VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ KATEDRA

FoodKept

ANTRASIS LABORATORINIS DARBAS

Programų sistemų inžinerija II

Darbą atliko: Programų sistemų II kurso studentai
Andrius Tumšys,
Arvydas Venskus,
Danielius Rėkus,
Ema Sinkevičiūtė

Vilnius
2022

Santrauka

Kadangi mes toliau mokomės programų inžinerijos principų ir taikome juos kurdami realią programą, šiame darbe projektuojame sistemą ir reikiamus pakeitimus. Nors pirmajame laboratoriniame darbe daugiausiai analizavome verslo principus, šį kartą daugiausiai dėmesio skirsime esamos sistemos elementų sąveikai ir jos atitikmenį keliamiems reikalavimams. Šiam laboratoriniam darbui mums buvo skirtos šios užduotys:

1. Architektūros dokumentui tvarkyti naudoti UML 4+1 sistemą
2. Įgyvendinti numatytus sistemos pakeitimus
3. Pristatyti CI/CD procesą

Tikėtinas darbo rezultatas yra sistemos architektūros dokumentas, įgyvendinti pakeitimai ir įdiegtas CI/CD procesas.

Turinys

1. Kontekstas

Kad kiekviena sistema būtų sėkminga, ji turi būti sukurta siekiant išspręsti tam tikrą realaus pasaulio problemą. Net labiausiai suplanuota ir gerai sukurta programinė įranga yra nenaudinga, jei ji neišsprendžia reikiamos problemos. Šioje dalyje analizuojame, kokia problema yra sprendžiama ir kaip ją ketinama spręsti.

1.1 Sistemos tikslas

Sumažinti maisto švaistymą.

1.1.1 Problema

Daugeliu atvejų besibaigiantis galioti maistas yra švaistomas. Tokie produktai maisto gaminimu užsiimančiose įmonėse yra minimaliai/visai neparduodami. Todėl tai yra didelė maisto švaistymo problema.

1.1.2 Sprendimas

Šiai problemai išspręsti buvo sukurta sistema, leidžianti pardavėjams susikurti savo paslaugų profilius, o prisijungusiems pirkėjams – ieškoti produktų. Pagrindiniai atributai, leidžiantys pasirinkti maisto produktus yra šie:

1. Produkto pavadinimas
2. Įstaigos pavadinimas
3. Produkto kategorija

Paieškos atributas yra labai svarbus, nes leidžia pirkėjui efektyviau surasti ieškomą prekę.

1.1.3 Pagrindiniai vartotojų tikslai

Naudotojų bazė gali būti skiriama į dvi grupes:

1. Pardavėjai
2. Pirkėjai

Pardavėjai labiausiai domisi reklamos paslaugomis ir klientų pritraukimu. Antrosios grupės pagrindinis tikslas – efektyviai surasti paslaugas, kurios parduoda/teikia kokybišką produkciją/paslaugas prieinama kaina.

1.2 Planuojami pakeitimai

Siekdami toliau tobulinti ir padidinti esamos sistemos funkcionalumą, gavome užduotį įgyvendinti keletą pakeitimų ir funkcijų. Patobulinimai apima pirkėjo paieškos palengvinimą, tiekėjo paslaugų galimybių išplėtimą įtraukiant sąrašus į paslaugas ir sukuriant pagrindines moderavimo funkcijas.

1.2.1 Pakeitimų sąrašas

Kadangi pirmajame dokumente buvo pateikti išsamesni reikalavimai, įtraukiame tik pirminius suinteresuotųjų šalių reikalavimus:

1. Sukurti aplikacijų programavimo sąsają (API), per kurią pardavėjai galėtų patogiau užregistruoti savo parduodamas prekes ar žaliavas.
2. Pardavėjams turi būti galima pridėti naujas prekes, redaguoti jau įkeltas prekes, ištrinti jas.
3. Pirkėjas privalo turėti galimybę parašyti atsiliepimą apie perkamą/parduodamą produktą
4. Sukurti administratoriaus rolę, galinčią trinti netinkamą naudotojų paskyrų turinį
5. CI/CD

1.2.2 Pakeitimų poveikis

Naujai sukurta aplikacijų programavimo sąsaja pardavėjams palengvintų registruojamų produktų procesą. Pridedamos naujos funkcijos pirkėjams, leis lengviau surasti ir atsirinkti kokybiškesnius ir pigesnius maisto produktus. Administratoriaus rolė ir atsiliepimai leidžia nuodugniau tvarkyti įrašus, apžvalgas ir kitą naudotojų sukurtą turinį. Taip bus sustabdyta netikra reklama, klaidinančios apžvalgos ir netinkamas turinys.

1.3 Dabartinės sistemos analizė

Kadangi mes nedalyvavome kuriant pirminę sistemą, buvo svarbu išanalizuoti dabartinę sistemą ir nuo pagrindų rasti jos apribojimus, kad galėtume išsiaiškinti ir dokumentuoti esamas problemas bei atsižvelgti į ribas kurdami pakeitimus.

1.3.1 Sistemos aplinka.

Pagrindinis internetinės aplikacijos serveris bus paliestas Windows OS įrenginyje naudojant IIS Express. Duomenų bazė bus paleista iš atskiro Linux serverio. Klientai galės prisijungti naudodami bet kokio tipo naršyklę. Senesnių naršyklių (pvz., „Internet Explorer“) palaikymas neįtrauktas.

1.3.2 Įrankiai ir technologijos

Pagrindinius kūrimo įrankius ir technologijas, kurios yra .NET Core programinės įrangos sistema ir ASP.NET serverio žiniatinklio programų sistema, nulėmė pirminiai universiteto pratybų reikalavimai. Jie puikiai tinka turimai sistemai, todėl galima lengvai ir greitai kurti žiniatinklio programas. Kitas reikalavimas buvo naudoti reliacinę duomenų bazę, ypač duomenų bazių valdymo sistemą, pagrįstą SQL. Pradinių kūrėjų pasirinkta duomenų bazių valdymo sistema buvo PostgreSQL, kuri yra nemokama ir atviro kodo, taip pat puikiai tinka mūsų sistemai.

Apskritai kuriamos sistemos įrankiai ir technologijos yra tinkamai parinkti.

1.3.3 Esančios problemos

Sistemos testavimo ir kodo tikrinimo metu sudarėme (nebaigtinį) esamų ribų ir problemų sąrašą:

1. Kode yra nemažai tekstų ir paveikslėlių, kurie galėtų būti saugomi duombazėje, kad galima būtų juos lengvai pakeisti kitais. Taip pat kai kuriose skiltyse paveikslėliai turėtų būti konfiguruojami
2. Kodas nėra dokumentuotas, tai pat nėra ir komentarų.
3. Nesuvaldomos kai kurios duomenų įrašymo klaidos, kas padėtų išvengti nesusipratimų
4. Trūksta įvedamų duomenų validacijos ne duomenų bazėje.
5. Pardavėjų įkeliami paveikslėliai nėra validuojami, todėl gali greitai užpildyti diską, taip pat gali būti įkeliamos ir kenkėjiškos programos
6. Trūksta požymių kurie laukai formose yra privalomi.
7. Atsiliepimų skiltyje galima rašyti neribotą kiekį atsiliepimų, kenkėjiškos programos galėtų tuo pasinaudoti ir trikdyti tinklapio veikimą
8. SMTP serveris nėra konfiguruojamas, o tiesiog įrašytas kode.

1.4 Plėtros valdymo aplinka

Versijų valdymo sistemos, šaltinio kodo valdymo sistemos daugelis kitų aspektų, kurie yra programinės įrangos konfigūracijos valdymo dalis, atlieka svarbų vaidmenį bet kuriame šiuolaikiniame programinės įrangos kūrimo projekte, labai pagerindamos produkto kokybę. Todėl nusprendėme trumpai paaiškinti, kaip atrodo mūsų darbo aplinka.

Mūsų versijų valdymo sistema yra Git, kurią pasirinkome dėl plačiai paplitusio naudojimo. Šaltinio kodas yra saugojamas GitHub, nes manome, kad jį lengviau naudoti nei kitus saugyklų valdymo įrankius, taip pat visa mūsų komanda yra su juo susipažinusi. Taip pat įdiegėme paprastą CI/CD procesą, kad automatiškai patikrintume, ar nėra stiliaus ir kūrimo klaidų.

2. 4+1 Architektūrinio Vaizdo Modelis

Siekdami geriau suprasti sistemą, jos esamą ir būsimą įgyvendinimą bei apibūdinti sąveikų tarp objektų ir procesų sekas, sudarėme diagramų rinkinį, naudodami UML 4+1 architektūrinio vaizdo modelį. Keturi modelio požiūriai yra loginis, vystymosi, procesų ir fizinis vaizdas. Be to, architektūrai iliustruoti naudojami pasirinkti panaudos atvejai arba scenarijai.

2.1 Loginė analizė

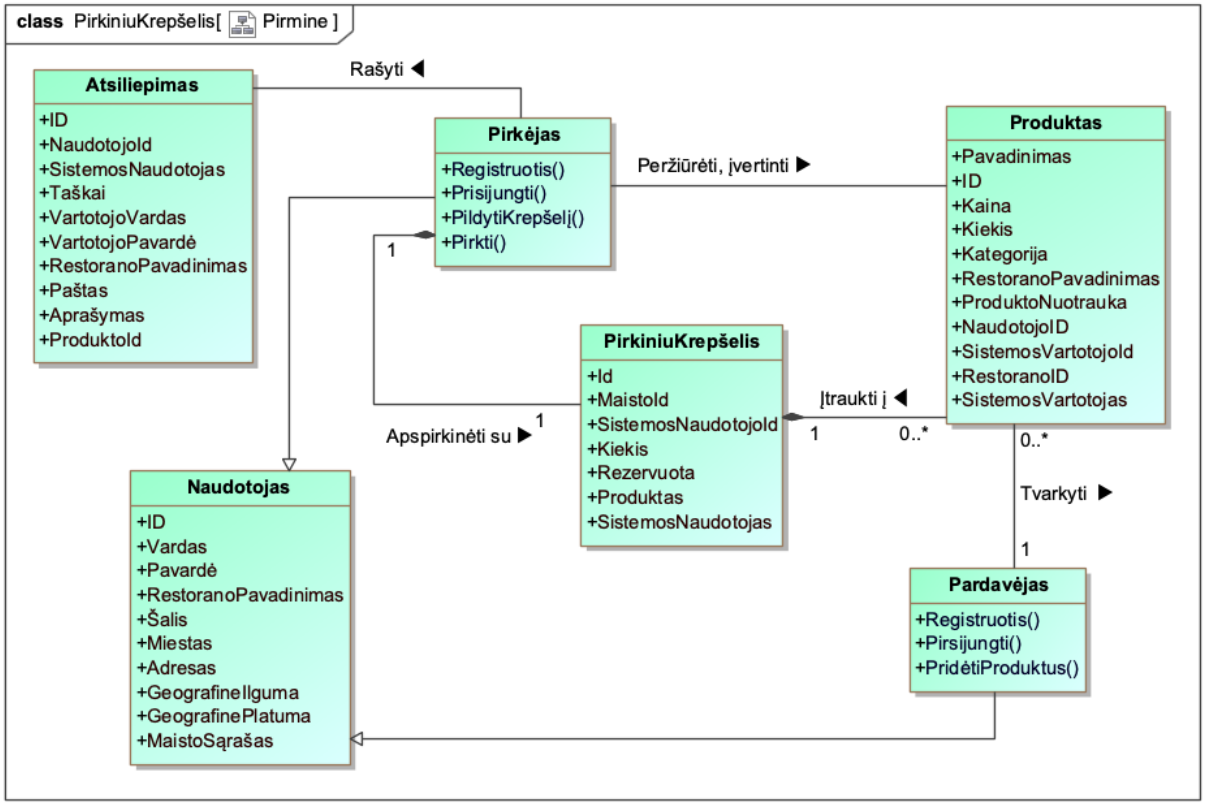
Loginė analizė aprašo funkcionalumą, kurį sistema teikia galutiniams vartotojams. Tai pasiekiama naudojant šias diagramas:

1. Klasių diagramos,
2. Objektų diagramos,
3. Ryšių diagrama,
4. Būsenų diagramos.

Kiekviena iš šių diagramų turi atskirą skyrių, kuriame pateikiamos diagramos ir aprašymai.

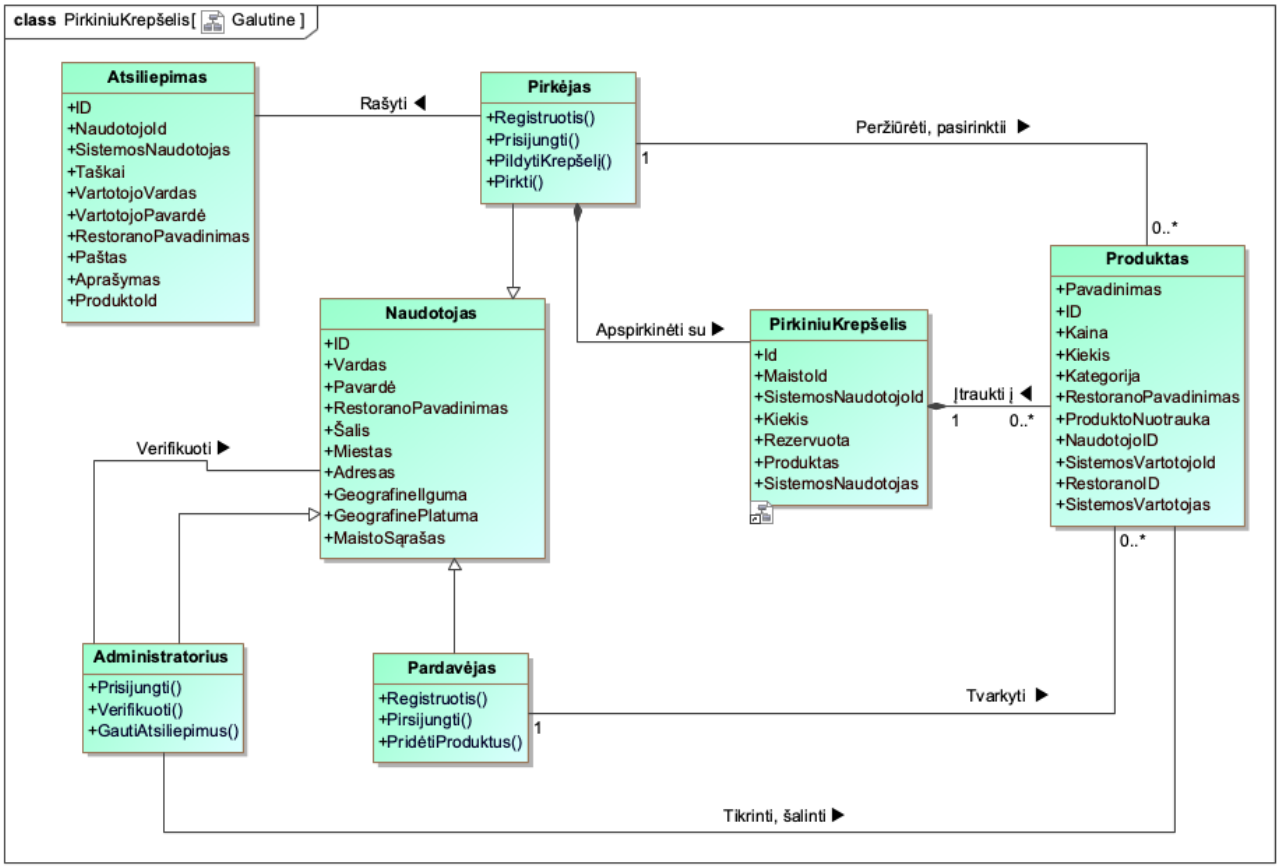
2.1.1 Klasių diagramos

Klasių diagrama prieš pakeitimus, vaizduojama 1 pav., rodo bendrą pagrindinių kodo dalių struktūros dabartinę būseną prieš bet kokius papildymus. Ši diagrama padeda geriau suprasti kodo struktūrą ir gali būti naudojama kaip atskaitos taškas kuriant kodą, nes joje yra pagrindinė kodo architektūra.



Pav. 1. Klasių diagrama prieš pakeitimus

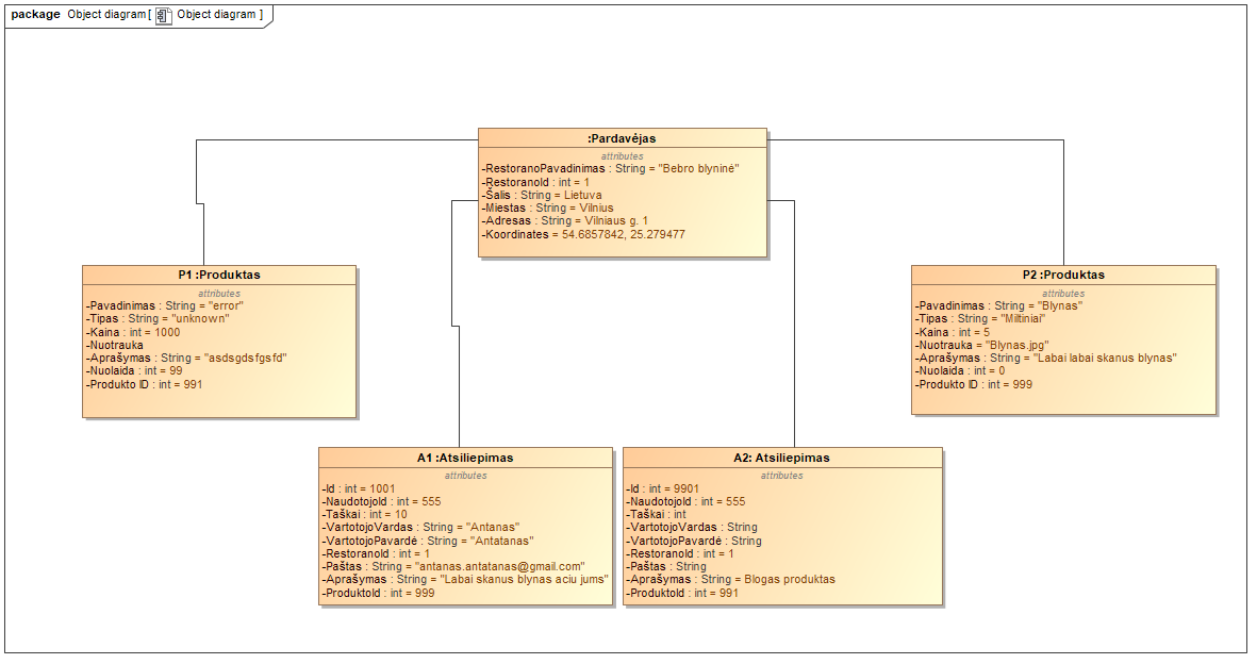
Klasių diagrama po pakeitimų parodyta 2 pav., joje vaizduojama kaip pasikeis bendras sistemos funkcionalumas įgyvendinus suplanuotus pakeitimus. Pavyzdžiui, „Administratoriaus“ ir „Pirkėjo“ klasės yra labai svarbios norint įvykdyti reikalavimus. Jų reikalingi metodai ir (arba) atributai, rodomi bendrame vaizde, pakankami, kad būtų sukurtas planas. Svarbiausia, kad reikalavimų pridėjimo modifikacijos turėtų būti nedidelės, todėl kodavimo procesą dažniausiai sudarys naujo kodo pridėjimas, o ne esamo kodo keitimas.



Pav. 2. Klasių diagrama po pakeitimų

2.1.2 Objektų Diagrama

Objektų diagrama, parodyta 3 pav., iliustruoja „Pardavėją“ ir su juo susijusius objektus konkrečiu laiko momentu. Objektai užpildyti demonstraciniais duomenimis, kad būtų parodytas klasių diagramos pavyzdys pagal planuojamus pakeitimus. Čia reikia sutelkti dėmesį į objektą „Produktas“, nes jis dar turi būti integruotas. Iš esmės diagrama yra vienas iš pavyzdžių, kaip sistemos duomenys gali atrodyti su užbaigtais diegimais po naudojimo. Tai gali būti naudojama kaip paprastas testas po kodavimo proceso. Jei ši situacija atkuriama programoje, tai reiškia, kad įtraukimo į sąrašą reikalavimas yra baigtas ir veikia.

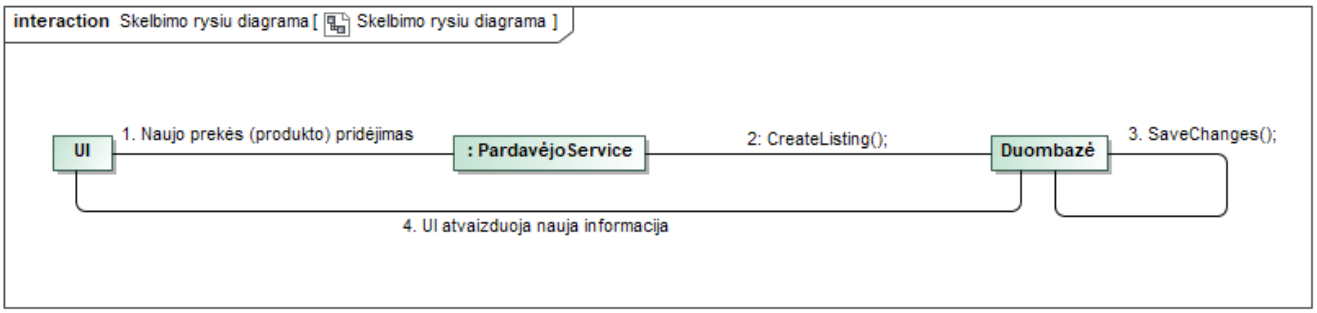


Pav. 3 Objektų diagrama

2.1.3 Ryšių diagrama

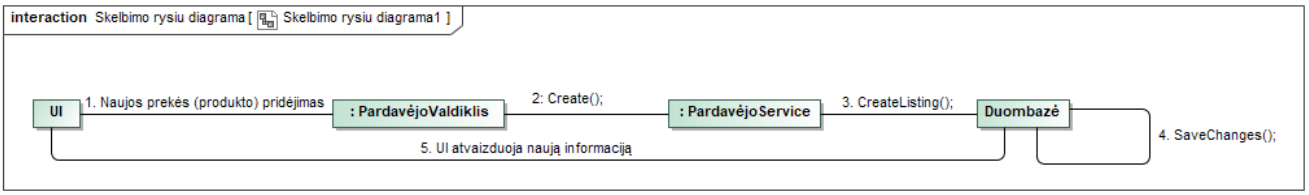
Ryšių diagramos šiame dokumente naudojamos sistemos veikimo principams ir objektų tarpusavio komunikacijai atvaizduoti. Kadangi naujuose reikalavimuose numatytas API naudojimas, ši diagrama padeda suvokti, kaip objektai komunikuoja tarpusavyje.

Diagrama pavaizduota 4 pav. vaizduoja naujos prekės (produkto) sukūrimo procesą prieš pakeitimus.



Pav. 4. Naujos prekės (produkto) sukūrimo diagrama prieš pakeitimus

Diagrama, kuri vaizduojama 5 pav. parodomas naujos prekės (produkto) sukūrimas įvykdžius pakeitimus. Galima pastebėti, jog komunikavimas su duomenų baze iš tiesioginio pakeistas į tarpinį per valdiklį, tai padaro įtaką aplikacijų programavimo sąsajos realizavimas.

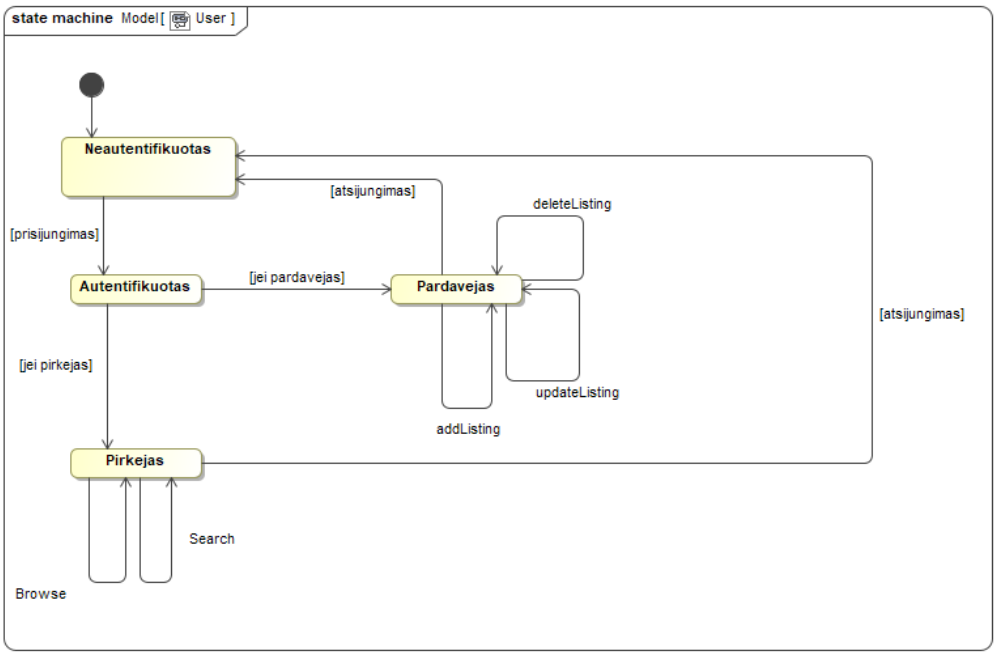


Pav. 5. Naujos prekės (produkto) sukūrimo ryšių diagrama po pakeitimų

2.1.4 Būsenų diagramos

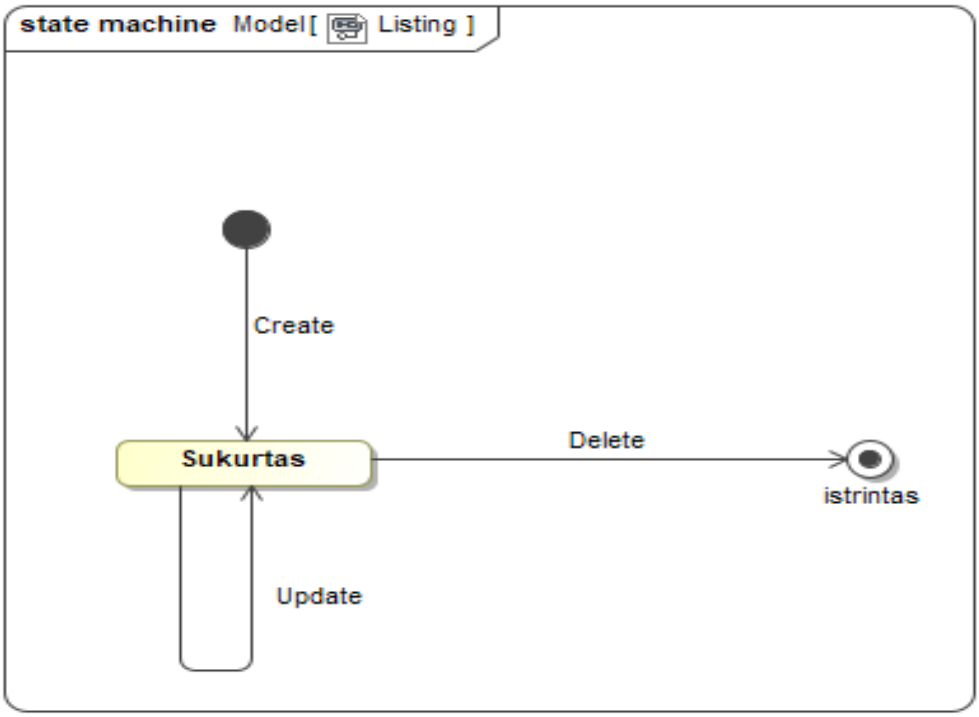
Šiame skyriuje būsenų diagramos naudojamos pavaizduoti sistemos esybių būsenoms bei jų pokyčiams.

Pav. 6 pavaizduota naudotojo būsenų diagrama. Naudotojas gali būti *autentifikuotas* (prisijungęs arba prisiregistravęs) arba *neautentifikuotas*. Svarbu tai, kad autentifikuotas vartotojas gali naudotis visais aplikacijos privalumais, kaip ieškoti perkamų produktų ir/ar ieškoti potencialių pirkėjų.

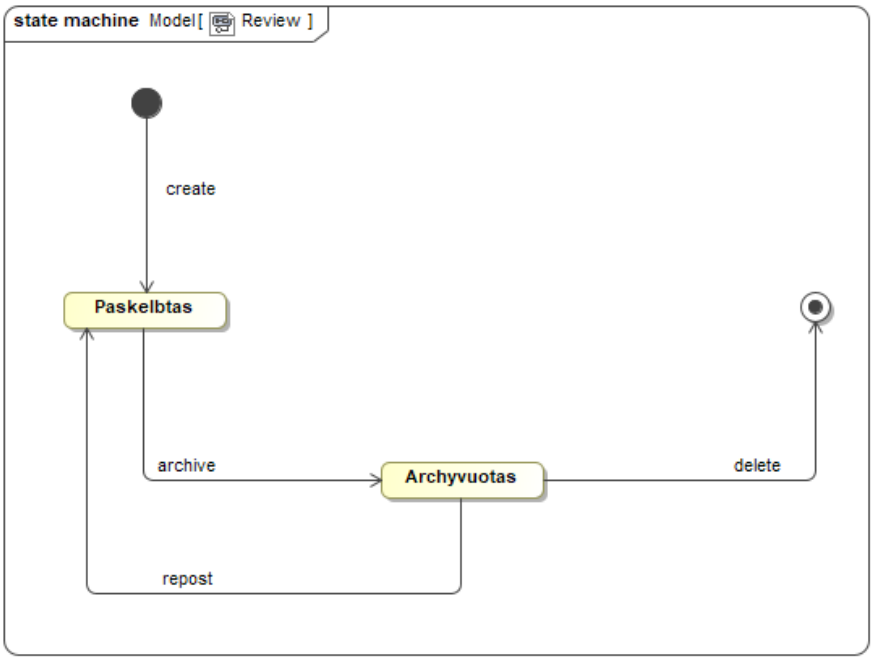


Pav. 6. Naudotojo Būsenų diagrama

Kita svarbi aplikacijos dalis turinti būsenas yra produktų įkėlimas. Pavaizduotas pav, 7. Išsaugotas produktas tik po jo paskelbimo gali būti modifikuojamas ir/ar ištrinamas. Įdiegus naują funkciją, kuri archyvuoja įkeltus produktus duomenų bazėje, keičiasi būsenų diagrama, tai galima pamatyti pav. 8. Pavaizduota, kad išsaugojus produktą informacija apie jį yra archyvuojama.



Pav. 7 Produkto įkėlimo Būsenų diagrama prieš pakeitimus



Pav. 8. Produkto įkėlimo Būsenų diagrama po pakeitimų

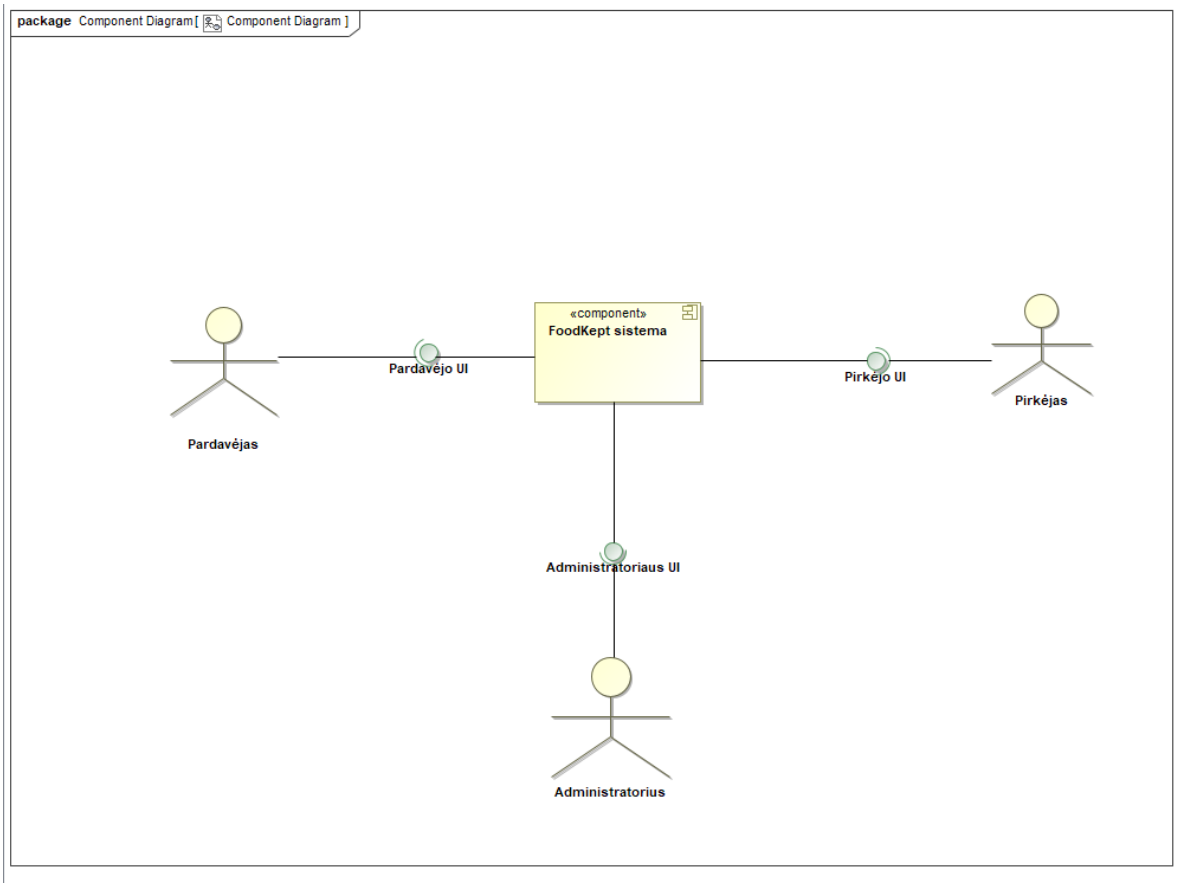
2.2 Plėtros analizė

Plėtros analizė iliustruoja sistemą iš programuotojo perspektyvos ir yra susijęs su programinės įrangos valdymu. Šiame rodinyje yra:

1. Komponentų diagrama,
2. Paketų diagrama.

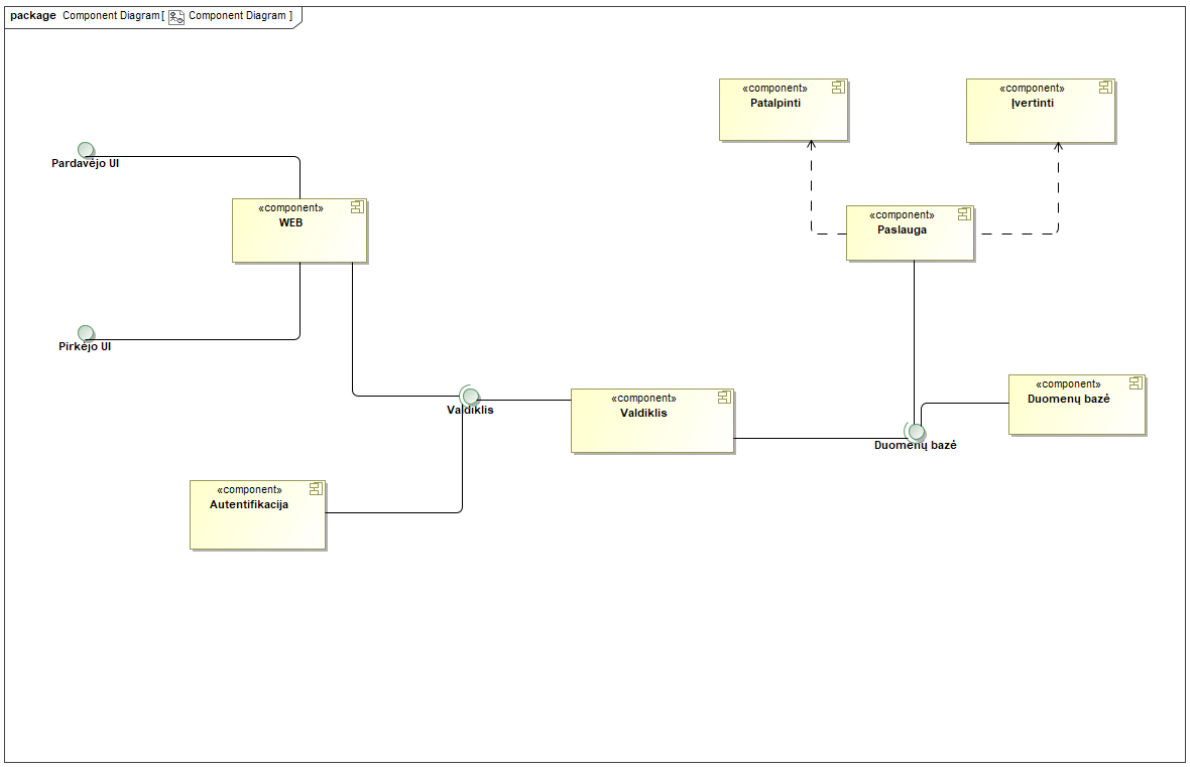
2.2.1 Komponentų diagrama

Šiame skyriuje parodysime sistemos įgyvendinimą ir komponentus. Bendras projekto komponentų diagramos vaizdas pateiktas 9 pav. Visi dalyviai naudoja skirtingas vartotojo sąsajas, tačiau visos šios vartotojo sąsajos yra prijungtos prie vienos sistemos. Rodoma sistema yra po pakeitimų įgyvendinimo. Pradinė diagrama būtų tokia pati, bet be administratoriaus vartotojo sąsajos ir administratoriaus kaip veikėjo.



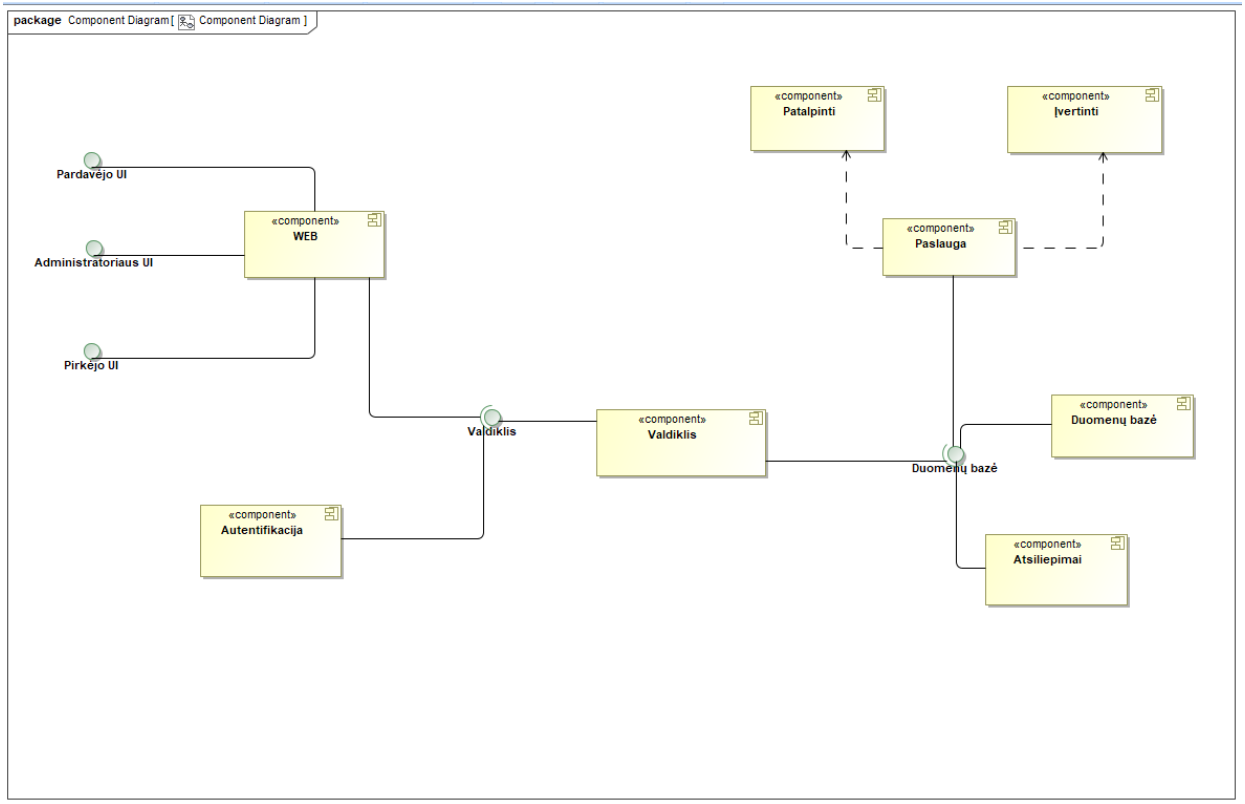
Pav. 9. Bendroji Komponentų diagrama

“FoodKept” sistemos komponentų išskaidymo diagramos pavaizduotos 10 - 11 pav. Diagrama iliustruota 10 pav. nurodo, iš kokių komponentų sistema buvo sudėta prieš mūsų modifikacijas. Pagrindinis komponentas yra valdiklis, kuris valdo visą šią sistemą.



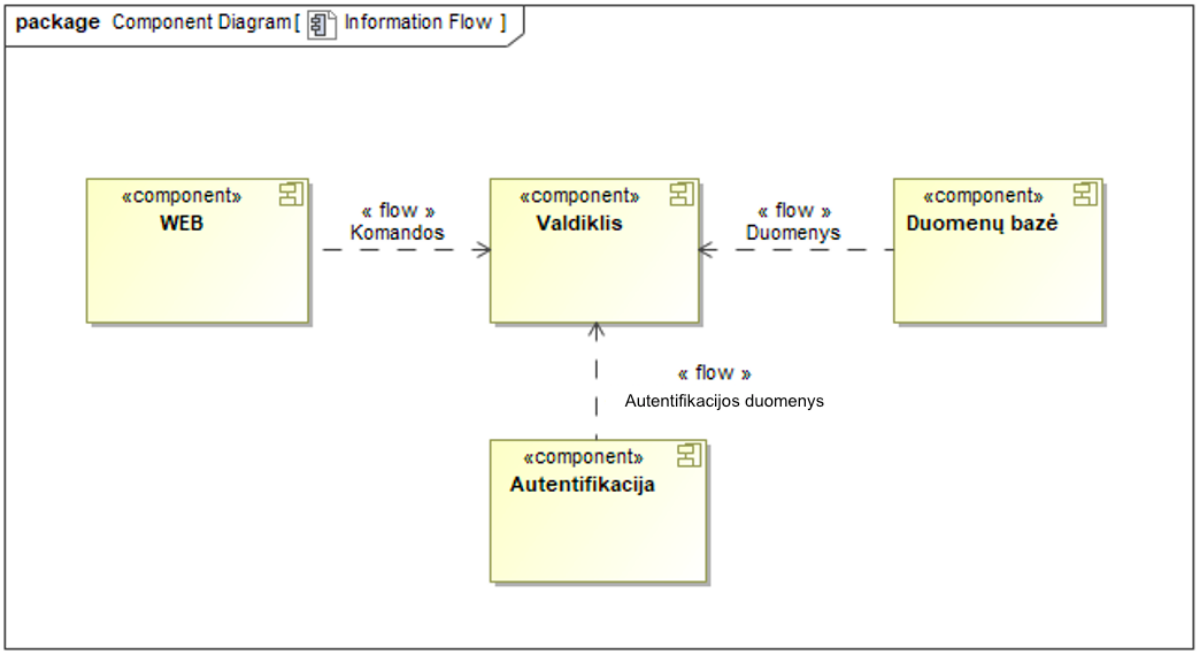
Pav. 10. Dekompozicijos diagrama prieš pakeitimus

11 pav. matome, kaip sistemos dekompozicija atrodo po mūsų modifikacijų. Pridėjome administratoriaus vaidmenį ir vartotojo sąsają. Kitas komponentas, kuris bus išsaugotas duomenų bazėje yra atsiliepimai.



Pav. 11. Dekompozicijos diagrama po pakeitimų

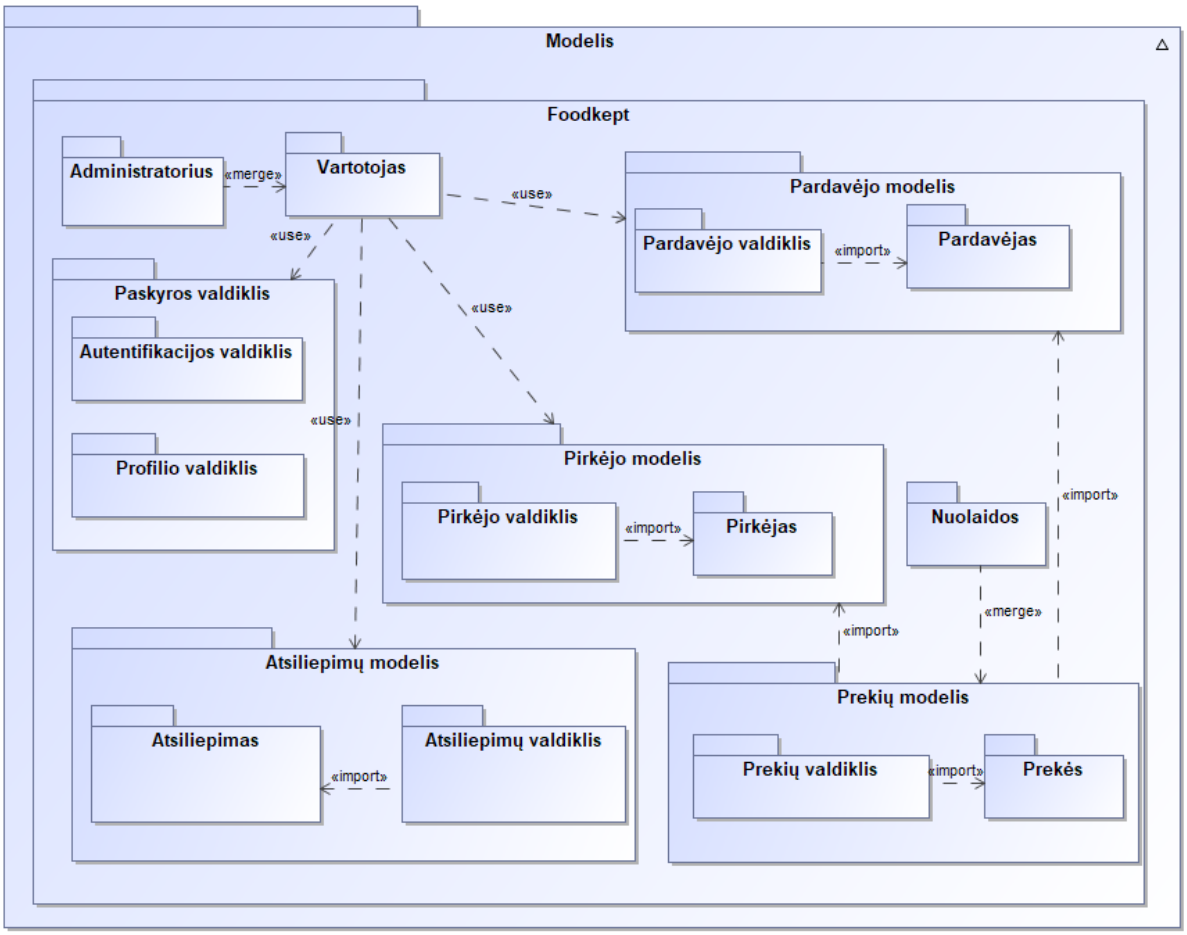
Informacijos srauto diagrama, parodyta 12 pav., demonstruoja kaip informacija perduodama iš šaltinių į gavėją.



Pav.12. Informacijos srauto diagrama

2.2.2 Paketų diagrama

Pav. 13., vaizduojama Paketų diagrama vaizduoja modelį sudarančių paketų tarpusavio priklausomybes.



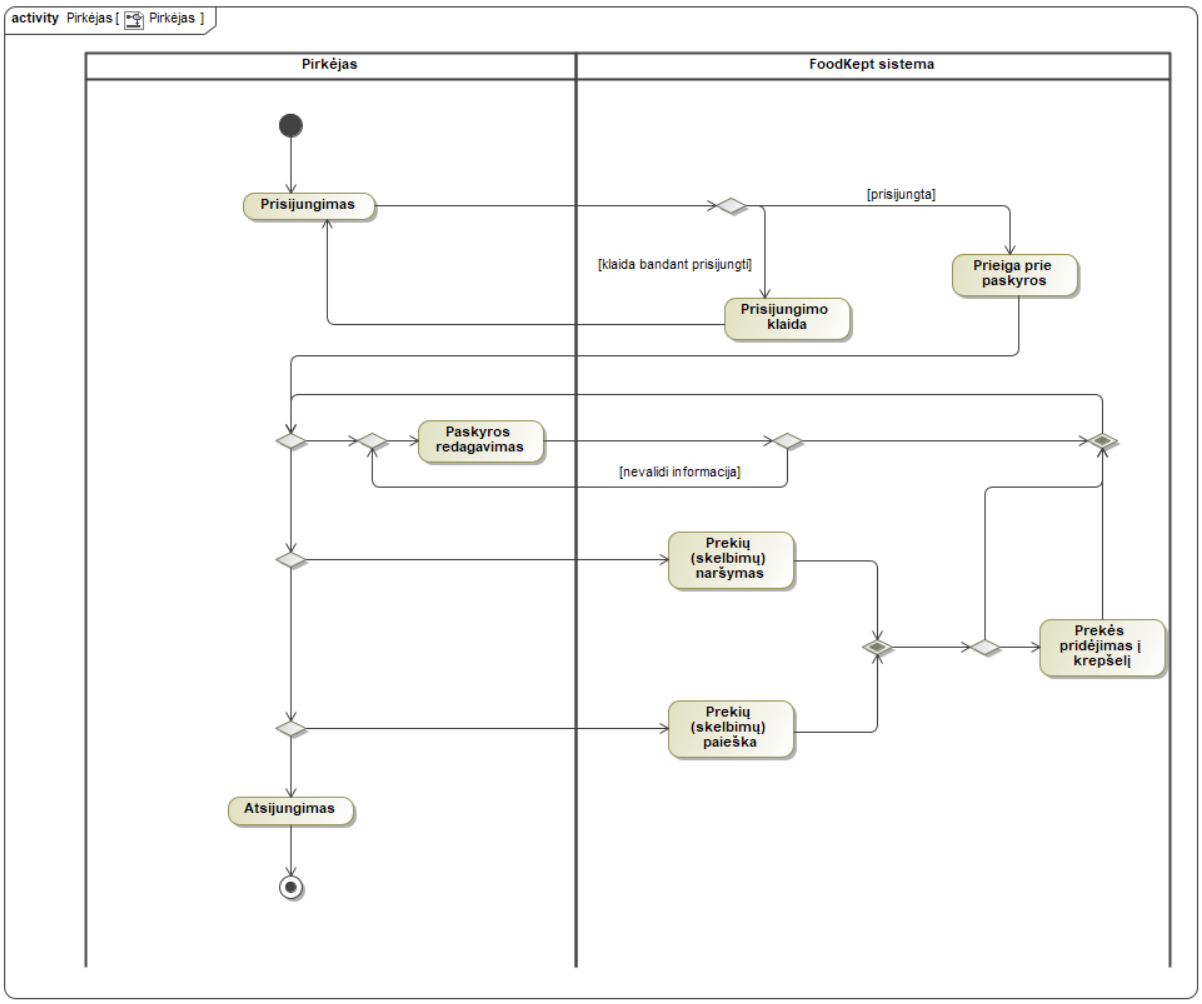
Pav.13. Paketų diagrama

2.3 Proceso analizė

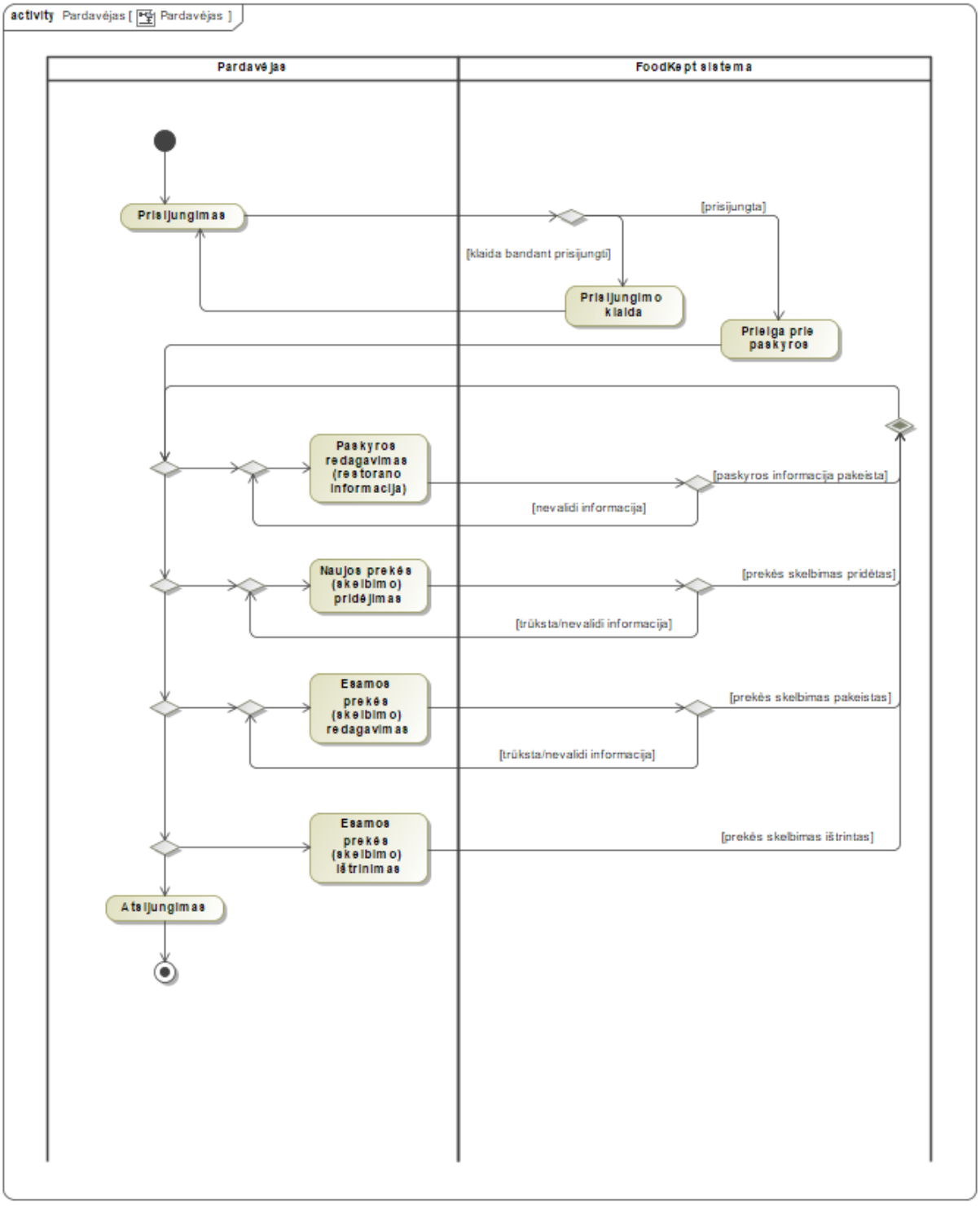
Proceso analizė iliustruoja ir paaiškina sistemos procesus. Pagrindinis dėmesys skiriamas jų veikimui ir sinchronizavimui. Pagrindinės diagramos, padedančios suprasti šiuos procesus, yra veiklos ir sekos diagramos.

2.3.1 Veiklos diagramos

Pav. 14 ir 15 vaizduojamos pardavėjo ir pirkėjo veiklos diagramos. Šios diagramos parodo visus procesus, kuriuos gali atlikti prisijungę vartotojai. Šiuo metu pardavėjas gali prisijungęs redaguoti savo paskyrą, pridėti, pakeisti ar ištrinti naują prekę (produktą), tuo tarpu pirkėjas gali po prisijungimo ieškoti arba naršyti po prekes, jas prisidėti į savo krepšelį, keisti savo paskyros informaciją ir atsijungti. Po sistemos pakeitimų veiklos diagramos išliks vienodos.



Pav. 14. Pirkėjo Veiklos diagrama



Pav. 15. Pardavėjo Veiklos diagrama

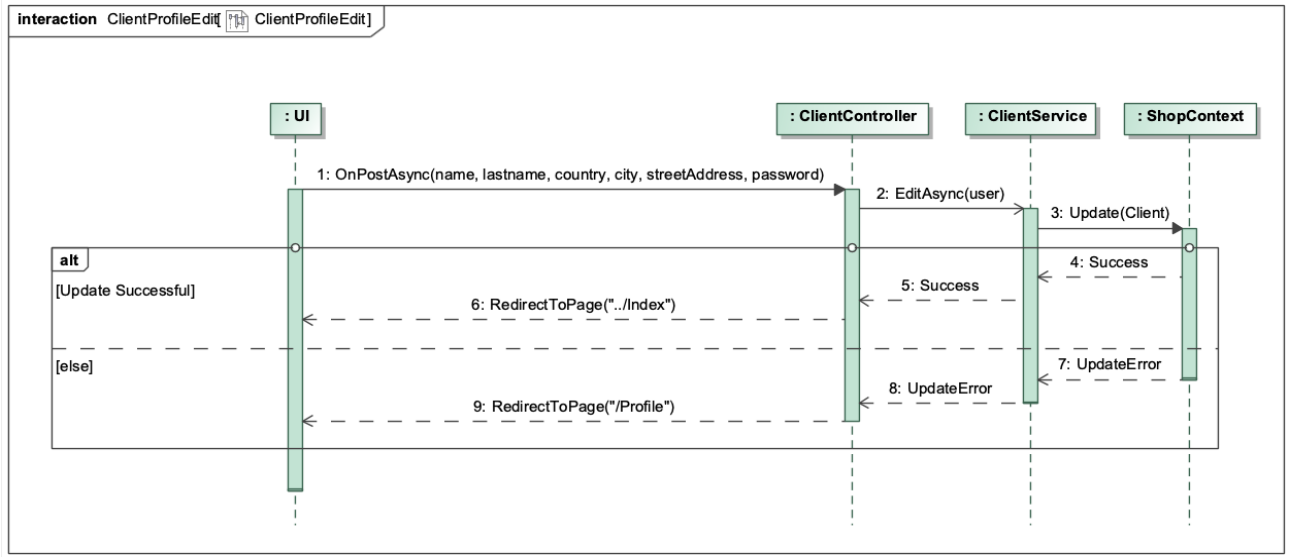
2.3.2 Sekų diagramos

Šiame skyriuje vaizduosime, kaip įgyvendinta panaudos atvejų tarpusavio komunikacija laiko atžvilgiu. Pateiktos diagramos vaizduoja veikimą po pakeitimų, pridėjus aplikacijų programavimo sąsają. Dėl aiškumo bei suderinamumo su kodu šios diagramos pateikiamos

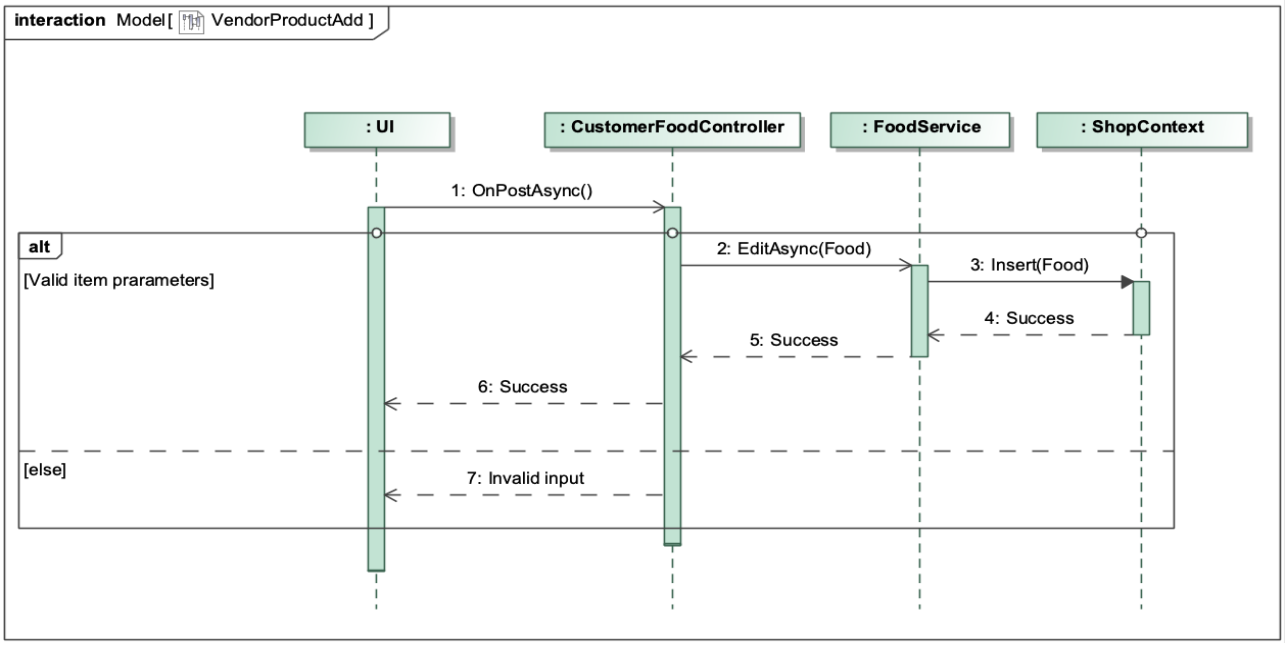
anglų kalba. Visų procesų veikimo pagrindas yra tas pat: iš naudotojo sąsajos valdymas perduodamas valdikliui, kviečiančiam atitinkamą servisą, kuriame formuojamas ryšys su duomenų baze. Visos vaizduojamos diagramos priklauso sistemai po pakeitimų, kadangi pradinės versijos diagramos skirtųsi tik valdiklio trūkumu.

2.3.2.1 Pardavėjo panaudos atvejų analizė

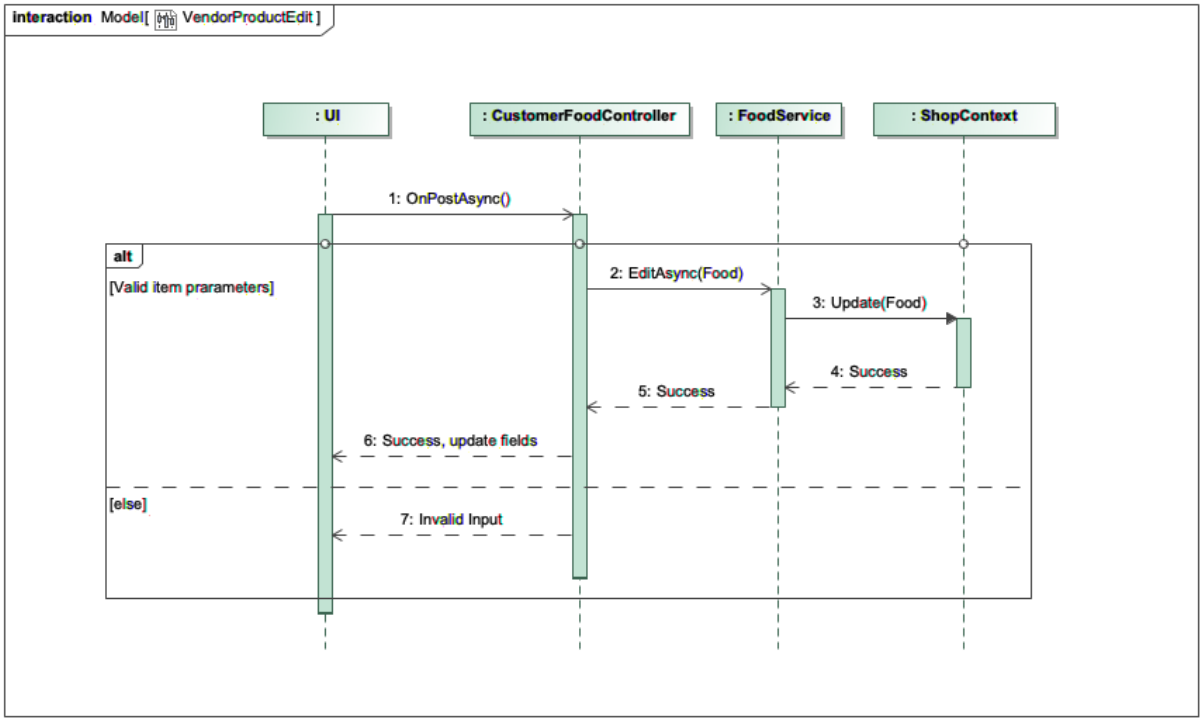
Pav. 16 diagrama vaizduoja pardavėjo paskyros redagavimą, o diagramos pažymėtos numeriais 17-19 galimybę pridėti, redaguoti arba ištrinti produktus, esančius pardavėjo paskyroje.



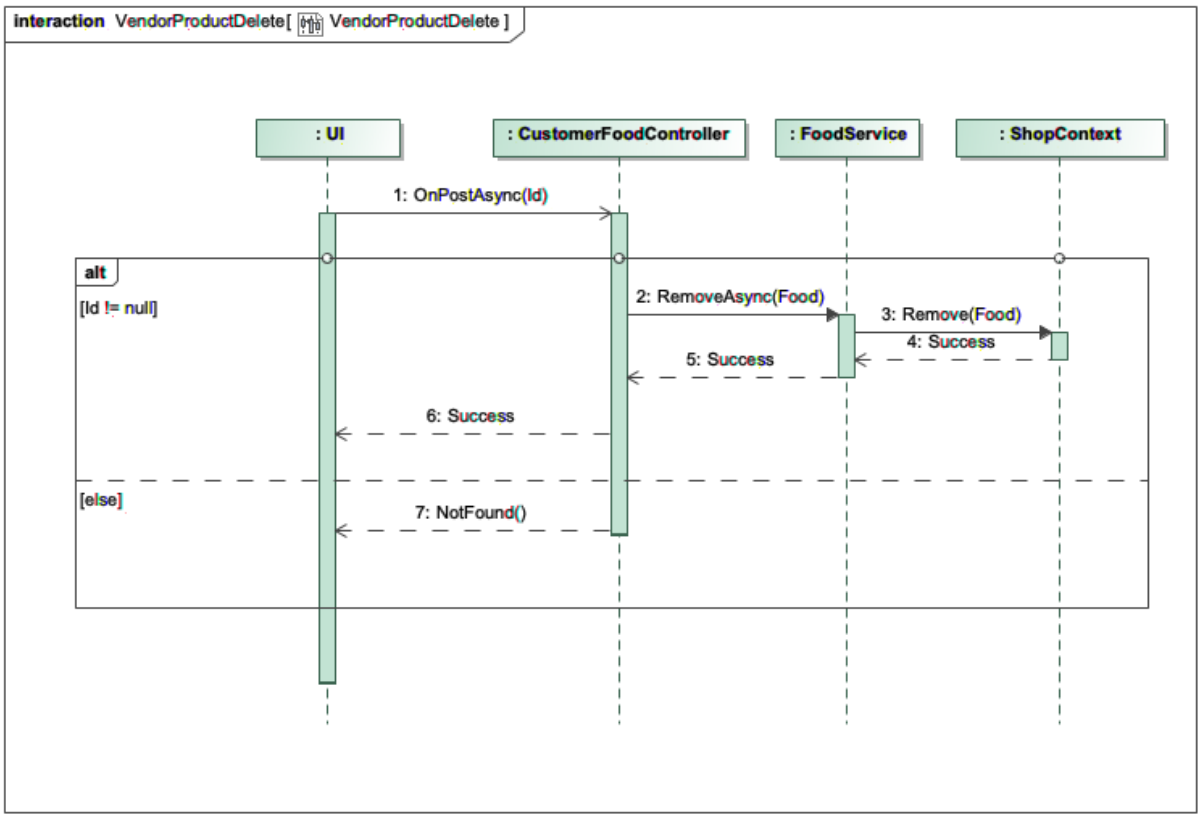
Pav. 16.. Pardavėjo paskyros redagavimas



Pav. 17. Produkto įkėlimas pardavėjo paskyroje



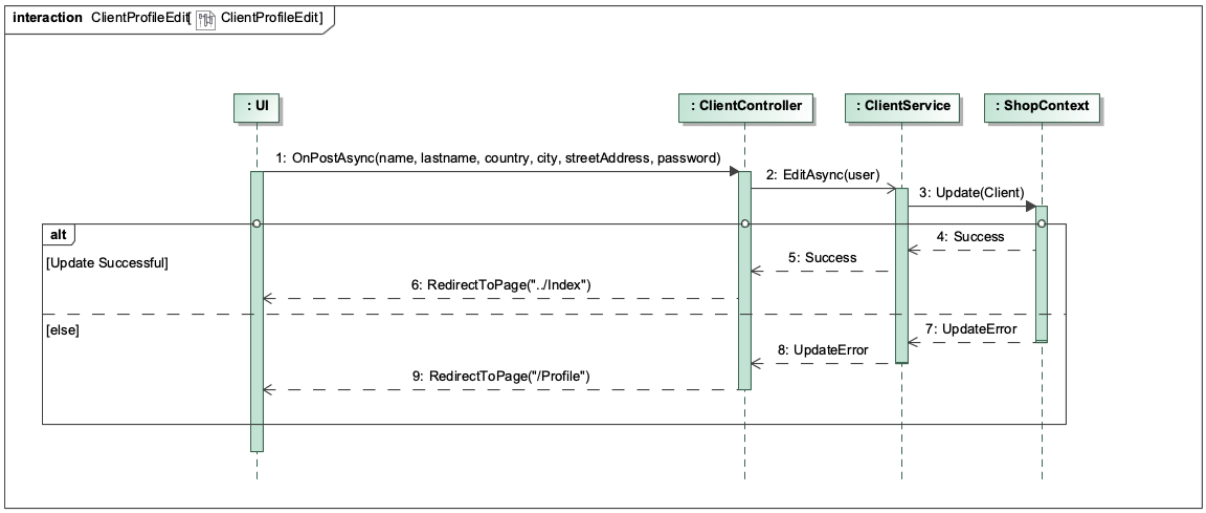
Pav. 18. Produkto redagavimas iš pardavėjo paskyros



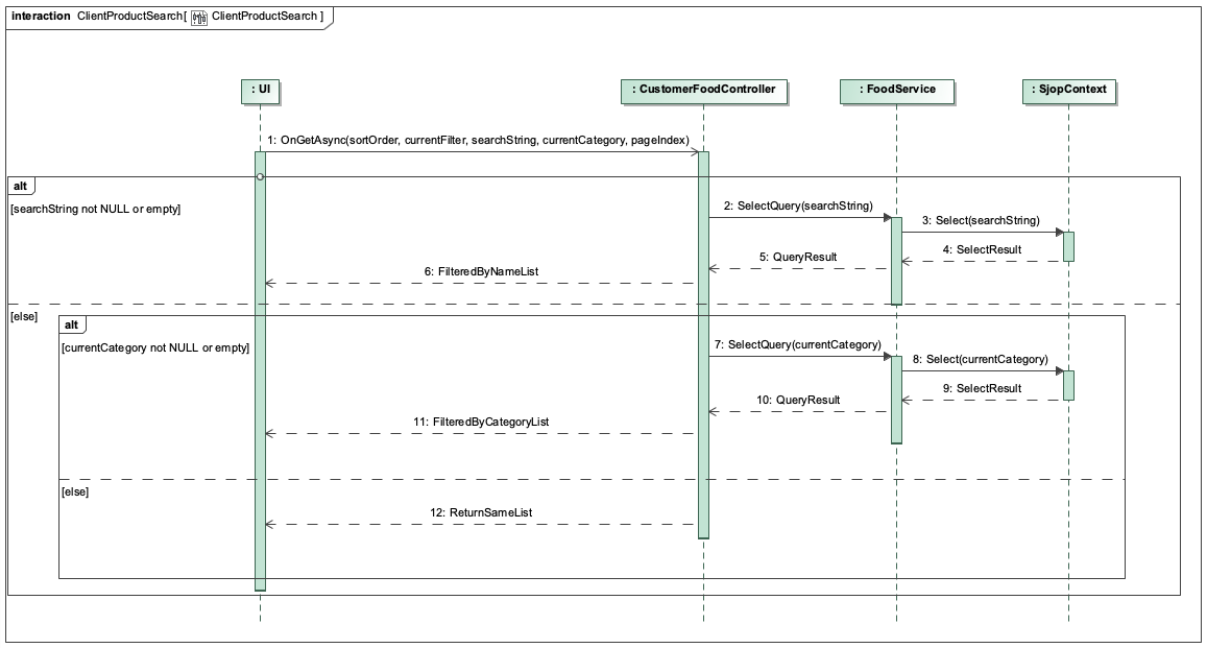
Pav. 19. Produkto trynimas iš pardavėjo paskyros

2.3.2.2 Pirkėjo panaudos atvejų analizė

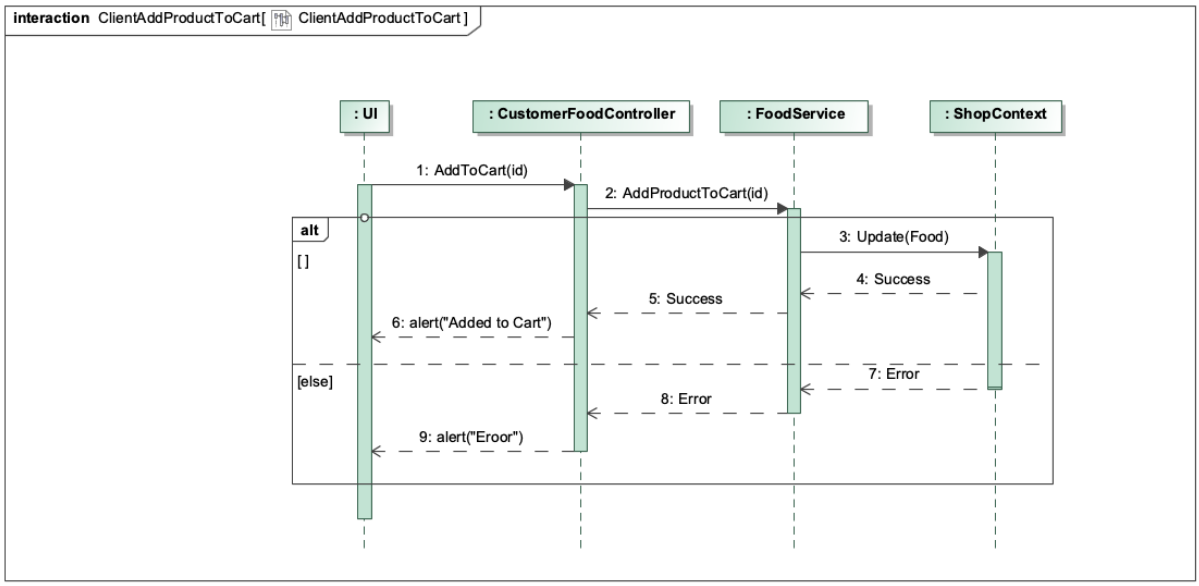
Diagramose su numeriais 20 – 23 vaizduojami pirkėjo panaudos atvejai: paskyros informacijos atnaujinimas, produktų paieška ir naršymas, produkto įkėlimas į pirkinių krepšelį bei jo įvertinimas.



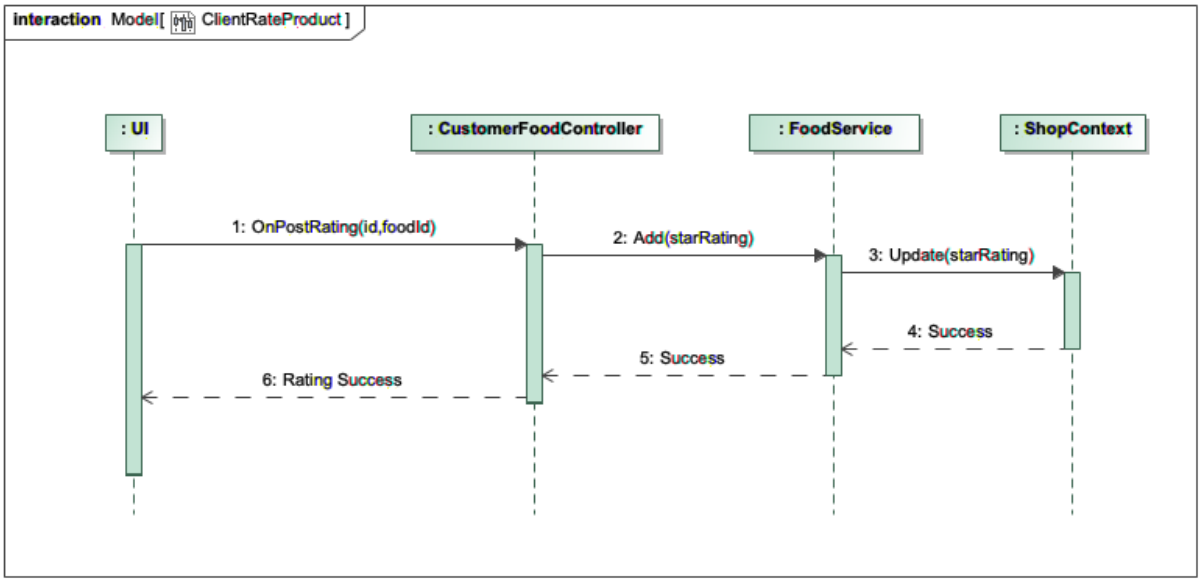
Pav. 20. Pirkėjo paskyros informacijos atnaujinimas



Pav. 21. Produktų paieška ir naršymas



Pav. 22. Produkto įkėlimas į pirkinių krepšelį

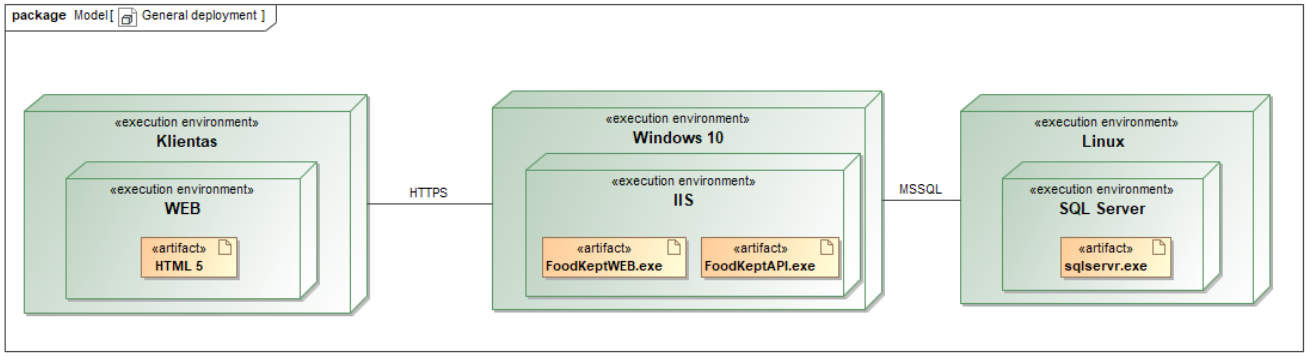


Pav. 23. Produkto įvertinimas

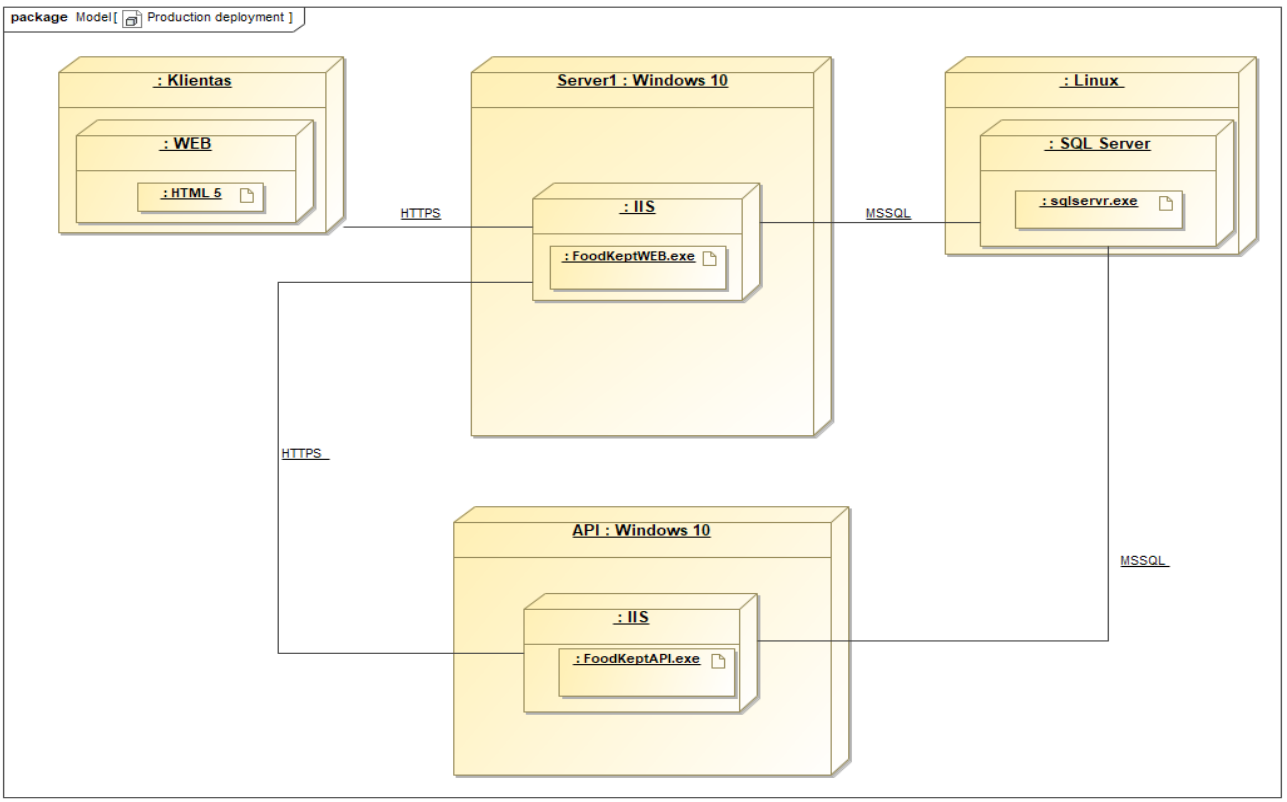
2.4 Techninė analizė

2.4.1 Diegimo diagramos

Šioje dalyje išanalizavome techninio lygmens programinės įrangos komponentų topologiją bei fizinius ryšius tarp šių komponentų. Sistemos vykdymo aplinkos parodytos 24 pav. Vartotojo įrenginio operacinė sistema nėra svarbi, nes visų tipų vartotojai prieis prie sistemos per naršyklę. Duomenų bazė nebūtinai turi būti įdiegta Linux vykdymo aplinkoje, tačiau tai rekomenduojama. Rekomenduojama, kad serverio valdiklių vykdymo aplinka būtų „Windows“ operacinė sistema, nes ji buvo sukurta naudojant ASP.NET sistemą.

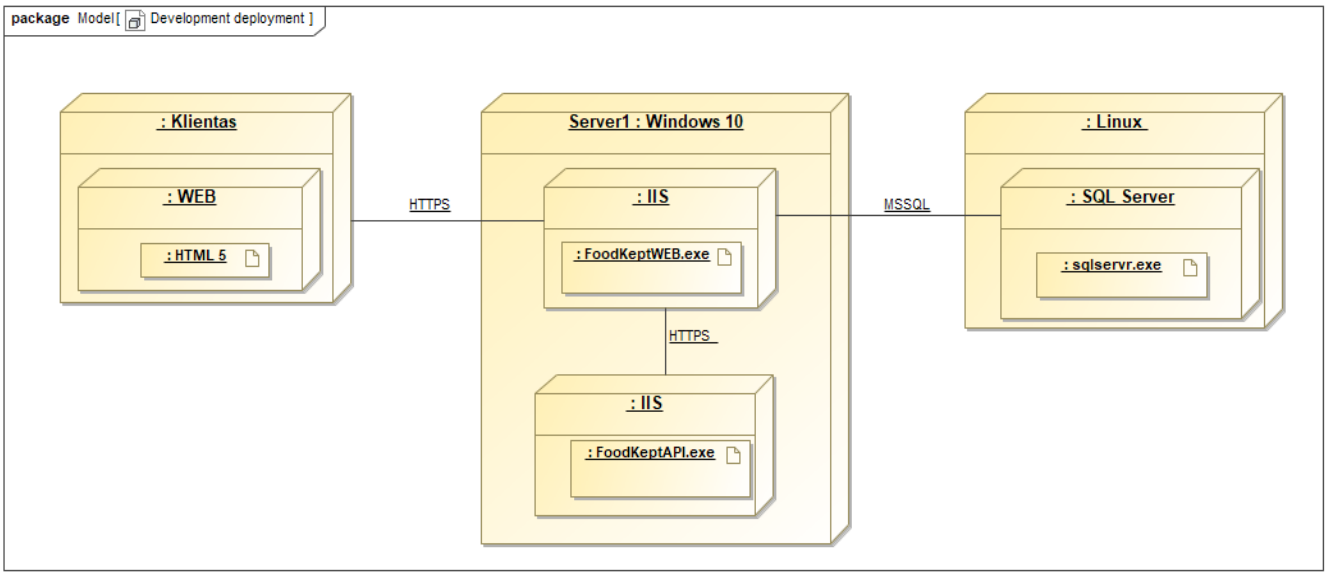


Pav.24. Pagrindinė diegimo diagrama



Pav. 25. Plėtros Aplinkos Diegimo diagrama

Kūrimo ir gamybos aplinkų skirtumai parodyti 25 ir 26 pav. Pagrindinis skirtumas yra tas, kad API turi būti talpinama atskirame serveryje.



Pav.26. Gamybos Aplinkos Diegimo diagrama

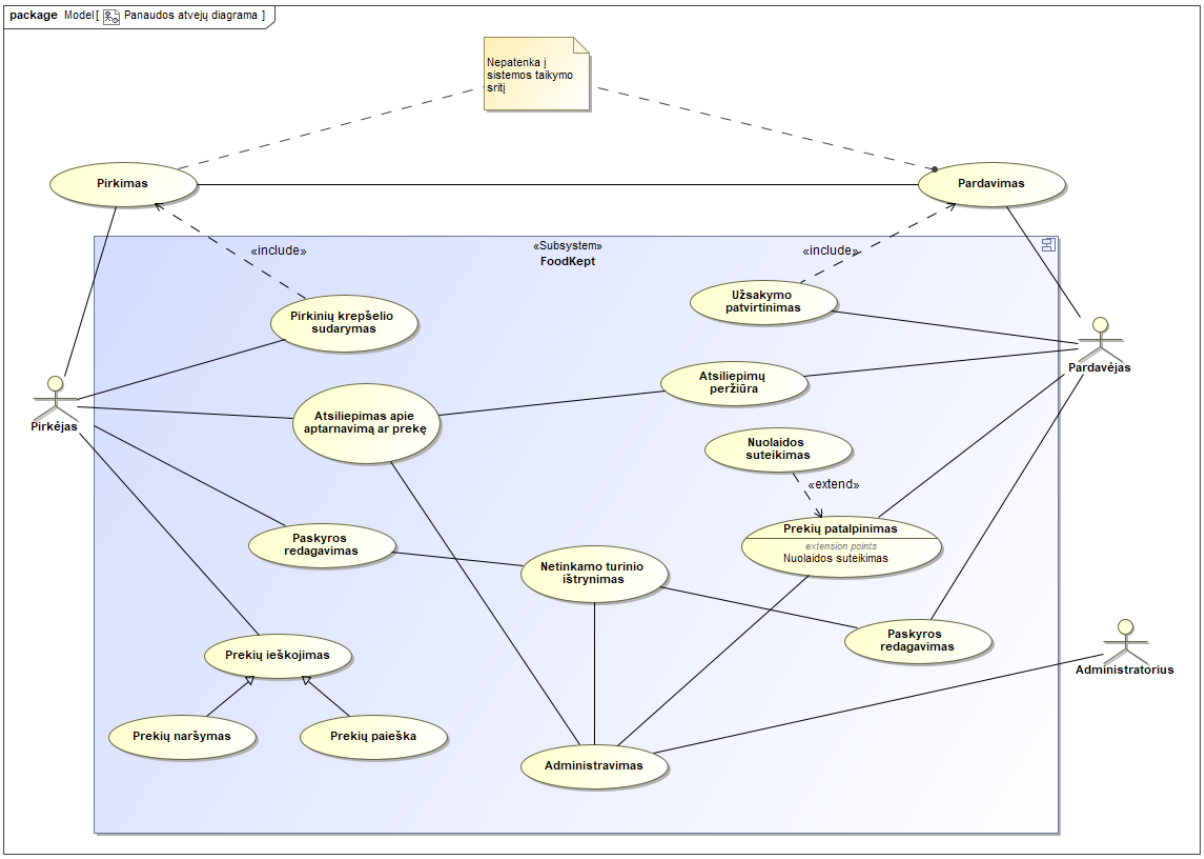
2.5Panaudos atvejų apžvalga

Šiame skyriuje daugiausia dėmesio bus skiriama mūsų architektūros modelio panaudos atvejų demonstravimui. Objektų, procesų ir veikėjų sąveikų sekoms iliustruoti pritaikėme panaudos atvejų diagramas. Turėdami šią informaciją galime patvirtinti architektūros projektą ir turėti atspirties tašką architektūros prototipo bandymams.

Pirmiausia sukūrėme panaudos atvejų diagramą, kurioje apžvelgiami bendrieji mūsų sistemos panaudojimo atvejai. Atsižvelgdami į tai, sukūrėme dar tris diagramas, kuriose išsamiau įsigilinta į kiekvienos iš pagrindinių vartotojų grupių panaudos atvejus.

2.5.1 Pagrindiniai Panaudos Atvejai

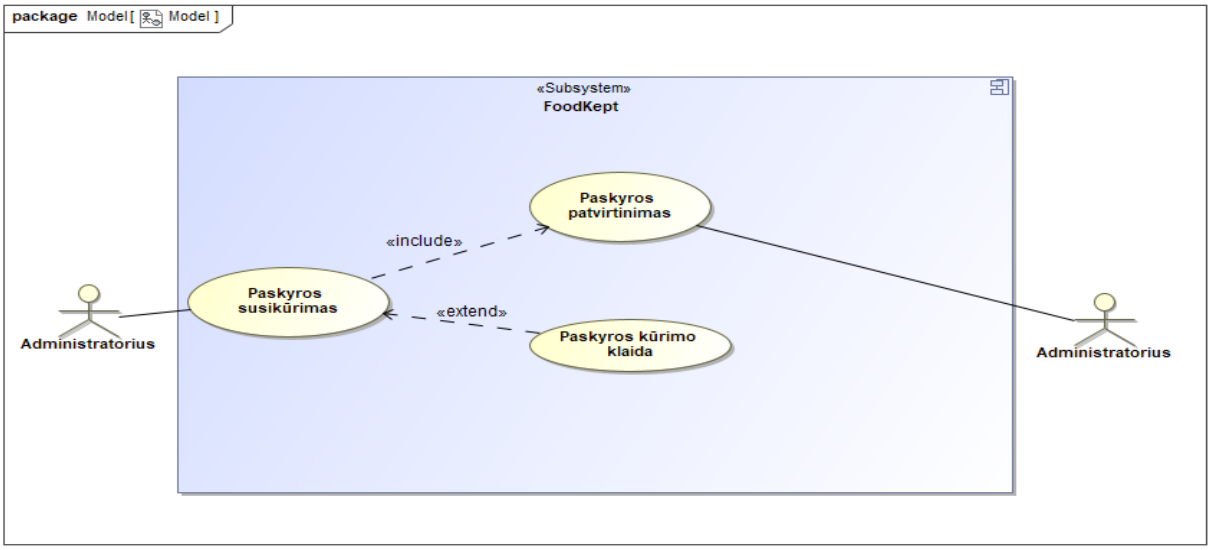
27 pav. vaizduojamoje pagrindinių panaudos atvejų diagramoje rodomi pagrindiniai mūsų įgyvendinti reikalavimai. Naujos prekės (produkto) pridėjimas ir sistemos naudojimo būdai visoms potencialių vartotojų grupėms. Svarbiausia, kad faktinis paslaugų pirkimas ir pardavimas nepatenka į mūsų sistemos taikymo sritį, nors tai yra pagrindinis visų platforma besinaudojančių vartotojų tikslas. Visas sąveikas galės stebėti administratorius.



Pav. 27. Pagrindinė Panaudos Atvejų diagrama

2.5.2 Neregistruoto naudotojo Panaudos Atvejai

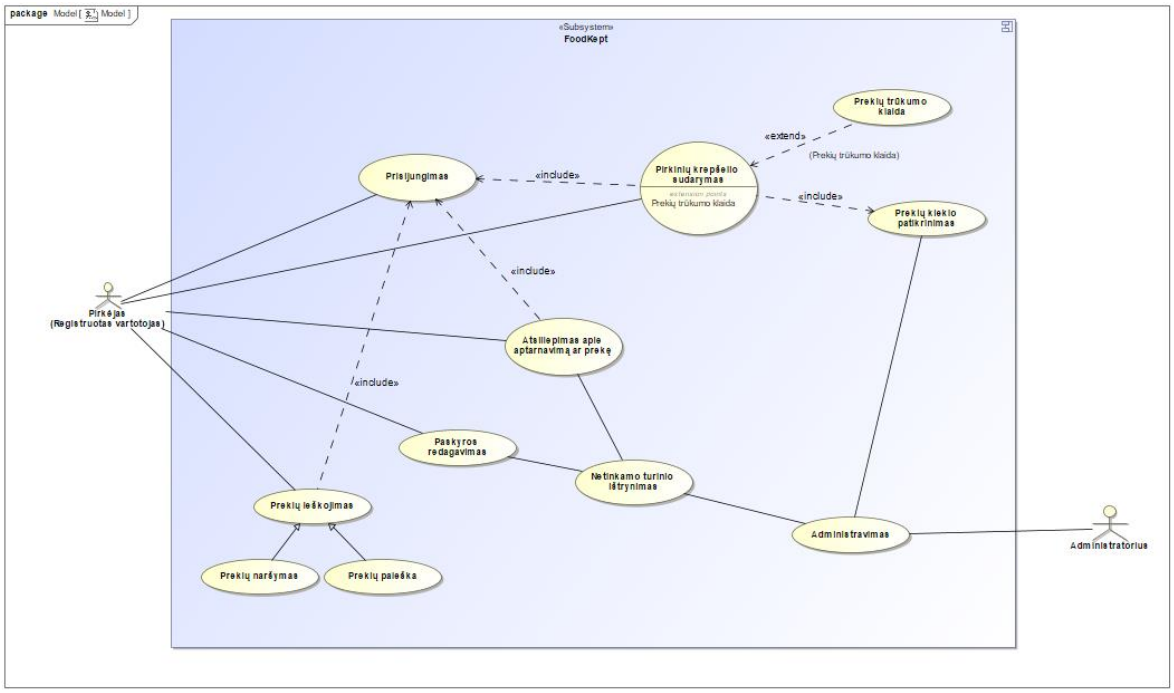
Kadangi neregistruoti vartotojai turi prieigą tik prie ribotų funkcijų, atitinkama neregistruoto vartotojo naudojimo atvejų diagrama, parodyta 28 paveiksle, taip pat yra paprasta. Neregistruotas vartotojas gali tik susikurti prie paskyros arba ją sukūręs prisijungti prie jos.



Pav. 28. Neregistruoto naudotojo Panaudos atveju diagrama

2.5.3 Pirkėjo (Registruoto Naudotojo) Panaudos Atvejai

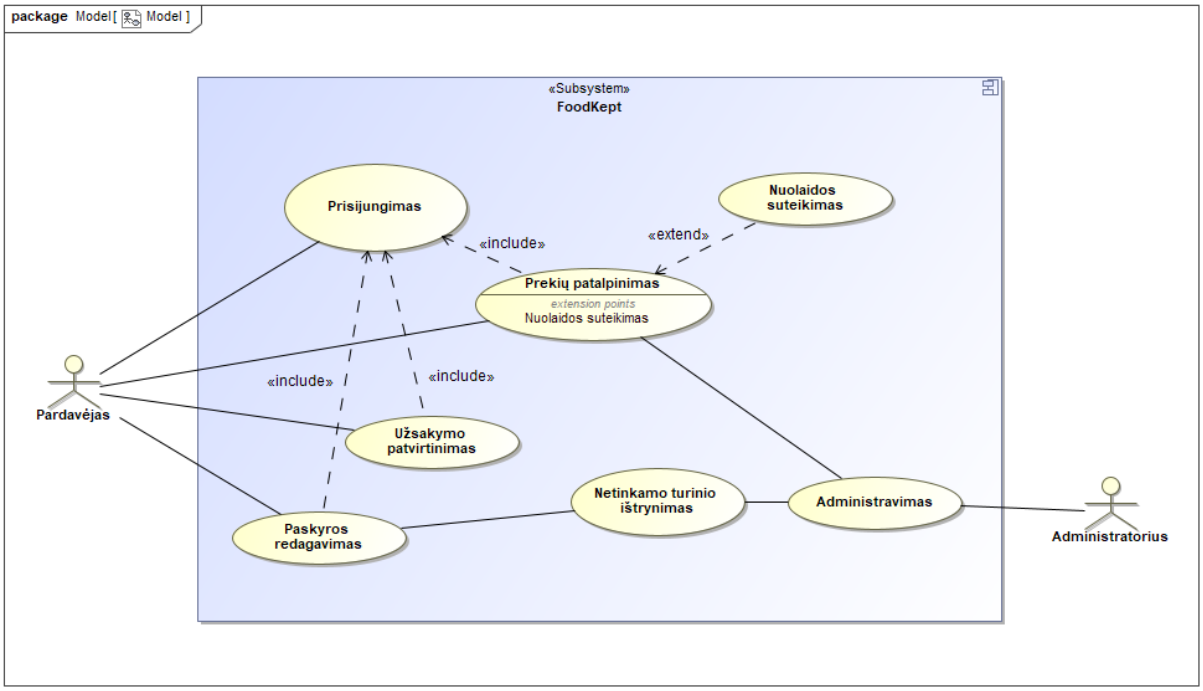
Pirkėjo panaudos atvejų diagrama, parodyta 29 pav., rodo, kad registruoti vartotojai turi prieigą prie daugiau funkcijų nei neregistruoti. Tačiau norint pasiekti šias funkcijas, reikalingas autentifikavimas. Atsižvelgdami į mūsų reikalavimus, mes įdiegsime lengvesnę prekių paiešką ir papildomus paieškos filtrus. Be to, administratorius galės stebėti registruotų vartotojų veiklą ir reaguoti į jų ataskaitas.



Pav. 29. Pirkėjo (Registruoto Naudotojo) Panaudos Atvejų diagrama

2.5.4 Pardavėjo Panaudos Atvejai

Pardavėjo panaudos atvejų diagrama, parodyta 30 pav., parodo, kaip pardavėjai gali naudoti mūsų sistemą. Kaip ir pirkėjo naudojimo atvejo diagramoje, pardavėjai reikalauja autentifikavimo, kad galėtų pasiekti visas sistemos funkcijas. Vėlgi, administratoriai galės reguliuoti visą pardavėjo veiklą.



Pav. 30. Pardavėjo Panaudos Atvejų diagrama

2.6 Modelio analizė reikalavimų atsekamumo lentelėje

Sukūrėme modelio analizė pagal reikalavimų atsekamumo lentelė, parodytą 1 lentelėje, kad padėtume mūsų komandai užtikrinti, kad visi reikalavimai būtų įtraukti į planuojamus architektūros pakeitimus

Lentelė 1 Modelio analizė reikalavimų atsekamumo lentelėje

Diagrama/ Modelis	Reikalavimai	Pardavėjai gali redaguoti įkeltas prekes	Pirkėjas gali parašyti atsiliepimą apie produktą	Administratoriaus gali trinti netinkamą turinį	Sukurta aplikacijų programavimo sąsają (API) pardavėjo prekėms	Padarytas CI/CD
Veiklos	Pirkėjas		X			
	Pardavėjas	X				
Klasių	Prieš pakeitimus		X			
	Po pakeitimų	X	X	X		
Komponentų	Bendroji	X	X	X	X	
	Bendroji dekompozicijos prieš pakeitimus					
	Bendroji dekompozicijos po pakeitimų	X	X	X	X	
	Informacijos srauto					
Plėtros	Prieš pakeitimus					
	Po pakeitimų					
Objektų	Objektų	X	X			
Ryšių	Prieš pakeitimus					
	Po pakeitimų				X	
Būsenų	Naudotojo	X	X			
	Produktų įkėlimas	X				
	Produktų atsiliepimas		X			
Diegimo	Pagrindinė					X
	Plėtros aplinkos					X
Panaudos atvejų	Pagrindinė	X	X	X		
	Neregistruoto pirkėjo					
	Registruoto pirkėjo		X	X		
	Pardavėjo	X		X		
Paketų	Dabar					
Sekų	Pardavėjo po	X			X	
	Pirkėjo po		X		X	