****

**Kauno technologijos universitetas**

Informatikos fakultetas

**Projektų valdymo įrankis**

Baigiamasis bakalauro studijų projektas

|  |
| --- |
|  |
| **Martynas Stumbra**  Projekto autorius |
|  |
| **Prof. Eduardas Bareiša**  Vadovas |
|  |

**Kaunas, 2021**

****

**Kauno technologijos universitetas**

Informatikos fakultetas

**Projektų valdymo įrankis**

Baigiamasis bakalauro studijų projektas

Programų sistemos (612I30002)

|  |  |
| --- | --- |
|  |  |
| **Martynas Stumbra**  Projekto autorius | (parašas) (data) |
|  |  |
| **Prof. Eduardas Bareiša**  Vadovas | (parašas) (data) |
|  |  |
| **Pareigų sutrumpinimas Vardenis Pavardenis**  Recenzentas / Recenzentė | (parašas) (data) |
|  |  |

**Kaunas, 2021**

****

**Kauno technologijos universitetas**

Informatikos fakultetas

Martynas Stumbra

**Projektų valdymo įrankis**

Akademinio sąžiningumo deklaracija

Patvirtinu, kad mano, Martyno Stumbros, baigiamasis projektas tema „Projektų valdymo įrankis“ 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) |

Stumbra, Martynas. Projektų valdymo įrankis. Bakalauro studijų baigiamasis projektas / vadovas Prof. Eduardas Bareiša; Kauno technologijos universitetas, Informatikos fakultetas.

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

Reikšminiai žodžiai: projektų valdymas, internetinis tinklapis, užduočių planavimas.

Kaunas, 2021. XX p.

Santrauka

Šiame darbe pristatomas projektų valdymo įrankis. Vienas iš svarbiausių aspektų, kuris lemia projekto sėkmę, yra veiksmingas užduočių planavimas. Atlikus užduotį laiku, neišeikvojus nustatytų projekto resursų, gerėja projekto kokybė. Kuriamas įrankis ne tik pagerins asmeninių ar komandinių projektų valdymo kokybę, bet ir individualaus vartotojo darbotvarkę. Taigi, pagrindinis darbo tikslas - supaprastinti ir pagerinti projektų valdymo ypatumus, o vartotojams padėti efektyviai planuoti savo darbą.

Analizuojant konkurentus buvo atsižvelgta į įrankių pranašumus ir trūkumus, pagal turimą funkcionalumą. Rasta, kad konkurentų įrankiai yra labai plačiai naudojami darbo rinkoje, tačiau turi vartotojo darbotvarkės planavimo funkcionalumo stoką. Reikalavimų analizės metu buvo apibrėžta įrankio apimtis, pasirinktos technologijos, kurių pagalba bus kuriamas įrankis, nustatomi sistemos naudotojai.

Įrankis sėkmingai realizuotas naudojant *Javascript* bei *Typescript* programavimo kalbas, *Apollo* infrastruktūrą, kuri naudoja *React* karkasą. Dalis įrankio yra padengta komponentų testais. Parengtas vartotojo vadovas, kuris leis vartotojams susipažinti su įrankio veikimu, diegimu bei palaikymu. Sukurtas įrankis yra patalpintas debesyse ir yra paruoštas naudojimui.

Darbo pabaigoje apibendrinami įrankio rezultatai bei pateikiamos išvados.

Stumbra, Martynas. Project Management Tool. Bachelor's  Final Degree Project / supervisor Prof., Eduardas Bareiša; Informatics Faculty, Kaunas University of Technology.

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

Keywords: project management, web application, task planning.

Kaunas, 2021. Number of pages.

Summary

This thesis introduces a project management tool. One of the more important aspects for the success of a project is successful task planning. Completing a task on time, without wasting the project's resources, improves the quality of the project itself. The system under development will not only improve the quality of personal or team project management, but also the individual user's agenda. Thus, the main objective of the work is to simplify and improve project management features and to help users to plan their agenda.

The competitor analysis took into account the strengths and weaknesses of the tools in terms of available functionality. The competitors' tools were found to be very widely used in the labour market, but there is a lack of functionality in the user's agenda planner. The requirements analysis defined the scope of the system, selected the technologies to be used for the development of the tool and identified the users of the system.

The system has been successfully implemented using *Javascript* and *Typescript* programming languages, *Apollo* infrastructure using *React* framework. Part of the system is covered by component tests. A user guide has been developed to allow users to familiarise themselves with the operation, installation and support of the tool. The developed tool is hosted in the cloud and is ready for use.

The thesis concldes with a summary of the system‘s results and conclusions.

Turinys

[Lentelių sąrašas 7](#_Toc70283990)

[Paveikslų sąrašas 8](#_Toc70283991)

[Santrumpų ir terminų sąrašas 9](#_Toc70283992)

[Įvadas 10](#_Toc70283993)

[1. Analizė 11](#_Toc70283994)

[1.1. Techninis pasiūlymas 11](#_Toc70283995)

[1.1.1. Sistemos apibrėžimas 11](#_Toc70283996)

[1.1.2. Bendras veiklos tikslas 11](#_Toc70283997)

[1.1.3. Sistemos pagrįstumas 11](#_Toc70283998)

[1.1.4. Konkurencija rinkoje 11](#_Toc70283999)

[1.1.5. Prototipai ir pagalbinė informacija 13](#_Toc70284000)

[1.1.6. Ištekliai, reikalingi sistemai sukurti 13](#_Toc70284001)

[1.2. Galimybių analizė 13](#_Toc70284002)

[1.2.1. Techninės galimybės 13](#_Toc70284003)

[1.2.2. Vartotojų pasiruošimo analizė 13](#_Toc70284004)

[2. Projektas 14](#_Toc70284005)

[2.1. Reikalavimų specifikacija 14](#_Toc70284006)

[2.1.1. Komercinė specifikacija 14](#_Toc70284007)

[2.1.2. Sistemos funkcijos 14](#_Toc70284008)

[2.1.3. Vartotojo sąsajos specifikacija 18](#_Toc70284009)

[2.1.4. Realizacijai keliami reikalavimai 18](#_Toc70284010)

[2.1.5. Techninė specifikacija 18](#_Toc70284011)

[2.2. Projektavimo metodai 19](#_Toc70284012)

[2.2.1. Projektavimo valdymas ir eiga 19](#_Toc70284013)

[2.2.2. Projektavimo technologija 19](#_Toc70284014)

[2.2.3. Programavimo kalbos, derinimo, automatizavimo priemonės, operacinė sistemos 19](#_Toc70284015)

[2.3. Sistemos projektas 19](#_Toc70284016)

[2.3.1. Statinis sistemos vaizdas 19](#_Toc70284017)

[2.3.2. Dinaminis sistemos vaizdas 20](#_Toc70284018)

[3. Testavimas 21](#_Toc70284019)

[3.1. Testavimo planas 21](#_Toc70284020)

[3.2. Testavimo kriterijai 21](#_Toc70284021)

[3.3. Komponentų testavimas 21](#_Toc70284022)

[3.4. Integracinis testavimas 21](#_Toc70284023)

[3.5. Vartotojo sąsajos testavimas 21](#_Toc70284024)

[4. Dokumentacija naudotojui 22](#_Toc70284025)

[4.1. Apibendrintas sistemos galimybių aprašymas 22](#_Toc70284026)

[4.2. Vartotojo vadovas 22](#_Toc70284027)

[4.3. Diegimo vadovas 22](#_Toc70284028)

[4.4. Administravimo vadovas 22](#_Toc70284029)

[Rezultatai ir išvados 23](#_Toc70284030)

[Literatūros sąrašas 24](#_Toc70284031)

Lentelių sąrašas

[**1 lentelė.** Konkurentų apžvalga 12](#_Toc70283967)

[**2 lentelė.** Pagrindiniai baigiamojo projekto stiliai ir jų aprašymai 25](#_Toc70283968)

Paveikslų sąrašas

[**2.1 pav.** Apibendrinta sistemos panaudos atvėjų diagrama 14](#_Toc70408789)

[**2.2 pav**. Vartotojų posistemės panaudos atvėjų diagrama 15](#_Toc70408790)

[**2.3 pav.** Grafikų posistemės panaudos atvėjų diagrama 15](#_Toc70408791)

[**2.4 pav.** Projektų valdymo posistemės panaudos atvėjų diagrama 16](#_Toc70408792)

[**2.5 pav.** Užduočių valdymo posistemės panaudos atvėjų diagrama 17](#_Toc70408793)

[**2.6 pav.** Lentos posistemės panaudos atvėjų diagrama 17](#_Toc70408794)

[**2.7 pav.** Sistemos diegimo diagrama 20](#_Toc70408795)

[**2.8 pav.** Sistemos esybių ryšių modelis 21](#_Toc70408796)

[**2.9 pav.** Sistemos paketų diagrama 22](#_Toc70408797)

[**2.10 pav.** Registracijos veiklos diagrama 23](#_Toc70408798)

[**2.11 pav.** Prisijungimo veiklos diagrama 24](#_Toc70408799)

[**2.12 pav.** Atsijungimo veiklos diagrama 25](#_Toc70408800)

[**2.13 pav**. Pakeisti aplikacijos kalbą veiklos diagrama 26](#_Toc70408801)

[**2.14 pav.** Priimti pakvietimą prisidėti prie projekto veiklos diagrama 27](#_Toc70408802)

[**2.15 pav.** Susieti projektą su Github veiklos diagrama 28](#_Toc70408803)

Santrumpų ir terminų sąrašas

**Santrumpos:**

**Prof.** – profesorius.

**API** (angl. *application programming interface*) – programų sąsaja, leidžianti programuotojui ar kitai sistemai pasiekti jos teikiamas funkcijas.

**HTTP(S)** (angl. *HyperText Transfer Protocol (Secure)*) – tai protokolas, naudojamas norint gauti informaciją pasauliniame tinkle (angl. *World Wide Web*).

**JSON** (angl. *JavaScript Object Notation*) – tai atviro standarto failų formatas, leidžiantis saugoti ir keistis objektų ar masyvų duomenimis žmogui skaitomu formatu.

**PA** – panaudojimo atvejis.

**UML** (angl. *Unified Modeling Language*) – modeliavimo ir specifikacijų kūrimo kalba, skirta specifikuoti, atvaizduoti ir konstruoti objektinių programų dokumentus.

**URL** (angl. *Uniform Resource Locator*) – nuoroda į tinklo resursą, nusakanti kelią iki jo ir mechanizmą jo gavimui.

**CRUD** (angl. *Create, read, update and delete*) - santrumpa, kuri reiškia duomenų kūrimą, skaitymą, atnaujinimą bei šalinimą.

**Terminai:**

**Kanban** – lankstaus (angl. *Agile*) projektų valdymo metodologija, besivadovaujantis komandos nustatytais laiko limitais, kurio pagalba užtikrinamas sklandus užduočių vykdymo procesas.

**Karkasas** (angl. *framework*) – programinės įrangos paketas su bibliotekomis, kurios palengvina pasikartojančių užduočių atlikimą.

**Scrum** – lankstaus (angl. *Agile*) projektų valdymo metodologija

Įvadas

Projektų valdymas yra viena iš svarbiausių įmonės veiklų. Ši sritis dažnu atveju lemia pačio projekto sėkmę. Valdant projektą svarbu įvertinti turimus resursus, nusistatyti užduoties terminą, užduotis prioretizuoti bei kategorizuoti. Atlikus įvertinimus svarbu tolygiai paskirstyti užduotis tarp komandos narių. Dažnai, projekto valdymo įrankių vartotojas savo darbo laiką gali planuoti tik lentos pagalba, ko pasekoje, vartotojui gali būti sudėtinga orientuotis laike ir numatyti savo dienos, savaitės ar mėnesio užduočių planą. Darbe kuriamas projektų valdymo įrankis ne tik leis vartotojui planuoti savo darbą lentos pagalba, bet ir suteiks naudingos statistikos bei naujausių įvykių įžvalgas. Įrankis bus orientuotas į programavimo veiklą vystančių projektų valdymą. Projekto valdytojai galės efektyviai valdyti savo produkto eigą, o darbuotojai - naudingai planuoti savo darbotvarkę.

Darbo tikslas - supaprastinti ir pagerinti projektų valdymo ypatumus, o vartotojams padėti efektyviau planuoti savo darbo laiką. Šiam tikslui pasiekti iškelti šeši uždaviniai:

1. Išanalizuoti rinkoje esančių konkurentų įrankius;
2. Atlikti reikalavimų analizę;
3. Remiantis analizės metu surinktais informacija, sudaryti įrankio projektą;
4. Realizuoti įrankį, užtikrinant, jog šis atitinkama keliamus reikalavimus;
5. Realizuotą įrankį ištestuoti;
6. Aprašyti realizuoto įrankio dokumentaciją.

Darbas yra sudaromas iš analizės, projektavimo, testavimo bei dokumentacijos skyrių. Analizės dalyje ištiriami esančių konkurentų įrankiai, apžvelgiamas kuriamo įrankio pagrįstumas bei tikslas. Projektavimo skyriuje pateikiami įrankio funkciniai bei nefunkciniai reikalavimai, kūrimo priemonės ir naudojamos technologijos, pateikiamas statinis bei dinaminis įrankio vaizdas. Testavimo skyriuje pateikiamas testavimo planas, apžvelgiami taikomi testavimo metodai, pateikiama testavimo eiga bei rezultatai. Dokumentacijoje pateikiamas vartotojo vadovas, nusakantis, kaip naudotis įrankiu. Dokumento pabaigoje pateikiamos išvados.

Kuriamas įrankis bus sudarytas iš dviejų pagrindinių dalių - serverinė dalis, kuri atsakinga už vartotojų bei projekto informacijos pasiekiamumą iš duomenų bazės, bei klientinė dalis - internetinė svetainė, kurios vartotojo sąsajos pagalba galima efektyviai bei patogiai valdyti įrankį. Verta paminėti, kad sistema palaiko anglų bei lietuvių kalbos parinktis. Įrankiui realizuoti buvo sukurta apie X kodo eilučių.

# Analizė

## Techninis pasiūlymas

### Sistemos apibrėžimas

Kuriama sistema yra skirta programavimo veiklą vystančių projektų valdymui. Norima, kad projekte dalyvaujantys vartotojai galėtų sėkmingai vystyti produktą bei efektyviai planuoti savo darbo laiką, bendradarbiauti su kolegomis atliekant užduotis. Vartotojui suteikiama galimybė matyti užduotis, statistiką, naujausius pasikeitimus jam priskirtose užduotyse bei sekti atliktus pakeitimus tam tikroje produkto versijoje.

### Bendras veiklos tikslas

Bendras veiklos tikslas - pagerinti vartotojų darbo našumą. Patogiai ir efektyviai planuojamas darbo laikas bei galėjimas numatyti dienos, savaitės ar mėnesio užduočių planą atneš naudos vartotojui, nes projekto vykdymui skirti resursai bus tinkamai paskirstyti. Šis įrankis atneš finansinės naudos vartotojams, nes dauguma plačiai naudojamų alternatyvių sistemų yra mokamos.

### Sistemos pagrįstumas

Moderniais laikais žmonėms laiko planavimas yra labai svarbus, nes jie nori optimaliai išnaudoti savo dienos laiką. Dauguma rinkoje esančių sistemų yra gan brangios vartotojui, turi per daug nereikalingo funkcionalumo, kas dažnai ne palengvina, o apsunkina darbo ir laiko planavimą. Be to įmonėms plečiant savo darbuotojų skaičių, didėja išlaidos tokiems projektų valdymo įrankiams licencijuoti. Tokiu atveju įmonės renkasi pigesnes ar nemokamas alternatyvas, tačiau jos turi minimalų funkcionalumą, ko dažniausiai nepakanka. Būtent šis kuriamas įrankis ir padėtų išspręsti šią problemą.

### Konkurencija rinkoje

Projektų valdymo įrankių yra nesuskaičiuojamas kiekis rinkoje. Kiekvienas iš jų yra unikalus, turintis išskirtinio funkcionalumo, kuris vienaip ar kitaip gerina projekto valdymą. Vieni iš populiariausių šio segmento įrankių yra: *Trello, Asana* bei *Jira*.

*Trello* tai 2011 metais ,,Fog Creek Software“ įmonės sukurta, vėliau *Atlassian* korporacijai parduotas Kanban stilistikos internetinė svetainė, kurios pagalba galima valdyti projektus. Šiuo įrankiu naudojasi vienos iš didžiausių pasaulio IT įmonių. Į sistema yra integruotos trečiųjų šalių įskiepiai, kaip *Slack*, *Dropbox* ar *Google Drive*. Įrankio kūrėjai teigia, kad šiuo metu juo naudojasi daugiau nei vienas milijonas komandų visame pasaulyje. Šiuo metu įrankis teikia trijų lygių planus: nemokamas planas, ,,*Bussiness Class*“ už $9.99 vartotojui per mėnesį, bei „*Enterprise*“ planą, kuris suteikia neribotą įrankio naudojimą, o mokestis yra individualiai sprendžiamas įmonių lygmenyje. Reikėtų atkreipti dėmesį, kad įrankį galima naudoti visiškai nemokamai, tačiau yra funkcionalumo apribojimų.

*Asana* - yra puikus tarpinis sprendimas, kuomet renkantis iš kitų, šiame darbe nurodomų, įrankių. Vartotojo sąsaja atrodo puikiai ir yra supaprastinta. Į projektus yra įdiegama tam tikra struktūra, todėl, kaip teigia įrankio kūrėjai, bus lengviau viską suvokti ir valdyti. Šis įrankis yra naudingas, jeigu dirbama prie vidutinio dydžio projekto, kuriame reikia atlikti daugiau nei 10 užduočių. Daugeliu atvejų ,,*Asana*“ yra labai panaši į ,,Trello“, tačiau pastarasis turi mažiau funkcionalumų bei trečiųjų šalių papildinių integracijos. Įrankis teikia du pagrindinius planus - nemokamą, skirtą komandoms iki 15 asmenų, bei mokamą planą už $12.11 vartotojui per mėnesį.

*Jira* - viena iš lyderių projektų valdymo srityje. Ypač paplitęs tarp programinės įrangos kūrėjų ir komandų. *JIRA* įrankis apima daugybę sričių ir leidžia įvairiais būdais valdyti darbą. Sistema sukurta Agile metodologiją praktikuojančių komandų darbui valdyti. *Jira* pagalba galime naudotis ,,Scrum“, „Kanban“ lentomis, ,,Agile“ ataskaitų rengimu, projektų bei jo problemų stebėjimu, prioritetų nustatymu, iteracijų planavimu bei daugybės kitų funkcijų. Sistemos naudojimo kaina priklauso nuo komandos dydžio. Mažoms komandoms nuo vieno iki dešimt darbuotojų metinė įrankio licencija kainuotų $100, o komandoms nuo 201 iki 300 darbuotojų - $15,050. Įrankiui įvaldyti reikia praktikos, kad pilnai išnaudoti jo teikiamas galimybes.

**1 lentelė.** Konkurentų apžvalga

| **Lyginimo kriterijai** | **Trello** | **Asana** | **Jira** | **Kuriama sistema ,,Projectus“** |
| --- | --- | --- | --- | --- |
| Vartotojų kiekis | ~ 50 mln. vartotojų | ~ 1,3 mln. vartotojų | ~ 180 tūkst. klientų | Dar nepublikuota |
| Licencijos kaina | Nemokamas planas, ,,*Bussiness Class*“ už $9.99 vartotojui per mėnesį, bei „*Enterprise*“ planas, kurio mokestis yra individualiai sprendžiamas įmonių lygmenyje | Nemokamas planas mokamas planas už $12.11 vartotojui per mėnesį | Komandoms nuo vieno iki dešimt darbuotojų metinė įrankio licencija kainuotų $100, o komandoms nuo 201 iki 300 darbuotojų - $15,050 | Nemokama |
| Internetinis puslapis | Yra | Yra | Yra | Yra |
| Mobilioji aplikacija | Yra | Yra | Yra | Nėra |
| Integracija su trečiųjų šalių papildiniais | Yra | Yra | Yra | Yra |
| Kanban metodologijos taikymas | Yra | Yra | Yra | Yra |
| Užduočių prioretizavimas | Yra, bet prioretizavimas vyksta antraščių kūrimu | Yra | Yra | Yra |
| Užduočių kategorizavimas | Yra, bet kategorizavimas vyksta antraščių kūrimu | Yra, bet kategorizavimas vyksta antraščių kūrimu | Yra | Yra |
| Kalbos parinktis | Nėra | 10 skirtingų kalbos parinkčių | Nėra | Anglų bei Lietuvių kalbos parinktys |
| Individuali statistika | Nėra | Nėra | Nėra | Yra |

Atlikus konkurentų apžvalga, galime teigti, kad sistemos sprendžia tą pačią problemą, tačiau pastebėta, kad konkurentų įrankiai yra gan brangūs. Individuali statistika yra tik mokamose sistemose, o nemokamose jos nėra išvis. Konkurentai turi net tik internetinį puslapį, bet ir mobiliąją aplikaciją, tokiu būdu yra pasiekiama didesnė vartotojų aprėptis.

Remiantis mano asmenine patirtimi, didžiausi pastebėti trūkumai yra tai, kad konkurentų įrankių vartotojo sąsaja yra gan komplikuota, ko pasekoje, įrankio naudojimas yra apsunkinamas. Taip pat pastebėta, kad Jira įrankis, atliekant tam tikrus veiksmus yra gan lėtas.

### Prototipai ir pagalbinė informacija

Sistema buvo kuriama nenaudojant jokių prototipų.

Buvo naudotasi Semantic-UI biblioteka, kurios pagalba buvo galima greičiau bei estetiškiau sukurti vartotojo sąsaja.

### Ištekliai, reikalingi sistemai sukurti

Žiūrint į kuriamos sistemos apimtį bei funkcionalumą, jai sukurti reikės vieno žmogaus. Pagal techninius išteklius, reikės bent vieno kompiuterio, kuris palaikytų *Windows* arba *MacOS* operacinę sistemą. Sistemos kodui saugoti reikės dviejų atskirų kodo talpyklų *Github* įrankyje, serverinei bei klientinei daliai saugoti.

## Galimybių analizė

### Techninės galimybės

Projektas bus realizuotas naudojantis trimis, skirtingomis duomenų bazėmis. Vartotojų bei projektų informacijai saugoti bus naudojama *MongoDB* duomenų bazė. *Redis* duomenų bazė, bus atsakinga už sesijų žetonų saugojimą, o *Google Cloud Storage* saugykla saugos vartotojų profilių nuotraukas.

Šiuo metu sistema vis dar vystoma ir yra pasiekama per *Netlify* viešą*,* sugeneruotą nuorodą. Kuomet sistema bus paruošta paleidimui į rinką, tikėtina, kad nemokamų duomenų bazių paslaugų neužteks dėl didėjančio vartotojų kiekio, todėl reikėtų įsigyti geresnius šių duomenų bazių planus.

Šiuo metu pasirinktos duomenų bazės turi nustatytus limitus, kuriuos viršijus tektų sumokėti už kiekvieną papildomą užklausą. Tai gali sudaryti kliūčių tolimesniam sistemos išlaikymui.

### Vartotojų pasiruošimo analizė

Įrankio tikslinė auditorija - pavieniai asmenys arba įmonės, kurios užsiima programinės įrangos kūrimu. Sistemos vartotojo sąsaja kurta su mintimi, kad bet kuris vartotojas, turintis IT žinių arba patirties su panašiais įrankiais, galėtų lengvai bet ir maksimaliai ją išnaudoti.

# Projektas

## Reikalavimų specifikacija

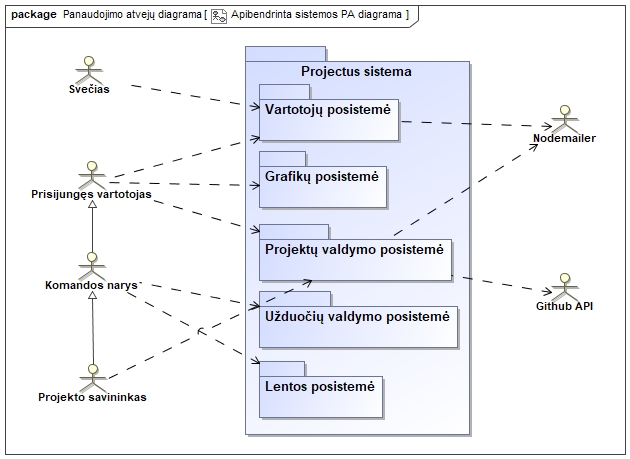
### Komercinė specifikacija

Šis įrankis yra universitetinis projektas, kuris kuriamas bakalauro baigiamajam darbui. Projekto užsakovas yra bakalauro darbo vadovas prof. Eduardas Bareiša, o jo vykdytojas - universiteto studentas Martynas Stumbra. Įrankio vartotojai - tai pavieniai asmenys arba įmonės, kurios užsiima programinės įrangos kūrimu.

Projektas yra asmeninis, todėl biudžeto neturi, o jo numatoma baigimo data - 2021 m. gegužės 4 d.

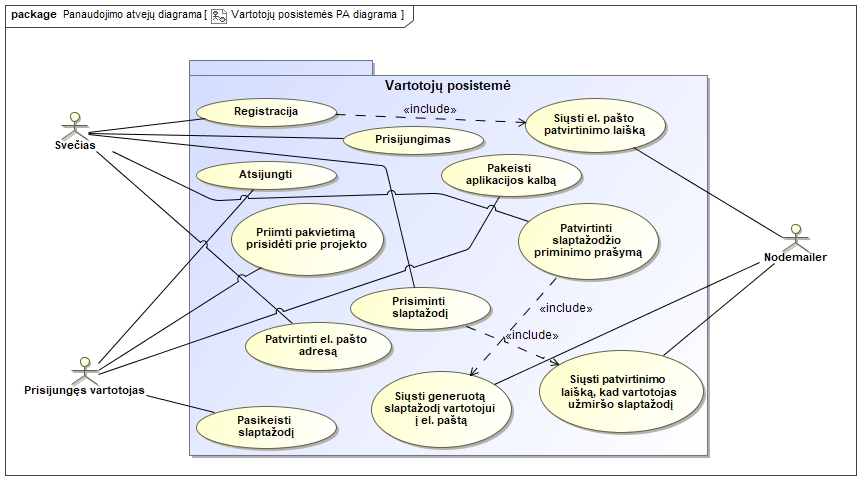
### Sistemos funkcijos

Žemiau aprašomi sistemos funkciniai reikalavimai, pavaizduojamos UML panaudos atvėjų diagramos.



**2.1 pav.** Apibendrinta sistemos panaudos atvėjų diagrama

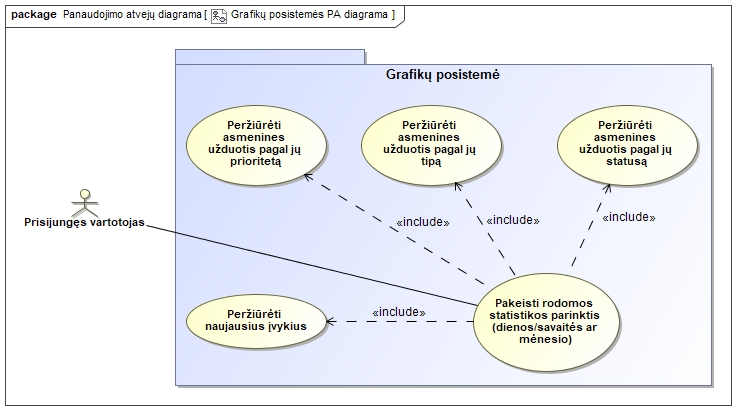
Apibendrintoje sistemos panaudos atvėjų diagramoje (2.1 pav.) matome pagrindinę kuriamą sistemą bei joje esančias penkias posistemes: vartotojų, grafikų, projektų, užduočių valdymo bei lentos. Sistemoje išskirtos keturios pagrindinės aktorių rolės: svečias, prisijungęs vartotojas, komandos narys bei projekto savininkas bei dvi, trečiųjų šalių funkcionalumams integruoti reikalingos – Nodemailer bei Github API aktorių rolės. Detalesnės posistemių diagramos pateikiamos žemiau.



**2.2 pav**. Vartotojų posistemės panaudos atvėjų diagrama

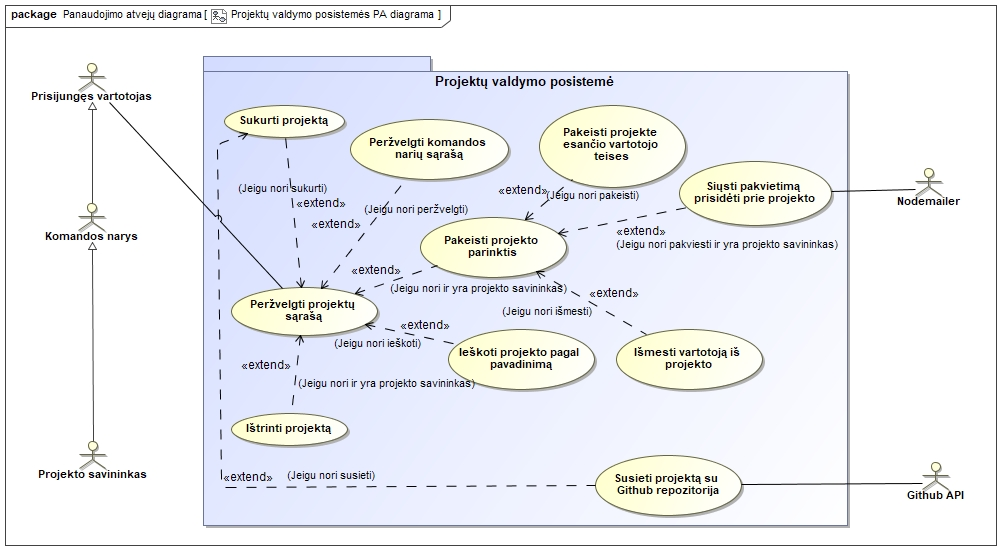
Ši diagrama (2.2 pav.) vaizduoja vartotojų posistemės funkcionalumus. Svečias gali registruotis, patvirtini savo el. pašto adresą registracijos metu, prisijungti bei prisiminti slaptažodį. Prisijungęs vartotojas gali priimti kvietimą prisidėti prie projekto, pakeisti aplikacijos kalbą, pasikeisti slaptažodį bei atsijungti.

Kuomet svečias atlieka registraciją, prašo priminti slaptažodį arba patvirtino slaptažodžio priminimo prašymą, Nodemailer siunčia laišką į vartotojo nurodytą kontaktinį el. pašto adresą.



**2.3 pav.** Grafikų posistemės panaudos atvėjų diagrama

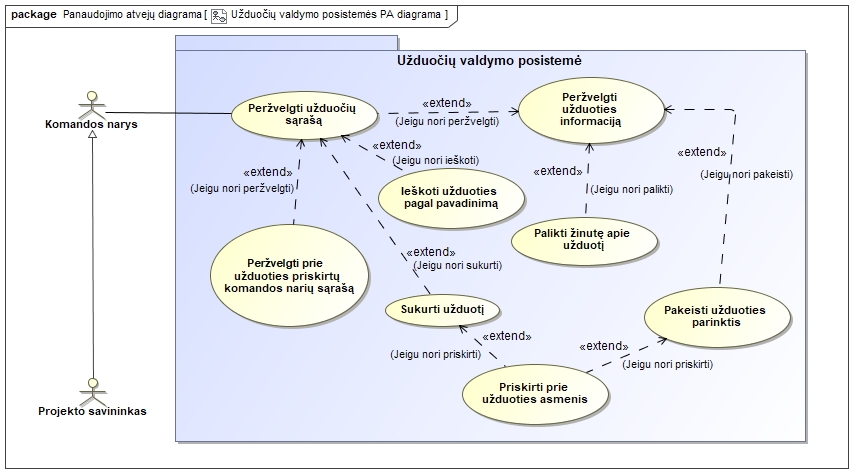
Šioje diagramoje (2.3 pav.) matome grafikų posistemės funkcionalumus, kurie teikia suasmenintą informaciją apie vartotojo darbus. Prisijungęs vartotojas gali pakeisti rodomos statistikos parinktis į dienos, rytojaus, savaitės ar mėnesio, o taip pat ir peržvelgti tos statistikos informaciją apie užduočių prioritetus, tipus, statusus bei naujausius įvykius, susijusius su užduotimis.



**2.4 pav.** Projektų valdymo posistemės panaudos atvėjų diagrama

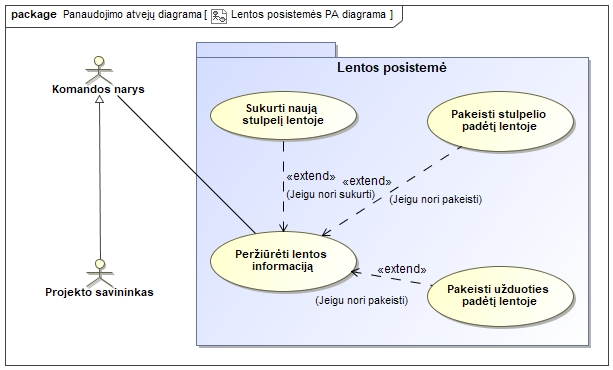
Diagramoje (2.4 pav.) vaizduojami projektų valdymo posistemės galimi funkcionalumai. Projekto savininkas, komandos narys bei svečias gali peržvelgti projektų sąrašą, projektuose esančių komandos narių sąrašą, bei ieškoti konkretaus projekto pagal pavadinimą. Projekto vadovas gali sukurti projektą ir jeigu nori jį susieti su Github repozitorija. Taip pat, projekto savininkas gali ištrinti projektą, keisti jo parinktis, pakeisti projekte esančio komandos nario teises, pakviesti prisidėti vartotoją prie projekto arba jį išmesti iš komandos narių.

Kai vartotojas yra pridedamas prie projekto komandos, Nodemailer siunčia pakvietimą į pakviesto vartotojo kontaktinį el. pašto adresą.



**2.5 pav.** Užduočių valdymo posistemės panaudos atvėjų diagrama

Šioje diagramoje (2.5 pav.) matome užduočių valdymo posistemės panaudos atvėjus. Projekto savininkas kaip ir komandos narys gali peržvelgti užduočių sąrašą, prie užduoties priskirtų komandos narių sąrašą, ieškoti užduoties pagal pavadinimą, sukurti užduotį, peržiūrėti detalesnę užduoties informaciją, palikti užduotyje žinutę, pakeisti užduoties parinktis arba priskirti prie užduoties komandos narius.



**2.6 pav.** Lentos posistemės panaudos atvėjų diagrama

Diagramoje (2.6 pav.) vaizduojami lentos posistemės panaudos atvėjai. Projekto savininkas kaip ir komandos narys gali peržiūrėti lentos informaciją, sukurti naują stulpelį, koreguoti užduoties arba stulpelio padėtį lentoje.

Iš pateiktų panaudos atvėjų diagramų matome, kad naudojamas Nodemailer, kuris atsakingas už el. paštu siunčiamus laiškus į vartotojo kontaktinį el. pašto adresą. Šio aktoriaus integracija sistemoje pagelbės užtikrinti saugų sistemos naudojimą ne tik eiliniam vartotojui bei ir sistemos kūrėjui. Dauguma su projektu susijusiu valdymo funkcijų, kaip redaguoti ar pašalinti, bus leidžiama tik projekto savininko teises turinčiam vartotojui projekto lygmenyje.

Kuomet yra kuriamas projektas ir jis yra susiejamas su egzistuojančia, viešai prieinama repozitorija, automatiniu būdu projekto lentoje yra sukuriami stulpeliai su produkto versijų žymomis. Laikui bėgant ir išleidus naują versiją, projekto lentoje bus sukuriamas naujas stulpelis su atitinkamu versijos pavadinimu.

### Vartotojo sąsajos specifikacija

Jokie reikalavimai grafinei vartotojo sąsajai nebuvo iškelti.

### Realizacijai keliami reikalavimai

Kuriamam įrankiui taikomi šie nefunkciniai reikalavimai:

1. Realizacijos:
   1. Duomenų validacija turi būti atliekama aplikacijų programavimo ir vartotojo sąsajose;
   2. Vartotojų profilių nuotraukos saugomos Google Cloud Storage saugykloje.
2. Saugumo:
   1. Sesijos turi būti saugomos serverinėje dalyje, slapuko pavidalu;
   2. Slapukas negali būti saugomas klientinėje dalyje;
   3. Visi serverio API metodai privalo būti apsaugoti ir pasiekiami tik autorizuotų vartotojų;
   4. Užtikrinamas saugus duomenų perdavimas internetu (aptarnaujamos tik HTTPS užklausos);
3. Sistemos priežiūrai:
   1. Kodo versijavimui naudoti Git versijų kontrolės sąsaja Github;
   2. Trigdžių sekimui naudoti šiame darbe sukurtą įrankį.
4. Kultūriniai - politiniai:
   1. Įrankio vartotojo sąsaja privalo palaikyti anglų bei lietuvių kalbos parinktis.
5. Sistemos išvaizdai:
   1. Vartotojo patirčiai pagerinti privalo būti bent kelios animacijos, kurios pagerintų vartotojo sąsają.
   2. Interneto svetainė turi būti realizuota su prisitaikančio dizaino metodika.

### Techninė specifikacija

Vartotojai norintys naudotis įrankiu privalo turėti įrenginį, kuris turi interneto prieigą ir interneto naršyklę.

## Projektavimo metodai

### Projektavimo valdymas ir eiga

Įrankio kūrimo metu buvo taikoma Agile projektų valdymo metodologija – Scrum. Taikant šią metodologiją buvo galima susiplanuoti darbus, skaidant juos į iteracijas. Pasirinkta 10 vienos savaitės trukmės iteracijų. Kiekviena iš šių iteracijų buvo sutelkta į nurodytų posistemių kūrimą. Kiekvienos iteracijos pabaigoje, atlikus sistemos peržiūrą, buvo planuojami sekančios iteracijos darbai bei taisomi sistemoje aptikti trigdžiai. Paskutinė iteracija buvo skirta vartotojo sąsajos tobulinimui bei klaidų aptikimui ir taisymui.

### Projektavimo technologija

Projektavimo darbams buvo naudojamas *MagicDraw* įrankis. Jo pagalba buvo braižomos UML diagramos: panaudos atvėjų, diegimo, paketų, esybių-ryšių modelis, veiklos bei sekų diagramos.

Sistemos kūrimo pradžioje užduočių sėkimui buvo naudojama Trello platforma, tačiau pasiekus paskutinę iteraciją buvo pereita į sukurtos sistemos įrankį. Tokiu būdu buvo galima efektyviai aptikti trigdžius kuriamoje sistemoje, juos šalinti bei įrašyti praleistų valandų kiekius.

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

Sistema kurta remiantis *Apollo* infrastruktūra. Serverinei įrankio daliai kurti buvo naudotasi Javascript programvimo kalba, o klientinei daliai - Typescript. Klientinei daliai realizuoti buvo naudojamas React karkasas su Apollo Client biblioteka. Sistemos programavimui buvo naudojamas MacOS operacinę sistemą palaikantis kompiuteris, nes tik šios operacinės sistemos pagalba buvo galima testuoti kaip veikia įrankis Safari internetinėje naršyklėje. Programinio kodo rašymui naudotas *Visual Code Studio* kodo redaktorius. Internetinio puslapio testavimui buvo naudojami *Chrome Developer Tools* bei  *Safari Developer Tools* papildiniai. Užklausų į programavimo sąsaja derinimui naudotas *Altair* įrankis.

## Sistemos projektas

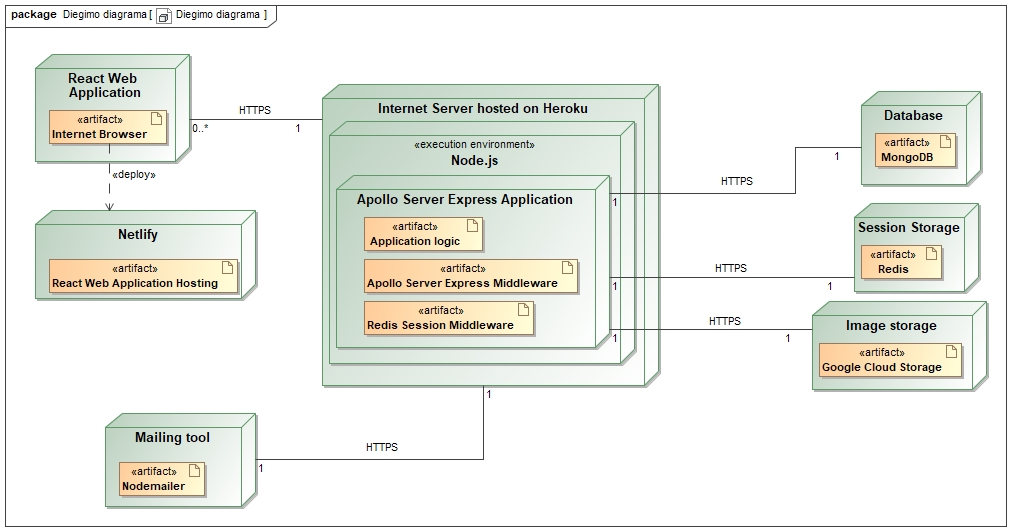
### Statinis sistemos vaizdas

2.7 pav. pateikiama sistemos diegimo diagrama, kurios pagalba galima tinkamai suprasti statinį sistemos vaizdą. Iš šios diagramos galime matyti, kad klientinė dalis yra *Netlify* įrankio pagalba patalpinta į debesį, o serverinė dalis patalpinta į Heroku debesį.

Interneto serveris yra sujungtas su laišku siuntimo servizu – *Nodemailer*, šis servizas siunčia elektroninius laiškus į vartotojų kontaktinius el. pašto adresus. *Google Cloud Storage* saugykla naudojama vartotojų profilio nuotraukų talpinimui debesyje. Vartotojų profilio nuotraukos yra dinamiškai generuojamos *Jdenticon* bibliotekos dėka, sukurtos nuotraukos yra išsaugomos failų sistemoje ir patalpinamos į *Google Cloud Storage* talpyklą.

Sesijų žetonai yra saugomi *Redis* duomenų bazėje. Būtent ši duomenų bazė buvo pasirinkta sesijų talpinimui, kadangi praėjus sesijos žetono galiojimo laikui šis yra automatiškai sunaikinamas, o serveriui nereikia papildomai šios įvykio apdoroti, o rezultate nuimamas krūvis nuo kuriamos aplikacijų programvimo sąsajos.

Pagrindinė duomenų bazė *MongoDB* atsakinga už vartotojų, projektų bei su projektai susijusių užduočių informacijos saugojimui.

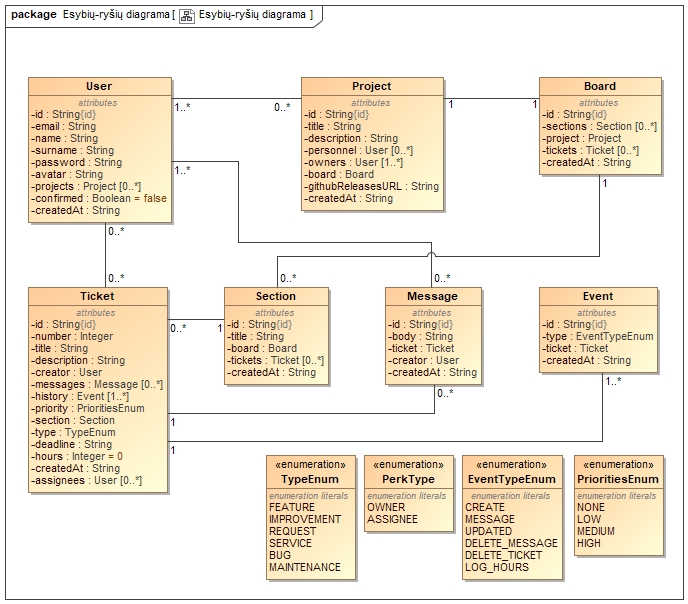


**2.7 pav.** Sistemos diegimo diagrama

*MongoDB* duomenų bazės struktūrai tinkamai pavaizduoti pasirinktas esybių ryšių modelis (2.8 pav.). Išskirti keturi enumeratorių tipai: *TypeEnum*, *PerkType*, *EventTypeEnum* bei *PrioritiesEnum*. Visi nurodyti enumeratoriai yra naudojami *Ticket* ir *Event* klasėse. Enumeratorių reikšmės gali ilgainiui kisti, kadangi plečiant sistema jų bus pridedama daugiau arba atitinkamai mažinama pagal sistemoje numatomus pokyčius.

Kiekviena, iš diagramoje pateiktų esybių privalo turėti unikalų atributą – id, kuris nusako saugomo objekto unikalių simbolių seka. Duomenų bazėje id atributas konvertuojamas į *MongoDB* traktuojama *ObjectId* tipą, šios konvertacijos bei *mongoose* bibliotekos pagalba galima lengviau valdyti duomenų bazėje esančius objektus.

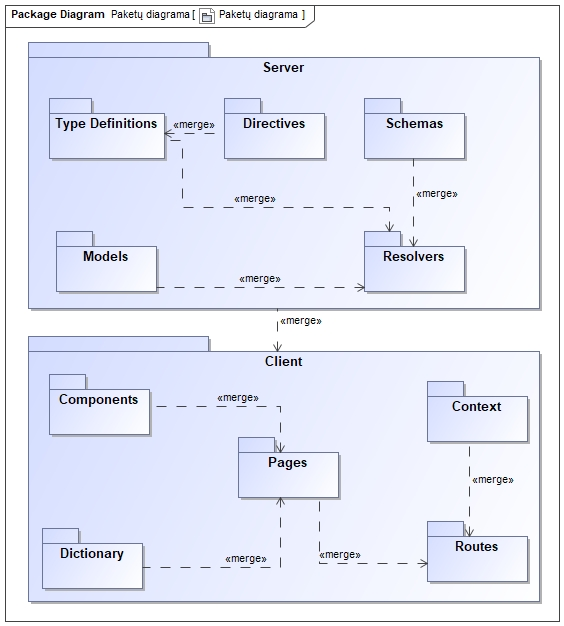
Modeliuojant duomenų bazės esybių ryšius nuspręsta taikyti *MongoDB* nuorodos (*angl. referencing*) modelį. Šio modeliavimo dėka yra taupomi sub-dokumentų informacijos talpa, kuri pagal *MongoDB* standartus negali viršyti 16MB. Taipogi, nuorodų modeliavimo pagalba yra palengvinamas objektų praplėtimas, kadangi saugomi išoriniai raktai apie kitus objektus tiesiogiai turi nuorodą į tam tikrą objektą duomenų struktūroje.



**2.8 pav.** Sistemos esybių ryšių modelis

Pagal pateiktą paketų diagrama (2.9 pav.) galima matyti kaip sistema yra išgrupuota paketais. Sistema sudaryta iš dviejų atskirų paketų: serverinė bei klientinė dalys. Serverinėje sistemos pakate yra išskaidomi dar penki paketai: *Type Definitions*, *Directives*, *Schemas*, *Models* ir *Resolvers.*

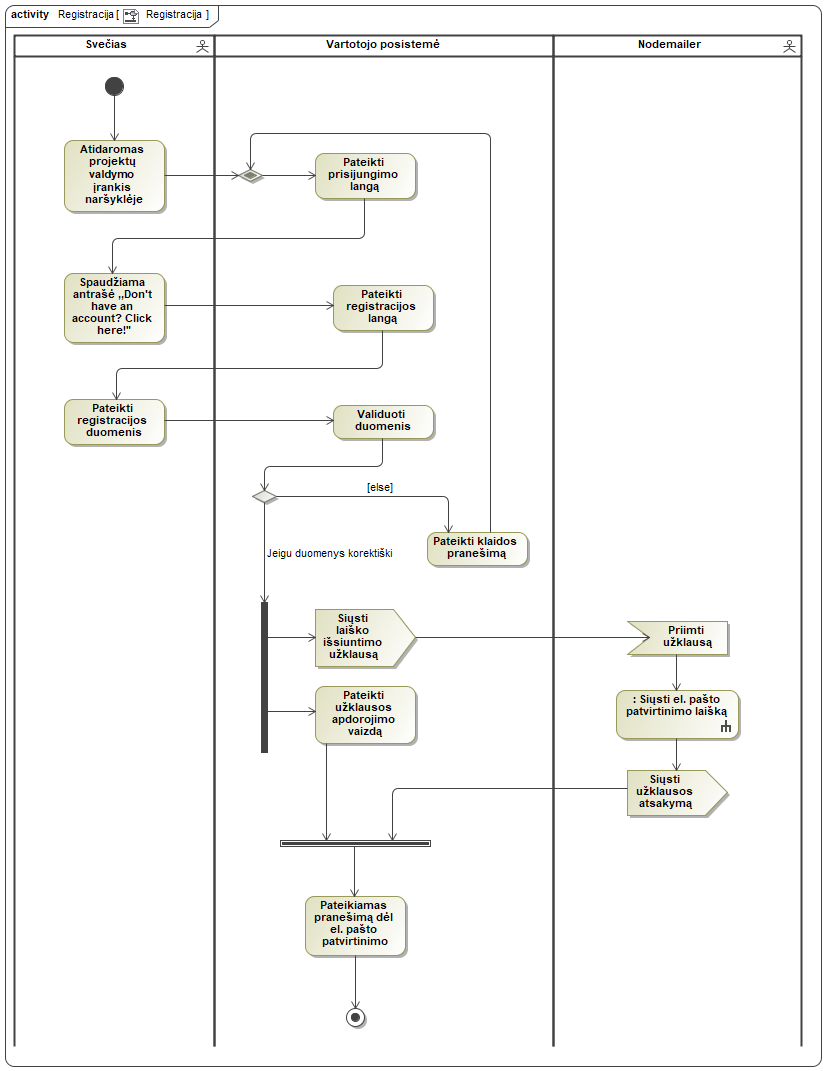
*Type Definitions* paketas skirtas GraphQL objektų tipams, jų galimų užklausų bei mutacijų aprašymas pateikti, Taip pat, šio paketo dėka yra automatiškai sukuriama API naudojimosi dokumentacija. *Directives* paketas reikalingas norint apsaugoti užklausų bei mutacijų metodus. Sistemoje numatomi du apribojimai susiję su direktyvomis – svečias bei autentifikuotas vartotojas. *Schemas* naudojamos validuoti į užklausas bei mutacijas pateikiamus parametrus. Parametrų validacijai naudojama *Yup biblioteka. Models* pakete yra aprašomi duomenų bazės objektai ir jų atributai. *Resolvers* pakete yra aprašomas užklausų bei mutacijų metodų atliekami veiksmai.



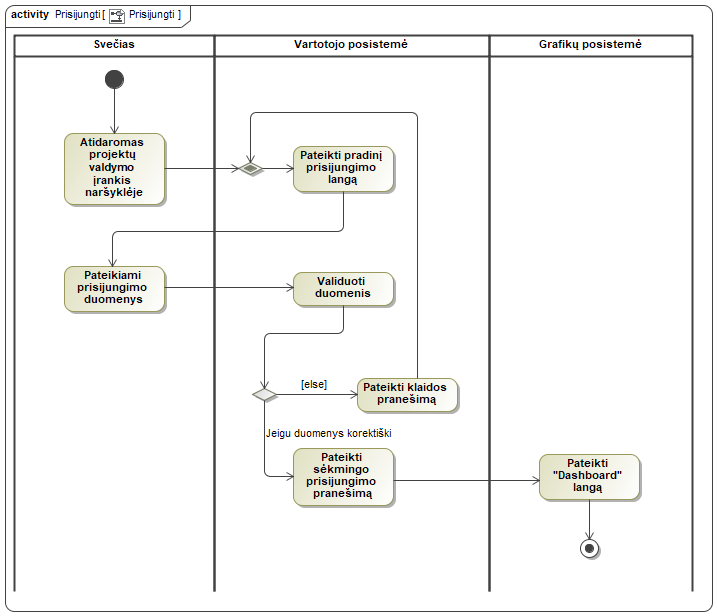
**2.9 pav.** Sistemos paketų diagrama

### Dinaminis sistemos vaizdas

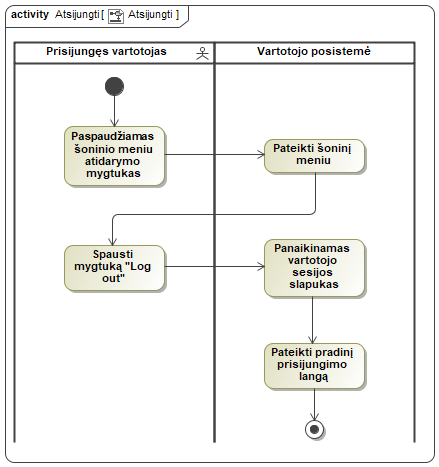
Šiame skyriuje bus pateikiamas sistemos dinaminis vaizdas. Pertekti vaizdą naudosiu veiklos diagramas. Iš viso nubraižyta trisdešimt viena veiklos diagrama. Pasirinktos braižymui tik tie panaudojimo atvėjai, kurie teikia pagrindinį įrankio funkcionalumą.



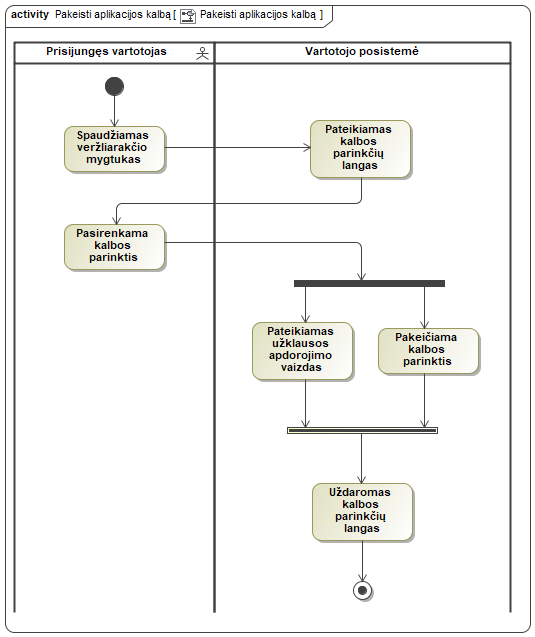
**2.10 pav.** Registracijos veiklos diagrama



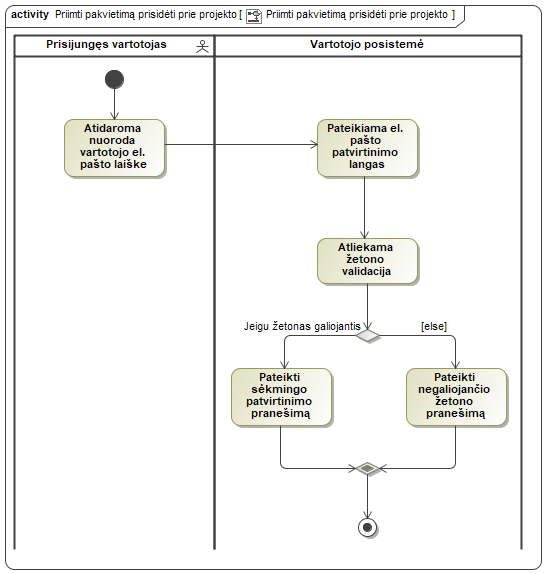
**2.11 pav.** Prisijungimo veiklos diagrama



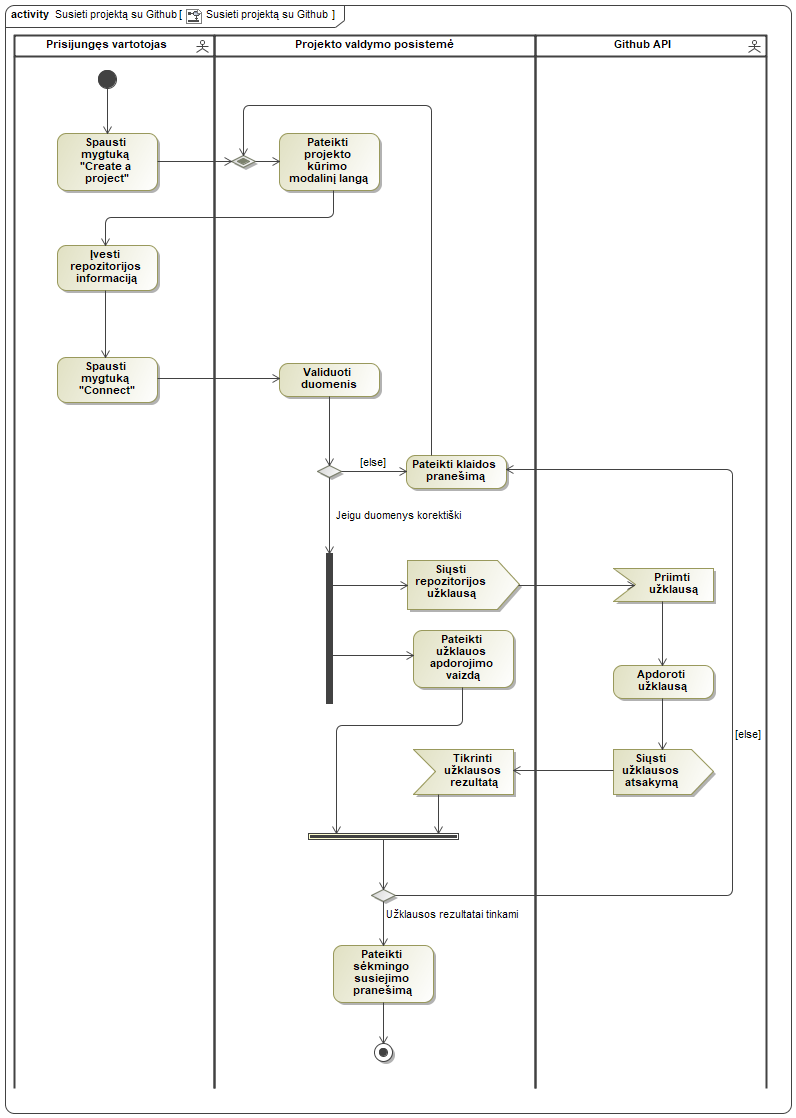
**2.12 pav.** Atsijungimo veiklos diagrama



**2.13 pav**. Pakeisti aplikacijos kalbą veiklos diagrama



**2.14 pav.** Priimti pakvietimą prisidėti prie projekto veiklos diagrama



**2.15 pav.** Susieti projektą su Github veiklos diagrama

# Testavimas

Aprašoma su sukurtos įrangos testavimu susijusi informacija (8 – 12 lapai). Skyriaus struktūra ir pavadinimas priklauso nuo baigiamojo darbo specializacijos ir pačios temos specifikos.

Nurodomas įrangos testavimo planas, testavimo duomenų rinkiniai ir gauti rezultatai. Nurodoma sistemos specifikacija ir sąlygos, prie kurių buvo atliekamas testavimas.

## Testavimo planas

Testavimo planas – tai jūsų pasirinkta testų atlikimo tvarka. Galimas testavimo planas: komponentų testavimas, po kurio seka integracinis testavimas, o vėliau būna sąsajos testavimas.

## Testavimo kriterijai

Šiame punkte aprašykite kriterijus, kurie jums buvo svarbūs testavimo metu. Tai gali būti ne tik informacijos ar skaičiavimų korektiškumas, bet ir kodo pertekliškumo analizė, informacijos perdavimo laikas, sistemos atitikimas funkciniams ir nefunkciniams reikalavimams.

## Komponentų testavimas

Šiame punkte reiktų aprašyti kokiais metodais testavote smulkias programos dalis (žr. wiki *unit testing*). Komponentų testavimas privalo būti atliekamas naudojant automatines testavimo priemones.

## Integracinis testavimas

Jei kurdami sistemą atlikote integracinį testavimą, jį aprašykite šiame skyrelyje. Integracinis testavimas privalo būti atliekamas naudojant automatines testavimo priemones.

## Vartotojo sąsajos testavimas

Šiame punkte reiktų aprašyti kokiais metodais testavote vartotojo sąsają. Dažniausiai pasitaikantis metodas – „rankinis“, t.y. kai sąsaja testuojama vartotojui (testuotojui) bandant atsitiktinai ar pagal scenarijų spaudyti mygtukus, įvedinėti tekstą į laukus ir kt. Kur kas geresnis variantas tuomet, kai testuojama automatiškai – pavyzdžiui, sukuriama programa ar testavimo tvarkyklė, kuris spaudymo ar įvedimo veiksmus atlieka be vartotojo įsikišimo. Panaudotas automatinis testavimas, dažniausiai, papildomai (teigiamai) įvertinamas baigiamojo darbo gynimo metu. Pasinaudokite automatizavimo priemonėmis, tokiomis kaip **Selenium IDE**.

# Dokumentacija naudotojui

Dokumento dalis, skirta naudotojui, kur aprašomas visas naudotojui aktualus programinės (aparatūrinės) įrangos funkcionalumas (4 – 10 lapų).

Dokumentacija naudotojui – tai instrukcija kaip naudotis sistema. Dokumentacijoje turi būti aiškiai aprašyti naudojimosi sistema ypatumai, pradedant diegimu ir baigiant įprastinėmis funkcijomis. Rašydami dokumentaciją atsižvelkite į naudojamą terminologiją. Pavyzdžiui, jei sistemą instaliuos administratorius, o naudos paprasti vartotojai, pastarųjų stenkitės neapkrauti sudėtingesnėmis sąvokomis.

## 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

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.

## Diegimo vadovas

Sistemos diegimo dokumentas yra skiriamas sistemos administratoriams (dažniausiai tai kompiuterius prižiūrintis personalas, tačiau šie žmonės nebūtinai būna ir sistemos naudotojai). Jame turi būti nurodytos diegimo konkrečioje aplinkoje detalės, turi būti supažindinama su sistemą sudarančiais failais, minimalia reikalingos techninės įrangos konfigūracija.

## Administravimo vadovas

Sistemos administratoriaus vadove turi būti aprašyti pranešimai, kaip sistema bendrauja su kitomis sistemomis ir kaip reaguoti į šiuos pranešimus. Būtų gerai nurodyti, kaip reaguoti į sistemos klaidas (sisteminių pranešimų paaiškinimai). Jei sistema apima ir techninę įrangą, jame turi būti aprašyti operatoriaus veiksmai palaikant šią techninę įrangą (pvz., kaip prijungti naujus periferinius įrenginius ir t.t.).

Rezultatai ir išvados

Bene svarbiausia viso darbo dalis – išvados. Išvados nenurodo, kas buvo padaryta darbe, bet pabrėžia atrastus dėsningumus, pastebėtas technologijų ar rinkos spragas, esminius įrangos privalumus. Išvados gali būti formuluojamos tik darbo metu sukurtos įrangos, technologijos, metodo ar susistemintos informacijos pagrindu (pvz., negalima cituoti šaltinių, vadovautis kitų autorių atrastais dėsningumais). Išvados numeruojamos, jų turėtų būti maždaug 4-9 (pvz., kiekvienam kūrimo etapui – reikalavimų analizei, projektavimui, realizacijai, testavimui, diegimui). Įprastai kiekviena išvada turėtų būti sudaryta iš atlikto veiksmo aprašymo ir gautų rezultatų. Išvadas galima gauti:

* Atlikus konkurentų analizę, kuomet būna išsiaiškinama esminiai konkurentų sistemų pranašumai ir trūkumai (pvz., „Buvo išanalizuotos analogiškos (konkrečiai nurodant kokios) sistemos, kurios pasižymėjo tokiais ir tokiais privalumais (apibendrintai), tačiau dėl tokių ar anokių trūkumų buvo nuspręsta kurti naują sistemą...“).
* Atlikus technologijų analizę, kuomet būna pagrindžiamas konkrečių programavimo kalbų, karkasų ar kitų technologijų pasirinkimas (pvz., „Išanalizavus x, y ir z technologijas buvo pasirinkta technologija z. Tai padėjo lengviau suprojektuoti, o vėliau ir realizuoti įrankio serverio pusės dalį, palaikyti vientisą programos kodo struktūrą...“).
* Atlikus testavimą, kuomet būna nurodoma kokį kodo padengimą pavyko pasiekti, kokias klaidas pavyko aptikti panaudojus pasirinktus testavimo metodus.
* Susidūrus su tam tikromis specifinėmis problemomis, kurioms išspręsti buvo panaudotas jūsų sugalvotas metodas („Kūrimo metu buvo susidurta su tokiomis ar anokiomis problemomis, kurios buvo sprendžiamos taip arba anaip...“). Galima įdėti ir išvadą apie nepasiteisinusius, tačiau jūsų išbandytus sprendimus (siekiant, kad kiti „neliptų ant to paties grėblio“). Jūsų parinkti problemų sprendimo būdai yra svarbios išvados, parodančios jūsų kompetenciją ir įsigilinimą į darbą.
* Realizavus pačią programą ar sistemą, kuri (greičiausiai) pakeitė ar pagerino iki tol vykusius verslo procesus (tai susiję su skyreliais „Bendras veiklos tikslas“ ir „Sistemos pagrindimas“) ar (jei tai buvo mokslinio pobūdžio darbas) tiesiog iki tol buvusius algoritmo / sprendimo rezultatus.

Šiame skyrelyje taip pat būtina pridėti ir papildomas išvadas-rezultatus apie tai:

* Kokia yra sistemos esamą būklė. Verta paminėti, jei sistema yra praktiškai naudojama įmonėje ar (programėlės kūrimo atveju) programėlė yra įkelta į Google Play ar AppStore parduotuvę.
* Kas planuojama atlikti tobulinant sistemą ateityje. Kadangi baigiamajam darbui sukurti yra skiriamas ribotas laikas, galbūt verta paminėti tas savybes, kurių dėl laiko apribojimų tiesiog nespėjote, bet planuojate įgyvendinti.

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.

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)“.