****

**Kauno technologijos universitetas**

Informatikos fakultetas

**Baigiamojo projekto pavadinimas**

Baigiamasis bakalauro studijų projektas

|  |
| --- |
|  |
| **Vardenis Pavardenis**  Projekto autorius / autorė |
|  |
| **Pareigų sutrumpinimas Vardenis Pavardenis**  Vadovas / Vadovė |
|  |

**Kaunas, 2019**

****

**Kauno technologijos universitetas**

Informatikos fakultetas

**Baigiamojo projekto pavadinimas**

Baigiamasis bakalauro studijų projektas

Programų sistemos (612I30002)

|  |  |
| --- | --- |
|  |  |
| **Vardenis Pavardenis**  Projekto autorius / autorė | (parašas) (data) |
|  |  |
| **Pareigų sutrumpinimas Vardenis Pavardenis**  Vadovas / Vadovė | (parašas) (data) |
|  |  |
| **Pareigų sutrumpinimas Vardenis Pavardenis**  Recenzentas / Recenzentė | (parašas) (data) |
|  |  |

**Kaunas, 2019**

****

**Kauno technologijos universitetas**

Informatikos fakultetas

Vardenis Pavardenis

**Baigiamojo projekto pavadinimas**

Akademinio sąžiningumo deklaracija

Patvirtinu, kad mano, Vardenio Pavardenio, baigiamasis projektas tema „Baigiamojo projekto pavadinimas“ 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) |

Autoriaus pavardė, vardas. Baigiamojo projekto pavadinimas. Bakalauro studijų baigiamasis projektas / vadovas / vadovė pareigų sutrumpinimas Vardenis Pavardenis; Kauno technologijos universitetas, Informatikos fakultetas.

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

Reikšminiai žodžiai: ................................(įrašykite).

Kaunas, 2019. XX p.

Santrauka

Lorem ipsum dolor sit amet, eam ex decore persequeris, sit at illud lobortis atomorum. Sed dolorem quaerendum ne, prompta instructior ne pri. Et mel partiendo suscipiantur, docendi abhorreant ea sit. Recteque imperdiet eum te.

Eu eum decore inimicus consetetur, cu usu habeo corpora intellegam. Ut antiopam efficiendi deterruisset sit. Mel sint eirmod id, qui quot virtute id, dolor nemore forensibus usu id. Fugit dolore voluptatum cu vim. An vix veniam graecis insolens, sit posse iusto id. Ut vim ceteros percipit, id quo ubique recusabo, eum sint lucilius ea. In sumo inani numquam has.

Author’s surname, name. Title of the Final Degree Project. Bachelor's  Final Degree Project / supervisor abbreviation of the position, name and surname of the supervisor; Informatics Faculty, Kaunas University of Technology.

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

Keywords: ................................ (type here).

Kaunas, 2019. Number of pages.

Summary

Lorem ipsum dolor sit amet, eam ex decore persequeris, sit at illud lobortis atomorum. Sed dolorem quaerendum ne, prompta instructior ne pri. Et mel partiendo suscipiantur, docendi abhorreant ea sit. Recteque imperdiet eum te.

Eu eum decore inimicus consetetur, cu usu habeo corpora intellegam. Ut antiopam efficiendi deterruisset sit. Mel sint eirmod id, qui quot virtute id, dolor nemore forensibus usu id. Fugit dolore voluptatum cu vim. An vix veniam graecis insolens, sit posse iusto id. Ut vim ceteros percipit, id quo ubique recusabo, eum sint lucilius ea. In sumo inani numquam has.

Turinys

[Lentelių sąrašas 8](#_Toc38460360)

[Paveikslų sąrašas 9](#_Toc38460361)

[Santrumpų ir terminų sąrašas 10](#_Toc38460362)

[Įvadas 11](#_Toc38460363)

[1. Analizė 12](#_Toc38460364)

[1.1. Techninis pasiūlymas 12](#_Toc38460365)

[1.1.1. Sistemos apibrėžimas 12](#_Toc38460366)

[1.1.2. Bendras veiklos tikslas 12](#_Toc38460367)

[1.1.3. Sistemos pagrįstumas 12](#_Toc38460368)

[1.1.4. Konkurencija rinkoje 12](#_Toc38460369)

[1.1.5. Prototipai ir pagalbinė informacija 14](#_Toc38460370)

[1.1.6. Ištekliai, reikalingi sistemai sukurti 14](#_Toc38460371)

[1.2. Galimybių analizė 14](#_Toc38460372)

[1.2.1. Techninės galimybės 14](#_Toc38460373)

[1.2.2. Vartotojų pasiruošimo analizė 15](#_Toc38460374)

[2. Projektas 16](#_Toc38460375)

[2.1. Reikalavimų specifikacija 16](#_Toc38460376)

[2.1.1. Komercinė specifikacija 16](#_Toc38460377)

[2.1.2. Sistemos funkcijos 16](#_Toc38460378)

[2.1.3. Vartotojo sąsajos specifikacija 22](#_Toc38460379)

[2.1.4. Realizacijai keliami reikalavimai 22](#_Toc38460380)

[2.1.5. Techninė specifikacija 23](#_Toc38460381)

[2.2. Projektavimo metodai 23](#_Toc38460382)

[2.2.1. Projektavimo valdymas ir eiga 23](#_Toc38460383)

[2.2.2. Projektavimo technologija 24](#_Toc38460384)

[2.2.3. Programavimo kalbos, derinimo, automatizavimo priemonės, operacinė sistemos 24](#_Toc38460385)

[2.3. Sistemos projektas 24](#_Toc38460386)

[2.3.1. Statinis sistemos vaizdas 24](#_Toc38460387)

[2.3.2. Dinaminis sistemos vaizdas 29](#_Toc38460388)

[3. Testavimas 49](#_Toc38460389)

[3.1. Testavimo planas 49](#_Toc38460390)

[3.2. Testavimo kriterijai 49](#_Toc38460391)

[3.3. Komponentų testavimas 49](#_Toc38460392)

[3.3.1. Automatinis testavimas 49](#_Toc38460393)

[3.3.2. Rankinis API testavimas 51](#_Toc38460394)

[3.4. Statinė kodo analizė 53](#_Toc38460395)

[3.5. Vartotojo sąsajos testavimas 54](#_Toc38460396)

[4. Dokumentacija naudotojui 57](#_Toc38460397)

[4.1. Apibendrintas sistemos galimybių aprašymas. 57](#_Toc38460398)

[4.2. Vartotojo vadovas 57](#_Toc38460399)

[Rezultatai ir išvados 58](#_Toc38460400)

[Literatūros sąrašas 59](#_Toc38460401)

[Priedai 60](#_Toc38460402)

[1 priedas. Priedo pavadinimas 60](#_Toc38460403)

Lentelių sąrašas

[**1 lentelė.** Konkurentų apžvalga 13](#_Toc3716736)

[**2 lentelė.** Pagrindiniai baigiamojo projekto stiliai ir jų aprašymai 22](#_Toc3716737)

Paveikslų sąrašas

[**1 pav.** Sistemos panaudojimo atvejų diagrama 15](#_Toc3716738)

Santrumpų ir terminų sąrašas

**Santrumpos:**

Doc. – docentas;

Lekt. – lektorius;

Prof. – profesorius.

**Terminai:**

**Saityno analitika** – lorem ipsum dolor sit amet, eam ex decore persequeris, sit at illud lobortis atomorum. Sed dolorem quaerendum ne, prompta instructior ne pri. Et mel partiendo suscipiantur, docendi abhorreant ea sit. Recteque imperdiet eum te.

**Tinklaraštis** – lorem ipsum dolor sit amet, eam ex decore persequeris, sit at illud lobortis atomorum. Sed dolorem quaerendum ne, prompta instructior ne pri. Et mel partiendo suscipiantur, docendi abhorreant ea sit. Recteque imperdiet eum te.

Beje, darbe rekomenduojame pateikti tik svarbesnes ir mažiau žinomas santrumpas bei terminus (tarkime tokių santrumpų kaip HTML, PC, IT paaiškinti nereikia)

Įvadas

Modernioje visuomenėje dėl vis didėjančio gyvenimo tempo tampa vis sunkiau planuoti ir paskirstyti darbus. Atlikti paskirtas užduotis, neatsilikti nuo darbo grafiko padeda išankstinis planavimas. Be tokio planavimo neretai atidedami svarbūs, net būtini, darbai, o vėlavimas juos atlikti žmogui sukelia stresą, kenkiantį jo fizinei ir psichologinei būklei, o įmonėms - grėsmę patirti nuostolius.

Mano sistemos tikslas išspręsti problemas, susijusias su laiko planavimu, paversti jį patogesniu, taip taupant vartotojo laiką. Norėdamas padėti išspręsti šią problemą sugalvojau sukurti programinę įrangą, kuri būtų patogi vartotojui, bei neperkrauta nereikalingo funkcionalumo. Tai bandysiu pasiekti atlikęs šios uždavinius:

1. Atlikti analizę
2. Palyginti kuriamą sistemą su konkurentais.
3. Sudaryti kuriamos sistemos projektą
4. Realizuoti sistemą
5. Ištestuoti sistemą
6. Parengti dokumentaciją

Darbas struktūrą sudaro šios pagrindinės dalys:

1. Analizė

Apžvelgiama į problemą, kuri bus išspręsta sukūrus sistemą. Apžvelgiami rinkoje jau esantys konkurentai, bandoma ieškoti kuriamos sistemos pranašumo.

1. Projektavimas

Šioje dalyje pristatomas projektas, jam iškelti reikalavimai, projektavimo metodika bei kūrimui naudotos technologijos.

1. Dokumentacija

Šioje dalyje pateikiama API specifikacija, vartotojo, diegimo bei administravimo vadovai.

1. Testavimas

Šioje dalyje matomas parengtas testavimo planas, pristatomas jo atlikimas ir gauti rezultatai.

Sistemos apimtis:

Bendras kodo eilučių kiekis : ~5700

* Testavimui ~1100
* Back-end-2600
* Front-end ~2000

Puslapių skaičius : 9

API metodų skaičius: 27

# Analizė

Šiame skyriuje bus pateikta darbo problematika susijusios informacijos analizė.

## Techninis pasiūlymas

### Sistemos apibrėžimas

Laiko planavimo sistema tai interneto svetainė skirta visiems kas turi naršyklę, bei moka ja naudotis. Sistema turėtų sudaryti šios posistemės.

1. Pakvietimų posistemė
2. Autentifikavimo posistemė
3. Tvarkaraščio posistemė
4. Pranešimų posistemė
5. Nustatymų posistemė

Laiko planavimo sistema padeda :

* Žmonėms lengviau planuoti veiklas
* Įmonėms lengviau organizuoti darbo procesą, paskirstyti darbus darbuotojams
* Prioretizuoti darbus.

### Bendras veiklos tikslas

Padėti lengviau susiplanuoti, paskirstyti darbus tiek žmonėms, tiek ir įmonėms. Taip pat vienas iš svarbiausių sistemai iškeltų tikslų būtų paprastumas, kad kiekvienas sugebėtų išmokti naudotis per nedidelį laiko tarpą.

### Sistemos pagrįstumas

Žmonės visada skuba, ir laiko taupymas jiems padeda nuveikti daugiau. Tai pagrindinė priežastis kodėl kuriama sistema yra reikalinga. Nesvarbu ar esi paprastas vartotojas, kuriam patogiau jau ryte turėti veiklų tvarkaraštį, nes tai padaro jo dieną produktyvesnę, ar žmogus atsakingas už kitų žmonių atliktus darbus ši sistema sumažins tau problemų dėliojant grafiką. Sudėtingas skaičiavimo algoritmas atsižvelgs į tavo nustatytus prioritetus ir taip iškels svarbiausius darbus į priekį.

### Konkurencija rinkoje

Mano pasirinkta sritis yra stipriai konkurencinga, kadangi joje jau yra nemažai šia veikla užsiimančių organizacijų. Bet daugumos sistemų funkcionalumas skirtingas. Toliau bus pateikta keletas pavyzdžių. Aptarsiu po 4 kiekvienos programos pagrindines funkcijas.

* **Appointy**

Šis produktas skirtas užsakymų registravimui.

Pagrindinės funkcijos:

1. Darbo valandų nustatymas
2. Pasirenkami laiko intervalai
3. Skirtingi tvarkaraščių tipai turintys skirtingo funkcionalumo.
4. Palaikomas Google, tai leidžia pritraukti daugiau klientų socialiniuose tinkluose.

* **Caledly**

Sistema skirta tvarkaraščio sudarymui.

1. Integracijos su skirtingomis programomis.
2. Susitikimų organizavimas
3. Nustatymai leidžiantys apsaugoti nuo paskutinės minutės renginių ir panašių nepageidaujamų dalykų.
4. Tvarkaraščiai pritaikomi tiek individualiam, tiek komandiniam darbui.

* **Fantastical 2**

Renginių, darbų organizavimo sistema. Iš pirmo žvilgsnio sistema nesudėtinga, bet turinti nemažai naudingo ir ne visai funkcionalumo.

Pagrindinės funkcijos:

1. Keli skirtingi kalendoriai
2. Pranešimų siuntimas
3. Veiklų ir užduočių pridėjimas, šalinimas.
4. Pakvietimai.

Išskirtinis dalykas, kurį galima butų panaudoti kuriamoje sistemoje yra susitikimų planavimas. Kviečiami žmonės patvirtina, jog dalyvaus. Tuomet visiems išrenka bendrą, tinkamą darbo laiką.

* **Sling**

Tai pati panašiausia sistema į mano kuriamą. Pagrindinis tikslas – darbuotojų tvarkaraščio sudarymas. Pagrindinis funkcionalumas:

1. Darbo pamainų sudarymas.
2. Savaitinių darbo grafikų sudarymas
3. Darbuotojų grafikų susikirtimai (konfliktai)
4. Šabloninių tvarkaraščių sudarymas, su numatytu valandų ar pamainų kiekiu.

Išskirtinumas yra, kad darbuotojai grafikus gali rinktis patys.

**1.1** **lentelė.** Konkurentų apžvalga

| **Lyginimo kriterijai** | Appointy | Caledly | Fantastical 2 | Sling | **Mano kuriama sistema** |
| --- | --- | --- | --- | --- | --- |
| Kaina | 0-79.99$ per mėnesį | 0-12$ per mėnesį | 0-4.99$ per mėnesį | 0-4$ per mėnesį | Nemokama |
| Naudojimas | Asmeninimas / komandai | Asmeninimas/komandai | Asmeninimas / komandai | Asmeninimas / komandai | Asmeninimas/komandai |
| Darbo valandų nustatymas | Taip | Ne | Taip | Ne | Taip |
| Sinchronizacija su kitomis programomis | Taip | Taip | Taip | Taip | Ne |
| Darbuotojų žymėjimas spalvomis | Ne | Taip | Taip | Taip | Ne |
| Vartotojų kiekis | Priklauso nuo kainos, galima pildyti už papildomą mokestį | Priklauso nuo kalendorių skaičiaus ir kainos | Nenurodyta | Priklausomai nuo perkamo plano | Nėra apribojimų |
| Priminimai | Ne | Ne | Taip | Ne | Taip |
| Pakvietimai | Iš dalies taip | Ne | Taip | Ne | Taip |
| Funkcionalumo kiekis | Specializuotas | Specializuotas | Specializuotas, platus | Specializuotas | Specializuotas, taikoma konkretiems vartotojams, siauras |

Buvo pasirinkti kriterijai, kurie atspindi numatomam sistemos vartotojui reikalingus funkcionalumas. Iš šių kriterijų matome, jog kuriama nauja sistema turi ir privalumų prieš jau esančias rinkoje. Pagrindinis jų, nereikalingo funkcionalumo nebuvimas, kaip daugumoje minėtų sistemų.

### Prototipai ir pagalbinė informacija

Produktas buvo kuriamas nuo pradžių, jokiais prototipais naudojamasi nebuvo.

### Ištekliai, reikalingi sistemai sukurti

Sistemą kūrimas užtuko apie 2 mėnesius. Tai pat dar 1 mėnesis buvo skirtas dokumentacijos rengimui, į kurią įėjo dokumentacijos rengimas, bei vartotojų gido parengimas. Norint tiksliau plėsti sistemą reikalinga vartotojų poreikių analizė dėl papildomo ar nereikalingo funkcionalumo. Jei funkcionalumas didėtų, gali tekti ir praplėsti personalą. Augant sistemai gali prireikti ir administratoriaus.

## Galimybių analizė

### Techninės galimybės

Sistemai realizuoti techninių kliūčių nėra. .Net Core technologija yra labai plačiai naudojama visame pasaulyje. Tai pat Angular karkasas yra vienas iš populiariausių ir dažniausiai naudojamų Javascript karkasų pasaulyje.

Vienintelė didesnė problema itin sudėtingas algoritmas skirtas darbų išdėstymui pagal laiką. Norint jį patobulinti yra tikimybė jo kodo eilučių skaičius išaugtų kelis kartus. Taip pat jo sudėtingumas padidėtų daug kartų. Todėl pasirinkta supaprastina, tačiau dalinai funkcionuojanti jo versija.

Sukurti pilnai veikiančią sistemą per tokį trumpą laiko tarpą yra gan sudėtinga užduotis. Todėl tai tik būsimos sistemos prototipas, pilnai funkcionuojantis su daug galimų tobulinimo sričių.

### Vartotojų pasiruošimo analizė

Vartotojas gali būti tiek individualus asmuo, tiek ir įmonė. Paruošimas projekto naudojimu neturėtų būti sudėtingas. Reikia užpildyti įvedimo formas, o jose jau paaiškinta kokius duomenis reikia įvesti. Taip pat stengiamasi jog visi paaiškinimai vartotojui nereikėtų ilgai galvoti. Taigi būtinas tik bendras vartotojo pasiruošimas naudotis kompiuteriu, naršykle bei skaityti paaiškinimus.

# Projektas

Šiame skyriuje aprašoma detali sistemos specifikacija.

## Reikalavimų specifikacija

### Komercinė specifikacija

Projektas sukurtas specialiai baigiamajam bakalauriniam darbui. Projekto užsakovas ir vadovas doc. Sigitas Drąsutis. Visus darbus, tiek programavimą, projektavimą, dokumentavimą. tiek ir rinkos analizę atliko studentas Mantvydas Razulevičius.

Biudžetas nėra planuojamas, kadangi tai ne komercinis projektas. Tik projekto sėkmės atveju planuojamos apmokestintos paslaugos. Sistemos naudotojai būtų tiek paprasti žmonės visame pasaulyje (reikia mokėti anglų kalbą, nes pradinėje versijoje jų daugiau nėra), tiek ir įmonės.

Trukmė pilnam paruošimui (programavimui ir dokumentacijai) apie 3 mėnesius. Pradžia – 2020 sausio 26d. Tiksli pabaigos data nėra numatyta, tačiau gegužės pradžioje projektas jau turi būti paruoštas pristatymui.

### Sistemos funkcijos

Sistemos reikalavimai išskirstomi į šias pagrindines posistemes :

1. Pakvietimų posistemė
2. Autentifikavimo posistemė
3. Tvarkaraščio posistemė
4. Pranešimų posistemė
5. Nustatymų posistemė

Toliau bus pateikta visi funkciniai reikalavimai, bet kiekvienos posistemės analizė bus pateikta atskirai.



2.1 pav. Sistemos panaudojimo atvejų diagrama

Pradžioje pateikiam diagrama (2.1), kuri parodo visą bendrą sistemos atliekamas funkcijas. Diagramoje aiškiai matyti visus sistemos veiklai būtinos dalys, bendras jų funkcionalumas. Visos posistemės turi susijusio funkcionalumo, pavyzdžiui pranešimų posistemė yra naudojama daugumoje kitų sistemos dalių.



Pav 2.2 Autentifikavimo posistemė

Autentifikavimo posistemė – viena svarbiausių visoje sistemoje. Ji užtikrina, kad vartotojas matytų tik savo informaciją, pavyzdžiui darbus, tvarkaraštį. Taip vartotojas gali matyti įvykdęs prisijungimą prie sistemos . Kita svarbi funkcija registracija. Jos forma panaši į daugelį šiuo metu internete egzistuojančių formų. Taip pat svarbu paminėti, jog po atsijungimo vartotojas nebegali pasiekti savo informacijos iki tol kol vėl prisijungs.



Pav 2.3 Darbų posistemė

Darbų posistemė – tai dėl ko buvo kuriamas šis projektas. Darbų sąrašas, naujų užduočių pridėjimas ir šalinimas. Kurdamas naują darbą vartotojas turi galimybę jį prioretizuoti. Redagavimas atliekamas iš sąrašo pasirenki viena darbą ir pakeiti jo informaciją, bei turi galimybę keisti ir prioritetą. Informacijos peržiūra atliekama prieš jį redaguojant, arba gavus pakvietimą atlikti darbą.



Pav 2.4 Nustatymų posistemė

Nustatymai yra svarbi sistemos dalis kadangi tai leidžia vartotojui patogiau jaustis sistemoje. Pavyzdžiui kai nustatomas puslapio dydis 10 elementų, tokiu atveju visoje sistemoje kur yra pasiskirstymas puslapiai bus atsižvelgta į prieš tai nustatytą reikšmę. Darbo pražios ir pabaigos nustatymai leis atsižvelgi į vartotojo norus ir apskaičiuoti darbams tinkamas vieta pagal šiuos laiko rėžius. Jei vartotojas mano, jog paskyra jam nereikalinga, jam suteikiama teisė ją pašalinti.



Pav 2.5 Pakvietimų posistemė

Pakvietimų posistemė skirta paprasčiau dalintis darbais. Vartotojui galimą išsiūti kvietimą kitam sistemoje registruotam asmeniui. Kvietimą gavęs asmuo gali jį priimti arba atmesti. Kaip ir visus čia turėtu būti filtras skirtas palengvinti reikalingų duomenų suradimą.



Pav 2.6 Pranešimų posistemė

Pranešimų posistemė – skirta vartotojo informavimui apie tam tikrus įvykius. Pagrindinis funkcionalumas yra atliekamas sistemos, kuri kuria pranešimus ir perduoda juos vartotojui. Vartotojas čia neturi daug funkcijų, tačiau gali peržiūrėti ir pašalinti pranešimus. Taip pat kaip ir daugelyje posistemių yra pranešimų filtravimo galimybė.



Pav 2.7 Tvarkaraščio posistemė

### Vartotojo sąsajos specifikacija

Vartotojo sąsajai reikalavimai sudaryti nebuvo.

### Realizacijai keliami reikalavimai

Realizacijai gali būti keliami tokie **nefunkciniai reikalavimai**: reikalavimai sistemos išvaizdai, reikalavimai panaudojamumui, reikalavimai vykdymo charakteristikoms, reikalavimai veikimo sąlygoms, reikalavimai sistemos priežiūrai, reikalavimai saugumui, kultūriniai-politiniai reikalavimai, teisiniai reikalavimai. Jie išvardinami ir aprašomi šiame skyrelyje. Pavyzdžiui:

1. Maisto perdavimo protokolas privalo būti saugus (neautentifikuoti kaimynai negali sužinoti kokį maistą siunčiasi vartotojas)
2. Maistas, suskaidytas paketais, turi pasiekti vartotoją nepagedęs
3. Maisto perdavimo protokolas turėtų palaikyti lietuviškos virtuvės produktus

Šiame punkte gali būti išvardinti (jeigu nustatyti) tokio tipo apribojimai: apribojimai sprendimui, diegimo aplinka, bendradarbiaujančios sistemos, komerciniai specializuoti programų paketai, numatoma darbo vietos aplinka, sistemos kūrimo terminai, sistemos kūrimo biudžetas. Jei reikalinga specifinė duomenų kontrolė (kokia informacija turi būti tikrinama įvedimo ar sistemos veikimo metu), ji taip pat aprašoma šiame skyrelyje.

### Techninė specifikacija

Norint savo kompiuteryje pasileisti šį projektą, reikalinga programinė įranga:

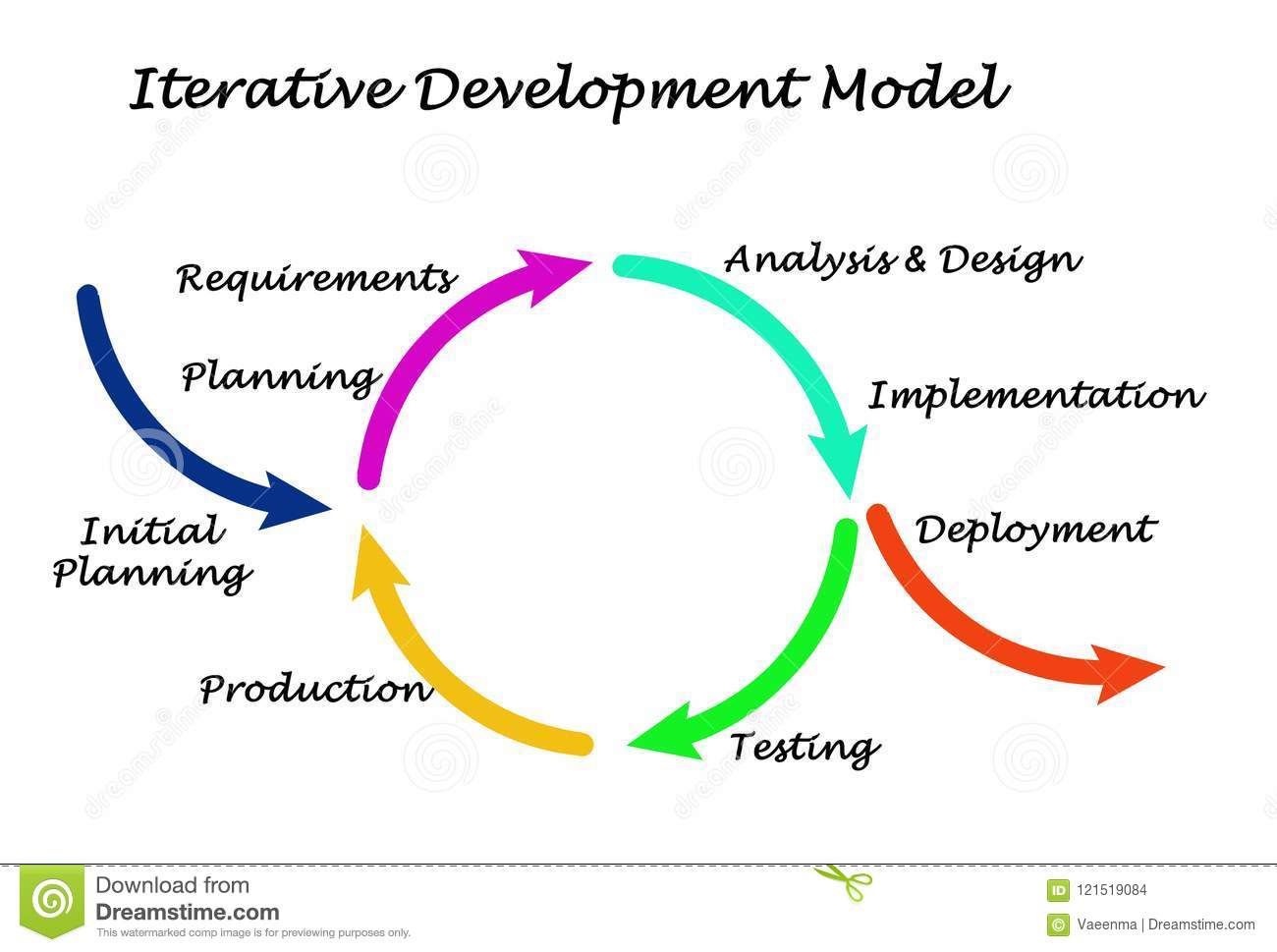
* Windows, Linux ar MacOS operacinė sistema.
* Visual Studio 2019
* Microsoft Sql Server Manager Studio (bet jau 2017 metų)

Taip pat reiktų nepamiršti būti įsirašius Angular 8. To pilnai pakanka norint dirbti su šiuo projektu.

## Projektavimo metodai

### Projektavimo valdymas ir eiga

Projektas buvo kuriamas naudojant iteracinį projektavimo modelį. Jo pavyzdinė schema pateikiama 2.8 paveikslėlyje.



2.8 pav. Iteracinis projektavimo modelis

Visa veika skirstoma į sprintus. Kiekvieno jų ilgis buvo po 2 savaites. Iš viso atlikti 6 sprintai. 2 paskutiniai šiek tiek neatitiko iteracinio projektavimo modulio, kadangi jis skiriamas dokumentacijos rengimui bei jau sukurtos sistemos testavimui, bandant visus įprastus atvejus bei ieškant galimų sistemos spragų.

Iteracijos pagal užduotis:

1. Pasiruošimas, reikalingos programinės įrangos susidiegimas, pavyzdinio projekto, duomenų bazės sukūrimas.
2. Darbų, pakvietimų posistemė, jų testavimas
3. Pranešimų, nustatymų posistemė, jų testavimas.
4. Tvarkaraščio posistemė, jos testavimas.
5. Testavimas, dokumentacija, kritinių klaidų paieška sistemoje.
6. Testavimas, dokumentacija, kritinių klaidų paieška sistemoje.

### Projektavimo technologija

Modeliavimui naudotas įrankis Magic Draw, kuris palaiko UML modeliavimo kalbą. Naudojau daugumą standartinių elementų kaip klasės, komponentai ir kita. Iš jų sudėjau keletą dažniausiai naudojamų diagramų pvz: klasių, paketų, komponentų, diegimo, būsenų bei sekų.

### Programavimo kalbos, derinimo, automatizavimo priemonės, operacinė sistemos

Sistema sukurta C# programavimo kalba. Programa sukompiliuojama ir jos paketai vykdomi .Net Core 3.1 aplinkoje. Sistema kurta Windows aplinkoje, tačiau naudojant .Net Core veiktų ir Linux ar MacOs aplinkose.

Programavimui naudota Windows10 aplinka, bei Visual Studio 2019 programavimo aplinką.

Duomenų bazė Microsft Sql Server Managment Studio 2019. Tai reliacinė duomenų bazė.

Vartotojo sąsajai programuoti naudota Angular 8 karkasas, kuris yra paremtas TypeScript programavimo kalba, kartu su HTML5 žymėjimo ir CCS3 stilių aprašymo kalbomis.

Testavimu naudota Xuint testavimo įrankių paketas.

## Sistemos projektas

### Statinis sistemos vaizdas

* Išdėstymo (*UML deployment diagram*) – nepakeičiama tuo atveju, jei sistema naudoja išorinius servisus ar yra paskirstyta per keletą įrenginių. Geriausia pradėti nuo šios diagramos, nes ji greičiausiai supažindina su bendra sistemos sudėtimi.



2.9 pav. sistemos išdėstymo diagrama



2.10 pav. sistemos komponentų diagrama



2.11 pav. sistemos paketų diagrama

Laiko planavimo sistemos paketų diagrama parodo jog pati sistema sudaryta iš dviejų dalių. Tai yra API skirtas duomenų apdorojimui ir perdavimui, bei UI (nuo angliškų žodžių – user interface) vartotojo sąsaja su visu reikalingu funkcionalumu. Duomenys atiduodami naudojantis įvairias kontroleriais siunčiant HTTP užklausą.

API detalizavimas:

* Dtos (angl. data transfer object) – skirti duomenų perdavimui tarp servisų ir tarp servisų – valiklių. Taip pat jie perduodami kartu su HTTP užklausomis.
* Controllers – valdikliai, skirti duomenų perdavimu ir surikimui.
* Services – dalis atsakinga už duomenų transformavimą ir reikalingą formą, skaičiavimų atlikimą.
* Repositories – atsakingos už duomenų surinkimą iš duomenų bazės.
* Database(Models) – modelis atitinka duomenų bazės lentelę.
* Migrations – tai naudojant sugeneruojamas kodas, iš kurio bus sukurta duomenų bazė. Kodo generavimas vyksta naudojant esamus modelius.

UI detalizavimas:

* models – struktūra atitinka Dtos struktūrą, tačiau naudojami norint atvaizduoti duomenis vartotojui arba gražinti pakeistus duomenis į valdiklius
* services – duomenų, surenka duomenis jos perduoda į komponentus
* helpers – pagalbinės funkcijos susijusios su vartotojų autentifikavimu
* components – apjungiama dizainais su duomenimis, informacija atvaizduojama vartotojui



2.12 pav. darbų posistemės klasių diagrama

Ši klasių diagrama atitinka dalies sistemos vaizdą, kitos dalys yra labai panašios. Tačiau tai yra puikus pavyzdys kaip duomenys iš duomenų bazės pasiekia valdiklį ir vėliau vartotoją. Tokiam atvaizdavimui pasirinkta darbų posistemė.

Taigi klasė *Work* yra darbo informaciją apibūdinantis modelis. Ji paveldi klasę *BaseEntity* (plačiau apie tai bus kitoje dalyje). Iš *DatabaseContext* klasės duomenys paimami naudoja *WorksRepository*, kuri paveldi iterafeisą *IWorksRepository* ir įgyvendina jame aprašytą funkcionalumą. Tada *Work* modelis perduodamas i servisą *WorkService,* kuris paveldi interfeisą *IWorkService,* taip pat du kitus servisus su savo funkcionalumu, kuris šiuo atveju nėra svarbus, bet turi įtakos kitoms posistemėms. Toliau šis servisas per *Dto*, šiuo konkrečiu atveju *WorkDto* į valdiklį.

Dar vienas svarbus dalykas *Configurations* paketas. Jo klasės turi labai svarbias funkcijas. *AutoMapperConfiguration* padeda paversti vieno tipo objektą į kito tipo, taip supaprastinant kodą ir padidinant jo skaitomumą, o *DependencyInjectionExtensions* leidžia vietoj klasės naudoti jos interfeisus.



2.13 pav. Duomenų bazės schema

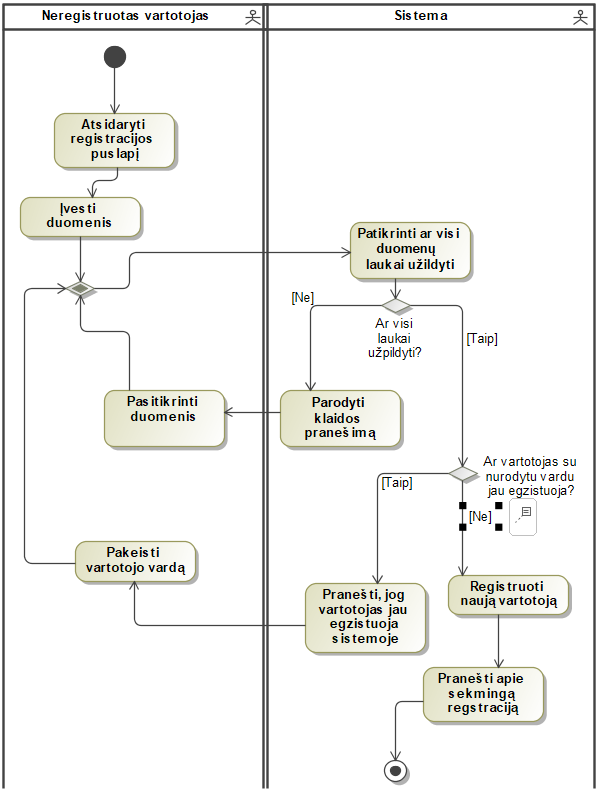
Duomenų bazė yra sudaryta iš 6 lentelių, tai pat dviejų išvardijimų. Lentelė *User* saugo informacija apie visus užsiregistravusius vartotojus. Čia saugomi visi identifikaciniai duomenys, juos naudojant galima pasiekti gerokai platesnės asmeninės informacijos kaip pavyzdžiui darbus, kurie saugomi *Work* lentelėje. Tai yra svarbiausia lentelė, kuri saugo visą kiekvieno darbo informaciją. Dar viena ne mažiau svarbi lentelė yra *UserSettings*. Jos tikslą atspindi ir pavadinimas, saugoti visus vartotojų nustatymus. Kiekvienas vartotojas turi tik vieną įrašą nustatymų lentelėje. Paskutinė likus lentelė – *Message* sauganti visus vartotojo gautus pranešimus. Pranešimų šablonai imami iš lentelės *MessageTempate,* papildomi reikalinga informacija ir sudedami i *Message* ir taip vartotojas gali juos pamatyti.

Taip pat pateikti dvi išvardinamojo duomenų tipo klasės, kurios saugomos jau pačiame kode. Tai yra *PriorityEnum* ir *InvitationStatusEnum.*

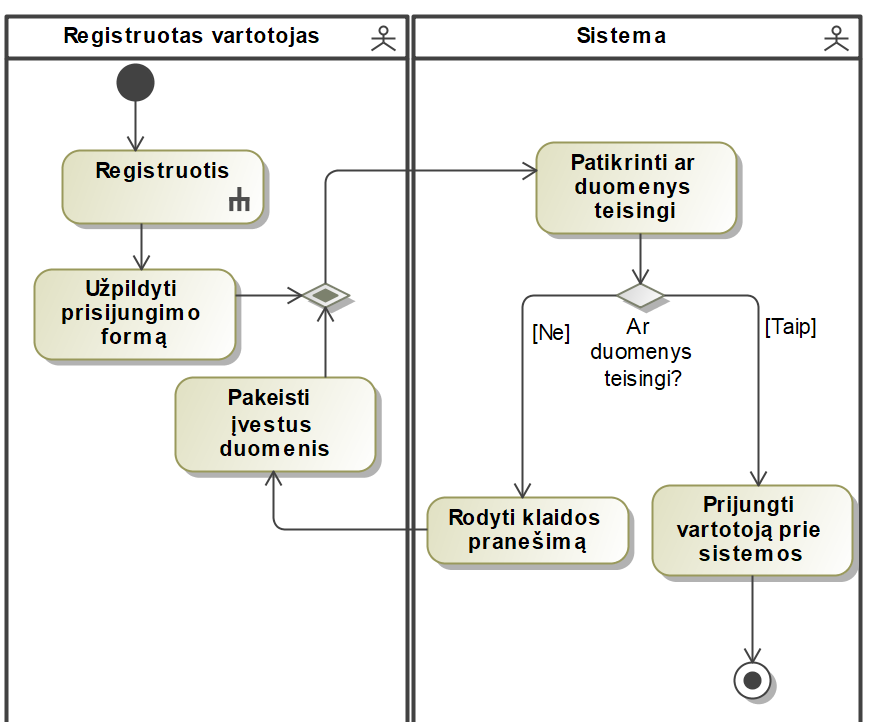
Visos klasės atitinkančios duomenų bazės lenteles paveldi klasę *BaseEntity*. Ši klasė turi tik vieną kintamąjį t.y. Id. Id galima laikyti įrašo numeriu. Tai reiškia kiekvienoje lentelėje bus įrašo numerio stulpelis.

### Dinaminis sistemos vaizdas

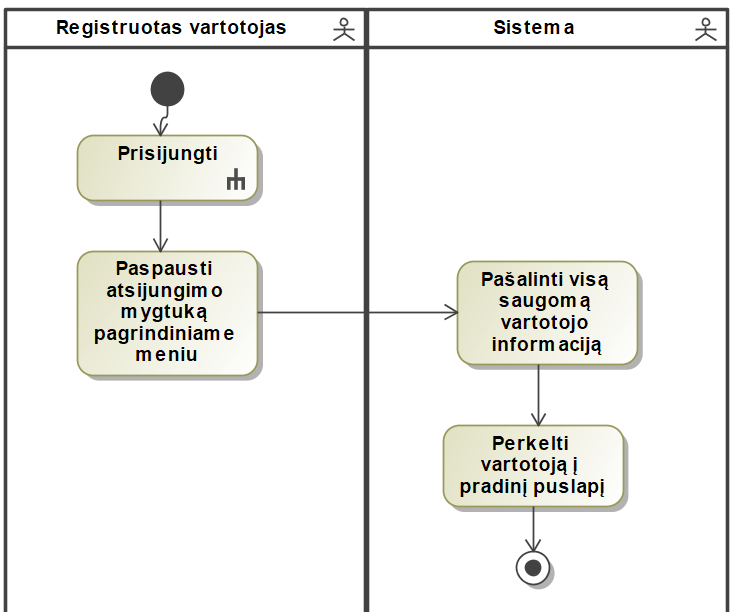
Toliau pateikiama visų panaudojimo atvejų veiklos diagramos(2.14 – 2.35). Jos atspindi vartotojo bei sistemos elgseną, kai gaunami teisingi rezultatai.



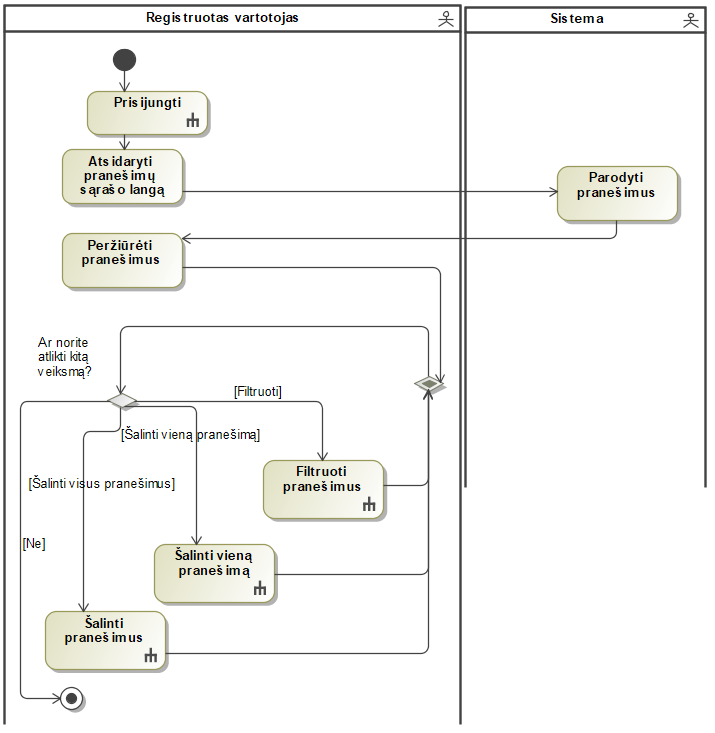
2.14 pav. registracijos veiklos diagrama



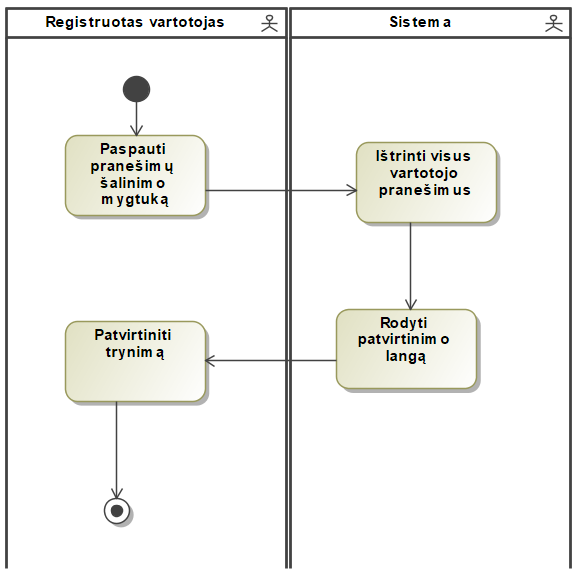
2.15 pav. prisijungimo veiklos diagrama



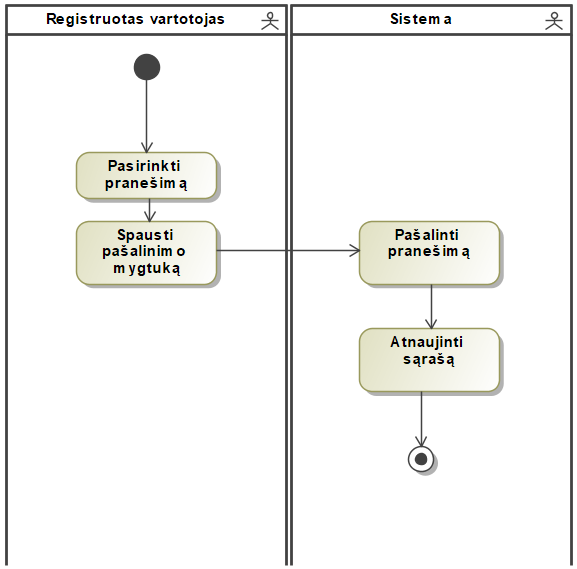
2.16 pav. Atsijungimo veiklos diagrama



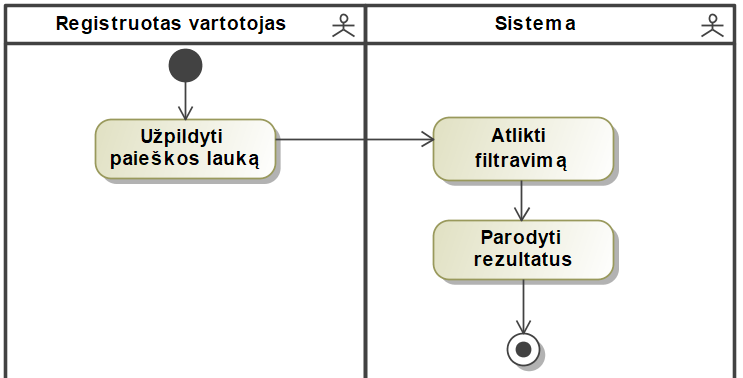
2.17 pav. pranešimų peržiūros veiklos diagrama



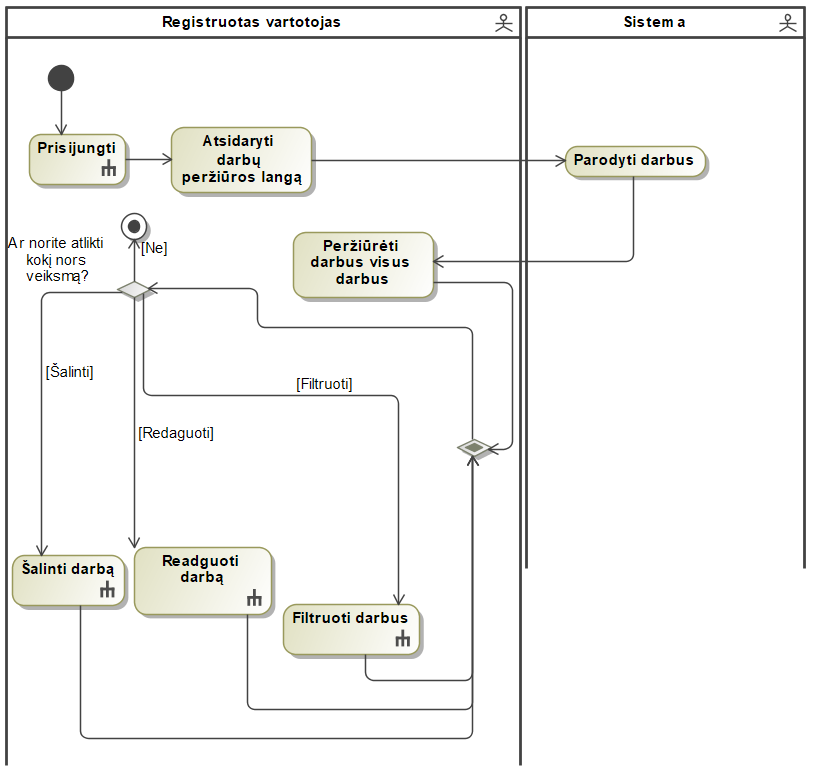
2.18 pav. Visų pranešimų šalinimo veiklos diagrama



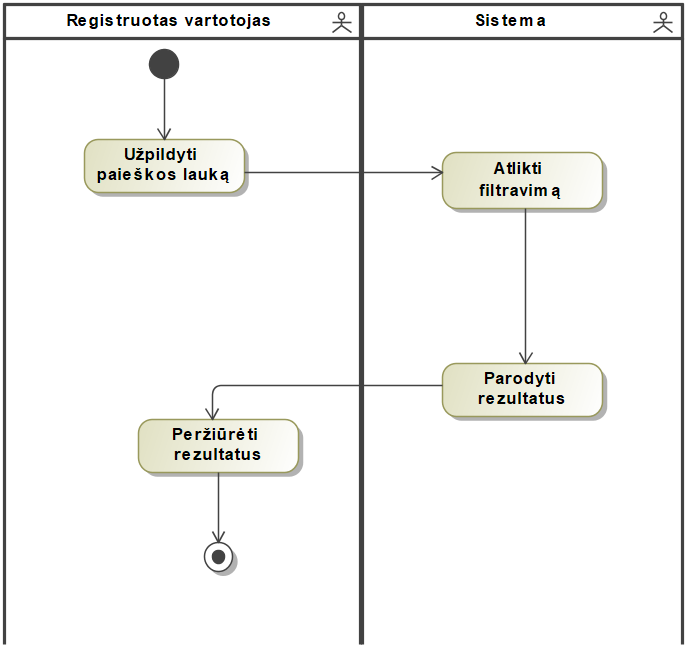
2.19 pav. pranešimo šalinimo veiklos diagrama



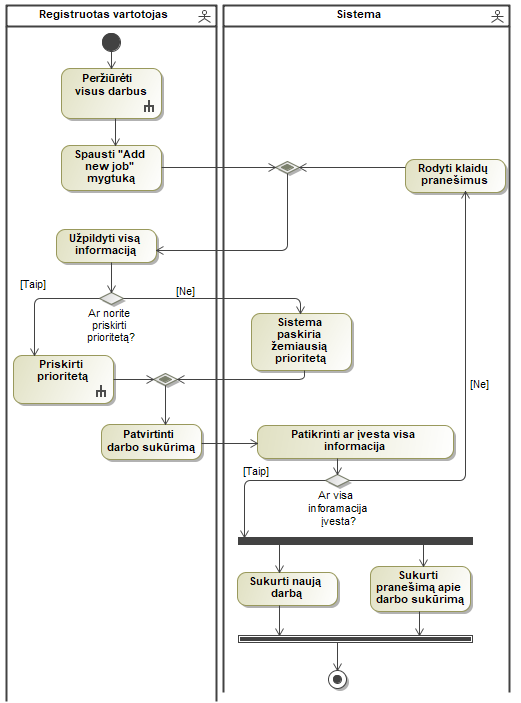
2.20 pav. pranešimų filtravimo veiklos diagrama



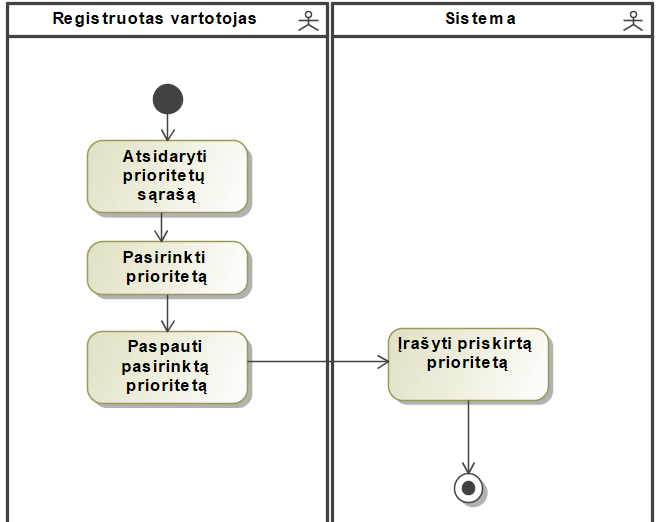
2.21 pav. darbų peržiūros veiklos diagrama



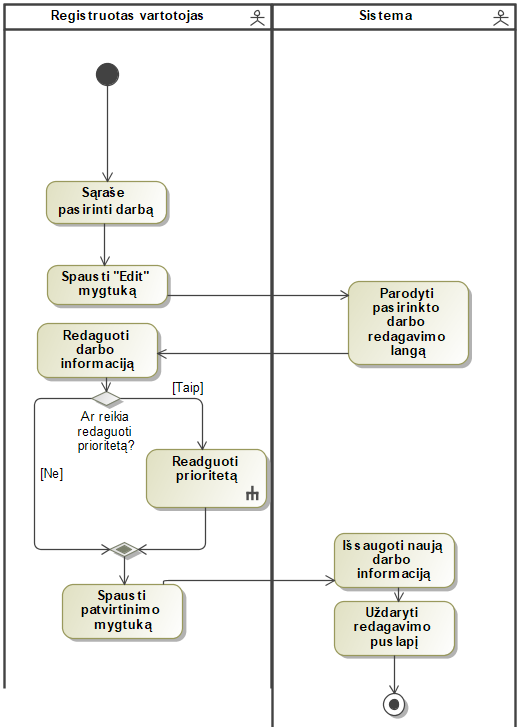
2.22 pav. darbų filtravimo veiklos diagrama



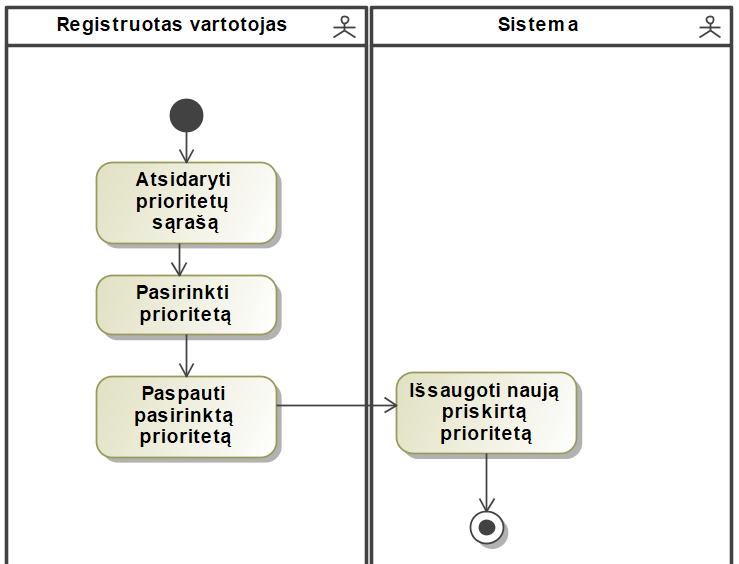
2.23 pav. darbo sukūrimo veiklos diagrama



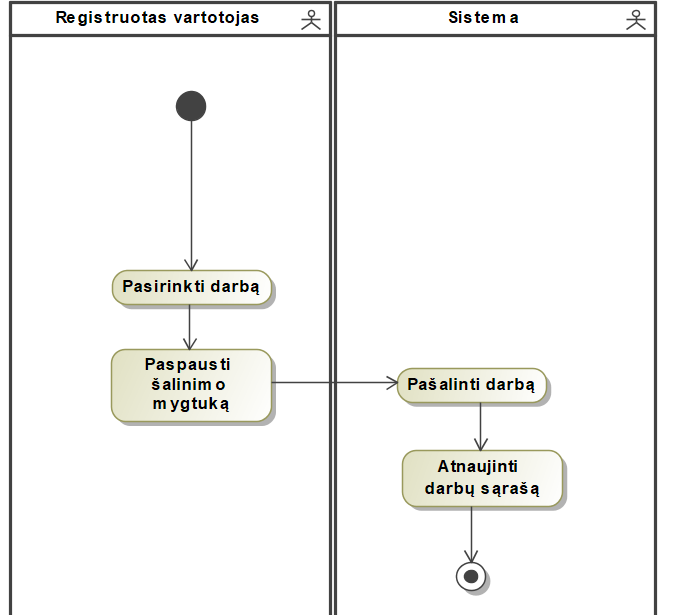
2.24 pav. prioriteto paskyrimo veiklos diagrama



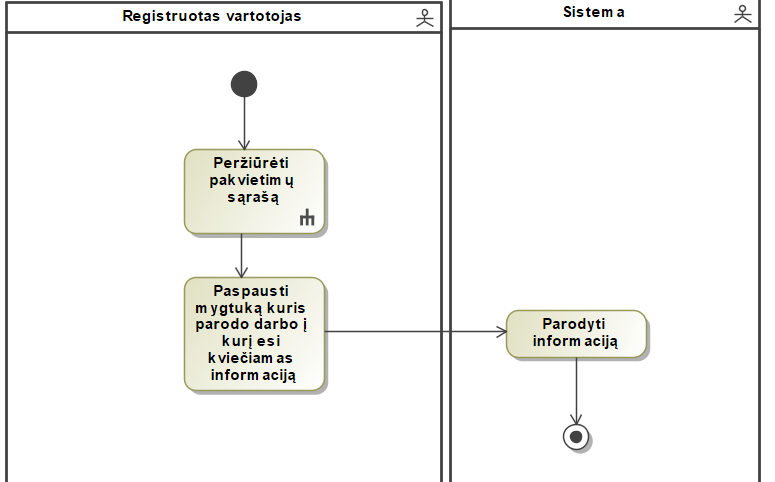
2.25 pav. darbo redagavimo veiklos diagrama



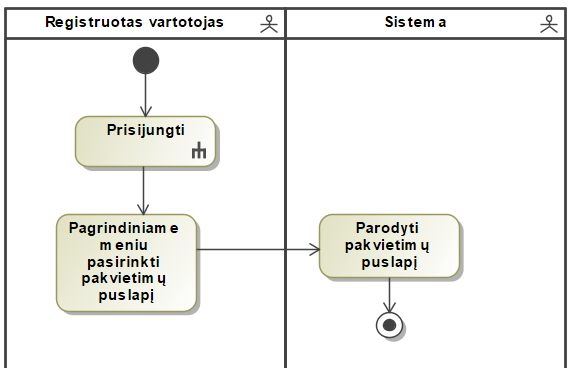
2.26 pav. prioriteto redagavimo veiklos diagrama



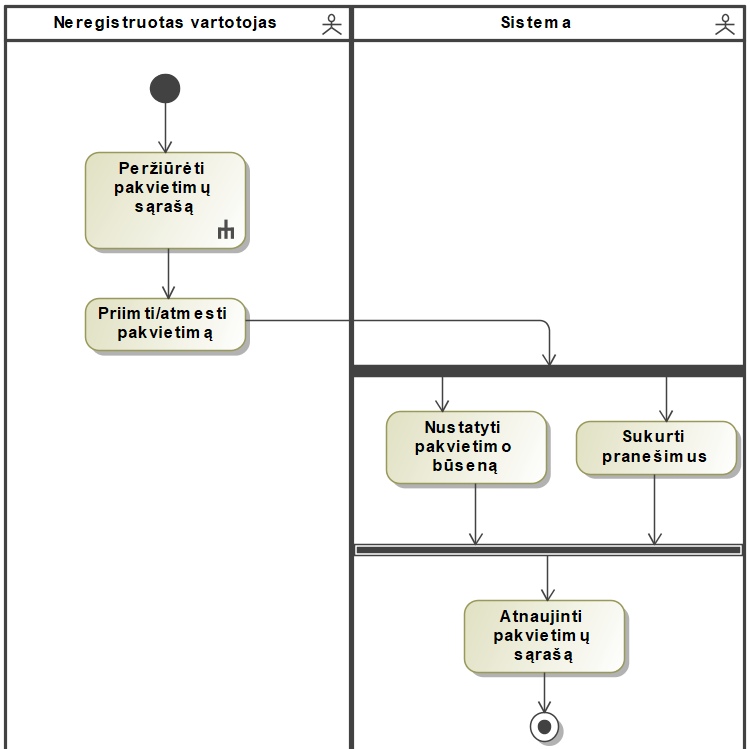
2.27 pav. darbo šalinimo veiklos diagrama



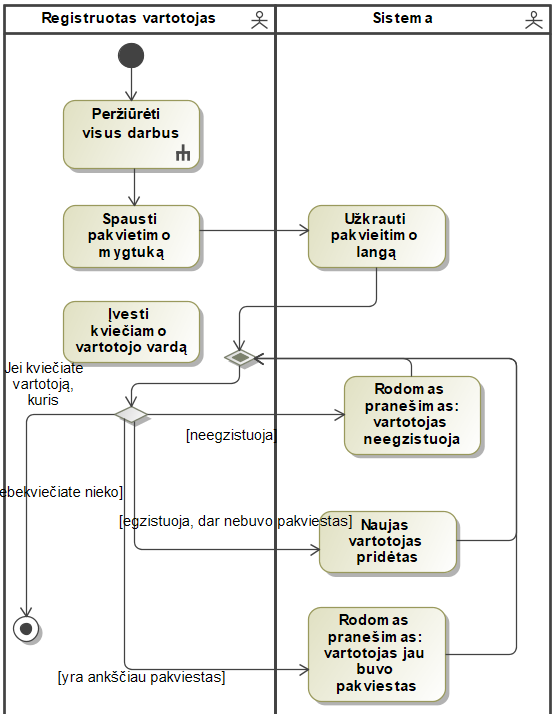
2.28 pav. darbo informacijos peržiūros veiklos diagrama



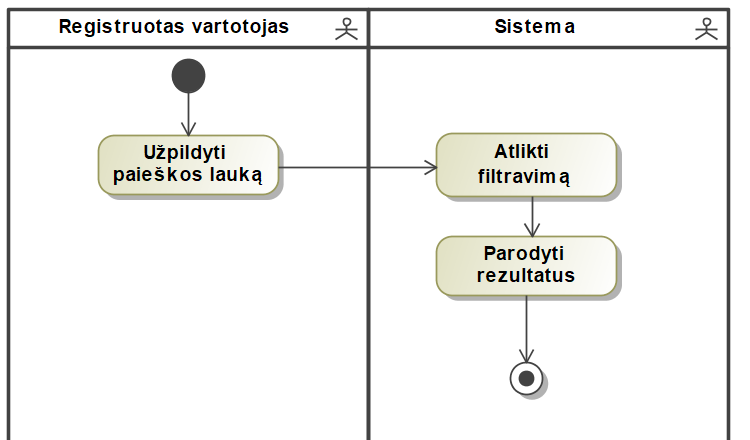
2.29 pav. pakvietimų sąrašo peržiūros veiklos diagrama



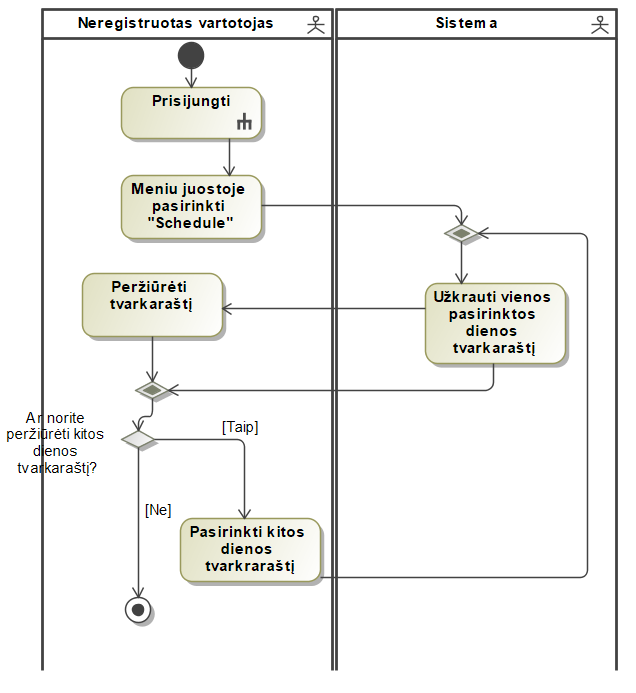
2.30 pav. pakvietimų priėmimo/atmetimo veiklos diagrama



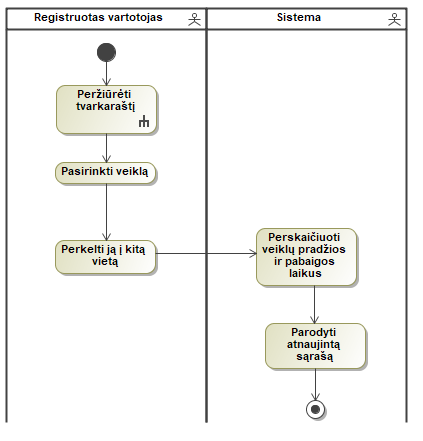
2.31 pav. pakvietimų siuntimo veiklos diagrama



2.32 pav. pakvietimų filtravimo veiklos diagrama



2.33 pav. tvarkaraščio peržiūros veiklos diagrama



2.34 pav. tvarkaraščio redagavimo veiklos diagrama



2.35 pav. tvarkaraščio generavimo veiklos diagrama

# Testavimas

Skyrius skirtas testavimo plano sudarymui, bei jo įgyvenimui. Planuoju pateiki ir naudojamas technologijas bei aprašyti gautus rezultatus.

## Testavimo planas

Sistemai būtinas testavimas įsitikinti, kad jo veikimas atitinka projektavimo metu nustatytus tikslus. Kad tikslus įgyvendinti teisingai buvo sudarytas toks testavimo planas:

1. Atlikti kiekvieno serviso komponentų testus.
2. Atlikti API metodų testus naudojant Swagger įrankį.
3. Atlikti statine kodo analizė, nepaliekant kodo dalių kurios ateityje gali turėti įtakos klaidos atsiradimui.
4. Atlikti svetainės grafinės vartotojo sąsajos testavimą rankiniu būdu

## Testavimo kriterijai

Kad testavimas būtų sėkmingai įvykdytas, turi būti įgyvendinti šiek kriterijai:

1. Servisų kodo padengimas virš 90%.
2. API testavimas visada gražina rezultatus kurių tikėtasi.
3. Statinė kodo analizė negali turėti nei vieno klaidos pranešimo, taip pat kuo mažiau perspėjimų.
4. Visi sistemos funkcijos turi būti atliekamos taip kaip nurodyta specifikacijoje.

## Komponentų testavimas

### Automatinis testavimas

Komponentų testavimas bus naudojamas tik servisams, nes juose naudodami visi kiti metodai. Taip jie bus ištestuojami kartu. Kontrolerių testuoti nėra prasmės, jie tik kviečia servisus.

Servisų kodo padengimo kriterijus įgyvendintas sėkmingai, kadangi pateiktas 90 % kriterijus yra pasiektas.



**3.1 pav. Kodo padengimas**

Plačiau apie komponentų testavimą galima pasakyti, jog atlikti 56 testai. Kaip matome iš paveikslėlio (3.1 pav.) visi testai buvo sėkmingi. Kodo padengimas svyruoja nuo 81% iki 100%. Tas parodo, kad testavimas atliktas gan nuosekliai, atsižvelgiant daugumą galimų scenarijų.

Iš to galime teigti, kad komponentų testavimas buvo įvykdytas. Svarbu paminėti jo testuoti ne visi metodai bei klasės. Kontrolerių testavimas aprašomas kitame skyriuje.

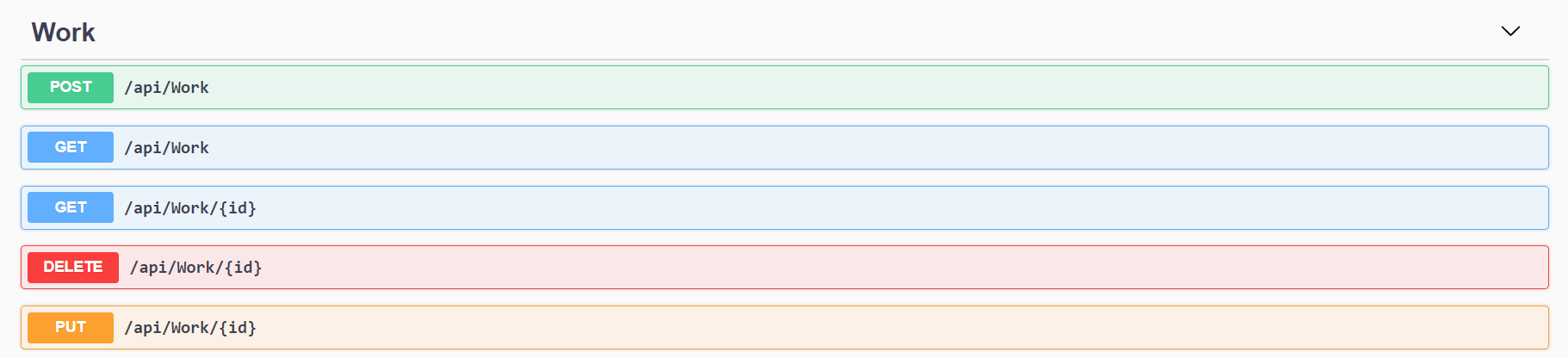


**3.2 pav. Visi atlikti testai**

Kiekvienam servisui parašyta nuo 1 iki 13 skirtingų testavimo scenarijų. Toks skaičius priklauso nuo serviso dydžio bei atliekamu funkcijų kiekio.

### Rankinis API testavimas

Rankinio API testavimo metu bus kviečiami visi metodai. Tam bus sudaryti scenarijai su tikėtinais rezultatais, gautais rezultatais juos įvykdžius ir tai pateikiama lentelėmis. Tie patys scenarijai tai pat bus panaudojami grafinės vartotojo sąsajos testavimui.



**3.3pav. Swagger grafinė vartotojo sąsaja**

|  |  |
| --- | --- |
| Metodo paskirtis | Sukurti naują darbą |
| Užlausos duomenys | {  "title": "work title",  "description": "long work description ",  "userId": 1,  "durationInMinutes": 20,  "activityPriority": 0,  "willBeParticipant": false  } |
| Sėkmingo veikimo scenarijus | Visi duomenys nurodyti |
| Sėkmingo veikimo rezultatas | Gražinamas 200 kodas, ir naujo darbo id |
| Nesėkmingo veikimo scenarijus | Trūksta dalies ar visų duomenų |
| Nesėkmingo veikimo rezultatas | Gražinamas 404 kodas |

**3.1 lentelė naujo darbo sukūrimas**

|  |  |
| --- | --- |
| Metodo paskirtis | Gauti visus darbus |
| Užlausos duomenys | {} |
| Sėkmingo veikimo scenarijus | Bent vienas darbas sukurtas |
| Sėkmingo veikimo rezultatas | Gražinamas 200 kodas ir visų darbų sąrašas |
| Nesėkmingo veikimo scenarijus | Nėra nei vieno darbo |
| Nesėkmingo veikimo rezultatas | Gražinamas 404 kodas |

**lentelė 3.2**

|  |  |
| --- | --- |
| Metodo paskirtis | Gauti darbą pagal jo id |
| Užlausos duomenys | {} |
| Sėkmingo veikimo scenarijus | Visi duomenys nurodyti |
| Sėkmingo veikimo rezultatas | Gražinamas 200 kodas, ir naujo darbo id |
| Nesėkmingo veikimo scenarijus | Trūksta dalies ar visų duomenų |
| Nesėkmingo veikimo rezultatas | Gražinamas 404 kodas |

**lentelė 3.3**

|  |  |
| --- | --- |
| Metodo paskirtis | Redaguoti darbą |
| Užlausos duomenys | {  "title": "updated title",  "description": "long updated description",  "userId": 1,  "durationInMinutes": 15,  "activityPriority": 1,  "willBeParticipant": true  } |
| Sėkmingo veikimo scenarijus | Visi duomenys nurodyti |
| Sėkmingo veikimo rezultatas | Gražinamas 204 kodas |
| Nesėkmingo veikimo scenarijus | Trūksta dalies ar visų duomenų |
| Nesėkmingo veikimo rezultatas | Gražinamas 404 kodas |

**lentelė 3.4**

|  |  |
| --- | --- |
| Metodo paskirtis | Šalinti darbą |
| Užlausos duomenys | {} |
| Sėkmingo veikimo scenarijus | Darbas buvo pašalintas |
| Sėkmingo veikimo rezultatas | Gražinamas 204 kodas |
| Nesėkmingo veikimo scenarijus | Darbo nebuvo rastas |
| Nesėkmingo veikimo rezultatas | Gražinamas 404 kodas |

**lentelė 3.5**

Visi 5 anksčiau pavaizduoti testai (3.1-3.5 lentelės) buvo atlikti naudojant Swagger grafinę sąsają (3.3 pav.). Iš viso atlikta 27 API metodų testavimo atvejai ir jie visi buvo sėkmingi.

## Statinė kodo analizė

Statinė kodo analizė buvo atlikta naudojant Resharper įrankį. Beje svarbu nepamiršti jog šis įrankis sugeba aptikti TypeScript, CSS, HTML bei C# kalbų, naudotų šiame projekte klaidas.

Pagrindinės šio įrankio aptinkamos klaidos yra šios:

* kintamųjų pavadinimo rašymo (išskyrus HTML ir CSS)
* formatavimo
* kodo stiliaus

Ir daug kitų, kurias ne mažiau svarbių aptikti, klaidų.

Jis įvykdė viso projekto kodo analizę bei joje aptiko 2 klaidas (3.3 pav). Tiesa jų nebuvo galima taisyti, kadangi tai yra Angular projektui paleisti reikalingas failas. Todėl ši statinės kodo analizės dalis laikoma sėkmingai įvykdyta.



**3.4 pav. Resharper aptiktos klaidos**

Peržiūrėjus informaciją apie klaidas buvo nuspręsta peržiūrėti ir perspėjimus. Iš viso aptikti 169 perspėjimai (3.3 pav.). Jie buvo net 56 skirtinguose failuose. Pamačius, jog jų skaičius yra gana didelis buvo nuspręsta patikrinti tuos failus ir įsitikinti ar galima jų kiekį sumažinti.



**3.5 pav. Perspėjimai dėl kodo kokybės, prieš peržiūrą**

Tada buvo atlikta kodo peržiūra tuose failuose kur aptikti perspėjimai, bei buvo ištaisyti visi, kurie neturėjo įtakos projekto veikimui ir liko tik tie, kurių nebuvo galima pakeisti pavyzdžiui migracijose, kurios yra generuojamos įvedus komandą ir jų redagavimas negalimas. Rezultatai (3.5 pav.) rodo gerokai sumažėjusį perspėjimų skaičių.



**3.6 pav. Perspėjimai dėl kodo kokybės, po peržiūros**

Galime padaryti išvadą, kad statinis kodo testavimas padėjo padaryti projekto kodą kokybiškesni, paprasčiau suprantamą ir padidino jo skaitomumą. Taip pat mažina klaidos tikimybę rašant didesnės apimties projektą.

## Vartotojo sąsajos testavimas

Vartotojo sąsajai testuoti pasirinkau rankinį testavimo būdą. Bus pateikta (3.6 lentelė) testavimo scenarijai bei numatomas rezultatas.

Kiekviena užduočiai atlikti yra būtina registracija ir prisijungimas, todėl atskiras šių panaudojimo atvejų testavimas nėra įtrauktas.

|  |  |  |
| --- | --- | --- |
| **Veiksmas** | **Aprašymas** | **Rezultatas** |
| Sukurti naują darbą: | * spaudžiamas „Works“ mygtukas pagrindiniame meniu * spaudžiama „Add new work“ mygtukas * užpildomi laukai * spaudžiamas „Create“ mygtukas | Sąraše atsirado naujas darbas |
| Redaguoti darbą: | * spaudžiamas „Works“ mygtukas pagrindiniame meniu * atsidariusiame sąraše paspausti mygtuką redaguoti * pakeisti kelis laukus * spausti „Edit“ | Darbas buvo atnaujintas |
| Pašalinti darbą: | * spaudžiamas „Works“ mygtukas pagrindiniame meniu * atsidariusiame sąraše paspausti raudoną „X“ mygtuką esantį sąraše | Darbas buvo ištrintas |
| Pakvieti kitus vartotojos atlikti darbą: | * spaudžiamas „Works“ mygtukas pagrindiniame meniu * spausti „Invite“ mygtuką esanti sąraše * įvesti norimo vartotojo vardą * spausti „Send invitation“ | Jei vartotojas yra atsiranda pranešimas apie išsiūtą pakvietimą, jei ne pranešimas apie klaidą |
| Perskaičiuoti darbų laiką | * Paspausti ant „Schedule“ pagrindiniame meniu * Paimti viena darbą ir pertemti į kitą vietą | Darbų laikas perskaičiuotas |
| Keisti diena | * Paspausti ant „Schedule“ pagrindiniame meniu * Pereiti diena pirmyn ir atgal, pažiūrėti kaip keičiasi dienos naudojant „Next“ ir „Previous“ mygtukus | Diena keičiasi teisingai, po „Previous“ paspaudimo mažėja, po „Next“ didėja |
| Atnaujinti tvarkaraštį | * Paspausti ant „Schedule“ pagrindiniame meniu * Paspausti mygtuką „Refresh“ | Tvarkaraštis atsinaujins |
| Peržiūrėti žinutes | * Paspausti ant „Messages“ mygtuko arba voko simbolio pagrindiniame meniu | Žinučių sąrašas matomas |
| Filtruoti žinutes | * Paspausti ant „Messages“ mygtuko arba voko simbolio pagrindiniame meniu * Įvesti įvarius filtrus, pažiūrėti kaip kinta žinučių sąrašas | Žinučių sąrašas filtruojamas |
| Pašalinti visas žinutes | * Paspausti ant „Messages“ mygtuko arba voko simbolio pagrindiniame meniu * Paspausti mygtuką „Delete all“ | Visos žinutės pašalintos |
| Pašalinti vieną žinutę | * Paspausti ant „Messages“ mygtuko arba voko simbolio pagrindiniame meniu * paspausti raudoną „X“ mygtuką esantį žinučių sąraše | Pasirinkta žinutė ištrinta |
| Peržiūrėti pakvietimų sąrašą | * Paspausti ant „Invitation“ mygtuko pagrindiniame meniu * Peržiūrėti ar sąraše yra pakvietimų | Pakvietimų sąrašas matomas |
| Priimti/atmesti pakvietimą | * Paspausti ant „Invitation“ mygtuko pagrindiniame meniu * Priimti/atmesti pakvietimą * Jei pakvietimas priimtas, pažiūrėti ar darbų sąraše atsirado naujas darbas | Pakvietimas pašalinamas iš sąrašo, priėmimo atveju darbas atsiranda darbų sąraše |
| Peržiūrėti darbo į kurį kviečiama informacija | * Paspausti ant „Invitation“ mygtuko pagrindiniame meniu * Spausti mygtuką „Details“ ant pasirinkto pakvietimo | Darbo informacija matomos |
| Pakeisti darbo pradžios ir pabaigos nustatymus | * Paspausti ant vartotojo vardo pagrindiniame meniu * Pasirinkti laikus * Spausti „Update“ mygtuką | Laikai atnaujinti |
| Šalinti paskyra | * Paspausti ant vartotojo vardo pagrindiniame meniu * Paspausti mygtuką „Delete“ * Patvirtinti paskyros šalinimą * Pabandyti prisijungti su buvusios paskyros vardu ir slaptažodžiu | Paskyra ištrinta, prisijungti negalima |
| Nustatyti puslapio dydį | * Paspausti ant vartotojo vardo pagrindiniame meniu * Pasirinkti iš sąrašo tinkama dydį * Spausti „Update“ mygtuką | Puslapio dydis pasikeitė |
| Atsijungti | * Spausti „Log out“ mygtuką pagrindiniame meniu | Bus atsijungta nuo paskyros, informacija taps nepasiekiama |

**3.6 lentelė vartotojo sąsajos testavimas**

Testuojant vartojo sąsają jokių problemų neaptikta. Vartotojas gali atlikti visus anksčiau (3.6 lentelėje) nurodytus veiksmus.

# Dokumentacija naudotojui

Šioje dalyje aprašoma sukurtos programinės įrangos funkcijos skirtos naudotojams.

## Apibendrintas sistemos galimybių aprašymas.

Sistemos galimybės nuo reikalavimuose aprašyto funkcionalumo skiriasi tuo, kad ne visiems vartotojams būtina žinoti technines projekto detales. Pavyzdžiui, internetinio portalo vartotojui svarbu žinoti kokios naudingos funkcijos yra portale (pvz., paieška, naujienlaiškio prenumerata ir kt.), tačiau ne visos funkcijos įprastam vartotojui yra aktualios (pvz., reklamos skydelių palaikymas, SSL protokolas vartotojų autentifikacijai ir t.t.).

## Vartotojo vadovas

Norint naudotis sistema yra būtina registracija.

Vartotojo vadovas yra neformalus įvadas į sistemą, aprašantis jos „normalų“ vartojimą. Kitaip tariant, vartotojui draugiška instrukcija su daug iliustracijų ir paaiškinimų. Neišvengiamai pradedantieji, nepriklausomai nuo patirties, daro klaidas. Lengvai randama informacija, kaip nuo šių klaidų grįžti prie naudingo darbo ir atstatyti galimus klaidų padarinius, turi būti sudėtinė šio dokumento dalis.

Rezultatai ir išvados

Atlikus darbą prieitą prie tokių išvadų:

1. Atlikus konkurentų analizę buvo surastos panašų funkcionalumą turinčios sistemos. Paaiškėjo, jog dauguma sistemų yra perkrauta nereikalingu funkcionalumu. Todėl nuspręsta kurti naują sistemą, kuri būtų paprastesnė, be nereikalingu dalių, todėl leistų vartotojui greičiau išmokti ja naudotis.
2. Atlikus technologijų analize, paaiškėjo, jog tokį patį rezultatą galima ir naudojant ir kitas tinklapių programavimo technologijas, todėl pasirinkimą lėmė sistemą kuriančio žmogaus įgūdžiai, tam tikrų subtilybių išmanymas.
3. Atlikus projektavimą, paaiškėjo geriausias ir mažiausiai laiko užimantis būdas sukurti naują sistemą. Naudojant šį projektavimo būdą pavyko sumažinti klaidų tikimybę, pagerinti kodo kokybę, bei rezultatas buvo pasiektas per trumpesnį laiko tarpą.
4. Atlikus testavimą, įsitikinta sistemos veikimu. Aukštas ištestuoto kodo procentas leidžia įsitikinti, kad sistemos rezultatai bus toki, kokius tikisi gauti vartotojas.
5. Ruošiant dokumentaciją, ji padaryta paprasta ir detali, kad kiekvienas naujas vartotojas suprastų sistemos veikimą bei rastų visus reikalingus atsakymus.

Gauti rezultatai parodė:

1. Sistemos būklė yra gera, tačiau norint pritaikyti platesnei vartotojų auditorijai reikia tam skirti didelius laiko resursus, ieškoti aukštesnės kvalifikacijos žmonių galinčių prisidėti prie jos tobulinimo.
2. Dauguma planuotų savybių įgyvendinta iki galo, tačiau plečiant sistemą būtų naudinga ją papildyti tokiu funkcionalumu:
   * Kelių vartotojų bendru darbu tuo pačiu metu kai priimami pakvietimai.
   * Pranešimų siuntimas elektroniniu paštu
   * Gražesnis/patogesnis kalendorius.
   * Darbų pradžios ir pabaigos laikų generatoriaus tobulinimas

Literatūros sąrašas

1. Apie LITNET. *Litnet.* [Tinkle] 2012 m. birželio 05 d. [Cituota: 2013 m. balandžio 04 d.] http://www.litnet.lt/index.php/apie-litnet.

2. *Transforming Ontology Representation from OWL to Relational Database.* Vyšniauskas, E. ir Nemuraitė, L. 3, 2006 m., Information Technology and Control, T. 35A, p. 333–343.

3. Masiulis, K. ir Krupavičius, A. *Valstybės tarnyba Lietuvoje: praeitis ir dabartis: kolektyvinė monografija.* Vilnius : Praction, 2007. p. 430.

4. *Spaudos draudimo klausimai.* Biržiška, V. 5, 1929 m., Kultūra, p. 249-235.

5. Valiulytė, Ieva. Išlaidos krašto apsaugai, jų pagrįstumas ir tikslingumas. *Sociumas.* [Tinkle] 2000 m. vasaris. [Cituota: 2001 m. gruodžio 12 d.] http://www.sociumas.lt.

6. Library, Dalhousie University. IEEE Citation style guide. [Tinkle] 2009 m. [Cituota: 2013 m. 04 11 d.] http://libraries.dal.ca/content/dam/dalhousie/pdf/library/Style\_Guides/IEEE\_Citation\_Style\_Guide.pdf.

7. *Hibridinis velomobilis.* Gradauskas, R. Kaunas : s.n., 2000. Transporto priemonės - 99. p. 81-83.

<https://www.google.com/search?q=iterative+development&sxsrf=ALeKk039ascCvEyAAC2erP2O-_HKgpfTuw:1587041012568&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjG6bC-_OzoAhVBfZoKHTeXB1MQ_AUoAXoECBQQAw&biw=1536&bih=722>

Darbe naudotos literatūros sąrašas (1 – 3 lapai). Sąrašas sudaromas vadovaujantis ISO 690 priimtu literatūros sąrašo ir citavimo stiliumi (1). Kaip sudarinėti literatūros sąrašą Word priemonėmis galite paskaityti <http://office.microsoft.com/en-us/word-help/create-a-bibliography-HA102809686.aspx> arba <http://office.microsoft.com/lt-lt/word-help/create-a-bibliography-HA102809686.aspx>.

Literatūros sąrašas turėtų apimti visus naudotus šaltinius. Literatūros šaltiniai pateikiami sunumeruoti citavimo tvarka. Darbo apraše turi būti pacituoti visi naudoti šaltiniai, pateikiant tekste nuorodas. Daugiau informacijos apie bendras citavimo taisykles galite rasti <https://biblioteka.ktu.edu/mokymai/#mokymosi-medziaga> „Kaip cituoti šaltinius ir parengti literatūros sąrašą. ISO 690:2010 standartas (skirta technologijos mokslams)“.

Priedai

Papildoma informacija ir dokumentai, neįeinanti į pagrindinį dokumentą. Dažniausiai į priedus keliamos specifikacijos, sąsajų (API) aprašai, diagramos ar kita informacija, kuri yra svarbi projektui, tačiau užima dokumente ganėtinai daug vietos. Jei priedai nėra naudojami, šis puslapis yra ištrinamas.

1. priedas. Priedo pavadinimas

**4.1** **lentelė.** Pagrindiniai baigiamojo projekto stiliai ir jų aprašymai

| **Stiliaus pavadinimas** | Stiliaus pavadinimas galerijoje | Stiliaus formalieji reikalavimai | Stiliaus naudojimo aprašymas |
| --- | --- | --- | --- |
| Antraštė non-TOC | Antraštė non-TOC | Šrifto dydis 12 pt, šriftas paryškintas, intervalas tarp eilučių – 1,15, atstumas prieš ir po antraštės – 10 pt, centruota lygiuotė. | Antraštėms, kurios nėra įtraukiamos į turinį: „Santrauka“, „Summary“, „Turinys“. |
| Antraštė be nr. | Antraštė be nr. | Šrifto dydis 12 pt, šriftas paryškintas, intervalas tarp eilučių – 1,15, atstumas prieš ir po antraštės – 10 pt, centruota lygiuotė, antraštė rašoma naujame puslapyje – po puslapio skirtuko. | Antraštėms, kurios įtraukiamos į turinį, bet nėra numeruojamos: „Lentelių sąrašas“, „Paveikslų sąrašas“, „Santrumpų ir terminų sąrašas“, „Įvadas“, „Išvados“, „Literatūros sąrašas“, „Informacijos šaltinių sąrašas“, „Priedai“. |
| 1. Heading 1,Skyrius | Skyrius | Šrifto dydis 12 pt, šriftas paryškintas, intervalas tarp eilučių – 1,15, atstumas po antraštės – 10 pt, abipusė lygiuotė, antraštė rašoma naujame puslapyje – po puslapio skirtuko. | Skyrių antraštėms, kurios įtraukiamos į turinį ir yra numeruojamos. |
| 1.1. Heading 2,Poskyris | Poskyris | Šrifto dydis 12 pt, šriftas paryškintas, intervalas tarp eilučių – 1,15, atstumas prieš ir po antraštės – 10 pt, abipusė lygiuotė, numeracija siejama su aukštesnio lygio antrašte. | Poskyrių antraštėms, kurios įtraukiamos į turinį ir yra numeruojamos. |
| 1.1.1. Heading 4,Skyrelis | Skyrelis | Skyrelių antraštėms, kurios įtraukiamos į turinį ir yra numeruojamos. |
| Tekstas | Tekstas | Šrifto dydis 12 pt, intervalas tarp eilučių – 1,15, atstumas po pastraipos – 10 pt, abipusė lygiuotė. | Tekstui visose projekto dalyse (santraukose, įvade, skyriuose, poskyriuose ir t.t.). |
| List Bullet;Sąrašas (suženklintas) | Sąrašas (suženklintas) | Pirmos pastraipos eilutės įtrauka – 0,63 cm, šrifto dydis 12 pt, intervalas tarp eilučių – 1,15, atstumas tarp tokio paties stiliaus pastraipų – 0 pt, atstumas po sąrašo – 10 pt, abipusė lygiuotė. | Tekstui, kuris pateikiamas suženklintu sąrašu. |
| List Number;Sąrašas (numeruotas) | Sąrašas (numeruotas) | Šrifto dydis 12 pt, intervalas tarp eilučių – 1,15, atstumas tarp tokio paties stiliaus pastraipų – 0 pt, atstumas po sąrašo – 10 pt, abipusė lygiuotė. | Tekstui, kuris pateikiamas sunumeruotu sąrašu. |
| Footnote Text;Išnašos tekstas | Išnašos tekstas | Šrifto dydis 10 pt, intervalas tarp eilučių – 1,15, atstumas prieš ir po sąrašo – 0 pt, abipusė lygiuotė. | Tekstui, kuris pateikiamas išnašose. |
| Lentelės pavad. | Lentelės pavad. | Šrifto dydis 11 pt, intervalas tarp eilučių – 1,15, atstumas prieš pavadinimą – 10 pt, po pavadinimo – 3 pt, lygiuotė prie kairiojo krašto. | Lentelių pavadinimams (numeris ir žodis lentelė rašomas paryškintu šriftu). |
| Lentelės I eil. | Lentelės I eil. | Šrifto dydis 10 pt, šriftas paryškintas, intervalas tarp eilučių – 1,15, atstumas prieš ir po pastraipos – 3 pt, lygiuotė prie kairiojo krašto. | Tekstui lentelės antraštinei (pirmai) eilutei. |
| Lentelė | Lentelė | Šrifto dydis 10 pt, intervalas tarp eilučių – 1, atstumas prieš ir po pastraipos – 3 pt, lygiuotė prie kairiojo krašto. | Tekstui lentelėje. |
| Caption,Paveikslo pavad. | Paveikslo pavad. | Šrifto dydis 11 pt, intervalas tarp eilučių – 1,15, atstumas prieš ir po pavadinimo – 10 pt, centruota lygiuotė. | Paveikslų pavadinimams (numeris ir santrumpa pav. rašoma paryškintu šriftu). |
| Figure;Paveikslas | Paveikslas | Atstumas prieš ir po paveikslo – 10 pt, centruota lygiuotė. | Paveikslui, iliustracijai . |
| Bibliography,Bibliografija | Bibliografija | Šrifto dydis 12 pt, intervalas tarp eilučių – 1,15, atstumas tarp tokio paties stiliaus pastraipų – 2 pt, abipusė lygiuotė. | Literatūros ir Informacijos šaltinių sąrašuose nurodytiems šaltiniams. |
| Priedas | Priedas | Šrifto dydis 12 pt, šriftas paryškintas, intervalas tarp eilučių – 1,15, atstumas prieš ir po antraštės – 10 pt, lygiuotė prie kairiojo krašto. | Priedo numeriui, žodžiui *priedas*, priedo pavadinimui. |
| TOC 1,Turinys 1 | Turinys 1 | Šrifto dydis 12 pt, šriftas paryškintas, intervalas tarp eilučių – 1,15, įtrauka – 0,64. | Turinyje esančioms antraštėms, kurios nėra numeruojamos („Lentelių sąrašas“, „Paveikslų sąrašas“, „Santrumpų ir terminų sąrašas“, „Įvadas“, „Išvados“, „Literatūros sąrašas“, „Informacijos šaltinių sąrašas“, „Priedai“) ir numeruojamai antraštei „Skyriaus pavadinimas“. |
| TOC 2,Turinys 2 | Turinys 2 | Šrifto dydis 12 pt, intervalas tarp eilučių – 1,15, įtrauka – 0,96. | Turinyje esančiai antraštei „Poskyrio pavadinimas“. |
| TOC 3,Turinys 3 | Turinys 3 | Šrifto dydis 12 pt, intervalas tarp eilučių – 1,15, įtrauka – 1,28. | Turinyje esančiai antraštei „Skyrelio pavadinimas“. |
| TOC 4,Turinys 4 | Turinys 4 | Šrifto dydis 12 pt, intervalas tarp eilučių – 1,15, įtrauka – 0,64. | Numeruotiems priedams. |