Latvijas Republikas izglītības un zinātnes ministrija

Profesionālās izglītības kompetences centrs

Daugavpils Tehnoloģiju un tūrisma tehnikums

**PRAKTISKAIS DARBS**

Programmēšanas tehniķis

specialitāte

Praktiskais darbs Nr.1

temats

Testēšanas sistēmas izveidošana

PASKAIDROJOŠAIS RAKSTS

**DTTT.P.D.0006.006PA**

Profesijas kods 33 481 031 Programmēšanas nodaļa

PR-21. grupa

Izveidoja V.Dimbovskis

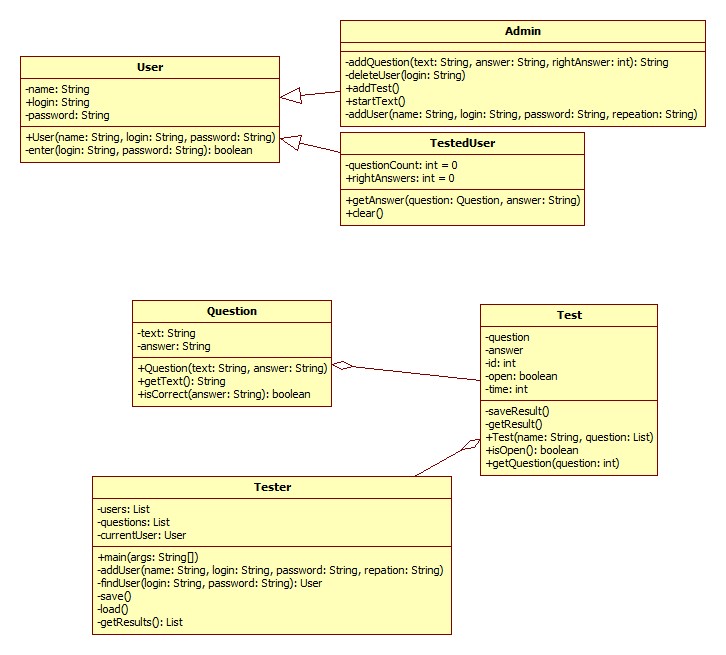
paraksts V.,Uzvārds

Pārbaudīja

paraksts V.,Uzvārds

2023./2024. m.g.

**Saturs**



[Ievads 3](#_Toc164839579)

[1. Uzdevuma nostādne 4](#_Toc164839580)

[1.1. Sistēmas modelis 4](#_Toc164839581)

[*1.1.1. Priekšmetiskās jomas informācijas modelis 4*](#_Toc164839582)

[*1.1.2. Sistēmas apkārtnes shēma 4*](#_Toc164839583)

[*1.1.3. Viedokļu analyze 5*](#_Toc164839584)

[*1.1.4. Datu modelēšana 6*](#_Toc164839585)

[*1.1.5. Datu vārdnīca 6*](#_Toc164839586)

[1.2. Sistēmas evolūcija 7](#_Toc164839587)

[1.3. Sistēmas funkcionālās prasības 7](#_Toc164839588)

[1.4. Sistēmas nefunkcionālās prasības 9](#_Toc164839589)

[2. Priekšmetiskās jomas klašu sistēmas izstrāde 12](#_Toc164839590)

[2.1. Klašu diagramma 12](#_Toc164839591)

# 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.

Šī ir programma, kas īpaši izstrādāta skolotājiem un skolēniem, lai vienkāršotu darbu vai uzdevumu vērtēšanas un pārbaudes procesu. Ar šīs programmas palīdzību skolotāji varēs ātrāk veikt vērtēšanu, bet skolēni - iegūt rezultātus. Programma reģistrē skolēna rezultātus, analizē tos ar pareizajām atbildēm un pēc tam sniedz skolēnam paveiktā darba rezultātu. Skolēns var detalizēti analizēt savu darbu un redzēt savas kļūdas.

# 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.).

Datu bāze

Rezultāti

Tests

Administrātors/ Skolotājs

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.).

Lietotājs

Lietotāja saskarne

Sistēma

Datu bāze

Lietotāja dati un rezultāti

1.2. att. Sistēmas apkārtnes modeļa piemērs

### Viedokļu analyze

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

Datu viedokļi

Funkcionālie viedokļi

Nefunkcionālie viedokļi

Viedokļi par lietotāju

1.3. att. Viedokļu burbuļu diagramma

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

Testēšanas sistēma(V0)

Lietotājs(V11)

Skolotājs(V11)

Audzēknis(V112)

Datu bāze(V12)

Lietotāju reģistrs(V121)

Tests(V13)

Testa jautājumu reģistrs(V122)

Skolotāja funkcijas(V131)

Testa izveide(V1311)

Testa manuāla novērtēšana(V1312)

Audzekņa funkcijas(V132)

Testa pildīšana(V1321)

Testa rezultātu saņemšana(1322)

1.4. att. Viedokļu hierarhiskā struktūra

### Datu modelēšana

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

Audzēknis

Skolotājs

Tests

Jautājums

Pilda

Satur

Izveido

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”**

**ID: 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 “Parādīt un saglabāt rezultātus”**

**ID: T7**

**Ievads:** Ļauj lietotājam skatīt savus rezultātus un saglabāt tos

**Ievade:** Lietotājs noklikšķina uz pogas "Apskatīt rezultātusun saglabāt tos"

Apstrāde: Sistēma nolasa pareizo atbilžu skaitu, salīdzina tās ar jautājumu skaitu un izvada atbildi procentos, pēc tam to saglabā.

**Izvade:**  Tiek atvērts dialoglodziņš, kurā tiek rakstīts procentos un punktu skaits.

## 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.

Vārds:

Uzvārds:

Lietotāvārds:

Parole:

Parole(Atkartoti):

Reģistrēties

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. poga “Reģistrēties”

* 1. Lietotāja saskarne “Lietotāja ielogošanas saskarne”. Saskarnes struktūra: skat. 1.7. att.

Lietotāvārds:

Parole:

Ielogoties

Reģistrēties

1.7. att. Lietotāja ielogošanas saskarne

Saskarnes elementi:

1. teksta lauks “Lietotājvārds”

2. teksta lauks ‘Parole”

3. poga “Ielogoties”

4. poga “Reģistrēties”

* 1. Lietotāja saskarne “Testa izpilde”. Saskarnes struktūra: skat. 1.8. att.

Tests “Nosaukums”

Rezultāti

Procenti: **x**%

Punkti: **x**p.

1.jautājums “Jautājuma teksts”

1. atbilde

2. atbilde

3. atbilde

a.)

b.)

c.)

Beigt

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

Saskarnes elementi:

1. 1. teksta lauks “Nosaukums”

2. poga “Sākt testu”

1. 1. teksta lauks “Jautājuma teksts”

2. teksta lauks “1.atbilde”

3. teksta lauks “2.atbilde”

4. teksta lauks “3.atbilde”

5. poga “1.atbilde”

6. poga “2.atbilde”

7. poga “3.atbilde”

c) 1. teksta lauks “Rezultāti”

2. teksta lauks “Procenti: %”

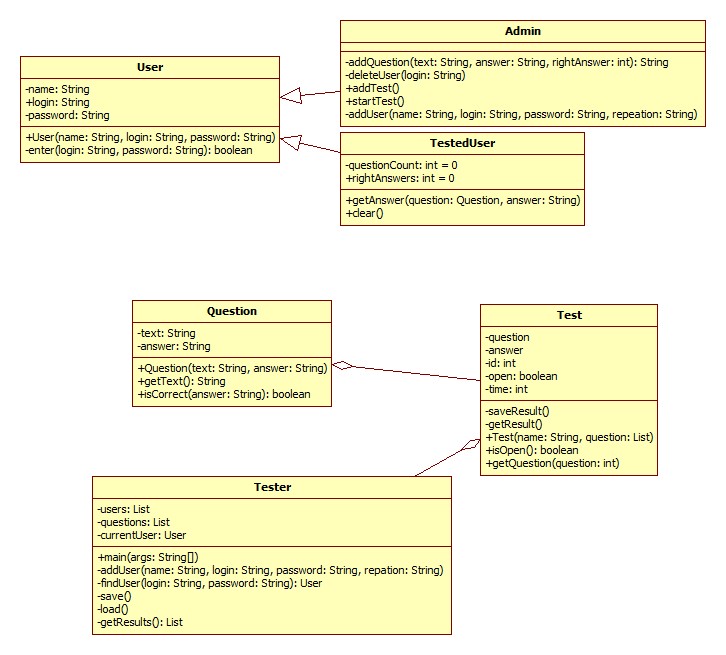
3. teksta lauks “Punkti: p.”

4. poga “Beigt”

# Priekšmetiskās jomas klašu sistēmas izstrāde

## Klašu diagramma

Klašu diagrammā ir realizēts vispārīgs klašu apraksts – klašu hierarhijas vispārējā struktūru, to sadarbība, atribūti, metodes, saskarnes un to savstarpējās attiecības, izmantojot UML valodu.



**2.1. att. Testēšanas sistēmas klašu diagramma UML valodā**

Balstoties uz izstrādāto sistēmas modeli var veikt sekojošu klašu iedalījumu (2.1. att.):

1. Klase User – ir atbildīga par darbu ar lietotājiem.

• User (name: String, login: String, password: String) – konstruktors, kurš izveido jaunu lietotāju ar norādītiem datiem;

• enter (login: String, password: String) – metode, kura nodrošina lietotāju pieslēgšanu sistēmai:

2. Klase Admin – nodrošina darbu ar lietotajiem un testiem.

* addQuestion(text: String, answer: String, rightAnswer: int): String -Pievieno testam jautājumus
* deleteUser(login: String) -Dzēš atlasīto lietotāju
* addTest() -Pievieno testu lietotājiem
* startTest() -Sāk lietotāja testu
* addUser(name: String, login: String, password: String, repeation: String) – Pievieno lietotāja piekļuvi testam.

3. Klase TestedUser – nodrošina audzēkņa testa pildīšanas darbus.

* getAnswer(question: Question, answer: String) –sniedz iespēju atbildēt uz uzdoto jautājumu.
* clear() -dzēš testu un ļauj to sākt no jauna.

4. Klase Question -darbs ar testa jautājumiem, jautājumu ievadīšana un izvadīšana.

* Question(text: String, answer: String) - izvirza jautājumu testētajam lietotājam
* getText(): String – saņem tekstu un izvada to lietotājam.
* getAnswer(answer: int): String[] - saņem atbildi no lietotāja un saglabā to.
* isCorrect(answer: String): boolean – pārbauda lietotāja atbildes pareizību.

5. Klase Test - strādā ar klasi Jautājumi, iegūst rezultātus, saglabā tos, pārbauda, vai tā ir atvērta, un sniedz jautājumus.

* saveResult() – saglabā iegūtos rezultātus;
* getResult() – parāda lietotājam viņa rezultātus.
* Test(name: String, question: List) - atrod un atver vēlamo testu;
* isOpen(): Boolean - pārbauda, vai ir atvērts tests.
* getQuestion(question: int) - izsniedz testa jautājumus.

6. Klase Tester - koordinē programmu. Tā nodrošina programmu

pamatmetožu izsaukumus un grafisko interfeisu, kā arī atrod lietotāju, pievieno lietotāju, saglabā, ielādē un izgūst rezultātus.

* main(args: String[]) – nodrošina programme pamatmetožu izsaukumus un grafisko interfeisu.
* addUser(name: String, login: String, password: String, repation: String) – pievieno lietotāju testam
* findUser(login: String, password: String): User – atrod pareizo lietotāju.
* save() – saglabā testu, rezultātus, lietotājus.
* load() – augšupielādē testu, rezultātus un lietotājus.
* getResults(): List – iegūst testa rezultātus sarakstā.

## Klašu realizēšana Java valodā

Izstrādātās klašu diagrammas (2.1. att.) realizēšanai NetBeans vidē tika izveidots projekts ar nosaukumu TestingSystem, kurā izstrādātās klases tika aprakstītas Java valodā. Šeit tiek piedāvāts klašu vispārīgā apraksta (klašu deklarācijas) programmas kods un metožu apraksti.

Pilns avota programmas kods ar klašu metožu definīcijām ir ievietots pielikumā “Pielikums A. Klašu sistēma”, bet programmas kods, kas realizē šīs klašu sistēmas testēšanu (klase ar grafisko lietotāja saskarni) ir ievietots pielikumā “Pielikums B. testa programmas kods”.

public class User {

public void User(String name, String login, String password){

}

private boolean enter(String login, String password){

return true;

}

}

public class TestedUser {

public void getAnswer(Question question, String answer){

}

public void clear(){

}

}

public class Question {

public void Question(String text, String answer){

}

public void getText(){

}

public boolean isCorrect(String answer){

return true;

}

}

public class Test {

private void saveResult(){

}

private void getResult(){

}

public void Test(String name,List question){

}

public boolean isOpen(){

return true;

}

public void getQuestion(int question){

}

}

public class Admin {

private void addQuestion(String text, String answer, int rightAnswer){

}

private void deleteUser(String login){

}

public void addText(){

}

public void startText(){

}

private void addUser(String name, String login, String password, String repeation){

}

}

public static void main(String args[]) {

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new TestingSystem().setVisible(true);

}

});

}

private void addUser(String name, String login, String password, String repation){

}

private User findUser(String login, String password){

return null;

}

private void save(){

}

private void load(){

}

private List getResult(){

return null;

}

}

# Testa programmas izstrāde

## Testēšanas metodikas

Pirms priekšmetiskās jomas klašu sistēmas testēšanas programmas izstrādes, ir jānosaka testēšanas metodika.

Ir jāatšķir koda testēšana un atkļūdošana. Atkļūdošanu veic programmētājs, izmantojot iebūvētos izstrādes vides rīkus un balstoties uz pieredzi programmas koda rakstīšanā. Būtībā runa iet par sintaktisko un semantisko kļūdu identificēšanu programmas tekstā.

Testēšana ir process, kurā nepieciešams plānot un īstenot vairākas sākotnējās procedūras, no kurām galvenā ir testa piemēru kopas izstrāde, kas veido testa plānu. Testpiemēri vairumā gadījumu ir balstīti uz sistēmas funkcionālajām prasībām un var ietekmēt dažādus attīstības līmeņus (vienības testēšana, integrācijas testēšana, sistēmas testēšana).

Projekta darba laikā nepieciešams veikt vienkāršotu vienības testēšanas versiju, kas realizē visu izstrādāto klašu metožu testēšanu. Ar testēšanas metodiku projekta darbā pieņem dažādu klases metožu izsaukšanas iespēju sarakstu ar gaidāmajiem rezultātiem un šo testēšanas izsaukumu izpildes kārtību.

## Testēšanas programmas struktūra

Testēšanas programmas projektēšana un programmatūras izstrāde ir ļoti līdzīga iepriekš aprakstītajam klašu sistēmas izstrādes procesam. Jāatzīmē tie momenti un prasības, kas pastāv, izstrādājot testēšanas programmu.

• Testēšanas programmai ir loga interfeiss, kas ietver sevī izvēlnes un interaktīvā interfeisa rīkus, ar kuru palīdzību var pārbaudīt izveidoto klašu sistēmu.

• Interfeisa elementu (pogas, saraksti, ievades lauki, dialoglodziņi, grafiskie elementi utt.) komplekts tiek noteikts, pamatojoties uz iepriekšējā sadaļā aprakstīto testēšanas metodiku.

• Testēšanas programmas interfeisa kvalitātei jābūt vienkāršai lietošanā un interaktīvai. Galvenā prasība ir spēja realizēt testēšanas metodes.

1. Ielogošanas procesa apraksts:

• Ielogošanas process paredzēts reģistrēto lietotāju pievienošanai sistēmā. Lai lietotājs varētu piekļūt testam viņam pēc ir nepieciešams autorizēties. Ielogošanas procesā lietotājs ievada iepriekš reģistrētos datus – lietotājvārdu un paroli (3.1. att. ). Ja lietotāja dati sakrīt ar validācijas datiem, tad lietotājam tiek atvērts programmas galvenais logs (3.x. att.).