Latvijas Republikas izglītības un zinātnes ministrija Daugavpils Tehnoloģiju un tūrisma tehnikums

PROJEKTA DARBS

Programmēšanas tehniķis

specialitāte

1. projekta darbs

temats

Testēšanas sistēmas izveidošana

PASKAIDROJOŠAIS RAKSTS

**DTTT.P.D.00018.018PR**

Profesijas kods 33 481 031 Programmēšanas nodaļa

PR-21 grupa

|  |  |  |
| --- | --- | --- |
| Izveidoja | paraksts | O.Nikitins  V.,Uzvārds |
| Pārbaudīja | paraksts | I. Dortiņa  V.,Uzvārds |

2024./2025. m.g.

**Saturs**

[Ievads 3](#_Toc193962522)

[1. Uzdevuma nostādne 4](#_Toc193962523)

[1.1. Sistēmas modelis 4](#_Toc193962524)

[1.1.1. Priekšmetiskās jomas informācijas modelis 4](#_Toc193962525)

[1.1.2. Sistēmas apkārtnes shēma 4](#_Toc193962526)

[1.1.3. Viedokļu analīze 5](#_Toc193962527)

[1.1.4. Datu modelēšana 6](#_Toc193962528)

[1.1.5. Datu vārdnīca 6](#_Toc193962529)

[1.2. Sistēmas evolūcija 7](#_Toc193962530)

[1.3. Sistēmas funkcionālās prasības 7](#_Toc193962531)

[1.4. Sistēmas nefunkcionālās prasības 9](#_Toc193962532)

[2. Priekšmetiskās jomas klašu sistēmas izstrāde 12](#_Toc193962533)

[2.1. Klašu diagramma 12](#_Toc193962534)

[2.2. Klašu realizēšana Java valodā 13](#_Toc193962535)

[3. Testa programmas izstrāde 74](#_Toc193962536)

[3.1. Testēšanas metodikas 74](#_Toc193962537)

[3.2. Testēšanas programmas struktūra 74](#_Toc193962538)

[3.3. Testēšanas rezultāti 76](#_Toc193962539)

[4. Lietotāja rokasgrāmata 77](#_Toc193962540)

[4.1. Lietotāja instrukcija 77](#_Toc193962541)

[4.2. Palīdzības sistēma (Help) 77](#_Toc193962542)

[4.3. Sistēmas ziņojumi 77](#_Toc193962543)

[Secinājumi 77](#_Toc193962544)

[Izmantotās literatūras (informācijas avotu) saraksts 79](#_Toc193962545)

[Pielikums A. Klašu sistēma 81](#_Toc193962546)

[Pielikums B. testa programmas kods 82](#_Toc193962547)

Ievads

Projekta darbā ir aprakstīta testēšanas programma, kura izpilda sekojošas funkcijas:

* lietotāju reģistrēšana
* datu ievade
* datu saglabāšana
* datu rediģēšana
* testa pildīšana
* rezultātu aprēķināšana

Sistēma nodrošina audzēkņa testēšanu par noteiktu tēmu/vielu, tas ir domāts, lai skolotājs varētu viegli izlikt atzīmi skolēnam digitālā veidā attālināti vai klātienē. Tāpat šī programma novērš cilvēcisko faktoru, piemēram, noapaļošanu noteiktā virzienā.

# Uzdevuma nostādne

## Sistēmas modelis

### Priekšmetiskās jomas informācijas modelis

Apskatāmā sistēmā tika izdalīti objekti: Datu bāze (Fails ar atbildēm), Tests, Rezultāti, Administrators / Skolotājs, Lietotājs. Šie objekti un saites starp tiem ir paradīti ([1.1. att.](#att1_1)).

Datu bāze

Administrātors/Skolotājs

Tests

Rezultāti

Lietotājs

1.1 att. Vienkāršots testēšanas sistēmas modelis

Saišu nozīme vienkāršotā nodaļas modelī:

* Skolotājs ir testa veidotājs un pārbaudītājs, kuram ir pieeja pie atbildēm, testam un lietotāja rezultātiem;
* Datu bāze satur failus ar jautājumiem un ar pariezām atbildēm;
* Lietotājs ir testa pildītājs, kas beigās var apskatīt savus rezultātus, nepieciešamības gadījumā viņš varēs sazināties ar skolotāju;
* Tests ir instruments, kas palīdzēs pārbaudīt lietotāju zināšanas un novērtēt to ar atzīmi;
* Rezultāti parāda atzīmi, pareizās un nepareizās atbildes skaitliski un procentuāli.

### Sistēmas apkārtnes shēma

Dati, kā atbildes, ko ievada audzēknis tiek saglabāti failā. Tomēr dažreiz atbildes netiek automātiski novērtētas un skolotājam ir iespēja manuāli novērtēt iesniegto atbildi. Saskarni ar lietotājiem nodrošinās interfeisa modulis ([1.2. att.](#att1_2)).

Lietotājs

Lietotāja saskarne

Sistēma

Datu bāze

Lietotāja dati un rezultāti

1.2. att. Sistēmas apkārtnes modeļa piemērs

### Viedokļu analīze

Savāktie un identificētie viedokļi par sistēmu ir attēloti ar burbuļdiagrammas palīdzību ([1.3. att.](#att1_3)), tas faktiski ir viedokļu kopums, kur katra viedokļa nosaukums ir ierakstīts atsevišķa elipsē.

1.3. att. Viedokļu burbuļu diagramma

Nefunkcionalie viedokli

Funkcionalie viedokli

Viedokli par lietotaju

Datu viedokļi

Otrais etaps viedokļu analīzē ir viedokļu klasifikācija, kad viedokļi ir sagrupēti grupās ([1.4. att.](#att1_4)).

1.4. att. Viedokļu hierarhiskā struktūra

Testēšanas sistēma (V0)

Datu bāze (V12)

Lietotājs (V11)

Tests (V13)

Lietotāju registrt (V121)

Skolotājs (V111)

Testa jautajumu registrs (V122)

Audzēknis (V112)

Audzekņa funkcijas (V132)

Skolotāja funkcijas (V131)

Testa pildīšana (V1321)

Testa izveide (V1311)

Testa rezultātu saņemšana (V1322)

Testa manuāla novērtēšana (V1312)

### Datu modelēšana

Datu modelēšanai ir tiek izmantots ER modelis. Testēšanas sistēmas ER modelis ([1.5. att.](#att1_5)) ietver datu entītiju kopumu un relāciju kopumu starp datu entītijām, pie kam ar šo modeli var attēlot dažāda tipa relācijas starp datu elementiem.

Audzēknis

Pilda

Tests

Skolotājs

Izveido

Jautājums

Satur

1.5. att. Vienkārsots testēšanas sistēmas ER modelis

### Datu vārdnīca

|  |  |  |
| --- | --- | --- |
| **Datu tips** | **Nosaukums** | **Apraksts** |
| **String** | **name** | Audzēknis ievada savu vārdu un uzvārdu |
| **String** | **questionText** | Neliels teksta apgabals, kas apraksta jautājumu |
| **String[]** | **answersText** | Atbilžu variantu masīvs. Tiek piedāvāti vairāki atbilžu  varianti, kur ir jāizvēlas viens pareizs |
| **int** | **correctAnswer** | Aile, kas nosaka, vai tika izvēlēts pareizs atbildes variants |
| **float** | **result** | Audzēkņa saņemto punktu skaits (0-100) |
| **int** | **mark** | Audzēkņa atzīme (0-10) |

## Sistēmas evolūcija

Testēšanas sistēma ir neatkarīga un pašpietiekama programma. Produktam nav paredzētas vairākas sastāvdaļas.

Programmai ir trīs ārējās saskarnes, kuras var apskatīt 1.4. nodaļā – viena paredzēta reģistrēšanai sistēmā, otrā – ielogošanai sistēmā un trešā – testa pildīšanai. Grafisko lietotāja saskarni var pilnveidot, papildinot to ar dažādām krasu un noformējuma shēmām.

Testēšanas sistēmā paredzēti divu lietotāju veidi: administrators un lietotājs, kurš pilda testu. Nākotnē plānots pievienot vēl vienu lietotāju grupu ar funkciju veidot jaunus testus un pārbaudīt izpildītos testus, līdz ar to lietotājiem būs divas lomas: audzēknis, kurš pilda testu un skolotājs, kurš veido un pārbauda testus. Administratoram plānots realizēt funkciju mainīt lietotāja vārdu, uzvārdu, lietotājvārdu vai paroli, ka arī pievienot un dzēst lietotājus un mainīt vīnu lomas.

Programma satur vienu testu, kurā ir 10 jautājumi, tomēr nākotnē sistēmu var pilnveidot, pievienojot jaunas testa tēmas, kļūdu uzrādīšanu un kļūdu skaidrojumu pēc testa izpildes, lietotāju reitingu, lietotāju datu izvadi lietotāja izvēlnē vai administratora izvēlnē.

Sistēmas dažiem datu tipiem tiks izmantoti dati, kuri tiek saglabāti teksta failā, nākotnē programmu var pilnveidot darbam ar datu bāzi (datu bāzes savienojums, testa jautājumi un atbildes uz tiem u.c.)

## Sistēmas funkcionālās prasības

1. **Funkcija “Reģistrēties sistēmā”**

***ID: T1***

**Ievads**: Ļauj ievadīt nepieciešamus datus, lai izveidotu savu kontu. Visi ievadīti dati tiek saglabāti failā (datu bāzē).

Ievade:

1. Lietotāja vārds
2. Lietotāja uzvārds
3. Lietotāja lietotājvārds
4. Lietotāja parole
5. Ievadītās paroles apstiprināšana.

**Apstrāde**:

1. Lietotāja vārda saglabāšana failā.
2. Lietotāja uzvārda saglabāšana failā.
3. Lietotāja lietotājvārda saglabāšana failā.
4. Lietotāja paroles saglabāšana failā.

Ja ievadītā parole ir nepareiza, tad tiek atvērts kļūdas dialoglodziņš.

**Izvade**: Dialoglodziņš, kurā uzrakstīts “Reģistrēšana ir veiksmīgi pabeigta!”.

1. **Funkcija “Ielogoties sistēmā”**

***ID: T2***

**Ievads**: Ļauj ievadīt nepieciešamus datus, lai ieiet sava kontā.

**Ievade**:

1. Lietotāja lietotājvārds
2. Lietotāja parole

**Apstrāde**:

1. Lietotāja lietotājvārda pārbaude un atrašana DB.
2. Lietotāja paroles pārbaude un atrašana DB.

Ja ievadītā parole vai/un lietotājvārds ir nepareiza, tad tiek atvērts kļūdas dialoglodziņš.

**Izvade**: Dialoglodziņš, kurā uzrakstīts “Ieeja ir veiksmīgi pabeigta!”.

1. **Funkcija “Sākt testu”**

***ID: T3***

**Ievads**: Ļauj administratoram sākt testu.

**Ievade**: Administrators noklikšķina uz pogu “Sākt testu”. Pēc tam sistēma uzdod jautājumu: "Vai Jūs vēlaties sākt testu?" Administratoram ir divi varianti atbildēm "Jā" vai "Nē".

**Apstrāde**: Sistēma pārbauda atbildi uz uzdoto jautājumu un pēc tam sāk testu, ja uz uzdoto jautājumu atbilde "Jā". Tad tests ir pieejams lietotājiem to veikšanai.

**Izvade**: Administratora ekrānā tiek parādīts dialoglodziņš ar tekstu "Tests ir atļauts!"

1. **Funkcija “Pildīt testu”**

**I*D: T4***

**Ievads**: Ļauj lietotājam sākt testa izpildi.

**Ievade**: Lietotājs noklikšķina uz pogu “Pildīt testu”.

**Apstrāde**: Ja tests ir pieejams izpildei, tad pēc testa izvēles tiek parādīts dialoglodziņš ar tekstu "Tests ir pieejams! Vai vēlaties izpildīt testu?". Lietotājam ir divi varianti atbildēm "Jā" vai "Nē". Ja tests nav pieejams izpildei, tad pēc testa izvēles tiek parādīts dialoglodziņš ar tekstu "Tests nav pieejams!"

**Izvade**: Lietotājā ekrānā tiek parādīts dialoglodziņš ar uzrakstu "Veiksmi testā!"

1. **Funkcija “Atbildēt uz testa jautājumu”**

***ID: T5***

**Ievads**: Ļauj ievadīt atbildi uz testa jautājumus.

**Ievade**: Lietotājs izvēlas 1 no 4 piedāvātajām atbildēm uz uzdoto jautājumu.

**Apstrāde**: Sistēma pārbauda atbildi uz jautājumu. Pēc katras pareizās atbildes sistēma pievieno 1.

**Izvade**: Pēc izvēlētās atbildes lietotājs noklikšķina uz pogas "Nākamais", pēc kura tiek parādīts nākamā izvēlne ar testa jautājumiem.

1. **Funkcija “Parādīt rezultātu”**

***ID: T6***

**Ievads**: Ļauj uzzināt nokārtotā testa rezultātus.

**Ievade**: Lietotājs noklikšķina uz pogas "Parādīt rezultātu".

**Apstrāde**: Sistēma saskaita pareizo atbilžu skaitu, nepareizo atbilžu skaitu un neatbildēto jautājuma skaitu, tad aprēķina pareizo atbilžu procentuālo vērtību, pēc tam nolasa vērtējumu par 10 balles skalu un atrod komentāru par saņemto atzīmi.

**Izvade**: Tiek atvērts dialoglodziņš, kurā tiek rakstīts šāds teksts:

1. Neatbildēto jautājuma skaits.
2. Pareizo atbilžu skaits.
3. Nepareizo atbilžu skaits.
4. Pareizo atbilžu procentuālā daļa.
5. Atzīme.
6. Sistēmas komentārs par saņemto atzīmi.
7. **Funkcija “Paskatit savos pedejos rezultatos”**

***ID: T7*** **Ievads**: Ļauj lietotājam apskatīt savus pēdējos testu vai uzdevumu rezultātus, kas tika iegūti sistēmā.

**Ievade**: Lietotājs izvēlas izvēlnes opciju “Paskatīt savus pēdējos rezultātus”.

**Apstrāde**: Sistēma izgūst lietotāja pēdējo rezultātu datus no datubāzes vai lokālās atmiņas, kārto tos hronoloģiskā secībā un sagatavo attēlošanai.

**Izvade**: Ekrānā tiek parādīts saraksts ar pēdējiem rezultātiem (piemēram, testa nosaukums, iegūtais punktu skaits).

1. **Funkcija “Dzēst lietotāja kontu”**

***ID: T8*** **Ievads**: Ļauj lietotājam dzēst savu kontu no sistēmas. Visi dati tiek neatgriezeniski izdzēsti.

**Ievade**: Lietotājs ievada savu lietotājvārdu un paroli, pēc tam nospiež pogu "Dzēst kontu".

**Apstrāde**: Sistēma pārbauda ievadītos datus, un ja parole ir pareiza, dzēš lietotāja informāciju no faila vai datubāzes. Ja parole ir nepareiza, tiek atvērts kļūdas dialoglodziņš.

**Izvade**: Dialoglodziņš, kurā uzrakstīts “Lietotāja konts ir veiksmīgi dzēsts!”

1. **Funkcija “Ievietot komentāru pie testa”**

***ID: T9*** **Ievads** Ļauj lietotājam pievienot savu komentāru pēc testa izpildes..

**Ievade**: ietotājs pēc testa izpildes ieraksta komentāru teksta laukā un noklikšķina uz pogas “Pievienot komentāru”.

**Apstrāde**: Sistēma pārbauda, vai komentārs nav tukšs, un saglabā komentāru pie attiecīgā testa datubāzē vai failā.

**Izvade**: Dialoglodziņš, kurā uzrakstīts “Komentārs pievienots veiksmīgi!”

1. **Funkcija “Skolotājs redz visus skolēnu rezultātus”**

***ID: T10*** **Ievads** Ļauj skolotājam apskatīt visu skolēnu rezultātus pēc testu izpildes.

**Ievade**: Skolotājs piesakās sistēmā, atver savu logu un noklikšķina uz pogas “Paskatīt”.

**Apstrāde**: Sistēma nolasa rezultātu datus no faila vai datubāzes, grupē pēc skolēniem un testiem, un attēlo skolotājam.

**Izvade**: Logs, kurā redzams saraksts ar skolēnu vārdiem, lietotājvārdiem, testu rezultātiem.

## Sistēmas nefunkcionālās prasības

1. Prasības produktam:
   1. Lietotāju saskarne ar sistēmu notiek latviešu valodā.
   2. Produktam nav paredzētas vairākas sastāvdaļas. Produktam ir trīs ārējās saskarnes: reģistrēšanai sistēmā, ielogošanai sistēmā, testa pildīšanai.
   3. Programma ir paredzēta vienam lietotājam, kurš saprot latviešu valodu. Lietotāja vecuma vai izglītības ierobežojumi nav paredzēti.
2. Ārējās saskarnes prasības
   1. Lietotāja saskarne: produkts paredz vienotu lietotāja saskarni.
   2. Ekrāna formāti: minimālais ekrāna izmērs ir 10” ar minimālo izšķirtspēju –800x600 punkti.
   3. Lietotāja saskarne “Lietotāja reģistrēšanas saskarne”. Saskarnes struktūra: skat. [1.6. att.](#att1_6)

Reģistreties

Uzvards:

Lietotajvards:

Parole:

Parole (atkartoti):

Vards:

Ielogoties

**reģistrēšanā**

esmu

* skolotājs
* skolēns

1.6. att. Lietotāja reģistrēšanas saskarne

**Saskarnes elementi:**

1. teksta lauks “Vārds”
2. teksta lauks “Uzvārds”
3. teksta lauks “Lietotājvārds”
4. teksta lauks ‘Parole”
5. teksta lauks “Parole (atkārtoti)”
6. Izvelē “Skolotājs vai skolēns”
   1. Lietotāja saskarne “Lietotāja ielogošanas saskarne”. Saskarnes struktūra: skat. [1.7. att.](#att1_7)

Reģistreties

Lietotājvārds:

Parole:

Ielogoties

Dzest Lietotaju kontu

1.7. att. Lietotāja ielogošanas saskarne

**Saskarnes elementi:**

1. teksta lauks “Lietotājvārds”
2. teksta lauks ‘Parole”
3. poga “Ielogoties”
4. poga “Reģistrēties”
5. poga “Dzest Lietotaju kontu”
   1. Lietotāja saskarne “Testa izpilde”. Saskarnes struktūra: skat. [1.8. att.](#att1_8)
6. b)

Test1

Testa Izvele

Test2

Help

Exit

Talak

1. Jautajums “Jautajuma teksts”

1. atbilde

2. atbilde

3. atbilde

Help

c)

Atpakaļ pie testa izvele

Rezultati

Procenti: X%

Atzime: A

Pievienot komentāru

Exit

Help

1.8. att. Testa izpildes saskarne: testa izvēle/sakums; b) atbilde uz jautājumu/-iem ; c) rezultāti

Saskarnes elementi:

1. Teksts”Tests Izvele”
2. Poga “Test 1”
3. Poga “Tests 2”
4. Poga “Help”
5. Poga “Exit”

b)

1. Teksts” Jautajums/Jautajuma teksts”
2. Atbilžu izvēle
3. Poga “Talak”
4. Poga “Help”

c)

1. Teksts”Rezultati”
2. Teksts”Procenti”
3. Teksts”Atzime”
4. Poga“Pievienotkomentāru”
5. Poga “Atpakaļ pie testa izvele”
6. Poga “Help
7. Poga “Exit”
   1. Skolotaja saskarne “Skolotaja logs”. Saskarnes struktūra: skat. [1.10. att.](#att1_8)

a)

Skolenu rezultati Un Komentariji

Atlaļaut Pildit Testu 2

Atlaļaut Pildit Testu 1

Dzest skolenu

Paskatit

Paskatit

Dzest skolenus

Atļaut pidlit testu

Skolenu konti

Lietotajvards

Parole

Iziet no sava konta

Help

Exit

1.10. att. Testēšanas sistēmas ielogošanas forma

a)

1. Teksts” Dzest Skolenus”
2. Teksts” Skolenu konti”
3. Teksts” Skolenu rezultati Un Komentariji”
4. Teksts” Lietotajvards”
5. Teksts” Parole”
6. Teksts” Atļaut pidlit testu”
7. Teksta lauks
8. Teksta lauks
9. Teksta panel
10. Teksta panel
11. Poga “Atlaļaut Pildit Testu 1”
12. Poga “Atlaļaut Pildit Testu 2”
13. Poga “Dzest skolenu”
14. Poga “Paskatit”
15. Poga “Paskatit”
16. Poga “Iziet no sava konta”
17. Poga “Help”
18. Poga “Exit”
19. Administratoru logs: skat. 1.11. att.

Atlaļaut Pildit Testu 2

Atlaļaut Pildit Testu 1

Dzest skolenu

Paskatit

Paskatit

Dzest Lietotajus

Atļauja Pildit Testu

Lietotaju konti

Lietotajvards

Parole

Iziet no sava konta

Help

Exit

Lietotaju rezultati Un Komentariji

1.11. att. Testēšanas sistēmas ielogošanas forma

a)

1. Teksts” Dzest Lietotajus”
2. Teksts” Lietotaju konti”
3. Teksts” Lietotaju rezultati Un Komentariji”
4. Teksts” Lietotajvards”
5. Teksts” Parole”
6. Teksts” Atļaut pidlit testu”
7. Teksta lauks
8. Teksta lauks
9. Teksta panel
10. Teksta panel
11. Poga “Atlaļaut Pildit Testu 1”
12. Poga “Atlaļaut Pildit Testu 2”
13. Poga “Dzest lietotaju”
14. Poga “Paskatit”
15. Poga “Paskatit”
16. Poga “Iziet no sava konta”
17. Poga “Help”
18. Poga “Exit”
19. Studentu logs: skat. 1.12. att.

Mana konta dati

Pedejie rezultati par testiem ar komentariju

Pariet uz Testa uzdevumim

Paskatīt

Paskatīt

Mans profils

1.12. att. Testēšanas sistēmas ielogošanas forma

a)

1. Teksts” Mana konta dati”
2. Teksts” Mans profils”
3. Teksts” Pedejie rezultati par testiem ar komentariju”
4. Poga” Paskatīt”
5. Poga” Paskatīt”
6. Poga” Pariet uz Testa uzdevumim”

3.8 Komentatirju logs: skat. 1.12. att.

Kontu Dzesešana

Parole

Lietorajvards

HelpHРHHHh

Exit

Dzest Kontu

3.1. att. Kontu Dzesešana ielogošanas forma

a)

1. Teksts” Kontu Dzesešana”
2. Teksts” Lietorajvards”
3. Teksts” Parole”
4. Teksta lauks
5. Teksata lauks
6. Poga” Dzest kontu”
7. Poga” Help
8. Poga” Exit”

# Priekšmetiskās jomas klašu sistēmas izstrāde

## Klašu diagramma



## Klašu realizēšana Java valodā

String text1 = jTextField3.getText();

String text2 = jTextField4.getText();

String text3 = jTextField5.getText();

String text4 = jPasswordField1.getText();

String text5 = jPasswordField2.getText();

if (text1.isEmpty() || text2.isEmpty() || text3.isEmpty() || text4.isEmpty() || text5.isEmpty()) {

JOptionPane.showMessageDialog(this, "Pirms testa sākšanas jums ir jāreģistrējas.", "Informācija", JOptionPane.INFORMATION\_MESSAGE);

return;

}

if (!text4.equals(text5)) {

JOptionPane.showMessageDialog(null, "Kļuda: Parole un Parole (atkārtoti) laukam jābūt vienādiem!");

return;

}

String role = "";

if (jRadioButton6.isSelected()) role = "skolens";

else if (jRadioButton5.isSelected()) role = "skolotajs";

else {

JOptionPane.showMessageDialog(this, "Lūdzu, izvēlieties lomu.");

return;

}

boolean userExists = false;

try (BufferedReader reader = new BufferedReader(new FileReader("LoginPassword.txt"))) {

String existingLine;

while ((existingLine = reader.readLine()) != null) {

String[] parts = existingLine.split(",");

if (parts.length > 2 && parts[2].equals(text3)) {

userExists = true;

break;

}

}

}

...

Session.currentUserLogin = text3;

if (!isFirstTestAllowed) {

JOptionPane.showMessageDialog(this, "Jums nav atļauts veikt šo testu.", "Kļūda", JOptionPane.ERROR\_MESSAGE);

return;

}

quiz1.setSize(805,600);

quiz1.setLocationRelativeTo(null);

quiz1.setVisible(true);

Tests.setVisible(false);

int correctAnswers = ...;

double percentage = ((double) correctAnswers / totalQuestions) \* 100;

String grade = ...;

String resultLine = currentUserLogin + "1.tests," + correctAnswers + "/" + totalQuestions + "," +

String.format("%.1f", percentage) + "%," + grade + "\n";

try (FileWriter writer = new FileWriter("rezultati.txt", true)) {

writer.write(resultLine);

}

int correctAnswers = ...;

double percentage = ((double) correctAnswers / totalQuestions) \* 100;

String grade = ...;

String resultLine = currentUserLogin + ",2.tests," + correctAnswers + "/" + totalQuestions + "," +

String.format("%.1f", percentage) + "%," + grade + "\n";

try (FileWriter writer = new FileWriter("rezultati.txt", true)) {

writer.write(resultLine);

}

tring komentars = jTextArea1.getText().replace(",", " ");

String lietotajs = Session.currentUserLogin;

File rezultatiFile = new File("rezultati.txt");

File tempFile = new File("temp\_rezultati.txt");

BufferedReader reader = new BufferedReader(new FileReader(rezultatiFile));

BufferedWriter writer = new BufferedWriter(new FileWriter(tempFile));

List<String> lines = new ArrayList<>();

int lastIndex = -1;

while ((line = reader.readLine()) != null) {

lines.add(line);

String[] parts = line.split(",");

if (parts.length >= 4 && parts[0].equals(lietotajs)) {

lastIndex = index;

}

index++;

}

for (int i = 0; i < lines.size(); i++) {

String currentLine = lines.get(i);

if (i == lastIndex) {

writer.write(currentLine + "," + komentars);

} else {

writer.write(currentLine);

}

writer.newLine();

}

# Testa programmas izstrāde

## Testēšanas metodikas

Pirms uzsākt priekšmetiskās jomas klašu sistēmas testēšanas programmas izstrādi, nepieciešams noteikt atbilstošu testēšanas metodiku.

Ir svarīgi nošķirt programmas koda testēšanu no atkļūdošanas. Atkļūdošanu parasti veic programmētājs, izmantojot izstrādes vides piedāvātos rīkus un balstoties uz savu pieredzi koda izstrādē. Šis process galvenokārt ir vērsts uz sintaktisko un semantisko kļūdu identificēšanu programmas kodā.

Savukārt testēšana ir strukturēts process, kas ietver iepriekšēju plānošanu un konkrētu darbību veikšanu.

Tā būtiskākā sastāvdaļa ir testa piemēru kopas izstrāde, kas kalpo kā pamats testa plānam.

Testa piemēri visbiežāk tiek veidoti, balstoties uz sistēmas funkcionālajām prasībām, un tie aptver dažādus izstrādes posmus – sākot no vienības testēšanas, līdz pat integrācijas un sistēmas testēšanai.

Projekta darba ietvaros plānots veikt vienkāršotu vienības testēšanu, kuras mērķis ir pārbaudīt visu izstrādāto klašu metožu darbību.

Par testēšanas metodiku šajā kontekstā uzskatāms visu klašu metožu izsaukumu saraksts, norādot gaidāmos rezultātus un šo izsaukumu secību testa izpildes laikā.

## Testēšanas programmas struktūra

1. Ielogošanās procesa apraksts:

Testēšanas programmas izstrāde un projektēšana ir cieši saistīta ar iepriekš aprakstīto klašu sistēmas izstrādes procesu – abu posmu darbības ir līdzīgas.

Svarīgi ir uzsvērt specifiskas prasības un aspektus, kas jāņem vērā tieši testēšanas programmas izveidē.

Testēšanas programmai ir jābūt aprīkotai ar grafisku lietotāja saskarni (GUI), kurā iekļautas izvēlnes un interaktīvi rīki, kas ļauj pārbaudīt izstrādāto klašu sistēmas darbību.

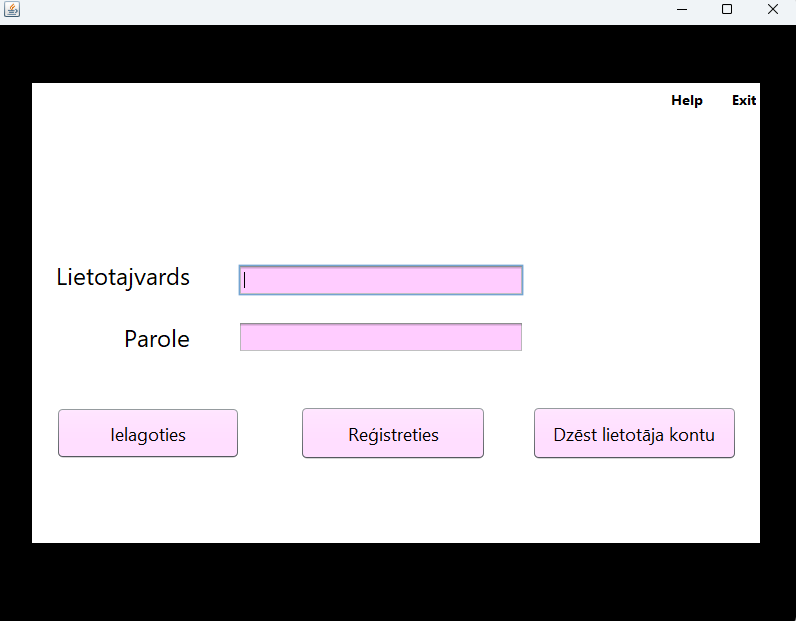
Saskarnes elementi, piemēram, pogas, saraksti, ievades lauki, dialoglodziņi un grafiskie komponenti, tiek definēti, balstoties uz iepriekš aprakstīto testēšanas metodiku.

Lietotāja saskarnei jābūt ērti lietojamai un interaktīvai, galvenā prasība – nodrošināt testēšanas metožu izpildi.

Ielogošanās funkcionalitāte paredzēta, lai piešķirtu piekļuvi sistēmai reģistrētiem lietotājiem. Lai lietotājs varētu uzsākt testēšanu, viņam vispirms jāautorizējas sistēmā.

Ielogošanās laikā lietotājs ievada savu lietotājvārdu un paroli, kas tika reģistrēti iepriekš (skat. 3.1. att.). Ja ievadītie dati sakrīt ar sistēmā glabātajiem verifikācijas datiem, tiek atvērts programmas galvenais logs (skat. 3.3. att.).

Šajā procesā var rasties dažādas kļūmes – piemēram, ja ievades lauki nav aizpildīti vai ievadītie dati ir kļūdaini.



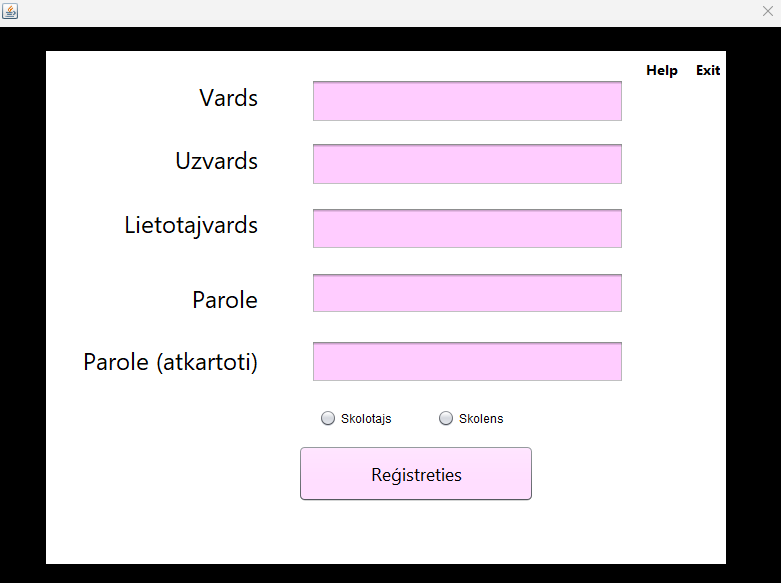
3.1. att. Testēšanas sistēmas ielogošanas forma

1. Reģistrācijas procesa apraksts

Reģistrācijas funkcionalitāte paredzēta jaunu lietotāju pievienošanai sistēmai. Lai lietotājs tiktu reģistrēts, viņam jāaizpilda visi nepieciešamie lauki, kurus pieprasa programma.

Reģistrācijas laikā lietotājs ievada savu vārdu, uzvārdu, lietotājvārdu, paroli, paroles apstiprinājumu, kā arī izvēlas savu lomu – skolotājs vai skolēns (skat. 3.2. attēlu). Ja visi dati ir pareizi aizpildīti, lietotāja informācija tiek saglabāta un tiek atvērts logs testu izveidei (skat. 3.3. attēlu).

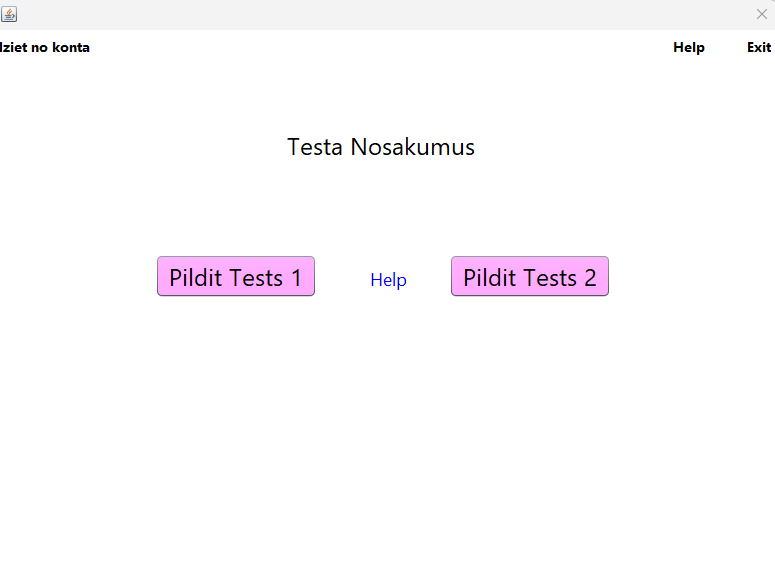
Reģistrācijas laikā var rasties dažādi kļūdu gadījumi, piemēram: tukši obligātie lauki, pārsniegts lietotājvārda vai paroles maksimālais garums, jau esošs lietotājvārds sistēmā, nesakritība starp paroli un tās apstiprinājumu, vai arī nav norādīta lietotāja loma (skolēns vai skolotājs).



3.3. att. Testēšanas sistēmas ielogošanas forma

1. Testa atlases procesa apraksts:

• Testa atlases process ir paredzēts, lai atlasītu un izpildītu testu lietotājiem. Lai sāktu testa izpildi, lietotājam ir jāreģistrējas vai jāpiesakās un jāizvēlas tests (3.3. attēls).



**1.3 att. Testēšanas sistēmas testa formas izvēle**

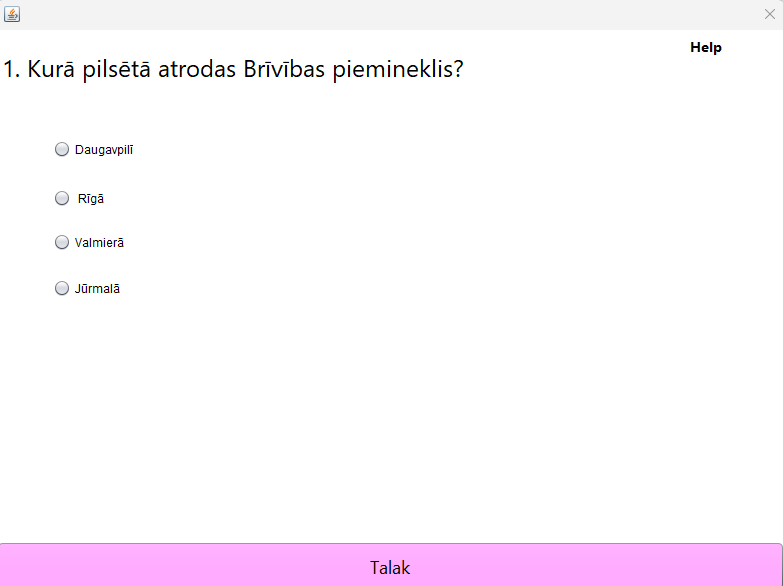
1. Testa izpildes procesa apraksts

Testa izpildes funkcionalitāte ir veidota tā, lai lietotājs varētu atbildēt uz jautājumiem un pēc tam apskatīt iegūto rezultātu.

Lai nokārtotu vienu no diviem piedāvātajiem testiem, lietotājam jāatbild uz 20 jautājumiem, kas saistīti ar konkrētu tēmu.

Testa gaitā lietotājam katrā jautājumā jāizvēlas pareizā atbilde no trim vai četrām piedāvātajām iespējām (skat. 3.4. attēlu).

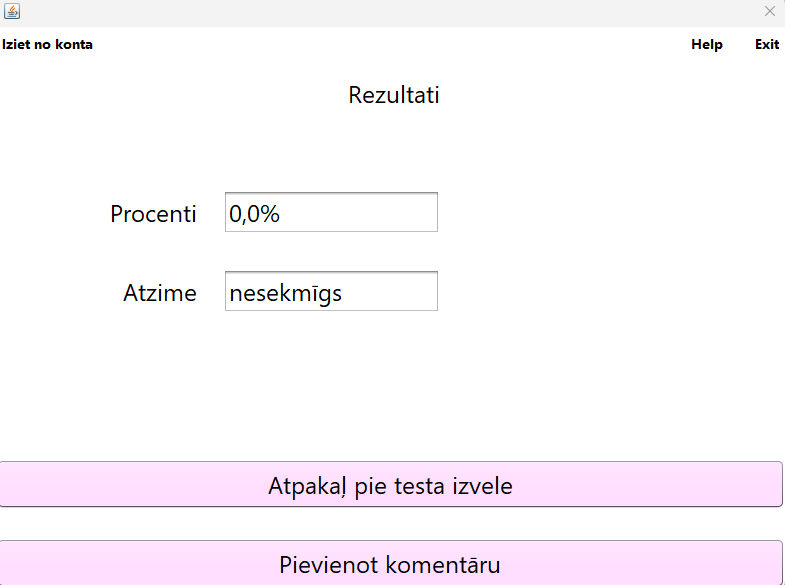
Pēc testa pabeigšanas tiek parādīta rezultātu tabula, kurā norādīts kopējais iegūto punktu skaits un rezultāts procentos (skat. 3.5. attēlu).



3.4att. Testa formas izpilde testa sistēmai

# Rezultāta iegūšanas procesa apraksts:

Rezultātu iegūšanas process ir paredzēts, lai saprastu, cik daudz lietotājs zina par attiecīgo tēmu. Lai iegūtu rezultātu, lietotājam ir jānokārto tests (3.4. attēls), pēc testa nokārtošanas lietotājs var Pabeigt,Apskatīt pareizas atbildes un Paskatit videju atzime klasse nokārtot testu un pabeigt testu (3.5. attēls).



3.4att. Testa sistēmas testa rezultātu forma.

1. Skolotāja loga procesa apraksts:

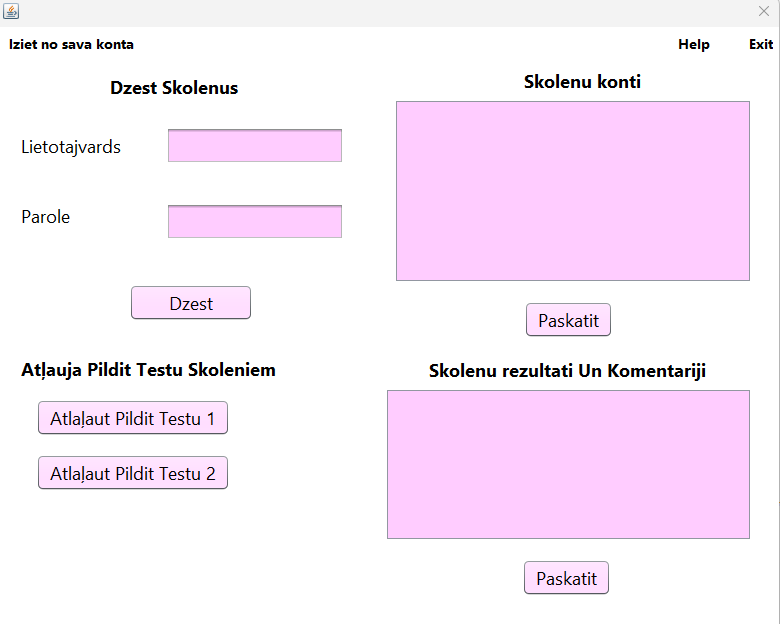
Skolotāja logs (skat. 3.8. attēlu) ir paredzēts tam, lai skolotājs varētu pārvaldīt testēšanas procesu un skolēnu datus.

Skolotājam šajā logā ir iespēja atļaut divu dažādu testu izpildi, izvēloties piemērotu tēmu konkrētam brīdim.

Tāpat skolotājs var apskatīt visu reģistrēto skolēnu datus, kā arī detalizēti pārskatīt visu skolēnu rezultātus un komentārus pēc testu izpildes.

Skolotājam ir arī iespēja nepieciešamības gadījumā dzēst skolēnus no sistēmas.

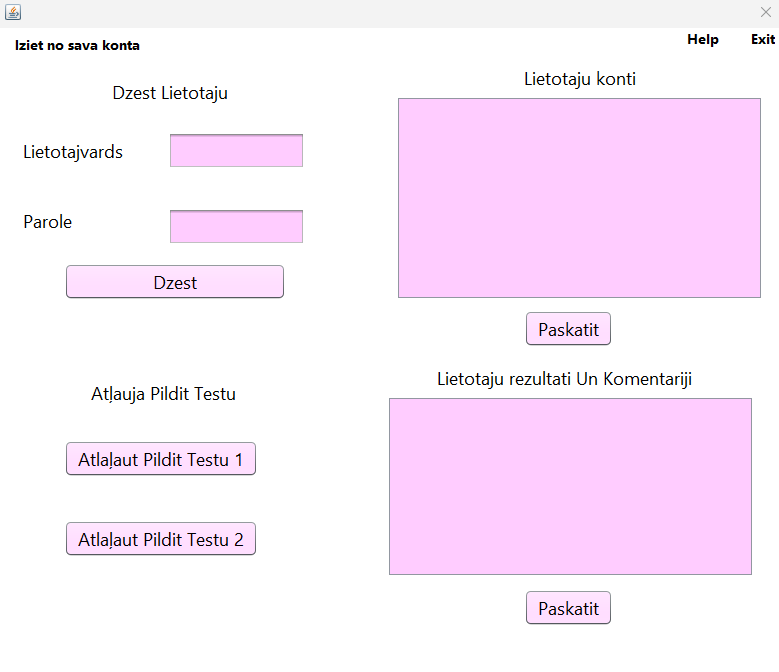
Šis logs nodrošina ērtu piekļuvi visām nepieciešamajām funkcijām skolotāja darbam ar testēšanas platformu, ļaujot efektīvi kontrolēt un analizēt skolēnu sasniegumus.



**3.8. att. Skolotajā logs forma.**

1. Admin loga procesa apraksts:

* Administratora logs (skat. 3.9. attēlu) nodrošina pilnu piekļuvi visām sistēmas pārvaldības funkcijām.
* Tāpat kā skolotājam, administratoram ir iespēja atļaut divu testu izpildi, pārskatīt skolēnu datus, redzēt visus rezultātus un komentārus, kā arī dzēst skolēnus no sistēmas.
* Papildus tam administrators var skatīt visu reģistrēto skolotāju informāciju – vārdu, uzvārdu, lietotājvārdu, kā arī citus ar viņiem saistītos datus.
* Administratoram ir arī tiesības dzēst skolotājus no sistēmas, ja nepieciešams.
* Šis logs paredzēts kā galvenais pārvaldības centrs, kas ļauj administratoram pilnībā kontrolēt gan lietotāju datus, gan testēšanas procesa organizēšanu visos līmeņos.



* 1. att. Administratorā logs forma.

1. Skolēna loga procesa apraksts

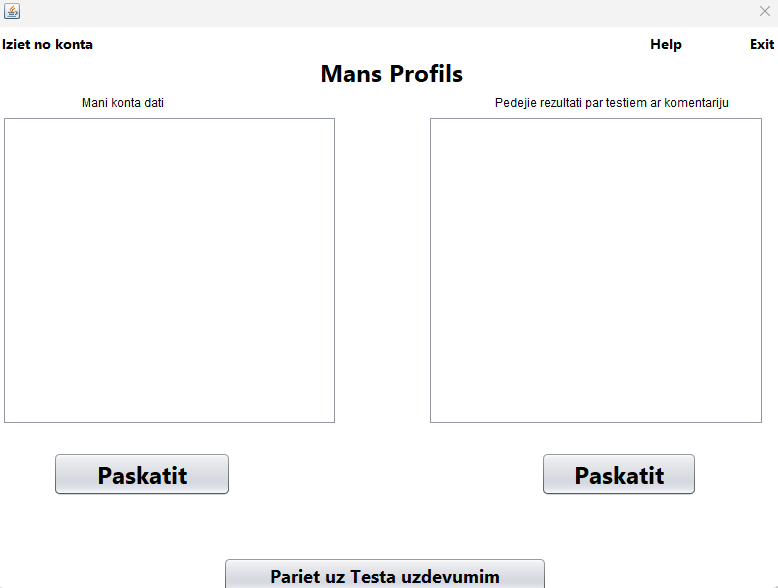
Skolēna logs (skat. 3.10. attēlu) ir paredzēts lietotājam ar skolēna lomu, lai ērti piekļūtu svarīgajai informācijai un testa izpildes iespējām.

Šajā logā skolēns var apskatīt savus reģistrācijas datus, piemēram: vārdu, uzvārdu, lietotājvārdu un lomu.

Lietotājam ir arī iespēja apskatīt savus iepriekšējos testa rezultātus, kuros norādīts iegūtais punktu skaits, rezultāts procentos, kā arī skolotāja komentāri (ja tādi pievienoti).

No šī loga skolēns var pāriet uz testa sākuma ekrānu, lai uzsāktu kādu no pieejamajiem testiem atbilstoši atļautajām tēmām.

Saskarne ir veidota tā, lai tā būtu vienkārša un saprotama, nodrošinot skolēnam patīkamu un ērtu lietošanas pieredzi.



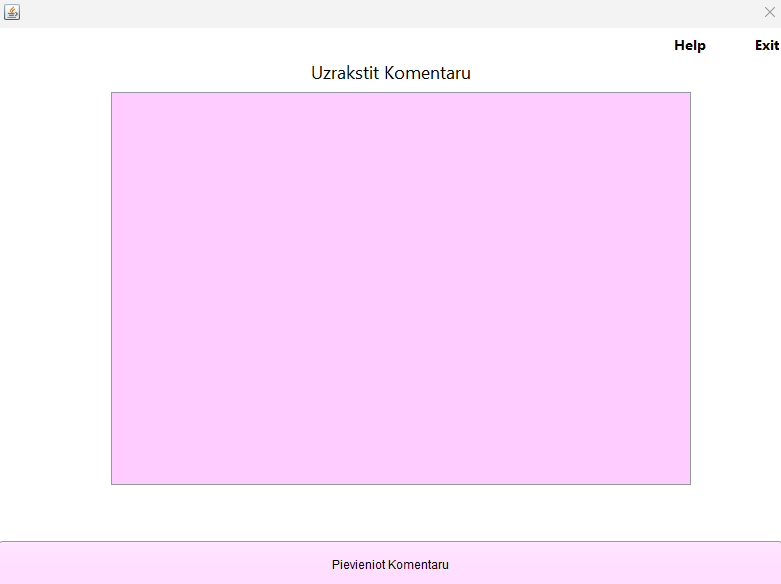
att. Studentu logs forma.

1. Komentāru loga apraksts

Komentāru logs paredzēts, lai lietotājs varētu iesniegt savu viedokli vai atsauksmi par testu vai sistēmas darbību.

Logs satur teksta ievades lauku, kurā lietotājs var ierakstīt savu komentāru, un pogu "Piedāvāt komentāru", lai nosūtītu ziņojumu.

Saskarne ir vienkārša un intuitīva – pietiek ar vienu klikšķi, lai komentārs tiktu iesniegts.



# att. Komentaru logs forma

1. Konta dzēšanas loga apraksts

Konta dzēšanas logs paredzēts lietotājiem, kuri vēlas dzēst savu profilu no sistēmas.

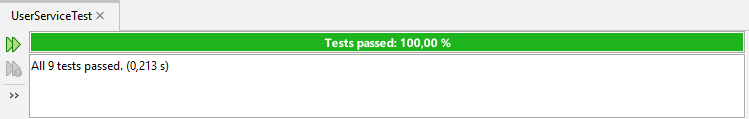
Logā atrodas divi ievades lauki – viens lietotājvārdam (loginam) un otrs parolei.

Pēc šo datu ievadīšanas lietotājs var nospiest pogu “Dzēst kontu”, lai apstiprinātu dzēšanu.

Ja ievadītie dati ir pareizi, lietotāja profils tiek neatgriezeniski izdzēsts no sistēmas. Ja dati neatbilst – parādās kļūdas paziņojums.



## Testēšanas rezultāti



Testēšanas kods testē lietotāju apkalpošanas funkcijas Java valodā. Tiek pārbaudīti:

• vai kāds no ievades laukiem ir tukšs.

• vai ievades garums ir derīgs.

• vai paroles sakrīt.

• vai lietotājs eksistē datu failā.

• vai lietotājvārds un parole ir pareizi.

• vai jauns lietotājs var tikt reģistrēts.

• vai esošs lietotājs nevar tikt reģistrēts atkārtoti.

• vai esošs lietotājs tiek veiksmīgi dzēsts.

• vai neeksistējošs lietotājs netiek dzēsts kļūdaini.

Visas testēšanas metodes ir izpildītas veiksmīgi — 100% testi ir izturēti (9 no 9).

# Lietotāja rokasgrāmata

## Lietotāja instrukcija

1. Pieteikšanās vai reģistrācija:
2. Poga «Reģistrēties» lai izveidotu jaunu lietotāja kontu.
3. Poga «Skolotājs» lai izvēlētos skolotāja lomu.
4. Poga «Skolēns» lai izvēlētos skolēna lomu.
5. Poga «Help» lai iegūtu palīdzību reģistrācijas logā.
6. Poga «Exit» lai aizvērtu programmu.
7. Testu izvēles logs
8. Poga «Pildīt Tests 1» lai sākt pirmo testu.
9. Poga «Pildīt Tests 2» lai sākt otro testu.
10. Poga «Iziet no konta» lai iziet no sava lietotāja profila.
11. Poga «Help» lai iegūtu palīdzību testu izvēles logā.
12. Poga «Exit» lai aizvērtu programmu.
13. Testa logs
14. 4.17. Poga «Atpakaļ pie testa izvēles» lai atgrieztos uz testu izvēli.
15. 4.18. Poga «Pievienot komentāru» lai pievienotu komentāru par testu.
16. 4.19. Poga «Iziet no konta» lai izrakstītos no sistēmas.
17. 4.20. Poga «Help» lai saņemtu palīdzību par rezultātu logu.
18. 4.21. Poga «Exit» lai aizvērtu lietotni.Rezultātu logs:
19. Skolotāja logs
20. Poga «Skolēnu konti» lai apskatītu visus skolēnus.
21. Poga «Rezultāti un komentāri» lai apskatītu skolēnu testu rezultātus un komentārus.
22. Poga «Atļaut Tests 1» lai atļautu pildīt 1. testu.
23. Poga «Atļaut Tests 2» lai atļautu pildīt 2. testu.
24. Poga «Help» lai saņemtu palīdzību skolotāja logā.
25. Poga «Exit» lai aizvērtu skolotāja logu.
26. Administratora logs
    1. Poga «Skolēnu konti» lai apskatītu visus reģistrētos skolēnus.
    2. Poga «Rezultāti un komentāri» lai skatītu testu rezultātus un komentārus.
    3. Poga «Dzēst lietotāju» lai dzēstu skolēnu kontu.
    4. Poga «Help» lai atvērtu palīdzības logu.
    5. Poga «Exit» lai aizvērtu administratora logu.
27. Skolēna logs (StudentLogs)
28. Poga «Rādīt rezultātus» lai apskatītu savus testa rezultātus.
29. Poga «Rādīt skolēnu sarakstu» lai redzētu visus skolēnus sistēmā.
30. Poga «Iziet no sava konta» lai izietu no skolēna profila.
31. Poga «Exit» lai aizvērtu lietotni.
32. Komentāru logs (KomentLogs)
33. Poga «Pievienot Komentāru» lai nosūtītu komentāru par testu skolotājam vai administratoram.  
    4.38. Poga «Iziet» lai atgrieztos atpakaļ uz galveno izvēlni.
34. Lietotāja dzēšanas logs (DeleteLietotajulogs)
35. Tekstlauki «Lietotājvārds» un «Parole» lai ievadītu konta informāciju.
36. Poga «Dzēst Kontu» lai neatgriezeniski dzēstu kontu.
37. Poga «Iziet no konta» lai atgrieztos bez dzēšanas.

## Palīdzības sistēma (Help)

Lai sāktu pildīt testu, ir jāpiesakās savā kontā, izmantojot lietotājvārdu un paroli. Ja jums nav konta programmā, jums jāreģistrējas, norādot vārdu, uzvārdu, lietotājvārdu un paroli, kā arī jāizvēlas loma – skolotājs vai skolēns. Pēc pieslēgšanās lietotājam ir iespēja izvēlēties testu vai treniņu. Pieejami divi testi – “Pildit Tests 1” un “Pildit Tests 2”. Ja skolotājs ir atļāvis, skolēns var pildīt treniņu vai testu.

Katram jautājumam testa laikā ir četri atbilžu varianti un poga “Tālāk”, kas ļauj pāriet uz nākamo jautājumu. Pēc testa beigām tiek parādīts rezultāts procentos, atzīme, kā arī iespējams pievienot komentāru. Visi rezultāti tiek saglabāti datu bāzē.

Lietotājiem pieejamas funkcijas:

“Help” – palīdzības logs ar instrukcijām

“Exit” – aizver programmu

“Iziet no konta” – izrakstīties no sava profila

“Dzēst kontu” – dzēst savu lietotāja kontu

“Reģistrēties” – jauna konta izveide

“Ielogoties” – esoša konta pieslēgšanās

“Paskatīt” – apskatīt rezultātus vai komentārus (pieejams skolotājiem un adminiem)

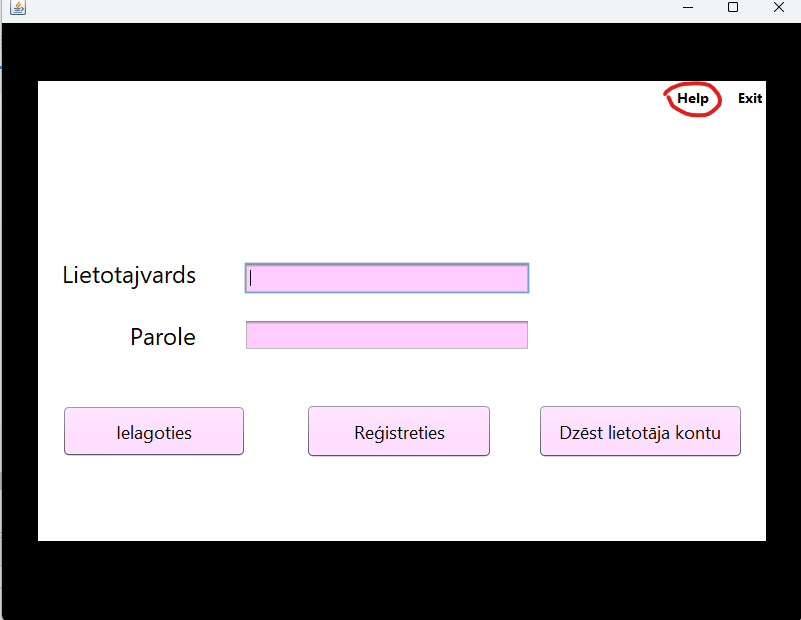
Skolotāji un administratori var:

Piešķirt piekļuvi testiem skolēniem

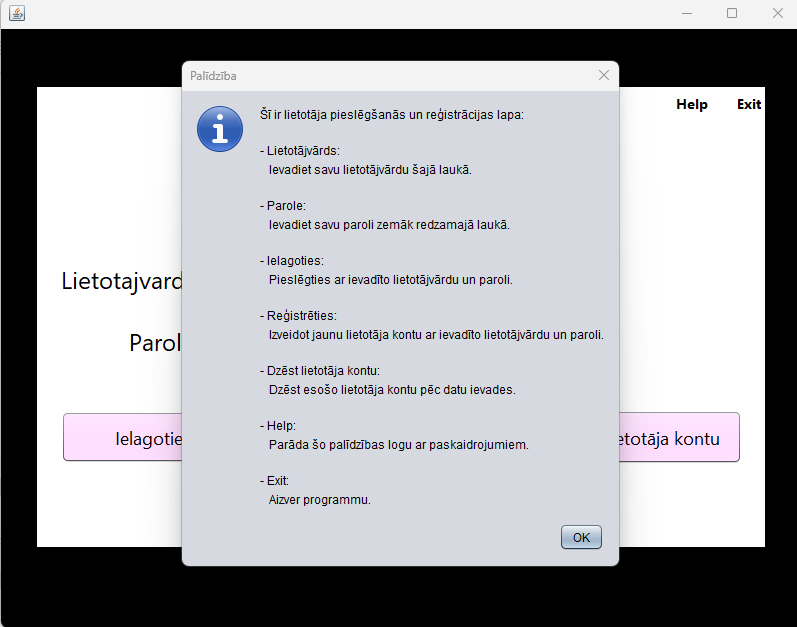
Apskatīt lietotāju rezultātus un komentārus

Dzēst lietotājus no sistēmas

Ja rodas problēmas ar reģistrāciju, lietotājs var noklikšķināt uz pogas “Help” (att. 4.1). Atvērsies dialoglogs, kurā būs aprakstīti reģistrācijas noteikumi (4.2. att.).



4.1. att. Pogas “Palīdzība” piemērs



4.2. att. Ielagošana Palidizba

# Secinājumi

Projekta “Testēšanas sistēma” izstrāde tika veikta, izmantojot Java programmēšanas valodu un Swing grafisko lietotāja saskarni. Tā ietvaros izveidota lietotāju reģistrācijas un autorizācijas sistēma, testu izpildes iespēja un rezultātu saglabāšana. Izstrādes laikā nācās saskarties ar dažādiem izaicinājumiem, īpaši failu apstrādes un datu validācijas jomā. Darba process bija interesants un ļāva iegūt jaunas zināšanas par grafisko interfeisu veidošanu, darbu ar failiem un lietotāju datu pārvaldību. Lai arī dažas idejas netika pilnībā īstenotas, galvenā funkcionalitāte ir realizēta un sistēma darbojas veiksmīgi. Šis projekts sniedza vērtīgu pieredzi, kas būs noderīga arī nākotnē. Ceru, ka ar katru nākamo projektu prasmes tikai augs!

# Izmantotās literatūras (informācijas avotu) saraksts

1. **Zaiceva, L.** *Programmatūras izstrādes tehnoloģija.* Rīga : RTU, 2002.

2. **Kodors, Sergejs.** *IEVADS PRASĪBU INŽENIERIJĀ. Mācību līdzeklis.* Rēzekne : Rēzeknes Tehnoloģiju akadēmija, 2019. gada. 978-9984-44-226-6.

3. **Dortiņa, I. un Dortiņš, A.** 1.1.1. Ievads Java. Java pamati. [Tiešsaiste] 2024. gada 13. 2. [Citēts: 2025. gada 20. 3.] https://skolo.lv/mod/resource/view.php?id=82126301.

4. **—.** 1.1.2. Java sazarotas struktūras operātori. [Tiešsaiste] 2024. gada 14. 2. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=82257905.

5. —. 1.1.3. Java cikliskās struktūras operatori. [Tiešsaiste] 2022. gada 22. 2. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=82383944.

6. —. 1.2.1.1. Java masīvi. [Tiešsaiste] 2024. gada 29. 2. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=82503734.

7. —. 1.2.1.2. Java rakstzīmju masīvi un rakstzīmju virknes. [Tiešsaiste] 2024. gada 7. 3. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=82608403.

8. —. 1.2.1.3. Java dinamiskie masīvi. [Tiešsaiste] 2024. gada 13. 3. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=82703677.

9. —. 1.2.2. Java ieraksti. [Tiešsaiste] 2024. gada 19. 3. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=82785148.

10. —. 1.2.3. Java Ievades/izvades sistēma. [Tiešsaiste] 2024. gada 3. 4. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=82944472&redirect=1.

11. —. 2.1.1.1. Java. OOP koncepcija. [Tiešsaiste] 2024. gada 16. 4. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=83119822.

12. **Dortiņa, I. un Stašanova, L.** 2.1.1.2. Java. Metodes un konstruktori. [Tiešsaiste] 2024. gada 2. 9. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=84426472.

13. **—.** 2.1.2. Java. Mantošana un metožu pārdefinēšana. [Tiešsaiste] 2024. gada 30. 9. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=85417326.

14. —. 2.1.3. Java pakotnes un interfeisi. [Tiešsaiste] 2024. gada 14. 10. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=86185154.

15. —. 2.2. Java izņēmumi un to apstrāde. [Tiešsaiste] 2022. gada 23. 10. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=86272136&redirect=1.

16. —. 4.1.1. Java. Grafiskais lietotāja interfeiss. [Tiešsaiste] 2025. gada 23. 1. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=87282004.

17. —. 4.1.2. Java GUI izveidošana NetBeans vidē. [Tiešsaiste] 2025. gada 5. 2. [Citēts: 2025. gada 20. 2.] https://skolo.lv/mod/resource/view.php?id=87334441.

18. Latvijas Nacionālais terminoloģijas portāls. *termini.gov.lv.* [Tiešsaiste] Izstrādātājs: Tilde., 2005-2023. gada. [Citēts: 2025. gada 20. 2.] https://termini.gov.lv/.

19. **Taylor Johnson, Dung X. Nguyen.** UML and more JAVA Syntax. [Tiešsaiste] COMP 212 LAB 1.5, 2007. gada 16. 1. [Citēts: 2025. gada 20. 2.] https://www.clear.rice.edu/comp212/07-spring/labs/01.5/.

# Pielikums A. Klašu sistēma

public class Admin extends User {

public Admin(String name, String surname, String username, String password) {

super(name, surname, username, password, "admin");

}

@Override

public boolean canDeleteUser(User target) {

return true;

}

@Override

public boolean canViewUser(User target) {

return true;

}

@Override

public boolean canViewResults(User target) {

return true;

}

@Override

public boolean canViewComments(User target) {

return true;

}

@Override

public boolean canGrantTestAccess(User target) {

return target.isStudent();

}

@Override

public boolean canRevokeTestAccess(User target) {

return target.isStudent();

}

}

public class DistanceExamenator extends User {

public DistanceExamenator(String name, String surname, String username, String password) {

super(name, surname, username, password, "eksamenators");

}

@Override

public boolean canDeleteUser(User target) {

return true;

}

@Override

public boolean canViewUser(User target) {

return true;

}

@Override

public boolean canViewResults(User target) {

return true;

}

@Override

public boolean canViewComments(User target) {

return true;

}

@Override

public boolean canGrantTestAccess(User target) {

return target.isStudent();

}

@Override

public boolean canRevokeTestAccess(User target) {

return target.isStudent();

}

@Override

public boolean canViewAdminPassword() {

return true;

}

public void viewAllUsers(List<User> allUsers) {

for (User user : allUsers) {

System.out.println(user.getUsername() + " (" + user.getRole() + ")");

}

}

public void deleteUser(String usernameToDelete, List<User> users) {

users.removeIf(user -> user.getUsername().equals(usernameToDelete));

}

}

public class Question {

private String text;

private List<String> answers;

private int correctIndex;

public Question(String text, List<String> answers, int correctIndex) {

this.text = text;

this.answers = answers;

this.correctIndex = correctIndex;

}

public String getText() { return text; }

public List<String> getAnswers() { return answers; }

public int getCorrectIndex() { return correctIndex; }

public boolean isCorrect(int selectedIndex) {

return selectedIndex == correctIndex;

}

}

public class Teacher extends User {

public Teacher(String name, String surname, String username, String password) {

super(name, surname, username, password, "skolotajs");

}

@Override

public boolean canDeleteUser(User target) {

return target.isStudent();

}

@Override

public boolean canViewUser(User target) {

return target.isStudent();

}

@Override

public boolean canViewResults(User target) {

return target.isStudent();

}

@Override

public boolean canViewComments(User target) {

return target.isStudent();

}

@Override

public boolean canGrantTestAccess(User target) {

return target.isStudent();

}

@Override

public boolean canRevokeTestAccess(User target) {

return target.isStudent();

}

}

public class User {

protected String name;

protected String surname;

protected String username;

protected String password;

protected String role;

public User(String name, String surname, String username, String password, String role) {

this.name = name;

this.surname = surname;

this.username = username;

this.password = password;

this.role = role;

}

public String getName() {

return name;

}

public String getSurname() {

return surname;

}

public String getUsername() {

return username;

}

public String getPassword() {

return password;

}

public String getRole() {

return role;

}

// --- Роль-помощники ---

public boolean isStudent() {

return "skolens".equalsIgnoreCase(role);

}

public boolean isTeacher() {

return "skolotajs".equalsIgnoreCase(role);

}

public boolean isAdmin() {

return "admin".equalsIgnoreCase(role);

}

public boolean isExaminer() {

return "eksamenators".equalsIgnoreCase(role);

}

public boolean canDeleteUser(User target) {

return false;

}

public boolean canViewUser(User target) {

return false;

}

public boolean canViewResults(User target) {

return false;

}

public boolean canViewComments(User target) {

return false;

}

public boolean canGrantTestAccess(User target) {

return false;

}

public boolean canRevokeTestAccess(User target) {

return false;

}

public boolean canViewAdminPassword() {

return false;

}

public String toFileFormat() {

return name + "," + surname + "," + username + "," + password + "," + role;

}

}

public class UserRepository {

private static final String FILE\_NAME = "LoginPassword.txt";

public static List<User> getAllUsers() throws IOException {

List<User> users = new ArrayList<>();

try (BufferedReader reader = new BufferedReader(new FileReader(FILE\_NAME))) {

String line;

while ((line = reader.readLine()) != null) {

String[] parts = line.split(",");

if (parts.length >= 5) {

String name = parts[0];

String surname = parts[1];

String username = parts[2];

String password = parts[3];

String role = parts[4];

switch (role) {

case "skolens" -> users.add(new Student(name, surname, username, password));

case "skolotajs" -> users.add(new Teacher(name, surname, username, password));

case "admin" -> users.add(new Admin(name, surname, username, password));

default -> users.add(new User(name, surname, username, password, role));

}

}

}

}

return users;

}

public static void saveUser(User user) throws IOException {

try (FileWriter writer = new FileWriter(FILE\_NAME, true)) {

writer.write(user.toFileFormat() + System.lineSeparator());

}

}

public static void deleteUser(String username, String password) throws IOException {

File inputFile = new File(FILE\_NAME);

File tempFile = new File("LoginPassword\_temp.txt");

try (

BufferedReader reader = new BufferedReader(new FileReader(inputFile));

BufferedWriter writer = new BufferedWriter(new FileWriter(tempFile))

) {

String line;

while ((line = reader.readLine()) != null) {

String[] parts = line.split(",");

if (parts.length >= 4 && parts[2].equals(username) && parts[3].equals(password)) {

continue;

}

writer.write(line + System.lineSeparator());

}

}

inputFile.delete();

tempFile.renameTo(inputFile);

}

}

# Pielikums B. testa programmas kods

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.File;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import static org.junit.Assert.\*;

import java.io.\*;

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

public class UserServiceTest {

private File tempFile;

@Before

public void setUp() throws IOException {

tempFile = File.createTempFile("LoginPassword", ".txt");

try (BufferedWriter writer = new BufferedWriter(new FileWriter(tempFile))) {

writer.write("Gunars,Gravitis,Alien,123,skolens\n");

writer.write("Anna,Berzina,Teacher1,pass456,skolotajs\n");

}

}

@After

public void tearDown() {

if (tempFile != null && tempFile.exists()) {

tempFile.delete();

}

}

@Test

public void testIsFieldEmpty() {

assertTrue(UserService.isFieldEmpty("text", "", "another"));

assertFalse(UserService.isFieldEmpty("name", "surname", "pass"));

}

@Test

public void testIsValidLength() {

assertTrue(UserService.isValidLength("Lietotajs", 3, 20));

assertFalse(UserService.isValidLength("ab", 3, 20));

}

@Test

public void testPasswordsMatch() {

assertTrue(UserService.passwordsMatch("parole123", "parole123"));

assertFalse(UserService.passwordsMatch("123", "321"));

}

@Test

public void testUserExists() throws IOException {

assertTrue(UserService.userExists("Alien", tempFile));

assertTrue(UserService.userExists("Teacher1", tempFile));

assertFalse(UserService.userExists("Unknown", tempFile));

}

@Test

public void testLoginCorrect() throws IOException {

assertTrue(UserService.loginCorrect("Alien", "123", tempFile));

assertFalse(UserService.loginCorrect("Alien", "wrongpass", tempFile));

}

@Test

public void testRegisterNewUser() throws IOException {

assertTrue(UserService.registerUser("Test", "User", "NewUser", "pass789", "skolens", tempFile));

assertTrue(UserService.userExists("NewUser", tempFile));

}

@Test

public void testRegisterDuplicateUser() throws IOException {

// Alien уже есть в начальных данных

assertFalse(UserService.registerUser("Someone", "Else", "Alien", "anotherPass", "skolens", tempFile));

}

@Test

public void testDeleteExistingStudent() throws IOException {

assertTrue(UserService.deleteUser("Alien", "skolens", tempFile));

assertFalse(UserService.userExists("Alien", tempFile));

}

@Test

public void testDeleteNonExistingUser() throws IOException {

assertFalse(UserService.deleteUser("GhostUser", "skolens", tempFile));

}

}

class UserService {

public static boolean isFieldEmpty(String... fields) {

for (String field : fields) {

if (field == null || field.trim().isEmpty()) return true;

}

return false;

}

public static boolean isValidLength(String input, int min, int max) {

return input != null && input.length() >= min && input.length() <= max;

}

public static boolean passwordsMatch(String pass1, String pass2) {

return pass1 != null && pass1.equals(pass2);

}

public static boolean userExists(String username, File file) throws IOException {

try (BufferedReader reader = new BufferedReader(new FileReader(file))) {

String line;

while ((line = reader.readLine()) != null) {

String[] parts = line.split(",");

if (parts.length >= 3 && parts[2].trim().equals(username.trim())) {

return true;

}

}

}

return false;

}

public static boolean loginCorrect(String username, String password, File file) throws IOException {

try (BufferedReader reader = new BufferedReader(new FileReader(file))) {

String line;

while ((line = reader.readLine()) != null) {

String[] parts = line.split(",");

if (parts.length >= 4 &&

parts[2].trim().equals(username) &&

parts[3].trim().equals(password)) {

return true;

}

}

}

return false;

}

public static boolean registerUser(String name, String surname, String username, String password, String role, File file) throws IOException {

if (userExists(username, file)) return false;

try (BufferedWriter writer = new BufferedWriter(new FileWriter(file, true))) {

writer.write(name + "," + surname + "," + username + "," + password + "," + role + System.lineSeparator());

}

return true;

}

public static boolean deleteUser(String username, String role, File file) throws IOException {

File tempFile = new File(file.getAbsolutePath() + ".tmp");

boolean deleted = false;

try (BufferedReader reader = new BufferedReader(new FileReader(file));

BufferedWriter writer = new BufferedWriter(new FileWriter(tempFile))) {

String line;

while ((line = reader.readLine()) != null) {

String[] parts = line.split(",");

if (parts.length >= 5) {

String login = parts[2].trim();

String userRole = parts[4].trim();

if (login.equals(username) && userRole.equals(role)) {

deleted = true;

continue;

}

}

writer.write(line + System.lineSeparator());

}

}

if (deleted) {

if (!file.delete() || !tempFile.renameTo(file)) {

return false;

}

} else {

tempFile.delete();

}

return deleted;

}

}