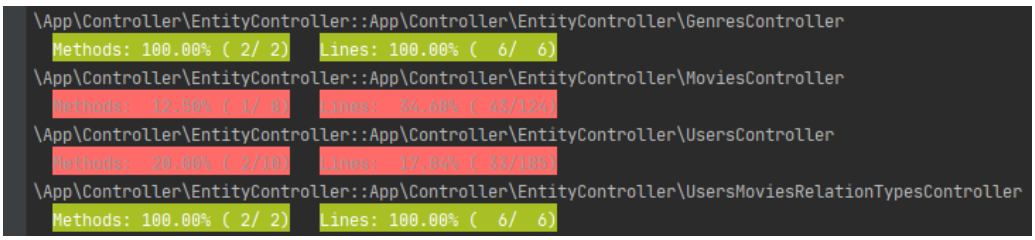
3.2 pav. Klasių testuojamo kodo padengimas

Duomenų saugykloms ir kontroleriams buvo atliktas automatinis funkcinis testavimas padengiantis pagrindines funkcijas. Šiam testavimui buvo sukurta 16 testų ir atlikta 218 patikrinimų (3.3 pav.). Patikrinimų kiekis didelis nes duomenų saugyklų metodų testavimo metu buvo tikrinama ar kiekvienas gautas rezultatas atitinka reikalavimus.

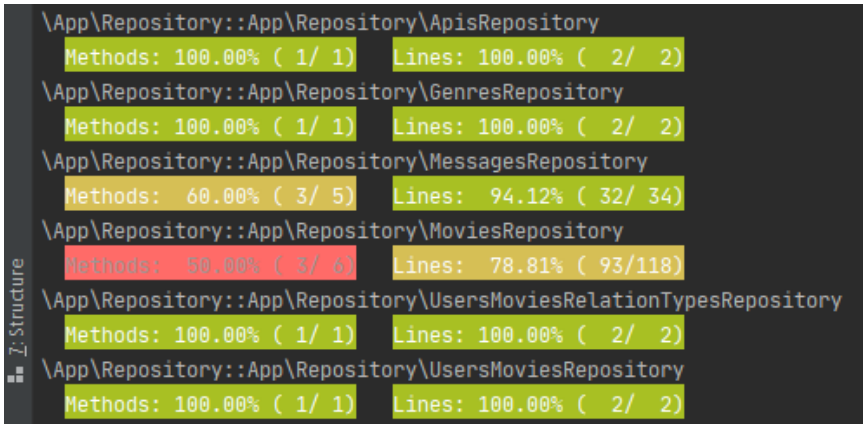


3.3 pav. Duomenų saugyklų ir kontrolerių galutiniai testavimo rezultatai

Taip pat matoma kiek klasių metodų ir kodo buvo padengta naudojant automatinį testavimą (3.4 pav. ir 3.5 pav.).



3.4 pav. Funkcinio testavimo kontrolerių padengimas



3.5 pav. Funkcinio testavimo duomenų saugyklų padengimas

Kitos, automatiniame testavime neišbandytos funkcijos buvo testuojamos rankiniu būdu, pagal priede 1 nurodytą API dokumentaciją, tikrinant, kad rezultatai atitiktų turimų gauti atsakymų reikalavimus naudojant skirtingus parametrus.

3.4. Vartotojo sąsajos testavimas

Vartotojo sąsajos testavimas buvo atliekamas rankiniu būdu naršyklės pagalba. Buvo naudojama naršyklė Google Chrome, o testavimas atliekamas sukuriant scenarijus, kurie apimtų visus panaudojimo atvejus. Rašant scenarijus buvo stengtasi įtraukti po kelis panaudojimo atvejus taip stengiantis padidinti efektyvumą. Toliau atvaizduojama keletas sukurtų scenarijų kiekvienam vartotojui pavyzdžių.

Neprisijungusio vartotojo - *svečio* rankinio testavimo scenarijai atvaizduojami 3.1 lentelėje.

3.1 lentelė. Vartotojo sąsajos Svečio rankinio testavimo scenarijų pavyzdžiai

Scenarijaus pavadinimas	**Testavimo veiksmai**	**Tikėtina baigtis**
Pagrindinio sistemos lango peržiūra	- Atidaromas svetainės puslapis.	Matomas į tris dalis vertikaliai padalintas pagrindinio puslapio langas. Pirmoje dalyje populiariausi filmai, antroje naujienų juosta su žinutėmis, o trečioji juosta neatvaizduojama ir rodoma žinutė prašanti registruotis.
Filmų peržiūra ir filtravimas	- Atidaromas svetainės puslapis. - Viršutinėje navigacijos juostoje pasirenkamas filmų puslapis.	Filmų puslapyje atvaizduojami filmai, kurie, naudojant filtrus, įvairiai atrūšiuojami.

	- Filmų sąrašas rūšiuojamas pasirenkant įvairius filtrus ir spaudžiant filtravimo mygtuką.	
Registracija	- Atidaromas svetainės puslapis. - Navigacijoje pasirenkamas registracijos puslapis - Užpildoma registracijos forma ir spaudžiamas registracijos mygtukas. - Prisijungiama su registracijoje užpildytais duomenimis	Vedant duomenis registracijoje patikrinamas duomenų klaidos atvaizdavimas. Sėkmingai prisijungiama su užregistruotais duomenimis.

Prisijungusio vartotojo rankinio testavimo scenarijai atvaizduojami 3.2 lentelėje.

3.2 lentelė. Vartotojo sąsajos prisijungusio vartotojo rankinio testavimo scenarijų pavyzdžiai

Scenarijaus pavadinimas	**Testavimo veiksmai**	**Tikėtina baigtis**
Pagrindinio sistemos lango peržiūra	- Atidaromas svetainės puslapis.	Matomas į tris dalis vertikaliai padalintas pagrindinio puslapio langas. Pirmoje dalyje populiariausi filmai, antroje naujienų juosta su žinutėmis, o trečiojoje juostoje atvaizduojamas žiūrimų bei planuojamų filmų sąrašas arba žinutė, kad tokių sąraše nėra.
Filmų peržiūra, filtravimas ir priėjimas prie sąrašo.	- Atidaromas svetainės puslapis. - Prisijungiama su paprastu vartotoju. - Viršutinėje navigacijos juostoje pasirenkamas filmų puslapis. - Filmų sąrašas rūšiuojamas pasirenkant įvairius filtrus ir spaudžiant filtravimo mygtuką. - Spaudžiant filmų kortelėse atsiradusius mygtukus, jie prisideda prie sąrašo. - Vartotojo navigacijos juostoje pasirenkame profilį. - Pasirenkame filmų sąrašo skirtuką.	Filmų puslapyje atvaizduojami filmai, kurie, naudojant filtrus, įvairiai atrūšiuojami. Prie kiekvieno filmo matomi mygtukai leidžiantys filmą pridėti prie sąrašo arba pažymėti kaip mėgstamiausią. Paspaudus mygtukus filmai prisideda prie sąrašo, kuris matomas profilyje, pasirinkus filmų sąrašo skirtuką, o profilio įjungimo pradžioje matomos vartotojo skelbtos žinutės.

Administratoriaus rankinio testavimo scenarijai atvaizduojami 3.3 lentelėje.

3.3 lentelė. Vartotojo sąsajos administratoriaus rankinio testavimo scenarijų pavyzdžiai

Scenarijaus pavadinimas	**Testavimo veiksmai**	**Tikėtina baigtis**
Vartotojų peržiūra, rolės keitimas, blokavimas ir atblokavimas	- Atidaromas svetainės puslapis. - Prisijungiama naudojant administratoriaus vartotoją. - Vartotojo navigacijos juostoje pasirenkame administratoriaus meniu	Matomas į tris dalis vertikaliai padalintas pagrindinio puslapio langas. Pirmoje dalyje populiariausi filmai, antroje naujienų juosta su žinutėmis, o trečiojoje juostoje atvaizduojamas žiūrimų bei planuojamų filmų sąrašas arba žinutė, kad tokių sąraše nėra.

4. Dokumentacija naudotojui

4.1. Apibendrintas sistemos galimybių aprašymas

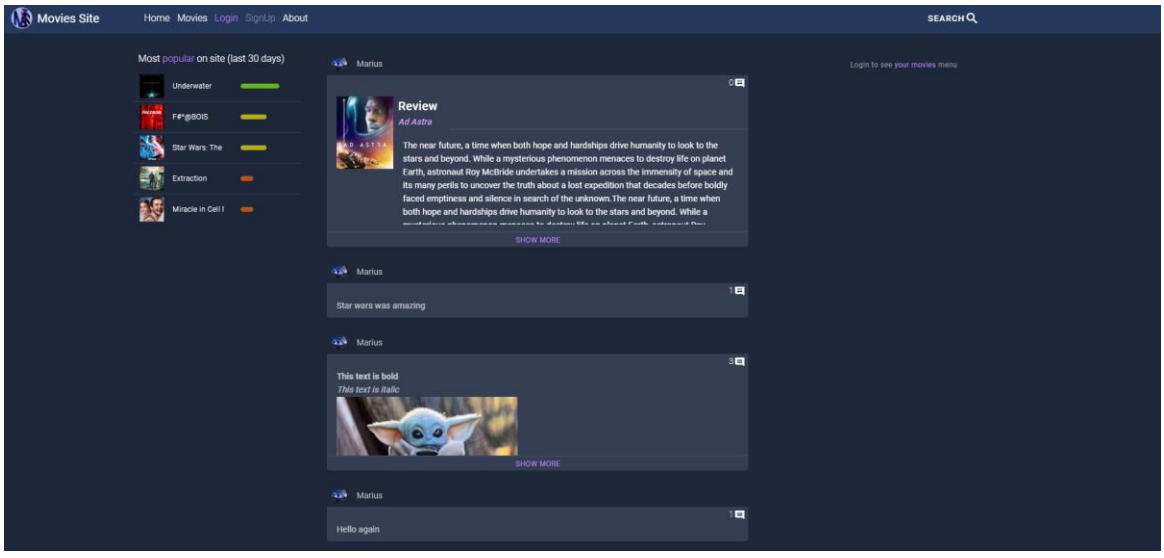
Projektas suteikia vartotojams galimybę naudojant realizuotą internetinę svetainę sukurti filmų sąrašą, kuriame galima įtraukti įvairius filmus priskiriant jiems skirtingus statusus, peržiūrėti kitų vartotojų sąrašus, rasti filmų informaciją bei ne tik bendrauti su kitais sistemos naudotojais bet ir rašyti filmų peržiūras ar netgi juos įvertinti. Didelė dalis svetainės funkcionalumo yra prieinama svečiams – neregistruotiems ir neprisijungusiems svetainės naudotojams, tačiau norint sukuti nuosavą sąrašą ar prisijungti prie bendravimo svetainėje registracija yra privaloma.

4.2. Vartotojo vadovas

1 priede pateikiamos sistemos API užklausų dokumentacijos. Kiekvienas API metodo iškvietimas atvaizduojamas 4.1 – 4.21 lentelėse, parodant ir gaunamų rezultatų pavyzdžius.

Toliau vartotojo vadove pateikiami aprašyti ir atvaizduoti paveikslėliais visi naudojimosi svetaine scenarijai pradedant nuo neregistruoto vartotojo galimybių, judant prie registracijos, prisijungimo bei sąrašo sudarymo, baigiant žinučių rašymu ir administratoriaus galimybėmis. Svetainės duomenų bazė užpildyta pavyzdiniais duomenimis, todėl paveikslėliai taip pat atvaizduos ir galimus kitų vartotojų atliktus veiksmus pvz. žinutes, filmų sąrašus ar filmų populiarumo rodiklius.

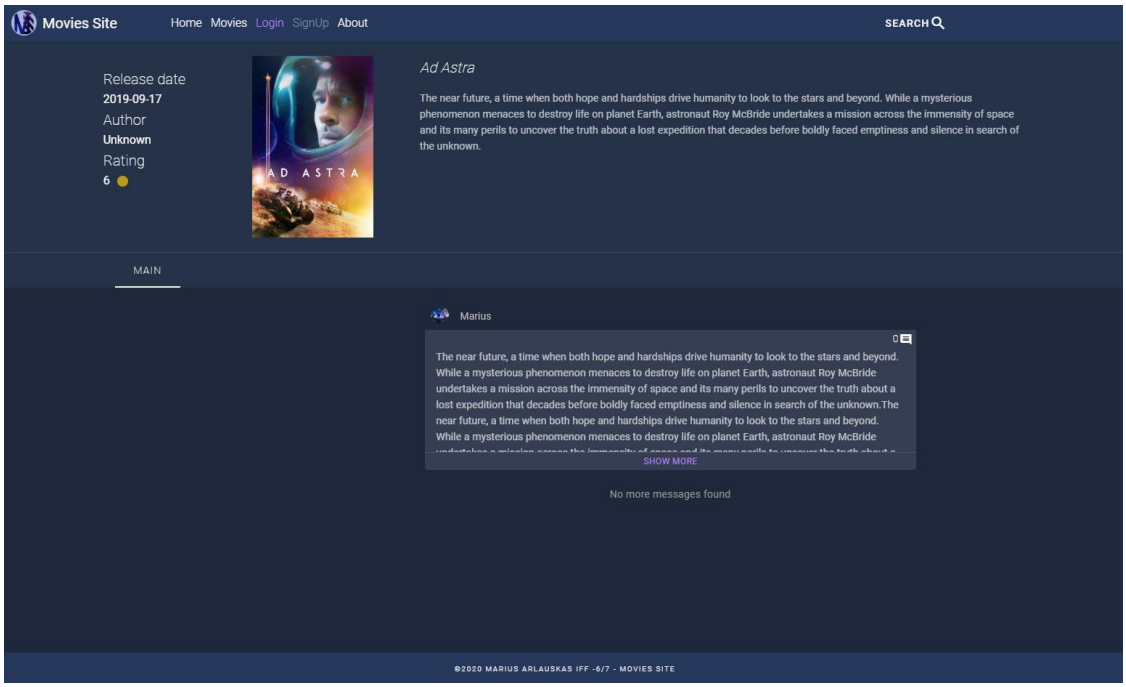
Pirmiausia naujas vartotojas atsidaręs internetinį puslapį bus laikomas svečiu ir matys pagrindinį svetainės langą (4.1 pav.), kuris padalintas į tris dalis vertikaliai. Kairinėje dalyje atvaizduojami populiariausi filmai, vidurinėje naujienų juosta, atvaizduojanti vartotojų žinutes bei filmų peržiūras, dešininėje prisijungusio vartotojo juosta, kurioje prisijungus bus matomas papildomas meniu.



4.1 pav. Pagrindinis svetainės langas

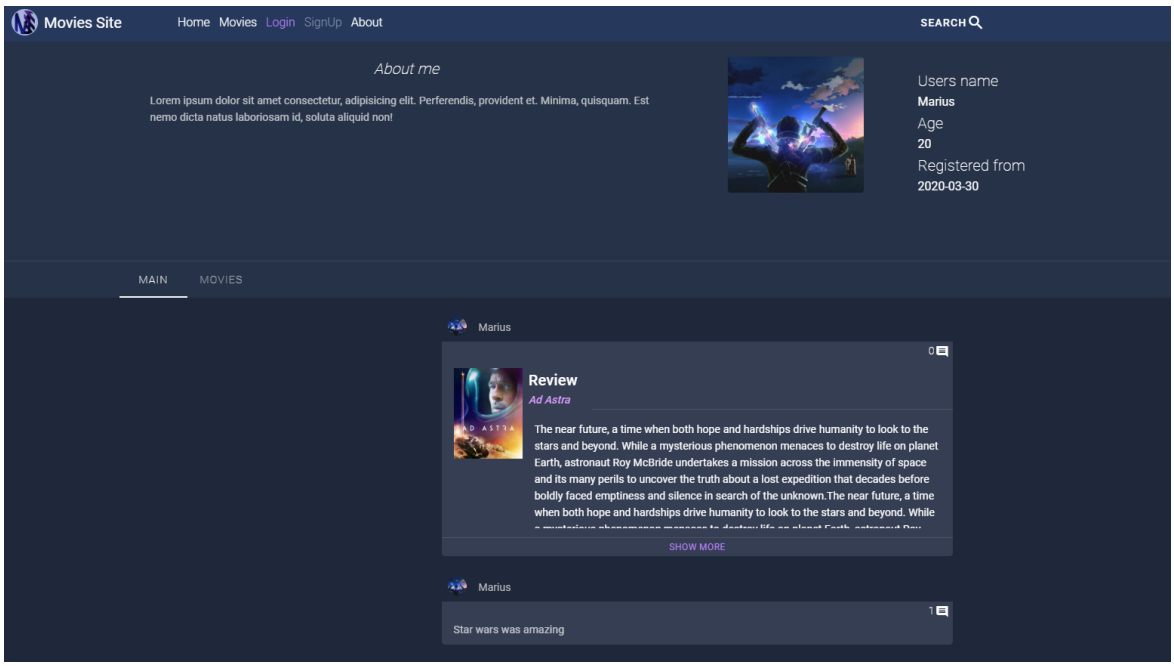
Paspaudus ant filmo paveikslėlio esančio kairinėje populiariausių filmų sąraše galima atsidaryti filmo modalinį langą (4.2 pav.) kuris leidžia peržiūrėti filmo informaciją ir naudojant apačioje dešinėje esantį mygtuką nukeliauti į filmo pagrindinį puslapį (4.3 pav.). Filmo puslapį taip pat galima atsidaryti ir spaudžiant filmo peržiūros žinutėje esantį jo paveikslėlį.

4.2 pav. Filmo modalinis langas

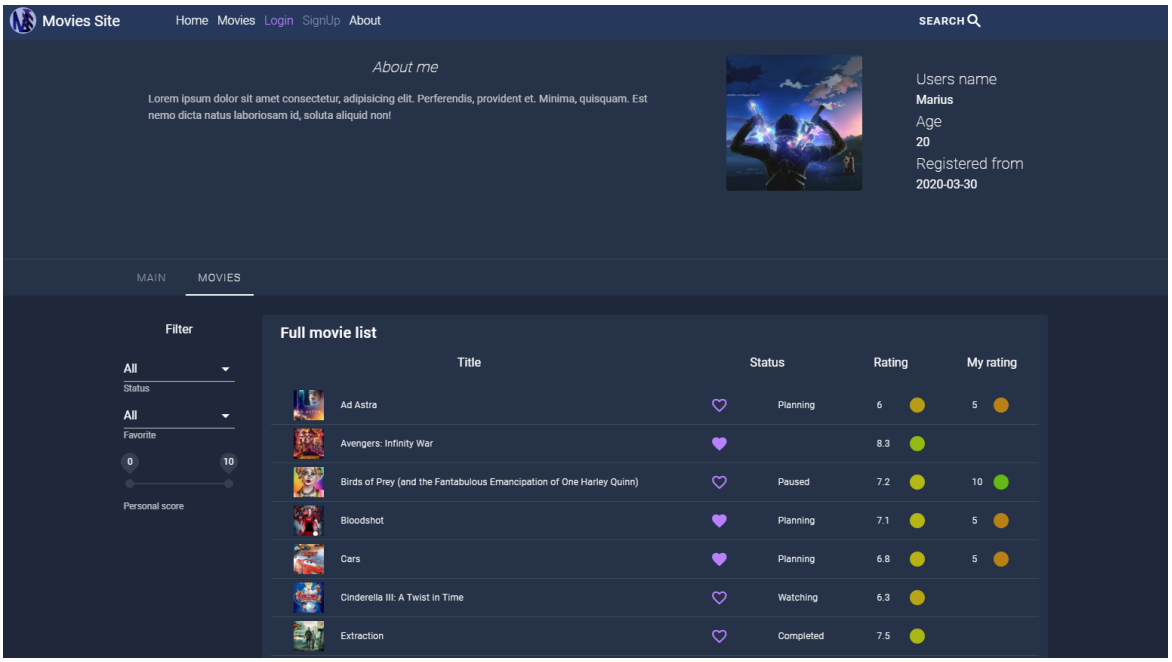


4.3 pav. Pagrindinis filmo puslapis

Vietoj filmo paveikslėlio naujienų juostoje spaudžiant ant vartotojo paveikslėlio ar vardo bus atidaromas vartotojo profilio puslapis (4.4 pav.). Šis puslapis suskaidytas į dalis horizontaliai, kur viršutinėje dalyje vartotojo informacija o apatinėje, pasirinktinai nuo atidaryto skirtuko, vartotojo parašytų žinučių sąrašas arba filmų sąrašas (4.5 pav.). Filmų sąrašas taip pat turi filtravimo funkciją, bei paspaudus ant įrašo galima atsidaryti modalinį filmo langą.

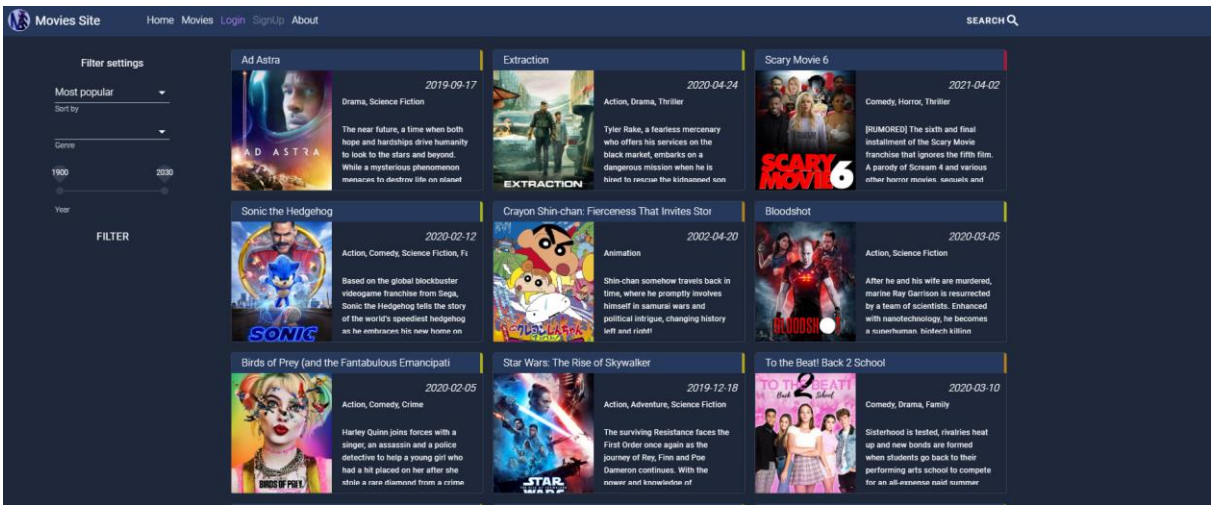


4.4 pav. Vartotojo profilio puslapis su atidarytu žinučių skirtuku



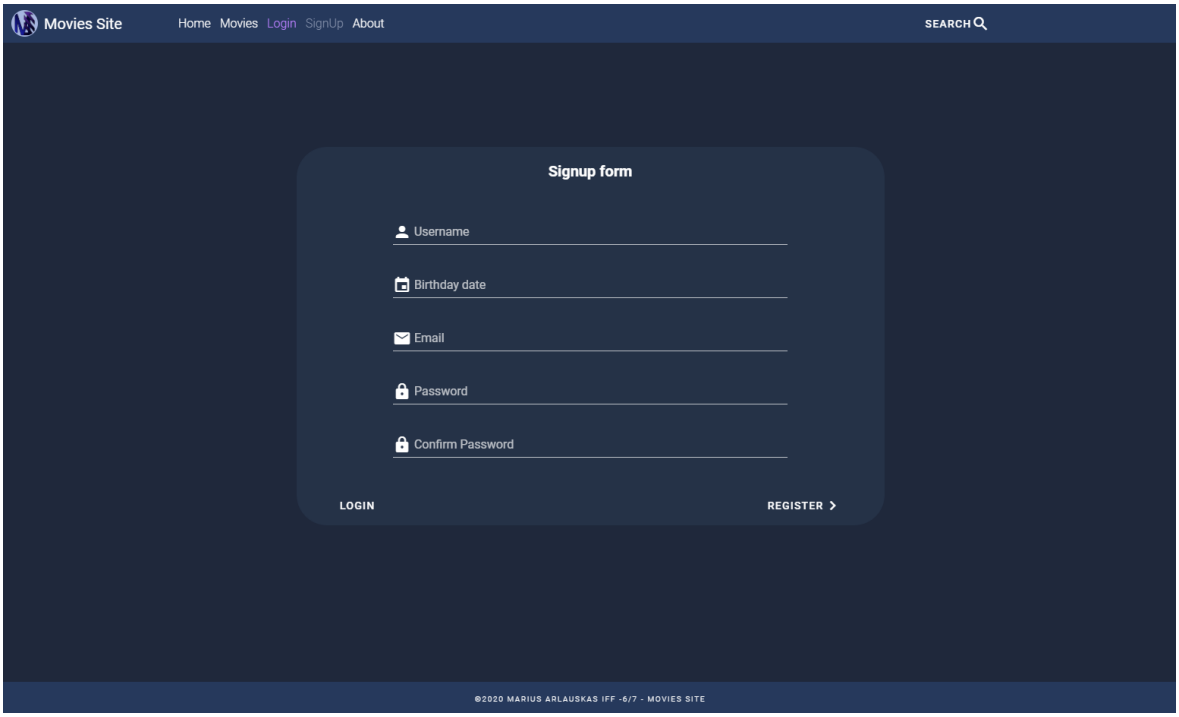
4.5 pav. Pagrindinis puslapis su atidarytu filmų sąrašo skirtuku

Vartotojas pasirinkęs filmų puslapį viršutinėje navigacijos juostoje gali nukeliauti į visų svetainės filmų sąrašo puslapį (4.6 pav.). Šiame puslapyje galima peržiūrėti filmus ir naudojant kairėje esančią filtravimo juostą juos atsifiltruoti – pasirinkus filtrus ir paspaudžiant filtravimo mygtuką.

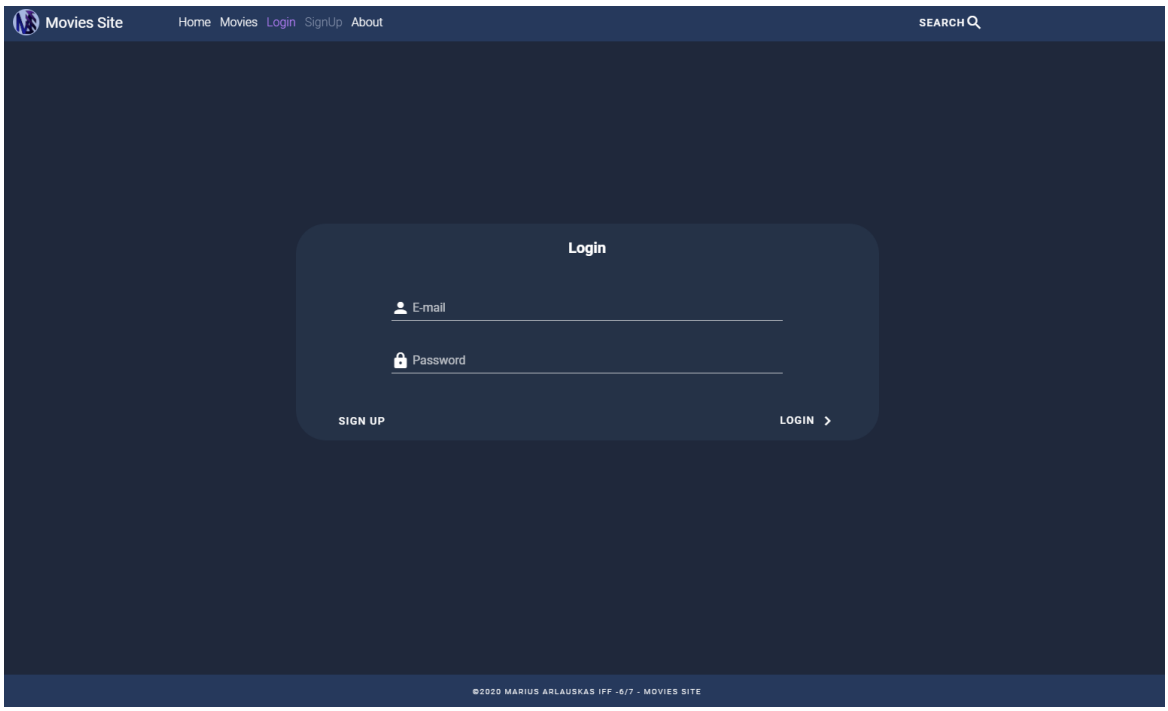


4.6 pav. Filmų puslapio langas

Pasinaudojus viršutine navigacijos juosta atsidarome registracijos puslapį (4.7 pav.), kuriame suvedus visus duomenis ir sėkmingai užsiregistravę būsime nukreipti į prisijungimo puslapį (4.8 pav.), kuris taip pat gali būti prieinamas iš navigacijos juostos.

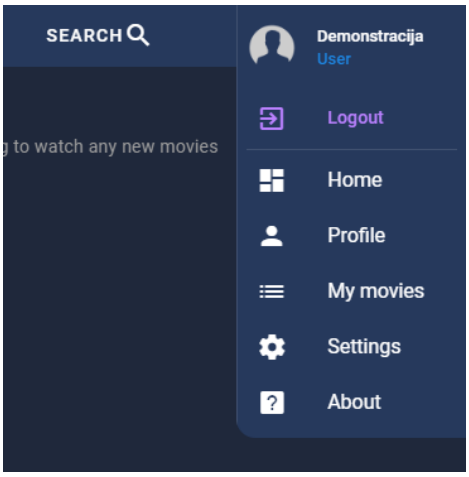


4.7 pav. Registracijos puslapis



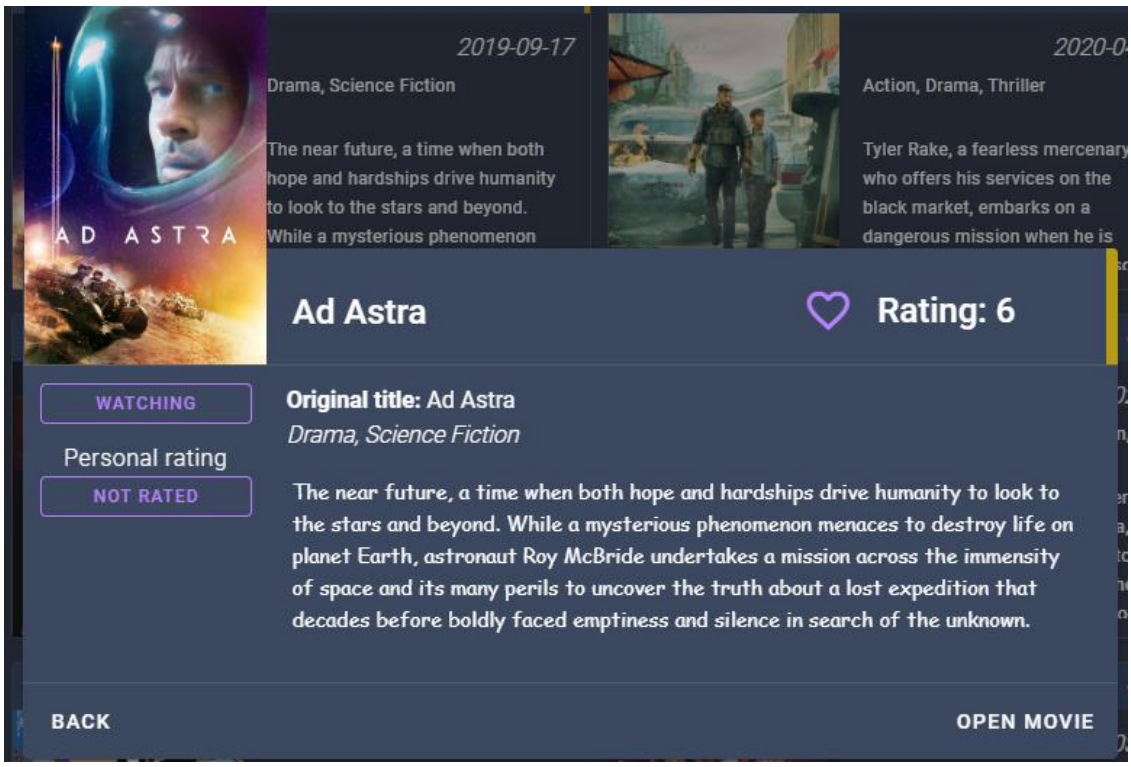
4.8 pav. Prisijungimo puslapis

Prisijungus dešiniajame viršutiniame svetainės kampe atsiranda vartotojo navigacijos meniu (4.9 pav.), kuris, užvedus pelę prasiplečia ir aprašo kiekvieno mygtuko atliekamus veiksmus.



4.9 pav. Vartotojo navigacijos meniu

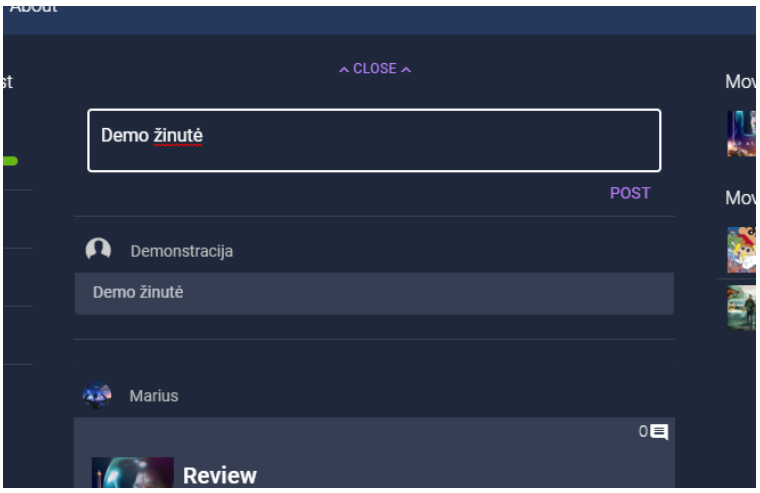
Prisijungęs vartotojas gali žiūrėdamas filmus juos taip pat prisidėti prie savo sąrašo pasirinkdamas filmo tipą, jį įvertinti arba pasižymėti kaip mėgstamiausią paspausdamas širdies paveikslėlį. Toliau pademonstruota kaip pasikeičia prisijungusio vartotojo modalinis filmo langas (4.10 pav.) bei visų filmų sąraše filmą atvaizduojančios kortelės (4.11 pav.).



4.10 pav. Prisijungusio vartotojo modalinis filmo langas

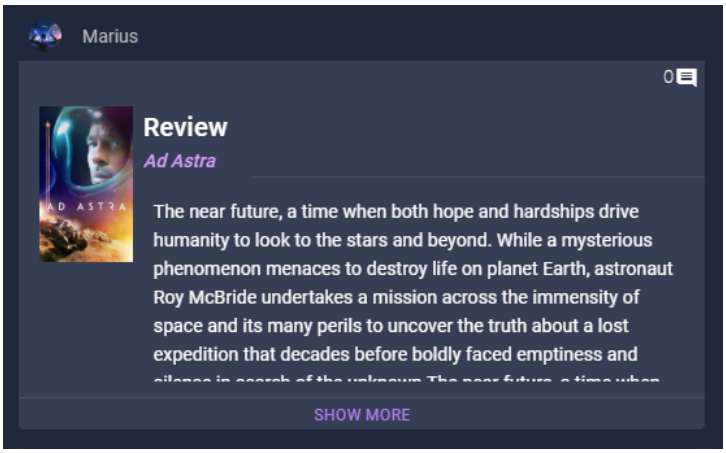
4.11 pav. Prisijungusio vartotojo visų filmų puslapio filmo kortelė

Pridėjus keletą filmų jie bus vaizduojami vartotojo profilio filmų sąraše, kuris buvo parodytas 4.5 paveikslėlyje. Prisijungęs vartotojas taip pat gali rašyti žinutes ar komentarus žinutėms filmo puslapyje arba pagrindinio puslapio naujienų juostoje. Rašant žinutę šalia taip pat atvaizduojamas jos rezultatas, o parašius ir paskelbus ji pasirodo naujienų juostoje. Jei žinutė buvo rašyta filmo puslapyje, ji dar taip pat bus ir performatuota pridedant filmo pavadinimą bei paveikslėlį. Toliau pademonstruotas žinutės rašymas (4.12 pav)., naujienų juostos žinutės išvaizda (4.13 pav). bei filmo puslapyje rašytos žinutės išvaizda (4.14 pav).



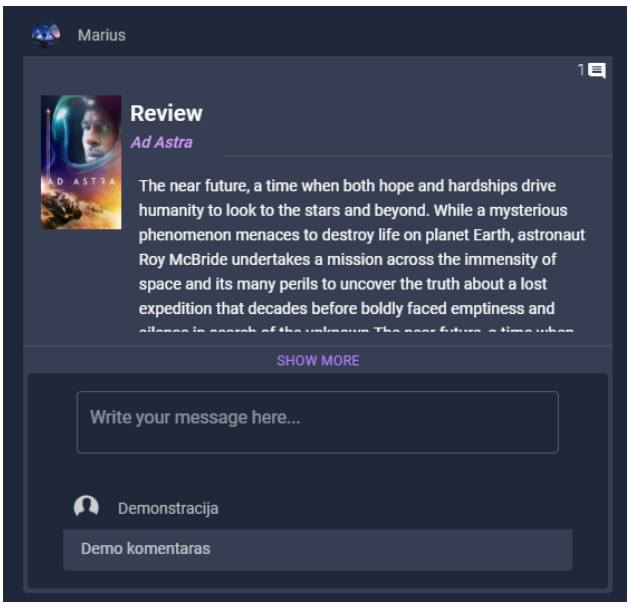
4.12 pav. Rašomos žinutės pavyzdys

4.13 pav. Parašytos žinutės naujienų juostoje pavyzdys



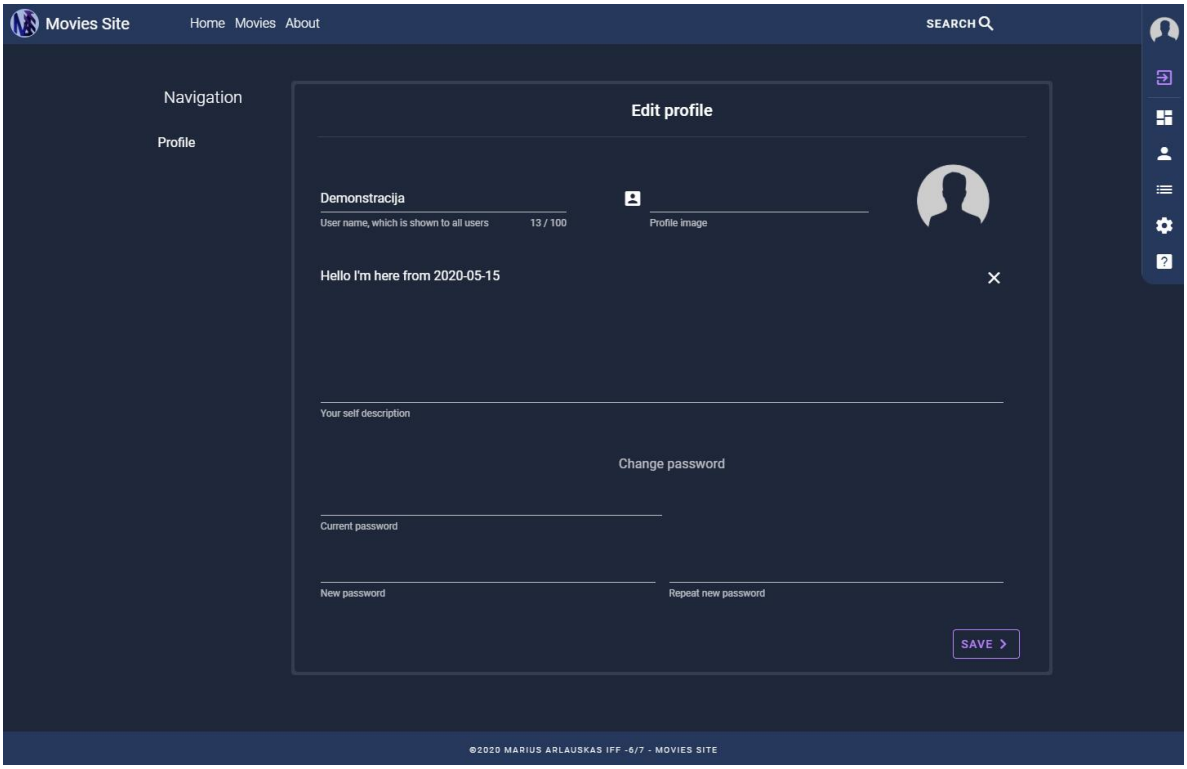
4.14 pav. Parašytos žinutės filmo puslapyje atvaizdavimas naujienų juostoje

Žinutės viršutiniame dešiniajame kampe paspaudus komentaro paveikslėlį galima matyti bei rašyti komentarus (4.15 pav.).



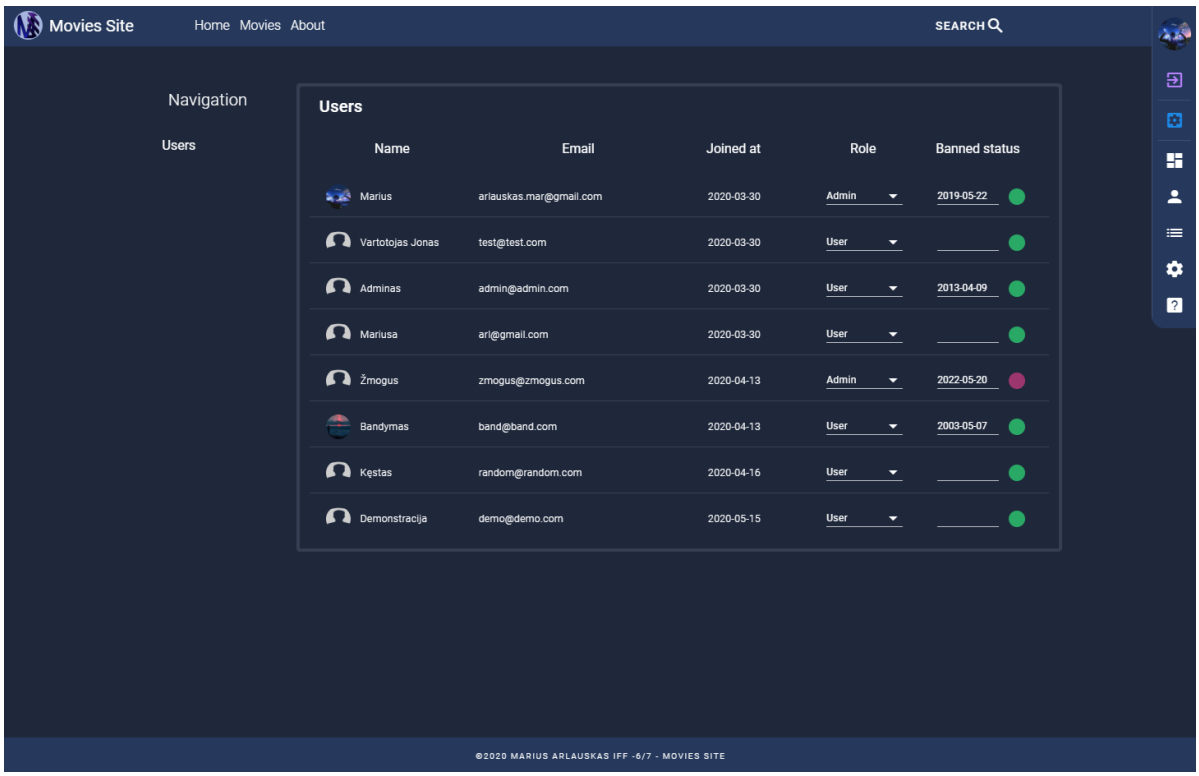
4.15 pav. Žinučių komentarai

Vartotojas taip pat gali redaguoti profilio informaciją ar pakeisti paveikslėlį (4.16 pav.) paspaudęs redagavimo mygtuką savo profilio puslapyje arba vartotojo navigacijos juostoje.



4.16 pav. Vartotojo profilio redagavimo langas

Administratorius turi papildomą funkciją blokuoti vartotojų žinučių rašymą iki tam tikros datos ar pakeisti vartotojų teises atsidaręs administratorių meniu (4.17 pav.) vartotojo navigacijos lange. Šis meniu matomas tik administratoriams.



4.17 pav. Vartotojų redagavimo langas

4.3. Diegimo vadovas

Projektas kaip jau minėta suskirstytas į dvi dalis ir darant projektą jos abi planuotos atskiruose serveriuose. Kadangi abi sistemos jau turi savyje reikalingas diegimo bibliotekas, jų diegimas yra nesudėtingas ir nereikalaujantis daug darbo.

1. Vartotojo sąsaja - *VueJs*

   Vartotojo sąsaja buvo kurta visiškai atskirai nuo API ir pati neturi didelių reikalavimų, tačiau ją reikia paruošti diegimui, kas reikalauja *Node.js* ir *npm* instaliacijų.

   Pirmiausia sistemoje *src/plugin* esančiame aplanke faile *axios.js* reikia nusistatyti sistemos API adresą, jis nustatomas parametrui *axios.defaults.baseURL* ir turėtų atrodyti taip *http://xxx.xxx.xxx.xxx:xxxx/api/*. Paruošimas vyksta projekto aplanke paleidžiant konsolės komandą *npm run build*, tačiau norint projektą paleisti lokaliai paruošimo nereikia ir pakanka vietoj šios komandos bei tolimesnių nustatymų naudoti *npm run serve* komandą. Leidžiant projektą lokaliai, dėl vartotojo žetono saugojimo, rekomenduojama naudoti adresą *http://127.0.0.1:8080/*.

   Naudojant projektų failų paruošimo komandą sukuriami projekto failai kėlimui į serverį ir sukeliami į *dist* aplanką. Tačiau serverio konfigūracija yra sudėtingesnė ir gali skirtis priklausomai nuo serverio, todėl paprasčiau būtų konfigūracijas atlikti pagal „*Example Server Configurations*“ aprašymą *Vue CLI* svetainėje [9] arba tiksliau „*Example Server Configurations*“ *Vue Router* svetainėje [10].
2. Sistemos API – *Symfony 4*

   Symfony karkasu kurtas API pasileidžia taip pat lengvai kaip ir prieš tai minėta vartotojo sąsajos dalis ir jo nereikia papildomai nustatinėti – sukėlus projektą jį paleisti reikia programos

aplanke naudojant konsolės komandą *php bin/console server:run*, tačiau taip pat reikia importuoti *MySQL* duomenų bazę bei atitikti keletą reikalavimų:

a. *PHP 7.1.3* arba didesnės versijos;
b. Įrašyti plėtiniai *Ctype, iconv, JSON, PCRE, Session, SimpleXML, Tokenizer*, kurie yra didžiojoje dalyje *PHP 7*;
c. Įrašytas *composer* [7].

Jei visi reikalavimai įgyvendinti ir duomenų bazė importuota, nusistatome jos ir vartotojo sąsajos adresus API sistemos pagrindiniame aplanke esančiame *.env* faile: *DATABASE_URL* parametras turi nurodyti duomenų bazės adresą, o *CORS_ALLOW_ORIGIN* – privalo turėti vartotojo sąsajos adresą. Gerai paruošus nustatymus, API gali būti paleidžiamas iš pagrindinio projekto aplanko naudojant komandą *php bin/console server:run*.

4.4. Administravimo vadovas

Kolkas sistemos administravimas didelio funkcionalumo neturi. Administratoriai profilio navigacijos juostoje gali pasirinkti administravimo meniu, kuriame galima valdyti vartotojus juos blokuojant nuo žinučių rašymo visoje sistemoje arba keičiant jų roles. Blokavimas atliekamas pasirenkant datą, iki kurios vartotojas bus užblokuotas. Vartotojo pusėje blokavimas pasireiškia vartotojams vietoj žinutės rašymo mygtuko atsirandant pranešimui, kad vartotojas užblokuotas iki administratoriaus nurodytos datos.

Administravime dar planuojama pridėti nemažą dalį funkcionalumo, tokio kaip įdiegtų API valdymas, vartotojų žinučių blokavimas arba atblokavimas, vartotojų elgesio žymėjimas, globalių žinučių rašymas ir dar daugiau.

Rezultatai ir išvados

Atlikus darbą buvo padarytos tokios išvados:

1. Panašių sistemų bei konkurentų analizės metu pastebėta, kad nėra tokių sistemų, kurios ne tik siūlytų daugybę informacijos apie filmus bet ir suteiktų galimybę vartotojams nesudėtingai pasialinti atrastais filmais su kitais. Nors dalis ištirtų konkurentų ir turėjo panašių bruožų, jų nepakanka pilnavertiškam vartotojų bendravimui. Būtent dėl šios priežasties ir buvo pasirinkta kurti tokią sistemą.
2. Kuriant vartotoją sąsają atskirai nuo API bei naudojant viešą API duomenų gavimui susipažinta ir išmokta apie sistemų adresų konfigūracijas bei jų tarpusavio bendravimą. Išmokti *Symfony* bei *VueJs* karkasai, bei pastebėta, kad pasirinkus populiarius karkasus ir atsiradus problemai darbe, lengva Google pagalba atrasti problemų bei efektyviai ją išspręsti, kad padidina viso projekto darbo našumą.
3. Panaudotas iteracinis kūrimo procesas leido kuriant svetainės puslapius atsižvelgti į tai ko juose trūksta ir kokių papildomų funkcijų būtų galima pridėti. Taip pat, sukūrus komponentus vienuose puslapiuose, kituose buvo labiau atkreipiamas dėmesys į tai kur jie dar tikrų, taip stengiantis pernaudoti jau esamą kodą.
4. Pastebėta, kad sistemos testavimo procesas užima nemažą dalį laiko. Atlikus automatinius testavimus pastebėta, kad testuojant sistemos kontrolerius reikia pamėgdžioti nemažą dalį sistemos elementų, todėl jei tai nėra privaloma ir komponentas nėra vienas iš pagrindinių ar svarbesnių komponentų, jų testavimas rankiniu būdu yra našesnis.
5. Kuriant sistemos dokumentaciją buvo labiausiai atsižvelgta į sistemos vartotojus, kadangi didžioji dalis sistemos funkcionalumo ir pati sistemos idėja sutelkta į būtent juos. Dėl šitos priežasties beveik visi svetainės teikiami privalumai ir prieinami nemokamai bei nemaža jų dalis ir net neprisiregistravus prie svetainės. Dokumentacijoje mažiau dėmesio buvo skirta administravimui ir sistemos diegimui, kadangi projektai buvo sukurti su diegimui padedančiomis bibliotekomis, kurie šį procesą gerokai palengvina.
6. Padaryta išvada, kad projektui skirto laiko nepakako ir visos projekto idėjos bei norimi pateikti funkcionalumai nebuvo įgyvendinti, todėl projektas dar bus plečiamas jam skiriant dar daug daugiau laiko. Taip pat dėl šios priežasties testuojant sistemą, tiksliau kodo analizės metu, buvo rasta klaidų, kurios negali būti ištaisytos, nes kaip ir minėta, jos atvaizduoja dalį nebaigto įgyvendinti funkcionalumo.

Literatūros sąrašas

1. [1] **Watson, Amy.** Preferred movie watching locations among adults in the United States as of February 2018. *Statista.* [Online] 25 spalis 2019. [Cited: 17 gegužė 2020.] https://www.statista.com/statistics/264399/preferred-place-of-movie-consumption-in-the-us/.

2. [2] —. Increased time spent on media consumption due to the coronavirus outbreak among internet users worldwide as of March 2020, by country. *Statistika.* [Tinkle] 2020 m. balandis 30 d. [Cituota: 2020 m. gegužė 17 d.] https://www.statista.com/statistics/1106766/media-consumption-growth-coronavirus-worldwide-by-country/.

3. [3] **IMDb.** Help center. *IMDb.* [Tinkle] 1990-2020 m. [Cituota: 2020 m. gegužė 17 d.] https://help.imdb.com/article/imdb/general-information/what-is-imdb/G836CY29Z4SGNMK5?ref_=helpart_nav_1#.

4. [4] **iCheckMovies.** Tour. *iCheckMovies.* [Tinkle] 2009-2020 m. [Cituota: 2020 m. gegužė 17 d.] https://www.icheckmovies.com/help/tour/.

5. [5] **Filmsomaniac.** Welcome. *Filmsomaniac.* [Tinkle] 2013-2020 m. [Cituota: 2020 m. gegužė 17 d.] https://www.filmsomniac.com/welcome.

6. [6] **S.R.L., S.C Evercoder Software.** *Moqups.* [Tinkle] S.C Evercoder Software S.R.L., 2020 m. [Cituota: 2020 m. gegužė 17 d.] https://moqups.com/.

7. [7] **GitHub, Inc.** *GitHub.* [Tinkle] GitHub, Inc, 2020 m. [Cituota: 2020 m. gegužė 17 d.]

8. [8] **Symfony™.** Installing & Setting up the Symfony Framework. *Symfony.* [Tinkle] Symfony™. [Cituota: 2020 m. gegužė 17 d.] https://symfony.com/doc/4.4/setup.html.

9. [9] **Pichler, Manuel.** PHPMD - PHP Mess Detector. *PHPMD.* [Tinkle] 2009 m. gruodis 20 d. [Cituota: 2020 m. gegužė 17 d.] https://phpmd.org/about.html.

10. [10] **Wu, Pine.** Vetur. *Marketplace - Visual Studio.* [Tinkle] Microsoft, 2019 m. [Cituota: 2020 m. gegužė 17 d.] https://marketplace.visualstudio.com/items?itemName=octref.vetur.

11. [11] **Bergmann, Sebastian.** Welcome to PHPUnit! *PHPUnit.* [Tinkle] 2018 m. [Cituota: 2020 m. gegužė 17 d.] https://phpunit.de/.

12. [12] **You, Evan.** Deployment. *Vue CLI.* [Tinkle] 2020 m. kovas 31 d. [Cituota: 2020 m. gegužė 17 d.] https://cli.vuejs.org/guide/deployment.html#routing-with-history-pushstate.

13. [13] —. HTML5 History Mode. *Vue Router.* [Tinkle] [Cituota: 2020 m. gegužė 17 d.] https://router.vuejs.org/guide/essentials/history-mode.html#example-server-configurations.

Priedai

1 priedas. Sistemos API specifikacija

4.1 lentelė. Vartotojo prisijungimo API metodo dokumentacija

Kelias	/api/login_check		
HTTP metodas	POST		
Paskirtis	Vartotojo prisijungimas		
Reikalingas žetonas	Ne		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
username	Taip	String	Vartotojo el. paštas
password	Taip	String	Vartotojo slaptažodis
Atsakymo kodas	**Apibūdinimas**		
200	Vartotojas prijungiamas		
401	Neteisingi įgaliojimai (blogi prisijungimo duomenys).		
Kodo 200 atsakymo duomenų struktūros pavyzdys	Duomenų grįžtant atsakymui negaunama, tačiau vartotojo naršyklėje išsaugomas „Cookie“ su vartotojo žetonu		

4.2 lentelė. Vartotojo atsijungimo API metodo dokumentacija

Kelias	/api/logout
HTTP metodas	POST
Paskirtis	Vartotojo atsijungimas
Reikalingas žetonas	Taip
Atsakymo kodas	**Apibūdinimas**
200	Vartotojo prisijungimo žetonas pripažįstamas negaliojančiu
401	Neteisingi įgaliojimai

4.3 lentelė. Filmų žanrų gavimo API metodo dokumentacija

Kelias	/api/genres
HTTP metodas	GET
Paskirtis	Filmų žanrų gavimas
Reikalingas žetonas	Ne
Atsakymo kodas	**Apibūdinimas**
200	Grąžinami filmų žanrai
404	Jei nerasta nė vieno žanro
Kodo 200 atsakymo duomenų struktūros pavyzdys	[{ "id":1,

	"apiId":1, "name":"Action", "genreId":28 }]

4.4 lentelė. Žinutės paskelbimo API metodo dokumentacija

Kelias	/api/messages		
HTTP metodas	POST		
Paskirtis	Žinutės paskelbimas		
Reikalingas žetonas	Taip		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
message	Taip	String	Žinutės turinys
parentId	Ne	Integer	Tėvinės žinutės ID, jei žinutė yra kitos žinutės komentaras
movieId	Ne	Integer	Filmo ID, jei žinutė rašoma filmo juostoje
Atsakymo kodas	**Apibūdinimas**		
200	Žinutės paskelbimas sėkmingas		
403	Vartotojas blokuotas.		

Kodo 200 atsakymo duomenų struktūros pavyzdys	{
	"id":79,
	"userId":5,
	"movieId":4866,
	"message":"Žinutė",
	"postDate":"2020-05-14",
	"parentId":null,
	"sharedApiId":null,
	"children":[]
	}		

4.5 lentelė. Paskelbtų žinučių gavimo API metodo dokumentacija

Kelias	/api/messages/{elementNumber}/{lastId}		
HTTP metodas	GET		
Paskirtis	Paskelbtų žinučių gavimas		
Reikalingas žetonas	Ne		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
elementNumber	Taip	Integer	Žinučių puslapio numeris
lastId	Ne	Integer	Žinučių naujinimui reikalingas žinutės ID. Po nustatyto laiko naujienų juosta atnaujinama ir imami naujesni įrašai
Atsakymo kodas	**Apibūdinimas**		

200	Grąžinamos žinutės. Jei žinutė buvo paskelbta filmo juostoje ji turi ir dalį filmo duomenų
404	Žinutė ar žinutės nerastos
Kodo 200 atsakymo duomenų struktūros pavyzdys	{ "id":"76", "movieId":"419704", "message":"Žinutė", "postDate":"2020-05-03 23:34:54", "parentId":4868, "sharedApiId":2, "userName":“Vartotojo vardas", "userId":"5", "userProfilePicture":"Kelias iki vartotojo profilio paveikslėlio", "title":"Filmo pavadinimas", "posterPath":"Kelias iki filmo paveikslėlio", "children":[] },

4.6 lentelė. Filmų gavimo API metodo dokumentacija

Kelias	/api/movies/{type}/page/{pageNumber}/user/{userId} arba /api/movies/{type}/page/{pageNumber}		
HTTP metodas	GET		
Paskirtis	Filmų gavimas su vartotojo filmų ryšiais arba be		
Reikalingas žetonas	Ne		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
type	Taip	String	Filmo sąrašo pagrindinis rūšiavimo tipas
pageNumber	Taip	Integer	Puslapio numeris
userId	Ne	Integer	Vartotojo ID, kuris naudojamas prie atvaizduojamų filmų prijungti vartotojo mėgstamiausius ar kitus jo sąraše esančius filmus.
Atsakymo kodas	**Apibūdinimas**		
200	Grąžinamas filmų sąrašas		
404	Filmų nerasta		
Kodo 200 atsakymo duomenų struktūros pavyzdys	[{ "id":"1", "title":"Filmo pavadinimas", "author":" Filmo autorius", "releaseDate":"2019-09-17", "overview":" Filmo apibūdinimas", "posterPath":"Kelias iki filmo paveikslėlio", "originalTitle":" Filmo pavadinimas", "rating":"6", "apiId":"1", "movieId":"419704",		

	"genres":"Drama, Science, Fiction", "mostPopular":10, "topRated":25, "upcoming":23, "latest":52, "nowPlaying":3, "isFavorite":"0", "relationTypeId":"1", "userRating":"5" }]

4.7 lentelė. Filmo informacijos gavimo API metodo dokumentacija

Kelias	/api/movies/{id}		
HTTP metodas	GET		
Paskirtis	Filmo informacijos gavimas		
Reikalingas žetonas	Ne		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
id	Taip	Integer	Filmo ID
Atsakymo kodas	**Apibūdinimas**		
200	Gaunama filmo informacija		
404	Filmas nerastas		

Kodo 200 atsakymo duomenų struktūros pavyzdys	{
	"id":4,
	"title":" Filmo pavadinimas",
	"author":"Filmo autorius",
	"releaseDate":"2020-02-12",
	"overview":" Filmo apibūdinimas",
	"posterPath":"Kelias iki filmo paveikslėlio",
	"originalTitle":"Filmo pavadinimas kilmės kalba",
	"rating":7.5,
	"apiId":1,
	"movieId":454626,
	"genres":"Action, Comedy, Science Fiction, Family",
	"mostPopular":4,
	"topRated":6,
	"upcoming":40,
	"latest":3,
	"nowPlaying":1
	}		

4.8 lentelė. Populiariausių filmų gavimo API metodo dokumentacija

Kelias	/api/movies/webMostPopular/page/{pageNumber}/{type}/{userId}

HTTP metodas	GET		
Paskirtis	Populiariausių filmų svetainėje gavimas pagal tipą, kur tipas nusprendžia statistikos periodą		
Reikalingas žetonas	Ne		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
pageNumber	Taip	Integer	Puslapio numeris
type	Taip	String	Statistikos tipas
userId	Taip	Integer	Vartotojo ID, tam kad būtų galima atvaizduoti ar vartotojas turi šiuos filmus savo sąraše
Atsakymo kodas	**Apibūdinimas**		
200	Grąžinamas filmų sąrašas.		
404	Filmų sąrašas tuščias		

Kodo 200 atsakymo duomenų struktūros pavyzdys	[
	{
	"id":"11",
	"title":"Filmo pavadinimas",
	"author":"Filmo autorius",
	"releaseDate":"2020-01-08",
	"overview":"Filmo apibūdinimas",
	"posterPath":"Kelias iki filmo paveikslėlio",
	"originalTitle":"Filmo pavadinimas kilmės kalba",
	"rating":"6.4",
	"apiId":"1",
	"movieId":"443791",
	"genres":"Action, Horror, Science Fiction, Thriller",
	"mostPopular":"11",
	"topRated":10,
	"upcoming":8,
	"latest":4,
	"nowPlaying":5,
	"webPopularity":"3"
	}
	]		

4.9 lentelė. Filmo žinučių gavimo API metodo dokumentacija

Kelias	/api/movies/{id}/messages/{elementNumber}		
HTTP metodas	GET		
Paskirtis	Filmo peržiūrų (žinučių) gavimas		
Reikalingas žetonas	Ne		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
id	Taip	Integer	Filmo ID
elementNumber	Taip	Integer	Praleidžiamų žinučių kiekis
Atsakymo kodas	**Apibūdinimas**		

200	Gaunamas filmo žinučių sąrašas
404	Filmas arba jo žinutės nerastos

Kodo 200 atsakymo duomenų struktūros pavyzdys	[
	{
	"id":"76",
	"message":"Žinutė",
	"postDate":"2020-05-03 23:34:54",
	"parentId":5,
	"movieId":"419704",
	"sharedApiId":null,
	"userName":"Marius",
	"userId":"5",
	"userProfilePicture":"Kelias iki vartotojo profilio paveikslėlio",
	"children":[]
	}
	]

4.10 lentelė. Vartotojo sukūrimo API metodo dokumentacija

Kelias	/api/users		
HTTP metodas	POST		
Paskirtis	Vartotojo sukūrimas, kuris naudojamas registracijoje.		
Reikalingas žetonas	Ne		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
email	Taip	String	Vartotojo el. paštas
password	Taip	String	Vartotojo slaptažodis
name	Taip	String	Vartotojo vardas
birthDate	Taip	Date	Vartotojo gimimo data
Atsakymo kodas	**Apibūdinimas**		
200	Vartotojas sukurtas sėkmingai		
400	Vartotojo el. paštas užimtas		
403	Priėjimas neleidžiamas, jei veiksmas atliekamas prisijungusio vartotojo		
Kodo 200 atsakymo duomenų struktūros pavyzdys	Duomenų grįžtant atsakymui negaunama		

4.11 lentelė. Vartotojo užblokavimo API metodo dokumentacija

Kelias	/api/users/{id}/update
HTTP metodas	POST
Paskirtis	Vartotojo užblokavimas
Reikalingas žetonas	Taip

Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
Id	Taip	Integer	Vartotojo ID
chatBannedUntil	Taip	Date	Vartotojo blokavimo periodo pabaiga
Atsakymo kodas	**Apibūdinimas**		
200	Grąžinama žinutė apie sėkmingą blokavimą		
403	Neteisingi įgaliojimai		
404	Vartotojas nerastas		
Kodo 200 atsakymo duomenų struktūros pavyzdys	"User banned from chat!!"		

4.12 lentelė. Rolės pakeitimo API metodo dokumentacija

Kelias	/api/users/{id}/updateRole		
HTTP metodas	POST		
Paskirtis	Vartotojo rolės pakeitimas		
Reikalingas žetonas	Taip		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
id	Taip	Integer	Vartotojo ID
role	Taip	String	Vartotojui suteikiama rolė
Atsakymo kodas	**Apibūdinimas**		
200	Vartotojo rolė pakeista, grąžinamas atnaujintas vartotojo profilis		
403	Neteisingi įgaliojimai		
404	Vartotojas nerastas		
Kodo 200 atsakymo duomenų struktūros pavyzdys	{ "id":7, "email":"el@paštas.com", "name":"Vartotojo vardas", "profilePicture":"Kelias iki vartotojo profilio paveikslėlio", "description":"Vartotojo profilio aprašymas", "birthDate":"Vartotojo gimimo data", "registerDate":"2020-03-30", "chatBannedUntil":"2013-04-09", "role":"ROLE_USER" }		

4.13 lentelė. Visų vartotojų sąrašo gavimo API metodo dokumentacija

Kelias	/api/users
HTTP metodas	GET
Paskirtis	Visu vartotojų sąrašo gavimas
Reikalingas žetonas	Taip

Atsakymo kodas	**Apibūdinimas**
200	Grąžinamas vartotojų sąrašas
401	Neteisingi įgaliojimai

Kodo 200 atsakymo duomenų struktūros pavyzdys	[
	{
	"id":5,
	"email":"el@pastas.com",
	"name":"Vartotojo vardas",
	"profilePicture":"Kelias iki vartotojo profilio paveikslėlio",
	"description":"Vartotojo profilio aprašymas",
	"birthDate":"2000-03-02",
	"registerDate":"2020-03-30",
	"chatBannedUntil":"2019-05-22",
	"role":"ROLE_ADMIN"
	}
	]

4.14 lentelė. Filmo, vartotojo sąraše, statuso pridėjimo API metodo dokumentacija

Kelias		/api/users/{userId}/apis/{apiId}/movies/{movieId}/status/{relationType}		
HTTP metodas		POST		
Paskirtis		Filmo, vartotojo sąraše, statuso pridėjimas / pakeitimas		
Reikalingas žetonas		Taip		
Parametras		**Būtinas**	**Tipas**	**Apibūdinimas**
userId		Taip	Integer	Vartotojo ID
apiId		Taip	Integer	API, kuriame saugojamas filmas ID (padaryta nes svetainės planuose kelių filmų API palaikymas)
movieId		Taip	Integer	Filmo ID
relationType		Taip	Integer	Filmo sąraše tipo ID
Atsakymo kodas		**Apibūdinimas**		
200		Filmo sąraše tipas pridėtas arba pakeistas		
403		Neteisingi įgaliojimai		
Kodo 200 atsakymo duomenų struktūros pavyzdys		"Set users 5 movie 920 in list with status id 1"		

4.15 lentelė. Filmo, vartotojo sąraše, vertinimo priėjimo API metodo dokumentacija

Kelias	/api/users/{userId}/apis/{apiId}/movies/{movieId}/rating/{rating}

HTTP metodas	POST		
Paskirtis	Filmo, vartotojo sąraše, vertinimo pridėjimas / pakeitimas		
Reikalingas žetonas	Taip		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
userId	Taip	Integer	Vartotojo ID
apiId	Taip	Integer	API, kuriame saugojamas filmas ID (padaryta nes svetainės planuose kelių filmų API palaikymas)
movieId	Taip	Integer	Filmo ID
rating	Taip	Integer	Vartotojo skiriamas filmui vertinimas
Atsakymo kodas	**Apibūdinimas**		
200	Filmo sąraše vertinimas pridėtas arba pakeistas		
403	Neteisingi įgaliojimai		
Kodo 200 atsakymo duomenų struktūros pavyzdys	"Rated movie - 5"		

4.16 lentelė. Vartotojo filmų sąrašo gavimo API metodo dokumentacija

Kelias	/api/users/{id}/movies		
HTTP metodas	GET		
Paskirtis	Vartotojo filmų sąrašo gavimas		
Reikalingas žetonas	Ne		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
id	Taip	Integer	Vartotojo ID
Atsakymo kodas	**Apibūdinimas**		
200	Gaunamas vartotojo filmų sąrašas		
403	Neteisingi įgaliojimai (blogi prisijungimo duomenys).		
404	Vartotojas nerastas		
Kodo 200 atsakymo duomenų struktūros pavyzdys	[{ "id":"1", "title":"Filmo pavadinimas","author":"Filmo autorius", "releaseDate":"2019-09-17", "overview":"Filmo aprašymas", "posterPath":"Kelias iki filmo paveikslėlio", "originalTitle":"Filmo pavainimas gimtąja kalba", "rating":"6", "apiId":"1", "movieId":"419704", "genres":"Drama, Science Fiction" ,"mostPopular":"1", "topRated":null, "upcoming":null, "latest":null,		

	"nowPlaying":null, "isFavorite":"0", "relationTypeId":"1", "userRating":"5" }]

4.17 lentelė. Vartotojų parašytų žinučių gavimo API metodo dokumentacija

Kelias	/api/users/{id}/messages/{elementNumber}		
HTTP metodas	GET		
Paskirtis	Vartotojo parašytų žinučių gavimas		
Reikalingas žetonas	Ne		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
id	Taip	Integer	Vartotojo ID
elementNumber	Taip	Integer	Praleidžiamų elementų kiekis
Atsakymo kodas	**Apibūdinimas**		
200	Vartotojas prijungiamas		
404	Vartotojas nerastas		

Kodo 200 atsakymo duomenų struktūros pavyzdys	[
	{
	"id":"23",
	"movieId":"0",
	"message":"Žinutė",
	"postDate":"2020-04-13 22:28:44",
	"parentId":null,
	"sharedApiId":null,
	"userName":"Vartotojo vardas",
	"userId":"12",
	"userProfilePicture":"Kelias iki vartotojo profilio paveikslėlio",
	"title":null,
	"posterPath":null,
	"children":[]
	}
	]		

4.18 lentelė. Visų filmų, vartotojų sąrašuose, tipų gavimo API metodo dokumentacija

Kelias	/api/lists/types
HTTP metodas	GET
Paskirtis	Visų filmų, vartotojų sąrašuose, tipų gavimas
Reikalingas žetonas	Ne
Atsakymo kodas	**Apibūdinimas**

200	Grąžinami visi galimi vartotojo sąraše esančio filmo tipai

Kodo 200 atsakymo duomenų struktūros pavyzdys	[
	{
	"id":1,
	"name":"Planning"
	}
	]

4.19 lentelė. Prisijungusio vartotojo savojo profilio gavimo API metodo dokumentacija

Kelias	/api/profile
HTTP metodas	GET
Paskirtis	Prisijungusio vartotojo savojo profilio gavimas
Reikalingas žetonas	Taip
Atsakymo kodas	**Apibūdinimas**
200	Grąžinama vartotojo informacija
403	Neteisingi įgaliojimai

Kodo 200 atsakymo duomenų struktūros pavyzdys	{
	"id":5,
	"email":"el@paštas.com",
	"name":"Vartotojo vardas",
	"profilePicture":"Kelias iki vartotojo profilio paveikslėlio",
	"description":"Vartotojo profilio apibūdinimas",
	"birthDate":"2000-03-02",
	"registerDate":"2020-03-30",
	"chatBannedUntil":"2019-05-22",
	"role":"ROLE_ADMIN"
	}

4.20 lentelė. Vartotojo profilio gavimo API metodo dokumentacija

Kelias	/api/profile/{id}		
HTTP metodas	GET		
Paskirtis	Vartotojo profilio gavimas		
Reikalingas žetonas	Ne		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
id	Taip	Integer	Vartotojo ID
Atsakymo kodas	**Apibūdinimas**		
200	Grąžinama vartotojo informacija		
403	Neteisingi įgaliojimai		
Kodo 200 atsakymo duomenų struktūros pavyzdys	{ "id":5,		

	"email":"el@paštas.com", "name":"Vartotojo vardas", "profilePicture":"Kelias iki vartotojo profilio paveikslėlio", "description":"Vartotojo profilio apibūdinimas", "birthDate":"2000-03-02", "registerDate":"2020-03-30", "chatBannedUntil":"2019-05-22", "role":"ROLE_ADMIN" }

4.21 lentelė. Prisijungusio vartotojo profilio reedagavimo API metodo dokumentacija

Kelias	/api/profile/{id}/update		
HTTP metodas	POST		
Paskirtis	Prisijungusio vartotojo profilio redagavimas		
Reikalingas žetonas	Taip		
Parametras	**Būtinas**	**Tipas**	**Apibūdinimas**
id	Taip	Integer	Vartotojo ID
name	Ne	String	Vartotojo vardas
description	Ne	Text	Vartotojo profilio apibūdinimas
password	Ne	String	Vartotojo slaptažodis
Atsakymo kodas	**Apibūdinimas**		
200	Grąžinamas atnaujintas vartotojo profilis		
403	Neteisingi įgaliojimai		
404	Vartotojas nerastas		

Kodo 200 atsakymo duomenų struktūros pavyzdys	{
	"id":5,
	"email":"el@paštas.com",
	"name":"Vartotojo vardas",
	"profilePicture":"Kelias iki vartotojo profilio paveikslėlio",
	"description":"Vartotojo profilio apibūdinimas",
	"birthDate":"2000-03-02",
	"registerDate":"2020-03-30",
	"chatBannedUntil":"2019-05-22",
	"role":"ROLE_ADMIN"
	}		