



VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMACINIŲ SISTEMŲ INŽINERIJOS STUDIJŲ PROGRAMA

Darbo vietos rezervavimas su Microsoft Copilot Studio

Workplace Booking with Microsoft Copilot Studio

Baigiamasis bakalauro darbas

Atliko: Ovidijus Vaštakas

VU el. p.: ovidijus.vastakas@mif.stud.vu.lt

Vadovas: dr. Laura Ringienė

Vilnius
2024

TURINYS

SANTRAUKA.....	3
SUMMARY	4
1. ĮVADAS.....	5
2. MICROSOFT POWER PLATFORM IR KITŲ MAŽAI KODO, JOKIO KODO PLATFORMŲ APŽVALGA	6
2.1. Power Platform	6
2.2. Mažai kodo, jokio kodo kūrimo platformos.....	7
2.3. Kiti mažai kodo, jokio kodo pokalbių robotų kūrimo įrankiai	10
2.4. Kodėl Microsoft Power Platform	13
2.5. Bot Framework Composer ir adaptyvios kortelės	14
3. SISTEMOS APRAŠYMAS	15
3.1. Microsoft Teams aplinka.....	15
3.2. Programos gyvavimo ciklo valdymas	15
3.3. Duomenų modelis.....	16
3.4. Microsoft Copilot Studio veikimas	17
3.5. Darbo vietos rezervavimo programėlės funkcionalumas	18
3.6. Patvirtinimo žingsnis rezervavimo procese	19
3.7. Darbo vietos rezervavimo procesas pokalbių robotu	20
4. DARBO VIETOS REZERVAVIMO SISTEMOS KŪRIMAS	22
4.1. Darbo aplinka	22
4.1.1. Darbo įrankiai	22
4.1.2. Aplinkos paruošimas	22
4.2. Duomenų bazės kūrimas.....	23
4.3. Programėlės kūrimas	25
4.3.1. Pagrindinis ekranas	26
4.3.2. Numatytojo ofiso pasirinkimo ekranas	26
4.3.3. Rezervacijos datos pasirinkimo ekranas	27
4.3.4. Aukšto arba teritorijos pasirinkimo ekranas	28
4.3.5. Naudotojo rezervacijų ekranas	29
4.3.6. Administratorių sąrašas	29
4.3.7. Visų rezervacijų apžvalga	30
4.4. Patvirtinimo žingsnis	32
4.5. Pokalbių roboto kūrimas.....	37
4.5.1. Pokalbio pradžios konfigūracija	37
4.5.2. Ofiso pasirinkimas	38
4.5.3. Datos pasirinkimas	41

4.5.4.	Aukšto arba teritorijos pasirinkimas	45
4.5.5.	Rezervacijos įrašo sukūrimas	47
4.6.	Darbo vietos rezervavimo programėlės ir pokalbių roboto palyginimas	49
5.	REZULTATAI IR APIBENDRINIMAS	51
	LITERATŪRA.....	52
	PRIEDAI.....	53

SANTRAUKA

Pagrindinis baigiamojo darbo tikslas – darbo vietos rezervavimo sistemai sukurti programėlę ir pokalbių robotą ir juos palyginti tarpusavyje. Darbo metu buvo sprendžiami šie uždaviniai: Išanalizuoti *Microsoft Copilot Studio* įrankį, jo galimybes, bei pritaikomumą darbo vietos rezervacijos sistemai. Palyginti *Microsoft Copilot Studio* su kitais įrankiais pokalbių robotų kūrimui. Sukurti programėlę ir pokalbių robotą darbo vietos rezervavimui. Palyginti darbo vietos rezervavimo procesą ir efektyvumą naudojantis programėle ir pokalbių robotu.

Raktiniai žodžiai: rezervavimas, automatizavimas, pokalbių, robotas.

SUMMARY

The main purpose of the final work – for a workplace booking system develop an app and a chatbot and compare them with each other. During the work, the following tasks were solved: analyze *Microsoft Copilot Studio* tool, its capabilities, and adaptability to the workplace reservations system. Compare *Microsoft Copilot Studio* with other tools for developing chatbots. Develop an app and a chatbot for the workplace reservation. Compare workplace reservation process and efficiency with the app and chatbot.

Keywords: reservation, automation, chat, copilot.

1. ĮVADAS

Apžvelgus įvairius darbo pasiūlymus matoma tendencija, jog dauguma įmonių siūlo hibridinį darbo modelį, tai yra, palieka darbuotojams galimybę rinktis darbą iš namų arba iš ofiso. Taigi visų pirma pati darbo vietos rezervavimo sistema padėtų įmonėms sekti darbuotojų skaičių kuris lankosi ofise ir taip optimizuoti darbo vietų skaičių sumažinant nuomos ar išlaikymo kaštus. Taip pat sumažinus darbo vietų skaičių padėtų darbuotojams užtikrinti jog atėję į ofisą jie ras vietą prie stalo. Verta paminėti, jog sistemą galima tuo pačiu naudoti ir konferencijų kambarių rezervavimui. Dar reiktų nepamiršti, jog visai neseniai Covid-19 pandemijos metu buvo įvairūs ribojimai žmonių kiekiui vienoje patalpoje. Taigi dar vienos tokios pandemijos atveju būtų galima riboti darbuotojų skaičių galintį ateiti į ofisą, taip pat užsikrėtimo atveju matyti kas tuo metu taip pat jame lankėsi. Bet visgi pagrindinė nauda šios sistemos darbo vietų skaičiaus optimizavimas ir nuomos ar išlaikymo kaštų optimizavimas. O darbo vietos rezervavimas su *Microsoft Copilot Studio*, tai yra, rašant užklausą pokalbių robotui palengvintų ir pagreitintų rezervavimo procesą. Kadangi sistema nėra susieta su patekimu į ofisą, tai reiškia, jog visi darbuotojai bet kada gali įeiti į ofisą. Tokiu atveju lengvesnis ir pagreitintas rezervavimo procesas padėtų lengviau išugdyti įprotį darbuotojams rezervuoti darbo vietą bent jau diena prieš ateinant į ofisą.

Darbo tikslas – darbo vietos rezervavimo sistemai sukurti programėlę ir pokalbių robotą ir juos palyginti tarpusavyje. Sprendžiami uždaviniai:

1. Išanalizuoti *Microsoft Copilot Studio* įrankį, jo galimybes, bei pritaikomumą darbo vietos rezervacijos sistemai. Palyginti *Microsoft Copilot Studio* su kitais įrankiais pokalbių robotų kūrimui.
2. Sukurti programėlę ir pokalbių robotą darbo vietos rezervavimui.
3. Palyginti darbo vietos rezervavimo procesą ir efektyvumą naudojantis programėle ir pokalbių robotu.

Berašant šį darbą *Microsoft* atnaujino *Power Virtual Agents* įrankį ir pervardino jį į *Microsoft Copilot Studio*, dokumentacija [1]. Taigi *Power Virtual Agents* 2023 metų gruodžio mėnesį tapo *Microsoft Copilot Studio*.

Baigiamąjį darbą sudaro šešios dalys: įvadas, analitinė, metodinė, ir įgyvendinimo dalys, bei rezultatai ir apibendrinimas, literatūros sąrašas.

2. MICROSOFT POWER PLATFORM IR KITŲ MAŽAI KODO, JOKIO KODO PLATFORMŲ APŽVALGA

Šiame skyriuje bus aptarta kas yra *Power Platform*, jos įrankiai, bei kam jie skirti. Taip pat papasakota apie mažai kodo, jokio kodo kūrimo platformas, pokalbių robotų kūrimo įrankius, kam naudojamas *Bot Framework Composer* ir adaptyvios kortelės.

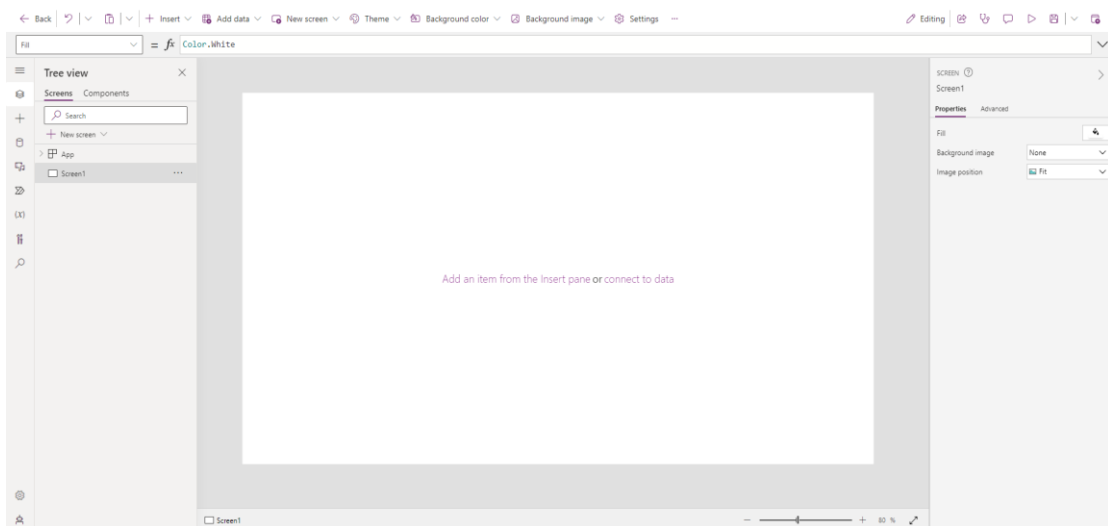
2.1. Power Platform

Darbo vietos rezervavimo sistema sukurta *Microsoft Power Platform* pagalba ir yra integruota į *Microsoft Teams*.

Microsoft Power Platform, tai platforma, kurios pagalba galima kurti įvairius sprendimus įmonių procesams skaitmenizuoti, automatizuoti, bei vizualizuoti. Tai yra programinė įranga kaip paslauga (angl. *Software as a Service*), tai reiškia, jog norint ja pasinaudoti mums reikės aktyvios *Microsoft* paskyros turinčios *Microsoft 365* arba *Office 365* licenciją.

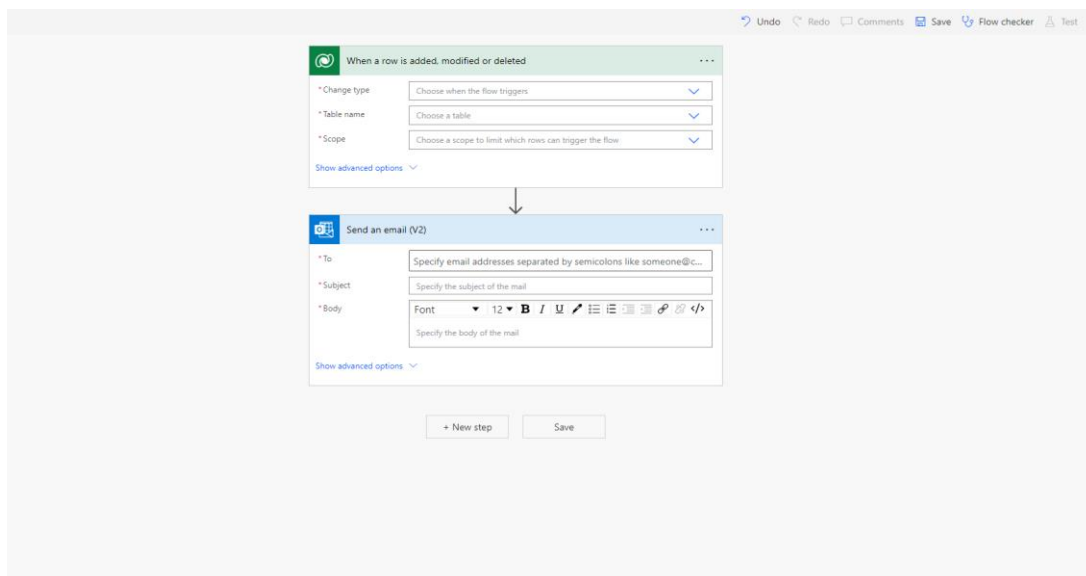
Power Platform įrankiai:

- Power BI – programinė įranga duomenų vizualizavimui.
- Power Apps – programinė įranga, programėlių kūrimui. Power Apps kūrimo aplinka pateikta 2.1 paveiksle.



2.1 pav. Power Apps programėlės kūrimo aplinka

- Power Automate – įrankis įvairių verslo procesų automatizavimui, priminimų siuntimui ir pan. Power Automate kūrimo aplinka pateikta 2.2 paveiksle.



2.2 pav. *Power Automate* automatizavimo kūrimo aplinka

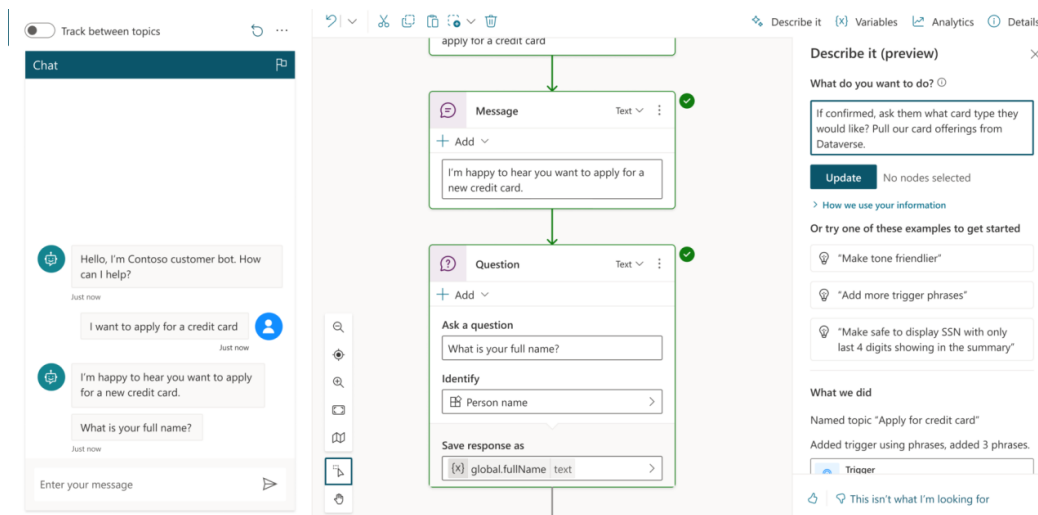
- *Power Automate Desktop* – robotų procesų automatizavimo (RPA) programinė įranga.
- *Microsoft Copilot Studio* – programinė įranga, pokalbių robotų kūrimui (angl. *Chatbots*).
- *Power Pages* – įrankis interneto svetainėms kurti, bei talpinti.

Šiame bakalauriniame darbe yra aprašoma darbo vietos rezervavimo sistemos kūrimo eiga, sistemos kūrimui naudoti trys aukščiau paminėti įrankiai: *Power Apps* programėlei, *Power Automate* rezervacijos patvirtinimui ir *Microsoft Copilot Studio* pokalbių roboto kūrimui.

Verta paminėti, jog *Microsoft Power Platform* išleista tik 2018 m. tai yra vis dar ganėtinai nauja, besiplečianti ir palaikoma technologija. Iki tol *Microsoft* nebuvo sukūrusi jokio panašaus produkto.

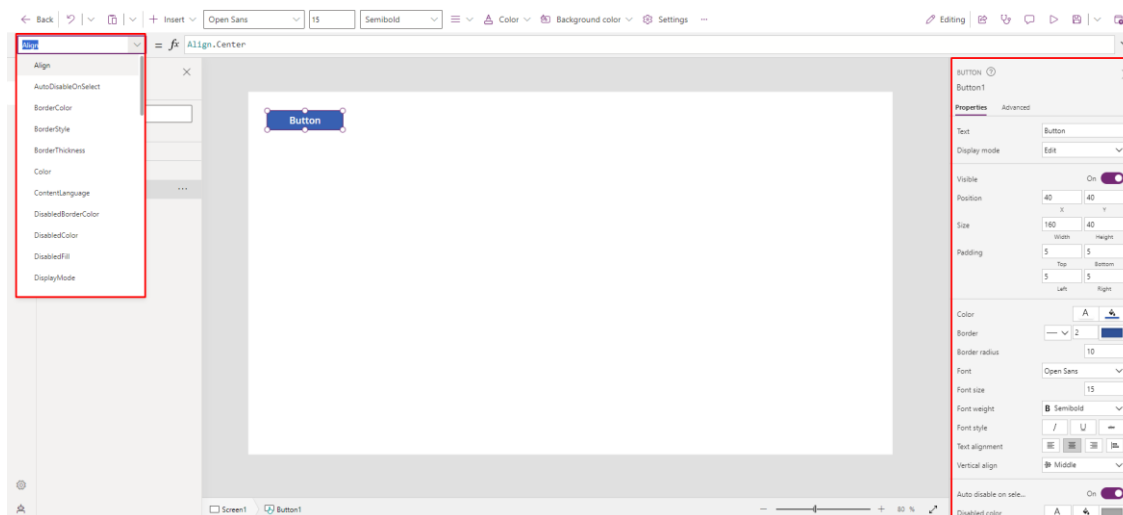
2.2. Mažai kodo, jokio kodo kūrimo platformos

Visi *Power Platform* įrankiai yra mažai kodo arba jokio kodo (angl. *low-code, no-code*). Tai reiškia, jog kurti programėles, pokalbių robotus ir automatizuoti procesus galima naudojant grafines vartotojo sąsajas ir konfigūracijas, o ne rašyti ilgą programinį kodą. Pavyzdžiui, kuriant pokalbių robotą viskas dedama lyg į grandinę (žr. 2.3 pav.). Išsiunčiama žinutė naudotojui, tuomet išsiunčiamas klausimas po kurio laukiama naudotojo įvesties. Įvestis išsaugoma į kintamąjį ir panašiai. Viskas konstruojama tiesiog pasirenkant įvairius veiksmus ir užpildant laukelius.



2.3 pav. Microsoft Copilot Studio pokalbių roboto kūrimo aplinka.

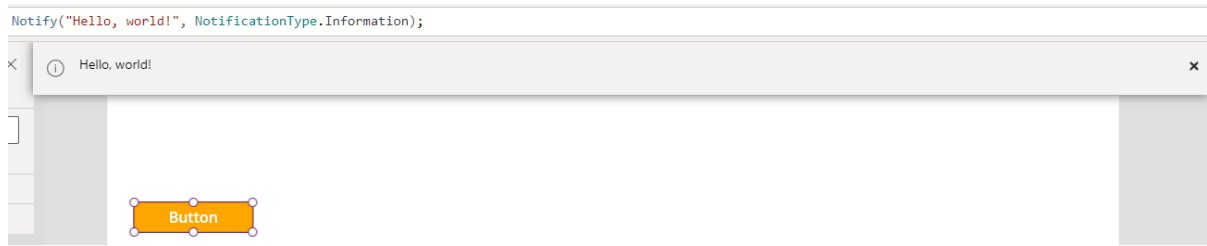
O pavyzdžiui kuriant programėlę galima, iš vartotojo sąsajos elementų sąrašo pasirinkti valdiklį – mygtuką, nutempti į kurią tik norima ekrano vietą. Galima keisti įvairias mygtuko savybes, tokias kaip, spalvą, apvadą ir pan. Iš esmės, programinės įrangos kūrėjai turi galimybę, jį konfigūruoti kaip tik nori grafines vartotojo sąsajos pagalbą. Žemiau pateiktame paveikslėlyje (žr. 2.4 pav.) kairėje pusėje pažymėtas laukas yra pasirinkimas visų mygtuko savybių kurias tuomet galima redaguoti šalia esančiame formulės įvedimo lange. Dešinėje pusėje, pažymėtame lange, yra tos pačios savybės, tačiau lengva prieiga jas taip pat įmanoma redaguoti ir čia, pavyzdžiui pasirinkti spalvą interaktyviai. Jei tai būtų norima padaryti rašant kodą – formulę, reikėtų kairėje pasirinkti savybę – Užpildas (angl. *Fill*) ir rašyti – Spalva.Raudona (angl. *Color.Red*).



2.4 pav. Power Apps kūrimo aplinka, mygtuko nustatymai

Taip pat egzistuoja mygtuko savybė – Pasirinkus (angl. *OnSelect*), nusakanti kas įvyks paspaudus mygtuką. Visos savybės aprašomos *Power Fx* programavimo kalba. Ši kalba yra grįsta formulėmis. Jos rašymas yra labai panašus į *Microsoft Excel* formulių rašymą. Tai yra labai glausta, paprasta ir aiškiai suprantama kalba. *Power Fx* apžvalga, taisyklės, pavyzdžiai, bei komandų sąrašas, [2]. Pavyzdžiui 2.5 paveiksle mygtuko pasirinkus (angl. *OnSelect*) savybėje parašyta funkcija – informuoti (angl. *Notify*), iškviečia informacinį lauką ir išveda norimą tekstą jame. Norint vartotojui atidaryti kitą langą galime naudoti funkciją Naršyti (angl. *Navigate*), joje

pateikus norimo atidaryti lango pavadinimą. Power Apps kūrimo aplinkoje langas vadinamas ekranu (angl. *Screen*).



2.5 pav. *Power Fx* kodo pavyzdys.

Taip pat mažai kodo, jokio kodo kūrimo platformos dažniausiai turi jau sukurtus sprendimus ar sprendimų šablonus įvairiems verslo procesams ar atvejams skaitmenizuoti ar automatizuoti. Naudojantis *Power Apps*, programėlę galima sukurti įkeliant *Excel* failą su pavyzdiniais duomenų įrašais arba kūrimo platformoje įrašant užklausą nurodant programėlės paskirtį. Tokiu būdu *Power Apps* pagal duomenų įrašus arba užklausą sukuria programėlę ir sukonfigūruoja ekranus. Taip kūrimo pradžia yra palengvinama, paspartinama. Todėl *Microsoft Power Platform* ir *Microsoft Copilot Studio* arba kitų mažai kodo, jokio kodo platformų ir įrankių pagalba pokalbių robotą ar programėlę galima sukurti ženkliai greičiau, pigiau ir išnaudojant mažiau resursų, nei tradiciniu programavimu. Mažai kodo, jokio kodo kūrimo privalumai prieš tradicinį programavimą [3]:

- Kūrimo sparta.
- Nereikalauja didelės programuotojų, specialistų komandos.
- Mažesnė kūrimo kaina.
- Geresnis rizikos valdymas, tai yra, galima greitai atlikti pakeitimus, kadangi nereikalingi konfigūravimo žingsniai, tokie kaip: kodo peržiūra, kompiliavimas, derinimas.
- Geresnis produktyvumas. Mažai kodo, jokio kodo kūrimo platformos sumažina atotrūkį tarp informacinių technologijų specialistų ir verslo atstovų. Todėl verslo atstovai gali patys kurti įvairius sprendimus.
- Geresnis saugumas. Atsiradus pataisymams ar saugumo atnaujinimams, jie įdiegiami automatiškai mažai kodo, jokio kodo kūrimo platformoje, todėl kūrėjams nereikia to daryti paties, atnaujinti kodo ar panašiai.

Mažai kodo, jokio kodo kūrimo platformų trūkumai:

- Naudojant mažai kodo, jokio kodo kūrimo platformas, atsiranda priklausomybė nuo pačios platformos ar paslaugos tiekėjo.
- Ribotas pritaikymas.
- Ribota integracija. Mažai kodo, jokio kodo kūrimo platformos turi ribotą integraciją, suderinamumą. Tai gali būti problema įmonėms naudojančioms kitas išorinės sistemas. Pavyzdžiui gali būti neįmanoma susieti kuriamą sistemą su išorine duomenų baze.
- Programuotojų trūkumas. Mažai kodo, jokio kodo kūrimas kol kas nėra stipriai paplitęs ir populiarus, taip pat skirtingų platformų kūrimo ypatybė skiriasi, todėl gali atsirasti iššūkių įmonėms rasti mažai kodo, jokio kodo programuotojų.

Bandant atrasti kūrimo spartos palyginimų naudojant mažai kodo platformas ir tradicinį programavimą, rasta ataskaita apie mažai kodo platformas – [4]. Joje atlikti tyrimai apie mažai kodo platformas, apklausus šios paslaugos tiekėjus, bei jų klientus. Ataskaitos pirmame

paveikslėlyje (žr. 2.6 pav.) galima matyti palyginimą kiek užtruko įvairių sistemų kūrimas tradiciniu kūrimo būdu (angl. pro-code), bei mažai kodo platformų pagalba. Pavyzdžiui sukurti sistemą skambučių centro operatoriui tradiciniu programavimu užtruko keturis mėnesius, o pasinaudojus mažai kodo platforma (naudota platforma nepaminėta), tik tris savaites.

Enterprise	Result	Code	Low-code
US government (Affordable Care Act)	Document compliance module	100 person-months	5 person-months
British insurance provider	Agent portal	Unknown*	10 days to minimum viable product (MVP)
Call center operator	Customer-specific app	4 months	3 weeks
Spanish insurance provider	Web channel and administration system	2.7 years (estimated)	13 weeks

2.6 pav. Kodo kūrimo greičio palyginimai tarp tradicinio programavimo ir mažai kodo platformų, [4].

2.3. Kiti mažai kodo, jokio kodo pokalbių robotų kūrimo įrankiai

Panaršius internete galima rasti ne vieną mažai kodo, jokio kodo įrankį: *AppSheet*, *OutSystems*, *Landbot.io* ir kiti. Pavyzdžiui, *AppSheet* gali pasiūlyti tik *Power Apps* ir dalinai *Power Automate* funkcionalumą. *Landbot.io* tik *Microsoft Copilot Studio* funkcionalumą. Nėra tokio produkto kuris pasiūlytų viską ką turi *Power Platform*, tai yra, programėles, svetaines, pokalbių robotus, robotų procesų automatizavimą, duomenų vizualizavimą. Taip pat verta paminėti, jog visi *Microsoft Power Platform* produktai yra suderinami tarpusavyje, pavyzdžiui: į *Power Apps* programėles galima integruoti *Power BI* įrankiu kurtas duomenų ataskaitas (angl. *Reports*) arba *Microsoft Copilot Studio* kurtus pokalbių robotus ir dar daugiau. *Google* paieškoje figūruoja šie įrankiai:

- Landbot.io
- Botpress
- AppSheet
- OutSystems
- Mendix

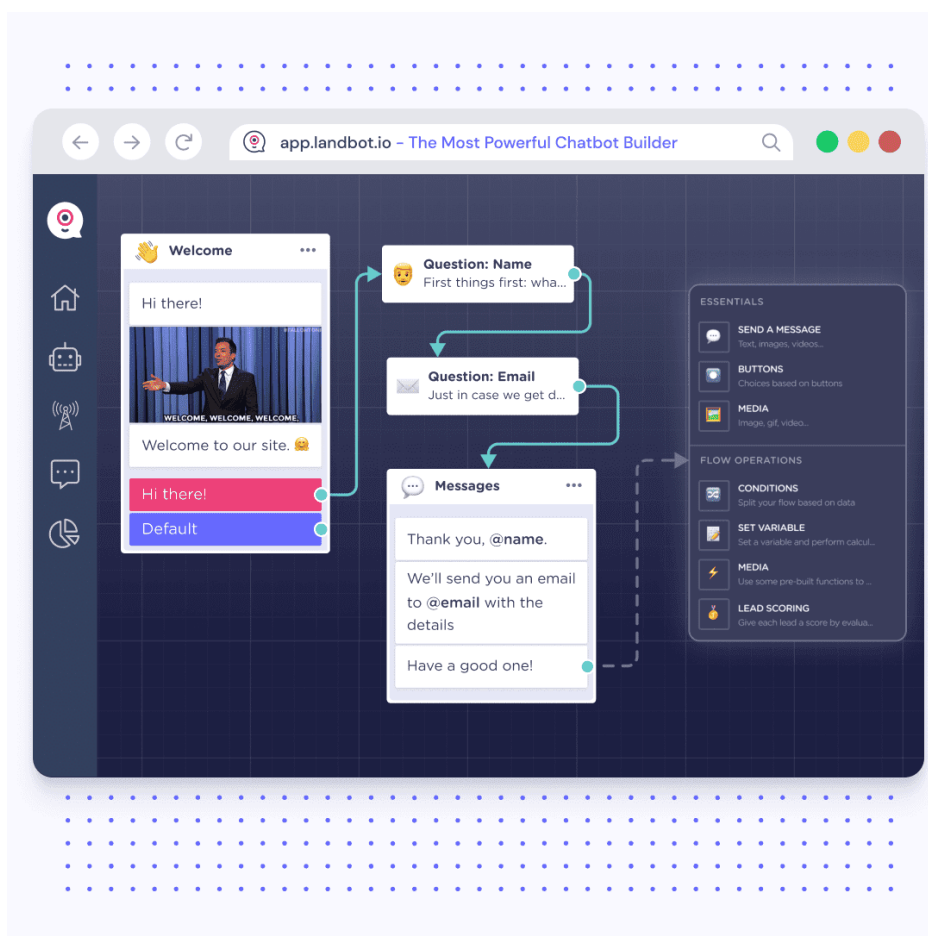
Aukščiau paminėti tik keli įrankiai, žinoma jų yra daug daugiau, tačiau šie buvo dažniausiai pastebimi.

Pirmasis dėmesį, dėl minimalistinio, bet patrauklaus dizaino, atkreipė *AppSheet*. Pradėti programėlės kūrimą galima tik su jau siūlomais šablonais tokiais kaip: apklausa, inventorių, projektų stebėjimas, pamainos valdymas ir pan. Taigi iš esmės, galime pasirinkti jau sukurtas aplikacijas ir jas konfigūruoti. Kaip jau minėta anksčiau, apie minimalistinį dizainą, tai jis iš tiesų toks ir yra, daug opcijų dizaino pakeitimams kūrimo aplinka nepasiūlo. *AppSheet* turi savo duomenų bazę ir duomenų bazės valdymo sistemą. Intuityvus įrankis leidžiantis labai greitai sukurti programėlę savo ar savo įmonės reikmėms. Suteikti prieigą prie programėlės galima bet kam, tačiau už kiekvieną aktyvų naudotoją, reikia mokėti po penkis dolerius. Daugiau informacijos apie programėlės kūrimą, [5].

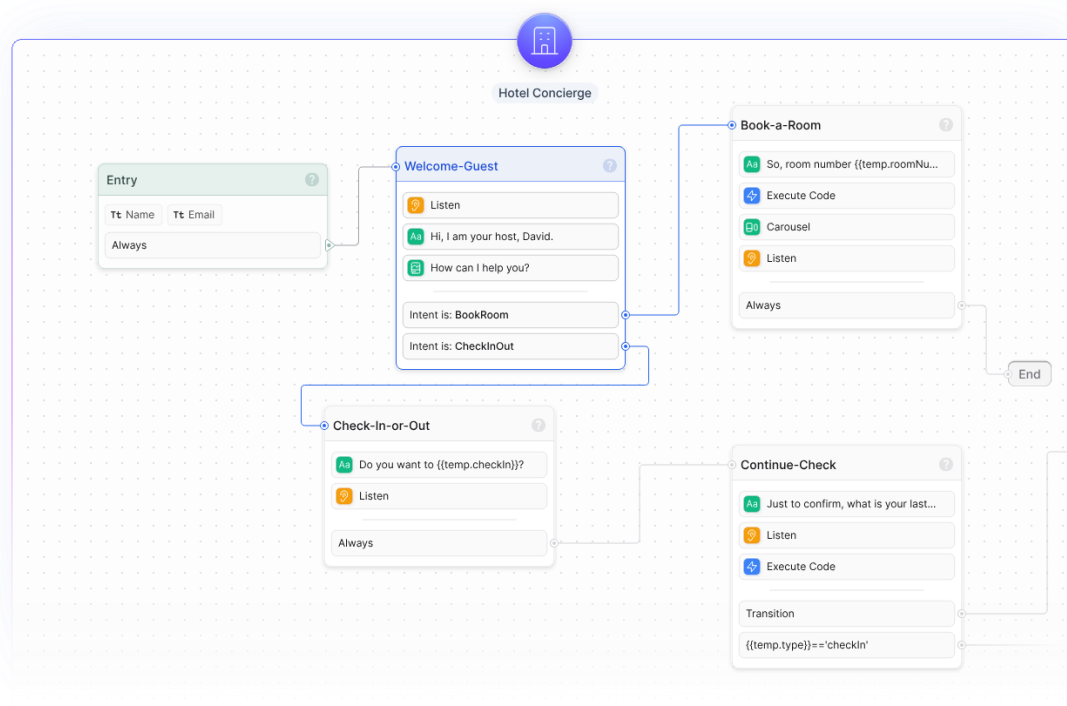
Kitas įrankis atkreipęs dėmesį yra *Mendix*. Iš programėlės kūrimo aplinkos galima susidaryti įspūdį, jog tai ne taip paprastai suprantamas įrankis, kaip *AppSheet*. Daugiau įvairiausių laukų, nustatymų, bei pasirinkimų. Dizaino prasme, konfigūravimo opcijų taip pat ne per

daugiausiai, kaip ir anksčiau minėtame *AppSheet*. Tiesa įmanoma stipriai pakeisti programėlės vartotojo sąsają, bet tam jau reiktų CSS žinių. Programėlės logikos valdymas, tai yra, kas nutinka po tam tikrų vartotojo įvesčių ar veiksmų yra grįstas diagramų principų, stumdant laukelius ir juos sujungiant. Tai labai panašu ir gali priminti objektų ryšių (angl. *Entity relationship*) diagramas. Šis įrankis kainuoja nuo 50-ies eurų per mėnesį, su šiuo pigiausiu planu, sukurta programėle naudotis galėtų iki 5-ių vartotojų. Mendix savo tinklalapyje pateiktame grafike rodo, jog tai yra populiariausia mažai kodo, jokio kodo platforma, [6].

Aukščiau minėti įrankiai neturi pokalbių robotų kūrimo aplinkos. Galima tik pokalbių roboto integracija į *Mendix* sistemą. Iš *Google* paieškoje rastų įrankių, siūlančių pokalbių roboto kūrimo galimybes visų įrankių kūrimo aplinka, principas ir būdas kaip kuriami ir dėliojami pokalbių roboto žingsniai atrodo labai panašūs. Skiriasi tik grafinė vartotojo sąsaja. Palyginimui *Landbot.io* (žr. 2.7 pav.) ir *Botpress* (žr. 2.8 pav.) pokalbių robotų kūrimo įrankių kūrimo aplinka.



2.7 pav. *Landbot.io* pokalbių robotų kūrimo įrankis [7]



2.8 pav. Botpress pokalbių robotų kūrimo įrankis [8]

Taip pat pats kūrimo principas labai panašus į *Microsoft Copilot Studio* (žr. 2.3 pav.), kuomet įvairūs veiksmai ir žingsniai, tokie kaip žinutės išsiuntimas arba naudotojo įvestis, sujungiami į vieną seką pagal kurią norima, jog vyktų pokalbis su pokalbių robotu.

Tačiau renkantis pokalbių robotų kūrimo įrankį esminis ir pagrindinis dalykas yra integracija. Tai yra kur, kokioje sistemoje sukurtas pokalbių robotas galės būti patalpintas ir veikti. Dažniausiai pokalbių robotai yra sutinkami elektroninėse parduotuvėse, kaip virtualūs asistentai arba susirašinėjimo programėlėje *Facebook Messenger*. Taigi integracija gali būti į svetainę, susirašinėjimo programėlės tokias kaip *Facebook Messenger*, *WhatsApp*, *Telegram* ir panašiai. Pavyzdžiui *Landbot.io* pagalba sukurtas pokalbių robotas gali būti integruotas į *Facebook Messenger*, žiniatinklio programą (angl. *web application*) arba *WhatsApp*. O *Botpress* turi daugiau integravimo galimybių, tokių kaip: *Facebook Messenger*, *WhatsApp*, *Gmail*, *Microsoft Teams*, *Viber* ir dar daugiau.

Darbo vietos rezervavimo sistema kuriama *Microsoft Power Platform* pagalba, *Microsoft Teams* aplinkoje, taigi, pokalbių robotą turi būti galima integruoti į *Microsoft Teams*. Tam gali būti naudojami du anksčiau minėti įrankiai: *Botpress* arba *Microsoft Copilot Studio*. Licencijų klainos palyginimui:

- *Botpress* [9] – nemokamas planas su 2000-iais žinučių arba 495-i doleriai mėnesiui už 250000-ių žinučių.
- *Microsoft Copilot Studio* [1] – pokalbių robotus galima kurti nemokamai *Microsoft Teams* viduje turint *Microsoft365* arba *Office365* licencijas.

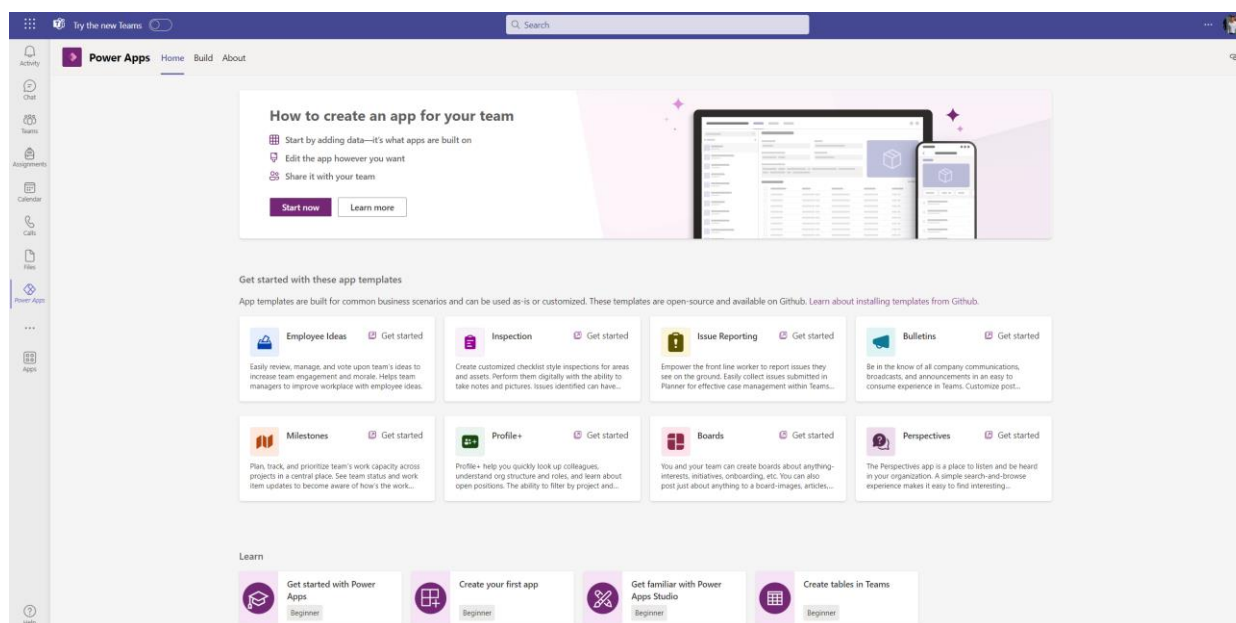
Kaip jau buvo minėta sistema kuriama ir veiks *Microsoft Teams* viduje, tai reiškia, jog jos kūrėjai ir naudotojai jau turi reikiamas licencijas pokalbių robotų kūrimui, bei naudojimuisi jais. Taip pat *Microsoft Copilot Studio* yra *Microsoft Power Platform* dalis. Dėl visų šių priežasčių pokalbių robotų kūrimui pasirinktas *Microsoft Copilot Studio* įrankis.

2.4. Kodėl Microsoft Power Platform

Prasidėjusi Covid-19 viruso pandemija didžiąją dalį žmonių išvarė iš ofisų ir privertė dirbi iš namų, pakeitė darbo tvarką bei įpročius. Akivaizdu, jog tam reikia tinkamo įrankio kuriuo būtų galima komunikuoti su komanda, rengti vaizdo konferencijas, bendradarbiauti ir atlikti kitus įprastus veiksmus darbe naudojantis virtualia aplinka. 2023 m. duomenimis [10], virš vieno milijono organizacijų, virš trijų šimtų dvidešimt milijonų naudotojų visame pasaulyje naudoja *Microsoft Teams*. Vartotojų skaičius ženkliai išaugo būtent pandemijos metu. Kai tuo tarpu *Mendix* savo oficialioje svetainėje [6] skelbia, jog jų paslaugomis naudojasi virš keturių tūkstančių įmonių, bei virš penkiasdešimties milijonų naudotojų. Taigi galima daryti prielaidą, jog didžioji dalis įmonių jau naudoja *Microsoft Teams* kaip bendradarbiavimo platformą. Tai reiškia, jog norint sukurti darbo vietos rezervacijos programėlę nereikės rūpintis dėl papildomų ar kitos platformos licencijų pirkimo, kadangi turint *Office 365* arba *Microsoft 365* licencijas, programėles galima naudoti ir kurti pačioje *Microsoft Teams* platformoje.

Kiekvienai komandai (angl. *Team*) galima sukurti *Microsoft Teams* duomenų saugyklos aplinką (angl. *Dataverse for Teams*), (žr. 2.9 pav.), kurioje bus talpinamos programėlės, pasikartojančių užduočių automatizavimas, pokalbių robotai, bei saugomi duomenys. Tam, kad sukurtos programėlės būtų lengvai pasiekiamos ir prieinamos *Microsoft Teams* platformoje, jas galima prisegti komandos kanale kaip skirtuką arba kairėje juostoje prie pagrindinių programėlių. *Microsoft Teams* duomenų saugykla yra reliacinė duomenų bazė. Reliacinėje duomenų bazėje duomenys yra saugomi lentelėse, tai yra, stulpeliuose, kur kiekvienas stulpelis atitinka duomenų tipą ir eilutėse, kur eilutė yra įrašas. Nereliacinės – dokumentuose, grafuose, kolekcijose.

Būtent dėl anksčiau išvardintų priežasčių, galima teigti, jog *Microsoft Power Platform*, šiuo atveju yra geriausias pasirinkimas sukurti darbo vietos rezervacijos programėlę



2.9 pav. Power Apps aplinka Microsoft Teams platformoje (Dataverse for Teams).

2.5. Bot Framework Composer ir adaptyvios kortelės

Bot Framework Composer sukurtas *Bot Framework* programinės įrangos kūrimo rinkiniu (angl. *Software Development Kit – SDK*) ir yra atvirojo kodo (angl. *Open-source*) integruota programavimo aplinka (angl. *Integrated Development Environment – IDE*). Įrankis skirtas pokalbių robotams kurti, taip pat kaip ir *Microsoft Copilot Studio*. Tačiau *Bot Framework Composer* turi daug daugiau galimybių, tokių kaip: natūralios kalbos supratimas, *REST* integracija, adaptyvios kortelės (angl. *Adaptive cards*), klausimų atsakymų dialogai ir dar daugiau. Įrankio įdiegimo instrukcija dokumentacijoje [11].

Adaptyvios kortelės yra grafinės sąsajos elementais perduodamas turinys kuris gali būti siunčiamas kaip žinutė. Tai gali būti: forma duomenų įvesčiai, pažymėtos lokacijos žemėlapyje atvaizdavimas, pasirinkimų įvestis, trumpa orų prognozė ir dar daugiau galimybių. Adaptyvios kortelės taip pat reaguoja į naudotojo veiksmus, pavyzdžiui: paspaudus mygtuką atidaroma internetinė svetainė arba žemėlapis, pasirinkimo atveju – išsaugomas atsakymas, išsiunčiami duomenys ir panašiai. Adaptyvios kortelės veikia su pokalbių robotais. Pavyzdžiui paklausus kokia rytojaus orų prognozė, naudotojui būtų sudėtinga ir nepatogu skaityti tekstą apie rytojaus orus. Daug geriau gauti paveikslėlį su ikonomis paros meto orui ir temperatūrai atvaizduoti. Tai tik vienas iš daugelio adaptyvių kortelių panaudojimo būdų. Adaptyvios kortelės kuriamos ir generuojamos tekstiniu duomenų formatu *JSON* (angl. *JavaScript Object Notation*). Toks formatas dažniausiai naudojamas duomenims tarp serverio ir kliento perduoti, daugiau adaptyvių kortelių pavyzdžių [12].

Bot Framework Composer kurtus dialogus ir pokalbių robotų atsakymus galima panaudoti *Microsoft Copilot Studio* pokalbių robotuose, taip pat adaptyvias korteles galima siųsti per *Microsoft Teams*. Šių dalykų prireiks darbo vietos rezervavimo procese naudojant pokalbių robotą. Norint naudotojui išvesti galimų ofisų pasirinkimus iš duomenų bazės (angl. *Datavers*) *Microsoft Copilot Studio* to padaryti negalima, nes kaip pasirinkimas yra priimamas tik įvedamas tekstas, negalima naudoti kintamojo. Kas sukeltų šiek tiek problemų, nes norint atnaujinti ofisų sąrašą tai reikėtų daryti per dvi vietas, atnaujinant duomenų lentelę, bei pridėdant pasirinkimą pokalbių robotui. Lygiai tas pats ir su ofiso aukštais, bei teritorijomis. Panaši problema ir su datos įvestimi, nes kaip ir programėlėje, naudotojas turi galėti pasirinkti tik dviejų savaitių datas. Teksto įvestis netinka, nes tuomet negalime apriboti naudotojo, o pasirinkimui ta pati bėda, nes negalima naudoti kintamųjų. *Bot Framework Composer* bus naudojamas būtent tam, jog gautus ofisų bei aukštų pasirinkimus būtų galima perduoti į pokalbių robotą ir išvesti naudotojui kaip pasirinkimą adaptyviose kortelėse. Taip pat naudojant adaptyvią kortelę naudotojas galės interaktyviai pasirinkti datą iš kalendoriaus.

3. SISTEMOS APRAŠYMAS

Norint sukurti programėlę *Power Apps* ir pokalbių robotą *Microsoft Copilot Studio* pagalba, reikia išsiaiškinti kaip šie įrankiai veikia, kokias aplinkas ir kūrimo metodus naudoti. Šiame skyriuje aptartas programos gyvavimo ciklo valdymas, duomenų modelis, išanalizuotas *Microsoft Copilot Studio* veikimas, kokius veiksmus galima naudoti. Taip pat aptartas darbo vietos rezervavimo procesas, patvirtinimo žingsnio kūrimas.

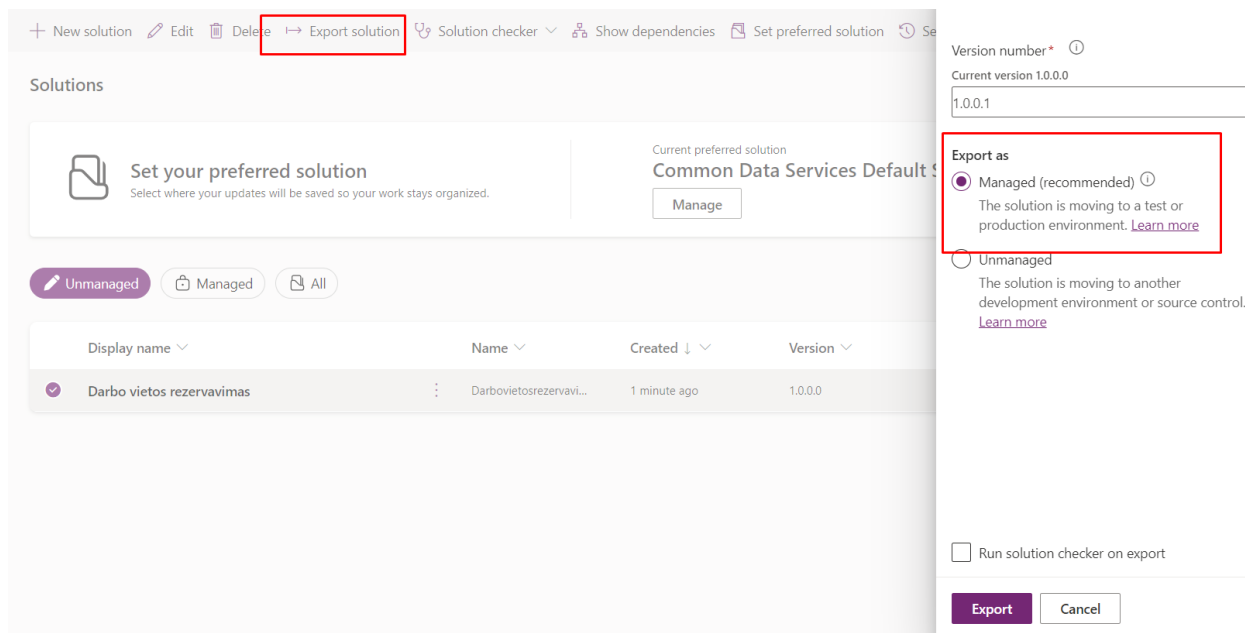
3.1. Microsoft Teams aplinka

Darbo vietos rezervacijos sprendimui kurti bus naudojama Microsoft Teams duomenų saugyklos aplinka (angl. *Dataverse for Teams*). Taip įmonės darbuotojams bus paprasčiausia pasiekti programėlę norint užsirezervuoti darbo vietą, nereikės atsisiųsti jokios programinės įrangos į savo įrenginį. Norint pradėti įgyvendinti šį sprendimą tereikės susikurti komandą (angl. *Team*), atsidaryti *Power Apps* iš Microsoft Teams programėlių, pasirinkti savo ką tik sukurtą komandą ir viskas, aplinka sukurta. Kiekvienoje aplinkoje yra integruota duomenų saugykla, tai yra, reliacinė duomenų bazė.

3.2. Programos gyvavimo ciklo valdymas

Programos gyvavimo ciklo valdymas (angl. *Application Lifecycle Lanagement*) apima programinės įrangos kūrimą, testavimą, priežiūrą, pakeitimų valdymą [13]. Darbo vietos rezervavimo programėlę laikant tik vienoje aplinkoje ir darbuotojams jau naudojantis ja, atnaujinimo darbai ir pakeitimai tiesiogiai darytų įtaką produkcinės programėlės veikimui. Tai reiškia, jog kiekvieną kartą išsaugojus pakeitimus jie būtų iš karto prieinami naudotojams, tuomet atsiranda klaidos rizika, kadangi pakeitimai gali būti nepilnai išbaigti ir neištestuoti. Taip pat testuojant, produkcinė duomenų bazė būtų užteršta testiniais duomenimis, kurie atsispindėtų ir rezervacijų apžvalgoje ir panašiai. Tam, jog to išvengti reikėtų turėti mažiausiai dvi aplinkas. Pirmoji kūrimo (angl. *Development*) ir testavimo (priėmimo) (angl. *Acceptance*), antroji, produkcinė (angl. *Production*). Kūrimo ir testavimo aplinkoje būtų atliekami pakeitimai, atliekamas testavimas ir saugomi testavimo duomenys. Tik po to kai pakeitimai ištestuojami ir patvirtinami, jog atitinka reikalavimus yra perkeliama į produkcinę aplinką, kuri pasiekama galutiniams naudotojams.

Programos gyvavimo ciklo valdymas *Microsoft Power Platform* yra gan paprastas, visus sistemos komponentus (programėles, automatizuotus procesus, lenteles, pokalbių robotus ir t.t.) reikia sudėti į vieną sprendimą (angl. *Solution*), kuris yra kaip aplanka. Jį išeksportuoti iš kūrimo aplinkos (žr. 3.1 pav.), atsisiųsti suarchyvuotą failą ir importuoti į produkcinę aplinką. Sprendimą reikėtų išeksportuoti kaip valdomą (angl. *Managed*), tai reiškia, jog sprendimo viduje esantiems komponentams nebus galima atlikti jokių pakeitimų. Daugiau apie sprendimus, programos gyvavimo ciklo valdymą *Power Platform* [13].



3.1 pav. Sprendimo (angl. *Solution*) išeksportavimas.

3.3. Duomenų modelis

Prieš pradėdant kurti darbo vietos rezervavimo sistemą svarbu nustatyti duomenų bazės struktūrą, modelį, nustatyti būsimas duomenų esybes ir ryšius. Tam padaryti ir vizualizuoti puikiai tinka esybių ryšių fizinė diagrama (angl. *Entity relationship diagram*). Ši diagrama darbo vietos rezervacijos sistemai pateikta 1 priede. Sistema orientuota į įmones, žemiau pateikta schema yra pateikta anglų kalba.

Darbo vietos rezervacijai bus naudojamos šios lentelės:

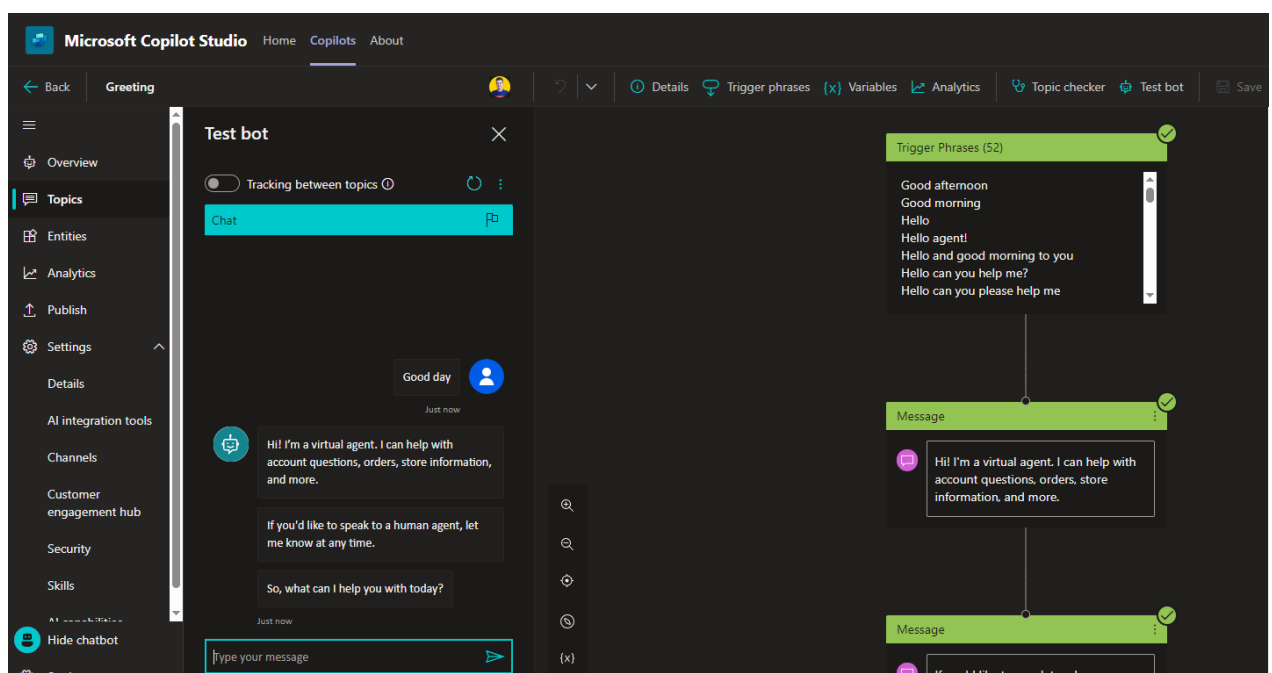
- Rezervacijos (angl. *Bookings*). Šioje lentelėje bus saugomos rezervacijos ir informacija apie jas: rezervacijos identifikacinis numeris (angl. *Reservation ID*), ofisas (angl. *Office*) bei vieta ar aukštas (angl. *Area*) kuriame atlikta rezervacija, rezervacijos data (angl. *Reservation date*), kam užrezervuota (angl. *Reserved For*), bei *Outlook* kalendoriaus įvykio identifikacinis numeris (angl. *OutlookEventID*). Taip pat saugomas rezervacijos statusas (angl. *Status*) pagal kurį bus filtruojama visų rezervacijų apžvalga.
- Naudotojai (angl. *Users*). Lentelė kaupti naudotojų duomenis tokius kaip: pirminis vartotojo identifikacinis numeris (angl. *Primary ID*), pilnas vardas (angl. *Full Name*), elektroninis paštas (angl. *Email*), bei pasirinktas ofisas (angl. *SelectedBuilding*). Pasirinktas ofisas bus saugomas tam, kad naudotojui nereikėtų pasirinkti ofiso kiekvieną kartą norint užrezervuoti darbo vietą, tai bus kaip standartinis pasirinkimas. Lentelėje taip pat egzistuoja naudotojo rolės (angl. *Role*) laukas pagal kurį programėlėje atpažįstami administratoriai.
- Pastatai (angl. *Buildings*). Šioje lentelėje bus įrašai visi įmanomi ofisų pasirinkimai. Vienas ir vienintelis laukas šioje lentelėje, tai – pavadinimas (angl. *Name*).
- Aukštai ir teritorijos (angl. *Floors and Areas*). Lentelė kurioje bus laikomi galimi aukštų ir teritorijų pasirinkimai atliekant darbo vietos rezervaciją. Pagrindinis laukas

tai teritorijos pavadinimas (angl. *Area name*), ofisas kuriam priklauso šis aukštas ar teritorija (angl. *Office*), bei kiek rezervacijų gali talpinti (angl. *Capacity*).

3.4. Microsoft Copilot Studio veikimas

Prieš pradėdant kurti darbo vietos rezervacijos pokalbio eigos ar proceso schemą, bei patį pokalbių robotą, reikėjo išsiaiškinti kaip *Microsoft Copilot Studio* pagalba kuriami pokalbių robotai, iš ko susideda, bei kaip veikia.

Pokalbių robotas pagrįdė susideda iš temų (angl. *Topics*). Temoje nurodoma ir dėliojama pokalbio eiga, tai yra, kuomet robotas išsiunčia žinutes ar atsakymus, arba kuomet laukiama naudotojo teksto įvesties ir panašiai. Tačiau robotui pagal naudotojo įvestį reikia atpažinti į kokią temą jį nukreipti. Tam yra naudojamos paleidimo frazės (angl. *Trigger phrases*). Kiekvienai temai nurodomas sąrašas tokių frazių, pagal kurias robotas nukreips pokalbio eigą. Pavyzdžiui norint sukurti temą kuri aktyvuojama kuomet naudotojas pradeda pokalbį su robotu galima būtų naudoti tokias frazes: labas, sveiki, laba diena ir panašiai. Nurodžius šias frazes, norint aktyvuoti temą naudotojui nebūtina parašyti žodį lygiai taip pat. Galima įvesti ir klaidą ar vietoj „laba diena“ parašyti „labas vakaras“, tokiu atveju vis tiek bus pereita į šią temą, kadangi robotas atpažįsta, jog tai pasisveikinimo frazė pagal jau pateiktą frazių sąrašą. Tuomet pokalbio eiga krypta pagal tai kaip nurodyta pasisveikinimo temoje, tai galėtų būti roboto atsakymas – „Sveiki, kuom galėčiau padėti?“. Žinoma, viskas vykta anglų kalba, nes kaip jau buvo minėta, robotas lietuvių kalbos nesupranta. Po roboto atsakymo nurodoma, jog laukiama naudotojo įvesties ir pagal naudotojo įvestį toliau ieškoma temų į kurias būtų galima pereiti. Atveju kuomet jokia tema nerandama pagal įvestą frazę, robotas automatiškai atsako, jog pagal šią įvestį jokių temų nerado ir prašo pakartoti žinutę. Verta paminėti, jog pasisveikinimo temos kurti nereikia, kadangi pokalbių robote sistema automatiškai sukuria tokią temą su 52-jomis temos atpažinimo frazėmis. Temoje taip pat pridėti trys veiksmai išsiunčiantys roboto pasisveikinimo žinutes, kurias galima redaguoti ar pridėti naujų veiksmų. Pasisveikinimo tema ir pokalbių robotas testavimui pavaizduoti 3.2 paveiksle.



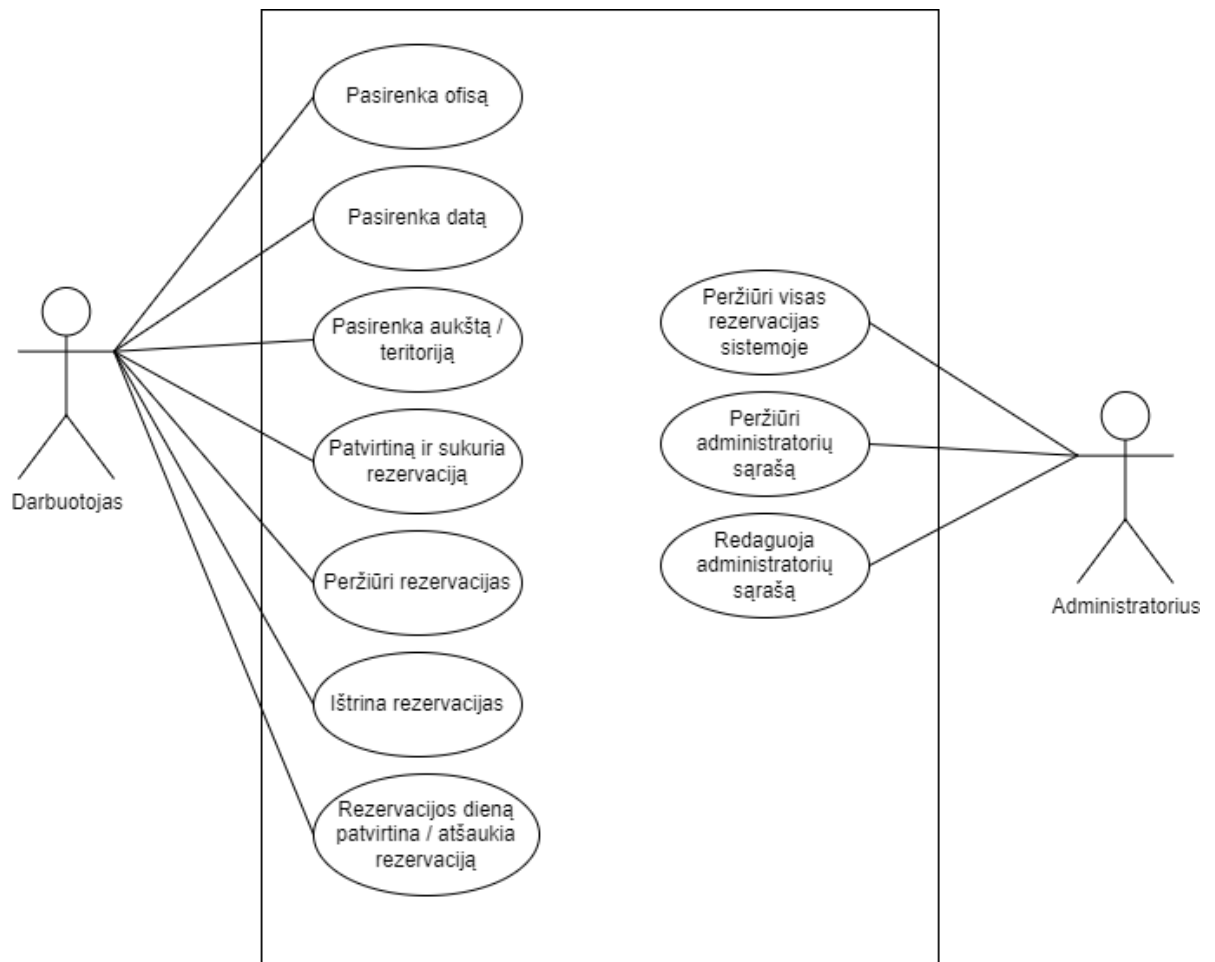
3.2 pav. Pasisveikinimo tema.

Temoje, galima pridėti šiuos veiksmus:

- Paklausti klausimo (angl. *Ask a question*) – naudotojui užduodamas klausimas ir jo atsakymas išsaugomas į kintamąjį. Atsakymą galima apriboti leidžiant pasirinkti vieną iš kelių atsakymų. Taip pat galima leisti įvesti savo tekstą ir išsaugoti jį visą. Dar galima pasirinkti, jog iš vartotojo atsakymo robotas atpažintų ir išsaugotų tam tikrą subjektą (angl. *Entity*). Galima pasirinkti, jog robotas įvestyje atpažintų ir išsaugotų amžių, datą, šalies pavadinimą ir dar daugiau. Galima sukurti ir savo subjektą kūrimo aplinkoje pasirinkus subjektai (angl. *Entities*). Savo subjektų pridėjimas vyksta nurodant sąrašą pavyzdžių arba reguliariąją išraišką pagal kurią bus atpažįstamas ir išsaugomas subjektas.
- Sąlyga (angl. *Condition*) – pasirenkamas kintamasis ir nurodoma sąlyga jam. Taip galima patikrinti vartotojo įvestį ir pagal ją nukreipti pokalbio eigą.
- Iškviesti veiksmą (angl. *Call an action*) – su šiuo veiksmu paleidžiamas *Power Automate* procesas. To prireiks kuomet pagal naudotojo įvestį reikės sukurti įrašą darbo vietos rezervacijų lentelėje.
- Išsiųsti žinutę (angl. *Show a message*) – robotas naudotojui išsiunčia laukelyje nurodytą tekstą.
- Nukreipti į kitą temą (angl. *Redirect to another topic*) – pasirenkama tema į kurią bus nukreiptas pokalbis pasiekus šį veiksmą.
- Užbaigti pokalbį (angl. *End the conversation*) – pokalbio užbaigimui. Galima pasirinkti užbaigti pokalbį su apklausa (angl. *End with survey*). Tokiu atveju pereinama prie sistemos automatiškai sukurtos temos – pokalbio pabaiga (angl. *End of conversation*), kurioje naudotojo prašoma įvertinti patirtį bendraujant su pokalbių robotu. Įvertinus teigiamai arba neigiamai pereinama prie atitinkamų temų su tolimesniais pasiūlymais arba pokalbis galutinai užbaigiamas. Taip pat galima pasirinkti, jog pokalbis būtų perduodamas agentui (angl. *Transfer to agent*). Tai yra visa pokalbio istorija perduodama žmogui su kuriuo tęsiamas pokalbis.

3.5. Darbo vietos rezervavimo programėlės funkcionalumas

Darbo vietos rezervavimo programėlės funkcionalumui atvaizduoti pasitelkta panaudos atvejų diagrama (angl. *Use case diagram*). Panaudos atvejų diagrama, tai vieningojo modeliavimo kalbos (angl. *Unified Modeling Language*), (toliau – UML), diagrama. UML, tai standartizuota modeliavimo ir specifikacijų kalba, UML diagramų pagalba galima atvaizduoti programinės įrangos sistemos architektūrą visiems komandos nariams vienodai suprantamai. Daugiau apie UML ir jos specifikacijas [14]. Taigi panaudos atvejų diagrama atvaizduoja projektuojamos sistemos funkcionalumus, vartotojus (roles) ir kokias funkcijas jie gali atlikti, (žr. 3.3 pav.). Panaudos atvejų diagrama taip pat kaip ir aukšto lygio struktūrinė schema (angl. *High level structure chart*) skirta sistemos funkcionalumui atvaizduoti, schema orientuota į procesą. Aukšto lygio struktūrinėje schemoje, pateiktoje 1 priede, pavaizduoti žingsniai norint atlikti rezervaciją, peržiūrėti ar ištrinti rezervacijas. Tačiau plečiant aukšto lygio struktūrinę schemą naujais funkcionalumais ji taptu sunkiai įskaitoma, nebūtų aišku kuri naudotojo rolę kuri funkcionalumą gali pasiekti. Žinoma būtų galima kurti dvi aukšto lygio struktūrines schemas kiekvienai naudotojo rolei atskirai. Tačiau tiek panaudos atvejų diagrama, tiek aukšto lygio struktūrinė schema vienodai detalios atvaizduoja funkcionalumus, todėl paprasčiau sukurti panaudos atvejų diagramą, kuri ir yra skirta būtent funkcionalumo atvaizdavimui per naudotojo rolę.



3.3 pav. Panaudos atvejų diagrama.

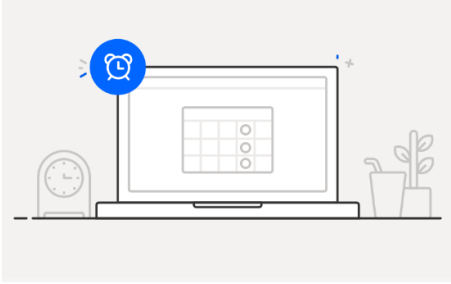
Šiuo atveju panaudos atvejų diagrama gan paprasta, kadangi darbo vietos rezervavimo sistemoje egzistuoja tik dvi naudotojų rolės. Tačiau plečiant sistemą ir pridėdant daugiau rolių būtų galima labai paprastai ir aiškiai atvaizduoti funkcijas pasiekiamas kiekvienai rolei.

3.6. Patvirtinimo žingsnis rezervavimo procese

Norima, jog tą dieną kai darbuotojas turi darbo vietos rezervaciją, jam elektroniniu paštu ar *Microsoft Teams* žinute būtų atsiunčiamas prašymas patvirtinti, jog darbuotojas į rezervuotą darbo vietą tikrai atvyko ar atvyks. Tokiu būdu nepatvirtintos rezervacijos būtų ištrinamos arba pažymimos kaip atšauktos, taigi rezervacijų sekimas būtų ženkliai tikslesnis ir atsirastų daugiau vietų norintiems rezervuoti tą pačią dieną.

Šio reikalavimo įgyvendinimui reikėtų kiekvieną rytą gauti sąrašą rezervacijų ir jas atlikusiems darbuotojams išsiųsti prašymą patvirtinti rezervaciją. Tam reikėtų *Power Automate* įrankio pagalba sukurti suplanuotą veiksmą (angl. *Scheduled flow*), dokumentacija [15], sukūrimas (žr. 3.4 pav.).

Build a scheduled cloud flow



Stay on top of what's important without the effort—you choose when and how often the flow runs.

Examples:

- Automate team reminders to submit expense reports
- Auto-backup data to designated storage on a regular basis

Flow name

Add a name or we'll generate one

Run this flow *

Starting 1/12/23 * at 10:00 AM

Repeat every 1 * Minute

This flow will run:

Every minute

Skip Create Cancel

3.4 pav. Suplanuto veiksmo (angl. *Scheduled flow*) sukūrimas.

Kuriant *shceduled flow* reikia įrašyti jo pavadinimą, kurią dieną ir valandą turėtų pradėti veikti ir kas kiek laiko kartoti. Kad tokio tipo automatizuotus procesus ištestuoti, nereikia laukti nustatyto laiko, kada jie patys automatiškai pasileidžia. Kūrimo aplinkoje išsaugojus *scheduled flow*, galima paspausti mygtuką testuoti (angl. *Test*) ir pasirinkus vieną iš testavimo būdų pradėti procesą.

3.7. Darbo vietos rezervavimo procesas pokalbių robotu

Darbo vietos rezervavimo sistemos procesas ir funkcionalumas vizualizuotas pasitelkiant aukšto lygio struktūrinę schemą (angl. *High level structure chart*), kuri pateikta 2 priede. Schemoje pavaizduoti žingsniai norint atlikti rezervaciją, peržiūrėti ar ištrinti rezervacijas. Schema orientuota į procesą. Schema atvaizduoja paprastam naudotojui pasiekiamą funkcionalumą, administratoriaus rolė neįtraukta, kadangi naudojantis pokalbiu robotu nėra tiesioginio būdo patikrinti ar naudotojas yra administratoriaus.

Taigi visi žingsniai norint rezervuoti darbo vietą pavaizduoti aukšto lygio struktūrinėje schemoje turėtų būti perkelti ir į pokalbių robotą. Naudotojui davus užklausą ir aktyvavus rezervacijos temą, pokalbių robotas turėtų paprašyti įvesti arba pasirinkti ofisą, tuomet datą, aukštą arba teritoriją ir atvaizduoti įvestus pasirinkimus, bei patvirtinti rezervaciją.

Visoms įvardintoms įvestims pokalbių robotas turėtų rodyti pasirinkimus. Žinoma paprasčiausia būtų, jog įvedus užklausą pavyzdžiui – „Labas, noriu rezervuoti darbo vietą ofise Beta, rytoj, 2-ame aukšte.“, pokalbių robotas atliktų rezervaciją. Tačiau problema atsiranda jau pirmoje įvestyje – ofiso. Naudotojas gali padaryti klaidą ofiso pavadinime ir *Power Automate* procesas atliekantis duomenų įrašymą jo neras, tas pats su ofiso aukštu. Datos atveju, galima pasirinkti sistemos automatiškai sukurtą subjektą kuris atpažįsta datą iš įvairių apibūdinimų. Pavyzdžiui įvedus žodį „rytoj“ – į kintamąjį būtų išsaugoma rytojaus data, „kitą pirmadienį“ – kito pirmadienio data ir panašiai. Tačiau problema čia, jog negalime apriboti naudotojo įvesties. Tai reiškia, jog naudotojas galėtų įvesti net gi kitų metų datą. Būtų galima po datos įvesties pradėti

Power Automate procesą ir patikrinti ar data atitinka numatytus režius, tačiau tai prailgintų rezervavimo procesą naudotojo klaidos atveju, todėl geriausia naudotojui duoti pasirinkti. Taigi tai nėra įmanoma, nes įvedami duomenys turi būti tikslūs, kadangi jie perduodami *Power Automate* procesui kuris paprasčiausiai neras tokių pasirinkimų kuriant rezervacijos įrašą.

Todėl darbo vietos rezervavimas pokalbių robotu bus išskaidomas į žingsnius ir viskas bus atliekama taip pat kaip programėlėje tik jau su pokalbių roboto pateiktais pasirinkimais. Viskas turėtų atrodyt maždaug taip:

1. Naudotojas pokalbyje nurodo, jog norėtų atlikti naują darbo vietos rezervaciją.
2. Pokalbių robotas paprašo naudotojo pasirinkti ofisą. Ofiso pasirinkimai gaunami iškviečiant *Power Automate* procesą, kuris iš pastatų (angl. *Buildings*) lentelės gauna visus įrašus ir gražina juos atgal į *Microsoft Copilot Studio*.
3. Kaip ir programėlėje taip ir pokalbių robote naudotojas turi galėti įvesti arba pasirinkti tik einamosios ir ateinančios savaitės datas. Taigi atvira įvestis čia irgi netinka, kaip minėta anksčiau, naudotoją reikia apriboti ir leisti tik pasirinkti.
4. Aukšto ar teritorijos pasirinkimai, lygiai taip pat kaip ir ofiso pasirinkimas. Skirtumas tik tas, jog proceso viduje aukštų ir teritorijų (angl. *Floors and Areas*) lentelės išfiltruojama pagal pasirinktą ofisą.
5. Naudotojui viename tekste parodomi jo pasirinkimai ir klausiama ar atlikti rezervaciją.

4. DARBO VIETOS REZERVAVIMO SISTEMOS KŪRIMAS

Šiame skyriuje aprašyti darbo įrankiai, aplinkos paruošimas, bei programėlės ir pokalbių roboto kūrimo procesai.

4.1. Darbo aplinka

4.1.1. Darbo įrankiai

Darbo vietos rezervacijos sistema sukurta *Microsoft Teams*, kurioje sukurta komanda (angl. *Team*) ir kūrimo aplinka (*Dataverse for Teams*). Programėlės kūrimui naudotas *Power Apps* įrankis, o patvirtinimo žingsnio procesui *Power Automate*.

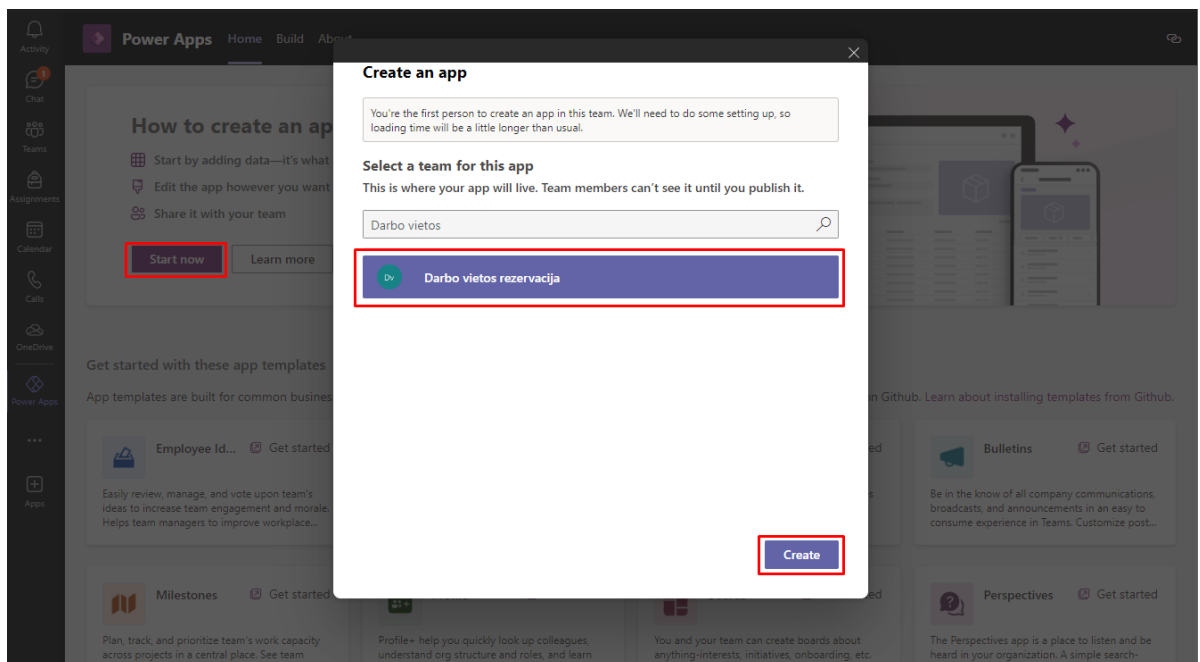
Pokalbių robotas kuriamas taip pat *Microsoft Teams* viduje, kūrimo ir testavimo aplinkoje, naudojant *Microsoft Copilot Studio* įrankį.

4.1.2. Aplinkos paruošimas

Pirmiausia organizacijos *Microsoft Teams* buvo sukurta komanda (angl. *Team*). Taip padaryti galima pasirinkus komandos (angl. *Teams*) ir paspaudus mygtuką prisijungti arba sukurti komandą (angl. *Join or create team*). Toliau pasirinkti pagal kokį šabloną sukurti komandą (šiuo atveju tai nebuvo svarbu) įrašyti komandos pavadinimą ir aprašymą.

Komanda sukurta, galima judėti prie dalies kurioje buvo sukurta duomenų saugykla (angl. *Dataverse for Teams*). Tam, reikėjo atsidaryti *Power Apps* programėlę *Microsoft Teams* viduje. Tai galima padaryti išskleidus daugiau programėlių (angl. *View more apps*), tai yra, paspaudus tris taškus kairėje juostoje ir pasirinkus *Power Apps*. Jeigu *Power Apps* programėlės pasirinkimo nėra, gali būti, jog ją reikia pridėti paspaudus gauti daugiau aplikacijų (angl. *Get more apps*) apačioje.

Tam, kad sukurti duomenų saugyklą komandoje, reikia paspausti pradėti dabar (angl. *Start now*), pasirinkti sukurta komandą ir paspausti sukurti (angl. *Create*) (žr. 4.1 pav.).

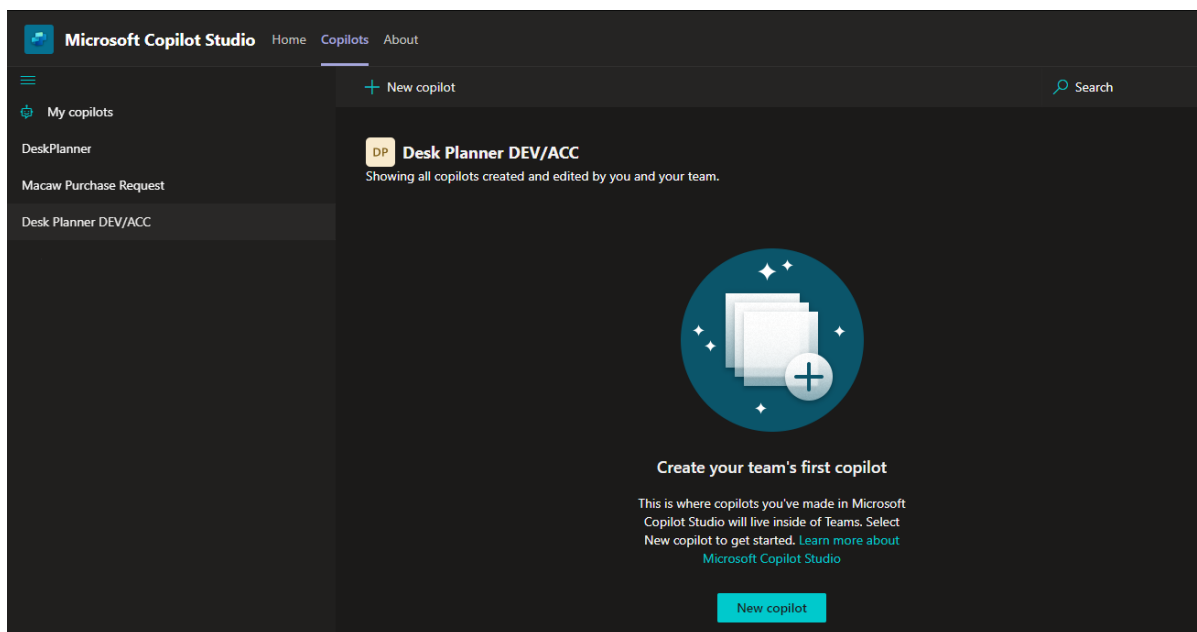


4.1 pav. Duomenų saugyklos ir kūrimo aplinkos sukūrimas (*Dataverse for Teams*).

Sukūrus aplinką iš pradžių galima sukurti tik programėlę arba pokalbių robotą. Taigi jokių kitų sistemos dedamųjų (lentelių, automatizavimo užduočių) pridėti negalima neturint programėlės. Todėl geriausia sukurti tuščią programėlę ir ją išsaugoti. Pasirinkus reikiamą aplinką (pavadinimas lygus komandos pavadinimui), spausti nauja (angl. *New*) ir pasirinkti programėlę (angl. *App*). Vėliau bus paprašyta įvesti programėlės pavadinimą ir ją išsaugoti. Ši programėlė nebus naudojama darbo vietos rezervacijai, kadangi pagal numatytuosius parametrus programėlė kuria inicializuojama kūrimo aplinka yra planšetinio kompiuterio formos. Geriausia būtų sukurti mobiliojo telefono formos programėlę, patogesniai naudojimui mobiliuosiuose įrenginiuose. Tai taip pat nesukels nepatogumų naudojantis programėle ir kompiuteriu.

Taigi aplinka paruošta ir joje bus pradėta konfigūruoti duomenų bazė, tai yra, kuriamos lentelės, pridėta mobiliojo telefono formos programėlė, bei saugomas *Microsoft Copilot Studio* pagalba sukurtas pokalbių robotas. Kaip jau buvo minėta programos gyvavimo ciklui turėtų egzistuoti, bent jau dvi aplinkos, viena iš jų kūrimo ir testavimo, bei kita produkcinė, pasiekiamą galutiniais naudotoja. Produkcinėi aplinkai buvo sukurta nauja *Microsoft Teams* komanda su pavadinimu identifikuojančiu, jog tai produkcinė aplinka (*Desk Planner PROD*).

Norint pradėti kurti pokalbių robotą, reikėjo iš *Microsoft Teams* programėlių sąrašo įdiegti *Microsoft Copilot Studio* įrankį. Įdiegus ir atsidarius *Copilot Studio*, buvo pasirinkta kūrimo ir testavimo aplinka (žr. 4.2 pav.), bei pridėtas naujas pokalbių robotas (angl. *New copilot*).

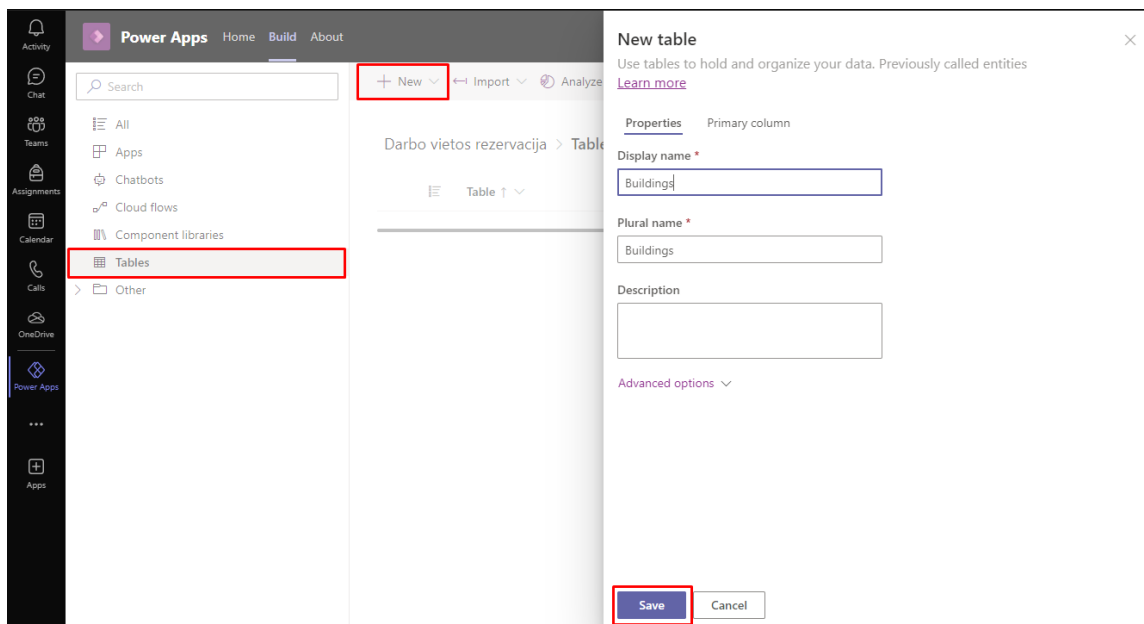


4.2 pav. *Microsoft Copilot Studio* pokalbių roboto pridėjimas.

Pridedant naują pokalbių robotą galima pasirinkti kokia kalba su juo bus bendraujama. Kadangi darbo vietos rezervacijos sistema sukurta anglų kalba, tai ir pokalbių robotui buvo pasirinkta anglų kalba. Lietuvių kalbos tarp galimų kalbų pasirinkimo nebuvo.

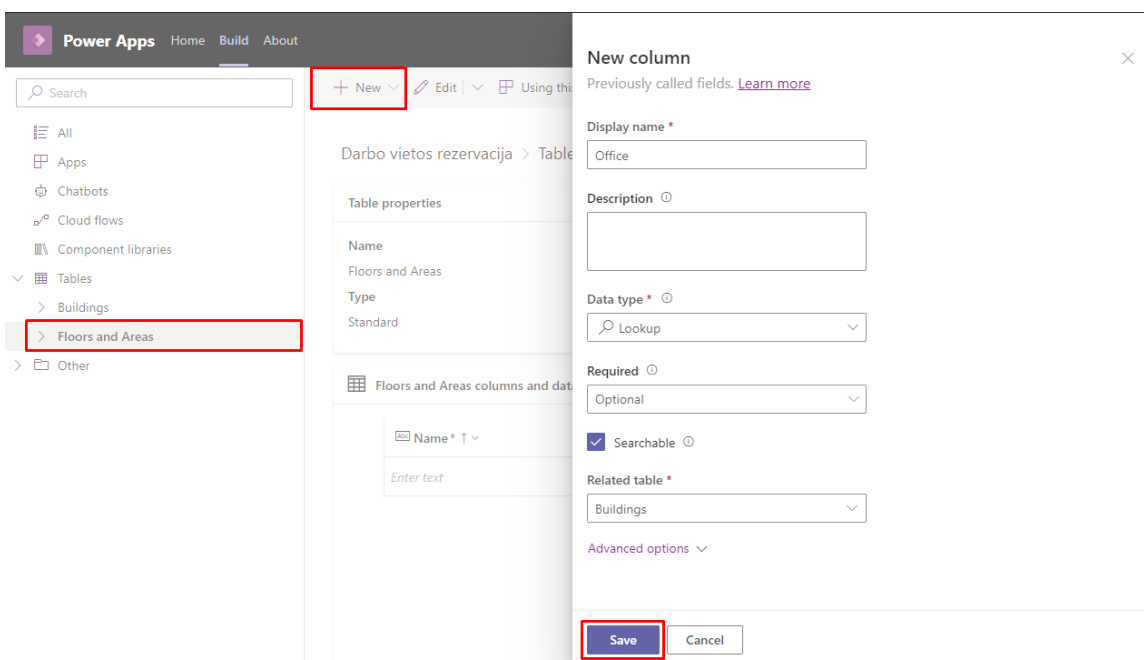
4.2. Duomenų bazės kūrimas

Pirmiausia buvo pradėta nuo lentelių sukūrimo. Lentelės sukūrimo žingsniai (angl. *Table*) (žr. 4.3 pav.).



4.3 pav. Lentelės sukūrimo žingsniai.

Verta paminėti, jog vartotojų (angl. *Users*) lentelės kurti nereikia, kadangi tai yra sisteminė lentelė, tai yra, sukuriamama automatiškai sukūrus aplinką. Joje reikia pridėti vieną papildomą duomenų stulpelį kuriame saugomas numatytasis vartotojo ofisas. Duomenų stulpelių kūrimas (4.4 pav.). Šiame pavyzdyje *Floors and Areas* lentelėje sukuriamas duomenų stulpelis *Office*. Jo duomenų tipas yra *Lookup*, tai reiškia, jog įrašas susietas su pasirinkta lentele (šiuo atveju *Buildings*). Kuriant aukšto ar teritorijos įrašą galima pasirinkti kuriam ofisui jis priklauso, tai yra sukuriamas ryšys tarp lentelių. Pavyzdžiui duomenų stulpeliui *Capacity*, duomenų tipas turėtų būti sveikasis skaičius (angl. *Whole number*).



4.4 pav. Duomenų stulpelių kūrimas.

Naudotojų (angl. *Users*) lentelėje pridėtas papildomas laukas vartotojo rolei nustatyti, bei saugoti (angl. *Role*), pagal kurį programėlėje bus identifikuojama ar naudotojas yra administratorius. Sukurtas pasirinkimo tipo (angl. *Choice*) laukas su dvejais pasirinkimais: *Basic user*, *Admin*.

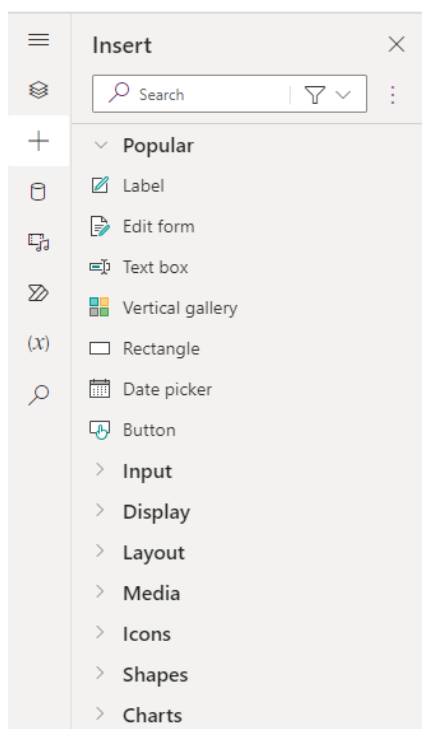
Tokiu pat būdu rezervacijų (angl. *Bookings*) lentelėje pridėtas papildomas laukas rezervacijos statusui nustatyti, bei saugoti (angl. *Status*), pagal kurį nepatvirtintos rezervacijos nebus rodomos visų rezervacijų sąraše. Rezervacijos statuso pasirinkimai: pateiktas, patvirtintas, nepasirodė (angl. *Submitted*, *Check-in*, *No-show*). Taip pat reikėtų nustatyti numatytuosius pasirinkimus šiems laukams. Naudotojo rolės numatytasis pasirinkimas – paprastas naudotojas (angl. *Basic user*), o rezervacijos statuso – pateiktas (angl. *Submitted*). Tokiu būdu programėlėje nereikės rašyti kodo, kuris nurodytų naudoti šiuos pasirinkimus naudotojui pirmą kartą atsidarius programėlę ar sukūrus naują rezervaciją.

Toliau žiūrint į esybių ryšių fizinę diagramą pateiktą 2 priede, sukuriamos reikiamos lentelės ir sukonfigūruotos taip, jog atitiktų numatytą duomenų modelį.

4.3. Programėlės kūrimas

Programėlę galima pradėti kurti nuo ekranų pridėjimo. Tikrai reikia, bent septynių ekranų: pagrindinio, rezervacijos datai pasirinkti, rezervacijos aukštui pasirinkti, rezervacijoms peržiūrėti, numatytam ofisui nustatyti, administratorių sąrašui konfigūruoti, bei visų rezervacijų apvažlgos ekranas administratoriams.

Geriausia pradėti nuo pagrindinio ekrano kuriame naudotojas galės pasirinkti ką nori daryti ar sukurti naują rezervaciją, ar peržiūrėti esamas, ar pakeisti numatytąjį ofisą. Pridėti naują ekraną galima kūrimo aplinkoje paspaudus mygtuką *New screen*. Norint pradėti pildyti ekraną, reikėtų pasirinkti valdiklius (angl. *Controls*) kuriuos norima pridėti. Pavyzdžiui, teksto atvaizdavimui naudojamas valdiklis *Label*, teksto įvesčiai *Text box*. Taip pat galima pridėti formą duomenų įrašams kurti, redaguoti ar peržiūrėti. Norint atvaizduoti daugiau įrašų galima pridėti galeriją, visi galimi valdikliai (žr. 4.5 pav.).

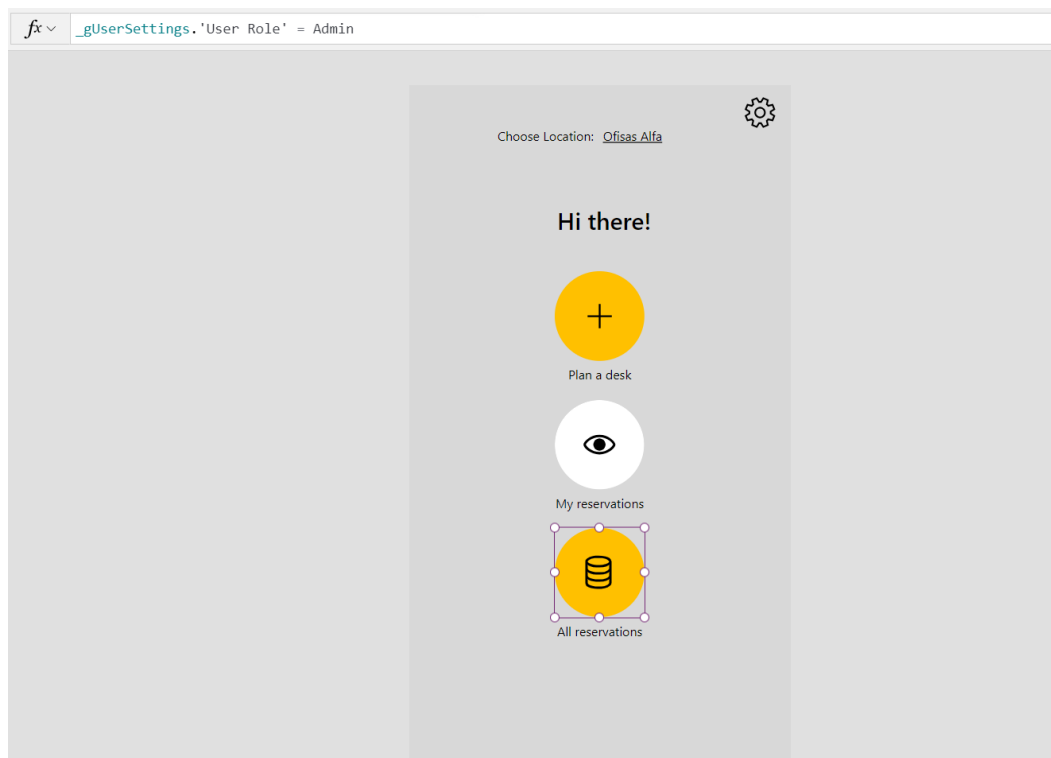


4.5 pav. Valdiklių pasirinkimas.

4.3.1. Pagrindinis ekranas

Pagrindiniam ekranui sukurti informaciniai tekstai ir mygtukai kurie naudotoją perkelia į kitus ekranus. Tam buvo panaudoti *Label*, *Button* ir *Icon* valdikliai. *Button* ir *Icon* valdikliai, savybėje *OnSelect* turi funkciją *Navigate_*(„Ekranas pavadinimas“), kuri naudotoją perkelia į kitą ekraną.

Tam, jog patekti į administratorių ekraną, pagrindiniame ekrane buvo pridėtas nustatymų mygtukas, kuris yra matomas tik administratoriams. Patikrinimui ar naudotojas yra administratorius, programėlės paleidimo (angl. *OnStart*) savybėje sukurtas kintamasis pavadinimu *_gUserSettings* kurio reikšmė yra programėlės naudotojo įrašas naudotojų (angl. *Users*) lentelėje. Mygtuko matomas (angl. *Visible*) savybėje parašytas kodas (žr. 4.6 pav.) kuris patikrina ar *_gUserSettings* kintamajame naudotojo rolė yra lygi *Admin*. Paspaudus ikoną naudotojas perkeliamas į administratorių ekraną. Tokiu būdu užtikrinama, jog ekranas būtų pasiekiamas tik administratoriams. Taip pat pagrindiniame ekrane buvo pridėtas mygtukas kurį paspaudus naudotojas perkeliamas į visų rezervacijų sąrašą. Visų rezervacijų sąrašą pasiekti gali tik administratorius. Taigi lygiai taip pat kaip ir ikonai kuria pasiekiamas administratorių sąrašas, buvo nustatyta, jog tik administratoriai mato mygtuką kuris naviguoja į visų rezervacijų sąrašą (žr. 4.6 pav.).

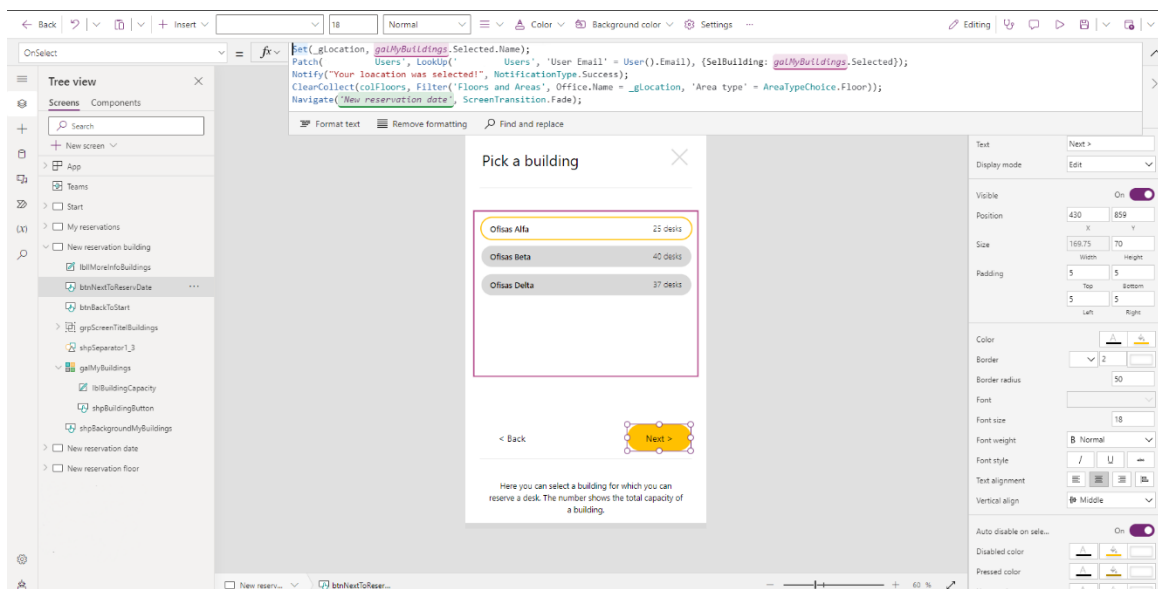


4.6 pav. Pagrindinis ekranas

4.3.2. Numatytojo ofiso pasirinkimo ekranas

Sekantis ekranas būtų numatytojo ofiso pasirinkimas. Jame pridėta galerija kuri atvaizduoja įrašus iš *Buildings* lentelės. Tam įgyvendinti reikėjo iš valdiklių pridėti galeriją ir jos savybėje *Items* įrašyti lentelės pavadinimą. Taip pat reikėjo pridėti mygtukus kurie perkelia vartotoją atgal į buvusį ekraną, arba išsaugo numatytąjį ofisą ir nusiunčia toliau į rezervacijos datos pasirinkimą. Pridėjus šiuos mygtukus, norint perkelti naudotoją atgal, į *OnSelect* savybę įrašyta funkciją *Back()*. Norint išsaugoti ofiso pasirinkimą ir tęsti rezervaciją, (4.7 pav.). Pirmoji kodo eilutė

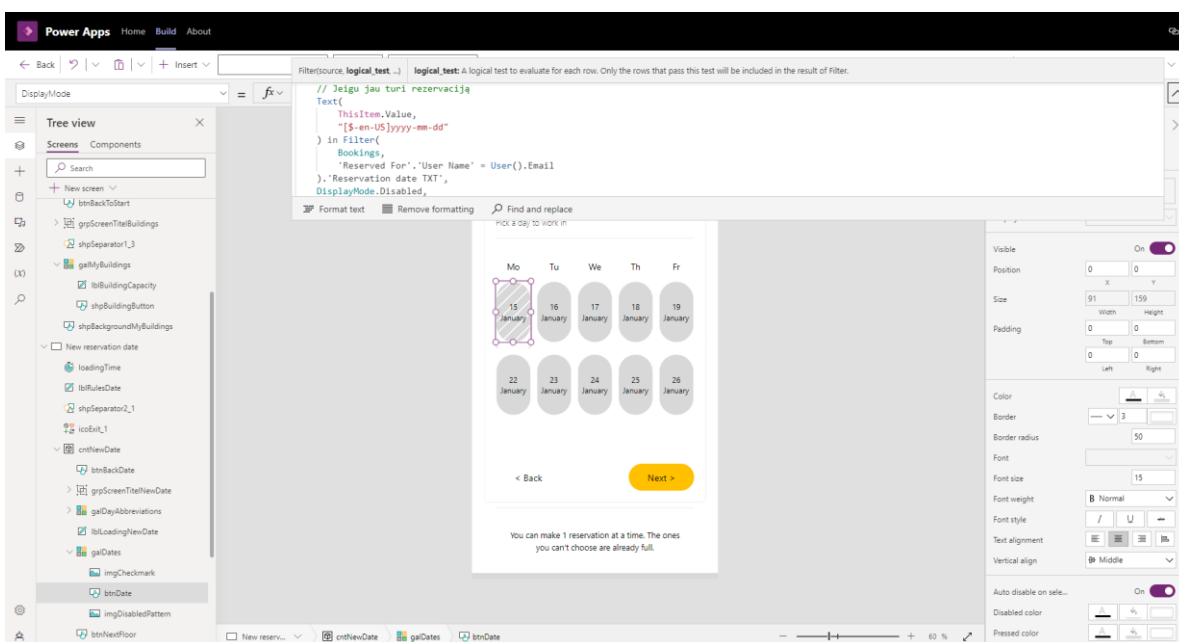
išsaugo ofiso pasirinkimą iš galerijos į globalų kintamąjį. Antra eilutė naudotojų lentelėje įrašo numatytojo ofiso pasirinkimą. Trečia eilutė informuoja naudotoją apie sėkmingai atnaujina numatytąjį pasirinkimą. Ketvirta eilutė iš *Floors and Areas* lentelės atranka tuos aukštus ir teritorijas kurie priklauso pasirinktam ofisui ir išsaugo kolekcijoje (programėlės viduje) kuri bus naudojama pasirinkimui vėlesniuose rezervacijos žingsniuose. Ir paskutinė eilutė perkelia naudotoją į ekraną kuriame reikia pasirinkti rezervacijos datą.



4.7 pav. Numatytojo ofiso pasirinkimas.

4.3.3. Rezervacijos datos pasirinkimo ekranas

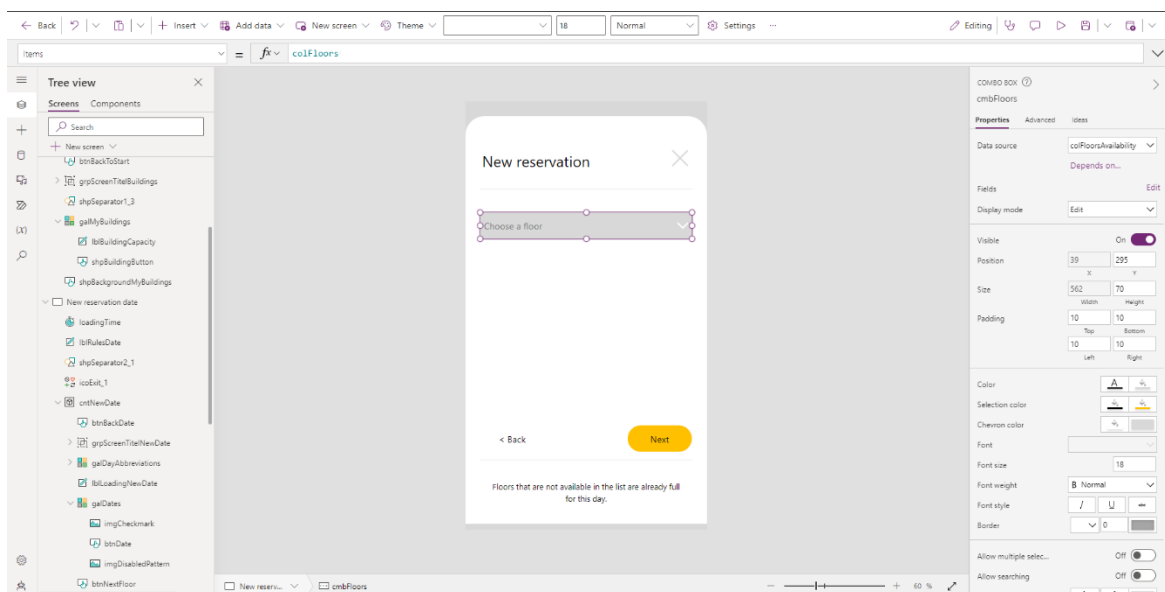
Toliau seka rezervacijos datos pasirinkimas. Pirmiausia reikia gauti visas šios savaitės ir ateinančios savaitės darbo dienų datas. Tai reikėjo padaryti sukuriant kolekciją programėlės paleidimo metu. Datų atvaizdavimui pasinaudota galerija ir jos *Items* savybėje įrašytas datų kolekcijos pavadinimas. Reikėtų nepamiršti neleisti pasirinkti datas, jei rezervacija tą dieną jau egzistuoja (4.8 pav.). Pasirinkus datą ją išsaugoti kaip kintamąjį.



4.8 pav. Datos pasirinkimas.

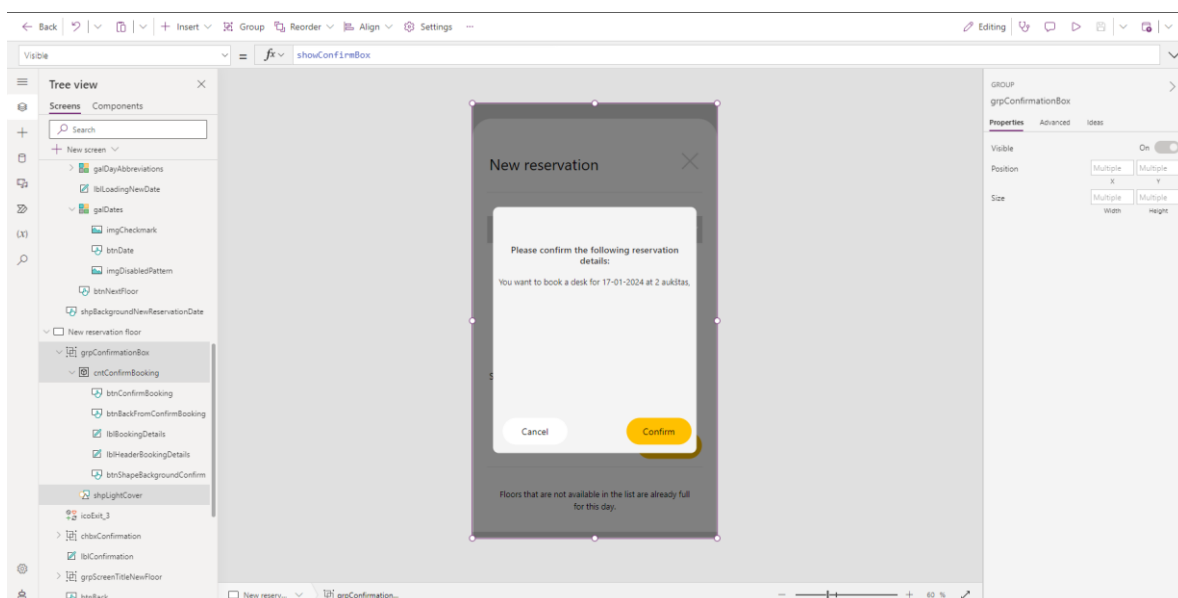
4.3.4. Aukšto arba teritorijos pasirinkimo ekranas

Kitas ekranas skirtas aukšto arba teritorijos pasirinkimui. Tam reikėjo panaudoti *Combo box*, valdiklį kuris yra pasirinkimo įvestis. Jame rodomi aukštai ir teritorijos iš kolekcijos *colFloors*, (4.9 pav.), kuri yra surenkama po numatytojo ofiso pasirinkimo arba pagal jį paleidžiant programėlę.



4.9 pav. Aukšto arba teritorijos pasirinkimas.

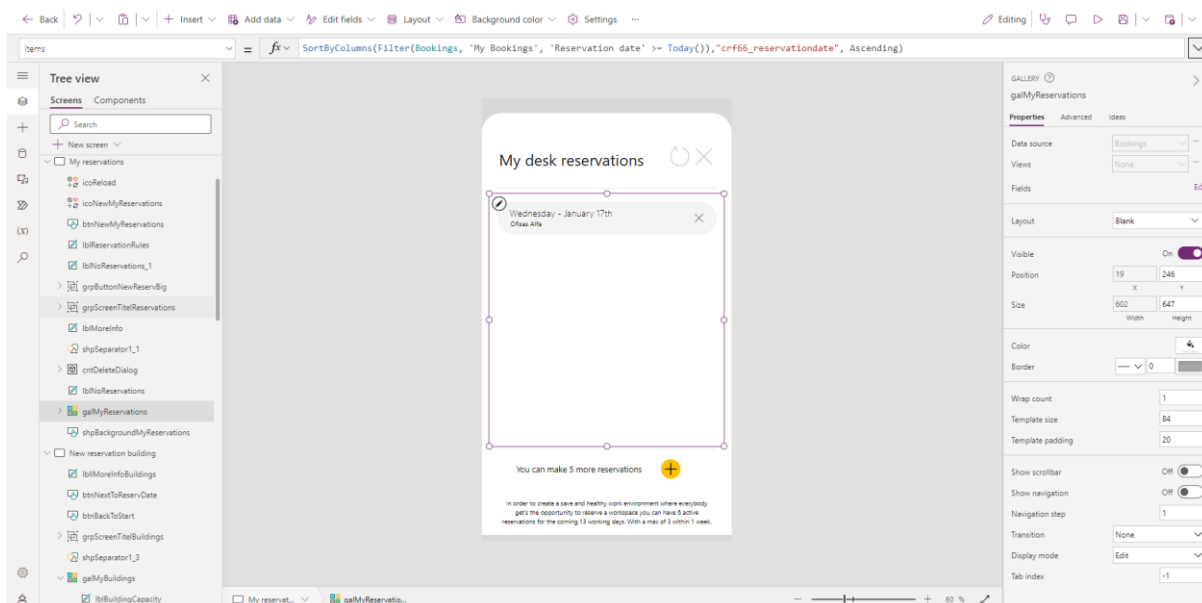
Taip pat galima pridėti iššokantį langą kuris prašo rezervacijos patvirtinimo (4.10 pav.). Paspaudus mygtuką *Confirm* duomenys išsiunčiami į *Bookings* lentelę. Eilutė *Defaults(Bookings)* reiškia, jog kuriamas naujas įrašas lentelėje, kadangi su *Patch()* funkcija taip pat galima atnaujinti įrašus. Po patvirtinimo naudotojas bus perkeltas į pagrindinį ekraną ir bus informuotas apie sėkmingą rezervaciją.



4.10 pav. Rezervacijos patvirtinimas

4.3.5. Naudotojo rezervacijų ekranas

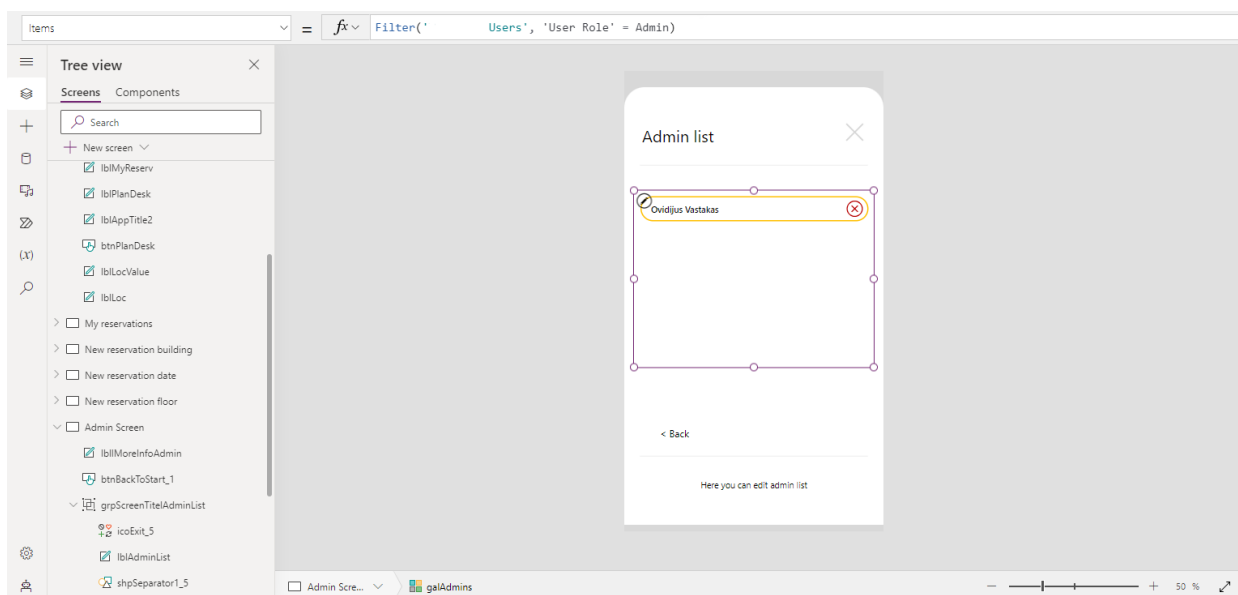
Ir paskutinis ekranas, kuriame matomos naudotojo rezervacijos. Tam taip pat užtenka galerijos, kuri iš *Bookings* lentelės atrenka naudotojui priklausančias rezervacijas, naujesnes nei vakar (4.11 pav.).



4.11 pav. Naudotojo rezervacijos.

4.3.6. Administratorių sąrašas

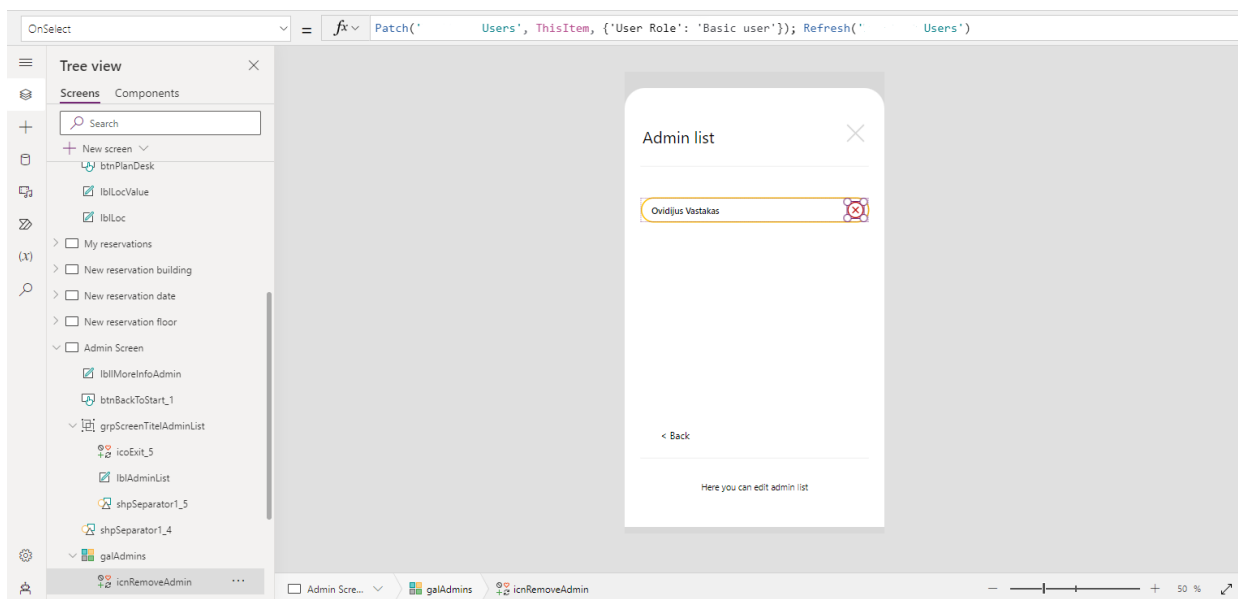
Administratorius gali matyti visų administratorių sąrašą, bei jį redaguoti (pridėti naujus arba pašalinti esamus administratorius). Tam buvo sukurtas naujas ekranas. Iš ofisų pasirinkimo ekrano nukopijuoti visi valdikliai (angl. *Controls*) ir pakeisti tekstai. Taip pat galerijos *Items* savybėje pakeista formulė (žr. 4.12 pav.) kuri iš naudotojų sąrašo paima tuos įrašus, kuriuose rolės reikšmė yra lygi *Admin*.



4.12 pav. Administratorių sąrašas.

Administratorių iš sąrašo galima pašalinti paspaudus raudoną mygtuką šalia administratoriaus vardo ir pavardės įrašo. Savybėje paspaudus (angl. *OnSelect*) parašytas kodas,

kuris naudotojų lentelėje suranda šios galerijos eilutės įrašą ir pakeičia rolę į *Basic user*. Taip pat iš naujo užkrauna lentelės duomenis. Tokiu būdą pašalintas administratorius iš karto dingsta ir iš galerijos (žr. 4.13 pav.).

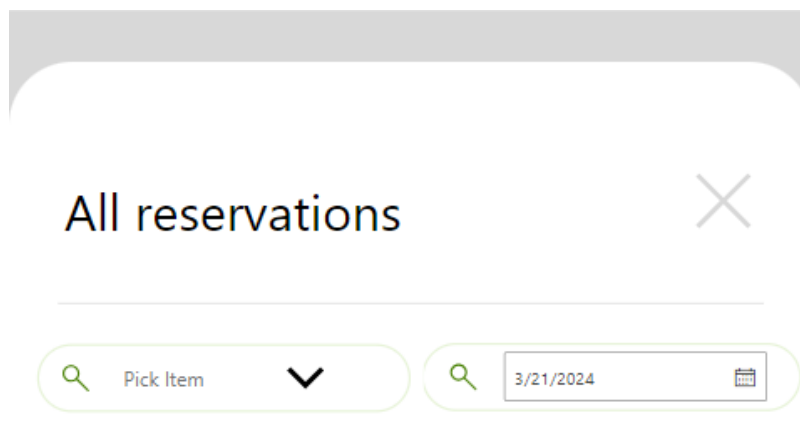


4.13 pav. Administratoriaus pašalinimas.

4.3.7. Visų rezervacijų apžvalga

Administratorius gali matyti visų rezervacijų sąrašą, bei jį filtruoti pagal datą ir rezervaciją atlikusį darbuotoją. Iš pradžių buvo nukopijuotas administratoriaus ekranas, kadangi jis jau turi dizainą pritaikytą kaip ir visos programėlės ir galeriją, taigi beliko tik pakoreguoti galeriją ir pridėti įvesties laukelius filtravimui.

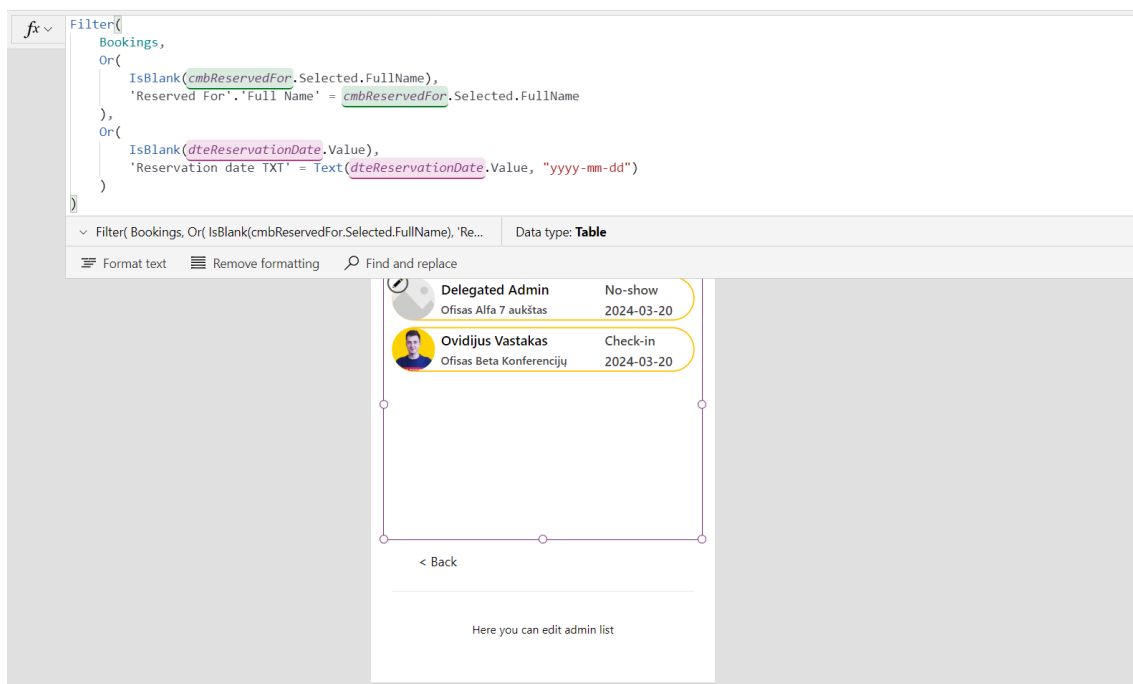
Iš pradžių buvo pridėti įvesties laukeliai filtravimui. Filtravimui pagal rezervacijos savininką panaudotas pasirinkimo (angl. *Combo box*) valdiklis. Galimi pasirinkimai šiam valdikliui – visi naudotojai. Tai yra, visi įrašai iš naudotojų lentelės rodantys vardą ir pavardę pasirinkime. Viskas labai paprasta, šio valdiklio duomenų (angl. *Items*) savybėje nurodytas naudotojų lentelės pavadinimas. Filtravimui pagal rezervacijos datą buvo pridėtas datos pasirinkimo (angl. *Date picker*) valdiklis (žr. 4.14 pav.).



4.14 pav. Rezervacijų sąrašo filtravimo įvesties laukai.

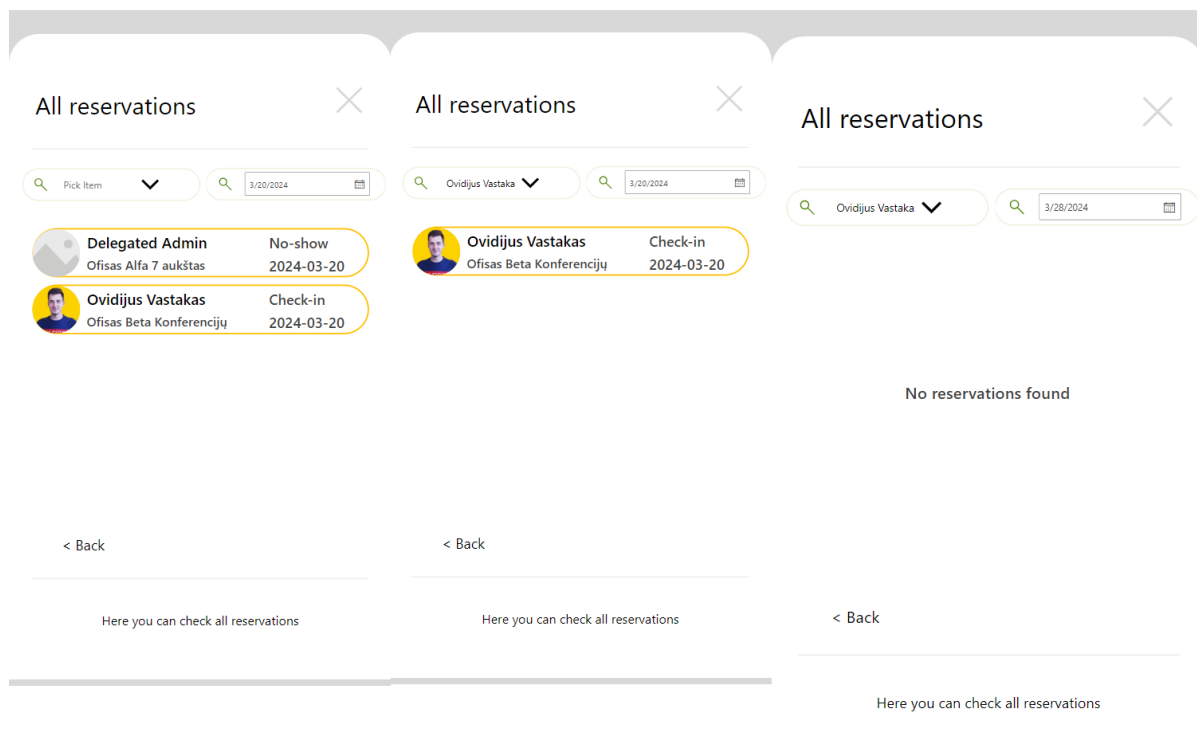
Tam, jog šie filtrai turėtų įtakos galerijos įrašams, buvo parašytas kodas kuris filtruoja rezervacijos lentelės įrašus pagal filtrų valdikliu pasirinktas reikšmes (žr. 4.15 pav.). Kode

Or(IsBlank) reiškia, jog filtro įvestyje nėra jokios reikšmės, filtravimas netaikomas. Kitu atveju paimami įrašai iš rezervacijų lentelės su atitinkamomis reikšmėmis nurodytame lauke. Kuriant programėlę *Microsoft Teams* aplinkoje valdikliai gali šiek tiek skirtis nuo kūrimo *Power Apps Maker* portale. Vienas iš skirtumų datos pasirinkimo valdiklis. *Microsoft Teams* aplinkoje nėra įmanoma keisti jo spalvų ir kitų savybių iš dizaino pusės. Taip pat negalima nurodyti numatytojo datos formato, kas buvo minusas, kadangi filtravimo kode datą reikėjo paversti į tekstą ir pakeisti jos formatą.



4.15 pav. Rezervacijų galerijos filtravimas.

Toliau buvo pakoreguota galerija. Jos viduje pridėti laukai rezervacijos duomenims. Pridėti teksto (angl. *Label*) valdikliai rezervacijos ofisui, aukštui, datai ir statusui atvaizduoti. Taip pat pridėtas nuotraukos (angl. *Image*) valdiklis kuris parodo rezervacijos savininko paskyros nuotrauką, lengvesniam identifikavimui (žr. 4.16 pav.). Buvo sukurti keli testiniai rezervacijų įrašai su skirtingais rezervacijų savininkais ir datomis. Filtravimas buvo ištestuotas, pasirinkta specifinė data ir atvaizduotos visos tos rezervacijos. Taip pat ištestuotas atvejis kai filtruojama pagal datą ir rezervacijos savininką. Pridėtas tekstas, kuomet pagal pasirinktas filtravimo opcijas rezervacijų nerasta, logika paprasta, teksto matomumo (angl. *Visible*) savybėje, parašytas kodas jeigu galerija tuščia (angl. *IsEmpty*) tuomet matomumo savybės reikšmė – tiesa (angl. *True*) ir tekstas parodomas. Visi išvardinti atvejai pavaizduoti 4.16 paveiksle.

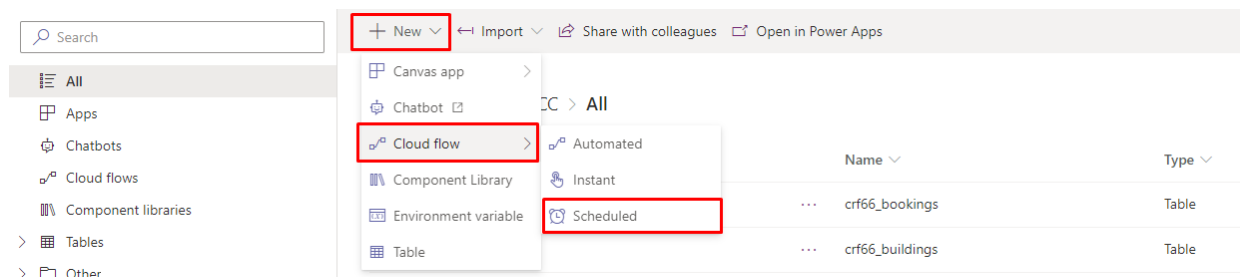


4.16 pav. Rezervacijų galerija.

4.4. Patvirtinimo žingsnis

Vartotojas rezervacijos dieną, devintą valandą ryto turi gauti žinutę į *Microsoft Teams* su prašymu patvirtinti arba atšaukti rezervaciją. Taip pat buvo minėta, jog šio reikalavimo įgyvendinimui reikėtų kiekvieną rytą gauti sąrašą rezervacijų ir jas atlikusiems darbuotojams išsiųsti prašymą patvirtinti rezervaciją. Tam reikėtų *Power Automate* įrankio pagalba sukurti suplanuotą veiksmą (angl. *Scheduled flow*).

Scheduled flow (toliau – *flow*) sukurti galima toje pačioje aplinkoje kurioje pridėtos ir sukurtos lentelės, bei programėlė. Taigi kūrimo aplinkoje buvo sukurtas *scheduled flow* (žr. 4.17 pav.).



4.17 pav. *Scheduled flow* pridėjimas kūrimo aplinkoje.

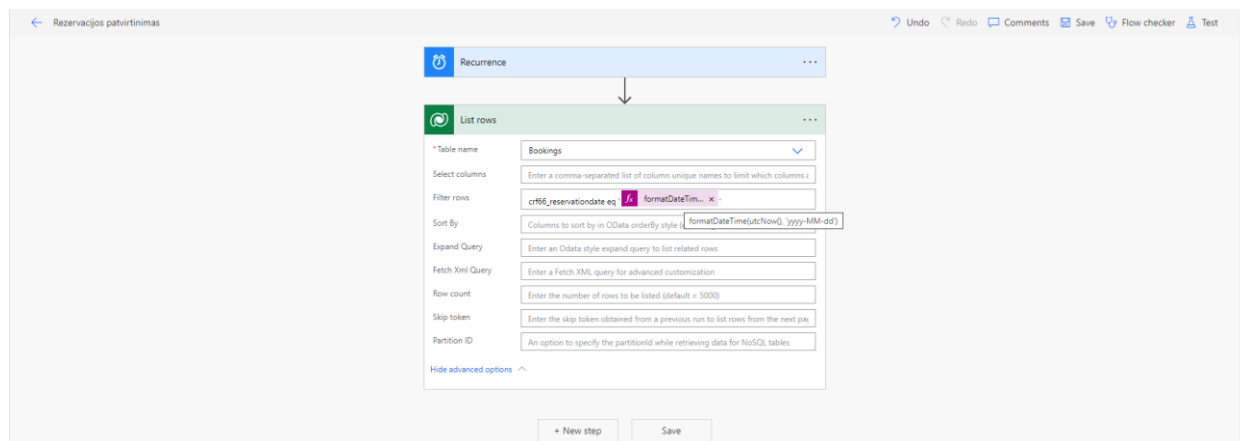
Flow viduje pirmiausia buvo pridėtas veiksmas (angl. *action*) kuris iš duomenų bazės pasirinktos lentelės gali gauti visus įrašus. Sistemoje naudojama *Dataverse* duomenų bazė, taigi reikėjo susirasti *Dataverse* veiksmų sąrašą, kuris yra *Power Automate Dataverse* dokumentacijoje [16]. *Flow* viduje pridėtas gauti įrašus (angl. *List rows*) veiksmas. Pasirinkta rezervacijų lentelė kaip duomenų šaltinis. Tam, jog būtų gautos tik šios dienos rezervacijos, reikia išfiltruoti gaunamus įrašus. Veiksmo viduje yra tam skirtas filtravimo užklausa (angl. *Filter query*) laukas. Jo viduje įrašytas toks kodas:

```
crf66_reservationdate eq '@{formatDateTime(utcNow(), 'yyyy-MM-dd')}'
```

Paiškinimas:

- crf66_reservationdate – tai yra rezervacijos datos lauko rezervacijų lentelėje loginis pavadinimas.
- eq – nurodo lygybę (angl. *equal*) tarp nurodyto lauko ir toliau nurodytos reikšmės.
- '@{formatDateTime(utcNow(), 'yyyy-MM-dd')}' – yra formulė kuri paima šios dienos datą.

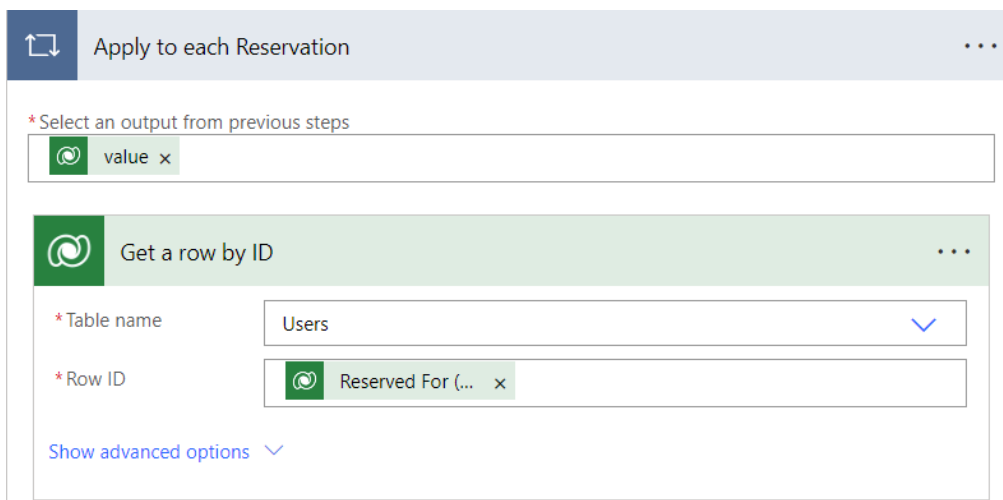
Taip bus gaunami tik *flow* paleidimo metu esantys rezervacijų įrašai (žr. 4.18 pav.).



4.18 pav. Rezervacijų įrašų gavimas.

Sekantis žingsnis turi būti patvirtinimo žinutės išsiuntimas. Pagal reikalavimą, patvirtinimo užklausa turėtų būti išsiųsta kaip *Microsoft Teams* žinutė.

Rezervacijų lentelėje nėra saugomas rezervacijos savininko elektroninis paštas, saugomas tik identifikacinis numeris įrašo naudotojų lentelėje. Taigi, kad išsiųsti patvirtinimo žinutę, reikia gauti rezervacijos savininko elektroninį paštą iš naudotojų (angl. *Users*) lentelės. Tam buvo pridėtas gauti įrašą pagal ID (angl. *Get a row by ID*) veiksmas. Pasirinkta naudotojų lentelė, bei *Row ID* nurodyta *Reserved for* lauko reikšmė paimta iš pradžioje sukurtos *List rows* veiksmo. Nurodžius šią reikšmę, buvo automatiškai sukurtas – pritaikyti kiekvienai rezervacijai (angl. *Apply to Each Reservation*) veiksmas (žr. 4.19 pav.). Taip atsitiko todėl, nes *List rows* gražina masyvą esamos dienos rezervacijų įrašų. Ir tuomet kiekvienai rezervacijai gaunami duomenys apie rezervacijos savininką iš naudotojų lentelės.



4.19 pav. Rezervacijos savininkų duomenų gavimas.

Turint visus reikiamus duomenis apie rezervaciją ir rezervacijos savininką, buvo prieita prie patvirtinimo užklauskos siuntimo. Kadangi reikalavimas yra patvirtinimą siųsti *Microsoft Teams* žinute, buvo surasta *Microsoft Teams connector* [17] dokumentacija. Joje pavyko rasti visus *Microsoft Teams*, veiksmus kuriuos galima pridėti į *flow*. Pagal aprašymą labiausiai tiko veiksmas nurodytam asmeniui nusiunčiantis žinutę su galimais pasirinkimais atsakymui į žinutę (angl. *Post a choice of options as the Flow bot to a user*). Taigi buvo pridėtas ką tik minėtas veiksmas (žr. 4.20 pav.).

The screenshot shows the configuration for the 'Post a choice of options as the Flow bot to a user' action. It includes a list of options (Yes, No), a recipient selection (Primary Email), a message template with dynamic content placeholders, and a headline. The message template is: 'You have a reservation today ({ utcNow(...) }) at { items(...) } . Please confirm your reservation.' The headline is: 'Are you coming to the office today?'.

4.20 pav. Rezervacijos patvirtinimo žinutės siuntimas.

Veiksmo viduje nurodytas žinutės gavėjas (angl. *Recipient*), tai yra, gavėjo elektroninis paštas, paimtas iš *Get a row by ID* veiksmo. Kadangi įmonė tarpautinė žinutės tekstas ir pasirinkimai parašyti anglų kalba. Pasirinkimai taip arba ne. Žinutės viduje pridėtos funkcijos:

- `utcNow('yyyy-MM-dd')` – šios dienos data nurodytų formatu.
- `items('Apply_to_each_Reservation')?['_crf66_office_value@OData.Community.Display.V1.FormattedValue']` – iš šios dienos rezervacijų masyvo įrašo gaunamas ofiso pavadinimas. Principas toks pat kaip rezervacijos savininko duomenų gavimo, tačiau nereikia kurti naujo veiksmo ofiso pavadinimui gauti iš pastatų (angl. *Buildings*) lentelės. Taip yra todėl, nes ofiso pavadinimas pastatų lentelėje yra pirminis laukas (angl. *Primary column*), todėl jo reikšmę galima gauti tiesiai iš rezervacijų lentelės.
- `items('Apply_to_each_Reservation')?['_crf66_area_value@OData.Community.Display.V1.FormattedValue']` – lygiai tas pats kaip ir ofiso pavadinimo gavimas. Tik čia gaunamas aukšto pavadinimas iš aukštų (angl. *Floors and Areas*) lentelės.

Šis veiksmas užbaigiamas ir einama prie kito tik tuomet, kai vartotojas atsako į žinutę paspausdamas ant vieno iš pasirinkimų žinutėje. Atsakymo patikrinimui buvo pridėtas sąlygos (angl. *Condition*) veiksmas (žr. 4.21 pav.). Jo viduje įrašyta sąlyga – pasirinktas atsakymas yra lygus taip. Jeigu naudotojo pasirinkimas yra taip tuomet rezervacijos statusas turėtų būti pakeistas į užregistruotą (angl. *Check-in*), reiškia, jog naudotojas atvyko / atvyks į užrezervuotą darbo vietą. Taigi, jei nurodyta sąlyga yra patenkinta (angl. *If yes*) sekcijoje pridėtas veiksmas įrašo atnaujinimui (angl. *Update a row*). Atnaujinamas šios dienos rezervacijų einašasis (angl. *Apply*

to each Reservation) masyvo įrašas. Lygiai taip pat, jeigu sąlyga netenkinama, atnaujinamas rezervacijos įrašas su įrašu, jog rezervacijos savininkas nepasirodė (angl. *No-show*).

The screenshot displays a Power Automate flow configuration. At the top, a 'Condition' step is set to 'selected...' is equal to 'Yes'. Below this, there are two parallel paths based on the condition's result.

If yes (green path): An 'Update a row' action for the 'Bookings' table. The 'Row ID' is selected from a dropdown. The 'Status' is set to 'Check-in'.

If no (red path): An 'Update a row 2' action for the 'Bookings' table. The 'Row ID' is selected from a dropdown. The 'Status' is set to 'No-show'.

4.21 pav. Rezervacijos patvirtinimo atsakymas.

Taip pat reikėtų nepamiršti, jog sistema gali nesulaukti atsakymo iš naudotojo, o pagal *Power Automate* taisyklės *flow* gali būti aktyvus trisdešimt dienų. Tai reiškia, jog nedarant jokių pakeitimų naudotojas gali atsakyti į žinutę po kelių dienų, kas neatitiktų reikalavimo, jog rezervacijos patvirtinimo žinutė turi būti aktyvi tris valandas nuo jos išsiuntimo ir, jeigu vartotojas neatsako per nurodytą laiką, rezervacija atšaukiama.

Būtent tokiems atvejams veiksmo nustatymuose galima nustatyti laiką kurį veiksmas yra aktyvus ir kuriam pasibaigus pereinama prie kito žingsnio (angl. *Timeout*). Microsoft Teams žinutės siuntimo veiksmo buvo nustatytas laikas iki veiksmo atšaukimo (žr. 4.22 pav.). Laikas turi būti nurodytas ISO 8601 formatu. PT3H reiškia, jog veiksmas bus neaktyvus po trijų valandų.

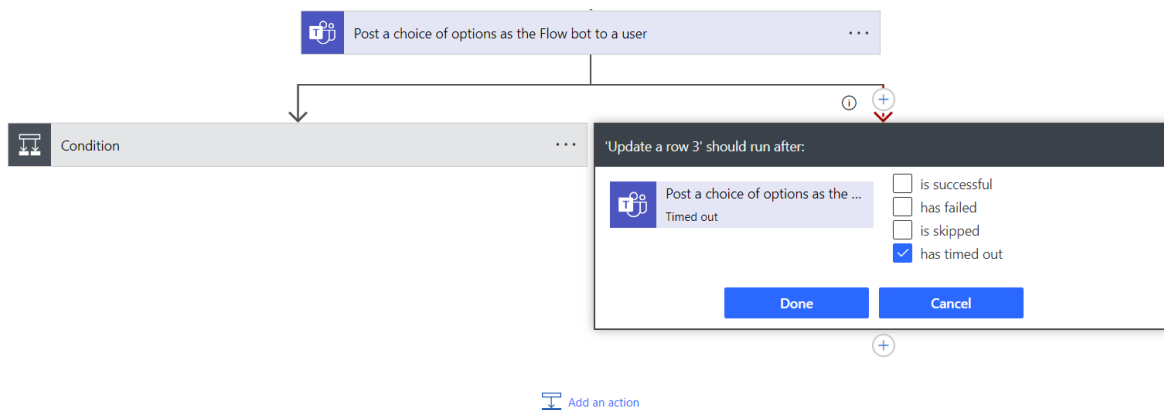
The screenshot shows the settings for the 'Post a choice of options as the Flow bot to a user' action. The settings are as follows:

- Secure Inputs:** Off
- Secure Outputs:** Off
- Timeout:** Duration is set to 'PT3H'.
- Retry Policy:** Type is set to 'Default'.
- Tracked Properties:** A table with 'Key' and 'Value' columns.

Buttons for 'Done' and 'Cancel' are at the bottom.

4.22 pav. Veiksmo atšaukimas.

Tačiau vien tik to neužtenka, kadangi atšaukus veiksmą bus pereinama tiesiai prie atsakymo tikrinimo. Kadangi atsakymas negautas, nei viena sąlyga nebus patenkinta ir *flow* tiesiog pasibaigs, be jokio atsakymo. Dėl to buvo pridėtas rezervacijos įrašo atnaujinimo veiksmas, toks pat kaip esant neigiamam atsakymui, kuris bus vykdomas paraleliai (angl. *Add a parallel branch*) atsakymo tikrinimo veiksmo. Tačiau, jo nustatymuose pridėta sąlyga, jog jis bus vykdomas tik tuomet kai (angl. *Configure run after*) prieš tai buvęs veiksmas, šiuo atveju *Microsoft Teams* žinutės siuntimas, yra atšauktas (angl. *Has time out*), (žr. 4.23 pav.).



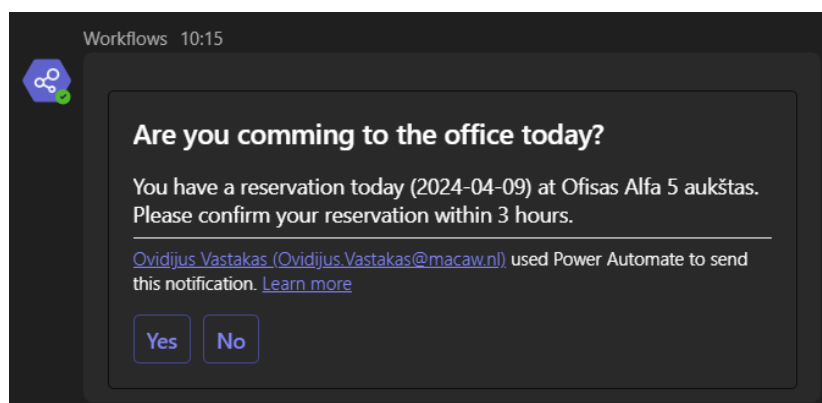
4.23 pav. Rezervacijos statuso atnaujinimas, kuomet nesulaukiama atsakymo į žinutę.

Flow ištestavimui buvo sukurtas duomenų įrašas rezervacijų lentelėje (žr. 4.24 pav.) su šios 2024 metų, Kovo 18-os dienos rezervacijos data.

Reservation ID * ↑ ↓	Reserved For *	Office *	Area *	Reservatio... *	Status *
1001	Ovidijus Vastakas	Ofisas Alfa	5 aukštas	3/18/2024	Submitted
	Select lookup	Select lookup	Select lookup	Enter or pick date	Select option

4.24 pav. Rezervacijos įrašas patvirtinimo testavimui.

Paleidus testavimo *flow* buvo gauta žinutė apie anksčiau minėtos dienos rezervaciją ir prašoma ją patvirtinti per ateinančias tris valandas (žr. 4.25 pav.).



4.25 pav. Rezervacijos patvirtinimo žinutė.

Pasirinkus atsakymą *Yes*, *flow* buvo užbaigtas sėkmingai, be jokių klaidų (žr. 4.26 pav), taip pat rezervacijos statusas pakeistas į *Check-in* (žr. 4.27 pav.).



4.26 pav. Sėkmingai užbaigtas *flow*.

Reservation ID * ↑	Reserved For *	Office *	Area	Reservatio... *	Status *
1001	Ovidijus Vastakas	Ofisas Alfa	5 aukštas	3/18/2024	Check-in

4.27 pav. Sėkmingai pakeistas rezervacijos statusas.

Taip pat atlikti testai, kuomet naudotojas pasirenka, jog neatvyks į ofisą arba neatsako per tris valandas. Juose klaidų nerasta, rezervacijos statusas pakeičiamas į No-show. Po sėkmingų testų, buvo pervardinti visi veiksmas tam, jog atsidarius *flow* būtų aišku kas ką atlieka.

4.5. Pokalbių roboto kūrimas

4.5.1. Pokalbio pradžios konfigūracija

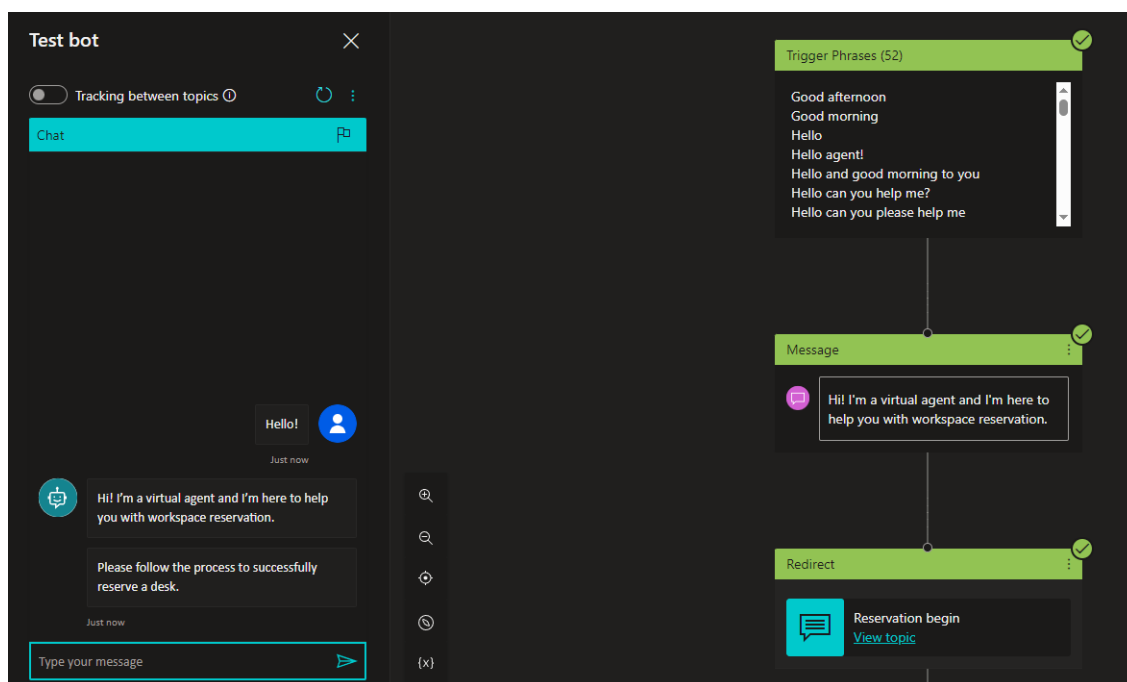
Kaip jau buvo minėta, pokalbio inicijavimo temos ir su ja susietų frazių kurti nereikia, nes tai yra padaroma sistemos, pokalbių roboto kūrimo metu. Galima tik pakoreguoti pasisveikinimo (angl. *Greeting*) temą. Kadangi pokalbių robotas skirtas būtent darbo vietos rezervacijos sistemai, buvo nuspręsta po pasisveikinimo naudotoją nukreipti tiesiai į rezervacijos proceso temą. Tokiu būdu naudotojas apribojamas nuo įvairių pašalinių įvesčių nesusijusių su darbo vietos rezervavimu. Tam reikėjo sukurti naują temą, tai galima padaryti temų skiltyje paspaudus mygtuką nauja tema (angl. *New topic*) ir kurti tuščią (angl. *From blank*). Tema buvo pavadinta – rezervacijos pradžia (angl. *Reservation begin*). Taip pat pradžiai pridėta dešimt aktyvavimo frazių anglų kalba:

- Reserve space
- Office reservation
- Book workspace
- Book a desk
- Desk reservation
- Workspace reservation
- Reserve
- Book
- I would like to book workspace
- Make a reservation

Šių frazių reikia tokiu atveju, jeigu naudotojas atėjęs iš karto įvestų užklausą rezervacijai, o ne pasisveikintų su robotu. Tolimesnėje eigoje testuojant robotą galima pridėti ir daugiau frazių, jeigu matoma, jog robotas netinkamai reguoja į frazes ir neparenka atitinkamos temos.

Sukūrus rezervacijos pradžios temą, reikėjo grįžti prie pasisveikinimo temos. Kaip buvo minėta naudotojas po pasisveikinimo turėtų būti nukreiptas iškart į rezervacijos temą. Taigi pasisveikinimo tema buvo pakoreguota ištrinant kelias nereikalingas žinutes ir pridėtas

svarbiausias žingsnis, pokalbio nukreipimas į kitą temą. Tam pasirinktas atitinkamas veiksmas – nukreipti į kitą temą (angl. *Redirect to another topic*) ir nurodyta ką tik sukurta rezervacijos pradžios tema. Taigi galutinė pasisveikinimo temos struktūra pavaizduota 4.28 paveiksle.



4.28 pav. Pasisveikinimo temos galutinis vaizdas.

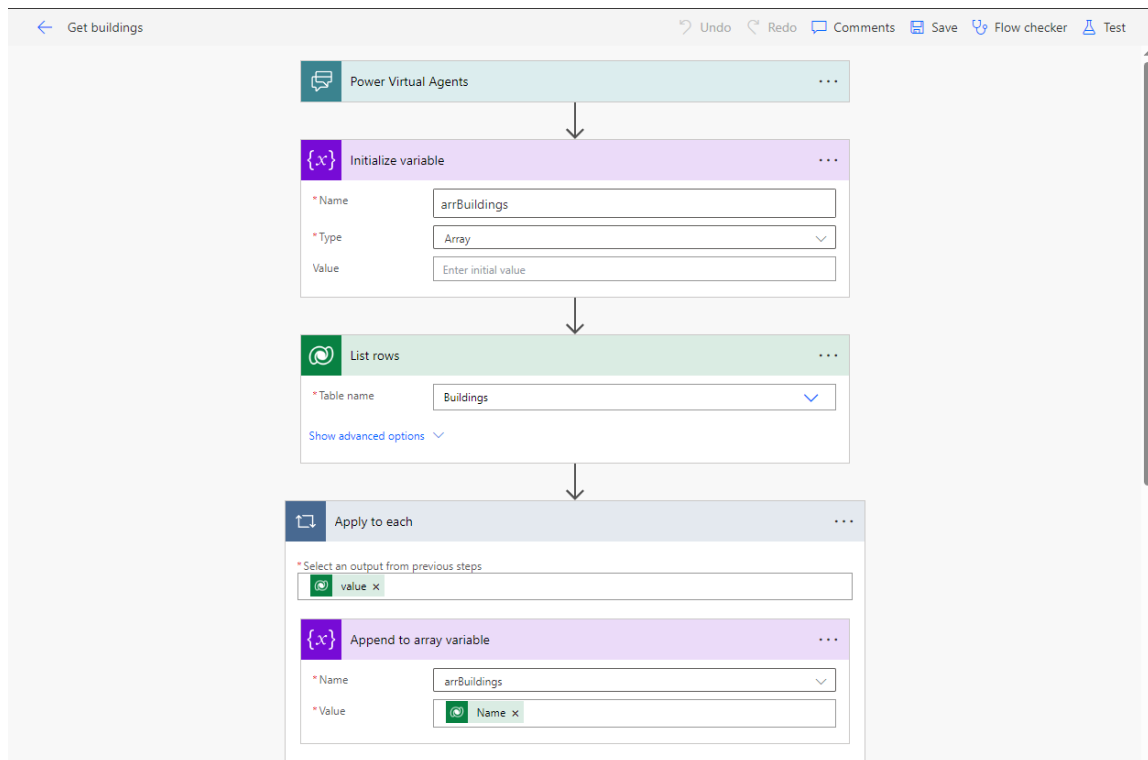
4.5.2. Ofiso pasirinkimas

Kaip buvo minėta, rezervacijos atlikti laisvu tekstu nėra įmanoma, todėl rezervacijos procesas, tai yra atliekami žingsniai ir jų eiga bendraujant su pokalbiu robotu, turi išlikti tokie patys kaip ir atliekant rezervaciją naudojantis programėle. Todėl pirmas žingsnis pokalbių robote naudotoją nukreipus į rezervacijos temą turėtų būti ofiso pasirinkimas. Galima pridėti veiksmą naudotojo įvesčiai ir nurodyti galimus ofisų pasirinkimus, bet taip ofisų sąrašą reikėtų atnaujinti dvejose vietose – *Microsoft Copilot Studio* ir *Dataverse for Teams* duomenų bazėje pastatų lentelėje. Geriausias variantas gauti įrašus iš pastatų lentelės duomenų bazėje ir juos pateikti kaip pasirinkimus. Kuriant pokalbių robotą, gauti duomenis tiesiogiai iš duomenų bazės nėra įmanoma. Būdas tai padaryti – per pokalbių robotą paleisti *Power Automate* procesą kuris gautu duomenis ir grąžintų pokalbių robotui. Tam yra skirtas iškviesti veiksmą (angl. *Call an action*) veiksmas kurį reikia pridėti į temą. Tuomet pasirenkamas naujas procesas (angl. *Flow*) ir *Power Automate* kūrimo aplinka atsidaro *Microsoft Copilot Studio* viduje, nereikia eiti niekur kitur, labai patogi integracija.

Pirmiausia buvo pridėta žinutė naudotojui, prašanti atlikti pokalbių roboto prašomus rezervacijos žingsnius, tam reikalingas, išsiųsti žinutę veiksmas. Sekantis žingsnis kaip ir buvo minėta, ofiso pasirinkimas. Sukurtas naujas procesas, kurio viduje atliekami šie veiksmas:

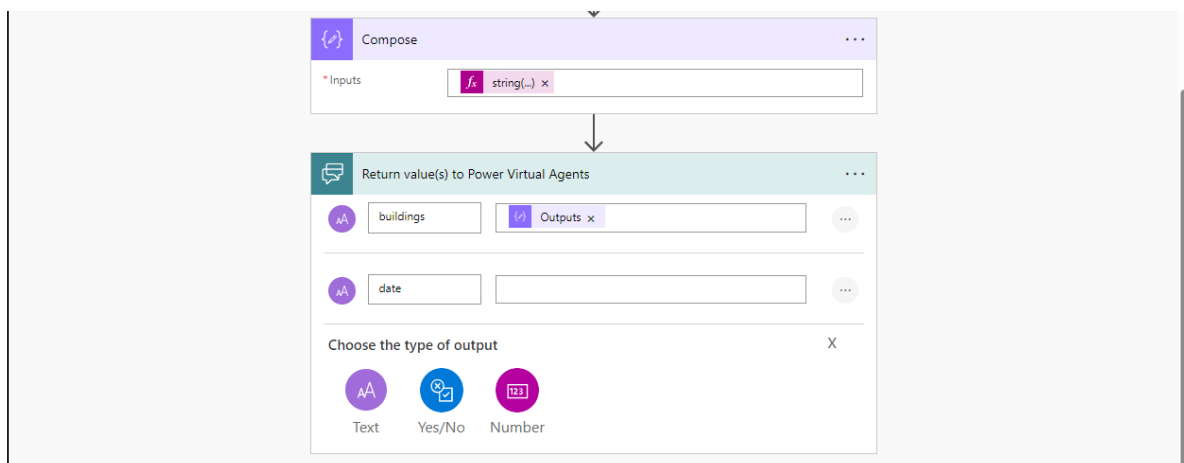
1. Kintamojo inicializacija (angl. *Initialize variable*) – sukuriamas masyvo tipo kintamasis, *arrBuildings*, kurio viduje bus saugomi visi galimi ofisų pasirinkimai (žr. 4.29 pav.).
2. Gauti duomenis (angl. *List rows*) veiksmas, kuris iš pastatų lentelės gauna visus duomenų įrašus (žr. 4.29 pav.).

3. Ciklas (angl. *Apply to each*), kuris kiekvienam duomenų lentelės įrašui atlieka veiksmą – pridėti į masyvo kintamąjį (angl. *Append to array variable*). Tokiu būdu į pastatų masyvą pridedami ofisų pavadinimai (žr. 4.29 pav.).



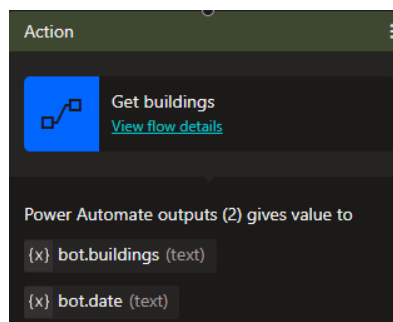
4.29 pav. Power Automate proceso pastatų įrašų gavimui pirmoji dalis.

4. Duomenų operacijos (angl. *Compose*) veiksmas, kurio viduje parašytas kodas – `string(variables('arrBuildings'))`, kuris nurodo kintamąjį *arrBuildings* paversti į teksto eilutės (angl. *String*) tipo kintamąjį. To reikia todėl, nes *Power Automate* procesas pokalbių robotui gali gražinti tik tekstines, taip arba ne, ir skaitines reikšmes (žr. 4.30 pav.).
5. Paskutinis veiksmas gražina reikšmes pokalbių robotui (angl. *Return value(s) to Power Virtual Agents*). Jo viduje nurodyta gražinti *buildings* reikšmę, kuri yra lygi paskutinio veiksmo rezultatui. Bei *date* be jokios reikšmės, nes tokiu atveju yra sukuriamas kintamasis pokalbių roboto viduje, kurio prireiks tolimesniems veiksmais (sukurti kintamojo, be jokios roboto išvesties, *Microsoft Teams* pokalbių roboto viduje nėra įmanoma) (žr. 4.30 pav.).



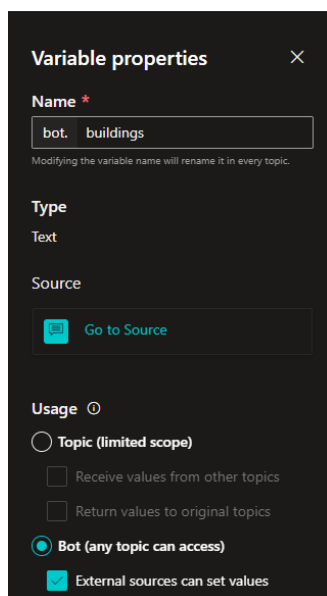
4.30 pav. Power Automate proceso pastatų įrašų gavimui antroji dalis.

Iš *Power Automate* gavus visus ofisų pasirinkimus ir juos išsaugojus *bot.buildings* (prie reikšmės pavadinimo iš *Power Automate* prisideda „bot.“ Nurodantis, jog tai pokalbių roboto kintamasis) kintamajame (žr. 4.31 pav.) reikėjo pakeisti šio kintamojo prieigą.



4.31 pav. Pokalbių roboto žingsnis paleidžiantis procesą ir gražinantis pastatų duomenų įrašus į *bot.buildings* kintamąjį.

Pakeisti prieigą reiškia, leisti visoms pokalbių robote esančios temos pasiekti ir keisti šio kintamojo reikšmę. Tai reikia atlikti dėl to, nes numatytieji kintamojo nustatymai yra tokie, jog kintamasis gali būti pasiekiamas tik temoje kurioje jis buvo sukurtas. Paspaudus ant *bot.buildings* kintamojo nurodytame po *Get buildings* veiksmu pakeisti kintamojo nustatymai (žr. 4.32 pav.).



4.32 pav. *bot.buildings* kintamojo nustatymai, tam jog jis būtų pasiekiamas visame pokalbių robote.

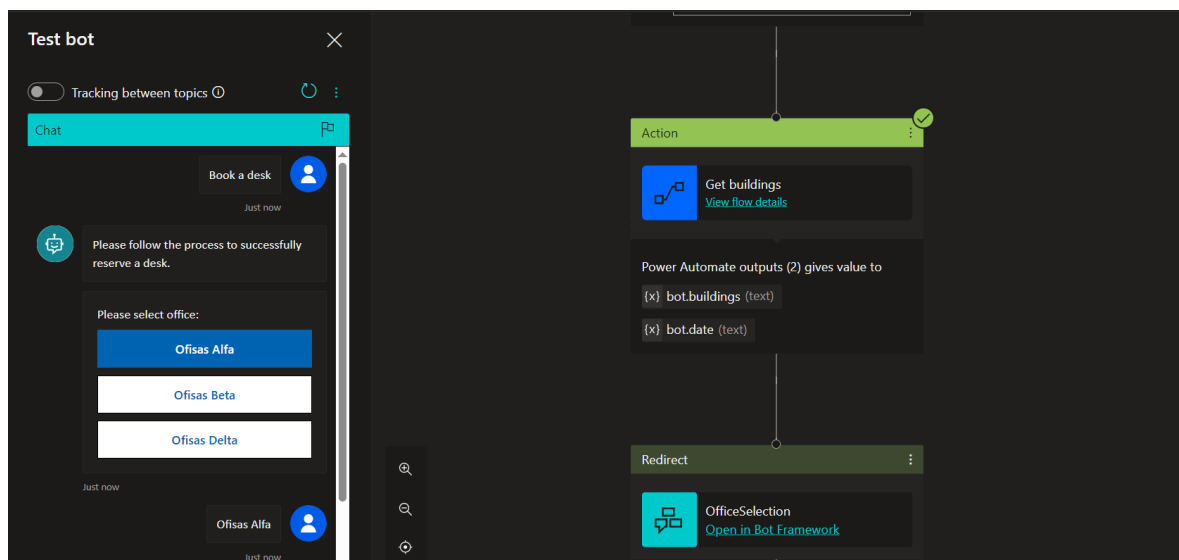
Sekantis veiksmas pokalbių robote – atvaizduoti ofisus kaip pasirinkimus naudotojui. Anksčiau buvo minėta, jog *Microsoft Copilot Studio* naudotojo įvesties veiksmo galimiams pasirinkimams išvesti negalima naudoti kintamojo, todėl sukurta *Bot Framework Composer* tema. Tai galima padaryti atliekant tokius pat žingsnius kaip kuriant įprastą temą, tik vietoj tuščios temos pasirinkti – atidaryti *Bot Framework Composer* (angl. *Open in Bot Framework Composer*). Atsidarius *Bot Framework Composer* (toliau –kompozitorius) kūrimo aplinkai, prisijungti su ta pačia *Microsoft* paskyra kuri naudojama *Microsoft Teams* kūrimo aplinkoje. Tokiu būdu kompozitorius atsisiunčia pokalbių roboto duomenis ir jį galima pradėti tobulinti. Sėkmingai atlikus visus šiuos veiksmus, kompozitoriaus viduje, pokalbių robotui buvo atlikti tokie žingsniai:

- Pridėtas naujas dialogas (angl. *Dialog*) ir pavadintas – *OfficeSelection*. Dialogas kompozitoriuje yra tas pats, kas tema, *Microsoft Copilot Studio*.

- Kompozitoriuje, pridėjus veiksmą nustatyti savybę (angl. *Set a property*), sukurtas kintamasis ofisų pasirinkimams saugoti – *dialog.offices*. Žodis – *dialog* čia nebūtinai, tai tik priedas prie kintamojo pavadinimo pagal numatytuosius nustatymus, skirtas kintamųjų grupavimui ar atskirimui skirtingiems dialogams. Tam, jog kompozitoriuje pasiekti kintamuosius iš *Microsoft Copilot Studio*, pirmiausia rašoma – *virtualagent*. ir tuomet kintamojo pavadinimas nurodytas kuriant pokalbių robotą *Microsoft Copilot Studio*. Kintamajam *dialog.offices* nustatyta reikšmė yra lygi – *virtualagents.buildings* kintamajam paverstam į *JSON* formatą, nes kaip buvo minėta visų ofisų masyvo kintamasis, iš *Power Automate* buvo grąžintas kaip teksto tipo kintamasis.
- Toliau seka naudotojo įvesties veiksmo pridėjimas. Pridėtas kelių pasirinkimų (angl. *Multi-choice*) veiksmas. Pridėjus automatiškai atsiranda veiksmas išsiunčiantis tekstą prieš pasirinkimą, tai yra, pokalbių roboto tekstas pasirinkimams apibūdinti ar panašiai. Pasirinkimų veiksmu nurodyta savybė, tai yra, kintamasis į kurį išsaugoma naudotojo įvestis. Šiuo atveju, tai – *dialog.office_choice*. Taip pat nurodytas pasirinkimų masyvas – anksčiau sukurtas kintamasis, *dialog.offices*.
- Paskutinis žingsnis, grąžinti pasirinktą reikšmę *Microsoft Copilot Studio* aplinkai. Tai padaryti galima tik pakeičiant joje jau sukurtą kintamojo reikšmę, kadangi kompozitoriaus viduje sukurtų kintamųjų naudoti negalime, o to prireiks kuriant rezervacijos įrašą. Paskutiniu veiksmu nustatyta kintamojo *virtualagents.buildings* reikšmė lygi *dialog.office_choice* kintamojo reikšmei, tai yra, ofiso pasirinkimui.

Visas ofiso pasirinkimo dialogas pateiktas 3 priede. Norint, jog kompozitoriuje sukurti dialogai atsirastų *Microsoft Copilot Studio* ir juos būtų galima ištestuoti, pokalbių robotą reikia atnaujinti paspaudus mygtuką – publikuoti (angl. *Publish*).

Ištestuotas visų ofisų gavimo *Power Automate* procesas ir ofisų pasirinkimas naudojant kompozitorių (žr 4.33 pav).



4.33 pav. Visų ofisų gavimo ir pasirinkimo testavimas pokalbių robote.

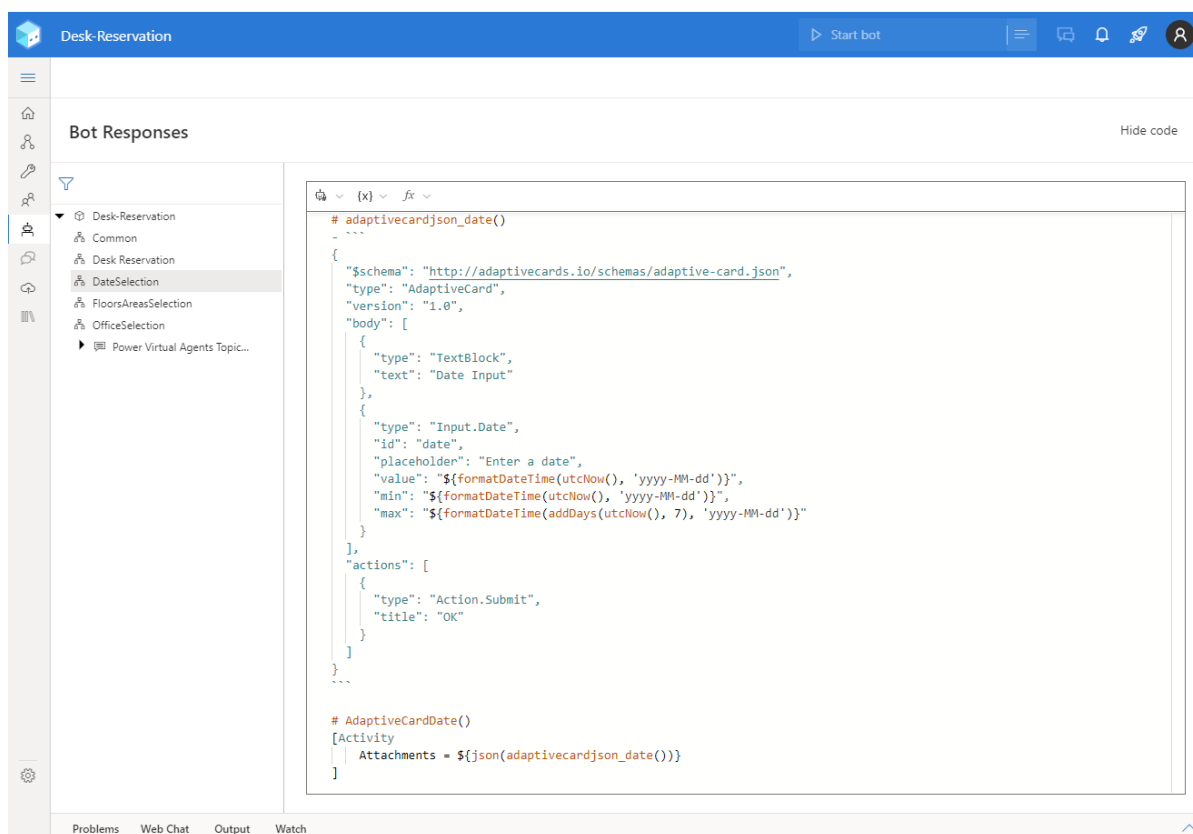
Pokalbių robotas sėkmingai gauna visus ofisus iš duomenų bazės ir leidžia naudotojui pasirinkti kuriame norima atlikti rezervaciją.

4.5.3. Datos pasirinkimas

Antras žingsnis pagal darbo vietos rezervavimo procesą – datos pasirinkimas. *Microsoft Copilot Studio* neturi jokio datos įvesties valdiklio, kaip kalendoriaus kuriuo būtų galima

interaktyviai pasirinkti rezervacijos datą. Galima naudoti paprastą naudotojo įvesties veiksmą ir nurodyti jau sukurtą datos ir laiko subjektą. Tai yra iš naudotojo įvesties pokalbių robotas atpažintų datos įvestį ir išsaugotų į kintamąjį. Naudotojas galėtų įvesti konkrečią datą arba parašyti žodį – rytoj ir pokalbių robotas išsaugotų rytojaus datą. Darbo vietos rezervavimo atžvilgiu šis variantas netinka, nes naudotojas galėtų įvesti bet kokią datą ir pokalbių robotas išsaugotų ją į kintamąjį ir atliktų rezervaciją. Įvestis neturi jokių apribojimų, todėl datą būtų galima validuoti tik *Power Automate* pagalba, kas būtų kiek per sudėtinga ir nepatogu naudotojui.

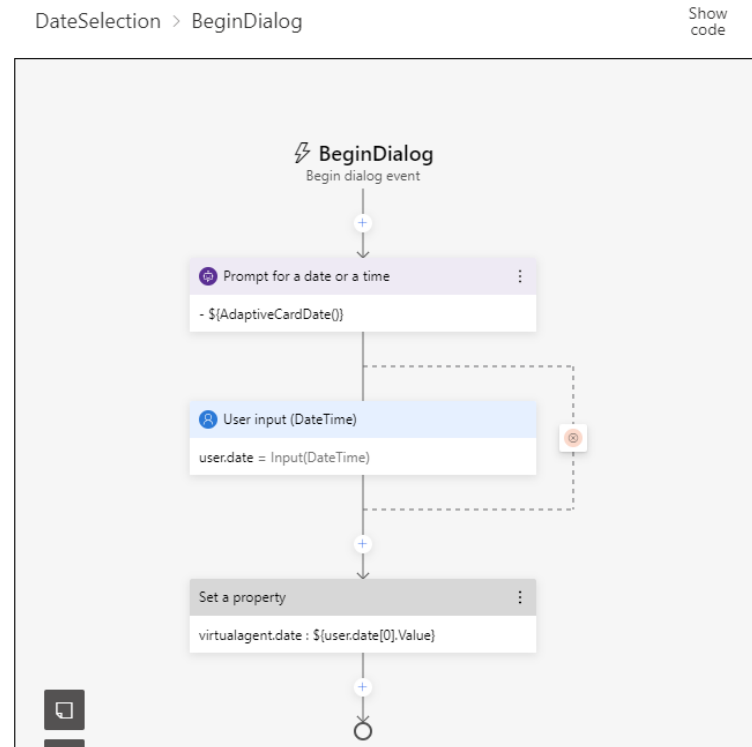
- Datos pasirinkimui pirmiausia sukurtas naujas dialogas kompozitoriuje – *DateSelection*.
- Dialoge pridėtas naudotojo įvesties veiksmas, įvesties tipas – data ir laikas.
- Adaptyvių kortelių dokumentacijoje [12] surastas datos įvesties kodas *JSON* formatu. Tam, jog pokalbių robote atvaizduoti adaptyvią kortelę, kompozitoriuje prie pokalbių roboto atsakymų (angl. *Bot responses*), datos pasirinkimo dialogui pirmiausia sukurta tekstinė savybė – *# adaptivecardjson_date()*, su *JSON* kodu. Tuomet sukurta veiklos (angl. *Activity*) savybė – *# AdaptiveCardDate()* kuri nurodo tekstinę savybę paversti *JSON* formatu (žr. 4.34 pav.).



4.34 pav. Pokalbių roboto atsakymas – adaptyvios kortelės sukūrimas.

- Datos įvesčiai nustatyta numatytoji data (angl. *Value*) – pokalbio dienos data. Nėra įmanoma atlikti skaičiavimų tokių kaip darbo vietos rezervavimo programėlėje datos pasirinkimui, todėl naudotojas apribotas datos įvesčiai nuo šiandien iki septynios dienos į priekį (žr. 4.34 pav.).
- Dialoge datos užklauso (angl. *Prompt for a date or a time*) veiksmui nurodyta adaptyvios kortelės savybė – *\${AdaptiveCardDate()}* tam, jog datos įvesčiai būtų naudojama adaptyvi kortelė (žr. 4.35 pav.).
- Įvesties veiksmo nurodoma naudotojo pasirinkimą saugoti į *user.date* kintamąjį (žr. 4.35 pav.).

- Paskutinis žingsnis pakeisti *Microsoft Copilot Studio* kintamojo *virtualagents.date* reikšmę į naudotojo pasirinktą datą tekstiniu formatu – `${user.date[0].Value}` (žr. 4.35 pav.).



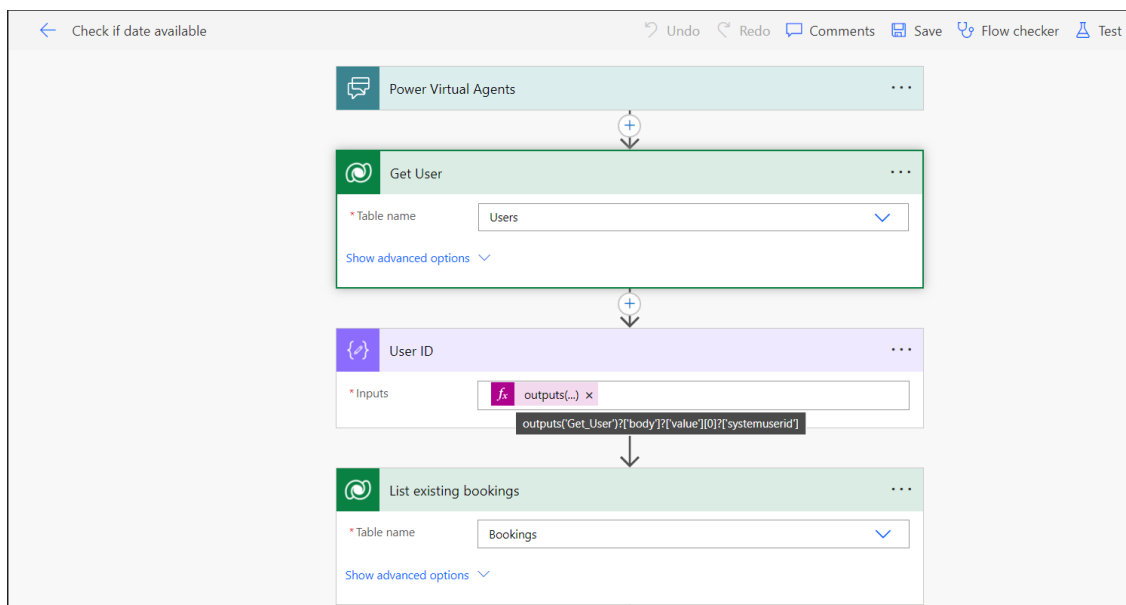
4.35 pav. Datos pasirinkimo dialogas.

Atlikus pakeitimus pokalbių robotas kompozitoriuje išsaugotas ir pridėtas į rezervacijos temą.

Reikia nepamiršti, jog naudotojas gali turėti tik vieną rezervaciją per dieną. Tai reiškia, jog reikia patikrinti ar naudotojas dar neturi rezervacijų pasirinktą dieną, jei turi – paprašyti pasirinkti datą iš naujo. Tai padaryti galima tik *Power Automate* proceso pagalba. Sukurtas *Power Automate* procesas, kuris iš pokalbių roboto gauna du kintamuosius: *bot.UserDisplayName*, tai yra sistemos kintamasis saugantis naudotojo pilną vardą ir *bot.date* – naudotojo pasirinkta data. Procese pridėti šie veiksmai:

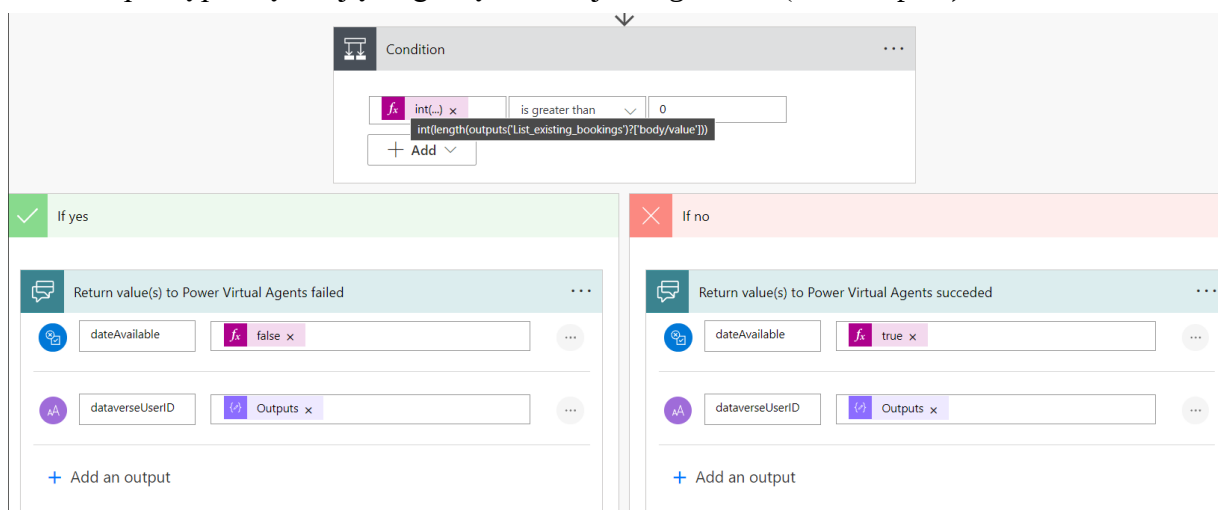
1. Pirmas žingsnis – iš naudotojų lentelės gauti naudotojo identifikacinį numerį pagal kurį vėliau bus ieškoma rezervacijų įrašų. Pridėtas gauti įrašus (angl. *List rows*) veiksmas (žr. 4.36 pav.) ir nurodyta filtruoti įrašus kuriuose pilnas vardas (angl. *Fullname*) lygus *user* kintamajam. Šio kintamojo reikšmė – *bot.UserDisplayName*.
2. Antras žingsnis – iš pirmo žingsnio paimti naudotojo identifikacinį numerį. Šis veiksmas reikalingas, nes gauti įrašus veiksmas, grąžina objektų masyvą, taigi imant naudotojo *ID* tiesiai iš pirmojo veiksmo sukurtu ciklą, nes manoma, jog bus gautas masyvas. Tačiau iš tikrųjų bus gražintas tik vienas įrašas. Todėl pridėtas duomenų operacijos veiksmas kuriame nurodyta iš gautų įrašų masyvo paimti pirmojo (vienintelio) įrašo *ID*. Norint gauti tik vieną duomenų įrašą galima naudoti veiksmą – gauti įrašą pagal *ID* (angl. *Get row by ID*), tuomet šis žingsnis būtų nereikalingas. Bet šiuo atveju *ID* yra ieškomas todėl reikia susitvarkyti su masyvu. Kodo eilutė pirmo įrašo *ID* gavimui – `outputs('Get_User')?['body']?['value'][0]?['systemuserid']` (žr. 4.36 pav.).

3. Trečias žingsnis – išfiltruoti rezervacijų lentelės įrašus pagal dabartinio naudotojo *ID*, kuris gautas antruoju žingsniu, bei pagal naudotojo pasirinktą datą. Panaudotas gauti įrašus veiksmas (žr. 4.36 pav.).



4.36 pav. Egzistuojančios rezervacijos paieškos procesas.

4. Ketvirtasis žingsnis – sąlygos veiksmas kuriame suskaičiuojama kiek įrašų masyve grąžino praeitas veiksmas. Kodas masyvo ilgiui gauti – `int(length(outputs('List_existing_bookings')[?body/value]))`. Sąlygos logika paprasta – masyvo ilgis daugiau už nulį (žr. 4.37 pav.).
5. Paskutiniu žingsniu grąžinamos reikšmės pokalbių robotui. Jeigu praeitame žingsnyje minėta sąlyga yra tiesa reiškia naudotojas jau turi rezervaciją šiai dienai. Tokiu atveju grąžinamas *dateAvailable* bulio tipo kintamasis kurio reikšmė lygi – netiesa. Priešingu atveju – tiesa. Taip pat abiem atvejais grąžinamas *dateverseUserID* kintamasis, kuris saugo naudotojo *ID*. Jo prireiks *Power Automate* procese išsaugant rezervacijos duomenis ir nurodant rezervacijos savininką. Todėl procese nebereikės pridėti tokių pačių pirmų dviejų žingsnių naudotojo *ID* gavimui (žr. 4.37 pav.).

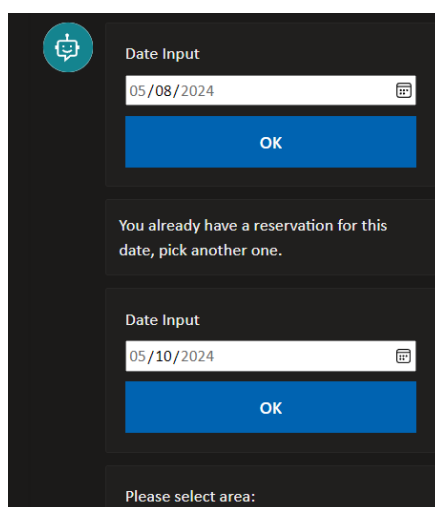


4.37 pav. Egzistuojančios rezervacijos paieškos proceso sąlyga.

Iš *Power Autoamte* proceso grąžintą reikšmę ar data laisva reikia patikrinti pokalbių roboto kūrimo aplinkoje. Pridėtas sąlygos žingsnis, kuriame pirmiausia tikrinama ar kintamasis yra lygus netiesai. Jeigu sąlyga tenkinama išsiunčiama žinutė, jog naudotojas jau turi rezervaciją šiai datai.

Tokiu atveju reikėtų naudotojui leisti pasirinkti datą iš naujo. Čia pasitarnavo žingsnių sujungimo (angl. *Rewire*) funkcija. Ši funkcija leidžia pakeisti žingsnių seką. Po žinutės apie netinkamą datą pokalbių robotas grįžta prie žingsnio kuriame kreipiamasi į datos pasirinkimo dialogą kompozitoriuje. Tai yra sukurtas ciklas, jeigu naudotojas pasirenką datą kuriai jau turi rezervaciją, bus visad grąžinamas į datos pasirinkimą. Jeigu data laisva pereinama prie kitų žingsnių rezervacijos procese. Visi datos pasirinkimo žingsniai pokalbių roboto kūrimo aplinkoje pateikti 4 priede.

Sukurtas testinis duomenų įrašas rezervacijai. Datos pasirinkimas pokalbių robote ištestuotas iš pradžių pasirenkant sukurtos rezervacijos datą ir tuomet pasirenkant datą kuriai rezervacija dar neegzistuoja. Naudotojui buvo pranešta, jog jis jau turi rezervaciją pasirinktai datai ir paprašyta įvesti naują datą. Įvedus datą kuriai rezervacijos dar nėra pereita prie kito žingsnio rezervacijos procese. Taip pat naudotojui nebuvo leista pasirinkti datos senesnės nei šiandien ir didesnės nei už savaitės. Datos pasirinkimas veikia tinkamai (žr. 4.38 pav.).



4.38 pav. Datos pasirinkimo pokalbių robote testavimas.

4.5.4. Aukšto arba teritorijos pasirinkimas

Sekantis žingsnis rezervacijos procese – aukšto arba teritorijos ofise pasirinkimas. Principas labai panašus kaip ir ofiso pasirinkimo – pagal pasirinktą ofisą gaunami aukštai ir teritorijos. Labai svarbu suskaičiuoti pasirinktos datos rezervacijas kiekviename aukšte ir teritorijoje ir palyginti su vietų skaičiumi tam, jog pilnai užimtuose aukštuose atlikti rezervacijos nebūtų įmanoma. Pokalbių robote, kai naudotojas pasirenka datą kuriai dar neturi rezervacijos, pridėtas Power Automate procesas – *Get Flors and Areas*. Procesui iš pokalbių roboto kūrimo aplinkos perduodami du kintamieji: *bot.buildings*, kuriame išsaugotas naudotojo ofiso pasirinkimas ir *bot.date*, kuriame išsaugota naudotojo pasirinkta data rezervacijai. Procese atliekami šie žingsniai:

1. Kaip ir ofiso pasirinkimo procese pirmas žingsnis – kintamojo inicializacija, sukuriamas masyvo tipo kintamasis, *arrFloors*, kurio viduje bus saugomi visi galimi aukštų ir teritorijų pasirinkimai.
2. Antras žingsnis, gauti pasirinkto ofiso įrašą iš pastatų lentelės tam, jog būtų galima gauti ofiso *ID*. Pridėtas gauti įrašus veiksmas kuris išfiltruoja pastatų lentelės įrašus pagal pavadinimą. Tai yra gaunamas pasirinkto ofiso įrašas.
3. Trečiajame žingsnyje iš pastatų lentelės įrašo gauti *ID*. Kaip minėta datos pasirinkimo procese, gauti įrašus veiksmas, visada grąžina objektų masyvą todėl reikia nurodyti, jog imame pirmą įrašą, kuris iš tikrųjų ir yra vienintelis pagal pasirinktą ofisą. Pridėtas

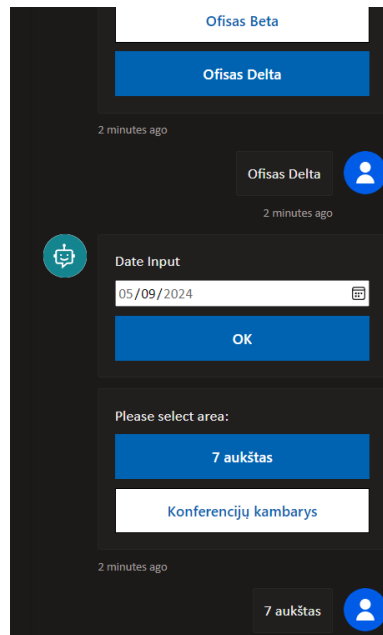
duomenų operacijos žingsnis, kuriame parašytas kodas –
`outputs('List_Buildings')?['body']?['value'][0]?['crf66_buildingsid']`. Taip
gaunamas pasirinkto ofiso *ID* reikalingas aukštų ir teritorijų filtravimui.

4. Ketvirtuoju žingsniu gaunami aukštai ir teritorijos pagal pasirinktą ofisą. Pridėtas gauti įrašus veiksmas, kuriame nurodyta paimti įrašus iš aukštų ir teritorijų lentelės. Aukštai ir teritorijos filtruojami – ofiso *ID* yra lygus naudotojo pasirinkto ofiso *ID*. Naudotojo pasirinkto ofiso *ID* gaunamas trečiajame žingsnyje.
5. Toliau reikėtų patikrinti ar kiekviename pasirinkto ofiso aukšte ar teritorijoje yra laisvų vietų. Pridėtas ciklo veiksmas kuriame einama per kiekvieną pasirinkto ofiso aukštą ir teritoriją.
6. Cikle pridėtas gauti įrašus veiksmas, tai yra, iš rezervacijų lentelės gaunami rezervacijų įrašai pagal aukštą ar teritoriją ir naudotojo pasirinktą rezervacijos datą.
7. Dar vienas žingsnis cikle – skaičiavimas kiek laisvų vietų yra aukšte ar teritorijoje. Pridėtas duomenų operacijos veiksmas, kuriame iš aukšto ar teritorijos teritorijos talpos (angl. *Capacity*) atimamas masyvo, gauto šeštame žingsnyje dydis, kadangi jame gaunamos šio aukšto ar teritorijos rezervacijos. Kodas skaičiavimui –
`int(sub(items('Apply_to_each')?['crf66_capacity'],
length(outputs('List_Bookings_with_area_today')?['body/value'])))`.
8. Toliau seka logika patikrinti praeitame žingsnyje gautam skaičiui. Pridėtas sąlygos veiksmas, kuriame nurodyta logika – praeitame žingsnyje gautas skaičius yra daugiau už nulį. Tai reiškia, jog šiame aukšte ar teritorijoje dar yra laisvos vietos rezervacijoms.
9. Kai sąlyga tenkinama pridėtas veiksmas kuris į *arrFloors* masyvą prideda aukšto ar teritorijos pavadinimą.
10. Kaip ir ofisų pasirinkimų masyvą, taip ir aukštų ar teritorijų, reikia paversti į teksto tipo kintamąjį, nes *Power Automate* negali grąžinti masyvo tipo kintamojo pokalbių robotui.
11. Paskutinis žingsnis grąžinti teksto tipo kintamąjį pokalbių robotui.

Visas aukštų ir teritorijų pasirinkimo gavimo procesas pateiktas 5 priede.

Aukštų ir teritorijų pasirinkimui pokalbių robote pateikti, kompozitoriuje sukurtas dialogas – *FloorsAreasSelection*. Atliekami lygiai tokie patys veiksmai, kaip ir ofiso pasirinkimo dialoge (3 priedas) tik vietoj ofiso kintamojo (*virtualagent.buildings*) imamas ir pasirinkimas išsaugomas aukštų ir teritorijų kintamajame – *virtualagent.floors*.

Išsaugojus dialogą ir pridėjus jį į rezervacijos temą, aukštų ir teritorijų pasirinkimas ištestuotas pokalbių robote (žr. 4.39 pav.).



4.39 pav. Aukštų ir teritorijų pasirinkimo pokalbių robote testavimas.

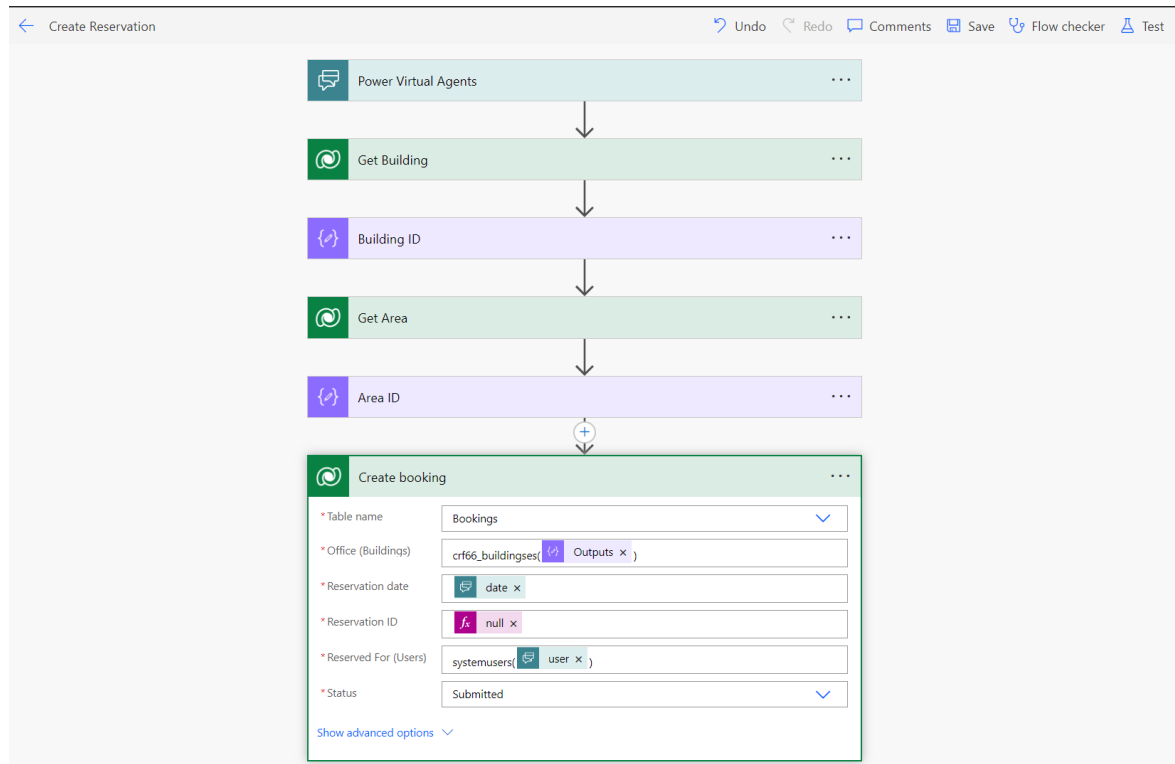
4.5.5. Rezervacijos įrašo sukūrimas

Vienas iš paskutiniųjų žingsnių pokalbių robote, pagal gautas naudotojo įvestis atlikti rezervaciją, tai yra sukurti duomenų įrašą rezervacijų lentelėje. Sukurtas Power Automate procesas, kuriam iš pokalbių roboto aplinkos perduodami šie kintamieji: *bot.buildings* – pasirinktas ofisas, *bot.date* – pasirinkta data, *bot.floors* – pasirinktas aukštas ar teritorija, *dataverseUserID* – naudotojo atliekančio rezervaciją ID naudotojų lentelėje. Procese atliekami šie žingsniai:

1. Pridėtas gauti įrašus veiksmas, kuris iš pastatų lentelės ieško naudotojo pasirinkto ofiso. Reikalingas pasirinkto ofiso ID suradimui pagal kurį bus susieta rezervacija.
2. Antras žingsnis iš gauti įrašus masyvo paimti pirmo ir vienintelio įrašo ID. Kodėl tai reikalinga minėta datos ir aukštų ir teritorijų pasirinkimų gavimo procesuose.
3. Kartojamas pirmas žingsnis, tačiau su aukštų ir teritorijų lentele.
4. Kartojamas antras žingsnis tik imamas aukštų ir teritorijų lentelės įrašo ID.
5. Paskutinis ir svarbiausias žingsnis – rezervacijos įrašo sukūrimas. Pridėtas sukurti naują įrašą (angl. *Add a new row*) veiksmas. Nurodyta rezervacijų lentelėje sukurti naują įrašą su tokiais nustatymais:
 - a. Ofisas yra lygus pastatų lentelės įrašui kurio ID yra naudotojo pasirinkto ofiso ID. Taip yra todėl, nes rezervacijos įrašė, ofiso laukas yra susietas su pastatų lentele.
 - b. Rezervacijos data yra naudotojo pasirinkta data išsaugota kintamajame kuris paimamas iš pokalbių roboto kūrimo aplinkos.
 - c. Rezervacijos ID nurodoma tuščia reikšmė (angl. *Null*), nes šis laukas yra privalomas todėl Power Automate matys tai kaip klaidą, bet iš tikrųjų tai yra unikalus numeris sukuriamas automatiškai atsiradus įrašui.
 - d. Rezervacija priklauso laukas yra lygūs naudotojų lentelės įrašui kurio ID yra naudotojo atliekančio rezervaciją ID.
 - e. Rezervacijos statusas – pateiktas (angl. *Submitted*). Pagal šio lauko nustatymus, numatytasis statusas ir yra *Submitted*, tačiau laukas yra privalomas ir vėl gi Power Automate matytų tai kaip klaidą ir neleistų išsaugoti proceso.

- f. Paskutinis laukas – aukštas ar teritorija yra lygus aukštų ir teritorijų lentelės įrašui kurio *ID* yra naudotojo pasirinkto aukšto ar teritorijos *ID*.

Visas rezervacijos įrašo sukūrimo procesas pateiktas 4.40 paveiksle.

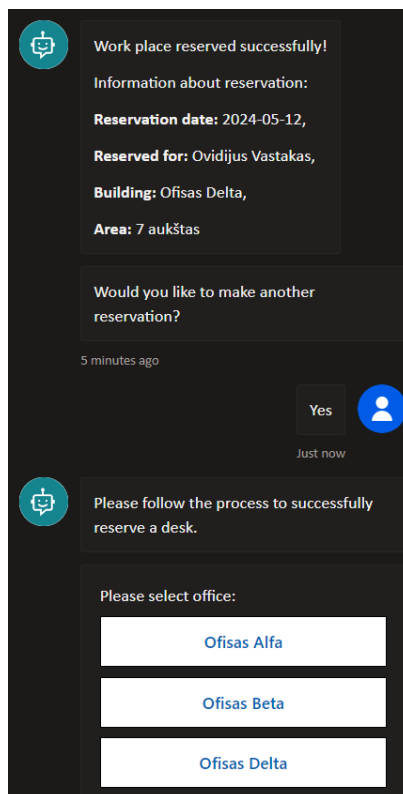


4.40 pav. Rezervacijos sukūrimas su *Power Automate*.

Po rezervacijos įrašo sukūrimo pokalbių robotui nurodyta išsiųsti žinutę su informacija apie atliktą rezervaciją. Parašytas tekstas apie sėkmingą rezervaciją įtraukiant rezervacijos datą, kas atliko rezervaciją, pasitinktas ofisas ir aukštas ar teritorija. Po žinutės apie rezervaciją pokalbis nukreipiamas į pokalbio pabaigos temą (angl. *End of Conversation*), kuri yra sisteminė tema, tai yra sukurta automatiškai po pokalbių roboto sukūrimo. Tema modifikuota taip, jog prieš užbaigiant pokalbį naudotojo būtų paklausama ar jis norėtų atlikti dar vieną rezervaciją ir jeigu atsakymas yra, taip, rezervacijos tema būtų paleidžiama iš naujo. Tokiu būdu naudotojui nereikės iš naujo inicijuoti pokalbio rašant užklausą, jog norima atlikti rezervaciją. Pokalbių robotas ištestuotas atliekant rezervaciją ir atsakius, jog norima atlikti dar vieną rezervaciją (žr 4.42 pav). Pagal 4.42 paveiksle matomus duomenis, *Power Automate* procesas sėkmingai sukūrė įrašą rezervacijų lentelėje (žr. 4.41 pav.).

Reservation ID* ↑ ▼	Reserved For* ▼	Office* ▼	Area ▼	Reservatio...* ▼	Status* ▼
1017	Ovidijus Vastakas	Ofisas Delta	7 aukštas	5/12/2024	Submitted

4.41 pav. *Power Automate* pagalba sėkmingai sukurtas rezervacijos įrašas pagal duomenis įvestus pokalbyje su pokalbių robotu.



4.42 pav. Pokalbyje su pokalbių robotu atlikta rezervacija.

Pokalbių robotas sėkmingai sukurtas ir pilnai atlieka savo funkciją padedant naudotojui atlikti darbo vietos rezervaciją. Taip pat naudotojui duodant pasirinkti tik įmanomus variantus nebėra galimybės netinkamo įrašo rezervacijų lentelėje sukūrimui.

4.6. Darbo vietos rezervavimo programėlės ir pokalbių roboto palyginimas

Pagrindinė tiek programėlės, tiek pokalbių roboto funkcija – atlikti darbo vietos rezervaciją. Darbo vietos rezervavimo sistemoje turint veikiančią programėlę ir pokalbių robotą galima palyginti rezervavimo procesą ir jo efektyvumą.

Nekalbant apie darbo vietos rezervavimą, programėlė turi platesnį funkcionalumą. Programėlės viduje galima ne tik atlikti rezervaciją, bet ir peržiūrėti bei pašalinti savo esamas rezervacijas, nustatyti numatytąjį ofisą, administratorius gali matyti, bei redaguoti administratorių sąrašą, bei atsidaryti visų rezervacijų sąrašą, bei jį filtruoti. Pokalbių robotas administratoriaus funkcionalumą neturi, nes pokalbių robote nėra įmanoma įgyvendinti rolių sistemos ir iš karto atpažinti naudotojo rolės. Lygiai taip pat pokalbių robotu negalima nustatyti numatytojo ofiso, bei ištrinti aktyvių rezervacijų.

Darbo vietos rezervavimo procese, kiekvienos rezervacijos metu, naudojant pokalbių robotą, reikia pasirinkti ofisą kuriame norima atlikti rezervaciją, todėl, nes nėra būdo prieš atliekant rezervaciją patikrinti naudotojo numatytojo ofiso. Pokalbių robote nėra įmanoma apriboti datos pasirinkimo taip, jog naudotojas negalėtų pasirinkti šeštadienių ir sekmadienių arba datų kuriose jau egzistuoja aktyvi naudotojo rezervacija. Todėl, jei naudotojas pasirenka datą kuriai jau turi rezervaciją, tenka prašyti iš naujo pasirinkti datą. Taip pat nėra įmanoma nustatyti, jog galimos datos būtų tik einamosios ir ateinančios savaitės, taip kaip yra programėlėje. Taip yra todėl, nes datos įvesčiai pokalbių robote apriboti galima nurodyti tik datų režį, tai yra, minimali

ir maksimali datos. Pokalbių robote, šalia aukštų ir teritorijų pasirinkimo nėra įmanoma išvesti laisvų vietų skaičiaus, todėl tenka daryti rezervacijų skaičiavimus pagal pasirinktą ofisą ir datą ir neatvaizduoti pilnai užrezervuotų aukštų. Taigi programėlėje galima pridėti ir atvaizduoti daug daugiau informacijos. Rezervacija ir kiti funkcionalumai yra interaktyvesni. Pavyzdžiui naudotojas pradėjęs rezervaciją ir rinkdamasis datą mato kuriose datose jau yra atlikę rezervaciją ir tos datos yra užblokuotos nuo pasirinkimo. Šalia kiekvieno aukšto ir teritorijos mato laisvų vietų skaičių. Verta paminėti, jog programėlę ir kurti paprasčiau. Duomenų atvaizdavimui naudojamos galerijos, duomenų atnaujinimui ar sukūrimui formos arba funkcija *Patch*, mygtukai navigacijai, pasirinkimo įvesties laukai pasirinkimams. Pati logika nėra sudėtinga, daugiau laiko užtrunka viską gražiai sudėlioti ir atvaizduoti. O norint gauti, sukurti ar atnaujinti duomenis tenka naudoti ir kurti *Power Automate* procesus, pasirinkimams atvaizduoti naudoti *Bot Framework Composer*, tai yra mažiau numatytųjų būdų, daugiau kompleksiskų ir nestandartinių sprendimų.

Atlikus 10 bandymų atlikti rezervaciją tiek programėle tiek pokalbių robotu, visas rezervacijos procesas programėlėje atliekamas per vidutiniškai 30 sekundžių. Pokalbių robote rezervacijos procesas vidutiniškai užtrunka 15 sekundžių įskaitant atvejus kuomet pasirenkama data kuriai jau egzistuoja rezervacija ir datos įvestis atliekama iš naujo. Taigi galima daryti prielaidą, jog rezervacijos procesas pokalbių robotu užtrunka dvigubai trumpiau.

5. REZULTATAI IR APIBENDRINIMAS

Baigiamojo darbo metu susipažinta su mažai kodo, jokio kodo programavimo ypatybėmis, pranašumais prieš tradicinį programavimą. Apžvelgta *Microsoft Power Platform*, joje esantys įrankiai. Taip pat apžvelgtos kitos mažai kodo, jokio kodo platformos skirtos programėlių, procesu automatizavimų ir pokalbių robotų kūrimui, bei jų galimybės ir skirtumai tarp *Microsoft Power Platform*.

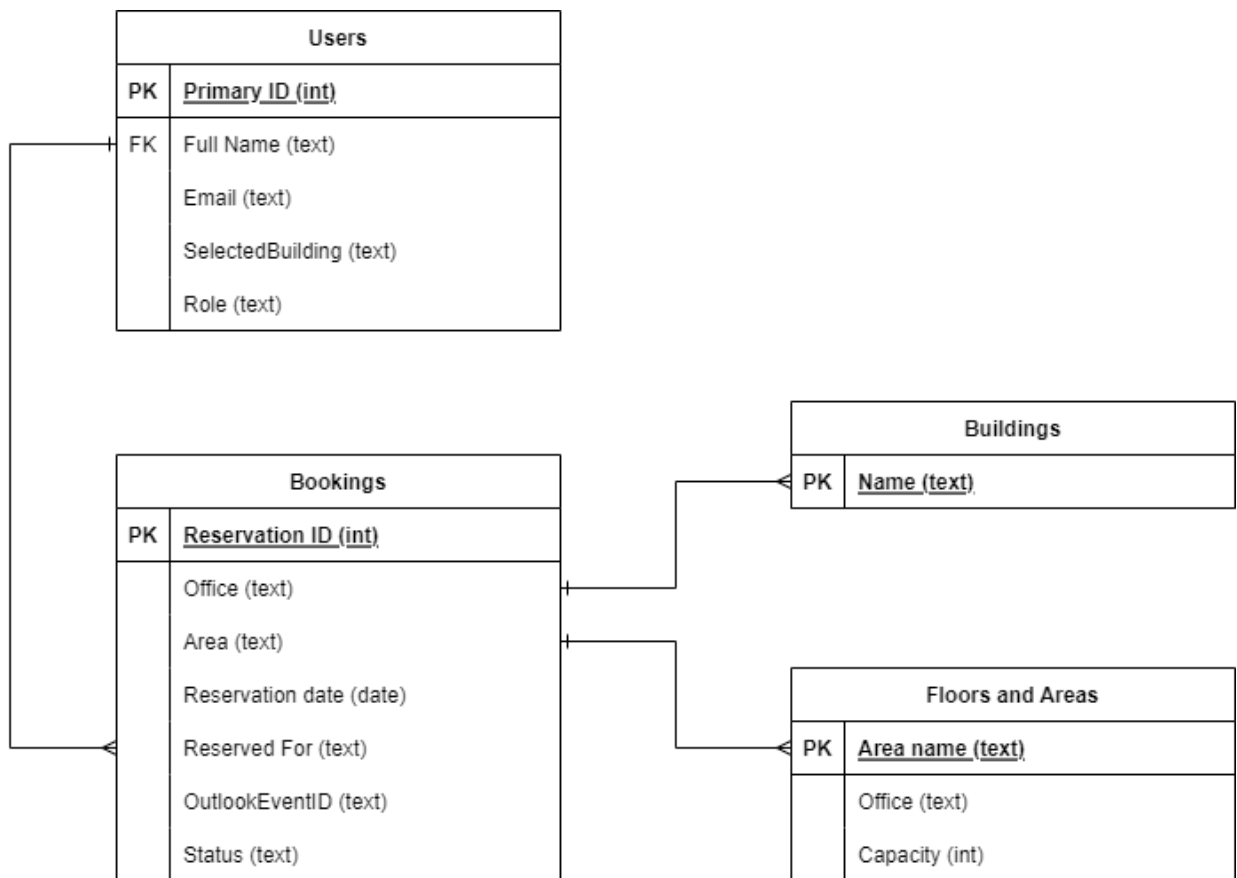
1. Išanalizuotas *Microsoft Copilot Studio* įrankis, tai yra jo veikimo principas, galimybės, bei pritaikomumas darbo vietos rezervacijos sistemai. Taip pat palygintas su kitais mažai kodo, jokio kodo įrankiais skirtais pokalbių robotų kūrimui, licencijų kainos ir integracijos darbo vietos rezervavimo sistemos atžvilgiu.
2. Sukurta darbo vietos rezervavimo sistema. Tai yra sukonfigūruota kūrimo, bei produkcinės aplinkos esančios *Microsoft Teams* viduje. Sukurta ir sukonfigūruota *Dataverse for Teams* duomenų bazė, kurią galima susieti su visais sistemos komponentais, tai yra programėle, procesų automatizavimu ir pokalbių robotu. Sukurta programėlė skirta darbo vietos rezervacijai, rezervacijų apžvalgai *Power Apps* įrankio pagalba. Su *Power Automate* automatizuotas rezervacijos patvirtinimo procesas. Taip pat darbo vietos rezervacijos procesas perkeltas į pokalbių robotą, kuris sukurtas *Microsoft Copilot Studio* ir *Bot Framework Composer* įrankių pagalba.
3. Palygintas darbo vietos rezervavimo procesas ir efektyvumas naudojantis programėle ir pokalbių robotu. Nustatyta, jog programėlėje galima sutalpinti daugiau funkcionalumo, lengviau keisti rezervavimo procesą. Išsiaiškinta, jog pokalbių robote nėra įmanoma apriboti datos pasirinkimo taip kaip programėlėje, tai yra pokalbių robote nėra įmanoma naudotojo apriboti tik darbo dienų pasirinkimais. Taip pat rezervavimo procesas programėlėje yra interaktyvesnis, galima identifikuoti ir pažymėti datas, kuriose jau egzistuoja rezervacija, tuo tarpu pokalbių robote, tai nustatyti įmanoma tik pasirinkus datą ir ją patikrinus. Atlikus rezervacijos eksperimentus, tiek programėlėje, tiek pokalbių robote rezervacijos atliekamos vienodai sėkmingai. Nustatyta, jog rezervavimo procesas yra net du kartus efektyvesnis, tai yra atliekamas du kartus greičiau, naudojantis pokalbių robotu.

Išanalizuotos ir palygintos mažai kodo, jokio kodo platformos tam, jog atrasti tinkamiausią darbo vietos rezervacijos sistemos kūrimui. Buvo išsirinkta *Microsoft Power Platform*, todėl norint sužinoti kokių įrankių prireiks sistemai sukurti apžvelgti joje esantys įrankiai. Išanalizuotas *Microsoft Copilot Studio* įrankis tam, jog suprasti pokalbių robotų kūrimo principą, kokius veiksmus reikės atlikti, bei kaip veikia pats pokalbių robotas. Sukurta kūrimo ir produkcinė aplinka tam, jog atliekamus pakeitimus būtų galima ištestuoti netrikdant galutinio naudotojo naudojimosi produkcinė aplinka ir neužteršiant produkcinės duomenų bazės testavimo duomenų įrašais. Sukurta darbo vietos rezervavimo programėlė padeda sekti darbuotojų skaičių kuris lankosi ofise, siekiant optimizuoti darbo vietų skaičių, taip sumažinant išlaikymo ar nuomos kaštus. Taip pat padeda valdyti konferencijų kambarių užimtumą. Su *Power Automate* automatizuotas rezervacijos patvirtinimo procesas padeda atfiltruoti neaktyvias, negaliojančias rezervacijas tam, jog darbuotojui neatvykus į ofisą jo rezervacija neiškreiptų statistikos. Sukurtas pokalbių robotas darbo vietos rezervavimui. Palygintas darbo vietos rezervavimas programėle ir pokalbių robotu siekiant išsiaiškinti kuriuo atveju rezervacijos procesas yra paprastesnis, greitesnis ir lengviau pasiekiamas. Pokalbių robotas supaprastina rezervavimo procesą iš naudotojo pusės ir leidžia atlikti rezervaciją perpus greičiau nei naudojantis programėle.

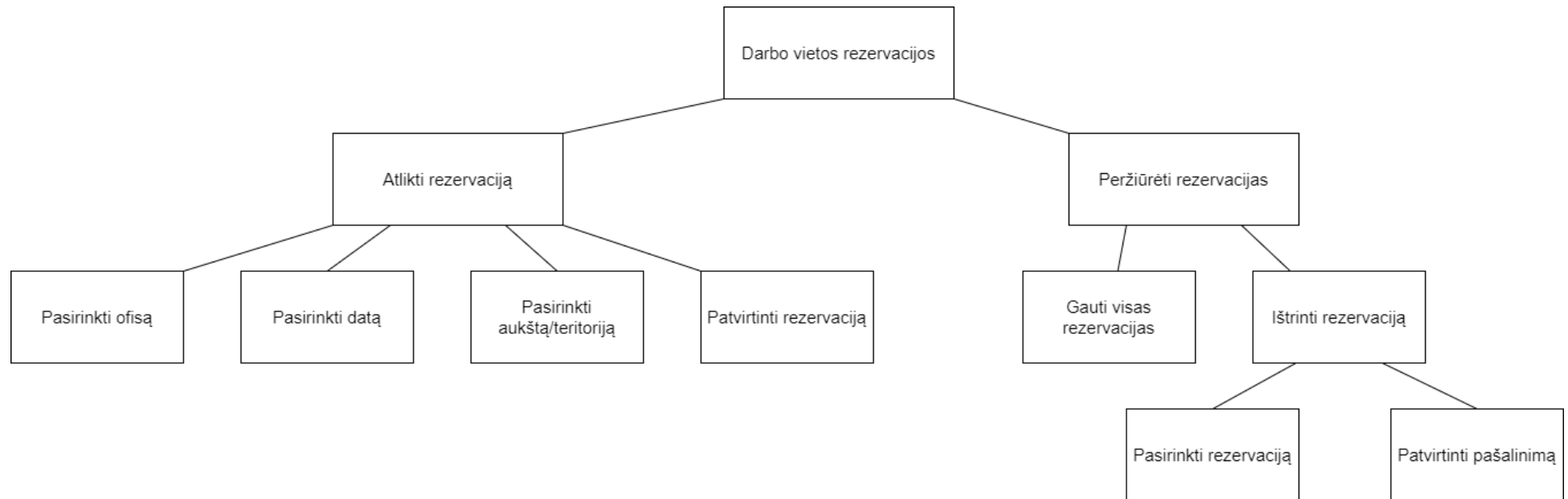
LITERATŪRA

- [1] „Microsoft Copilot Studio documentation,“ 2024. Available: <https://learn.microsoft.com/en-us/microsoft-copilot-studio/>.
- [2] „Microsoft Power Fx overview,“ 2023. Available: <https://learn.microsoft.com/en-us/power-platform/power-fx/overview>.
- [3] S. Shridhar, S. Bose, „Analysis of Low Code-No Code Development,“ *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, pp. 3-6, 2021.
- [4] Richardson, C., Rymer, J.R, „Vendor Landscape: The Fractured, Fertile Terrain Of Low-Code Application Platforms,“ 15 01 2016. Available: <https://rb.gy/h392ll>.
- [5] „AppSheet,“ . Available: <https://about.appsheet.com/how-to-create-an-app/>.
- [6] „Mendix,“ . Available: <https://www.mendix.com>.
- [7] „Landbot.io,“ 2024. Available: <https://www.g2.com/products/landbot/reviews>.
- [8] „Botpress development,“ 2024. Available: <https://botpress.com/features/conversation-studio>.
- [9] „Botpress Pricing,“ 2024. Available: <https://botpress.com/pricing>.
- [10] R. Shewale, „Demansage,“ 29 lapkričio 2023. Available: <https://www.demandsage.com/microsoft-teams-statistics/>.
- [11] „Bot Framework Composer,“ 2024. Available: <https://learn.microsoft.com/en-us/composer/introduction?tabs=v2x>.
- [12] „Adaptive Cards,“ 2024. Available: <https://adaptivecards.io/>.
- [13] „Microsoft Power Platform ALM,“ 2024. Available: <https://learn.microsoft.com/en-us/power-platform/alm/overview-alm>.
- [14] „UML,“ 2024. Available: <https://www.uml.org>.
- [15] „Microsoft Power Automate scheduled tasks documentation,“ 2023. Available: <https://learn.microsoft.com/en-us/power-automate/run-scheduled-tasks?tabs=classic-designer>.
- [16] „Microsoft Power Automate Dataverse,“ 2024. Available: <https://learn.microsoft.com/en-us/power-automate/dataverse/overview>.
- [17] „Microsoft Teams Actions,“ 2024. Available: <https://learn.microsoft.com/en-us/connectors/teams/#actions>.

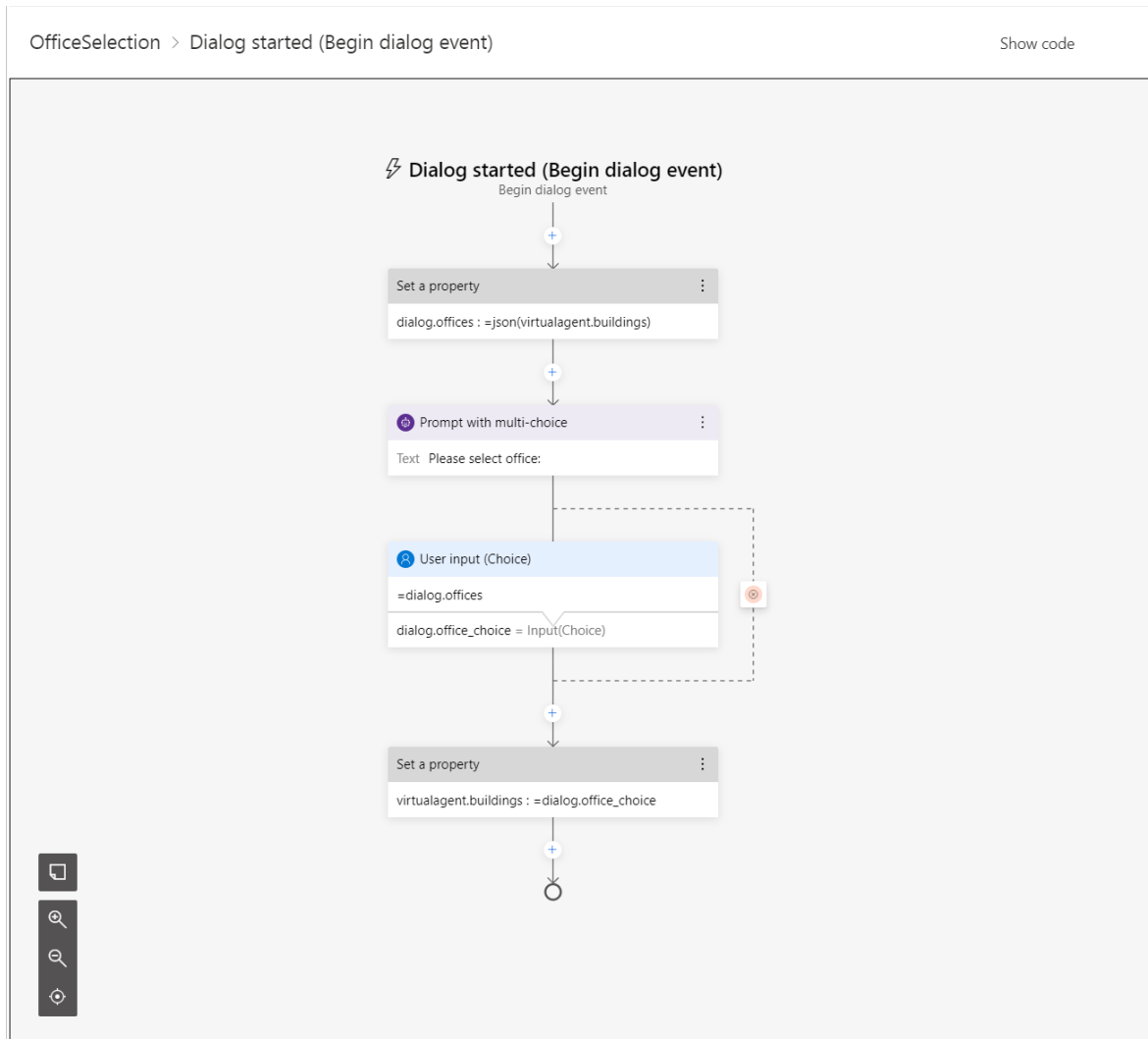
1 PRIEDAS DARBO VIETOS REZERVACIJOS SISTEMOS DUOMENŲ MODELIS



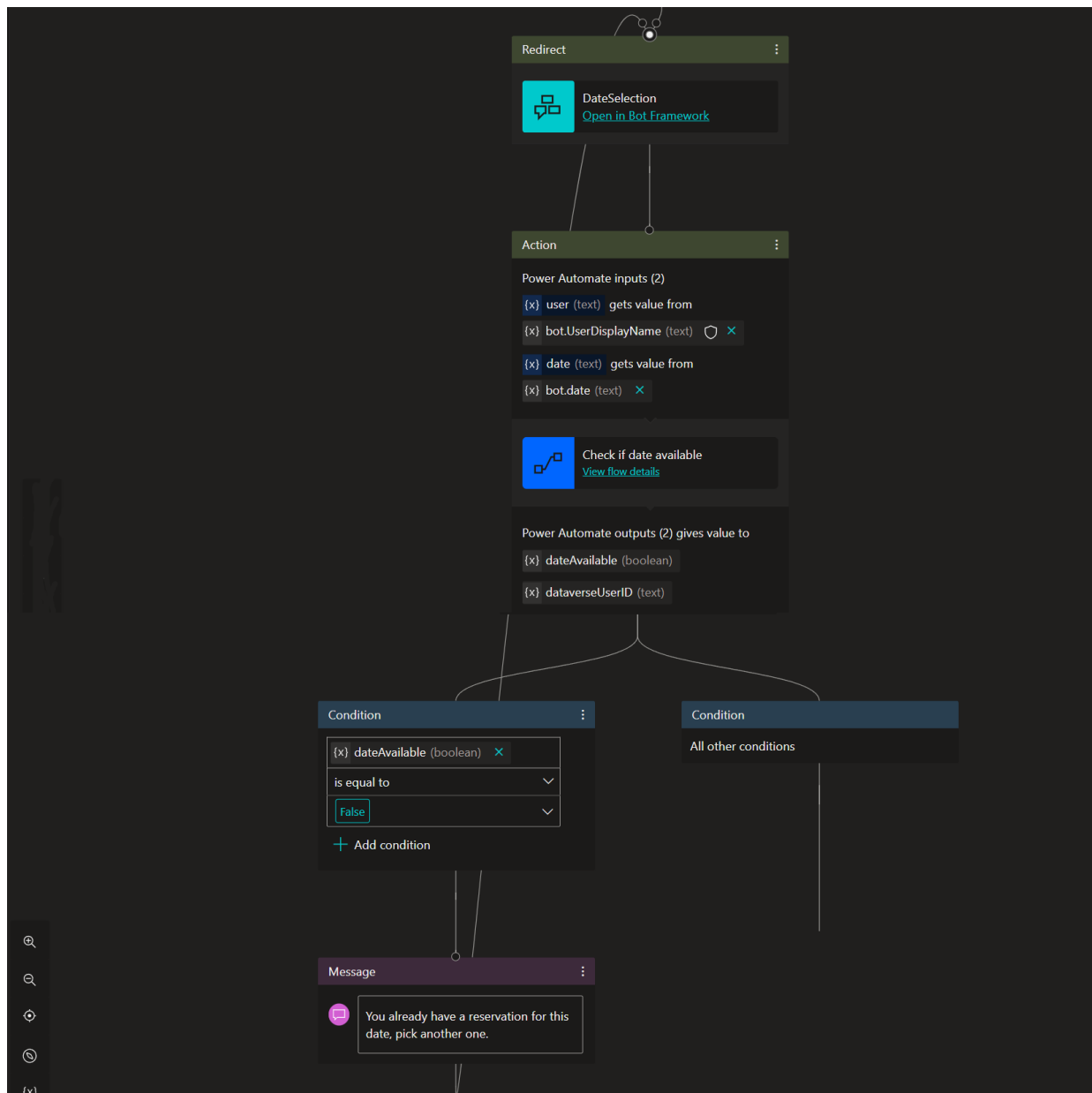
2 PRIEDAS DARBO VIETOS REZERVACIJOS AUKŠTO LYGIO STRUKTŪRINĖ SCHEMA



3 PRIEDAS BOT FRAMEWORK COMPOSER OFISO PASIRINKIMAS



4 PRIEDAS DATOS PASIRINKIMAS POKALBIŲ ROBOTE



5 PRIEDAS AUKŠTŲ IR TERITORIJŲ PASIRINKIMO GAVIMAS

