****

**Kauno technologijos universitetas**

Informatikos fakultetas

**Įvykių ir renginių sklaidos mobilioji programėlė**

Baigiamasis bakalauro studijų projektas

|  |
| --- |
|  |
| **Paulius Tomas Kalverš**  Projekto autorius |
|  |
| **Prof. Eduardas Bareiša**  Vadovas |
|  |

**Kaunas, 2021**

****

**Kauno technologijos universitetas**

Informatikos fakultetas

**Įvykių ir renginių sklaidos mobilioji programėlė**

Baigiamasis bakalauro studijų projektas

Programų sistemos (612I30002)

|  |  |
| --- | --- |
|  |  |
| **Paulius Tomas Kalverš**  Projekto autorius | (parašas) (data) |
|  |  |
| **Prof. Eduardas Bareiša**  Vadovas | (parašas) (data) |
|  |  |
| **Rasa Mažutienė**  Recenzentė | (parašas) (data) |
|  |  |

**Kaunas, 2021**

****

**Kauno technologijos universitetas**

Informatikos fakultetas

Paulius Tomas Kalverš

**Įvykių ir renginių sklaidos mobilioji programėlė**

Akademinio sąžiningumo deklaracija

Patvirtinu, kad mano, Pauliaus Tomo Kalverš, baigiamasis projektas tema „Įvykių ir renginių sklaidos mobilioji programėlė“ 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) |

Kalverš, Paulius Tomas. Įvykių ir renginių sklaidos mobilioji programėlė. 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: Įvykių ir vietų sklaida, geolokacija, socialiniai tinklai.

Kaunas, 2021. XX p.

Santrauka

Šiame darbe pristatoma įvykių ir renginių sklaidos mobilioji programėlė. Didžioji dalis renginių bei lankytinių vietų informacijos internete yra išmėtyta po kelis šaltinius bei pateikiama neišsamiai. Kuriama programėlė yra aktuali dėl šios informacijos sutraukimo į vieną vietą, patogaus jos pasiekiamumo bei papildomo funkcionalumo, leidžiančio tiksliau matyti sklaidą. Pagrindinis darbo tikslas – supaprastinti įvykių bei lankytinų vietų informacijos prieinamumą ir sukurti socialinį tinklą skirtą šiai sferai.

Konkurentų analizės metu buvo nagrinėjami kitų įvykių sklaidos programėlių pranašumai ir trūkumai, lyginamas funkcionalumas, turimos informacijos išsamumas bei patogumas naudoti. Nustatyta, kad šiuo metu mobiliųjų programėlių rinkoje nėra kai kurių esminių šioje programėlėje taikomų sprendimų. Reikalavimų analizės metu buvo apibrėžti funkciniai ir nefunkciniai reikalavimai, tokie kaip naudojimosi paprastumas, reikalingo funkcionalumo apimtis.

Sistema buvo realizuota naudojant Dart ir Python programavimo kalbas, Beautiful Soup bei Google\_maps\_flutter bibliotekas ir Flutter karkasą. Sistemos kodas padengtas komponentų testais, kurie užtikrina svarbiausių komponentų korektišką veikimą. Parengtas vartotojo vadovas, kuris supažindina su sistemos diegimu, veikimu bei palaikymu.

Kalverš, Paulius Tomas. Event promotion mobile app. 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: Dissemination of events and places, geolocation, social networks.

Kaunas, 2019. Number of pages.

Summary

This paper presents a mobile app for disseminating events and happenings. Most of the information on events and places of interest on the Internet is scattered from several sources and is incomplete. The developed app is relevant due to the compilation of this information in one place, its convenient accessibility and additional functionality that allows to see the distribution more accurately. The main goal of the work is to simplify the availability of information about events and places of interest and to create a social network dedicated to this area.

During the analysis of competitors, the advantages and disadvantages of other events dissemination gadgets were examined, functionality, completeness of available information and ease of use were compared. It has been found that there are currently some key solutions in the mobile app market. During the requirements analysis, functional and non-functional requirements were defined, such as ease of use, scope of required functionality.

The system was implemented using Dart and Python programming languages, Beautiful Soup and Google\_maps\_flutter libraries and Flutter framework. The system code is covered by component tests that ensure the correct operation of the most important components. A user manual has been prepared, which introduces the system installation, operation and support.

Turinys

[Lentelių sąrašas 8](#_Toc3806737)

[Paveikslų sąrašas 9](#_Toc3806738)

[Santrumpų ir terminų sąrašas 10](#_Toc3806739)

[Įvadas 11](#_Toc3806740)

[1. Analizė 12](#_Toc3806741)

[1.1. Techninis pasiūlymas 12](#_Toc3806742)

[1.1.1. Sistemos apibrėžimas 12](#_Toc3806743)

[1.1.2. Bendras veiklos tikslas 12](#_Toc3806744)

[1.1.3. Sistemos pagrįstumas 12](#_Toc3806745)

[1.1.4. Konkurencija rinkoje 12](#_Toc3806746)

[1.1.5. Prototipai ir pagalbinė informacija 13](#_Toc3806747)

[1.1.6. Ištekliai, reikalingi sistemai sukurti 13](#_Toc3806748)

[1.2. Galimybių analizė 13](#_Toc3806749)

[1.2.1. Techninės galimybės 13](#_Toc3806750)

[1.2.2. Vartotojų pasiruošimo analizė 14](#_Toc3806751)

[2. Projektas 15](#_Toc3806752)

[2.1. Reikalavimų specifikacija 15](#_Toc3806753)

[2.1.1. Komercinė specifikacija 15](#_Toc3806754)

[2.1.2. Sistemos funkcijos 15](#_Toc3806755)

[2.1.3. Vartotojo sąsajos specifikacija 16](#_Toc3806756)

[2.1.4. Realizacijai keliami reikalavimai 16](#_Toc3806757)

[2.1.5. Techninė specifikacija 16](#_Toc3806758)

[2.2. Projektavimo metodai 16](#_Toc3806759)

[2.2.1. Projektavimo valdymas ir eiga 16](#_Toc3806760)

[2.2.2. Projektavimo technologija 16](#_Toc3806761)

[2.2.3. Programavimo kalbos, derinimo, automatizavimo priemonės, operacinė sistemos 17](#_Toc3806762)

[2.3. Sistemos projektas 17](#_Toc3806763)

[2.3.1. Statinis sistemos vaizdas 17](#_Toc3806764)

[2.3.2. Dinaminis sistemos vaizdas 17](#_Toc3806765)

[3. Testavimas 18](#_Toc3806766)

[3.1. Testavimo planas 18](#_Toc3806767)

[3.2. Testavimo kriterijai 18](#_Toc3806768)

[3.3. Komponentų testavimas 18](#_Toc3806769)

[3.4. Integracinis testavimas 18](#_Toc3806770)

[3.5. Vartotojo sąsajos testavimas 18](#_Toc3806771)

[4. Dokumentacija naudotojui 19](#_Toc3806772)

[4.1. Apibendrintas sistemos galimybių aprašymas 19](#_Toc3806773)

[4.2. Vartotojo vadovas 19](#_Toc3806774)

[4.3. Diegimo vadovas 19](#_Toc3806775)

[4.4. Administravimo vadovas 19](#_Toc3806776)

[Rezultatai ir išvados 20](#_Toc3806777)

[Literatūros sąrašas 21](#_Toc3806778)

[Priedai 22](#_Toc3806779)

[1 priedas. Priedo pavadinimas 22](#_Toc3806780)

Lentelių sąrašas

[**1 lentelė.** Konkurentų apžvalga 16](#_Toc70907939)

[**2 lentelė.** Pagrindiniai baigiamojo projekto stiliai ir jų aprašymai 65](#_Toc70907940)

Paveikslų sąrašas

[**pav. 1.1** „Miesto žirafos“ pagrindinio lango ir žemėlapio vaizdai 12](file:///C:\Users\Programuotojas-TC\Desktop\Project\Doc\BAKALAURAS.docx#_Toc70909101)

[**pav. 1.2** „Eventbrite“ pagrindinio lango ir paieškos laukų vaizdai 13](file:///C:\Users\Programuotojas-TC\Desktop\Project\Doc\BAKALAURAS.docx#_Toc70909102)

[**pav. 1.3** „VilniusGo“ pagrindinio lango ir mėgstamų vietų vaizdai 14](file:///C:\Users\Programuotojas-TC\Desktop\Project\Doc\BAKALAURAS.docx#_Toc70909103)

[**pav. 1.4** „Places Near Me“ pagrindinio lango ir žemėlapio vaizdai 15](file:///C:\Users\Programuotojas-TC\Desktop\Project\Doc\BAKALAURAS.docx#_Toc70909104)

[**pav. 2.1** „Let’s go“ Sistemos panaudojimo atvejų diagrama 19](#_Toc70909105)

[**pav. 2.2** „Peržiūrėti renginių sąrašą“ panaudojimo atvejo veiklos diagrama 20](#_Toc70909106)

[**pav. 2.3** „Peržiūrėti detalesnę renginio/vietos informaciją“ panaudojimo atvejo veiklos diagrama 21](#_Toc70909107)

[**pav. 2.4** „Naudotis interaktyviu žemėlapiu“ panaudojimo atvejo veiklos diagrama 22](#_Toc70909108)

[**pav. 2.5** „Peržiūrėti vietų sąrašą“ panaudojimo atvejo veiklos diagrama 23](#_Toc70909109)

[**pav. 2.6** „Prisijungti“ panaudojimo atvejo veiklos diagrama 24](#_Toc70909110)

[**pav. 2.7** „Rašyti komentarą“ panaudojimo atvejo veiklos diagrama 25](#_Toc70909111)

[**pav. 2.8** „Pridėti vietą/renginį prie mėgstamų sąrašo“ panaudojimo atvejo veiklos diagrama 26](#_Toc70909112)

[**pav. 2.9** „Sukurti privatų renginį“ panaudojimo atvejo veiklos diagrama 27](#_Toc70909113)

[**pav. 2.10** „Sukurti privačią vietą“ panaudojimo atvejo veiklos diagrama 28](#_Toc70909114)

[**pav. 2.11** „Keisti renginio/vietos informaciją“ panaudojimo atvejo veiklos diagrama 29](#_Toc70909115)

[**pav. 2.12** „Ištrinti renginį/vietą“ panaudojimo atvejo veiklos diagrama 30](#_Toc70909116)

[**pav. 2.13** „Siųsti paskyros verifikavimo užklausą“ panaudojimo atvejo veiklos diagrama 31](#_Toc70909117)

[**pav. 2.14** „Sukurti viešą renginį“ panaudojimo atvejo veiklos diagrama 32](#_Toc70909118)

[**pav. 2.15** „Sukurti viešą vietą“ panaudojimo atvejo veiklos diagrama 33](#_Toc70909119)

[**pav. 2.16** „Verifikuoti vartotoją“ panaudojimo atvejo veiklos diagrama 34](#_Toc70909120)

[**pav. 2.17** sistemos diegimo diagrama 36](#_Toc70909121)

[**pav. 2.18** komponentų diagrama 37](#_Toc70909122)

[**pav. 2.19** duomenų bazės schemos struktūra 38](#_Toc70909123)

[**pav. 2.20** sistemos paketų diagrama 39](#_Toc70909124)

[**pav. 2.21** kliento klasių diagrama 40](#_Toc70909125)

[**pav. 2.22** serverio klasių diagrama 41](#_Toc70909126)

[**pav. 2.23** MVVM architektūrinis modelis 42](#_Toc70909127)

[**pav. 2.24** „Peržiūrėti renginių sąrašą“ sekų diagrama 42](#_Toc70909128)

[**pav. 2.25** „Naudotis interaktyviu žemėlapiu“ sekų diagrama 43](#_Toc70909129)

[**pav. 2.26** „Prisijungti“ sekų diagrama 44](#_Toc70909130)

[**pav. 2.27** „Rašyti komentarą“ sekų diagrama 45](#_Toc70909131)

[**pav. 2.28** „Pridėti vietą/renginį prie mėgstamų sąrašo“ sekų diagrama 46](#_Toc70909132)

[**pav. 2.29** „Sukurti privatų renginį“ sekų diagrama 47](#_Toc70909133)

[**pav. 2.30** „Siųsti paskyros verifikavimo užklausą“ sekų diagrama 47](#_Toc70909134)

[**pav. 2.31** „Sukurti viešą renginį“ sekų diagrama 48](#_Toc70909135)

Santrumpų ir terminų sąrašas

**Santrumpos:**

Prof. – profesorius

**Terminai:**

Dart – mobiliųjų programėlių programavimo kalba

Flutter – Dart programavimo kalbos karkasas

Web scraping – automatinis tinklalapių naršymas ieškant reikiamos informacijos

Web scraper – objektas/servisas atliekantis automatinį tinklapių naršymą

Beautiful Soup – tinklapių naršymo karkasas python programavimo kalbai

Hot reload – Flutter karkaso ypatybė kaskart po pakeitimų greitai perkrauti programėlės vaizdą

Widget stiliaus programavimas – komponentų medžio stiliaus programavimas, kai viskas paremta komponentais ir jų išdėstymu

Widget – Flutter karkaso valdiklis, kuris atsakingas už ekrano vaizdą

MVVM (model-view-viewModel) – sistemos architektūros tipas, kai vietoj įprasto MVC valdiklio yra naudojamas viewModel valdiklis, kuris yra tarpininkas tarp vaizdo komponentų ir modelių

Testavimas

Perziuret srifta ir dydi

Sutvarkyt, kad graziai atrodytu

Paveikslelius is naujo sudeliot

Daugiau lentelių??

Literatūros sąrašas ir active linkai

Psl skaičius

Pratestuot ikelima i sistema

Įvadas

Didėjant išmaniųjų įrenginių naudojimui ir didesnio pasiekiamumo poreikiui vis daugiau sistemų yra pritaikomos arba perkeliamos į šią terpę. Taip pat auga tendencija visą informaciją turėti vienoje vietoje. Pasirinkta įvykių ir lankytinų vietų sklaidos niša, nes šiuo metu esantys sprendimai yra arba nepakankamai išvystyti funkcionalumo prasme, turi per mažai duomenų. Taip pat didžioji dalis sistemų yra arba nebetobulinamos ir nepalaikomos arba yra užsienio rinkos, dėl ko nukenčia informacijos kiekis ir kokybė. Kuriamos programėlės „Let’s go“ vartotojai galės matyti įvairialypę informaciją apie renginius bei lankytinas vietas, populiarumo statististikas bei kitų vartotojų atsiliepimus. Taip pat pridėta aktyvi geolokacija, kad būtų iliustruotai matomos renginių bei lankytinų vietų pozicijos žemėlapyje.

Darbo tikslas – padaryti patogesnius, išsamesnius, interaktyvesnius renginių bei vietų atvaizdavimą ir analizę. Šiam tikslui pasiekti išsikelti šie uždaviniai:

1. Išanalizuoti rinkoje esančias panašias programėles
2. Sudaryti sistemos reikalavimų analizę
3. Remiantis turimais duomenimis sudaryti sistemos projektą
4. Realizuoti sistemą
5. Sudaryti sistemos dokumentaciją

Darbo rezultatai – pilnai realizuota mobilioji programėlė, suteikianti vartotojams galimybę interaktyviai sekti renginių bei lankytinų vietų sklaidą, su galimybe pridėti savus renginius ir vietas.

Darbas susideda iš keturių pagrindinių skyrių – analizės, projekto, testavimo ir dokumentacijos. Analizės skyriuje yra aptariami bendri programėlės kūrimo tikslai bei pagrįstumas, apžvelgiami techniniai sprendimai gauti naujausiai įvykių bei renginių informacijai, taip pat ir naujai užregistruotų lankytinų vietų informacijai. Projekto dalyje aprašomi programėlės veikimo funkciniai bei nefunkciniai reikalavimai, komercinė specifikacija, renginių bei lankytinų vietų sklaidos veikimo principai ir pagrindinės funkcijos. Testavimo dalyje pateikiamas testavimo planas, strategija, eiga bei rezultatai. Dokumentacijos skyriuje yra apžvelgiama sukurta sistema bei pateikiami vartotojo, sistemos diegimo bei administravimo vadovai. Darbo pabaigoje yra pateikiamos darbo išvados bei rezultatai.

Sistemos apimtis yra apie 7000 kodo eilučių, iš kurių:

* Apie 1500 eilučių yra serverio dalyje (Python kalba)
* Apie 5500 eilučių yra kliento dalyje (Dart kalba, Flutter karkasas)

# Analizė

## Techninis pasiūlymas

### Sistemos apibrėžimas

„Let’s go“ programėlė – tai sistema, kuri leis žmonėms patogiai pasiekti, kurti bei vizualiai matyti aplinkui juos vykstančius renginius ir lankytinas vietas. Interaktyvus žemėlapis leis lengviau susidaryti įspūdį, kokios veiklos yra aplink vartotoją. Taip pat rodoma populiarumo statistika bei komentarų skiltis parodys kitų žmonių susidomėjimą ir leis pasidalinti įspūdžiais.

### Bendras veiklos tikslas

Bendras veiklos tikslas – renginių bei lankytinų vietų informacijos apjungimas į vieną platformą, kuri skatins žmones ieškoti naujų veiklų bei bendraminčių. Toliau vystant sistemą bei pridedant naujo funkcionalumo galima įsitvirtinti rinkoje bei siekti iš to komercinės naudos ne tik sau (užsidirbti), bet ir renginių organizatoriams, kadangi bus populiarinami jų renginiai. Nekomercinė nauda bus ta, jog žmonės turės konkrečią platformą skirtą veikloms aplink juos, kas skatins veiklumą.

### Sistemos pagrįstumas

Šiuo metu nėra sistemos, kuri Lietuvoje vartotojams leistų interaktyviai ir patogiai matyti visų aplink juos vykstančių renginių bei lankytinų vietų informaciją, suteiktų galimybę naudotis renginių bei lankytinų vietų žemėlapiu. Didžioji sistemų dalis turi po vieną išskirtinį funkcionalumą, tačiau „Let’s go“ programėlės tikslas yra sujungti juos visus į vieną bei pridėti papildomų. Taip bus sudarytos sąlygos turėti vieną programėlę, kurioje bus visa reikiama informacija vartotojų veikloms ir viskas bus vienoje vietoje.

Dabartinė situacija rinkoje atspindi tai, jog egzistuoja dviejų pobūdžių programėlės – skirtos lankytinoms vietoms/vietovėms pažymėti, bei tos, kurios rodo netoliese esančių renginių informaciją. „Let’s go“ išsprendžia problemą šiuos du funkcionalumus sujungdama į vieną programėlę.

Kuriama programėlė taip pat turės papildomus funkcionalumus, kurių trūksta kitose programėlėse – savų renginių ir vietų nemokamas pridėjimas, galimybė rinkti „likes“ bei susilaukti kitų naudotojų komentarų.

Dėl pridėto realaus laiko žemėlapio bus galimybė matyti tiek vykstančių renginių tiek lankytinų vietų pozicijas žemėlapyje, kas ženkliai palengvina programėlės naudojimą.

### Konkurencija rinkoje

Šiuo metu egzistuoja panašių sprendimų į „Let’s go“ programėlę tiek užsienio tiek Lietuvos rinkoje, kurie gali būti naudojami renginių arba vietų paieškai ir leidžia vartotojams interaktyviai atsirinkti norimą informaciją.

Populiariausia lietuviška renginių programėlė yra „Miesto žirafos“ [1]. Naudojantis programėle galima matyti pagrindiniuose Lietuvos miestuose (Vilnius, Kaunas, Klaipėda) vykstančius arba vyksiančius renginius, atlikti renginių paiešką, filtravimą pagal kategoriją, uždėti „pamėgti“ ant patikusio renginio. Visi pamėgti renginiai yra vaizduojami atskirame lange. Taip pat galima paspausti „Pridėti savo renginį“ ir taip būti nukreiptiem į „Miesto žirafos“ tinklalapį. Dar vienas funkcionalumas yra žemėlapis, kuriame vaizduojamos lankytinos vietos.





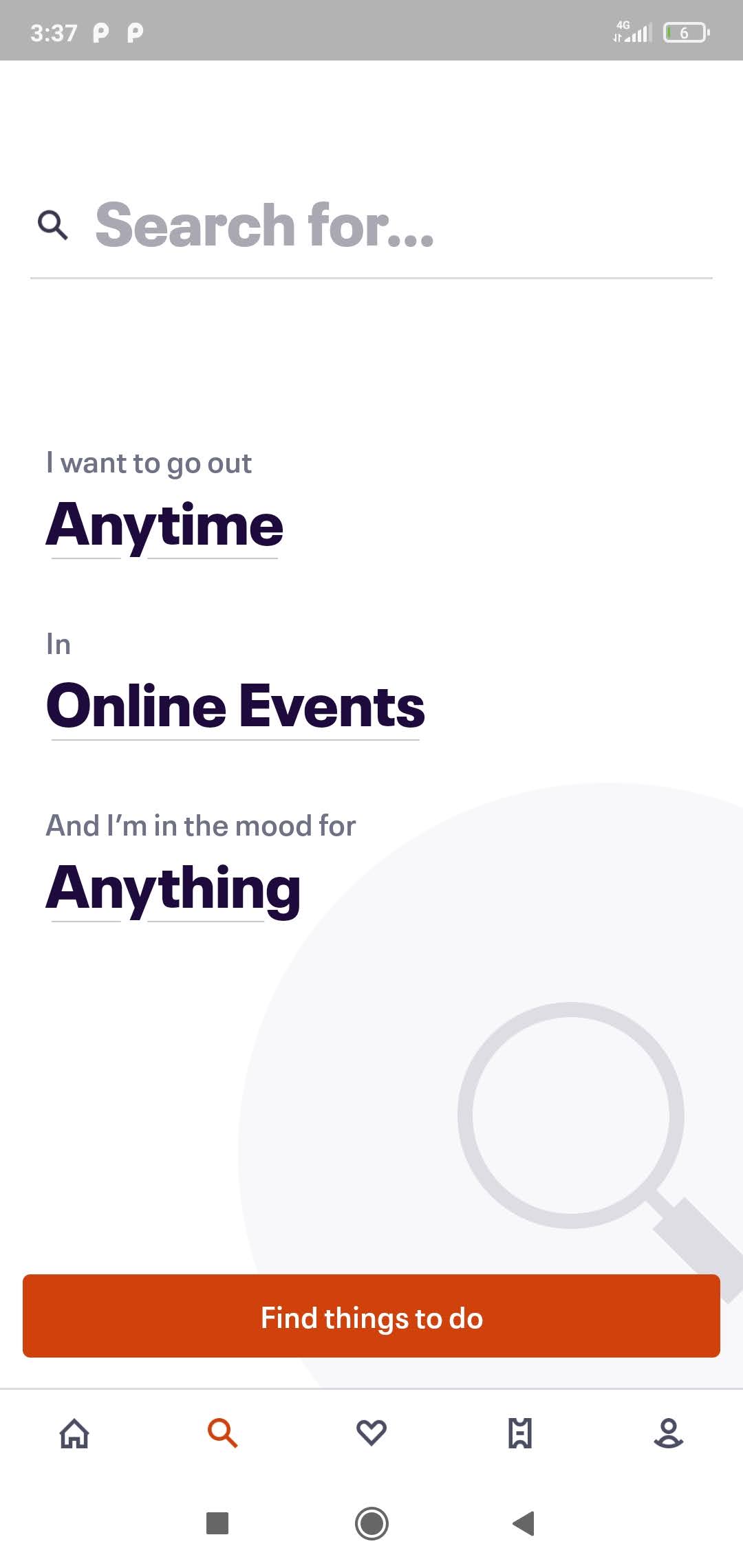
**pav. 1.1** „Miesto žirafos“ pagrindinio lango ir žemėlapio vaizdai

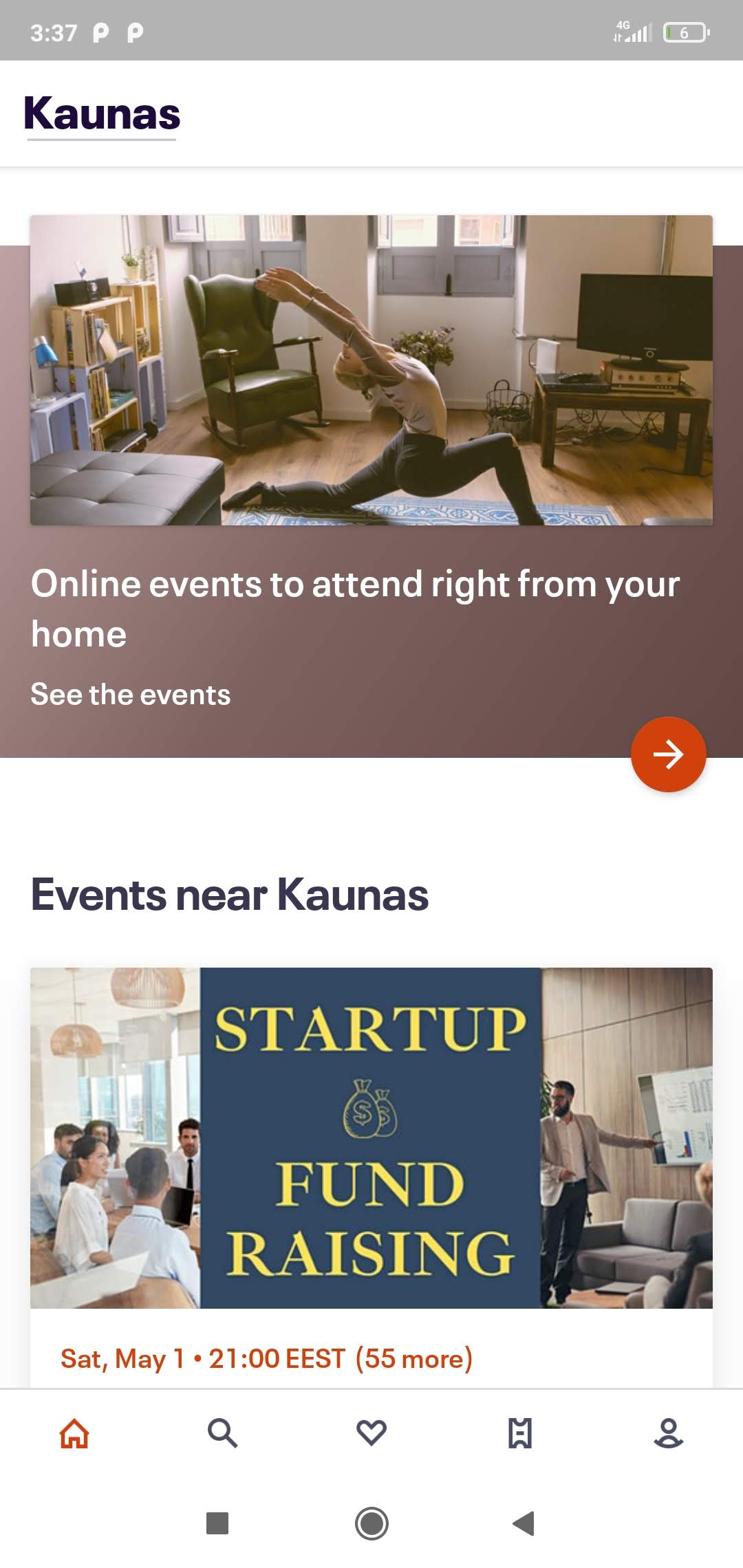
*Papildomas šaltinis: https://www.mzirafos.lt/*

Eventbrite [2] yra programėlė rodanti renginius aplink nurodytą vietą. Renginius galima pridėti prie „mėgiamų“ sąrašo. Programėlė turi galimybę atlikti išplėstinę paiešką nurodžius tris raktažodžius ir išskirtinę funkciją nusipirkti bilietą į renginį.

**pav. 1.2** „Eventbrite“ pagrindinio lango ir paieškos laukų vaizdai

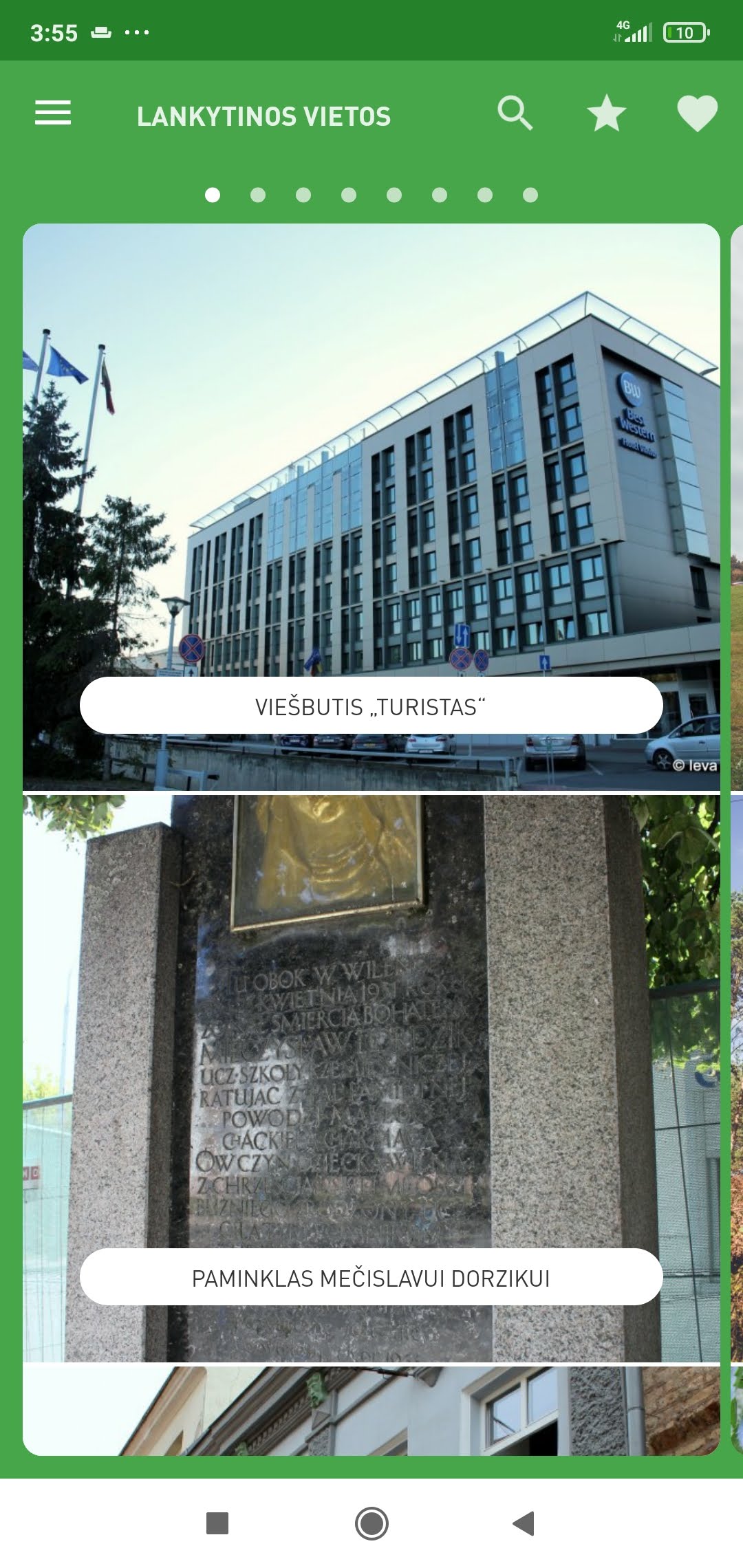
*Papildomas šaltinis: https://www.eventbrite.com/*

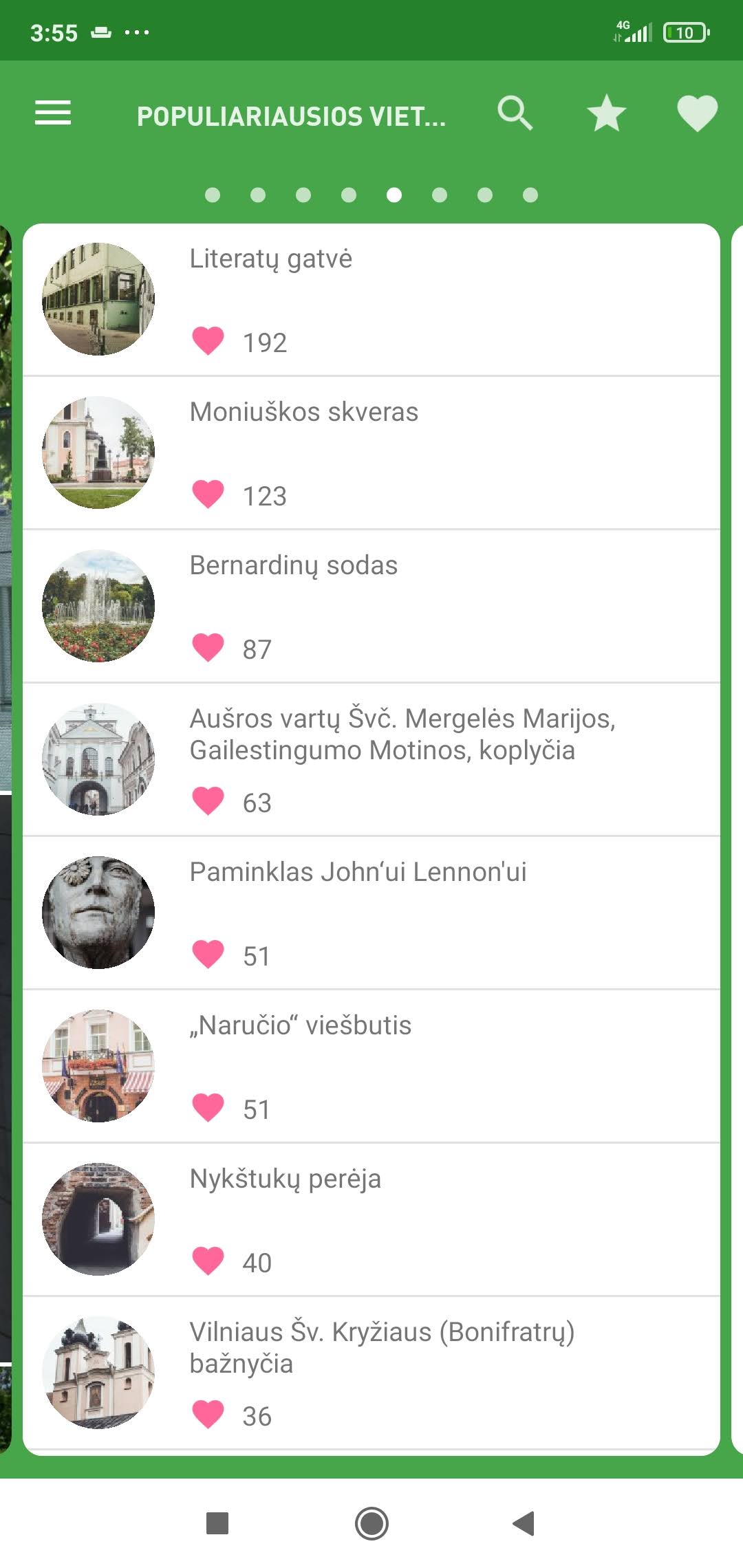




Programėlė pavadinimu VilniusGo [3] yra skirta lankytinų vietų bei objektų atvaizdavimui ir informacijos apie juos pateikimui. Ši sistema turi kelis esminius funkcionalumus – mėgstamų vietų sąrašą, populiarių vietų sąrašą bei pažintinius maršrutus Vilniuje. Taip pat visi objektai yra sugrupuoti į: lankytinas vietas, gamtinius objektus, istorinius objektus bei kultūrinius objektus.

**pav. 1.3** „VilniusGo“ pagrindinio lango ir mėgstamų vietų vaizdai

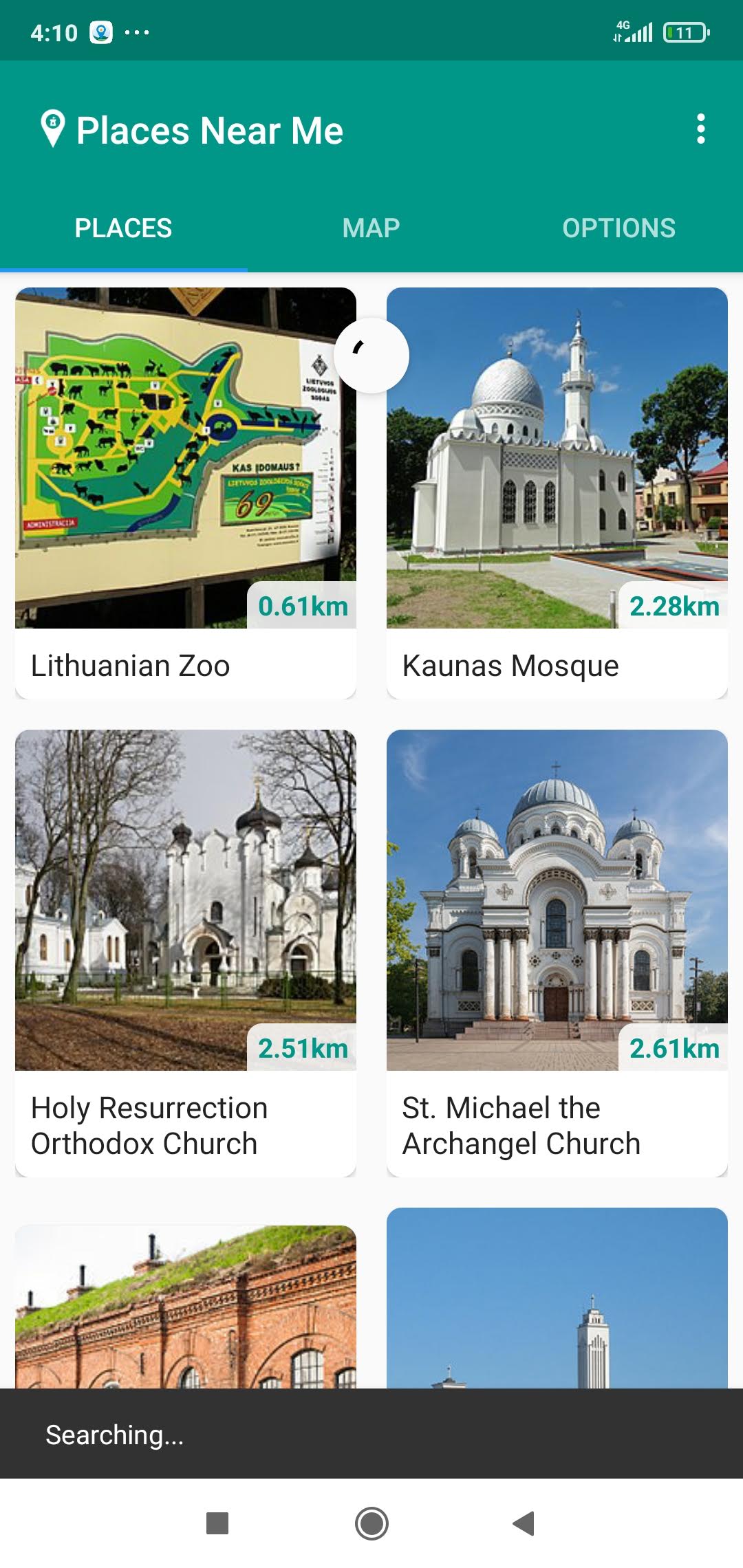




*Papildomas šaltinis: http://www.vilniusgo.lt/*

Dar viena programėlė rinkoje yra „Places Near Me“ [4]. Tai sistema, kuri pagal nurodytą atstumą nuo vartotojo lokacijos ieško lankytinų vietų bei objektų, tada pateikia sąrašą, kuriame nurodyta objekto nuotrauka, pavadinimas bei atstumas nuo vartotojo. Taip pat yra žemėlapio funkcija, kuri sužymi visus rastus objektus žymekliais.

**pav. 1.4** „Places Near Me“ pagrindinio lango ir žemėlapio vaizdai





**1 lentelė.** Konkurentų apžvalga

| **Lyginimo kriterijai** | Miesto žirafos | Eventbrite | VilniusGo | | Places near me | **Let’s go** |
| --- | --- | --- | --- | --- | --- | --- |
| Žemėlapis renginiams | - | - | - | - | | + |
| Žemėlapis lankytinoms vietoms | + | - | - | + | | + |
| Renginiai ir lankytinos vietos | + | - | - | - | | + |
| Galimybė pridėti savo vietą/renginį | + | - | - | - | | + |
| Vieši/privatūs renginiai/vietos | Vieši | Vieši | Vieši | Vieši | | Vieši ir privatūs |
| Mėgstamų sąrašas | + | + | + | - | | + |
| Pirkti bilietą | - | + | - | - | | -- |
| Komentarai | - | - | - | - | | + |

Iš lentelės duomenų matyti, kad artimiausias konkurentas kuriamai sistemai yra „Miesto žirafos“ programėlė. Kitų sistemų trūkumai pagrinde yra duomenų bei funkcionalumo stoka. Taip pat verta paminėti, kad nei vienas nagrinėtas konkurentas neturėjo komentarų skilties funkcionalumo. „Let’s go“ programėlė išsiskiria iš kitų savo lankstumu ir apimamu duomenų kiekiu. Dar labai svarbūs paminėti faktai: „Miesto žirafos“ programėlė paskutinį kartą atnaujinta prieš tris metus, kas reiškia, jog jinai nebepalaikoma. Kitos, nors ir palaikomos, nėra vietinės kilmės (išskyrus VilniusGo), tad nusileidžia turimu aplinkinių vietų bei renginių informacijos kiekiu. Darant projektą pagrinde buvo žiūrima į du konkurentus – „Miesto žirafos“ ir „Eventbride“ bei iš jų semiamasi idėjų: mėgstamų vietų sąrašas antrame bei žemėlapio kompozicija pirmame labai padėjo „Let’s go“ programėlės vystymui.

Šie vertinimo kriterijai buvo pasirinkti, nes mano ir kitų apklaustų žmonių nuomone jie yra svarbiausi ir suteikiantys daugiausio pranašumo rinkoje prieš panašaus pobūdžio programėles. Pavyzdžiui iš programėlių sąrašo labai išsiskiria „Eventbrite“, turintis integruotą bilietų pirkimo posistemę, kas prideda nemažai patogumo bei išsikirtinumo iš kitų jo konkurentų. Šią funkciją ateity planuoju pridėti taip pat, kadangi šiuo metu tam pritrūko laiko.

### Prototipai ir pagalbinė informacija

Kuriant „Let’s go“ programėlę daugiausia kaip pavyzdžiais buvo naudojamasi konkurentų sukurtais produktais. Daugiausia padėjo „Miesto žirafos“ ir „Eventbride“ programėlės iš kurių buvo semtasi renginių bei lankomų vietų atvaizdavimo. Taip pat buvo analizuojama skiltis „Facebook events“, kurioje patogiai atvaizduojami renginių sąrašai.

### Ištekliai, reikalingi sistemai sukurti

„Let’s go“ sistemos kodo bazei saugoti reikalinga viena kodo talpykla Github serveryje. Šiam darbui skyriau apie 150 valandų, į kurias nebuvo įtraukti naujos kalbos bei karkaso mokymasis. Reikėjo mokytis dart kalbą bei jos karkasą Flutter. Sistemos modeliavimui ir architektūros planavimui buvo naudojamas MagicDraw, serverio kodo rašymui – Atom kodo redaktorius, kliento – Android Studio.

## Galimybių analizė

### Techninės galimybės

Sistemos serveris buvo realizuotas naudojant Python programavimo kalbą. Kadangi vienas iš serveryje esančių funkcionalumų yra „web scraping“, šalia Python kalbos buvo naudojama Beutiful Soup biblioteka. Jos dėka serveris turi galimybę nuskaityti bei išsaugoti duomenis iš pasirinktų tinklalapių bei įvarių šaltinių. Be šios, dar viena priežastis kodėl buvo pasirinkta ši technologija yra ta, jog kadangi Python yra aukšto lygmens programavimo kalba, su ja labai lengva ir greita rašyti kodą serverio užklausų logikai. Taip pat palaikomos gijos, kurių reikia vienu metu paleisti daugiau nei vieną „web scraper“, kad būtų būtų surenkama daugiau informacijos ir tai būtų daroma greičiau.

Programėlės klientas parašytas Dart kalba naudojantis Flutter karkasu. Šios technologijos buvo pasirinktos dėl savo paprastumo, bet tuo pačiu ir turimo didelio našumo bei papildomų funkcijų spartesniam darbui (pvz.: Hot reload funkcija). Patogiai naudojamas „Widget“ stiliaus programavimas, kai visi matomi komponentai yra komponentų medyje padėjo nesunkiai išdėstyti bei vizualizuoti objektus ekrane. Taip pat verta paminėti, jog Flutter gali kompiliuotis tiek į Android tiek į iOS mašininį kodą, kas ženkliai sumažina kaštus norint perkelti programėlę į kitą platformą.

Serverio pusėje vyksta komunikacija su PostgreSQL serveriu. Šis pasirinktas vietoj tokių serverių kaip MySQL dėl tokių pranašumų kaip geresnis gijų konkuravimo palaikymas bei duomenų suverenitetas, kas reiškia, jog yra didesnė galimybė, kad kreipiantis į serverį kelioms gijoms vienu metu, duomenys nebus sugadinti.

Kliento pusėje vyksta komunikacija su LiteSQL servisu. Papildoma duomenų bazė kliento pusėje pasirinkta dėl greitesnio veikimo ir serverio resursų taupymo.

Taikant pasirinktas technologijas yra suteiktos pilnos galimybės pilnai realizuoti pasirinktą projektą tiek serverio, tiek kliento pusėje.

### Vartotojų pasiruošimo analizė

Įprastam vartotojui nereikės turėti jokių specialių žinių norint naudotis „Let‘s go“ programėle, kadangi grafinės sąsajos realizacija yra gan paprasta. Sąsaja buvo ištestuota su keliais vartotojais be jokio pradinio apmokymo ir vidutinis sąsajos perpratimo laikas buvo kelios minutės.

# Projektas

## Reikalavimų specifikacija

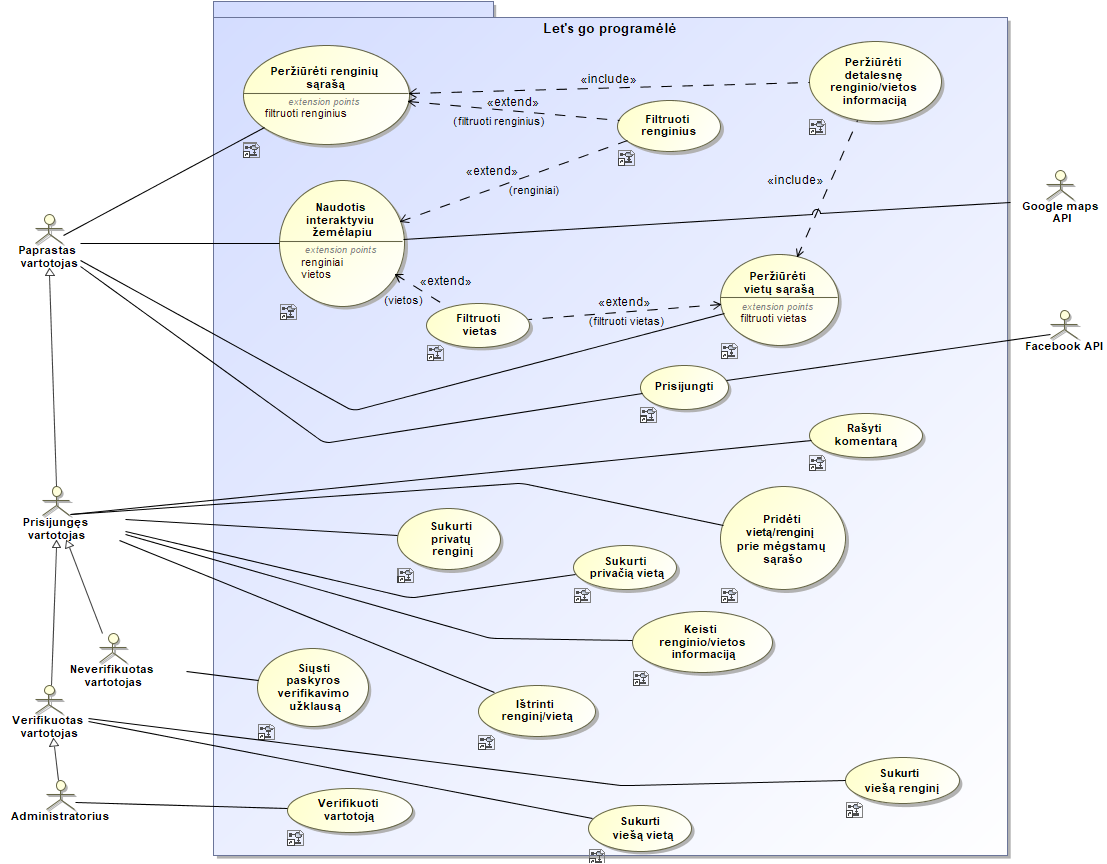
### Komercinė specifikacija

Tai yra universitetinis projektas, kuriamas bakalauro baigiamajam darbui. Projektinio darbo bei tiriamosios dalies sumanytojas bei vykdytojas – universiteto studentas Paulius Tomas Kalverš. Projektinio darbo naudotojai: žmonės organizuojantys bei norintys dalyvauti įvairiuose renginiuose, barų, klubų, kitų laisvalaikio erdvių savininkai ir tų vietų lankytojai.

Kadangi tai yra asmeninis projektas, projekto biudžeto nėra. Numatyta projekto baigimo data – 2021 m. gegužės 5d. Iki šios datos projektas ir darbo tyrimas turi būti baigti.

### Sistemos funkcijos

„Let’s go“ sistemos funkciniai reikalavimai yra pateikiami UML panaudojimo diagramoje **pav. 2.1**.



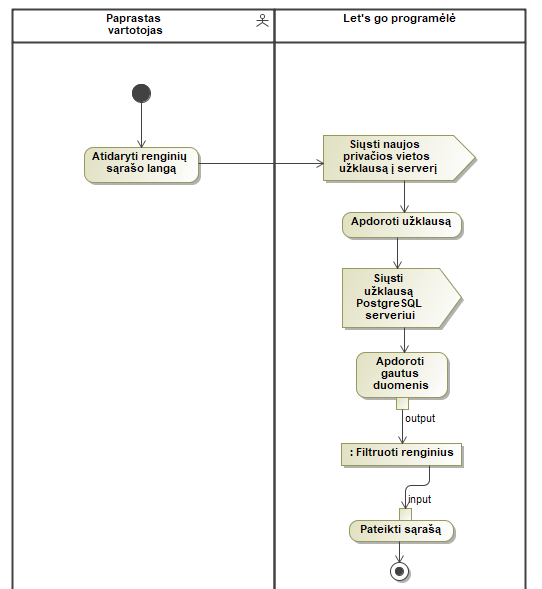
**pav. 2.1** „Let’s go“ Sistemos panaudojimo atvejų diagrama

Sistema turi šiuos aktorius:

* Paprastas vartotojas – vartotojas, kuris yra neprisijungęs prie sistemos
* Prisijungęs vartotojas – vartotojas, kuris prisijungęs pries sistemos su savo „Facebook“ paskyra
* Neverifikuotas vartotojas – prisijungęs vartotojas, kuriam nėra suteikta paskyros verifikacija
* Verifikuotas vartotojas – prisijungęs vartotojas, kuriam suteikta paskyros verifikacija
* Administratorius – už neverifikuotų vartotojų verifikavimą atsakingas vartotojas
* Facebook API – trečios šalies sitema, atsakinga už prisijungimo sesijos žetonų gerenavimą
* Google maps API – trečios šalies sistema, atsakinga už žemėlapio atvaizdavimą bei lokacijų paieškos logiką

Posistemių panaudojimo atvejai yra pateikiami UML veiklos diagramomis, kurios nurodo supaprastintus panaudojimo atvejų vykdymo scenarijus, nurodant, kokie veikmai sistemoje yra atliekami aktoriaus ir kaip sistema reaguoja į jo atliktus veiksmus. **pav. 2.2- pav. 2.16** paveikslėliuose yra pateikiamos programėlės panaudojimo atvejų veiklos diagramos.

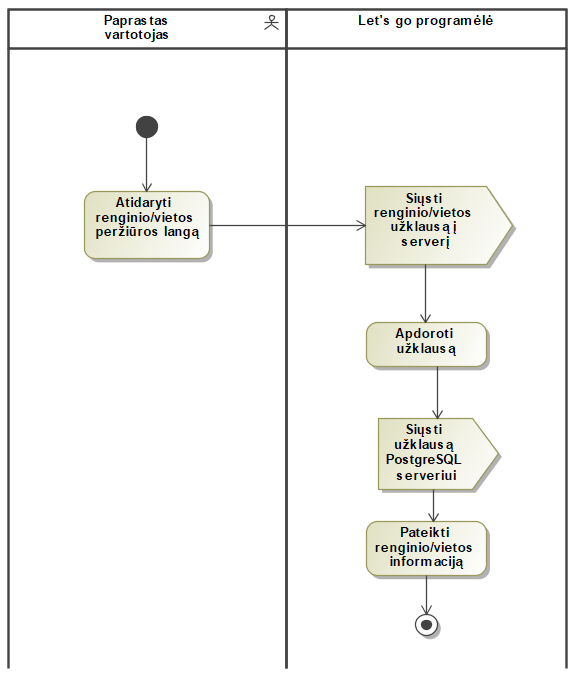
**pav. 2.2** yra pavaizduota „Peržiūrėti renginių sąrašą“ panaudojimo atvejo veiklos diagrama.



**pav. 2.2** „Peržiūrėti renginių sąrašą“ panaudojimo atvejo veiklos diagrama

Norint peržiūrėti renginių sąrašą pirma reikia paspausti ant sąrašo ikonos ekrano apačioje. Tą padarius sugeneruojamas bei parodomas šios dienos renginių sąrašas.

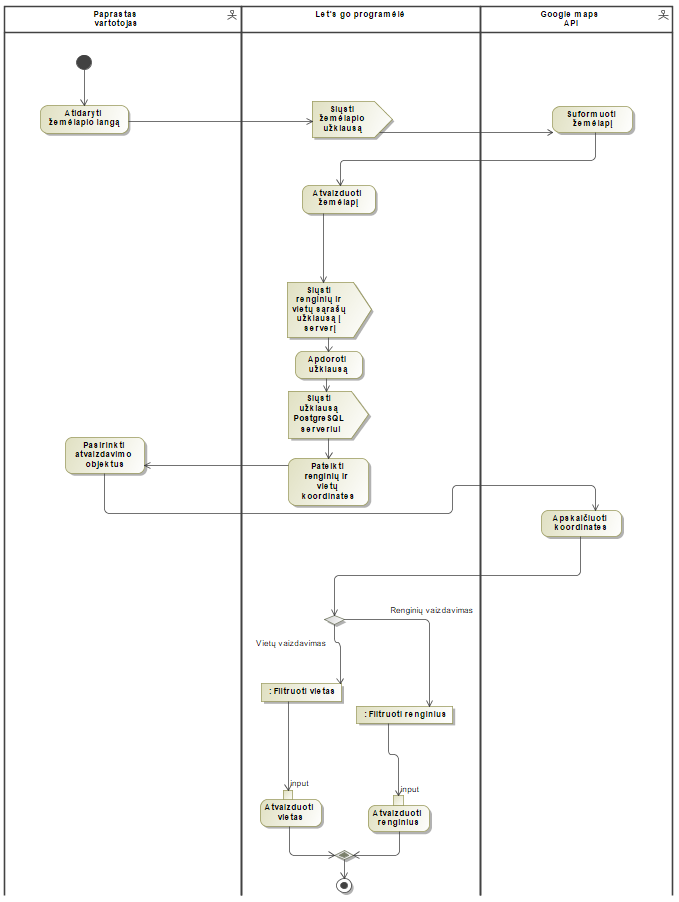
**pav. 2.3** yra pateikiama „Peržiūrėti detalesnę rengino/vietos informaciją“ veiklos diagrama.



**pav. 2.3** „Peržiūrėti detalesnę renginio/vietos informaciją“ panaudojimo atvejo veiklos diagrama

Paspaudus ant konkretaus renginio arba vietos siunčiama užklausa į serverį, kuris grąžina įvykio/vietos informaciją.

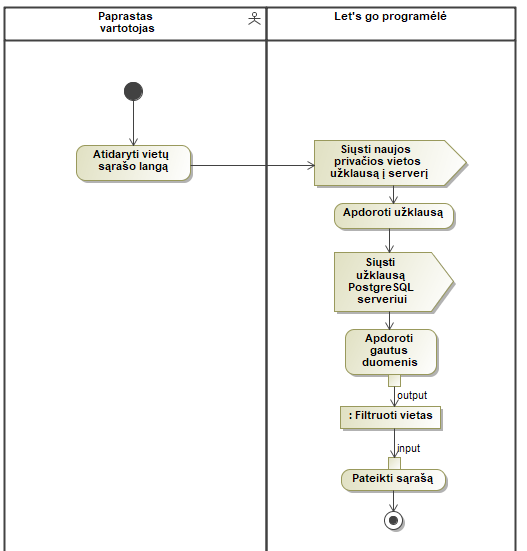
**pav. 2.4** yra pateikiama naudojimosi interaktyviu žemėlapiu veiklos diagrama.



**pav. 2.4** „Naudotis interaktyviu žemėlapiu“ panaudojimo atvejo veiklos diagrama

Atidarius žemėlapio langą siunčiama užklausa į serverį, kuris grąžina vietų ir renginių sąrašus. Tada priklausomai nuo to, ką vartotojas pasirinks atvaizduot žemėlapyje, įvyks vietų arba renginių filtravimas ir jie bus atvaizduoti.

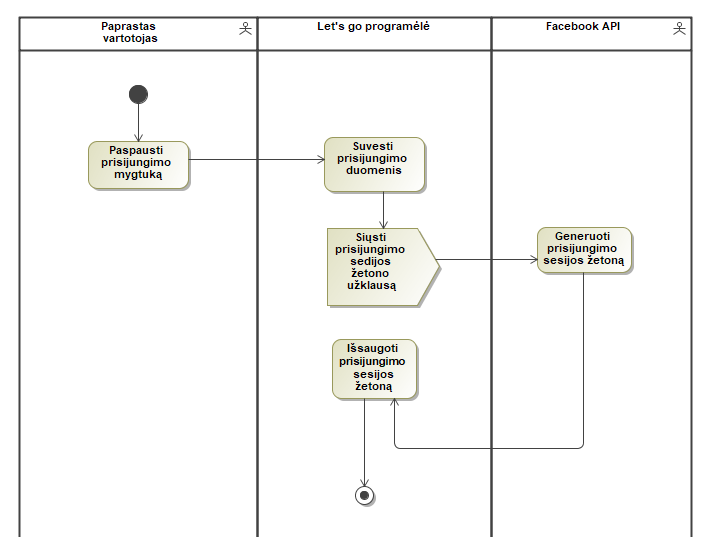
**pav. 2.5** yra pavaizduota „Peržiūrėti vietų sąrašą“ panaudojimo atvejo veiklos diagrama.



**pav. 2.5** „Peržiūrėti vietų sąrašą“ panaudojimo atvejo veiklos diagrama

Norint peržiūrėti vietų sąrašą pirma reikia paspausti ant sąrašo ikonos ekrano apačioje. Tą padarius sugeneruojamas bei parodomas vietų sąrašas.

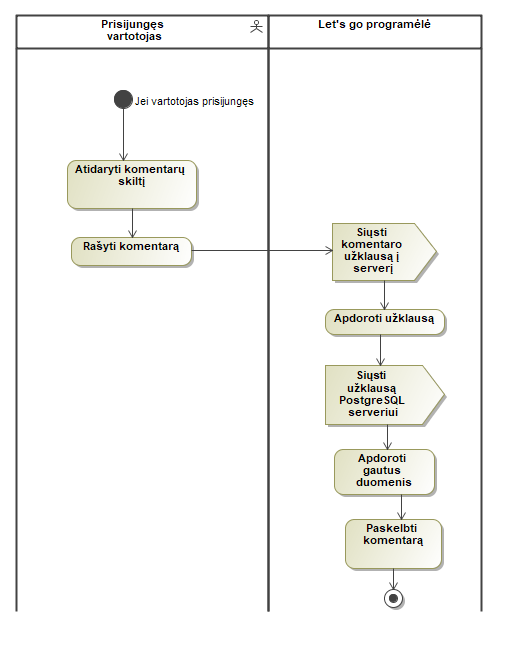
**pav. 2.6** yra pavaizduota „Prisijungti“ panaudojimo atvejo veiklos diagrama.



**pav. 2.6** „Prisijungti“ panaudojimo atvejo veiklos diagrama

Paspaudus prisijungimo mygtuką vartotojas nukreipiamas į „Facebook“ prisijungimo puslapį, kur suveda savo prisijungimo duomenis. Tada būna grąžinamas į programėlės sąsają.

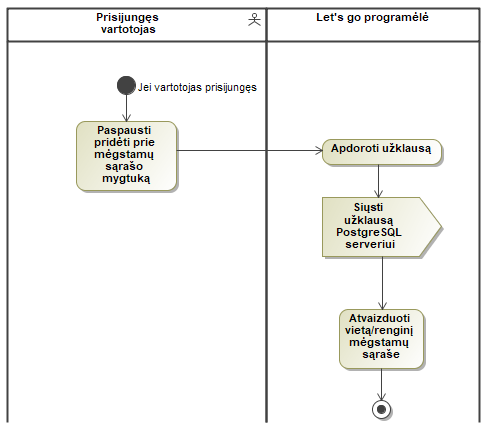
**pav. 2.7** yra pavaizduota „Rašyti komentarą“ panaudojimo atvejo veiklos diagrama.



**pav. 2.7** „Rašyti komentarą“ panaudojimo atvejo veiklos diagrama

Paspaudus mygtuką „rašyti komentarą“ vartotojas nukreipiamas į naują langą, kuriame mato kitų komentarus bei gali parašyti savąjį. Parašius komentarą ir paspaudus „siųsti“ mygtuką, komentaras išsaugojamas duomenų bazėje ir atvaizduojamas šalia kitų komentarų.

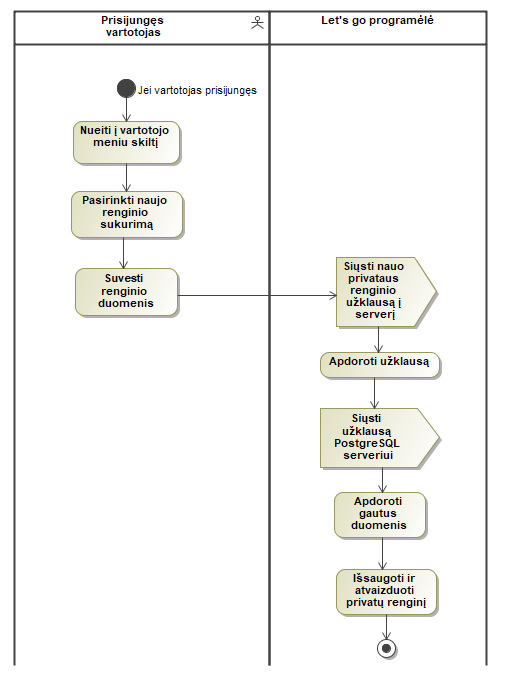
**pav 2.8** yra pavaizduota „Pridėti vietą/renginį prie mėgstamų sąrašo“ panaudojimo atvejo veiklos diagrama.



**pav. 2.8** „Pridėti vietą/renginį prie mėgstamų sąrašo“ panaudojimo atvejo veiklos diagrama

Paspaudus mygtuką „pridėti prie mėgstamų sąrašo“ renginys arba lankytina vieta būna pridedami bei atvaizduojami mėgstamų sąraše.

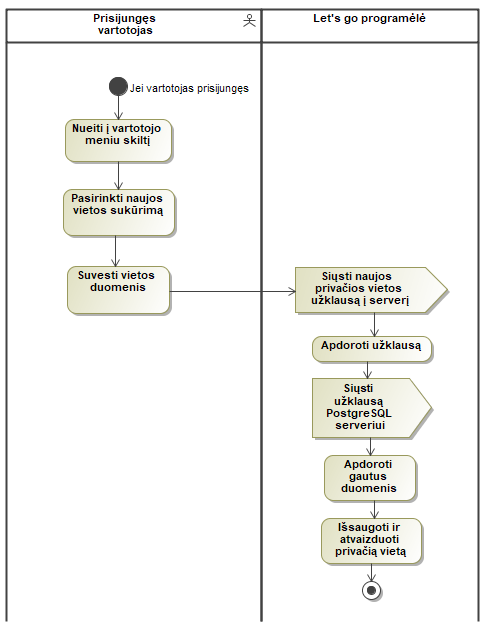
**pav 2.9** yra pavaizduota „Sukurti privatų renginį“ panaudojimo atvejo veiklos diagrama.



**pav. 2.9** „Sukurti privatų renginį“ panaudojimo atvejo veiklos diagrama

Prisijungęs vartotojas meniu skiltyje turi pasirinkimą „sukurti naują renginį“, kurį pasirinkus galės sukurti naują privatų renginį. Kad renginys būtų viešas yra reikalinga profilio verifikacija.

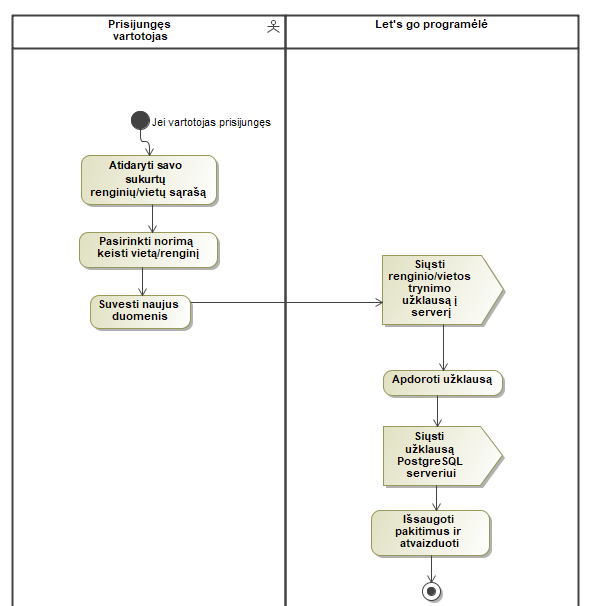
**pav. 2.10** yra pavaizduota „Sukurti privačią vietą“ panaudojimo atvejo veiklos diagrama.



**pav. 2.10** „Sukurti privačią vietą“ panaudojimo atvejo veiklos diagrama

Prisijungęs vartotojas meniu skiltyje turi pasirinkimą „sukurti naują vietą“, kurį pasirinkus galės sukurti naują privačią vietą. Kad vieta būtų vieša yra reikalinga profilio verifikacija.

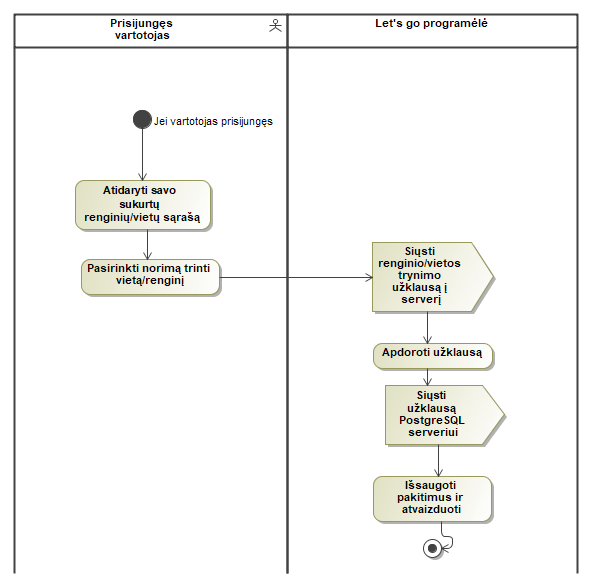
**pav. 2.11** yra pavaizduota „Keisti renginio/vietos informaciją“ panaudojimo atvejo veiklos diagrama.



**pav. 2.11** „Keisti renginio/vietos informaciją“ panaudojimo atvejo veiklos diagrama

Atidaręs savo sukurtų vietų/renginių sąrašą prisijungęs vartotojas paspaudęs ant konkretaus elemento gali keisti jo duomenis.

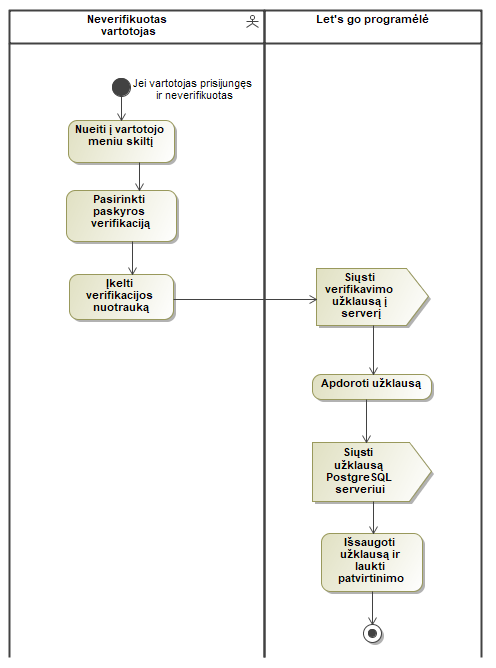
**pav. 2.12** yra pavaizduota „Ištrinti renginį/vietą“ panaudojimo atvejo veiklos diagrama.



**pav. 2.12** „Ištrinti renginį/vietą“ panaudojimo atvejo veiklos diagrama

Atidaręs savo sukurtų vietų/renginių sąrašą prisijungęs vartotojas paspaudęs ant konkretaus elemento gali jį ištrinti.

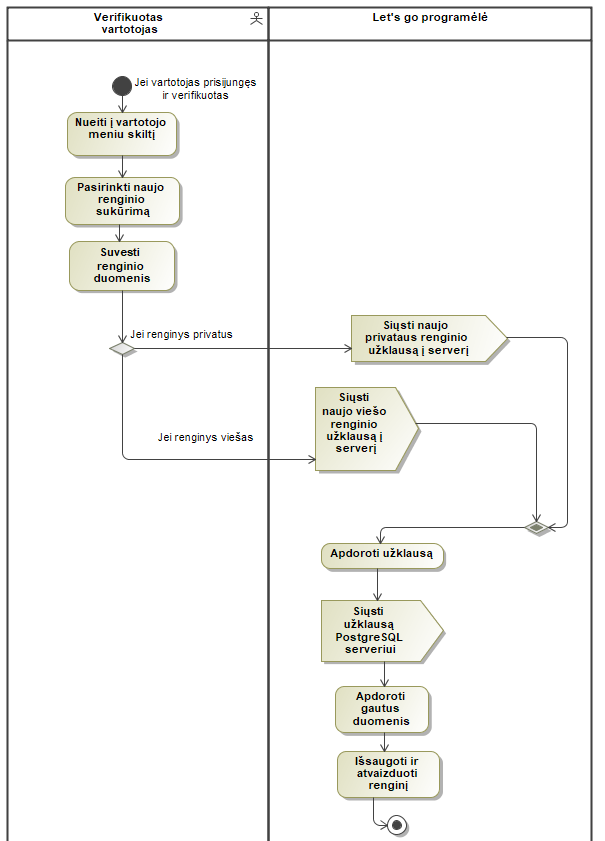
**pav. 2.13** yra pavaizduota „Siųsti paskyros verifikavimo užklausą“ panaudojimo atvejo veiklos diagrama.



**pav. 2.13** „Siųsti paskyros verifikavimo užklausą“ panaudojimo atvejo veiklos diagrama

Jei vartotojas yra prisijungęs ir neverifikuotas, jis negalės kelti viešų, kitiem matomų renginių bei lankytinų vietų. Tokio tipo vartotojui rodomas „verfikuoti paskyrą“ pasirinkimas, kurį pasirinkęs jis sukuria verifikacijos užklausą.

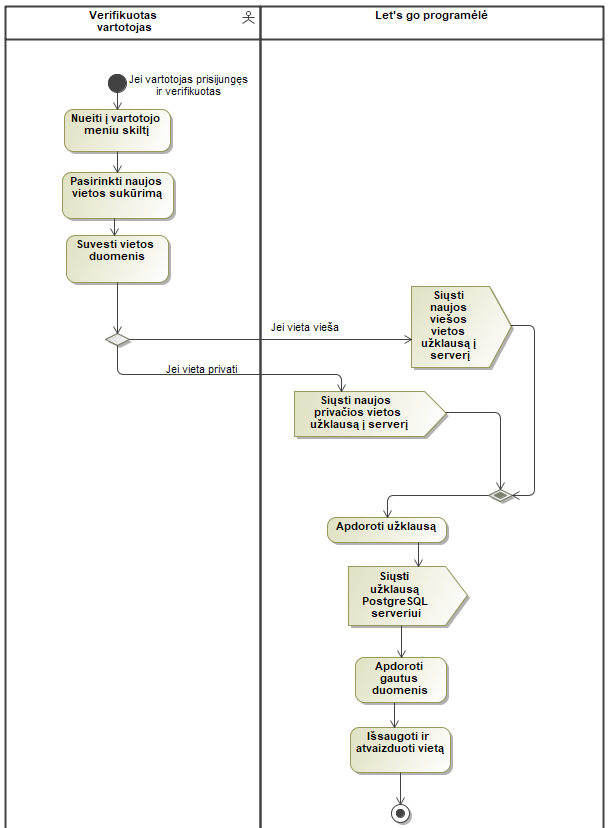
**pav. 2.14** yra pavaizduota „Sukurti viešą renginį“ panaudojimo atvejo veiklos diagrama.



**pav. 2.14** „Sukurti viešą renginį“ panaudojimo atvejo veiklos diagrama

Prisijungęs verifikuotas vartotojas meniu skiltyje turi pasirinkimą „sukurti naują renginį“, kurį pasirinkus galės sukurti naują privatų arba viešą renginį.

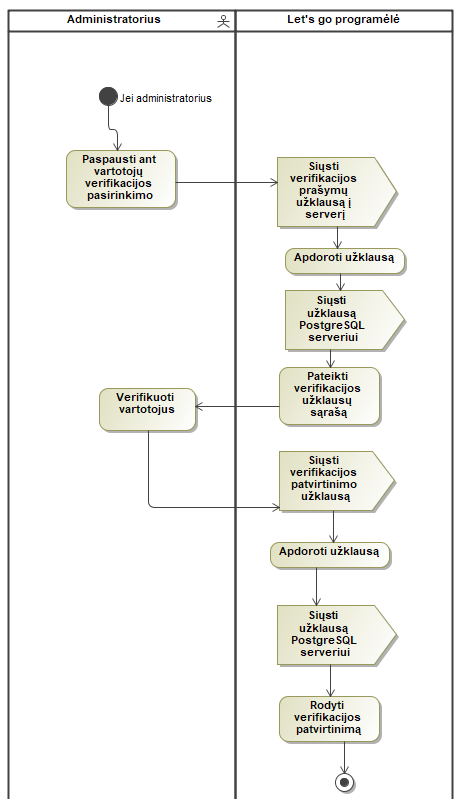
**pav. 2.15** yra pavaizduota „Sukurti viešą vietą“ panaudojimo atvejo veiklos diagrama.



**pav. 2.15** „Sukurti viešą vietą“ panaudojimo atvejo veiklos diagrama

Prisijungęs verifikuotas vartotojas meniu skiltyje turi pasirinkimą „sukurti naują vietą“, kurį pasirinkus galės sukurti naują privačią arba viešą vietą.

**pav. 2.16** yra pavaizduota „Verifikuoti vartotoją“ panaudojimo atvejo veiklos diagrama.



**pav. 2.16** „Verifikuoti vartotoją“ panaudojimo atvejo veiklos diagrama

Prisijungęs administratorius meniu skiltyje turi pasirinkimą „verifikuoti vartotojus“, kurį pasirinkus galės įvertinti vartotojus ir juos verifikuoti.

### Vartotojo sąsajos specifikacija

Vartotojo sąsaja buvo kuriama prototipo kūrimo būdu. Vartotojo sąsaja aprašyta **4.2** skyrelyje.

### Realizacijai keliami reikalavimai

„Let’s go“ programėlei keliami realizacijos reikalavimai:

1. Saugumo reikalavimai:
   1. Duomenų bazėse neturi būti saugoma jokia kritinė vartotojų informacija
   2. Neverifikuotų vartotojų sukurti renginiai bei lankytinos vietos negali būti pasiekiami kitiems naudotojams, nebent bus naudojama „dalintis“ funkcija, kuri leis dalintis sukurto objekto informacija kituose socialiniuose tinkluose
2. Naudojamumo reikalavimai:
   1. Vartotojui nereikia papildomo pasirengimo, kad galėtų naudotis programėle
3. OS reikalavimai:
   1. Minimali palaikoma Android Sdk versija yra Android 5.1 (API level 22)
4. Architektūriniai reikalavimai:
   1. Sistema susideda iš dviejų dalių – klientinės ir serverinės
5. Testavimo reikalaviami
   1. Svarbiausi kodo vienetai turi būti padengti vienetų testais
   2. Svarbios sistemos dalys turi būti testuojamos integraciniais testais

### Techninė specifikacija

„Let’s go“ programėlės eksplotacijos techniniai parametrai ir reikalavimai:

1. Android įrenginys su ne mažesne negu Android 5.1 versija (Lollipop - Marshmallow)
2. Duomenų tarp serverio ir įrenginio perdavimui yra reikalingas interneto ryšys
3. Lokacijos nustatymui ir vietų bei renginių koordinačių paieškai reikalingas GPS ryšys
4. Kad būtų suteikta prieiga prie pilnų sistemos funkcijų, vartotojas turi turėti Facebook paskyrą

## Projektavimo metodai

### Projektavimo valdymas ir eiga

Kuriant „Let’s go“ sistemą buvo naudojamas prototipavimo procesas, kurio eigoje buvo šie žingsniai:

1. Sudaryta reikalavimų analizė, per kurią buvo nustatyti minimalūs sistemos reikalavimai ir funkcionalumai
2. Nustatytas preliminarus sistemos dizainas bei architektūra, kurie apibūdina pagrindinių sistemos funkcijų reikalavimus bei komunikavimą tarp skirtingų komponentų
3. Prototipo kūrimas, per kurį buvo realizuoti minimalūs dizaino bei architektūros reikalavimai bei numatyti planai
4. Atliekami pakeitimai, jei atsirado sistemos reikalavimų ar architektūros pokyčių ar neatitikčių
5. Kai realizuoti visi pagrindiniai reikalavimai, kuriamos papildomos funkcijos, gražinami grafinės sąsajos elementai ir sukuriama pilna sistema

### Projektavimo technologija

„Let’s go“ sistemos projektas buvo sukurtas naudojantis UML diagramomis bei grafiniais elementais. Kiekviena diagrama yra aprašyta papildomai, kad būtų aiškiau apibūdintas funkcionalumas ir, kad būtų paprasčiau suprasti sistemą. Buvo naudojama MagicDraw programinė įranga bražyti panaudos atvejų, komponentų, klasių, veiklos ir sekų diagramoms.

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

Sistema buvo sukurta naudojant Dart ir Python programavimo kalbas. Kliento dalyje buvo naudojama Dart 2.12.2 kalba su Flutter 2.0.3 karkasu, kurie buvo programuojami Android Studio integruotoje programavimo aplinkoje. Serverio programavimui buvo naudojama Python 3.9.1 kalba, kuris buvo atliktas Atom kodo redaktoriuje. Duomenų bazės valdymo sistema buvo PostgreSQL serverio klientas.

## Sistemos projektas

### Statinis sistemos vaizdas

**pav. 2.17** pateikiama UML sistemos diegimo diagrama, kurioje vaizduojamas produkcijos aplinkoje esantis sistemos išdėstymas. Sistema yra kliento-serverio architektūros tipo:

* Serveris:

1. „Web scrapers“ logika
2. Duomenų bazės konfigūracijos ir prisijungimo logika
3. REST užklausų apdorojimo logika

* Klientas:

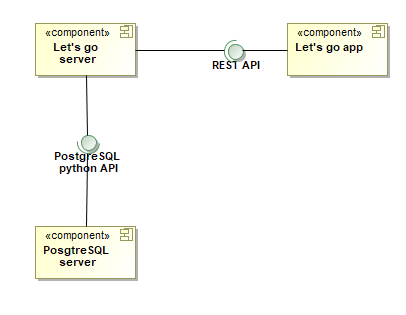
1. Programėlės logika
2. Lokali duomenų bazė

Su klientu taip pat komunikuoja ir Google maps API bei Facebook API.



**pav. 2.17** sistemos diegimo diagrama

„Let’s go“ sistemos komponentų vaizdas pateikiamas **pav. 2.18**:



**pav. 2.18** komponentų diagrama

Komponentai:

* Let’s go server – serverinė sistemos dalis. Komponentas atsakingas už kliento užklausų vykdymo logiką bei komunikaciją su PostgreSQL serveriu.
* Let’s go app – klientinė sistemos dalis. Komponentas atsakingas už pagrindines programėlės funkcijas vartotojo įrenginyje.
* PostgreSQL serveris – pagrindinis sistemos duomenų bazės serveris.

Duomenų bazė yra sudaryta iš atskirų lentelių, kuriose saugomi renginių, vietų, „patikusių“, vartotojų ir komentarų informacija, **pav 2.19**:

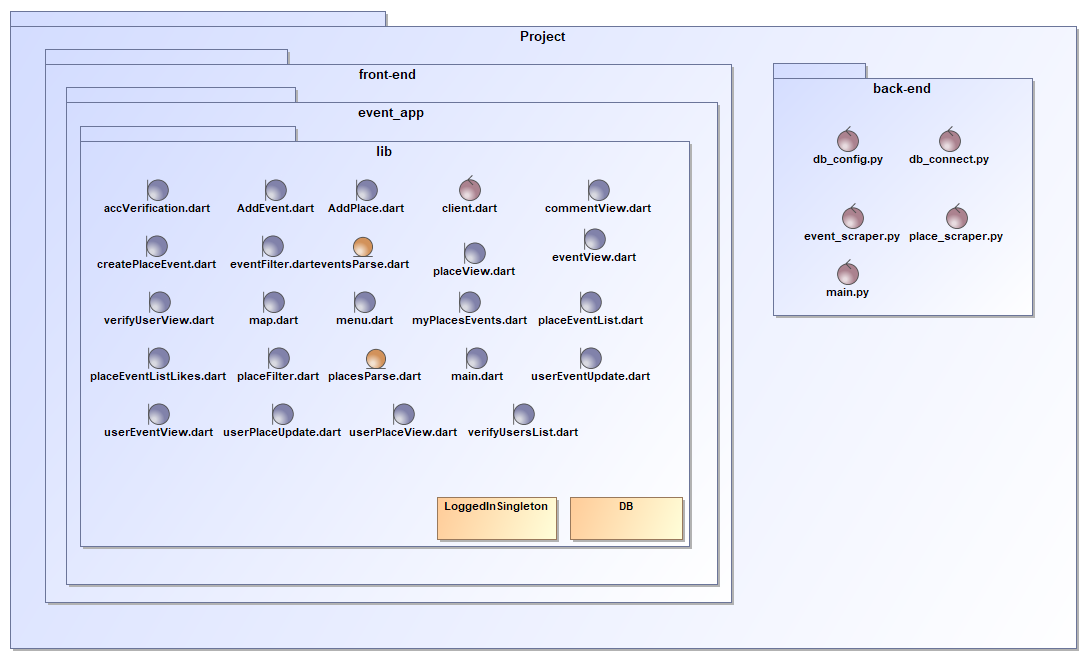


**pav. 2.19** duomenų bazės schemos struktūra

**pav. 2.20** pateikiama sistemos paketų diagrama, kurioje matosi pagrindis visų paketų išsidėstymas.

Sistema skirstoma į pagrindinius du paketus:

* „front-end“, skirtą sistemos klientui
* „back-end“, skirtą sistemos serverio logikai



**pav. 2.20** sistemos paketų diagrama

Toliau pateikiama klientinės dalies - „lib“ aplanko klasių diagrama, **pav. 2.21**. Visi failai ir klasės yra viename aplanke, kadangi Flutter karkaso failų organizavimas kiek skiriasi nuo kitų programavimo technologijų. Kadangi Flutter yra „widget tree“ konceptu paremtas karkasas ir kiekviename faile persipina skirtingos valdiklių funkcijos, tai kategorizavimas būtų labai nežymus. Projektui didėjant ir susidarant aiškiom ribom tarp skirtingų „widget tree“ failų funkcionalumų, paketų skirstymo klausimas taptų aktualesnis.

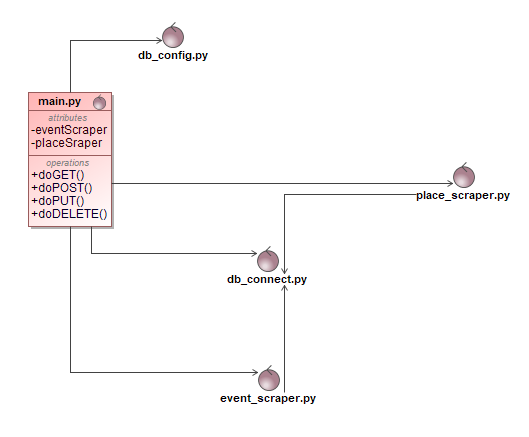
*Papildomas šaltinis: https://blog.codemagic.io/flutter-tutorial-organising-your-flutter-app/*



**pav. 2.21** kliento klasių diagrama

Taip pat serverio klasių diagrama, **pav. 2.22**. Pagrindinė main.py klasė atsakinga už:

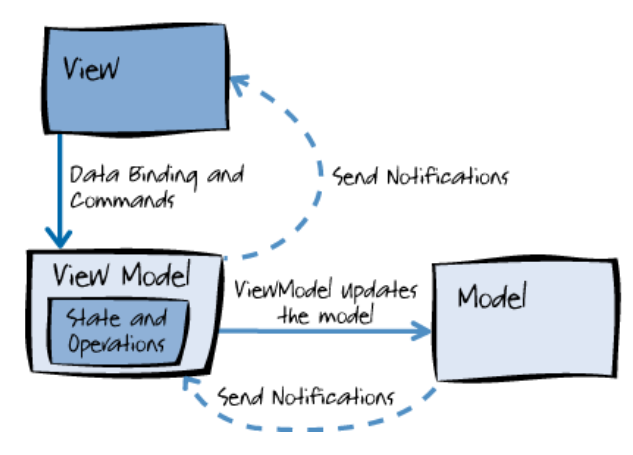
* Komunikaciją su klientine sistemos dalimi
* Komunikaciją su PostgreSQL serveriu
* **Už paralelių gijų paleidimą ir valdymą** - kiekvienoje gijoje veikia skirtingas „web sraper“, kadangi taip greičiau apdorojama renkama informacija ir padidėją sistemos greitis



**pav. 2.22** serverio klasių diagrama

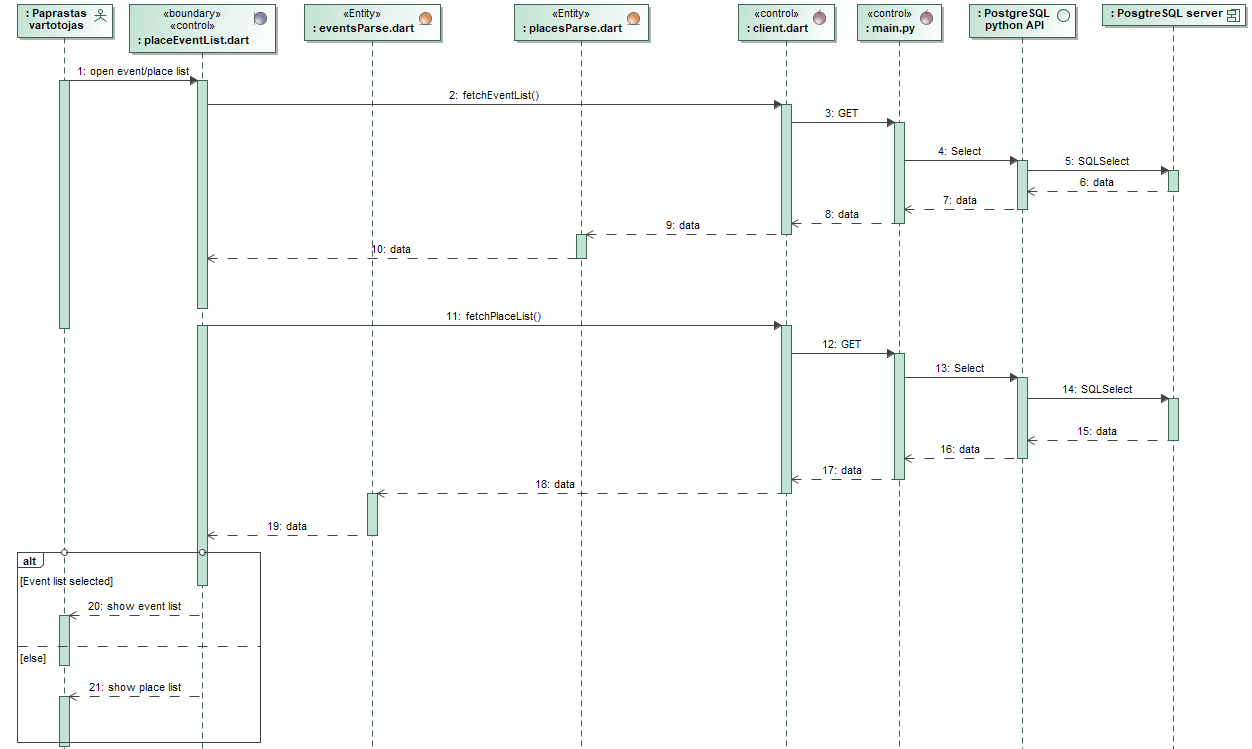
### Dinaminis sistemos vaizdas

Projektui artimiausias architektūrinis modelis yra MVVM, kadangi visi ekrane vaizduojami komponentai taip pat yra ir valdikliai, kitaip tariant „ViewModel“. Be to Dart kalba yra paremta „event-driven“ architektūriniu sprendimu, o visi vaizduojami komponentai, kurie Flutter karkase yra vadinami „widgets“ turi savo atskirą logiką, kas panaikina dalį įprastos MVC architektūros modeliui „controller“ tipo komponentų logikos. Grafinis arhitektūrinio modelio atvaizdavimas parodytas **pav. 2.23**.



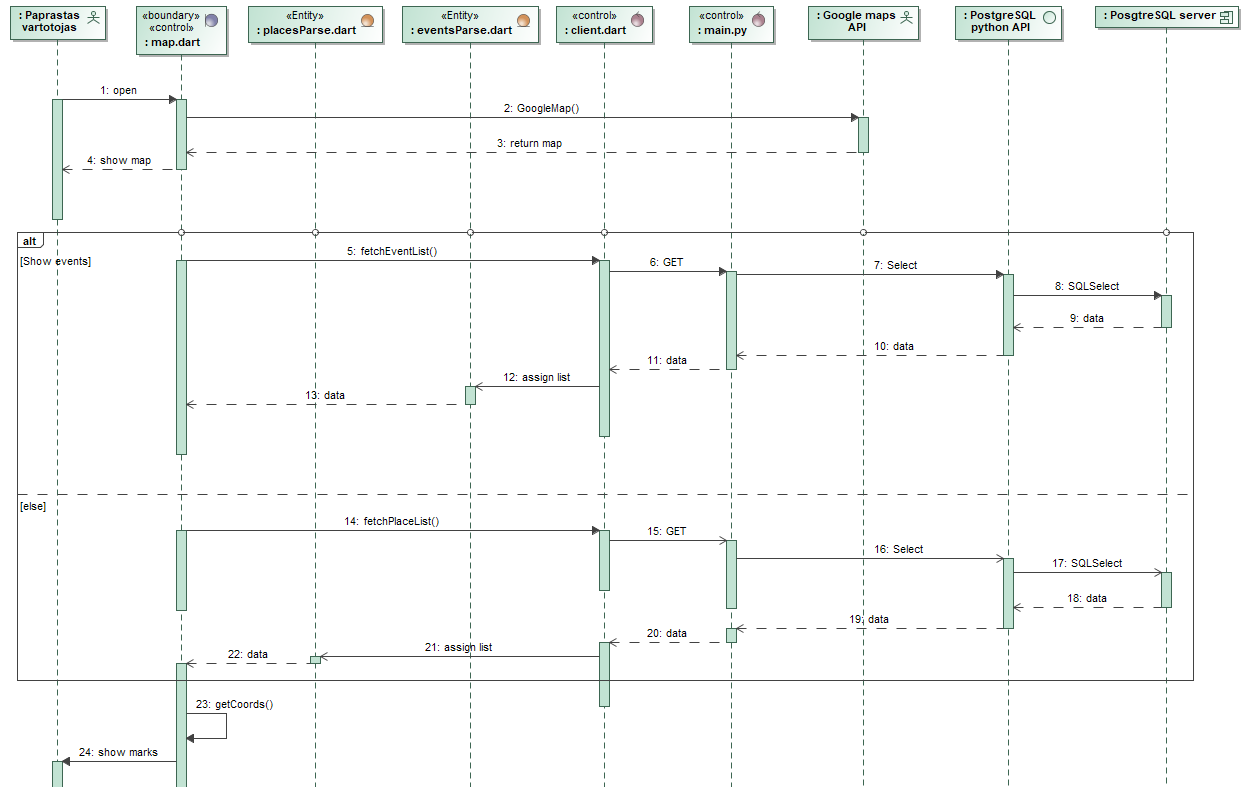
**pav. 2.23** MVVM architektūrinis modelis

**pav. 2.24** yra pateikiama „Peržiūrėti renginių sąrašą“ ir „Peržiūrėti vietų sąrašą“ sekų diagrama. Paprastam vartotojui atidarius renginių sąrašą, „Let‘s go app“ kreipiasi į serverį, kuris grąžina sąrašą. Tada jis atvaizduojamas programėlės ekrane.



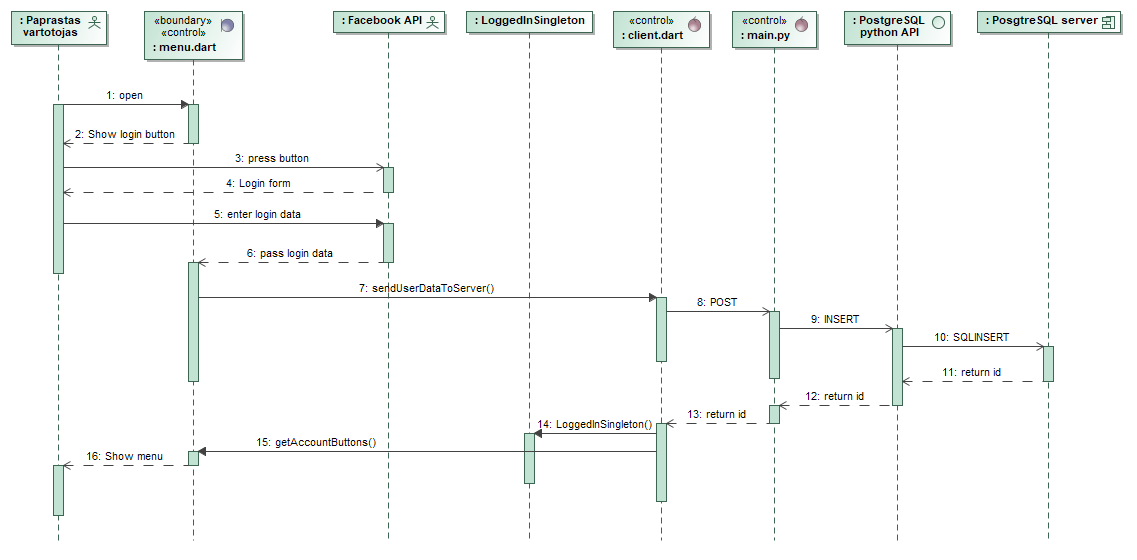
**pav. 2.24** „Peržiūrėti renginių sąrašą“ sekų diagrama

**pav. 2.25** yra pateikiama„Naudotis interaktyviu žemėlapiu“ sekų diagrama. Paprastam vartotojui atidarius žemėlapį, šis pirma užkraunamas iš Google maps API ir tada, priklausomai ar buvo pasirinkta atvaizduoti vietas ar renginius, jie yra prafiltruojami pagal uždėtą filtrą ir atvaizduojami žemėlapyje. Atvaizdavimo koordinatėms gauti taip pat naudojamas Google map API.



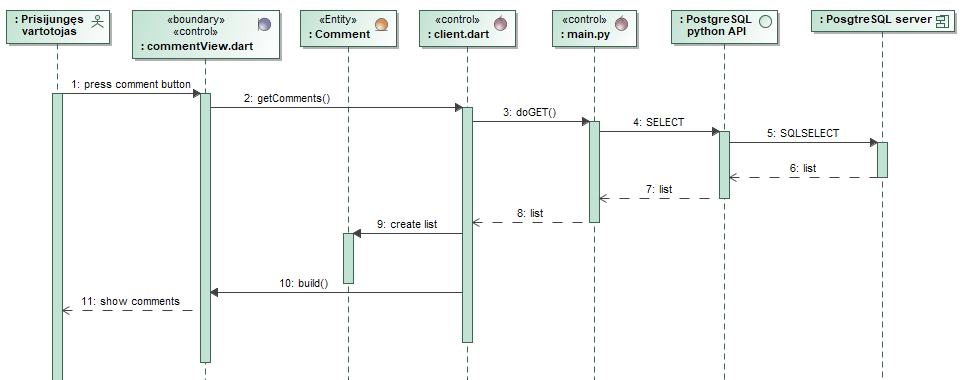
**pav. 2.25** „Naudotis interaktyviu žemėlapiu“ sekų diagrama

**pav. 2.26** yra pateikiama„Prisijungti“ sekų diagrama. Paprastam vartotojui paspaudus prisijungimo mygtuką, įvyksta nukreipimas į Facebook API, kuris ekrane grąžina prisijungimo langą. Suvedus prisijungimo duomenis šie siunčiami į Facebook serverius, kur yra validuojami ir tada grąžinamas prisijungimo sesijos žetonas bei vartotojo id. Pagal vartotojo id nustatomi vartotojo įgaliojimai.



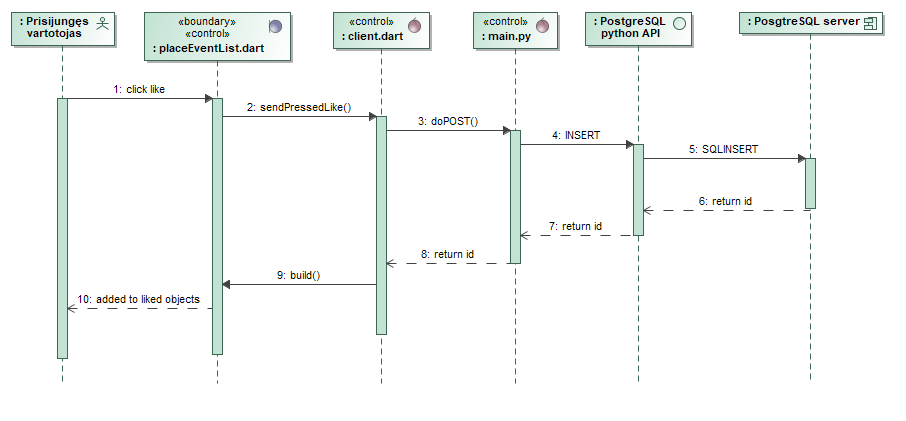
**pav. 2.26** „Prisijungti“ sekų diagrama

**pav. 2.27** yra pateikiama„Rašyti komentarą“ sekų diagrama. Prisijungusiam vartotojui parašius komentarą, šis siunčiamas į serverį, kuris komunikuoja su PostgreSQL serveriu ir išsaugomas duomenų bazėje.



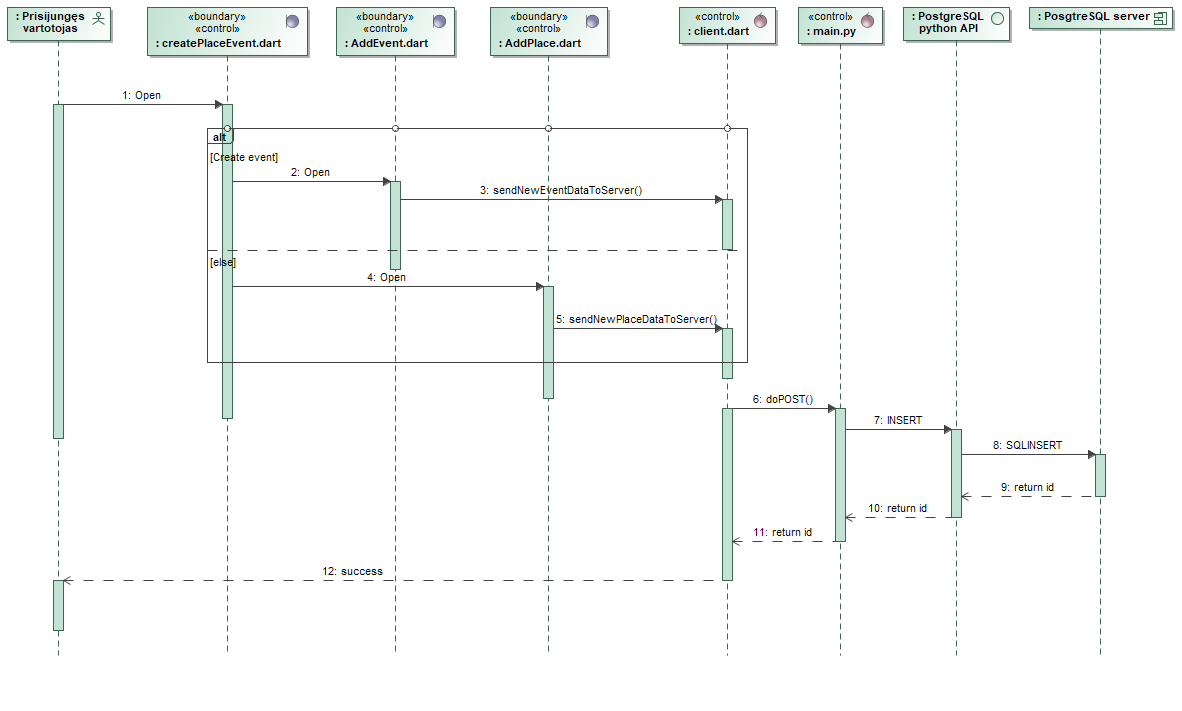
**pav. 2.27** „Rašyti komentarą“ sekų diagrama

**pav. 2.28** yra pateikiama„Pridėti vietą/renginį prie mėgstamų sąrašo“ sekų diagrama. Prisijungusiam vartotojui paspaudus pridėjimo prie mėgstamų sąrašo mygtuką, komanda siunčiama į serverį, kuris komunikuoja su PostgreSQL serveriu ir pridėjimas išsaugomas duomenų bazėje.



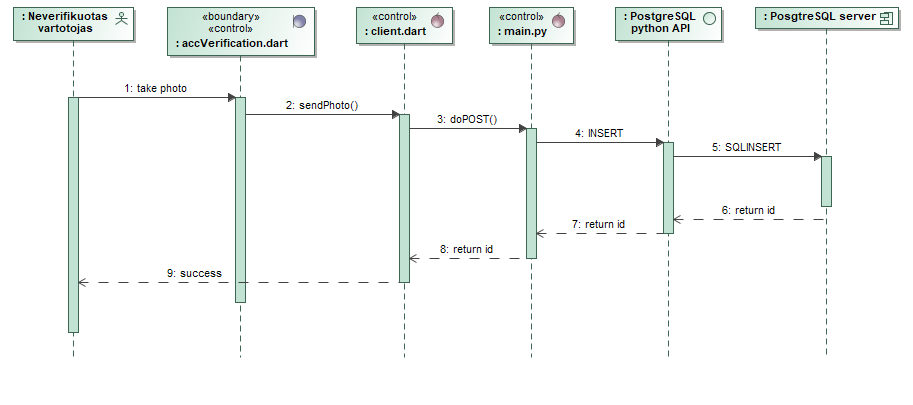
**pav. 2.28** „Pridėti vietą/renginį prie mėgstamų sąrašo“ sekų diagrama

**pav. 2.29** yra pateikiama„Sukurti privatų renginį“ sekų diagrama. Prisijungusiam vartotojui sukūrus naują renginį, šis siunčiamas į serverį, kuris komunikuoja su PostgreSQL serveriu ir išsaugomas duomenų bazėje.



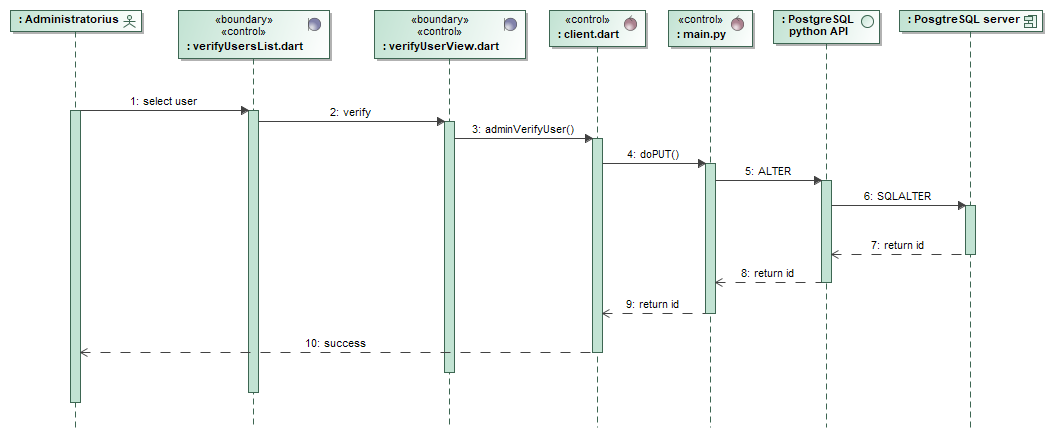
**pav. 2.29** „Sukurti privatų renginį“ sekų diagrama

**pav. 2.30** yra pateikiama„Siųsti paskyros verifikavimo užklausą“ sekų diagrama. Neverifikuotas vartotojas paspaudžia ant verifikavimo mygtuko, užklausa siunčiama į serverį, kuris komunikuoja su PostgreSQL serveriu ir išsaugoma duombazėje.



**pav. 2.30** „Siųsti paskyros verifikavimo užklausą“ sekų diagrama

**pav. 2.31** yra pateikiama„Sukurti viešą renginį“ sekų diagrama. Verifikuotam vartotojui kuriant naują renginį duodamas pasirinkimas „viešas/privatus“. Pasirinkus ir sukūrus naują renginį, šis siunčiamas į serverį, kuris komunikuoja su PostgreSQL serveriu ir išsaugomas duomenų bazėje.



**pav. 2.31** „Sukurti viešą renginį“ sekų 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 eiga:

1. Atliekama statinė kodo analizė buvo vykdoma, kai buvo rašomas programinis kodas. Per kiekvieno naujo funkcionalumo realizavimą buvo atliekama statinė kodo analizė Android studio integruotos programavimo aplinkos bei Flutter karkaso pagalba.
2. Automatiniai vienetų testai buvo rašomi po kiekvieno svarbesnio funkcionalumo įgyvendinimo siekiant ištestuoti ar teisingai veikia pavieniai kodo komponentai.
3. Automatiniai integraciniai testai Papildyti
4. API testai papildyti

Pakomentuoti šiek tiek

## 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.

Pagrindiniai kriterijai yra skirstomi į dvi grupes.

Statinės kodo analizės kriterijai:

1. Negali būti kodo, kuris yra nevykdomas
2. Jei yra specifinių ar sunkiau suprantamų funkcionalumo vietų, jos turi būti komentuojamos
3. Turi būti išlaikyti kodo formatavimo principai
4. Naudoti šiuo metu naujausią Flutter karkaso versiją 2.0.3

Sistemos funkcionalumo kriterijai:

1. Visi serverio API mazgai turi veikti korektiškai ir grąžinti norimus rezultatus
2. Turi būti užtikrinamas įvedamų duomenų validumas
3. Turi būti korektiškai atvaizduojamas koordinačių žymėjimas žemėlapyje

## 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

## Apibendrintas sistemos galimybių aprašymas

„Let‘s go“ programėlė suteikia galimybę matyti kitų sukurtus bei įkeltus renginius ir lankytinas vietas. Prisijungę su Facebook paskyra vartotojai papildomai galės:

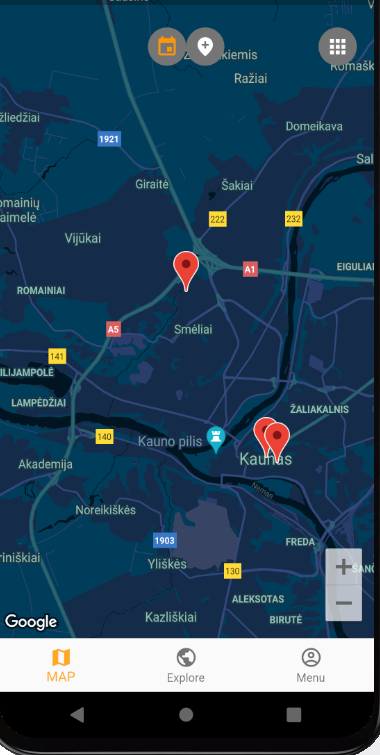
1. Kurti savo asmeninius renginius bei lankytinas vietas sistemoje
2. Pridėti patinkančius prie „pamėgti“ sąrašo
3. Matyti kitų žmonių susidomėjimą bei praeitos savaitęs susidomėjimo statistiką
4. Palikti bei matyti kitų žmonių komentarus po renginiais ir lankytinomis vietomis

Pirmą kartą prie sistemos prisijungęs vartotojas bus neverifikuotas. Neverifikuotas vartotojas turės visas aukščiau išvardintas galimybės, išskyrus viešų, kitiem vartotojam matomų renginių ir lankytinų vietų kūrimą. Jis galės sukurti tik jam sistemoje matomą įrašą. Norėdamas verifikuoti paskyrą bei gauti viešai matomų objektų kūrimo privilegijas, vartotojas turės įkelti tapatybę įrodantį dokumentą.

Sistemos administratorius turi galimybę peržiūrėti vartotojų verifikacijos užklausas bei jas peržiūrėjęs patvirtinti ir taip verifikuoti jų paskyras.

## Vartotojo vadovas

Pirmą kartą atidarius programėlę vartotojas yra nukreipiamas į renginių bei lankytinų vietų žemėlapį paveiklelio nr , kuriame jis galės peržiūrėti renginių bei lankytinų vietų lokacijas.



Paveikslelio nr

Žemėlapyje yra du pasirinkimai- renginiai ir lankytinos vietos paveiklelio nr



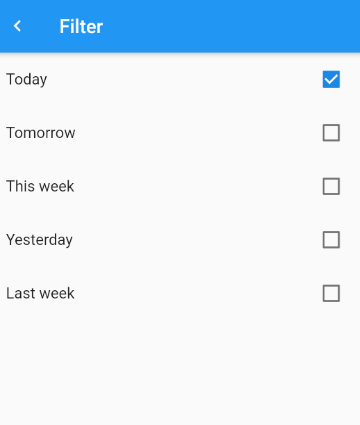
Paveikslėlio nr

Bei filtrų pasirinkimas paveikslėlio nr



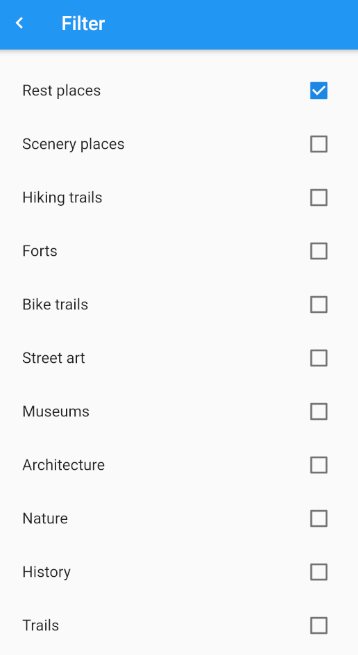
Paveikslelio nr

Paspaudus ant filtravimo mygtuko, vartotojas yra nukreipiamas į naują langą priklausomai nuo zemelapio pasirinkimai foro nr pasirinkimo. Renginių filtras paveiklelio nr



Paveikslelio nr

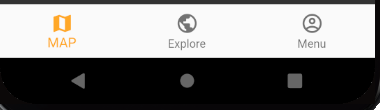
Lankytinų vietų filtras paveikslelio nr



Paveikslelio nr

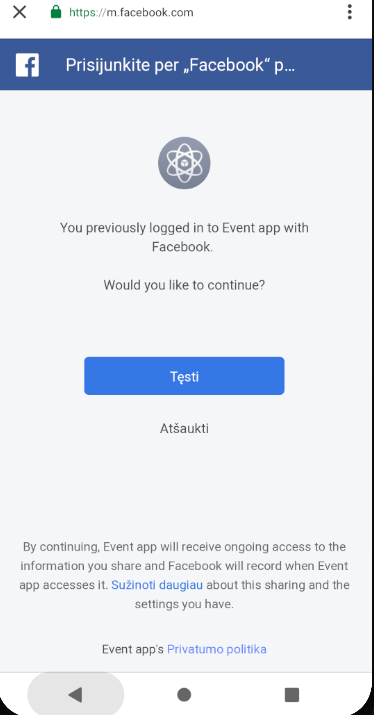
Vartotojas turi tris pagrindinius langus į kuriuos jis gali naviguoti paveikslelio nr:

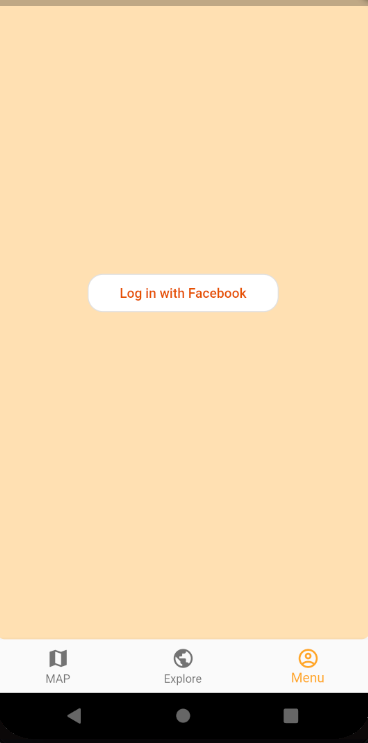
1. Žemėlapis
2. Renginių/vietų sąrašas
3. Vartotojo meniu



Paveikslelio nr

Norėdamas prisijungti vartotojas turės paspausti ant vartotojo meniu ikonos ir prisijungti paveiksleliu nr

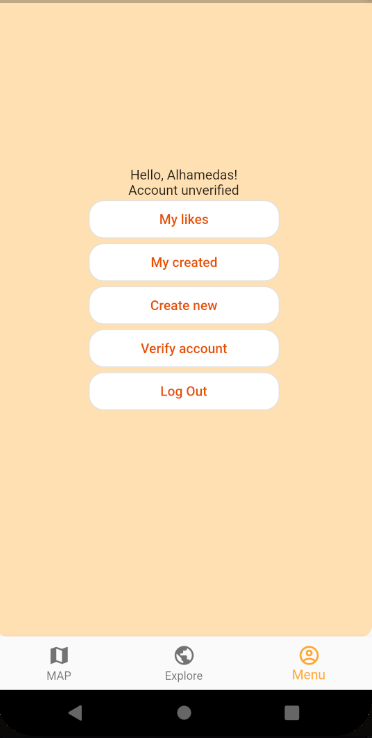




Paveikslelio nr

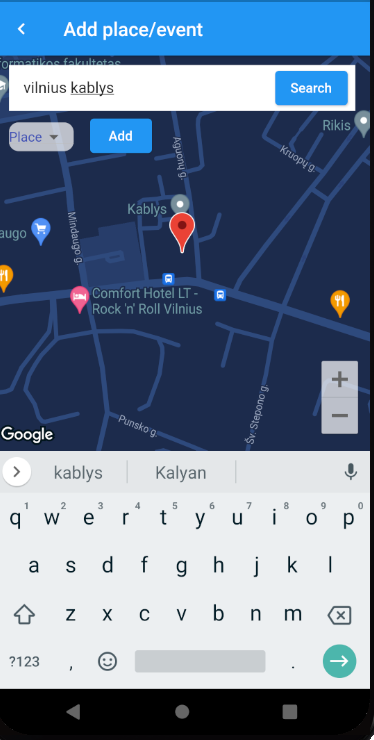
Paveikslelio nr

Prisijungusio (neverifikuoto) vartotojo matomas langas paveikslelio nr



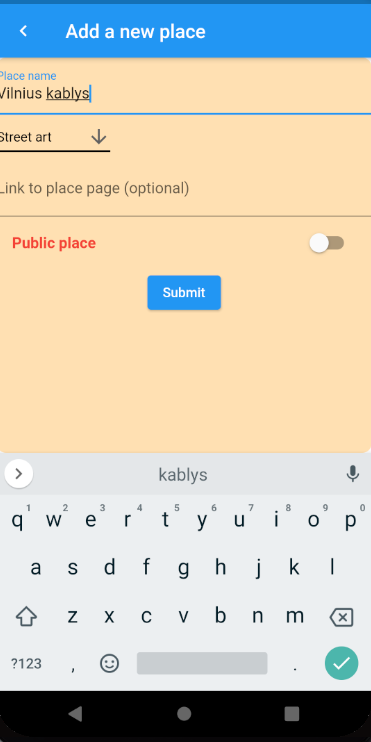
Paveikslelio nr

Paspaudus „Create new“ mygtuką vartotojas bus nukreipiamas į naują langą paveikslelio nr



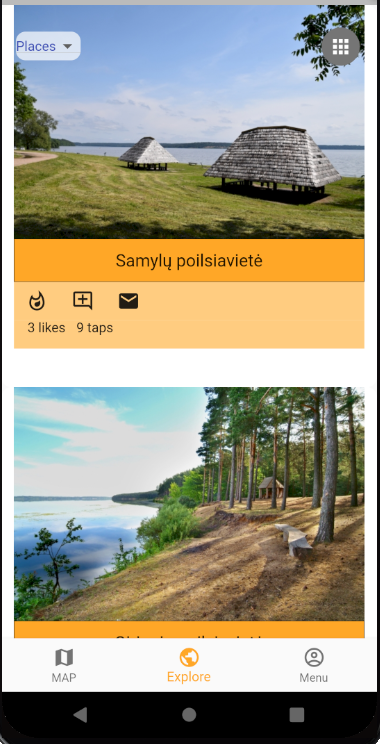
Paveikslelio nr

Kuriamas naujas įrašas (lankytina vieta) paveikslelio nr



Paveikslelio nr

Paspaudus ant sąrašo matomas vaizdas paveikslelio nr



Paveikslelio nr

Paspaudus ant renginio/lankytinos vietos matoma informacija paveiklelio nr



Paveikslelio nr

## Diegimo vadovas

Kadangi sistema sudaryta iš dviejų dalių, tai bus du skirtingi diegimai. Klientinė programėlė diegiama į Android įrenginį, kurio versija ne mažesnė nei 5.1 (API level 22). Šiuo metu pagrindinio serverio bei duomenų bazės failai talpinami lokaliam tinkle, tačiau planuojama kelti į viešą serverį naudojant Docker konteinerį.

## Administravimo vadovas

Pagrindiniai pranešimai yra rodomi serverio pusės komandinėje eilutėje:

1. Prisijungimai prie duomenų bazės
2. „Web scrapers“ funkcionavimas
3. Klientų užklausos
4. Duomenų bazės serverio užklausos

Rezultatai ir išvados

1. Nustatytas kuriamos sistemos poreikis ir pagrįstumas – rinkoje nėra programėlės, kuri sudarytų galimybes matyti vykstančių renginių bei tuo pačiu lankytinų vietų informaciją, ją atvaiduotų žemėlapyje bei sudarytų sąlygas vienoje vietoje rinkti žmonių atsiliepimus ir susidomėjimą
2. Atlikus konkurentų analizę, kurioje buvo vertinami panašūs bei išsiskyrę iš kitų funkcionalumai, nustatyta, kad rinkoje egzistuoja panašių sprendimų į „Let‘s go“ programėlę. Išnagrinėjus konkurentų bei šios sistemos privalumus ir trūkumus, nustatyta, kad pastaroji turi daugiau privalumų ir papildomų funkcijų nei kitos sistemos. Prie išskirtinų funkcijų priskiriama- žemėlapis renginiams ir lankytinoms vietoms, privačių arba viešų renginių/lankytinų vietų kūrimas, komentarų skiltis, susidomėjimo statistikos vedimas.
3. Reikalavimų analizės metu apibrėžti sistemos reikalavimai – planuojamos kurti sistemos funkcijos, panaudos atvejai bei nefunkciniai reikalavimai. Taip pat nustatyti naudotojų tipai – paprastas vartotojas, prisijungęs-neverifikuotas, prisijungęs-verifikuotas vartotojai bei administratorius.
4. Vykusio darbo metu parengtas „Let‘s go“ programėlės projektas, apibrėžti kliento, serverio, duomenų bazės, facebook serverių, google location serverių komponentai ir sąveika tarp jų. Sudarytas architektūrinis planas bei modeliai buvo sėkmingai įgyvendinti sistemos realizacijos etape.
5. Sistema ištestuota sėkmingai praplest
6. Sistema sėkmingai sukurta naudojant Dart ir Python programavimo kalbas. Kliento dalyje buvo naudojama Dart 2.12.2 kalba su Flutter 2.0.3 karkasu, kurie buvo programuojami Android Studio integruotoje programavimo aplinkoje. Serverio programavimui buvo naudojama Python 3.9.1 kalba, kuris buvo atliktas Atom kodo redaktoriuje. Duomenų bazės valdymui buvo pasirinktas PostgreSQL serverio klientas. Pasirinktos technologijos sudarė galimybes patogiau kurti bei plėtoti „Let‘s go“ programlėlę.
7. Šiuo metu „Let‘s go“ programėlė nėra įkelta Į Play Store ar kitas alternatyvias programėlių parduotuves, kadangi dar planuojama perkurti dalį esamo bei pridėti naujo funkcionalumo. Vieni iš planuojamų pridėti funkcijų – žmonių skaičius renginyje/vietoje, rekomenduojamų renginų/vietų skiltis, įvykusių renginių skiltis, įvykusių renginių nuotraukų bei video skiltis ir kita.

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.

1. <https://www.wintellect.com/model-view-viewmodel-mvvm-explained/>
2. <http://www.vilniusgo.lt/>
3. <https://www.eventbrite.com/>
4. <https://wit-hdip-computer-science.github.io/semester-2-ent-web-development/topic-16-aurelia-2/talk-1-ui-patterns/ui-patterns.pdf>
5. Sekų diagramų taikymai MVC Project ir komponentai Diegimo diagramos T120B029 2021 pavasaris
6. <https://www.mzirafos.lt/>
7. https://blog.codemagic.io/flutter-tutorial-organising-your-flutter-app/

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

**2 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. |