

Mažeikių Gabijos gimnazija

Informacinės technologijos

IV b klasė

Automatizuotos programavimo uždavinių tikrinimo sistemos sukūrimas naudojant  
„.NET Core“ karkasą

Benas Budrys

---

Mokinio vardas ir pavardė

Sigitas Kažukalovas, informacinių  
technologijų mokytojas metodininkas

---

Mokytojo, brandos darbo vadovo vardas ir  
pavardė, kvalifikacija

2019

## TURINYS

TURINYS.....	2
NAUDOJAMŲ SĄVOKŲ ŽODYNAS.....	3
ĮVADAS.....	4
1. SISTEMOS MINIMALUS VEIKIANTIS PRODUKTAS.....	6
1.1 Technologijų pasirinkimas.....	6
1.2 Veikimo principas .....	6
1.3 Kodo testavimas .....	8
2. SISTEMA INTERNETINĖJE APLINKOJE .....	9
2.1 Modelio–Vaizdo–Valdiklio dizaino modelis.....	9
2.2 Duomenų bazė.....	10
2.3 Uždavinio modelis.....	11
2.4 Autentifikavimas ir autorizavimas .....	13
2.5 Sistemos naudojimas nario atžvilgiu .....	16
2.6 Sistemos naudojimas administratoriaus atžvilgiu .....	18
2.7 Lokalizacija.....	19
3. SISTEMOS ĮDIEGIMAS DEDIKUOTAME ĮRENGINYJE IR PARUOŠIMAS NAUDOJIMUI	20
IŠVADOS.....	21
LITERATŪROS ŠALTINIŲ SĄRAŠAS .....	22

## NAUDOJAMŲ SĄVOKŲ ŽODYNAS

1. Algoritmas – baigtinė seka pažingsninių tikslų instrukcijų, skirtų atlikti tam tikrą darbą.
2. Minimalus veikiantis produktas (angl. minimum viable product) – produktas, tik su tiek funkcionalumo, kiek užtenka patenkinti pirmuosius norus.
3. Karkasas (angl. framework) – programinės įrangos rinkinys, suteikiantis bendrinę funkcionalumą, kuriuo remiantis programuotojas gali vystyti savo projektą.
4. Šaltinio failas (angl. source file) – failas, kuriame aprašyta programa kuria nors programavimo kalba.
5. Autentifikavimas – procesas, kai vartotojas pateikia savo prisijungimo duomenis, kurie palyginami su esančiais duomenų bazėje ir jų sutapimo atveju vartotojui duodamas leidimas vykdyti veiksmus, kuriems jis yra autorizotas.
6. Autorizavimas – procesas, nustatantis kokius veiksmus vartotojas gali atlikti.
7. Lokalizacija – procesas, kai tam tikras produktas ar programinė įranga yra pritaikoma konkretaus regiono kalbinei ir kultūrinei terpei.

## IVADAS

Sprendžiami (mokyklos kurso) programavimo uždaviniai mokinio reikalauja sukurti programą, kuri vykdytų algoritmą<sup>1</sup>. Sukurta programa priima pradinis duomenis, juos apdoroja, atlieka skaičiavimus ir išveda rezultatą.



1 pav. programos veikimo procesas

Toliau mokytojas rankiniu būdu (įvesdamas pradinis duomenis) gali patikrinti programos teisingumą, įvertinti ar programos išvestas rezultatas sutampa su tuo, kurio tikimasi. Tokia uždavinių tikrinimo eiga atskiria programavimą nuo „tradicinių“ mokslų (matematikos, literatūros, biologijos). Šiuose moksluose vertinimas neretai subjektyvus, nėra tikslaus apibrėžimo, kas yra teisingas atsakymas. Pavyzdžiui, rašinė mokytojas tikrina pasiremdamas vertinimo instrukcijomis, tačiau visiškai teisingas užduoties atsakymas nėra apibrėžtas. Nors kituose moksluose ir yra tikslus, apibrėžtas atsakymas, pavyzdžiui matematikoje, mokytojas neteisingo atsakymo atveju gali atsižvelgti į sprendimo eigą, skirti dalį taškų. Šios situacijos negalioja (mokyklos kurso) programavimo uždaviniams, kuriuos sprendžiant galimas tik arba teisingas, arba neteisingas atsakymas (arba programa gauna teisingą rezultatą, arba ne).

Šie uždavinių tikrinimo skirtumai itin pastebimi atkreipus dėmesį į tikrinimo proceso užtrukimą laiką. Kadangi programavimo uždavinių tikrinimo procesas sąlyginai nesudėtingas, vienos užduoties tikrinimo laikas ganėtinai mažas lyginant su kitomis disciplinomis, kur tikrinimas gali užtrukti nuo kelių iki keliolikos minučių. Programavimo užduočių tikrinimas gali būti supaprastintas iki 2 žingsnių:

1. Pateikiami pradiniai duomenys.
2. Įvertinamas algoritmo gautas rezultatas.

Dėl tokių nesudėtingų instrukcijų šis procesas gali būti automatizuotas, t. y. šį procesą gali atlikti kompiuteris. Automatizavimo nauda tikrinimo proceso atžvilgiu:

- Tikrina žymiai greičiau nei įprastas žmogus. Dažnai patikrinti užduotį kompiuteris geba per kelias sekundes.

---

<sup>1</sup> Algoritmas – baigtinė seka pažingsninių tikslų instrukcijų, skirtų atlikti tam tikrą darbą.

- Galimybė tikrinti daug kartų parenkant skirtingus pradinis duomenis. Tokiu būdu nustatomas algoritmo universalumas - tinkamumas bet kokiems pradiniais duomenimis.
- Galimybė naudoti anksčiau įvestus pradinis duomenis. Dėl to padidėja proceso efektyvumas, nes nereikia gaišti laiko iš naujo įvedant pradinis duomenis.

Šios įžvalgos priveda prie tokios **problemos** – kaip automatizuoti programavimo uždavinių tikrinimą?

Norint išspręsti šią problemą, iškeliamas brandos darbo **tikslas** – sukurti automatizuotą programavimo uždavinių tikrinimo sistemą ir pritaikyti ją sprendžiant programavimo užduotis.

Brandos darbo **tyrimo objektas** – programavimo uždavinių tikrinimo sistema. Egzistuoja panašių sistemų, pavyzdžiui Lietuvos mokinių informatikos olimpiadoje naudojama [konkurso valdymo sistema](#) arba tinklalapiuose kaip [codeforces](#) ar [codewars](#), tačiau yra esminių skirtumų, kuo šio brandos darbo produktas išsiskiria iš kitų sistemų:

1. Sistema veikia lokaliame tinkle, pavyzdžiui mokyklos.
2. Didelis dėmesys sistemos naudojimo paprastumui. Svarbu, kad naudotis sistema būtų nesudėtinga ir greitai perprantama.
3. Daugiau dėmesio skiriama mokyklos kurso programavimo uždaviniams.
4. Sistema pritaikyta naudojimui informacinių technologijų, programavimo pamokose, leidžia mokytojui nesunkiai įvertinti mokinių programavimo uždavinių sprendimo įgūdžius.

Šio produkto realizavimo **uždaviniai**:

1. Sukurti sistemos minimalų veikiančią produktą<sup>2</sup>.
2. Perkelti sistemą į internetinę aplinką.
3. Įdiegti sistemą dedikuotame įrenginyje ir paruošti naudojimui.

**Metodai:**

1. Technologijų pasirinkimas ir mokymasis jas pritaikyti.
2. Sistemos funkcionalumo nustatymas.
3. Sistemos dizaino, išvaizdos pasirinkimas.
4. Sistemos realizavimas, sukūrimas.

---

<sup>2</sup> Minimalus veikiantis produktas (angl. minimum viable product) – produktas, tik su tiek funkcionalumo, kiek užtenka patenkinti pirmuosius norus.

# 1. SISTEMOS MINIMALUS VEIKIANTIS PRODUKTAS

## 1.1 Technologijų pasirinkimas

Pradedant bet koki programavimą apimančią projektą svarbu teisingai pasirinkti naudojamas technologijas. Reikia atsižvelgti į keletą aspektų:

- Technologijos paskirtis. Svarbu žinoti kam skirta technologija ir ką ji geba daryti. Galbūt ji per plati pasiekti norimam tikslui arba atvirkščiai, galbūt jai net trūksta funkcionalumo.
- Technologijos populiarumas. Šis kriterijus lemia ar internete bus galima rasti daug informacijos, kuri galėtų pagelbėti susidūrus su iškilusia problema.
- Turimos žinios apie technologiją. Reikia suvokti, jog mokytis naują technologiją užtrunka laiko, tad projektas gali būti įgyvendintas greičiau, jei pasirenkama pažįstama technologija.

Atsižvelgiant į šiuos kriterijus, nusprendžiau sistemą kurti naudojant *.NET Core* karkasą<sup>3</sup>. Veiksniai, lėmę tokį pasirinkimą:

- Turima patirtis su šia technologija. Kadangi moku programuoti *C#* programavimo kalba, darbą pavyks įgyvendinti profesionaliau ir greičiau.
- Kelių platformų suderinamumas leis naudoti šį projektą įvairiose platformose (Windows, Linux).
- *.NET Core* yra nemokamas ir [atviro kodo](#), todėl tai pakankamai populiarus variantas tarp programuotojų.
- *ASP.NET Core* yra dalis *.NET Core* karkaso, o tai padės plėsti sistemą ir perkelti ją į internetinę aplinką.

## 1.2 Veikimo principas

Konkretus minimalus veikiantis produktas, pritaikytas šiam projektui, yra programa, neturinti grafinės sąsajos (visą informaciją vartotojui pateikia per konsolės langą), gebanti priimti *C++* programavimo kalbos šaltinio failą<sup>4</sup> (su *.cpp* plėtiniu), jį sukompiliuoti, įvykdyti, perskaityti rezultatą, jį įvertinti (palyginti su vartotojo nurodytu tikimusi rezultatu) ir išrašyti rezultatą.

---

<sup>3</sup> Karkasas (angl. framework) – programinės įrangos rinkinys, suteikiantis bendrinį funkcionalumą, kuriuo remiantis programuotojas gali vystyti savo projektą.

<sup>4</sup> Šaltinio failas (angl. source file) – failas, kuriame aprašyta programa kuria nors programavimo kalba.

Ši programa turi pagrindinius 4 veikimo etapus:

1. Šaltinio failo nustatymas. Programa nustato kelią į vartotojo pateiktą failą, išsiaiškina ar šis failas egzistuoja, ar pateiktas failas turi tinkamą plėtinį.
2. Kompiliacija. Norint paleisti vartotojo programą, iš pradžių šaltinio failą reikia sukompiliuoti. Kadangi mūsų kompiliuojamas šaltinio failas yra C++ kalbos, šiame etape programa sukuria naują procesą, kuris iškviečia populiarią g++ kompiliatorių, pateikia jam kelią į šaltinio failą ir taip pat nurodo išvesties kelią. Etapo rezultatas yra iš vartotojo šaltinio failo naujai sukurtas .exe plėtinio failas.
3. Vartotojo programos vykdymas. Paleidžiamas dar vienas procesas, tačiau šį kartą jis iškviečia ankstesniame etape sukurtą vartotojo programos .exe plėtinio failą. Jeigu vartotojas nurodo pradinis duomenis, pateikia juos šiam procesui (jie gali būti bet kokio primityvaus duomenų tipo, nes vartotojo programai jie pateikiami kaip simbolių eilutės *string* tipo) ir nuskaityto šio proceso išvestį. Gali atsitikti, jog vartotojo programa veikdama užtrunka per ilgą laiką, tad jos veikimas automatiškai nutraukiamas po 2000 ms.
4. Įvertinimas. Programa įvertina ar vartotojo pateiktos programos išvestis sutampa su nurodytu tikimusi rezultatu ir pateikia viso veikimo ataskaitą (angliškai, kadangi ji nebus matoma galutinės sistemos vartotojui).

Šią programą galima paleisti *.NET Core* komandinės eilutės sąsajos pagalba su tokiais argumentais:

dotnet MVP.dll <kelias į šaltinio failą> <pradiniai duomenys> <tikimasis rezultatas>

Pavyzdžiui:

dotnet MVP.dll "D:\sum.cpp" "5 6" "11"

```
#include <iostream>

using namespace std;

int main()
{
    int a, b;
    cin >> a >> b;
    cout << a+b << endl;
}
```

2 pav. sum.cpp šaltinio failas

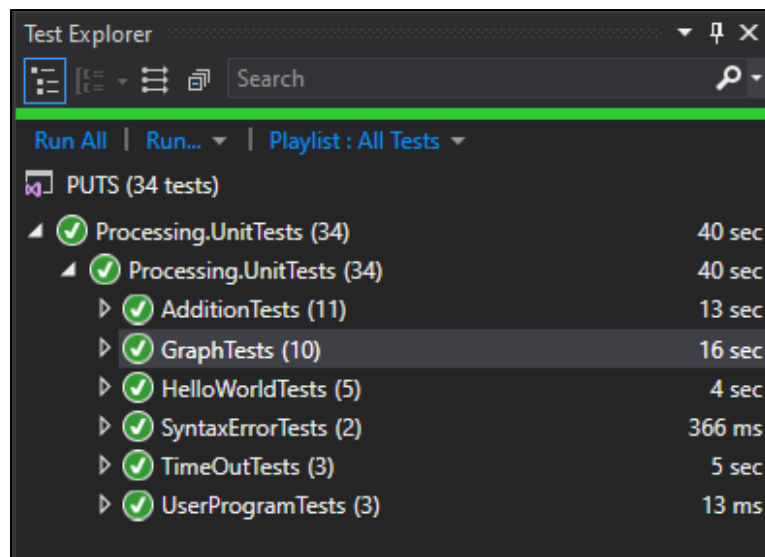
```
dotnet MVP.dll "D:\sum.cpp" "5 6" "11"
Source file set
Compiled successfully
Program executed successfully
Output matches
```

3 pav. programos ataskaita (išvestis sutampa)

### 1.3 Kodo testavimas

Siekiant užtikrinti sėkmingą minimalaus veikiančio produkto kodo veiklą, kodas buvo testuojamas *XUnit* testavimo karkaso pagalba. Automatiškai buvo tikrinami 6 specifiniai scenarijai su įvairiais pradiniais duomenimis:

1. Šaltinio failas yra paprastas tekstinis failas (.txt plėtinys). Programa turėjo neleisti nustatyti kelio į šaltinio failą, kadangi jo plėtinys netinkantis.
2. Šaltinio failas turi tinkamą plėtinį, tačiau turi sintaksės klaidų. Šis šaltinio failas negalėjo kompiliuotis.
3. Šaltinio failas yra geras, tačiau vykdymas užtrunka per ilgai. Programa šiuo atveju turi automatiškai nutraukti vartotojo programos vykdymą po 2000 ms.
4. Šaltinio failas yra geras ir vartotojo programa nereikalauja jokių pradinių duomenų. Programa turėjo teisingai įvertinti tokį šaltinio failą.
5. Šaltinio failas yra geras ir vartotojo programa reikalauja pradinių duomenų. Programa turėjo teisingai įvertinti tokį šaltinio failą.
6. Šaltinio failas yra geras ir vartotojo programa reikalauja pradinių duomenų, tačiau šį kartą vartotojo programa sudėtingesnė, reikalauja daugiau laiko ir pradinių duomenų nei ankstesnės. Programa turėjo teisingai įvertinti tokį šaltinio failą.



4 pav. kodo testų rezultatai



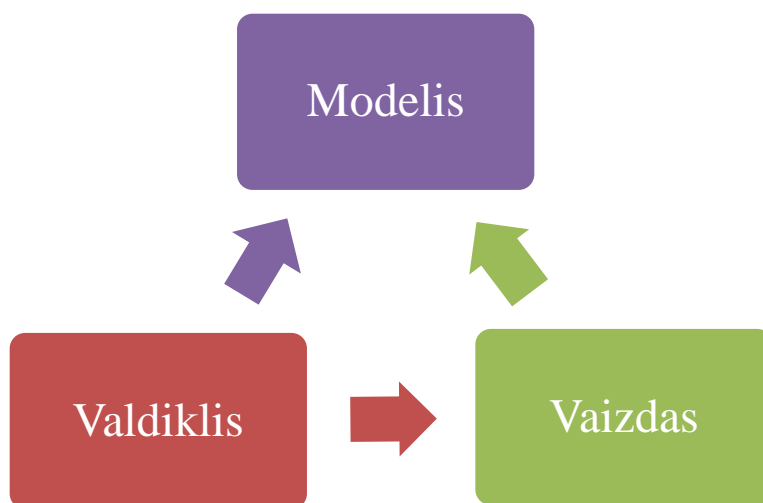
## 2. SISTEMA INTERNETINĖJE APLINKOJE

### 2.1 Modelio–Vaizdo–Valdiklio dizaino modelis

Perkeliant sistemą į internetinę aplinką, naudojamas jau anksčiau minėtas *ASP.NET Core* karkasas. Vienas iš šio karkaso palaikomų internetinės aplikacijos dizaino modelių yra Modelio–Vaizdo–Valdiklio arba trumpiau MVC (dėl angliško termino Model–View–Controller). MVC aplikacijos veikimą paskirsto į 3 pagrindinius komponentus: modelius, vaizdus ir valdiklius. Šis dizaino modelis padeda įgyvendinti funkcionalumo atskyrimo principą:

- Modelis apibūdina aplikacijos būseną ir bet kokią jos atliekamą verslo logiką ar operaciją.
- Vaizdas atsakingas už turinio pateikimą vartotojui.
- Valdiklis atsakingas už vartotojo užklausų priėmimą, darbą su modeliu, rodomo vaizdo parinkimą.

Atsižvelgiant į tai, jog MVC yra vienas iš populiariausių internetinės aplikacijos dizaino modelių, atskiria funkcionalumą, padeda išlaikyti kodą organizuotą ir yra palaikomas *ASP.NET Core* karkaso, paskatino pasirinkti jį kuriant programavimo uždavinių tikrinimo sistemą.



5 pav. MVC komponentų ryšiai

## 2.2 Duomenų bazė

Kaupti duomenims reikalinga duomenų bazė, tad šiai užduočiai buvo pasirinkta populiari reliacinių duomenų bazių valdymo sistema *MySQL*. Nors įprastai darbui su *.NET Core* karkasu naudojama *Microsoft SQL Server*, kitaip pasirinkti paskatino didesnis *MySQL* populiarumas ir tai, jog *MySQL* yra atviro kodo.

Norint pasiekti duomenų bazėje esančius duomenis, aplikacijai, naudojančiai *.NET Core* karkasą, reikalinga duomenų prieigos technologija *Entity Framework Core (EF Core)*. Ši technologija padeda aplikacijai asocijuoti pačios aplikacijos objektus su duomenimis, esančiais duomenų bazėje. Naudojant EF Core, pritaikomas kodo pirmenybės metodas, o tai reiškia, kad siejant aplikaciją su duomenų baze, iš pradžių kodu sukuriamas modelis, pagal kurį vėliau kuriamos duomenų bazės schema ir lentelės. Pavyzdžiui, modelis *Test* (6 pav.) po *EF Core* duomenų bazės atnaujinimo sugeneruoja naują lentelę (7 pav.). Pastebima, jog nors modelis turėjo ne primityvų duomenų tipą (*Problem*), jis *MySQL* lentelėje automatiškai atvaizduojamas kaip primityvus duomenų tipas (*ProblemID*), identifikuojantis eilutę kitoje lentelėje. Tokiu būdu buvo sukurta visa sistemos duomenų bazė (8 pav.).

```
public class Test
{
    public int TestID { get; set; }

    public string GivenInput { get; set; }

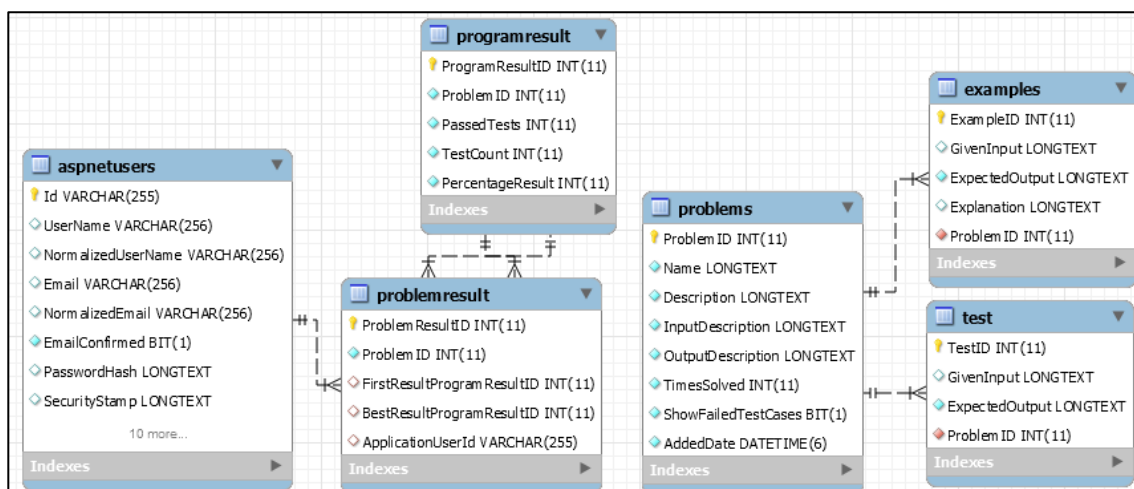
    public string ExpectedOutput { get; set; }

    public Problem Problem { get; set; }
}
```

6 pav. kodu sukurtas modelis

	TestID	GivenInput	ExpectedOutput	ProblemID
▶	8	2 4 6	Taip	3
	9	5 8 3	Ne	3
	10	456456 ...	Ne	3
*	NULL	NULL	NULL	NULL

7 pav. *EF Core* sugeneruota *MySQL* lentelė



8 pav. sistemos duomenų bazė

## 2.3 Uždavinio modelis

Programavimo uždavinių tikrinimo sistemoje svarbu tiksliai ir teisingai apibrėžti uždavinį. Sistemoje uždavinio modelį apibūdina šie parametrai:

- Uždavinio ID. Skirtas identifikuoti unikalų uždavinį ir šį parametą sistema parenka automatiškai.
- Pavadinimas.
- Sąlyga.
- Įvesties apibūdinimas. Nurodo kokio įvesties formato vartotojui reikia tikėtis sprendžiant uždavinį.
- Išvesties apibūdinimas. Nurodo kokio išvesties formato tikėtis sistema tikrinant vartotojo sprendimą.
- Pavyzdžių rinkinys. Neprivalomas parametras, kurį nurodo sistemos vartotojas. Pavyzdį sudaro:
  - Pavyzdžio ID. Skirtas identifikuoti unikalų pavyzdį ir šį parametą sistema parenka automatiškai.
  - Įvestis. Nurodo konkretų pradinių duomenų pavyzdį.
  - Tikimasi išvestis. Nurodo konkretų rezultatą, kuris gaunamas su duotais pradiniais duomenimis.
  - Paaiškinimas. Neprivalomas parametras, kuris paaiškina sudėtingesnę pavyzdį.
- Testų rinkinys. Apibūdina testus, su kuriais bus tikrinamas vartotojo sprendimas. Šie testai vartotojui nematomi ir jie nustato ar vartotojo sprendimas teisingas. Rinkinyje turi būti bent vienas testas. Testą sudaro:
  - Testo ID. Skirtas identifikuoti unikalų testą ir šį parametą sistema parenka automatiškai.
  - Įvestis. Nurodo konkrečius pradinius duomenis.
  - Tikimasi išvestis. Nurodo konkretų rezultatą, kurio sistema tikisi.
- Išsprendimų skaičius. Nurodo unikalių sistemos vartotojų pilnai (surinko 100%) išsprendė uždavinį. Ši parametą sistema skaičiuoja automatiškai.
- Rodyti nepavykusius testus. Šis parametras nurodo ar uždavinio sprendimo išvestinėje vartotojui rodyti testus, kurių jo pateiktas sprendimas nepaėjo.
- Pridėjimo data. Nurodo kada uždavinys buvo pridėtas į sistemą. Šį parametą sistema parenka automatiškai.

**Pavadinimas**  

Jonuko saldainiai

**Sąlyga**  

Jonukui mama kas savaitę duoda n saldinių. Apskaičiuokite kiek Jonukas turės saldinių po t savaitių jei 0-tą savaitę jis turi a salaidnių.

**Ivestis**  

Pirmoje eilutėje trys sveikieji skaičiai.  
n t a

**Išvestis**  

Išveskite vieną sveikąjį skaičių - kiek turės saldinių Jonukas po t savaitių

**Rodyti nepavykusius testus** ☒

**Pavyzdžiai**

**Pateikiama įvestis** Ištrinti pavyzdį  

3 4 2

**Tikėtina išvestis**  

14

**Paaiškinimas**  

Jonukas kas savaitę gauna po 3 saldinius, tad po 4 savaitių jis turės  $2+3+3+3+3 = 14$  saldinių

Pridėti pavyzdį

**Testai**

**Pateikiama įvestis** Ištrinti testą  

5 2 9

**Tikėtina išvestis**  

19

**Pateikiama įvestis** Ištrinti testą  

78 84 456

**Tikėtina išvestis**  

7008

Pridėti naują testą

Kurti

9 pav. naujo uždavinio kūrimo pavyzdys

Sukūrus uždavinį, jis pridamas prie bendro uždavinių sąrašo ir yra prieinamas visiems vartotojams sistemos *Uždaviniai* skiltyje. Ši skiltis pateikia visus sistemos uždavinius ir suteikia prieigą prie sprendimo puslapio. Laikui bėgant sistemos uždavinių skaičius didėja, todėl gali pasidaryti sunku ieškoti uždavinių. Dėl šios priežasties *Uždaviniai* skiltis turi paieškos langelį, pagal kurio tekstą sistema suranda uždavinius, kurių pavadinime ar ID parametre yra šis tekstas. Taip pat siekiant padaryti sistemos naudojimą patogų, sukurtas uždavinių rūšiavimas. Juos (mažėjančiai ir didėjančiai) rūšiuoti galima pagal 4 sąrašo pradžioje esančius kriterijus:

1. ID.
2. Pavadinimą.
3. Išsprendimų skaičių.
4. Pridėjimo datą.

Uždaviniai			
		ieškoti	Q
ID	Pavadinimas	Išspręsta	Pridėta
9	Laipsnis	7	2019-11-03
8	Atstumas tarp miestų	11	2019-11-01
7	Kaina	5	2019-11-05
6	Lektuvas	26	2019-11-04
5	Modulis	14	2019-11-08
4	Jonuko saldainiai	2	2019-11-02

10 pav. skiltis *Uždaviniai*

## 2.4 Autentifikavimas ir autorizavimas

Autentifikuoti<sup>5</sup> ir autorizuoti<sup>6</sup> sistemoje yra naudojama *ASP.NET Core Identity* sistema. Ji palengvina šių procesų vykdymą ir užtikrina saugumą. Neautentifikuotas vartotojas negali spręsti uždavinių ir jo prieiga prie sistemos minimali, todėl sistemą pradedantis naudoti vartotojas visų pirma turi užsiregistruoti. Tai jis gali padaryti navigavimo juostoje (esančioje viršuje) pasirinkęs dešiniausią

<sup>5</sup> Autentifikavimas – procesas, kai vartotojas pateikia savo prisijungimo duomenis, kurie palyginami su esančiais duomenų bazėje ir jų sutapimo atveju vartotojui duodamas leidimas vykdyti veiksmus, kuriems jis yra autorizotas.

<sup>6</sup> Autorizavimas – procesas, nustatantis kokius veiksmus vartotojas gali atlikti.

elementą *Registruotis*. Vartotojas bus nukreiptas į registracijos formą. Šiam procesui reikalingi tik būtiniausi duomenys apie asmenį:

1. Prisijungimo vardas. Sistemoje naudojamas tik autentifikavimui.
2. Vardas. Naudojamas nustatyti vartotojo tapatybę vertinant jo spęstus uždavinius.
3. Pavardė. Naudojamas nustatyti vartotojo tapatybę vertinant jo spęstus uždavinius.
4. Grupė. Naudojama sugrupuoti vartotojus lengvesniam ieškojimui. Tai gali būti klasės ar grupės pavadinimas.
5. Slaptažodis. Sistemoje naudojamas tik autentifikavimui.

Baigus pildyti registracijos formą ir ją pateikus, vartotojas automatiškai prijungiamas prie naujos paskyros. Taip pat vartotojas prisijungti gali pasirinkęs *Prisijungti* navigavimo juostos elementą. Įvykus situacijai, kai neprisijungęs vartotojas bando pasiekti jam neleistiną informaciją, jis automatiškai nukreipiamas į prisijungimo formą. Prisijungimo procesui reikalingi:

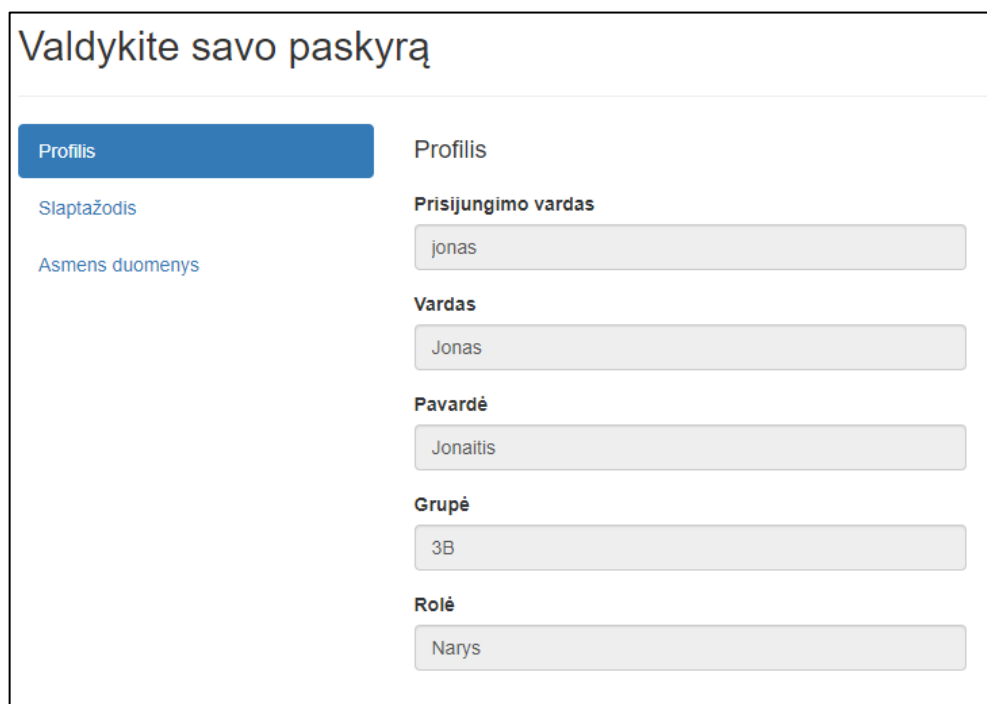
1. Prisijungimo vardas.
2. Slaptažodis.

Registruotis	Prisijungti
<b>Prisijungimo vardas</b> <input type="text"/>	<b>Prisijungimo vardas</b> <input type="text"/>
<b>Vardas</b> <input type="text"/>	<b>Slaptažodis</b> <input type="password"/>
<b>Pavardė</b> <input type="text"/>	<input type="checkbox"/> Prisiminti mane
<b>Grupė</b> <input type="text"/>	<input type="button" value="Prisijungti"/>
<b>Slaptažodis</b> <input type="password"/>	
<b>Patvirtinkite slaptažodį</b> <input type="password"/>	
<input type="button" value="Registruotis"/>	

11 pav. registracijos ir prisijungimo formos

Prisijungęs vartotojas gali pažiūrėti savo profilio informaciją navigavimo juostoje pasirinkus *Profilis* elementą. Čia jis turės 3 pasirinkimus:

1. Peržiūrėti profilio informaciją.
2. Pasikeisti slaptažodį.
3. Tvarkyti savo asmens duomenis. Šioje skiltyje vartotojas gali parsisiųsti asmens duomenis, kuriuos apie jį kaupia sistema, arba ištrinti savo paskyrą ir visus savo asmens duomenis kartu.



Valdykite savo paskyrą

**Profilis**

Slaptažodis

Asmens duomenys

**Profilis**

**Prisijungimo vardas**

jonas

**Vardas**

Jonas

**Pavardė**

Jonaitis

**Grupė**

3B

**Rolė**

Narys

12 pav. profilio informacija

Kiekvienas sistemos vartotojas turi priskirtą rolę (arba kelias roles). Sistemoje išskiriamos 3 rolės (autorizavimo lygmenys):

1. Narys. Įprastas vartotojas, ši rolė priskiriama vartotojui prisiregistravus, galintis spręsti uždavinius ir matyti savo rezultatus. Sistemai veikiant informacinių technologijų pamokose tokią rolę įprastai turėtų mokiniai.
2. Administratorius. Vartotojas, galintis kurti, redaguoti ir pašalinti užduotis, taip pat ir matyti visų narių įvertinimus. Sistemai veikiant informacinių technologijų pamokose tokią rolę įprastai turėtų mokytojai.
3. Super vartotojas. Sistemoje egzistuoja tik vienas toks vartotojas. Jis skirtas sistemos diegimo procese priskirti administratoriaus rolę kitiems vartotojams.

## 2.5 Sistemos naudojimas nario atžvilgiu

Narys, skiltyje *Uždaviniai* išsirinkęs norimą arba radęs paskirtą užduotį ir paspaudęs ant užduoties pavadinimo, bus nukreiptas į užduoties *Sprendimo* skiltį. Čia jis gali matyti informaciją apie uždavinį:

- Pavadinimas.
- Sąlyga.
- Įvestis.
- Išvestis.
- Pavyzdžiai (jei tokių yra).

### 4. Jonuko saldainiai

Sprendimas

Rinktis

Pasirinkite failą...

Įkelti

Geriausias rezultatas 100%

Sąlyga

Jonukui mama kas savaitę duoda n saldainių. Apskaičiuokite kiek Jonukas turės saldainių po t savaitių jei 0-tą savaitę jis turi a salaidnių.

Įvestis

Pirmoje eilutėje trys sveikieji skaičiai.  
n t a

Išvestis

Išveskite vieną sveikąjį skaičių - kiek turės saldainių Jonukas po t savaitių

Pavyzdžiai

Pateikiama įvestis

3 4 2

Tikėtina išvestis

14

Paaiškinimas

Jonukas kas savaitę gauna po 3 saldainius, tad po 4 savaitių jis turės  $2+3+3+3+3 = 14$  saldainių

13 pav. *Sprendimo* skiltis

Taip pat narys šioje skiltyje gali rasti vietą užduoties sprendimo įkėlimui. Išsprendus užduotį, šioje vietoje narys turi įkelti savo parašytą programą. Įkėlus šaltinio failą, narys bus nukreiptas į puslapį, kur jam teks palaukti kol sistema ištikrins ir įvertins jo sprendimą.





14 pav. vartotojas laukia kol sistema baigs darbą

Sistemai baigus darbą, visi su sprendimu susiję failai iš sistemos pašalinami ir narys nukreipiamas į galutinę, *Rezultatų*, skiltį. Čia jis galės pamatyti savo sprendimo rezultatus, kurie pateikiami tokia tvarka:

1. Kompiliavimo rezultatas. Nurodo ar nario pateikta programa susikompiliavo. Jei ne, nariui išrašoma kompiliatoriaus klaidos žinutė.
2. Testų rezultatas. Pateikiamas sėkmingų testų skaičius, visų testų skaičius ir procentinis sprendimo įvertinimas apskaičiuojamas pagal formulę  $\frac{\text{sėkmingų testų skaičius}}{\text{visų testų skaičius}} \times 100\%$  ir rezultatas suapvalinamas iki vienetų
3. Toliau seka visi testai. Jei uždavinyje buvo pasirinkta rodyti nepavykusius testus, prie nepavykusių testų bus rodoma kokius pradinis duomenis šiame teste sistema pateikė programai ir kokios išvesties tikėjosi.

## 5. Modulis

---

Kompiliavimas Sėkmingas

Testai 3/4 75%

---

Testas 1 Sėkmingas

---

Testas 2 Sėkmingas

---

Testas 3 Nepavykęs

Išvestis nesutampa

**Tikėtasi**

Taip

**Gauta**

Ne

---

Testas 4 Sėkmingas

15 pav. *Rezultatų* skiltis

## 2.6 Sistemos naudojimas administratoriaus atžvilgiu

Prie visko, ką gali atlikti narys, administratorius turi papildomų veiksmų. Jis yra atsakingas už sistemos prižiūrėjimą, užduočių tvarkymą. Apart jau minėtų užduočių kūrimo, redagavimo, šalinimo privilegijų, administratorius gali žiūrėti narių sprendimų rezultatus. Nuėjęs į bet kurios užduoties *Sprendimo* skiltį, jis viršutiniame dešiniajame kampe pamatys mygtuką *Vertinimas*. Pastarąjį paspaudus, administratorius bus nukreiptas į *Vertinimo* skiltį.

5. Modulis			
		ieškoti	Q
Vardas ir pavardė	Grupė	Pirmas rezultatas	Geriausias rezultatas
Jonas Jonaitis	3B	<span style="background-color: #ffc107; padding: 2px 5px;">3/4</span> <span style="background-color: #ffc107; padding: 2px 5px;">75%</span>	<span style="background-color: #ffc107; padding: 2px 5px;">3/4</span> <span style="background-color: #ffc107; padding: 2px 5px;">75%</span>
Petras Petraitis	4C	<span style="background-color: #ffc107; padding: 2px 5px;">1/4</span> <span style="background-color: #ffc107; padding: 2px 5px;">25%</span>	<span style="background-color: #28a745; color: white; padding: 2px 5px;">4/4</span> <span style="background-color: #28a745; color: white; padding: 2px 5px;">100%</span>

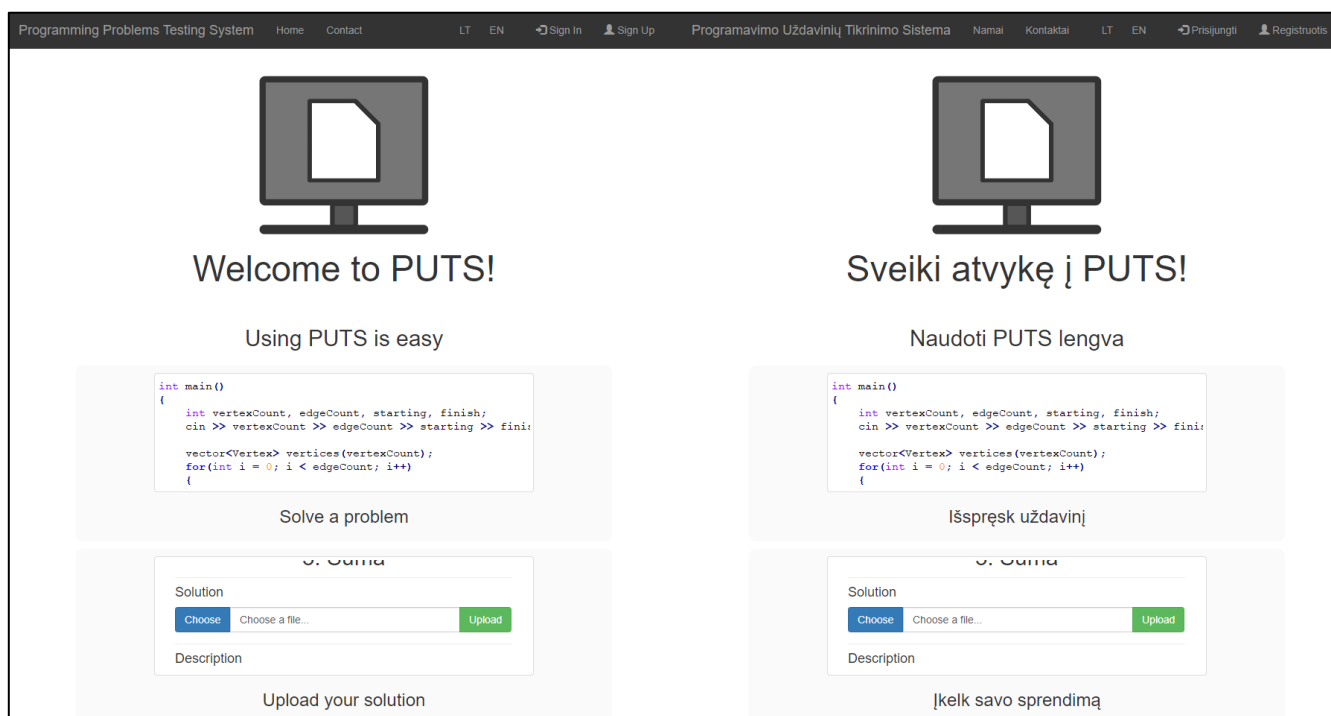
16 pav. *Vertinimo* skiltis

Šioje skiltyje nurodomi visi sistemos nariai, kurie bandė spręsti šia užduotį. Norint greičiau ir patogiau atrasti narius, galima pasinaudoti paieškos langeliu, kuris suras narius, kurių varde ir pavardėje arba grupėje yra paieškos langelio tekstas. Administratorius mato du vertinimo rezultatus:

1. Pirmas rezultatas. Nario rezultatas, gautas pateikiant sprendimą užduočiai patį pirmą kartą. Jo pakeisti nariui neįmanoma.
2. Geriausias rezultatas. Nario geriausias rezultatas pasiektas sprendžiant šia užduotį. Net jei ir narys gauna mažesnę įvertinimą, tai nepakeičia jo geriausio rezultato.

## 2.7 Lokalizacija

Visa sistema yra lokalizuota<sup>7</sup>, palaiko anglų ir lietuvių kalbas. Tai padaro sistemą universalesnę ir labiau prieinamą. Tačiau suprantama, kad uždavinių informacija bus tokia kalba, kuria administratorius rašė kurdamas naują uždavinį. Kalba pakeičiama spaudžiant ant norimos kalbos navigavimo juostos dešinėje pusėje.

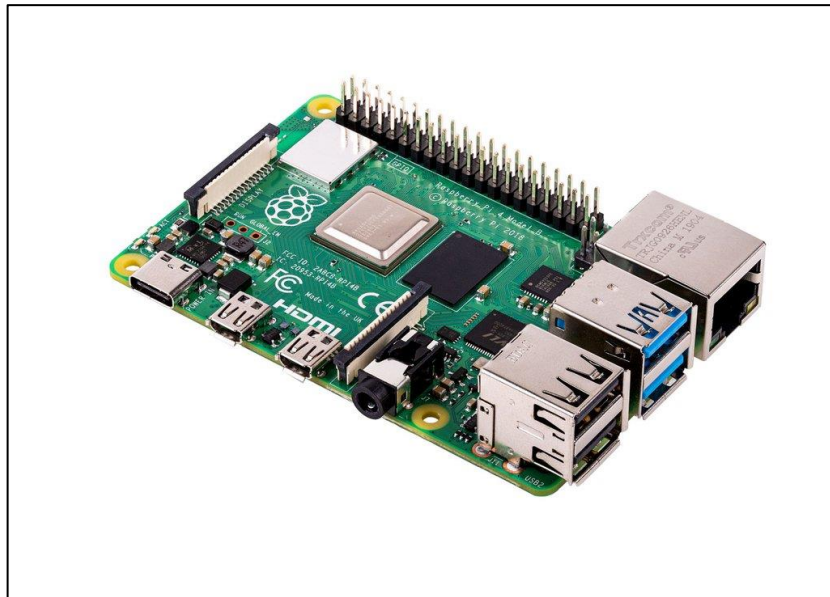


17 pav. sistema angliškai (kairė) ir sistema lietuviškai (dešinė)

<sup>7</sup> Lokalizacija – procesas, kai tam tikras produktas ar programinė įranga yra pritaikoma konkrečiau regiono kalbinei ir kultūrinei terpei.

### 3. SISTEMOS ĮDIEGIMAS DEDIKUOTAME ĮRENGINYJE IR PARUOŠIMAS NAUDOJIMUI

Kadangi sistema kurta su *.NET Core* karkasu, ji suderinama su keliomis platformomis, yra ganėtinai universali. Susidomėję sistema ir norintys ją įsidiegti, galės tai nesunkiai padaryti *Windows* operacinėje sistemoje, pavyzdžiui su *IIS* tinklapio serveriu, ir *Linux* operacinėje sistemoje, pavyzdžiui su *Apache* ar *nginx* tinklo serveriu. Atsižvelgiant į tai, jog sistema vienu metu neturėtų naudotis daug vartotojų, galimas variantas įdiegti šia sistemą tokiam įrenginyje kaip *Raspberry Pi*. Tokiu atveju, šios sistemos kaina būtų visiškai minimali.



18 pav. *Raspberry Pi* kompiuteris

Paleidus sistemą, lieka ją tik paruošti naudojimui. Sistemoje kai kuriems nariams (pavyzdžiui mokytojams) reikia priskirti administratoriaus rolę, tad tai reikia padaryti per jau anksčiau minėtą Super Vartotoją. Šio vartotojo prisijungimo duomenis galima rasti viename iš sistemos failų (*appsettings.json*) arba tiesiog prisijungti prie sistemos su prisijungimo vardu *superuser* ir slaptažodžiu *superuser*, tačiau pirmą kartą prisijungus, reikia nepamiršti pakeisti šio vartotojo slaptažodžio. Norint paprastam nariui priskirti administratoriaus rolę, naršyklės viršuje, prie pradžios URL reikia prikabinti štai tokią komandą:

*/account/makeadmin?username=<prisijungimo vardas>&makeAdmin=<true arba false>*

Pavyzdžiui, komanda po apačia vartotojui, kurio prisijungimo vardas *petras*, priskirs administratoriaus rolę (nes reikšmė *true*):

*/account/makeadmin?username=petras&makeAdmin=true*

## **IŠVADOS**

Sukurta programavimo uždavinių tikrinimo sistema:

1. Automatizuoja programavimo uždavinių tikrinimo procesą.
2. Palengvina informacinių technologijų mokytojų darbą, taupo laiką.
3. Vartotojus priverčia domėtis, sekti savo rezultatus, įvertinti įgūdžius.
4. Naudotis lengva ir patogiu.
5. Universali, suderinama su keliomis platformomis.

## LITERATŪROS ŠALTINIŲ SĄRAŠAS

1. S. Ragaišis. *Algoritmai*. Prieiga per internetą: <http://klevas.mif.vu.lt/~ragaisis/InfIvadas/Algoritmai.htm>.
2. *Minimum viable product*. Prieiga per internetą: [https://en.wikipedia.org/wiki/Minimum\\_viable\\_product](https://en.wikipedia.org/wiki/Minimum_viable_product).
3. *Overview of ASP.NET Core MVC*. Prieiga per internetą: <https://docs.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-2.1>
4. *Home repository for .NET Core*. Prieiga per internetą: <https://github.com/dotnet/core>
5. *Entity Framework Core*. Prieiga per internetą: <https://docs.microsoft.com/en-us/ef/core/>
6. *Introduction to Identity on ASP.NET Core*. Prieiga per internetą: <https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-2.1&tabs=visual-studio>
7. *Overview of ASP.NET Core Security*. Prieiga per internetą: <https://docs.microsoft.com/en-us/aspnet/core/security/?view=aspnetcore-2.1>
8. *Create a web app with ASP.NET Core MVC*. Prieiga per internetą: <https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-mvc-app/?view=aspnetcore-2.1>
9. *ASP.NET Core MVC with EF Core - tutorial series*. Prieiga per internetą: <https://docs.microsoft.com/en-us/aspnet/core/data/ef-mvc/?view=aspnetcore-2.1>
10. *Globalization and localization in ASP.NET Core*. Prieiga per internetą: <https://docs.microsoft.com/en-us/aspnet/core/fundamentals/localization?view=aspnetcore-2.1>
11. *Lokalizacija*. Prieiga per internetą: <https://lt.wikipedia.org/wiki/Lokalizacija>
12. *Raspberry Pi*. Prieiga per internetą: <https://www.raspberrypi.org>