|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | |
| **TECHNOLOGIJŲ FAKULTETAS**  **INŽINERIJOS IR INFORMATIKOS** **katedra** | | | | |
|  | | | |  |
| **MATEMATIKOS FUNKCINIŲ GRAFIKŲ MOKOMASIS ŽAIDIMAS** | | | | |
|  | | | |  |
| Kursinis darbas | | | | |
|  | | | |  |
| Informatikos studijų programos | | | | |
| valstybinis kodas 6531BX004 | | | | |
| Informatikos studijų krypties | | | | |
|  | | | |  |
| Autorius Renaldas Bumblauskis | |  |  |  |
|  |  | *(parašas)* |  | *(data)* |
| Vadovas doc. dr. Aleksas Narščius | |  |  |  |
|  |  | *(parašas)* |  | *(data)* |
| Klaipėda, 2023 | | | | |

**TURINYS**

[LENTELIŲ SĄRAŠAS 2](#_Toc131680444)

[PAVEIKSLŲ SĄRAŠAS 3](#_Toc131680445)

[ĮVADAS 4](#_Toc131680446)

[1. PROGRAMŲ SISTEMOS PROJEKTAVIMAS 5](#_Toc131680447)

[1.1. Duomenų bazės projektavimas 5](#_Toc131680448)

[1.2. Naudojami duomenys 8](#_Toc131680449)

[1.2.1. Duomenų objektai 8](#_Toc131680450)

[1.2.2. Duomenų struktūros 10](#_Toc131680451)

[1.3. Programinis projektas 11](#_Toc131680452)

[1.4. Projektavimo šablonai 12](#_Toc131680453)

[2. PROGRAMŲ SISTEMOS REALIZACIJA 16](#_Toc131680454)

[2.1. JPA realizavimas 16](#_Toc131680455)

[2.2. DB užklausos 17](#_Toc131680456)

[2.3. Algoritmai 17](#_Toc131680457)

[2.4. Grafinė naudotojo sąsaja 19](#_Toc131680458)

[3. PROGRAMŲ SISTEMOS KOKYBĖS UŽTIKRINIMAS 22](#_Toc131680459)

[3.1. Testavimas 22](#_Toc131680460)

[3.2. Kodo versijų kontrolė 23](#_Toc131680461)

[IŠVADOS 24](#_Toc131680462)

[LITERATŪROS IR KITŲ INFORMACIJOS ŠALTINIŲ SĄRAŠAS 25](#_Toc131680463)

[PRIEDAI 26](#_Toc131680464)

# LENTELIŲ SĄRAŠAS

# PAVEIKSLŲ SĄRAŠAS

[Duomenų bazės struktūra. 5](#_Toc131680465)

[Administratorių lentelė. 6](#_Toc131680466)

[Visi mokytojai prisiregistravę prie svetainės. 6](#_Toc131680467)

[Visi priregistruoti moksleiviai prie svetainės. 7](#_Toc131680468)

[Visų moksleivių rezultatai. 7](#_Toc131680469)

[„Graph“ klasė. 8](#_Toc131680470)

[Sudėtinis objektas iš kelių duomenų bazės lentelių. 9](#_Toc131680471)

[„Interval“ klasė. 10](#_Toc131680472)

[Intervalų generavimas ir įdėjimas į masyvą. 10](#_Toc131680473)

[Apskritimų sukūrimas ir sudėjimas į masyvą. 11](#_Toc131680474)

[Projekto architektūrinis modelis. 12](#_Toc131680475)

[„Singleton“ kūrimo šablonas. 13](#_Toc131680476)

[„Promise“ struktūrinis šablonas. 13](#_Toc131680477)

[„Factory“ elgesio šablonas. 14](#_Toc131680478)

[Funkcija nustatanti HTML elementą. 14](#_Toc131680479)

[Duomenų bazės pasiekimas PHP kalboje. 16](#_Toc131680480)

[DB užklausa pakeisti moksleivio slaptažodį, kai jis prisijungia pirmą kartą. 17](#_Toc131680481)

[Sudėtinė DB užklausa gauti visus studentus ir jų rezultatus pagal mokytojo id. 17](#_Toc131680482)

[Mokytojo registracijos algoritmas. 18](#_Toc131680483)

[Tempiamo apskritimo intervalo pozicijos nustatymo algoritmas. 18](#_Toc131680484)

[Apskritimo judėjimo animacijos į intervalo poziciją algoritmas. 19](#_Toc131680485)

[Registracijos mokytojui ir prisijungimas vartotojams. 20](#_Toc131680486)

[Administratoriaus puslapis kai jis prisijungia. 20](#_Toc131680487)

[Lygių meniu. 21](#_Toc131680488)

[Lygio pavyzdys. 21](#_Toc131680489)

[Našumo testas metodui. 22](#_Toc131680490)

[Griežtas lygybės testas metodui. 22](#_Toc131680491)

[Github kodo versijų kontrolė. 23](#_Toc131680492)

# ĮVADAS

**Tikslas*.*** Įsisavinti duomenų struktūrų, duomenų bazių, grafinės naudotojo sąsajos programavimą, projektavimo šablonų taikymą, kodo versijų kontrolės, dokumentavimo bei testavimo įrankių naudojimą, programuojant dalykinės srities vientisą taikomąją programą.

Siekiant įgyvendinti numatytą tikslą, keliami šie praktikos **uždaviniai**:

|  |
| --- |
| 1. Suprojektuoti programų sistemą:    1. Suprojektuoti sistemos pradinius duomenis ir rezultatą;    2. Apibrėžti programoje naudojamas duomenų struktūras;    3. Aprašyti programinio projekto struktūrą;    4. Parinkti ir pritaikyti programavimo šablonus projektuojant architektūrą. |
| 1. Realizuoti programų sistemą:    1. Suprogramuoti duomenų įvedimo/išvedimo srautus;    2. Suprogramuoti programos skaičiavimo algoritmus;    3. Suprogramuoti grafinę naudotojo sąsają; |
| 1. Užtikrinti programų sistemos kūrimo proceso valdymą ir kokybės užtikrinimą:    1. Sukurti kodo tikrinimo automatinių testus;    2. Naudoti kodo versijų kontrolės įrankius. |

# PROGRAMŲ SISTEMOS PROJEKTAVIMAS

Matematikos funkcinių grafikų mokomasis žaidimas yra interaktyvus internetinis žaidimas, skirtas padėti moksleiviams geriau suprasti matematikos funkcinių grafikų bruožus ir savybes. Žaidimo tikslas yra padėti moksleiviams išmokti atpažinti pagrindinius funkcinių grafikų elementus. Šiame skyriuje bus aprašoma duomenų bazės projektavimas ir joje naudojami duomenys, taip pat ir esančios duomenų struktūros bei duomenų objektai šiame projekte. Po to bus aprašomas iš ko sudarytas programinis projektas, kokios priemonės yra naudojamos, visi esantys projektavimo šablonai.

## Duomenų bazės projektavimas

Duomenų bazės projektavimas yra svarbus etapas kuriant bet kokią duomenų saugyklą. Tai apima reikalavimų nustatymą ir duomenų modeliavimą. Reikalavimų nustatymo procesas sudaro supratimą apie duomenų bazės naudojimo atvejus ir vartotojų poreikius. Dėl to reikia apibrėžti, kokie duomenis bus saugomi duomenų bazėje, kaip jie bus organizuoti, kaip bus atliekamos užklausos ir kokia bus duomenų bazės naudojimo apimtis, o duomenų modeliavimas: lentelių, stulpelių ir ryšių tarp jų kūrimą. Duomenų bazės modelis yra abstraktus duomenų bazės vaizdavimas, kuris leidžia nustatyti duomenų struktūrą ir sąryšius tarp jų (Beconytė, 2012). 1 pav. pateikta duomenų bazės struktūros diagrama.

Graphical user interface, application

Description automatically generated

pav. Duomenų bazės struktūra.

1 pav. parodo esančius sąryšius tarp duomenų bazės lentelių. Duomenų bazėje yra keli ryšiai: pirmas ryšis yra 1:\*, kiekvienas mokytojas galintis registruoti moksleivius, antras ryšis 1:1, kiekvienas moksleivis turi savo rezultatų lentelė priskirta jiems.

2 pav. parodo kaip laikomi administratoriai, kurie duoda/neduoda leidimus mokytojams, kai jie prisiregistruoja, atstato slaptažodžius mokytojams/moksleiviams.

Graphical user interface, text, application

Description automatically generated

A picture containing graphical user interface

Description automatically generated

pav. Administratorių lentelė.

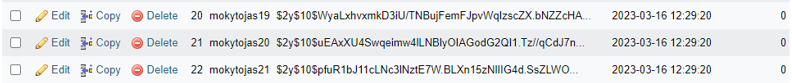
2 pav. yra tokie stulpeliai:

* „id“ – administratoriaus unikalus id;
* „username“ – administratoriaus (vartotojo) vardas;
* „password“ – administratoriaus užšifruotas slaptažodis;

3 pav. laikoma visi mokytojai, kurie užsiregistravo svetainėje ir laukia/gavo leidimą iš administratoriaus prie svetainės.

A picture containing diagram

Description automatically generated

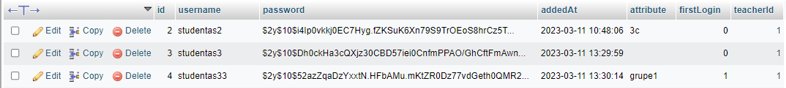


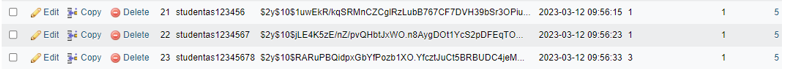
pav. Visi mokytojai prisiregistravę prie svetainės.

3 pav. yra tokie stulpeliai:

* „id“ – mokytojo unikalus id;
* „username“ – mokytojo (vartotojo) vardas;
* „password“ – mokytojo užšifruotas slaptažodis;
* „addedAt“ – kada mokytojas prisiregistravo prie svetainės.
* „isApproved“ – ar mokytojas gavo leidimą iš administratoriaus naudotis svetainė.

4 pav. laikomi moksleiviai, kurie buvo užregistruoti mokytojų svetainėje. Kiekvienas moksleivis yra priskirtas vienam mokytojui, nes mokytojas užregistruoja juos pagal norimą požymį ir mato jo rezultatus.





pav. Visi priregistruoti moksleiviai prie svetainės.

4 pav. yra tokie stulpeliai:

* „id“ – moksleivio unikalus id;
* „username“ – moksleivio (vartotojo) vardas;
* „password“ – moksleivio užšifruotas slaptažodis;
* „addedAt“ – kada moksleivis buvo užregistruotas.
* „attribute“ – mokytojo norimas požymis pagal ką moksleivis yra skirstomas.
* „firstLogin“ – nurodo ar moksleivis jau buvo prisijungęs pirmą kartą ir pasikeitęs slaptažodį.
* „teacherId“ – mokytojo id, kuris užregistravo moksleivį.

5 pav. laikomi kiekvieno moksleivio rezultatai, kai kuriuos pats moksleivis gali matyti, o visus rezultatus gali matyti jo mokytojas.

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

pav. 5 Visų moksleivių rezultatai.

5 pav. yra tokie stulpeliai:

* „id“ – moksleivio rezultatų unikalus id;
* „level“ – kuriame lygyje yra moksleivis;
* „points“ – kiek iš viso moksleivis yra surinkęs taškų;
* „luckIndicator“ – moksleivio sėkmės rodiklis.
* „studentId“ – šių rezultatų studento id.

Taigi matome kelias lentelės, kurios saugo informaciją apie mokytojus, moksleivius ir jų tarpusavio ryšius, taip pat moksleivių rezultatus. Matome, kad gerai suprojektuota duomenų bazė gali suteikti veiksmingesnį ir patikimesnį būdą saugoti ir tvarkyti duomenis. Duomenų bazė yra organizuotas duomenų rinkinys, kuris palaiko efektyvią prieigą ir atnaujinimo operacijas (Lieponienė, Ragalytė ir Paukštienė, 2006).

## Naudojami duomenys

Duomenų objektas – tai duomenys ir su jais susiję veiksmai (metodai), kuriuos galima atlikti su šiais duomenimis (Kybartas, 2015). Duomenų objektai yra pagrindinis programavimo elementas, kuris leidžia programuotojams saugoti ir manipuliuoti informacija. Duomenų objektai gali būti tiesioginis objektas, tokio tipo kaip skaičius ar tekstas, arba sudėtinis objektas, kuris susideda iš kelių tiesioginių objektų.

## Duomenų objektai

6 pav. aprašoma „Graph“ klasę, kurios objektai yra skirti saugoti duomenis apie grafiką.

Graphical user interface, text, application

Description automatically generated

pav. 6 „Graph“ klasė.

6 pav. „Graph“ klasė turi šiuos atributus:

* static „graphNumber“ – kelintas grafikas iš visų esančių;
* „path“ – kelias į grafiko paveikslėlio failą;
* „start“ – pradinė reikšmė x/y ašyje;
* „end“ – pabaigos reikšmė x/y ašyje;
* „intervalSize“ – intervalo dydis (skirtumas tarp pradinės ir pabaigos reikšmės x/y ašyje);
* „intervalStep“ – žingsnio dydis x/y ašyje;
* „graphConfig“ – masyvas, saugantis papildomus grafiko konfigūracijos parametrus;
* „intervalAnswers“ – masyvas, saugantis intervalo atsakymus;
* „inputAnswer“ – vartotojo įvestas atsakymas;
* „graphName“ – grafiko pavadinimas;
* „currentLevel“ – dabartinis lygis;
* „trueNumber“ – daliklis, kuris naudojamas gauti tikrąjį intervalo vietos skaičiu;
* „brackets“ – skliaustai atsakyme.

7 pav. gauna sudėtinį objektą iš duomenų bazės kelių lentelių, pagal kuriuos sudaro mokytojams jų priregistruotus moksleivius ir jų rezultatus lentelė:

Graphical user interface, text, application, email

Description automatically generated

pav. 7 Sudėtinis objektas iš kelių duomenų bazės lentelių.

7 pav. iš gautų duomenų sudaro HTML lentelė ir surašo į ją visus gautus ir reikalingus mokytojui duomenys apie moksleivius.

Duomenų objektai yra esminis programavimo konceptas, leidžiantis sukurti ir manipuliuoti informacija taip, kaip reikia programoje. Jie leidžia suskirstyti programos duomenis į atskirus blokus ir taip palengvina jų valdymą. Be to, duomenų objektai padeda sumažinti programavimo klaidų riziką ir palengvina programavimo procesą.

## Duomenų struktūros

Duomenų struktūra – tai duomenų organizavimo būdas. Duomenys gali būti organizuojami įvairiai: masyvuose, sąrašuose, medžiuose ir kt. Duomenų struktūrą sudaro du komponentai: duomenys ir operacijos su jais (Venčkauskas, 2007).

8 pav. aprašo „Interval“ klasę, kurios objektai yra skirti saugoti duomenis apie intervalus, naudojamus grafiko sudaryme.

A picture containing graphical user interface

Description automatically generated

pav. 8 „Interval“ klasė.

8 pav. „Interval“ klasė turi šiuos atributus:

* „open“ - kintamasis, nurodantis, ar intervalas yra atviras ar uždaras;
* „element“ - HTML elementas, reprezentuojantis intervalą;
* „position“ - intervalo pozicijos vieta pikseliais x/y ašyje;
* „labelValue“ - kintamasis, kuris nurodo intervalo reikšmė.

„Interval“ klasė yra skirta naudoti apskaičiuojant ir saugant duomenis apie intervalus ir jų pozicijas grafike.

9 pav. yra rodoma kaip būna sudaromi „Interval“ klasės objektai ir sudedami į masyvą.  
Text, application

Description automatically generated

pav. Intervalų generavimas ir įdėjimas į masyvą.

9 pav. yra sukuriamas „intervals“ masyvas, kuris saugo informaciją apie intervalus, naudojamus grafiko apskritimų esamų pozicijų nustatymui. Funkcija „generateIntervals()“ yra skirta sugeneruoti reikiamą skaičių intervalų priklausomai nuo dabartinio lygio. Funkcija prasideda gaunant intervalų konteinerio HTML elementą ir nustatant, ar reikalingi begalybės intervalai, atsižvelgiant į dabartinį lygį. Tada nustatoma, kiek intervalų bus sugeneruota, ir pradedama iteracija per šiuos intervalus. Tada yra sukuriami „Interval“ objektai, bei intervalas būna pridedamas prie intervalų konteinerio ir įdėtas į masyvą „intervals“, kurio indeksas sutampa su iteracijos numeriu. Be to, kiekvieno intervalo pozicija yra skaičiuojama, naudojant HTML elementų pozicijas ir jų metodus.

Taigi, „intervals“ masyvas yra sukurtas ir gali būti pildomas, kad būtų saugoma informacija apie sugeneruotus intervalus ir jų pozicijas, taip žinant kur apskritimai gali stovėti. Tai yra būtinas žingsnis siekiant sukurti dinaminį ir veiksmingą grafiką.

10 pav. yra sukuriamas „circles“ masyvas, kuris laiko informaciją apie „circles“, naudojamus grafiko apskritimams manipuliuoti.

A picture containing chart

Description automatically generated

pav. 10 Apskritimų sukūrimas ir sudėjimas į masyvą.

Kuriant „new Circle()“ yra duodamas HTML elementas, kuris atitinka apskritimą, tada HTML elementas, kuris atitinka linija kiekvienam apskritimui, ašį, tuomet HTML elementas, kuris nurodo intervalo reikšmė, bei HTML elementas, nurodantis apskritimo numerį ir ant kokios pozicijos šis apskritimas stovi.

## Programinis projektas

Tapimas programuotoju – tai ne tik sintaksės ir programavimo kalbos sąvokų mokymasis: tai – supratimas, kaip šias žinias panaudoti kuriant programas (Khan Academy, n.d.). Šiame projekte sukuriamas mokomasis žaidimas, skirtas moksleiviams suprasti matematines funkcijas ir jų grafikus. Žaidimas veikia internetinėje svetainėje ir naudoja duomenų bazę, kurioje yra saugomi duomenys kaip moksleivio vartotojo vardas, slaptažodis ir jo žaidimo rezultatai. Projektas yra sukurtas naudojant HTML, CSS, JavaScript ir PHP programavimo kalbas, taip pat MySQL duomenų bazių valdymo sistemą. Projekto architektūros modelis susideda iš trejų dalių, kuriame yra duomenų bazė, logikos dalis ir vartotojo sąsaja.

11 pav. pavaizduotas projekto architektūrinis modelis.

Diagram, schematic

Description automatically generated

pav. 11 Projekto architektūrinis modelis.

11 pav. parodoma kaip projektas susideda iš trejų dalių, duomenų bazės dalinasi duomenimis su logikos dalimi, o logikos dalis su vartotojo sąsaja.

Duomenų bazėje yra saugojami duomenys, bei duoda prieigą redaguoti, pridėti ir išimti juos. MySQL duomenų bazių valdymo sistema naudojama kaip duomenų bazės valdymo įrankis. Logikos dalis yra atsakinga už žaidimo taisyklių ir mechanikos įgyvendinimą, yra aprašomi veikimo algoritmai, kurie valdo mokomąjį žaidimą. Vartotojo sąsaja yra atsakinga už interakcija su vartotoju, tai yra žaidimo sąsaja, kurią moksleivis matys ir naudos žaidime, ji yra sukurta naudojant HTML, CSS ir JavaScript programavimo kalba.

Veikimo algoritmai žaidime apima mokymąsi matematikos funkcijų ir jų grafikų, pradedant nuo paprastų lygių iki sudėtingesnių. Moksleivis turės tempti apskritimus ant ašies pagal grafiko intervalus ir parašyti apibrėžimo sritį nurodytame grafike. Visų dalių funkcionalumas yra svarbus projekto veikimui ir turi būti tobulinamas, kad būtų užtikrintas sėkmingas mokomasis procesas.

## Projektavimo šablonai

Projektavimo šablonai yra skirti išspręsti dažnai pasitaikančius programavimo problemas. Jie yra naudingi tam, kad galėtumėte sukurti gerai struktūrizuotą ir lengvai išlaikomą programinę įrangą. Šie šablonai yra sukurti kaip bendros taisyklės, kurios galėtų būti taikomos įvairiose programavimo kalbose ir platformose. Projektavimo šablonas turėtų būti taikomas tik tada, kai iš tikrųjų reikia jo suteikiamo lankstumo (Gamma ir kt., 1994). Projektavimo šablonai yra naudingi programuotojams, kad galėtų sukurti tvarkingą ir išlaikomą kodą, kuris yra lengvai suprantamas ir modifikuojamas.

12 pav. yra pritaikomas „Singleton“ kūrimo šablonas. „Singleton“ yra vienas iš paprasčiausių, bet galingiausių programinės įrangos kūrimo modelių (Digital Guide IONOS, 2019).

Graphical user interface, text, application

Description automatically generated

pav. 12 „Singleton“ kūrimo šablonas.

12 pav. „Singleton“ kūrimo šablonas pritaikomas, nes testavimo klasė reikalinga tik viena, kuri pratestuoja visus galimus testus ir neefektyvu turėti daugiau nei vieną.

13 pav. yra pritaikomas „Promise“ struktūrinis šablonas, skirtas animuoti apskritimus į jų atitinkamas intervalų pozicijas. „Promise” yra objektas, kuris naudojamas asinchroniniams skaičiavimams. Tai yra operacija, kuri dar nebaigta, tačiau tikimasi, kad bus baigta ateityje. Promises suteikia kelias privalumų prieš atgalinius kvietimus: funkcine kompozicija ir klaidų tvarkymas (Lindesay, n.d.).

Graphical user interface, text, application, email

Description automatically generated

pav. 13 „Promise“ struktūrinis šablonas.

13 pav. eiga būtų: funkcija „animateInterval()“ grąžina naują pažadą, kuris aptraukia funkcijos „moveToPosition()“ iškvietimą. Funkcija „moveToPosition()“ pasirūpina atstumo, pradžios laiko ir animacijos eigos skaičiavimu bei atitinkamai atnaujina apskritimo padėtį. Pažadas išsprendžiamas, kai animacija baigta, ir pasirinktinai rodomas spustelėtas mygtukas arba išsprendžiamas neparodžius spustelėjusio mygtuko pagal „currentGraph.currentLevel“ reikšmę.

14 ir 15 pav. yra pritaikomas „Factory“ elgesio šablonas, kuris naudojamas, siekiant gauti konkretų objektą pagal nurodytus parametrus. Apibrėžiama objekto kūrimo sąsają, leidžiama poklasiams nuspręsti, kurią klasę sukurti. „Factory“ šablonas leidžia klasei atidėti egzempliorių kūrimą poklasiams (Gamma ir kt., 1994).

Text

Description automatically generated

pav. „Factory“ elgesio šablonas.

Graphical user interface, text, application

Description automatically generated

pav. Funkcija nustatanti HTML elementą.

14 ir 15 pav. „Factory“ šablonas yra naudojamas sukurti grafiko ašį, bei visus reikalingus HTML elementus jam, tokius kaip intervalų linijos, apskritimai, intervalų žymėjimai ir pan. Šablonas leidžia kurti grafiko ašį su skirtingais parametrais ir taip palengvina kodo priežiūrą bei išplėtimą ateityje.

Kiekvienas projektavimo šablonas yra aprašytas konkrečiu būdu, kuris apima savo tikslus ir pritaikymo būdus. Šie šablonai gali būti pritaikyti įvairiose programavimo kalbose ir projektavimo srityse. Taigi projektavimo šablonai yra labai naudingi įrankiai kuriant programinės sistemas bei programinė įranga.

# PROGRAMŲ SISTEMOS REALIZACIJA

Programų sistemų realizacijos procesas yra sudėtingas ir dažnai susijęs su dideliu rizikos lygiu. Todėl reikia taikyti tinkamas metodologijas ir technologijas, kurios padėtų užtikrinti kokybišką ir efektyvų programavimą (Butleris, Nemuraitė ir Matulevičius, 2010). Programos realizacijoje bus reikalinga naudoti duomenų bazės užklausas, tokias kaip gauti visus moksleivius mokytojui. Taip pat bus reikalingi algoritmai, tokie kaip mokytojo registracija, tempiamo apskritimo intervalo pozicijos nustatymas, apskritimo judėjimo animacijos į intervalo poziciją algoritmai. Galiausiai grafinė naudotojo sąsaja, kuri padės vartotojamas interaktyviai naudotis šią svetainė.

## JPA realizavimas

JPA (Java Persistence API) yra Java technologijų rinkinys, skirtas lengvai atlikti objektišką duomenų bazės pasiekimą ir valdymą. Pagrindinė JPA funkcija, skirta programų perkeliamumui spręsti, yra galimybė priskirti duomenų bazių lenteles į klases (Coelho ir Kiourtzoglou, n.d.). JPA buvo sukurtas kaip atsakas į sunkumus, susijusius su tradiciniais duomenų bazės pasiekimo būdais. JPA realizavimas yra plačiai naudojamas moderniose Java programavimo projektuose dėl savo galimybių lengvai pasiekti ir valdyti duomenų bazę per Java objektus, kas padidina programavimo produktyvumą ir pagerina sistemos priežiūros ir plečiamumo galimybes.

Kadangi šiam projektui nebuvo naudojama Java programavimo kalba, taigi teko naudoti alternatyvų būdą PHP kalbai duomenų bazės pasiekimui ir valdymui, tai parodyta 16 pav.

Text

Description automatically generated

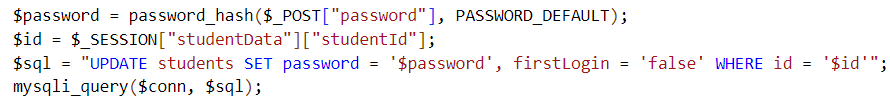
pav. Duomenų bazės pasiekimas PHP kalboje.

16 pav. vyksta prisijungimas prie duomenų bazės su PHP programavimo kalba, iš pradžių nustatomas laiko zona bei seanso pradžia, nustatomas serverio pavadinimas, prieigos vartotojas, slaptažodis ir duomenų bazės pavadinimas, tada prisijungiama prie jos, jei įvyksta problema prisijungiant, tai parodoma kokia klaida buvo gaunama neprisijungiant.

Apibendrinant, alternatyvus pakeitimas JPA PHP programavimo kalboje yra veiksmingas ir efektyvus būdas pasiekti duomenų bazė bei ją naudotis.

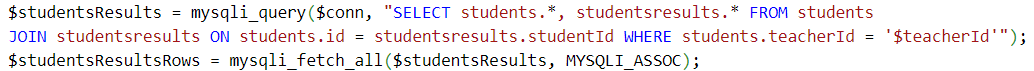
## DB užklausos

Duomenų bazės užklausos yra sakiniai, kuriais nurodoma, kokius duomenis norima gauti iš duomenų bazės. Užklausos gali būti paprastos arba sudėtingos, priklausomai nuo to, kiek ir kokios rūšies duomenų reikia (Laukaitis, Butleris ir Nemuraitė, 2008). 17 ir 18 pav. parodomos kokios užklausos yra naudojamos.



pav. DB užklausa pakeisti moksleivio slaptažodį, kai jis prisijungia pirmą kartą.

17 pav. užklausa pakeičia moksleivio slaptažodį duomenų bazėje, kai jis prisijungia pirmą kartą prie savo priregistruotos paskyros ir pasikeičia slaptažodį.



pav. Sudėtinė DB užklausa gauti visus studentus ir jų rezultatus pagal mokytojo id.

18 pav. sudėtinė užklausa gauna visus studentus ir jų rezultatus, kurie buvo užregistruoti tam tikro mokytojo ir tada parodo visą lentelė mokytojui.

Pagal duomenų bazės užklausų paveikslėlius matome, kad duomenų bazės užklausos yra svarbi dalis duomenų bazės valdyme. Jos yra būtinos priemonės, kad užtikrintų duomenų saugumą ir apsaugotų nuo neleistino prieigos prie jų.

## Algoritmai

Geras algoritmas – kaip aštrus peilis – atlieka tiksliai tai, ką turi atlikti, su minimaliomis pastangomis (Petrauskas ir Skūpienė, 2006). Algoritmai yra svarbus ir neišvengiamas kompiuterijos bei informacinių technologijų elementas. Tai yra taisyklių rinkinys, skirtas išspręsti tam tikrą problemą. Jie leidžia optimizuoti informacijos apdorojimo procesą, kad būtų gauti norimi rezultatai.

19 pav. yra mokytojų registracijai ir įdėjimas į duomenų bazė algoritmas.

A picture containing graphical user interface

Description automatically generated

pav. 19 Mokytojo registracijos algoritmas.

19 pav. algoritmas suveikia kai registracijos mygtukas būna paspaudžiamas, patikrinami ar yra tušti laukai: vartotojo vardas, slaptažodis, slaptažodžio patvirtinimas, jei ne tai duodamas perspėjimas, kad visi laukai turi būti užpildyti. Toliau tikrina ar slaptažodis yra ne trumpesnis nei 6 simbolių ilgumo, tikrina ar slaptažodis su slaptažodžiu patvirtinimu yra vienodi. Tuomet tikrina ar iš jau esančių mokytojų yra tokių pačių vartotojų vardai, jei viskas tvarkoje, tada užšifruoja slaptažodį ir siunčia užklausą į duomenų bazė įrašyti šį mokytoją.

20 pav. yra nustatoma poziciją tempiamo apskritimo ant artimiausio esančio intervalo algoritmas.

Text

Description automatically generated

pav. 20 Tempiamo apskritimo intervalo pozicijos nustatymo algoritmas.

20 pav. algoritmas suveikia, kai apskritimas būna tempiamas, tada iteruojamasi per visus esančius intervalus, tikrinami ar jie tušti(be kito apskritimo jame). Tuomet gaunamos pozicijos intervalo kairės ir dešinės vieta pikseliais, patikrinama ar naujai tempiama pozicija yra tarp šių pozicijų, jei taip tada praeitas intervalas, kuriame buvo apskritimas atsilaisvina ir užimamas būna naujai užimtas intervalas. Jei apskritimas buvo nutemptas į kurį nors begalybės intervalą, jis pakeičia tipą į tuščiavidurį apskritimą ir grąžina dabartinę apskritimo poziciją intervale bei jo etiketės reikšmę. Jei iš visų esančių intervalų nėra randamas naujas intervalas, kuriame apskritimas turi būti, tai jis nebūna judinamas ir grąžinama „NONE“ reikšmė.

21 pav. yra algoritmas, kuriame apskritimas daro animacijos judėjimą į tam tikrą intervalą.

Graphical user interface, text, application, email

Description automatically generated

pav. Apskritimo judėjimo animacijos į intervalo poziciją algoritmas.

21 pav. funkciją „animateInterval()“ būna iškviečiama, kai yra rodomas lygio pavyzdys arba moksleivis norintis pamatyti kaip reikia išspręsti lygyje esančią užduotį. Kai tai įvyksta būna iškviečiama funkciją „moveToPosition()“, kurioje gaunamą pradinė apskritimo poziciją, tada kur turės intervalas būti poziciją, atstumas tarp jų ir dabarties laikas. Toliau prasideda žingsnio funkciją, kuri save vykdo sinchroniškai 1500 milisekundžių ir laiko progresą. Tada patikrinama ar progresas jau pasiekė pabaigą ir ar apskritimas pasiekė reikiamą intervalo poziciją, jei ne tai apskritimas po truputį juda link tos pozicijos, o jei jau pasiekė, tada sutvarkoma tolygiai apskritimo poziciją ant šio intervalo ir patikrinama ar jau yra lygis, kuriame gali reikėti keisti apskritimo tipą, ir jis būna pakeičiamas jei reikalinga ir išeinama iš funkcijos.

Pateikti algoritmai yra konkrečios programinės įrangos dalys, kurios vykdo tam tikras užduotis. Šie algoritmai yra labai svarbūs, kad programinė įranga galėtų teisingai ir efektyviai atlikti savo funkcijas. Taigi, algoritmai yra programavimo pagrindas, be kurių sunku įsivaizduoti bet kokią sėkmingą programų kūrimo veiklą.

## Grafinė naudotojo sąsaja

Sudėtinė grafinė naudotojo sąsaja (GUI - graphical user interface) yra projekto dalis, kuri leidžia naudotojui vizualiai matyti ir manipuliuoti duomenys. Tai yra esminė dalis mokomajame žaidime, kuri suteikia galimybę moksleiviams išmokti matematikos ir grafikų interpretavimo. GUI yra sukurtas naudojant HTML, CSS ir JavaScript technologijas. Ši sąsaja leidžia moksleiviams išsiaiškinti kur turi būti apskritimai pažymėti grafike, bei sužinoti visokius grafike esančius žymėjimus ir parašyti apibrėžimo sritį nurodytame grafike. Be to, GUI suteikia galimybę mokytojams sekti moksleivių pažangą ir turėti jų rezultatus. Visa informacija apie moksleivius ir jų pažangą yra laikoma duomenų bazėje, kuri yra valdoma su PHP ir MySQL.

22 pav. pavaizduotas pradinis puslapis.

Graphical user interface

Description automatically generated

pav. 22 Registracijos mokytojui ir prisijungimas vartotojams.

22 pav. mokytojai gali užsiregistruoti, taip pat visi vartotojai gali prisijungti(administratoriai, mokytojai, moksleiviai, ir svečiai).

23 pav. yra pavaizduotas administratoriaus puslapis.

Graphical user interface

Description automatically generated

pav. 23 Administratoriaus puslapis kai jis prisijungia.

23 pav. administratorius kai prisijungia mato laukiančius užtvirtinimo mokytojus, užtvirtintus mokytojus, gali atstatyti jų slaptažodžius, ir taip pat visus moksleivius bei gali atstatyti jų slaptažodžius.

24 ir 25 pav. pavaizduotas moksleivio puslapis.

A picture containing graphical user interface

Description automatically generated

pav. 24 Lygių meniu.

Chart

Description automatically generated

pav. Lygio pavyzdys.

24 pav. moksleivis kai prisijungia būna perkeliamas į lygių meniu, kur pasirenka lygį pradedama nuo pirmojo lygio, kiti lygiai atsidaro kai moksleivis pereina praeitus. 25 pav. yra trečiojo lygio pavyzdinė užduotis.

Taigi, ši sudėtinė grafinė naudotojo sąsaja yra esminis projektų elementas, kuris suteikia naudotojams galimybę vizualiai matyti ir interaktyviai naudotis šią sąsają.

# PROGRAMŲ SISTEMOS KOKYBĖS UŽTIKRINIMAS

Kokybė turi būti primesta, kitaip jos nebus. Mes, programuotojai, turime būti priversti rašyti testus, kitaip to nepadarysime (Bugayenko, 2016). Programų sistemų kokybės užtikrinimas apima daugybę skirtingų veiksmų, tokių kaip programinio kodo testavimas, atliekamų funkcijų ir užduočių tikrinimas, klaidų paieška ir taisymas, naudotojų patirties gerinimas, saugumo užtikrinimas ir kt.

## Testavimas

Vienetinis testavimas nėra skirtas klaidų paieškai. Tai yra įrodymas, kad mūsų kodas daro tai, ką norėjome (Khorikov, 2020). Testavimas yra svarbus programinio kodo dalis, kuri padeda užtikrinti, kad kodas yra funkcionuojantis ir atitinka reikalavimus. Tai yra procesas, kai testai yra sukurti siekiant patikrinti, ar programinis kodas elgiasi taip, kaip turėtų, ir yra kokybiškas. Keli pavyzdžiai iš šios svetainės testavimo:

26 pav. testas ištestuoja metodo findNearestIntervalLabel() našumą.

Text

Description automatically generated with medium confidence

pav. Našumo testas metodui.

26 pav. findNearestIntervalLabel() metodas būna panaudojamas 100 000 kartų su atsitiktinėmis reikšmėmis ir grąžina kiek laiko tai užtrunka.

27 pav. testas ištestuoja metodą animateInput().

A picture containing chart

Description automatically generated

pav. 27 Griežtas lygybės testas metodui.

27 pav. animateInput() metodas būna ištestuojamas ar į įvestį surašomas duotas tekstas bei tekstą parašo per tam tikrą laiko tarpą.

Testavimas yra svarbus programinio kodo procesas, skirtas užtikrinti, kad kodas veiktų teisingai ir atitiktų reikalavimus. Tai ne tik patikrinimo proceso etapas, bet ir įrodymas, kad kodas daro tai, ko tikimasi. Norint užtikrinti programinio kodo kokybę, reikia kurti testus, kurie tikrintų, ar kodas elgiasi taip, kaip turėtų, ir ar jis yra efektyvus. Šiame projekte neprivaloma turėti testavimo anotacijų, nes šio programinio kodo kokybę galima užtikrinti kruopščiai kuriant testus, tokius kaip našumo ar elgsenos testai, gali padėti užtikrinti, kad kodas yra kokybiškas ir atitinka reikalavimus.

## Kodo versijų kontrolė

Kodo versijų kontrolė yra procesas, kuris leidžia programuotojams sekti ir valdyti savo programinio kodo versijas. Tai yra būtina dalis programavimo darbo, kadangi programų kūrimas paprastai vyksta per kelias etapas ir per keletą dienų ar net savaičių. Versijų kontrolė leidžia programuotojams saugoti ir peržiūrėti kodo istoriją, kuri apima pakeitimus, atnaujinimus ir kitas modifikacijas.

28 pav. matoma, kad kodo pateikimas į github buvo sklandus ir gana dažnas, kai kuriomis dienomis daugiau nei vienąsykį.

A picture containing graphical user interface

Description automatically generated

pav. Github kodo versijų kontrolė.

Versijų valdymas yra sistema, kuri įrašo failo ar failų rinkinio pakeitimus laikui bėgant, kad vėliau galėtumėte prisiminti konkrečias versijas (Chacon ir Straub, 2014). Kodo versijų kontrolė yra svarbi dalis programavimo darbo ir leidžia programuotojams išsaugoti savo kodo istoriją, patobulinti bendradarbiavimą ir tvarką bei išlaikyti kodo saugumą. Nuoroda: <https://github.com/just-renis>

# IŠVADOS

1. Projektuojant programų sistemą buvo nustatyta, kad bus reikalinga mažiausiai 3 objektų, mažiausiai 5 algoritmų ir 3 projektavimo šablonų. Suprojektuojant programų sistemą buvo įsitikinta, kad tai padeda pagreitinti ir pagerinti programų sistemos realizavimą.
2. Atlikus realizaciją buvo padaryta MySQL duomenų bazė, suprogramuoti reikiami algoritmai ir grafinę naudotojo sąsają su HTML, CSS, JS, PHP. Realizuojant programų sistemą buvo įsitikinta, kad užtenka JS ir PHP kuriant ir įgyvendinant šią programų sistemą.
3. Padarius automatinius testus ir naudojant versijų kontrolės įrankį Github buvo nustatyta, kad programų sistemą veikia bet kuriame scenarijuje kaip turėtų veikti. Kuriant automatinius testus buvo sužinota, kad jie yra reikalingi užtikrinant programos sistemos valdymą. Versijų kontrolei buvo naudojamas įrankis Github, kuriame buvo keliamos šios programų sistemos versijos bei yra galimybė atkurti iš Github šią programų sistemą.

|  |
| --- |
|  |

# LITERATŪROS IR KITŲ INFORMACIJOS ŠALTINIŲ SĄRAŠAS

1. Beconytė, G. (2012). *Duomenų bazių valdymas.* Vilnius: Technika.
2. Bugayenko, Y. (2016). *Elegant Objects, Volume 1.* CreateSpace Independent Publishing Platform.
3. Butleris, R., Nemuraitė, L. ir Matulevičius, R. (2010). *Programavimo metodologijos: nuo vandens krioklio iki agilinio modelio. Informacijos mokslai, 53,* 5.
4. Chacon, S. ir Straub, B. (2014). *Pro Git.* New York City: Apress.
5. Coelho, H. ir Kiourtzoglou, B. (n.d.). *Java Persistence API Mini Book.* Prieiga per: <https://enos.itcollege.ee/~jpoial/java/naited/JPA_Mini_Book.pdf>
6. Digital Guide IONOS (2019). *The singleton pattern – a class in itself.* Prieiga per: <https://www.ionos.com/digitalguide/websites/web-development/singleton-design-pattern/>
7. Gamma, E., Helm, R., Johnson, R. ir Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software.* Boston: Addison-Wesley Professional.
8. Khan Academy (n.d.). *Planning a programming project.* Prieiga per: <https://www.khanacademy.org/computing/computer-programming/programming/good-practices/a/planning-a-programming-project>
9. Khorikov, V. (2020). *Unit testing: Principles, Practices, and Patterns.* Shelter Island, New York: Manning Publications.
10. Kybartas, R. (2015). *Duomenų struktūrų objektinio programavimo praktikumas.* Kaunas: Kauno technologijos universiteto leidykla.
11. Laukaitis, A., Butleris, E. ir Nemuraitė, L. (2008). *Duomenų bazių projektavimas ir programavimas.* Vilnius: Vilniaus universiteto leidykla.
12. Lieponienė, J., Ragalytė, B. ir Paukštienė, A. (2006). *Duomenų bazės projektavimas*. Vilnius: Ciklonas.
13. Lindesay, F. (n.d.). *Patterns.* Prieiga per: <https://www.promisejs.org/patterns/>
14. Petrauskas, L. ir Skūpienė, J. (2006). *Informatikos olimpiados: algoritmai ir taikymo pavyzdžiai.* Prieiga per: <https://siom.lmio.lt/m/t/knyga.pdf>
15. Venčkauskas, A. (2007). *Duomenų struktūros, algoritmai ir jų analizė.* Vilnius: Technika.

# PRIEDAI

1 lentelė. Minimalių reikalavimų lentelė

|  |  |
| --- | --- |
| **Minimalūs reikalavimai:** | **Pildo dėstytojas** |
| Laikomasi kodo **vardinimo taisyklių** |  |
| **Ataskaita** be gramatinių ar formatavimo klaidų |  |
| **Gitlab** pateiktas kodas ir ataskaita |  |
| Yra ataskaita su užpildytais visais duoto šablono skyriais |  |
| Kiekviename ataskaitos skyriuje yra aiškiai nurodyta kurioje kodo dalyje yra realizuotas rezultatas |  |

2 lentelė. Vertinimo kriterijų lentelė

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Vertinama dalis (skyrius ataskaitoje)** | **Vertė** | **Slenkstinis (5-6)** | **Tipinis (7-8)** | **Puikus (9-10)** |
| **Duomenų bazės projektavimas (1.1)** | 5 % | DB su bent 3 lentelėmis (po min 20 įrašų po min 3 laukus). | DB su bent 4 lentelėmis (po min 20 įrašų po min 3 laukus). Panaudoti kelių tipų ryšiai. | Yra skaitymas iš DB su bent 4 lentelėmis (po min 20 įrašų po min 3 laukus). Panaudoti kelių tipų ryšiai.  Sutvarkytos lietuvių kalbos rašmenų koduotės. |
| **Duomenų objektai (1.2.1)** | 5 % | Dirbama bent su vienu duomenų objektu sudarytu iš min 3 savybių | Dirbama su keliais duomenų objektais arba vienu sudėtiniu duomenų objektu | Dirbama su keliais duomenų objektais ir bent vienas yra sudėtinis duomenų objektas |
| **Duomenų struktūros (1.2.2)** | 10 % | Parinkta viena duomenų struktūra ir pagrįstas jos tinkamumas | Panaudotos kelios duomenų struktūros, apibrėžta sava jų kombinacija. Arba yra galimybė papildyti naujais objektais. | Panaudotos kelios duomenų struktūros, apibrėžta sava jų kombinacija. Ir yra galimybė papildyti naujais objektais. |
| **Programinis projektas (1.3)** | 5 % | Aprašytas programinis projektas ir naudojamos technologijos | Aprašytas programinis projektas ir naudojamos technologijos, nurodytas pilnas projekto architektūrinis modelis | Aprašytas programinis projektas ir naudojamos technologijos, nurodytas pilnas projekto architektūrinis modelis. Aprašyti veikimo algoritmų modeliai. |
| **Projektavimo šablonai (1.4)** | 10 % | Programos kode pritaikyta po 1 kūrimo, struktūrinį bei elgesio projektavimo šabloną | Programos kode pritaikyti 5 kūrimo, struktūrinį bei elgesio projektavimo šablonai | Programos kode pritaikyti 7 (po minimum 2 kūrimo, struktūrinį bei elgesio) projektavimo šablonai |
| **JPA realizavimas (2.1)** | 5 % | Sukonfiguruota JPA, naudojant Hibernate arba alternatyvų karkasą. Sukurta bent viena esybė ir atliekamos CRUD operacijos. | Sukonfiguruota JPA, naudojant Hibernate arba alternatyvų karkasą. Sukurtos bent viena sudėtingesnė esybių struktūra (viena esybė sudaryta iš kelių kitų, viena esybė privalomai turi turėti kitą esybę ir pan.). | Sukonfiguruota JPA, naudojant Hibernate arba alternatyvų karkasą. Sukurtos kelios sudėtingesnės esybių struktūros (viena esybė sudaryta iš kelių kitų, viena esybė privalomai turi turėti kitą esybę ir pan.) |
| **DB užklausos (2.2)** | 7 % | Atliekamos visos CRUD operacijos. | Atliekamos sudėtinės užklausos apimančios kelis parametrus arba kelias DB lenteles | Atliekamos sudėtinės užklausos apimančios kelis parametrus ir kelias DB lenteles |
| **Algoritmai (2.3)** | 8 % | Atliekama viena operacija iš:  Elementų paieška kolekcijoje,  Elementų atranka (filtravimas) kolekcijoje,  Elementų rūšiavimas kolekcijoje | Atliekamos dvi operacijos iš:  Elementų paieška kolekcijoje,  Elementų atranka (filtravimas) kolekcijoje,  Elementų rūšiavimas kolekcijoje | Atliekamos trys operacijos:  Elementų paieška kolekcijoje,  Elementų atranka (filtravimas) kolekcijoje,  Elementų rūšiavimas kolekcijoje |
| **Grafinė naudotojo sąsaja  (2.4)** | 10 % | Yra grafinė naudotojo sąsaja, atvaizduojanti duomenis ir leidžianti manipuliuoti jais. | Yra sudėtinė grafinė naudotojo sąsaja, atvaizduojanti duomenis ir leidžianti manipuliuoti jais. | Yra kelios skirtingos sudėtinės grafinės naudotojo sąsajos (mobile app, web, kt.), atvaizduojančios duomenis ir leidžiančios manipuliuoti jais. |
| **Testavimas (3.1)** | 4 % | Sukurtas kodas testuojamas automatiniais testais (padengimas 20 proc.) | Sukurtas kodas testuojamas automatiniais testais (padengimas 50 proc.). | Sukurtas kodas testuojamas automatiniais testais (padengimas 70 proc.) |
| 3 % | Yra panaudoti bent 3 rūšių assert metodai. | Yra panaudoti bent 4 rūšių assert metodai. | Yra panaudoti bent 5 rūšių assert metodai. |
| 3 % | Yra panaudotos bent 3 rūšių anotacijos. | Yra panaudotos bent 4 rūšių anotacijos. | Yra panaudotos bent 5 rūšių anotacijos. |
| 4 % | Realizuota viena iš testavimo kategorijų:  Išimčių testavimas, Performance testavimas,  Parametrizuoti testai. | Realizuotos dvi iš testavimo kategorijų: Išimčių testavimas, Performance testavimas,  Parametrizuoti testai. | Realizuotos trys iš iš testavimo kategorijų:  Išimčių testavimas, Performance testavimas,  Parametrizuoti testai. |
| **Kodo versijų kontrolė (3.2)** | 8 % | Minimum 25% kassavaitiniai tarpiniai kodo pateikimai | Minimum 50% kassavaitinių tarpinių kodo pateikimų | Minimum 75% kassavaitinių tarpinių kodo pateikimų |
| **Sprendimų pagrindimas (Informacinių šaltinių sąrašas)** | 13 % | Projektuojant ir realizuojant pacituoti bent 5 moksliniai šaltiniai | Projektuojant ir realizuojant pacituoti bent 8 moksliniai šaltiniai | Projektuojant ir realizuojant pacituoti bent 10 mokslinių šaltinių |