LATVIJAS UNIVERSITĀTE

DATORIKAS FAKULTĀTE

**PROGRAMMĒŠANAS UZDEVUMU VIETNE “CODERFIELD”**

PROJEKTS PROGRAMMINŽENIERIJAS KURSĀ

Autori:

Artis Pauniņš ap19122

Ainārs Dukaļskis ad19068

Anete Alise Upeniece au19025

Niks Derums nd19009

Artis Jasinovičs aj19090

Darba vadītājs: Docents Dr.dat. Pēteris Paikens

RĪGA 2020

# ANOTĀCIJA

Dokumentā ir iekļauta kursa "Programminženierija" ietvaros izstrādātās sistēmas "CoderField" programmatūras prasību specifikācija un programmatūras projektējuma apraksts.

Sistēmas "CoderField" mērķis ir nodrošināt iespēju jebkuram lietotājam pilnveidot savas programmēšanas prasmes, risinot uzdevumus un saņemot risinājumu novērtējumus. Sistēmas galvenā funkcija ir iesniegto risinājumu automātiskā testēšana ar jau iepriekš uzdevuma autora definētiem testpiemēriem, kas pilnvērtīgi pārbauda risinājuma kvalitāti gan pēc izpildes laika un izmantotās brīvpiekļuves atmiņas, gan pēc testpiemēra izvaddatu pareizības. Lietotāji var uzdevumus meklēt pēc grūtības pakāpes, popularitātes un dažādiem atslēgvārdiem, lai atrastu savām prasmēm un interesēm atbilstošāko. Papildus ikviens var uzdevumus arī veidot un publiskot citiem lietotājiem risināšanai.

# ABSTRACT

The subject of this document is the software requirements specification and the software design description for the system "CoderField", which have been developed for the course "Software Engineering".

The goal of the system "CoderField" is to provide the means to any user to improve their programming skills by solving tasks and receiving the results of submitted tasks. The main function of the system is the automatic testing of submitted solutions with predefined test cases, which evaluates the quality of the solution in terms of execution time, random-access memory usage and test case output correctness. Users can search tasks by filtering by difficulty and popularity or by using keywords to find the tasks that match their skills and interests the most. In addition, anyone can create and publish tasks for other users to solve.

# SATURA RĀDĪTĀJS

[**IEVADS**](#_uclpajtyskno) **7**

[**APZĪMĒJUMU SARAKSTS**](#_k95tfhvr3q0) **8**

[**1. VISPĀRĪGS APRAKSTS**](#_7zxxg990immp) **9**

[1.1 Esošā stāvokļa apraksts](#_aq949l784su1) 9

[1.2 Pasūtītājs](#_vhjw4rdr8it0) 9

[1.3 Produkta perspektīva](#_x0thfmctxv68) 9

[1.4 Sistēmas lietotāji](#_ae1o896t23tq) 9

[1.5 Darījumprasības](#_5xq7atdzpryr) 10

[1.6 Vispārējie ierobežojumi](#_g8pv6ll0xpf5) 11

[1.7 Pieņēmumi un atkarības](#_r4rv3xv1zgrx) 11

[**2. PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA**](#_7ciq87k524r) **13**

[2.1 Konceptuālais datu bāzes apraksts](#_xbh962g6fs2m) 13

[2.2 Funkcionālās prasības](#_yub43di0paux) 15

[2.2.1 Funkciju sadalījums pa moduļiem/komponentiem](#_ngflwsly29ed) 15

[2.2.2 Lietotāju modulis](#_g44nqudop8wa) 16

[2.2.2.1 Lietotāja reģistrācija](#_pf9ruumzbwg5) 16

[2.2.2.2 Lietotāja attēla nomaiņa](#_nv00umovvvoo) 17

[2.2.2.3 Lietotāja meklēšana](#_h1tcvkrffewx) 18

[2.2.2.4 Pieteikšanās sistēmā](#_w0ek5rvo03xo) 19

[2.2.2.5 Lietotāja dzēšana](#_n4wr72x4iuzk) 20

[2.2.2.6 Lietotāja apskate](#_7m9uhgykpr1f) 21

[2.2.2.7 Lietotāju tiesību rediģēšana](#_1uotmenn88pe) 22

[2.2.3 Uzdevumu modulis](#_e0yxqw25vl9u) 23

[2.2.3.1 Uzdevuma izveidošana](#_fxx5oplwp6sx) 23

[2.2.3.2 Uzdevuma rediģēšana](#_n11occd7itvs) 24

[2.2.3.3 Uzdevuma koplietošana](#_atp112rtwa1) 26

[2.2.3.4 Uzdevuma publicēšana](#_1j75l6x7nbih) 26

[2.2.3.5 Uzdevuma dzēšana](#_h2yst2sw8a0y) 27

[2.2.3.6 Testpiemēra dzēšana](#_5go88dx2e8ht) 28

[2.2.3.7 Uzdevumu meklēšana](#_8grmrgr9721m) 29

[2.2.3.8 Uzdevuma kvalitātes novērtēšana](#_ns6zwaubyqnz) 30

[2.2.3.9 Uzdevuma apskate](#_lcpxmgdblljq) 31

[2.2.4 Risinājumu modulis](#_6kdjo9x9f8rj) 33

[2.2.4.1 Risinājuma iesniegšana un testēšana](#_r9vldr6qyk) 33

[2.2.4.2 Risinājuma dzēšana](#_jk9dbta60wr2) 35

[2.2.4.3 Risinājuma apskate](#_7zpcjwewqu63) 35

[2.2.4.4 Risinājuma meklēšana](#_90egdckl5edu) 36

[2.2.4.5 Uzdevuma grūtības pakāpes noteikšana](#_h2xm8kous1du) 37

[2.3 Nefunkcionālās prasības](#_rhucj3y6nhpq) 39

[2.3.1 Veiktspējas prasības](#_njxa8tmw2txm) 39

[2.3.2 Drošība](#_uzhsx1lemxlz) 39

[2.3.3 Pieejamības prasības](#_go7e5wq0uvgs) 39

[2.3.4 Izmantojamība](#_915za6hmjogv) 39

[**3. PROGRAMMATŪRAS PROJEKTĒJUMA APRAKSTS**](#_5hu86sjp0joa) **40**

[3.1 Datu bāzes projektējums](#_qhjxowc3h74w) 40

[3.1.1 Datu bāzes loģiskais modelis](#_w1xkwtde8p4d) 40

[3.1.2 Datu bāzes fiziskais modelis](#_75xl857yl42q) 41

[3.1.3 Datu bāzes tabulu apraksts](#_z2st1vu71fet) 42

[3.1.3.1 Lietotāju tabula](#_9cdy2jq9yteu) 42

[3.1.3.2 Uzdevumu tabula](#_uhfeja69kf67) 43

[3.1.3.3 Testpiemēru tabula](#_jv16qvpkwb9h) 44

[3.1.3.4 Testu Rezultātu tabula](#_81erxefwwpl0) 45

[3.1.3.5 Risinājumu tabula](#_8mgy6mf0r3rd) 46

[3.1.3.6 Novērtējumu tabula](#_3dv8k3uqe77d) 48

[3.1.3.7 Uzdevumu redzamības tabula](#_jy8hyw7na50g) 48

[3.1.3.8 Lietotāju Lomu tabula](#_kbp67vprsx41) 49

[3.1.3.9 Programmēšanas valodu tabula](#_ex535qrx8o7g) 50

[3.1.3.10 Rezultātu tipu tabula](#_b3e6ilhm497b) 50

[3.2 Daļējs funkciju projektējums](#_v5p2wpu18bb0) 51

[3.2.1 Ekrānskatu plūsmu projektējums](#_5zyfk1tbfafj) 51

[3.2.2 Uzdevuma rediģēšanas un testpiemēra dzēšanas projektējums](#_bhsj5w9ufdgt) 54

[3.2.3 Uzdevumu meklēšanas projektējums](#_44xcb9lt21vb) 55

[3.2.4 Risinājumu iesniegšanas un testēšanas projektējums](#_8h6v2gsdhyiq) 59

[3.2.5 Uzdevuma publiskošanas un koplietošanas projektējums](#_9oiq3879p5b) 61

[3.3 Daļējs lietotāja saskarņu projektējums](#_erql9ma54tpe) 62

[3.3.1 Par saskarnēm](#_p0eabst6buig) 62

[3.3.2 Uzdevumu meklēšana](#_gdl4r8lbd26a) 64

[3.3.3 Uzdevuma apskate](#_p1aksbc7ei1t) 65

[3.3.4 Uzdevuma rediģēšana](#_5cnfbxyu79pq) 67

[3.3.5 Risinājumu meklēšana](#_rj4rxer9mvzt) 68

[3.3.6 Risinājuma apskate](#_wict96t728dr) 69

[**IZMANTOTĀ LITERATŪRA UN AVOTI**](#_vhpma8py0o6n) **70**

# 

# IEVADS

**Nolūks**

Programmatūras prasību specifikācijas (PPS) nolūks ir aprakstīt izstrādājamās programmēšanas mācīšanās vietnes programmatūras prasības. Dokumentā tiks detalizēti aprakstītas sistēmas funkcijas un ierobežojumi, kā arī lietotāju saskarne.

Dokuments paredzēts visām ieinteresētajām pusēm, tostarp izstrādātājiem. Atbilstoši tam tiks realizēta sistēmas pirmā versija.

**Darbības sfēra**

Programmēšanas uzdevumu vietni “CoderField” izmantos cilvēki, kas vēlas apgūt programmēšanu vai māca programmēšanu. Cilvēki, kas programmēšanu mācās, tajā varēs atrast un risināt programmēšanas uzdevumus, kā arī saņemt savu iesniegto risinājumu novērtējumus, kas ir iegūti veicot automātisko testēšanu ar uzdevumam pievienotajiem testpiemēriem. Cilvēki, kas programmēšanu māca, tajā pievienos jaunus uzdevumus — gan publiskus, gan tikai viņu skolēniem pieejamus.

**Saistība ar citiem dokumentiem**

Dokumenta noformējumā ievērotas standarta LVS 68:1996 “Programmatūras prasību specifikācijas ceļvedis” vadlīnijas.

**Pārskats**

Dokuments sastāv no 4 daļām:

1. pirmajā daļā ir ievadinformācija, kas satur dokumenta nolūku, mērķi, darbības sfēru, citu dokumentu izmantošanu dokumenta tapšanas laikā;
2. otrajā daļā tiek dots ieskats, par vispārējo produkta struktūru, kā funkcijas, ierobežojumi, kā arī atkarības no citiem faktoriem;
3. trešajā nodaļā tiek norādītas funkcionālās un nefunkcionālās prasības izstrādājamai programmai;
4. ceturtajā nodaļā ir projektējuma apraksts.

# APZĪMĒJUMU SARAKSTS

Programmēšanas uzdevums — problēmsituācija, kuru iespējams reducēt uz komandrindas saskarnes programmai padodamiem ievaddatiem un kur katriem ievaddatiem ir viens pareizais rezultāts.

Testpiemērs — programmēšanas uzdevumam izstrādāta ievaddatu, izpildes noteikumu un sagaidāmo rezultātu kopa, lai dotu iespēju pārbaudīt testējamā objekta atbilstību iepriekš definētajām prasībām.

Testpiemēru kopa — visi vienam uzdevumam atbilstošie testpiemēri.

# 1. VISPĀRĪGS APRAKSTS

## 1.1 Esošā stāvokļa apraksts

Šobrīd ir pieejami vairāk nekā 30 dažādu interneta vietņu, kurās var mācīties programmēšanu un pildīt programmēšanas uzdevumus. Šajā izpētē tika apskatītas vietnes *CodeWars* un *CodeForces*, kā arī latviešu vietne *CleverCode*.

Visām vietnēm ir līdzīga pieeja uzdevumu risināšanai - ir uzdevumu saraksts un lietotājs var meklēt tos pēc nosaukuma vai arī kārtot pēc citiem kritērijiem, piemēram, grūtības pakāpes vai uzdevuma tēmas. Izņemot *CodeWars*, neviena no šīm vietnēm nepiedāvā iespēju veidot savus uzdevumus, un šī iespēja būs šeit aprakstītajai vietnei, bet *CodeWars* šādu iespēju piedāvā tikai pēc noteikta skaita uzdevumu risināšanas. Šeit aprakstītā vietne piedāvās šo iespēju uzreiz, kā lietotājs reģistrējies.

Šeit aprakstītajā sistēmā būs iespēja dot atsauksmes par uzdevumiem, kas nav pieejams citur, un atkarībā no jaunāko pozitīvo atsauksmju skaita tiek rādīti aktuālākie uzdevumi, kā arī varēs redzēt arī jaunākās uzdevumu augšupielādes.

## 1.2 Pasūtītājs

Sistēma aprakstīta pēc studentu grupas iniciatīvas programminženierijas kursa ietvaros.

## 1.3 Produkta perspektīva

Reģistrācija un pieteikšanās notiek ar *Google* kontu, izmantojot *Google API* un *OAuth 2.0* protokolu.

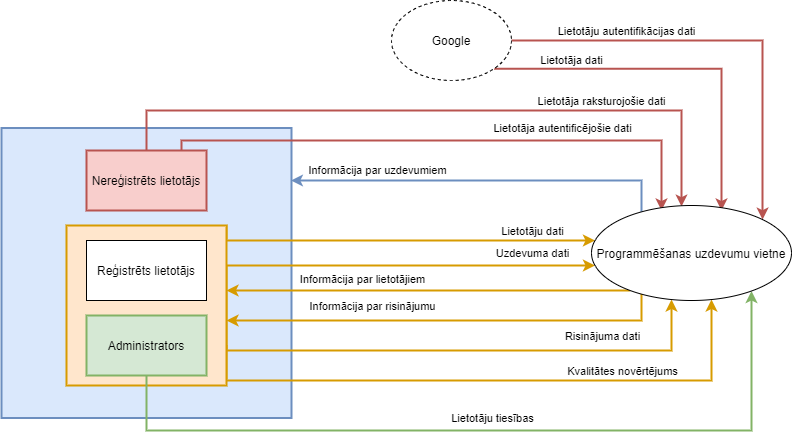
## 1.4 Sistēmas lietotāji

Reģistrēti lietotāji:

* reģistrēti lietotāji ir cilvēki ar jebkādas pieredzes daudzumu (necik vai daudz) dažādās programmēšanas valodās — gan cilvēki, kas mācās pašmācības ceļā, gan skolēni un studenti, gan profesionāļi, kas vēlas uzturēt un celt savas programmēšanas prasmes;
* lietotāji, kas pievieno uzdevumus, ir cilvēki, kas pārzina testēšanu, prot izveidot kvalitatīvu testpiemēru kopu, piemēram, programmēšanas skolotāji un pasniedzēji.

Administratori:

* administratori pārzina lietošanas noteikumus un spēj izvērtēt lietotāju rīcības un uzdevumu atbilstību tiem.



*1.1. att.* **0. līmeņa DPD diagramma**

## 1.5 Darījumprasības

Sistēmas būtiskākā prasība ir spēt droši izpildīt reģistrēta lietotāja augšupielādētu risinājumu, pārbaudīt tā izpildes pareizību pret uzdevumam piesaistītiem testpiemēriem un saglabāt iegūto rezultātu.

Sistēmai jāpiedāvā nereģistrētam lietotājam:

* apskatīt sistēmā esošos uzdevumus, kas atzīmēti kā publiski;
* apskatīt uzdevumu grūtību, kas aprēķināta, balstoties uz tā esošajiem atrisinājumiem šajā sistēmā;
* reģistrēt lietotāja kontu;
* pieteikties sistēmā.

Reģistrētam lietotājam jābūt iespējai:

* apskatīt sistēmā esošos uzdevumus, kas atzīmēti kā publiski vai kuri šim lietotājam atzīmēti kā redzami;
* augšupielādēt savu uzdevuma risinājumu — programmas kodu;
* par visiem saviem risinājumiem redzēt katra testpiemēra pārbaudes rezultātu;
* sniegt savu uzdevuma kvalitātes novērtējumu;
* pievienot sistēmai jaunu uzdevumu;
* dzēst vai labot paša pievienotos uzdevumus.

Administratoram jābūt tām pašām iespējām, kas ir reģistrētam lietotājam, kā arī iespējai:

* apskatīt jebkuru uzdevumu;
* dzēst uzdevumus;
* ierobežot citu lietotāju tiesības.

## 1.6 Vispārējie ierobežojumi

* Sistēmas darbināšanai jābūt iespējamai, to neinstalējot un izmantojot pārlūkprogrammu.
* Uzdevumu risinājumu augšupielāde iespējama programmēšanas valodā C++, Java, Python vai C#.
* Sistēmai nav nepieciešams izstrādāt jaunus kompilatorus — risinājumu kompilēšanai var izmantot gatavus kompilatorus.

## 1.7 Pieņēmumi un atkarības

* Sistēmas lietotājs izmantos kādu no trīs pārlūkprogrammām: *Google Chrome*, *Safari*, *Mozilla Firefox*.
* Sistēmas lietotājam ir stabils interneta savienojums.
* Sistēmas lietotājs prot attiecīgā uzdevuma programmēšanas valodas vienkāršo sintaksi.
* Sistēma lietotājiem nemāca programmēšanas teoriju, tāpēc tās apgūšanai lietotājs var izmantot papildu rīkus.
* Sistēmas lietotājam nav pieejama pilna uzdevuma testpiemēru kopa, bet tikai daļa no tās.
* Sistēma nepiedāvā izstrādes vidi uzdevuma risinājuma rakstīšanai.

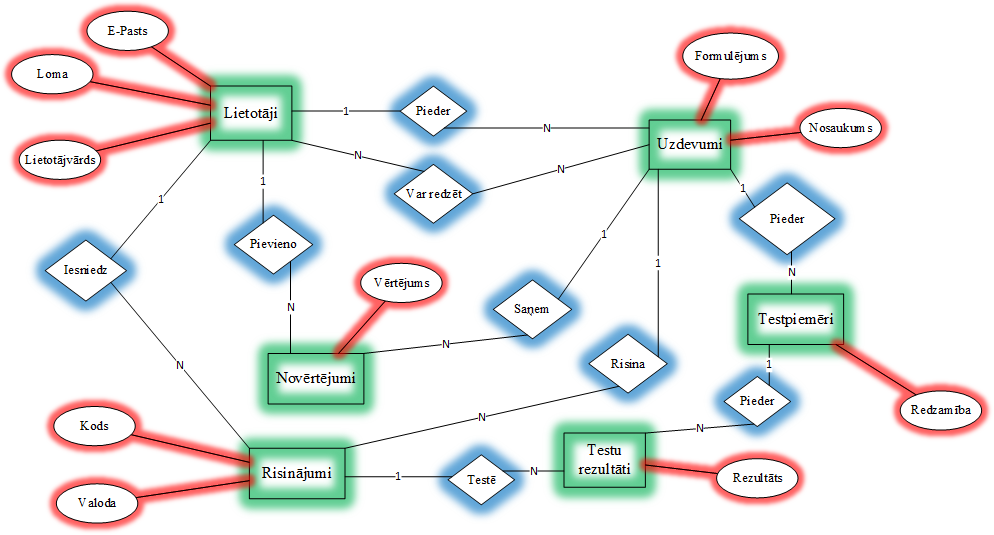
# 

# 2. PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA

## 2.1 Konceptuālais datu bāzes apraksts

Sistēmas datu bāzei ir 6 entitātes [2.1 att.]:

* Lietotāji - glabā informāciju par lietotāja vārdu, lomu (vai tas ir parasts lietotājs vai administrators) un lietotāja e-pastu.
* Uzdevumi - glabā informāciju par uzdevuma formulējumu, kā arī uzdevuma nosaukumu un redzamību. Uzdevumam ir piesaistīts lietotājs, kurš šo uzdevumu izveidoja, kā arī atsevišķie lietotāji, kas šo uzdevumu redz, ja nav norādīts, ka uzdevums ir publisks.
* Risinājums - glabā programmas kodu un valodu, kurā kods ir uzrakstīts. Jāzina, kurš lietotājs iesniedz risinājumu, kā rezultātā ir attiecība starp risinājumiem un uzdevumiem un svarīgi arī zināt, kuram uzdevumam risinājums pieder.
* Testu rezultāti - glabā to, kā uzdevums tika veikts. Šeit ir svarīgi zināt, vai izdevās programmu palaist līdz galam, kā arī to, vai programmkoda darbināšanas rezultātā ir iegūti pareizie izvaddati no dotiem ievaddatiem. Nepieciešams norādīt, kuram uzdevuma risinājumam šie testa rezultāti pieder, kā arī svarīgi zināt uzdevuma ievaddatus un izvaddatus.
* Testpiemēri - glabā uzdevuma ievaddatus un atbilstošos izvaddatus, kā arī norāda, kurus datus lietotāji redz un kurus neredz. Testēšanas dati pieder pie kāda konkrēta uzdevuma.
* Novērtējumi - glabā informāciju par lietotāja novērtējumiem par uzdevumiem.



*2.1. att.* **Konceptuālais datu bāzes modelis**

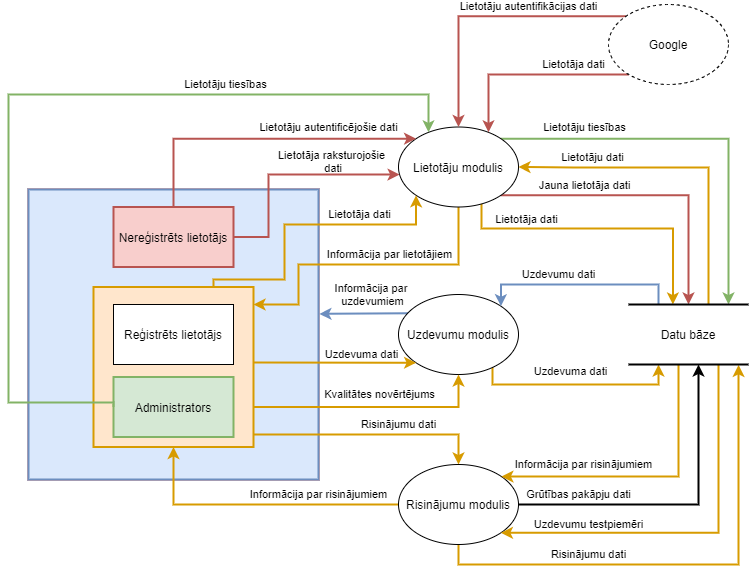
## 2.2 Funkcionālās prasības

### 2.2.1 Funkciju sadalījums pa moduļiem/komponentiem

Sistēma sastāv no 3 moduļiem — Lietotāju, Uzdevumu un Risinājumu. Lietotāju modulis atbild par lietotāju tiesībām, autorizāciju, autentifikāciju, reģistrāciju, lietotāju attēlošanu.

Uzdevumu modulis nodrošina uzdevumu pievienošanu, labošanu, dzēšanu, apskatīšanu, meklēšanu, novērtēšanu un grūtības pakāpes noteikšanu.

Risinājumu modulis nodrošina uzdevumu risinājumu augšupielādi un pārbaudi.



*2.2. att.* **1. līmeņa DPD diagramma**

### 

### 2.2.2 Lietotāju modulis

### 

*2.3. att.* **2. līmeņa DPD diagramma lietotāju modulim**

#### 2.2.2.1 Lietotāja reģistrācija

**Nolūks**

Funkcija nepieciešama, lai reģistrētu jaunu lietotāju.

**Ievade**

Funkcijai ir šādi ievaddati:

* *Google* konts;
* Lietotājvārds;
* Attēls (nav obligāts).

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Reģistrējošā lietotāja Google konts jau netiek izmantots sistēmā;
* Lietotājvārds ir unikāls;
* Attēls nepārsniedz 100 KB.

Jauna lietotāja reģistrācija notiek ar *Google API* izmantojot *OAuth 2.0* protokolu. Lietotājs tiek nosūtīts uz *Google* autorizācijas vietni, kur piesakās *Google* sistēmā ar saviem attiecīgajiem lietotāja datiem. Pie veiksmīgas pieteikšanās uz servera tiek saglabāta attiecīgā lietotāja *Google API* piekļuves pilnvara, kura tiek izmantota, lai iegūtu lietotāja *Google* konta ID un E-pastu [6].

Ja kļūdu nav, datubāzei tiek pievienots jauns lietotājs. Uz servera tiek saglabāts attēla fails ar nosaukumu “User\_[IDn]”, kur [IDn] ir lietotāja ID datubāzē.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja kļūdu nav, redzams paziņojums: “Jūs tikāt veiksmīgi reģistrēts!”;
* Ja reģistrējošā lietotāja Google konta ID sakrīt ar kāda eksistējoša sistēmas lietotāja Google konta ID, tad ir paziņojums: “Šis *Google* konts jau tiek izmantots!”;
* Ja attēls pārsniedz 100 KB, tad ir paziņojums: “Lūdzu, izvēlieties attēlu, kas nepārsniedz 100 KB”.

#### **2.2.2.2 Lietotāja attēla nomaiņa**

**Nolūks**

Funkcija nepieciešama, lai rediģētu jau esošu lietotāju.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Attēls (nav obligāts).

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Lietotājs ir reģistrēts;
* Attēls nepārsniedz 100 KB.

Ja kļūdu nav, tad pašreizējā attēla datne no failsistēmas tiek dzēsta un, ja vien funkcijai ir padots jauns attēls, tad tā datne pievienota failsistēmai ar nosaukumu “User\_[IDn]”, kur [IDn] ir lietotāja ID datubāzē.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja kļūdu nav, tad ir redzams paziņojums: “Izmaiņas saglabātas”;
* Ja attēls pārsniedz 100 KB, tad ir paziņojums: “Lūdzu, izvēlieties attēlu, kas nepārsniedz 100 KB”.

#### **2.2.2.3 Lietotāja meklēšana**

**Nolūks**

Funkcija nepieciešama, lai meklētu lietotāju.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Simbolu virkne, kuru meklēt lietotāju lietotājvārdos.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Aktīvais lietotājs ir reģistrēts.

Tiek atlasīti visi lietotāji, kuru lietotājvārdos atrodama ievadītā simbolu virkne. Lietotāju saraksts tiek kārtots alfabēta secībā.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja apstrādes rezultātā atrasts kaut viens lietotājs, tad redzams izveidotais sakārtotais lietotāju saraksts;
* Ja nav atrasts neviens lietotājs, tad ir redzams paziņojums: “Pēc dotajiem kritērijiem lietotājs nav atrasts”;
* Ja aktīvais lietotājs nav reģistrēts, tad piedāvā reģistrēties.

#### **2.2.2.4 Pieteikšanās sistēmā**

**Nolūks**

Funkcija nepieciešama, lai lietotājs pieslēgtos sistēmai.

**Ievade**

Funkcijai ir šādi ievaddati:

* *Google* konts.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Tika saņemtas autorizācijas pilnvaras.

Lietotāja pieteikšanās sistēmā notiek ar *Google API* izmantojot *OAuth 2.0* protokolu. Lietotājs tiek nosūtīts uz *Google* vietni, kur lietotājs piesakās *Google* sistēmā ar saviem attiecīgajiem lietotāja datiem, kas ļauj sistēmai saņemt autorizācijas pilnvaras [6].

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja autentifikācijas process ir bijis veiksmīgs un sistēma ir saņēmusi autorizācijas pilnvaras, tad lietotājs tiek pieteikts sistēmā;
* Ja autentifikācijas process nav bijis veiksmīgs un netika saņemtas autorizācijas pilnvaras, tad tiek uzrādīts paziņojums “Ir pieļauta kļūda, norādot Google kontu.”

#### **2.2.2.5 Lietotāja dzēšana**

**Nolūks**

Funkcija nepieciešama, lai izdzēstu esošu lietotāju.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Lietotāja, kuru dzēš, ID.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Aktīvais lietotājs ir administrators;
* Ja aktīvais lietotājs nav administrators, tad pārbauda, vai lietotājs, kuru dzēš, ir aktīvais lietotājs.

Ja pārbaudes laikā kļūdas nav radušās, no datubāzes tiek meklēta šāda informācija:

* Tabulā “Uzdevumi” meklē visus ierakstus, kur lauka “Autora\_ID” sakrīt ar dzēšamā lietotāja ID, un nomaina šo lauku vērtību uz *null*;
* Tabulā “Risinajumi” meklē visus ierakstus, kur lauka “Autora\_ID” sakrīt ar dzēšamā lietotāja ID, un nomaina šo lauku vērtību uz *null*;
* Tabulā “Novertejumi” meklē visus ierakstus, kur lauka “Lietotaja\_ID” sakrīt ar dzēšamā lietotāja ID, un izdzēš šos ierakstus;
* Tabulā “UzdevumuRedzamiba” meklē visus ierakstus, kur lauka “Lietotaja\_ID” sakrīt ar dzēšamā lietotāja ID, un izdzēš šos ierakstus.

Kad šī apstrāde ir beigusies, no datubāzes tiek izdzēsts norādītais lietotājs.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja aktīvais lietotājs ir administrators, tad redzams paziņojums “Lietotājs izdzēsts”;
* Ja lietotājs, kuru dzēš, ir aktīvais lietotājs, tad lietotāju izraksta no sistēmas un parāda paziņojumu “Lietotājs izdzēsts”;
* Ja lietotājs, kuru dzēš, nav aktīvais lietotājs, tad redzams paziņojums “Jums nav pilnvaru, lai dzēstu norādīto lietotāju”.

#### **2.2.2.6 Lietotāja apskate**

**Nolūks**

Funkcija nepieciešama, lai apskatītu lietotāja profilu.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Lietotāja, kura profilu vēlas apskatīt, ID.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Aktīvais lietotājs ir administrators;
* Ja nav administrators, tad aktīvais lietotājs ir tas pats, kura profilu apskata.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja apstrāde ir veiksmīga, tad pāriet uz norādītā lietotāja profila skatu;
* Ja apstrāde ir neveiksmīga, tad parādās paziņojums “Jums nav pilnvaru apskatīt šo profilu”.

#### **2.2.2.7 Lietotāju tiesību rediģēšana**

**Nolūks**

Funkcija nepieciešama, lai esošam lietotājam izmainītu lomu, līdz ar to lietotāja tiesības.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Lietotāja, kura lomu maina, ID;
* Loma (Parasts lietotājs = 1; Administrators = 2).

**Apstrāde**

Tiek veiktas šādas pārbaudes:

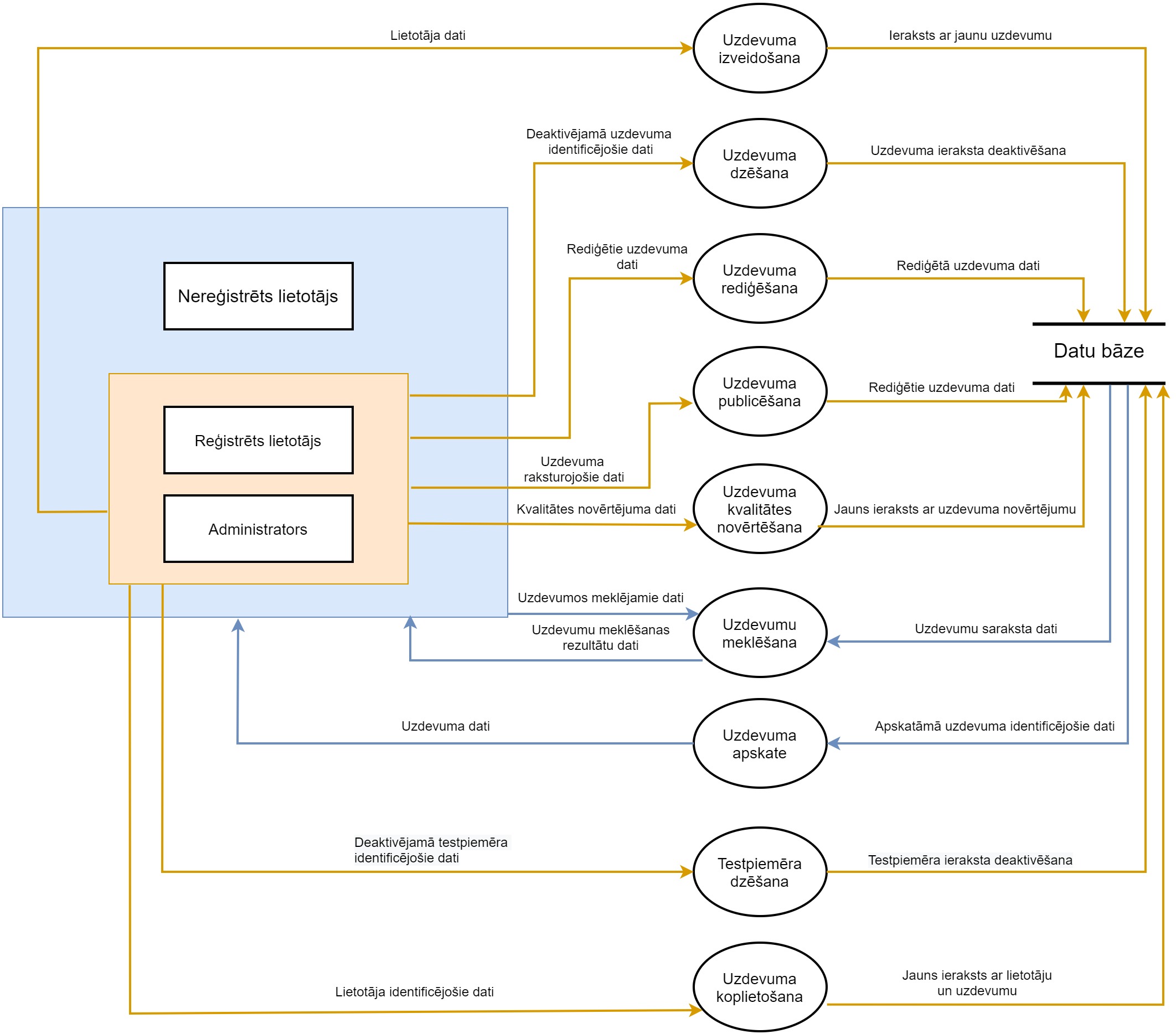
* Aktīvais lietotājs ir administrators.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja aktīvais lietotājs ir administrators, tad norādītajam lietotājam datubāze laukā “loma” tiek nomainīta vērtība atkarībā no lomas ievaddatiem. Ja vērtība ir *1*, tad arī datubāzē norādītajam lietotājam laukā “loma” ieraksta *1*. Ja vērtība ir *2*, tad *2*.
* Ja aktīvais lietotājs nav administrators, tad parādās paziņojums “Jums nav pilnvaru mainīt lietotāja tiesības.”

### 2.2.3 Uzdevumu modulis

****

*2.4. att.* **2. līmeņa DPD diagramma uzdevumu modulim**

#### **2.2.3.1 Uzdevuma izveidošana**

**Nolūks**

Funkcija nepieciešama, lai pievienotu jaunu uzdevumu.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Lietotājs ir reģistrēts.

Ja pārbaudes laikā nav radušās kļūdas, datubāzei tiek pievienots jauns tukšs uzdevums ar unikālu ģenerētu ID un atribūta “Publiskots” vērtību *false*.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja pārbaude izdevusies, notiek pāreja uz funkciju Uzdevuma rediģēšana[2.2.4.2];
* Ja lietotājs nav reģistrēts, tad lietotājam piedāvā reģistrēties.

#### **2.2.3.2 Uzdevuma rediģēšana**

**Nolūks**

Funkcija nepieciešama, lai rediģētu eksistējošu programmēšanas uzdevumu. Uzdevumu drīkst rediģēt tikmēr, kamēr uzdevums nav publicēts. Ievadāmie dati nav obligāti, lai uzdevumu varētu jebkurā brīdī saglabāt un tā rediģēšanu atsākt vēlāk.

**Ievade**

Funkcijai ir šādi ievaddati:

* Uzdevuma ID;
* Aktīvais lietotājs;
* Uzdevuma nosaukums (nav obligāts);
* Uzdevuma formulējums (nav obligāts);
* Testēšanas dati (nav obligāti):
  + Testpiemēru ievaddatu faili;
  + Testpiemēru sagaidāmo rezultātu faili;
  + Testu redzamība;
* Laika limits (milisekundēs, nav obligāts);
* Atmiņas limits (megabaitos, nav obligāts).

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Uzdevuma “Publiskots” atribūts vienāds ar *false*;
* Aktīvais lietotājs ir uzdevuma autors;
* Uzdevuma nosaukums ir teksts UTF8 un nepārsniedz 90 simbolus;
* Uzdevuma formulējums ir teksts UTF8 un nepārsniedz 16 KB;
* Laika limits ir milisekundēs vesels skaitlis no 1 līdz 20000 ieskaitot;
* Atmiņas limits ir megabaitos vesels skaitlis, no 1 līdz 512 ieskaitot;
* Testēšanas dati ir teksta UTF8 datnes un datnes kopā nepārsniedz 10 MB.

Ja pārbaudes laikā kļūdas nav radušās, datubāzē tiek saglabātas uzdevuma izmaiņas. Informācija par testpiemēriem tiek ievietota datubāzē, bet datnes tiek saglabātas uz servera. Datņu nosaukumi tiek veidoti pēc šablona test\_[uID]\_[tID].in (ievaddatu datne) un test\_[uID]\_[tID].out (izvaddatu datne), kur [uID] ir uzdevuma ID datubāzē un [tID] ir testpiemēra ID datubāzē.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja kļūdu nav, redzams paziņojums: “Izmaiņas saglabātas”;
* Ja uzdevuma “Publiskots” atribūts vienāds ar *true*, tad ir paziņojums: “Uzdevumu nedrīkst rediģēt pēc tā publicēšanas”;
* Ja aktīvais lietotājs nav uzdevuma autors, tad redzams kļūdas paziņojums “Nav rediģēšanas tiesību” un lietotājs tiek novirzīts uz vietnes sākumlapu;
* Ja kāds no datiem pārsniedz pieļaujamo atmiņas apjomu, tad ir paziņojums: “Šis lauks nedrīkst pārsniegt *X*”, kur *X* ir maksimālā lauka atmiņas apjoms, piemēram, uzdevuma formulējums pārsniedz 16 KB, tad ir paziņojums: ”Šis lauks nedrīkst pārsniegt 16 KB”.

#### 

#### **2.2.3.3 Uzdevuma koplietošana**

**Nolūks**

Funkcija nepieciešama, lai uzdevumam pievienotu lietotājus, kas uzdevumu var skatīt.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Uzdevuma ID;
* Lietotāja ID (iegūts izvēloties lietotāju no lietotāja saraksta, kas iegūts no funkcijas Lietotāja meklēšana[2.2.3.3]).

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Aktīvais lietotājs ir uzdevuma autors.

Ja pārbaudes izdevušās, lietotāja ID kopā ar uzdevuma ID tiek pievienots datubāzē.

**Izvade**

Funkcija neko neizvada.

#### **2.2.3.4 Uzdevuma publicēšana**

**Nolūks**

Funkcija nepieciešama, lai atzīmētu uzdevumu kā gatavu un citi lietotāji to varētu sākt pildīt. Uzdevumu publicēt iespējams tikai tad, ja visi uzdevuma atribūti ir netukši.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Uzdevuma ID.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Aktīvais lietotājs ir uzdevuma autors;
* Visi uzdevuma atribūti ir aizpildīti (nevar būt tukši).

Ja kļūdu nav, tad uzdevuma “Publiskots” atribūts tiek pamainīts uz *true*.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja apstrādes laikā visas veiktās pārbaudes bijušas veiksmīgas un uzdevums ticis publicēts, redzams funkcijā Uzdevuma apskate[2.2.4.9] aprakstītais skats;
* Ja aktīvais lietotājs nav uzdevuma autors, tad redzams kļūdas paziņojums “Nav rediģēšanas tiesību” un lietotājs tiek novirzīts uz vietnes sākumlapu;
* Ja visi lauki nav aizpildīti, tad redzams paziņojums: “Visiem laukiem ir jābūt aizpildītiem”.

#### **2.2.3.5 Uzdevuma dzēšana**

**Nolūks**

Funkcija nepieciešama, lai dzēstu jau esošo programmēšanas uzdevumu.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Uzdevuma ID.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Lietotājs ir uzdevuma autors vai administrators.

Ja apstrādes laikā kļūdas nav radušās, tad uzdevuma dati tiek izdzēsti no datubāzes kopā ar visiem tam piesaistītajiem testpiemēriem (un to datnēm), novērtējumiem un risinājumiem (arī risinājumu datnēm), kā arī risinājumu testu rezultātiem.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja visas pārbaudes bijušas veiksmīgas, redzams paziņojums “Uzdevums dzēsts” un lietotājs tiek novirzīts uz funkcijā Uzdevumu meklēšana[2.2.4.7] aprakstīto skatu, kur sarakstā redzami tikai uzdevumi, kuru autors ir lietotājs;
* Ja lietotājam nav tiesību dzēst šo uzdevumu, tad redzams kļūdas paziņojums “Nav dzēšanas tiesību” un lietotājs tiek novirzīts uz vietnes sākumlapu.

#### **2.2.3.6 Testpiemēra dzēšana**

**Nolūks**

Funkcija nepieciešama uzdevuma testpiemēra dzēšanai.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Testpiemēra ID.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Aktīvais lietotājs ir uzdevuma, kuram pieder testpiemērs, autors;
* Uzdevuma, kuram pieder testpiemērs, Publiskots atribūts vienāds ar *false*.

Ja pārbaudes laikā kļūdas nav radušās, tiek izdzēstas gan testpiemēra datnes, gan testpiemēra ieraksts datubāzē.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja visas pārbaudes bijušas veiksmīgas, redzams paziņojums “Testpiemērs dzēsts”;
* Ja lietotājs nav uzdevuma autors, redzams paziņojums “Nav rediģēšanas tiesību”;
* Citādi redzams kļūdas paziņojams “Testpiemēru dzēst nav iespējams”.

#### **2.2.3.7 Uzdevumu meklēšana**

**Nolūks**

Funkcija nepieciešama, lai atrastu uzdevumu pēc konkrētiem kritērijiem.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs (nav obligāts, būs tukšs, ja lietotājs nav reģistrēts);
* Simbolu virkne, kuru meklēt uzdevuma nosaukumā un tekstā (nav obligāta);
* Simbolu virkne, kuru meklēt uzdevuma autora datos (nav obligāta);
* Grūtības pakāpe: nav / viegls / vidējs / grūts (noklusētā vērtība ir *nav*);
* Uzdevuma publicēšanas datums: specifisks / visi datumi / pēdējās 24 h / pēdējā nedēļā / pēdējā mēnesī / pēdējā gadā (noklusētā vērtība ir *visi datumi*);
* Datuma izvēlne - publicēšanas sākuma datums (ievadlauks redzams tikai, ja publicēšanas datuma laukā izvēlētā vērtība ir “specifisks”);
* Datuma izvēlne - publicēšanas beigu datums (ievadlauks redzams tikai, ja publicēšanas datuma laukā izvēlētā vērtība ir “specifisks”);
* Kārtojums pēc vērtējuma: nav, dilstoši vai augoši (noklusētā vērtība ir *nav*);
* Kārtojums pēc popularitātes: nav, dilstoši vai augoši (noklusētā vērtība ir *nav*).

**Apstrāde**

Ņem vērā visus tos parametrus, kas nav tukši. Atlasa visus tos publiskos (“Publiskots” atribūts vienāds ar *true*) uzdevumus, kuriem izpildās visas netukšās prasības vienlaikus (pēc loģiskā “un” principa) un kuriem dotajam lietotājam ir piekļuves tiesības (ja pie uzdevuma norādītais lietotāju saraksts ir tukšs, tad visi lietotāji var to apskatīt, ja nav tukšs, tad pārbauda, vai aktīvais lietotājs sakrīt ar kādu no sarakstā norādītajiem lietotājiem, ir uzdevuma autors vai arī ir administrators). Var tikt atlasīti arī nepubliski uzdevumi, ja to autors ir aktīvais lietotājs.

Uzdevumi tiek kārtoti sekojoši:

* Ja ir kārtojums pēc vērtējuma, tad uzdevumus kārto augoši vai dilstoši pēc vērtējuma (skaitļa, kas iegūts, no visiem uzdevuma “patika” novērtējumiem atņemot “nepatika” novērtējumus);
* Ja ir kārtojums pēc popularitātes, tad kārto uzdevumus atkarībā no uzdevumam iesniegto risinājumu skaita augoši vai dilstoši;
* Ja atlasīti abi kārtojumi, tad vispirms kārto pēc vērtējuma un tad pēc popularitātes;
* Ja nav atlasīts neviens no kārtojumiem, tad uzdevumi tiek kārtoti alfabēta secībā pēc nosaukuma.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja apstrādes rezultātā atrasts kaut viens uzdevums, tad redzams izveidotais sakārtotais uzdevumu saraksts;
* Ja nav atrasts neviens uzdevums, tad ir paziņojums: “Pēc dotajiem kritērijiem uzdevums nav atrasts”.

#### **2.2.3.8 Uzdevuma kvalitātes novērtēšana**

**Nolūks**

Funkcija nepieciešama, lai novērtētu uzdevumu pēc tā izpildes.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Uzdevums;
* Novērtējums: nepatika / vidēji / patika.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Lietotājs iesūtījis kaut vienu šī uzdevuma risinājumu.

Ja pārbaude izdevusies, vērtējums tiek saglabāts datubāzē.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja pārbaudes laikā kļūdas nav radušās, redzams paziņojums “novērtējums saglabāts”;
* Ja lietotājs nav iesūtījis nevienu šī uzdevuma risinājumu, tad ir paziņojums: “Lai novērtētu uzdevuma kvalitāti, nepieciešams iesniegt kaut vienu uzdevuma risinājumu”.

#### **2.2.3.9 Uzdevuma apskate**

**Nolūks**

Funkcija nepieciešama, lai apskatītu noteiktu uzdevumu.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Uzdevuma ID.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Uzdevums ar šādu ID eksistē;
* Uzdevuma “Publiskots” atribūts vienāds ar *true*;
* Aktīvajam lietotājam ir piekļuves tiesības.

Ja pie dotā uzdevuma lietotāju saraksts, kuriem atļauts uzdevumu skatīt, ir tukšs, tad jebkuram lietotājam ir piekļuves tiesības dotajam uzdevumam.

Ja pie uzdevuma lietotāju saraksts, kas uzdevumu var skatīt, nav tukšs, tad tiek veikta pārbaude, vai dotais lietotājs sakrīt ar kādu no šajā sarakstā norādītajiem lietotājiem. Ja sakrīt vai arī lietotājs ir uzdevuma autors, tad lietotājam ir piekļuves tiesības šim uzdevumam, pretējā gadījumā šo tiesību nav.

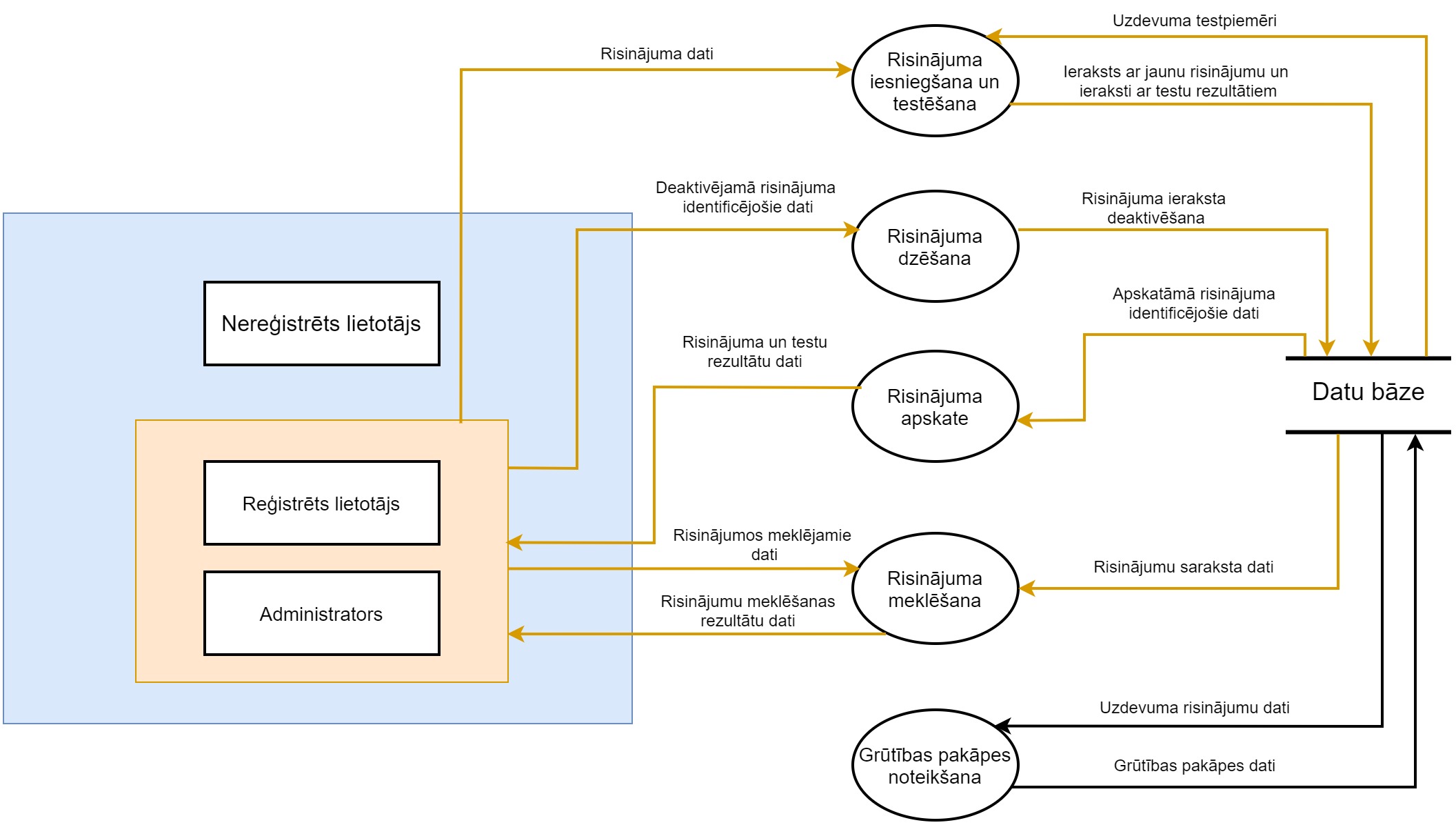
Ja lietotājs ir administrators, tam ir piekļuves tiesības jebkuram uzdevumam.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja lietotājam ir piekļuves tiesības, tad redzams uzdevuma nosaukums, formulējums, aprēķinātā grūtības pakāpe (ja uzdevumam ir iesniegts kaut viens risinājums), uzdevuma atmiņas un laika limiti, publicēšanas datums, kā arī tie testpiemēri, kas norādīti kā lietotājiem redzami;
* Ja lietotājam nav piekļuves tiesību, vai tā Publiskots atribūts vienāds ar *false*, tad ir paziņojums “Nav piekļuves tiesību”;
* Ja šāds uzdevums neeksistē, redzams paziņojums “Uzdevums neeksistē” un lietotājs tiek novirzīts uz vietnes sākumlapu.

### 2.2.4 Risinājumu modulis



*2.5. att.* **2. līmeņa DPD diagramma risinājumu modulim**

#### **2.2.4.1 Risinājuma iesniegšana un testēšana**

**Nolūks**

Funkcija nepieciešama, lai iesniegtu risinājumu un iegūtu visus testpiemēru rezultātus.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Programmēšanas valoda;
* Risinājuma programmkoda fails;
* Uzdevuma\_ID.

Programmkoda palaišanai tiek meklēti šādi ievaddati:

* Testpiemēri, kuriem laukā Uzdevuma\_ID vērtība ir vienāda ar sākotnēji norādīto Uzdevuma\_ID.

**Apstrāde**

Ja lietotājs nav pieslēdzies sistēmai, tad viņš tiek novirzīts uz pieslēgšanās lapu. Ja lietotājs ir pieslēdzies sistēmai, tiek veiktas šādas pārbaudes:

* Ir norādīta programmēšanas valoda;
* Ir augšupielādēts risinājuma programmkoda fails.

Ja pārbaudes laikā kļūdas nav radušās, informācija par risinājumu tiek ievietota datubāzē, bet risinājuma programmkoda datne tiek saglabāta uz servera. Datnes nosaukums tiek veidots pēc šablona solution\_[sID].txt, kur [sID] ir risinājuma ID datubāzē.

Ja kompilators var kompilēt iesūtīto programmkodu, tad notiek programmkoda palaišana visiem atrasto testpiemēru ievaddatiem. Iegūst šādu informāciju:

* Programmkoda izpildei nepieciešamais laiks milisekundēs;
* Programmkoda izpildei nepieciešamais brīvpiekļuves atmiņas daudzums megabaitos;
* Izpildes rezultāts [3.1.3.10].

Pēc katras programmas izpildes, datubāzē tiek ievietota tās informācija, bet risinājuma izvaddatu datne tiek saglabāta uz servera. Datnes nosaukums tiek veidoti pēc šablona result\_[rID].out, kur [rID] ir risinājuma testa rezultāta ID datubāzē.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja, iesniedzot uzdevumu, kļūdu nav, redzams paziņojums: “Risinājums iesniegts”;
* Ja kompilators nevar kompilēt programmu, redzams paziņojums “Nepareizi norādīta programmēšanas valoda vai augšupielādēts nepareizs fails”.

#### **2.2.4.2 Risinājuma dzēšana**

**Nolūks**

Funkcija nepieciešama, lai dzēstu risinājumu.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Risinājuma ID.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Aktīvais lietotājs ir risinājuma autors.

Ja apstrādes laikā kļūdas nav radušās, tad tiek izdzēsta gan risinājuma datne, gan risinājuma ieraksts datubāzē kopā ar visiem tam piesaistītajiem testu rezultātu ierakstiem.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja lietotājs ir risinājuma autors, redzams paziņojums “Testpiemērs dzēsts”;
* Ja lietotājs nav risinājuma autors, redzams paziņojums “Nav dzēšanas tiesību”, un lietotājs tiek novirzīts uz vietnes sākumlapu.

#### **2.2.4.3 Risinājuma apskate**

**Nolūks**

Funkcija nepieciešama, lai apskatītu iesniegtu risinājumu un tā testu rezultātus.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Risinājuma ID.

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Risinājums ar šādu ID eksistē;
* Aktīvais lietotājs ir risinājuma autors.

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja šāds risinājums neeksistē, redzams paziņojums “Risinājums neeksistē”;
* Ja lietotājs nav risinājuma autors, tad redzams paziņojums “Nav piekļuves tiesību”;
* Ja testēšana vēl nav beigusies, t.i., ja risinājumam atbilstošo testu rezultātu skaits ir mazāks par risinātajam uzdevumam atbilstošo testpiemēru skaitu, tad redzams paziņojums “Risinājuma testēšana nav beigusies”;
* Ja lietotājs ir risinājuma autors, tad redzams iesniegtā risinājuma uzdevuma nosaukums, risinājuma kods, datums, kurā risinājums iesniegts, katrā testā izmantotā atmiņa un patērētais laiks, kā arī redzams katra testa rezultāts, kas parāda, vai testēšanā iegūtie izvaddati sakrīt ar sagaidāmajiem un vai atmiņas un laika limiti nav pārsniegti.

#### **2.2.4.4 Risinājuma meklēšana**

**Nolūks**

Funkcija nepieciešama, lai meklētu noteiktu lietotāja iesniegto risinājumu.

**Ievade**

Funkcijai ir šādi ievaddati:

* Aktīvais lietotājs;
* Simbolu virkne, kuru meklēt uzdevuma nosaukumā un tekstā (nav obligāta).

**Apstrāde**

Tiek veiktas šādas pārbaudes:

* Lietotājs ir reģistrēts.

Ja ievadāmā simbolu virkne ir tukša, tiek atlasīti visi risinājumi, kuru autors ir aktīvais lietotājs. Ja simbolu virkne nav tukša, atlasa visus tos risinājumus, kuriem autors ir aktīvais lietotājs un kuriem ievadītā simbolu virkne atrodama risinājumam piesaistītā uzdevuma nosaukumā vai formulējumā.

Visi atlasītie risinājumi tiek kārtoti pēc to pievienošanas datuma dilstošā secībā (jaunākie vispirms).

**Izvade**

Izvade ir atkarīga no apstrādes:

* Ja apstrādes rezultātā atrasts kaut viens risinājums, tad redzams izveidotais sakārtotais risinājumu saraksts;
* Ja nav atrasts neviens risinājums, tad ir redzams paziņojums: “Pēc dotajiem kritērijiem risinājums nav atrasts”;
* Ja lietotājs nav reģistrēts, tad piedāvā reģistrēties.

#### **2.2.4.5 Uzdevuma grūtības pakāpes noteikšana**

**Nolūks**

Funkcija nepieciešama, lai aptuveni noteiktu un regulāri atjauninātu, cik sarežģīts ir uzdevums, balstoties uz jau esošajiem risinājumiem, un noteikto grūtību ierakstītu datubāzē pie atbilstošā uzdevuma. Funkcija tiek izsaukta katru reizi, kad uzdevumam tiek pievienots kāds risinājums.

**Ievade**

Funkcijai ir šādi ievaddati:

* Uzdevuma ID.

**Apstrāde**

No datubāzes tiek iegūti visi šī uzdevuma risinājumi, un tiek veikts grūtības aprēķins.

N = visu risinājumu skaits

A = atrisinājumu skaits (visi tie atrisinājumi, kuriem visi testpiemēri ir izpildīti pareizi)

R = A/N

R no 0 līdz 0,35 = grūts

R no 0,36 līdz 0,64 = vidējs

R no 0,65 līdz 1 = viegls

R tiek saglabāts datubāzē pie attiecīgā uzdevuma.

**Izvade**

Funkcija neko neizvada.

## 

## 2.3 Nefunkcionālās prasības

### 2.3.1 Veiktspējas prasības

* Jānodrošina normāla sistēmas darbība vismaz 100 lietotājiem vienlaikus;
* Testpiemēru kopa tiek izpildīta ne ilgāk kā 10 sekundēs normālajos apstākļos;
* Uzdevuma rediģējuma saglabāšana notiek ne ilgāk kā 1 sekundē;
* Uzdevumu meklēšana notiek ne ilgāk kā 1 sekundē, ja ir 5000 uzdevumi vai mazāk.

### 2.3.2 Drošība

* Autorizēšanās sistēmā notiek tikai caur *Google* kontu;
* Lietotāju privātie uzdevumi nav pieejami citiem lietotājiem, izņemot administratorus;
* Aizsardzība pret SQL injekcijām.

### 2.3.3 Pieejamības prasības

* Sistēmas atjauninājumi jāveic laikā no 02.00 līdz 06.00 pēc Latvijas laika;
* Sistēmas darbībai jābūt atbalstītai uz trijām populārākajām pārlūkprogrammām: *Chrome*, *Safari* un *Firefox*.

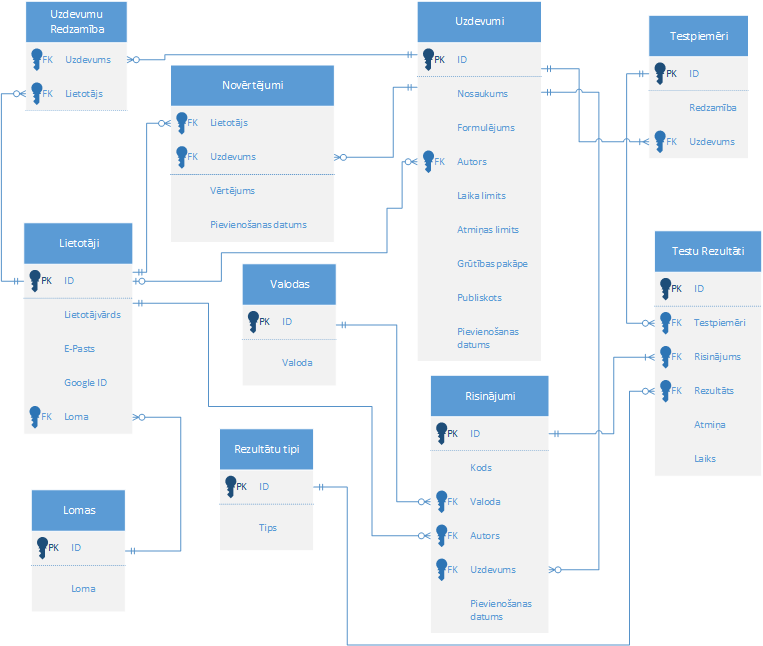
### 2.3.4 Izmantojamība

* 90 % jaunu lietotāju jāspēj lietot visas tiem pieejamās funkcijas bez palīdzības;
* Sistēmai jāatbalsta adaptīvs dizains - jāpielāgojas dažādiem ekrānu izmēriem;
* Vismaz 90 % saturs ir latviešu valodā (10 % var būt angļu valodā);
* Visi paziņojumi ir latviešu valodā.

# 3. PROGRAMMATŪRAS PROJEKTĒJUMA APRAKSTS

## 3.1 Datu bāzes projektējums

### 3.1.1 Datu bāzes loģiskais modelis



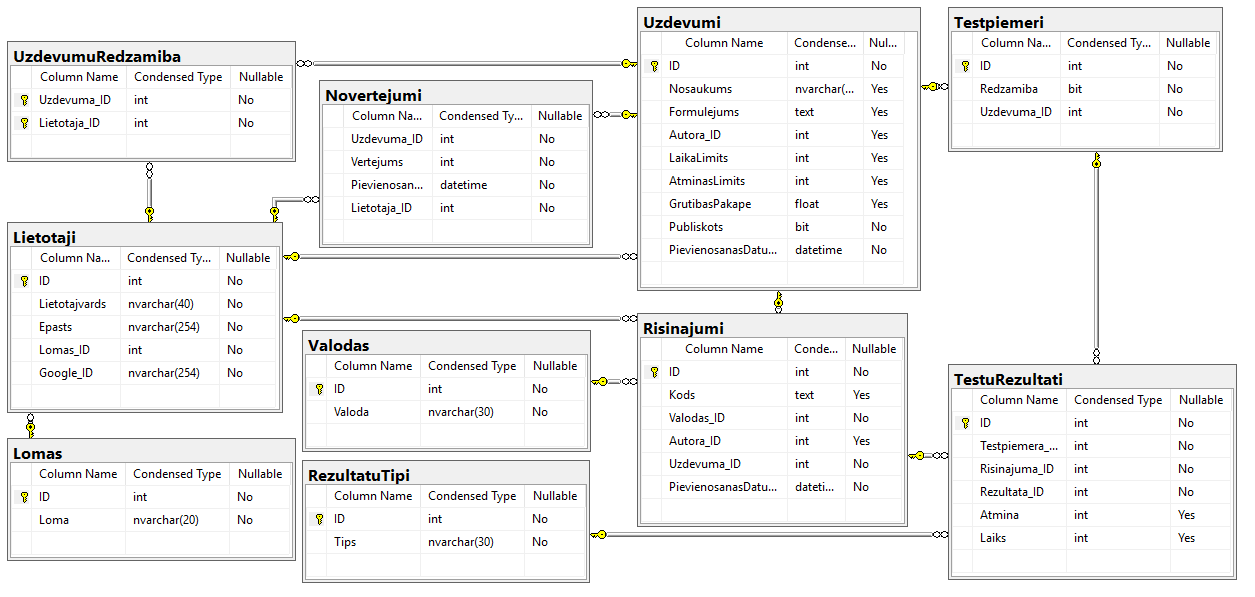
*3.1. att.* **Datu bāzes loģiskais modelis**

Pārejot no konceptuālā moduļa uz loģisko, tika ņemtas vērā attiecības. Ja no vienas entitātes uz citu entitāti aiziet ar kardinalitāti 1 un tikai 1 vai 0, vai 1, tad otrai entitātei ir ārējās atslēgas atribūts.

Loģiskajā modelī ir jauna entitāte “Uzdevumu redzamība”, jo starp entitātēm “Uzdevumi” un “Lietotāji” ir N:N attiecība “Var redzēt”. Šajā entitātē ir ārējie atribūti “Lietotājs”, kas norāda lietotāju, kurš var redzēt noteiktu uzdevumu, un “Uzdevums”, kas norāda, kuru uzdevumu lietotājs varēs redzēt.

Iekļautas arī tabulas “Lomas” un “Valodas”, kuras ir domātas lomu veidu un programmēšanas valodu glabāšanai, lai nevajadzētu pie katra jauna lietotāja un risinājuma ierakstīt lomu un valodu kā tekstu, jo tādējādi var rasties kļūdas datu apstrādē, kā arī iespējama datu pazaudēšana. Vēl ir tabula “Rezultātu tipi”, kur norādīts, kāds rezultāts ir iegūts - vai rezultāts ir pareizs vai nepareizs, vai ir pārsniegts atminās daudzums vai laiks, vai programma nokompilējās vai nē.

### 3.1.2 Datu bāzes fiziskais modelis



*3.2. att.* **Datu bāzes fiziskais modelis**

Pārejot no loģiskā modeļa uz fizisko tika ņemtas vērā tabulas un to lauki. Nosaukumi tabulām un to laukiem tika rakstīti bez atdalošiem tukšumiem un katram laukam tika pierakstīts klāt datu tips un katram laukam tika noradīts, vai laukā ir obligāti jābūt kādai vērtībai.

### 3.1.3 Datu bāzes tabulu apraksts

#### 3.1.3.1 Lietotāju tabula

Šī tabula glabā informāciju par visiem lietotājiem, kas ir reģistrējušies sistēmā.

*3.1. tab.* **Lietotāju tabula**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nosaukums datu bāzē | Apraksts un speciālās prasības | Datu tips | Atslēga | Saites tabulas nosaukums | Nullable | Unikāls |
| ID | Unikāls identifikators | int | PK |  | No | Yes |
| Lietotajvards | Glabā informāciju par lietotāja vārdu, pieslēdzoties sistēmā | nvarchar(40) |  |  | No | Yes |
| Epasts | Glabā informāciju par lietotāja e-pastu, kuru lietotājs izmantoja reģistrējoties vai var izmantot pieslēdzoties sistēmai | nvarchar(254) |  |  | No | Yes |
| Google\_ID | Glabā informāciju par lietotāja Google konta identifikatoru, kas tika saņemts kad lietotājs veiksmīgi reģistrējās sistēmā izmantojot Google kontu | nvarchar(254) |  |  | No | Yes |
| Lomas\_ID | Glabā informāciju par lietotāja lomu sistēmā, kas viņam nosaka tiesības | int | FK | Lomas | No | No |

#### 

#### 3.1.3.2 Uzdevumu tabula

Šī tabula glabā informāciju par visiem uzdevumiem, kurus ir izveidojuši sistēmā reģistrētie lietotāji.

*3.2. tab.* **Uzdevumu tabula**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nosaukums datu bāzē | Apraksts un speciālās prasības | Datu tips | Atslēga | Saites tabulas nosaukums | Nullable | Unikāls |
| ID | Unikāls identifikators | int | PK |  | No | Yes |
| Nosaukums | Glabā informāciju par uzdevuma nosaukumu | nvarchar(90) |  |  | Yes | No |
| Formulejums | Glabā informāciju par uzdevuma formulējumu | text |  |  | Yes | No |
| Autora\_ID | Norāde uz lietotāja, kurš izveidoja uzdevumu, identifikatoru. Ja lietotājs tiek izdzēsts, tad vērtība nomainās uz “null” | int | FK | Lietotaji | Yes | No |
| LaikaLimits | Glabā informāciju par laika vērtību milisekundēs, kuru nedrīkst pārsniegt, veicot risinājuma testu izpildi | int |  |  | Yes | No |
| AtminasLimits | Glabā informāciju par brīvpiekļuves atmiņas vērtību megabaitos, kuru nedrīkst pārsniegt, veicot risinājuma testu izpildi | int |  |  | Yes | No |
| GrutibasPakape | Glabā informāciju par uzdevuma grūtības pakāpi, kuru rēķina izmantojot pilnībā atrisināto uzdevumu skaitu (visi risinājumi, kuriem visi testpiemēri ir izpildīti veiksmīgi) attiecību ar visiem iesniegtajiem uzdevuma risinājumiem | float |  |  | Yes | No |
| Publiskots | Glabā informāciju par to, vai uzdevums ir nopublicēts risināšanai (vērtība ir “true”) vai arī tās autors var tikai rediģēt (vērtība ir “false”). Pēc noklusējuma vērtība ir “false” | bit |  |  | No | No |
| PievienosanasDatums | Glabā informāciju par to, kad uzdevums tika pievienots sistēmā | datetime |  |  | No | No |

#### 3.1.3.3 Testpiemēru tabula

Šī tabula glabā informāciju par visiem testpiemēriem (ievaddati un tam atbilstošie sagaidāmie izvaddati), kas ir piesaistīti pie lietotāju izveidotajiem uzdevumiem.

*3.3. tab.* **Testpiemēru tabula**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nosaukums datu bāzē | Apraksts un speciālās prasības | Datu tips | Atslēga | Saites tabulas nosaukums | Nullable | Unikāls |
| ID | Unikāls identifikators | int | PK |  | No | Yes |
| Redzamiba | Glabā informāciju par to, vai testpiemērs ir publisks. Ja vērtība ir “true”, tad testpiemērs ir publisks, ja nē, tad nav | bit |  |  | No | No |
| Uzdevuma\_ID | Norāde uz uzdevuma, kuram testpiemērs ir veidots, identifikatoru | int | FK | Uzdevumi | No | No |

#### 

#### 3.1.3.4 Testu Rezultātu tabula

Šī tabula glabā informāciju par visiem rezultātiem, kas ir iegūti no lietotāju veidotiem programmkodiem, kuri tika izmantoti, lai risinātu uzdevumus.

*3.4. tab.* **Testu rezultātu tabula**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nosaukums datu bāzē | Apraksts un speciālās prasības | Datu tips | Atslēga | Saites tabulas nosaukums | Nullable | Unikāls |
| ID | Unikāls identifikators | int | PK |  | No | Yes |
| Testpiemers | Norāde uz testpiemēru, kurš tiek testā izmantots, identifikatoru | int | FK | Testpiemeri | No | No |
| Risinajums | Norāde uz risinājuma, kura kods tiek darbināts, identifikatoru | int | FK | Risinajumi | No | No |
| Rezultats | Norāde uz kodifikatoru tabulu, kurā glabājas rezultātu tipi | int | FK | RezultatuTipi | No | No |
| Atmina | Glabā informāciju par brīvpiekļuves atmiņu megabaitos, kas tika izmantota, darbinot risinājuma kodu | int |  |  | Yes | No |
| Laiks | Glabā informāciju par laiku milisekundēs, kas tika izmantota, darbinot risinājuma kodu | int |  |  | Yes | No |

#### 

#### 3.1.3.5 Risinājumu tabula

Šī tabula glabā informāciju par visiem risinājumu kodiem, kurus iesniedza lietotāji kādam konkrētam uzdevumam.

*3.5. tab.* **Risinājumu tabula**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nosaukums datu bāzē | Apraksts un speciālās prasības | Datu tips | Atslēga | Saites tabulas nosaukums | Nullable | Unikāls |
| ID | Unikāls identifikators | int | PK |  | No | Yes |
| Kods | Lietotāja iesniegtais uzdevuma atrisinājuma kods | text |  |  | Yes | No |
| Valoda\_ID | Norāde uz kodifikatoru tabulu, kur ir norādīta izmantotā programmēšanas valoda risinājuma kodā | int | FK | Valodas | No | No |
| Autora\_ID | Norāde uz lietotāja, kurš iesniedza risinājuma kodu, identifikatoru. Ja lietotājs tiek izdzēsts, tad vērtība nomainās uz “null” | int | FK | Lietotaji | Yes | No |
| Uzdevuma\_ID | Norāde uz uzdevuma, kuram kods tiek iesniegts, identifikatoru | int | FK | Uzdevumi | No | No |
| PievienosanasDatums | Glabā informāciju par to, kad risinājums tika iesniegts | datetime |  |  | No | No |

#### 3.1.3.6 Novērtējumu tabula

Šī tabula glabā informāciju par visiem novērtējumiem, kurus lietotāji ir iesnieguši konkrētiem uzdevumiem.

*3.6. tab.* **Novērtējumu tabula**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nosaukums datu bāzē | Apraksts un speciālās prasības | Datu tips | Atslēga | Saites tabulas nosaukums | Nullable | Unikāls |
| Lietotaja\_ID | Norāde uz lietotāja, kurš iesniedz novērtējumu, identifikatoru | int | FK | Lietotaji | No | No |
| Uzdevuma\_ID | Norāde uz uzdevuma, kuram vērtējums ir iesniegts, identifikatoru | int | FK | Uzdevumi | No | No |
| Vertejums | Glabā informāciju par vērtējuma vērtību | int |  |  | No | No |
| PievienosanasDatums | Glabā vērtību par to, kad vertējums tika iesniegts | datetime |  |  | No | No |

#### 

#### 3.1.3.7 Uzdevumu redzamības tabula

Šī tabula glabā informāciju par visiem uzdevumiem, kuriem ir tikai konkrēti lietotāji vai to grupas, kuras šos uzdevumus var redzēt.

*3.7. tab.* **Uzdevumu redzamības tabula**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nosaukums datu bāzē | Apraksts un speciālās prasības | Datu tips | Atslēga | Saites tabulas nosaukums | Nullable | Unikāls |
| Uzdevuma\_ID | Norāde uz uzdevuma, kuram ir norādīts lietotājs laukā “Lietotājs”, kurš šo uzdevumu var redzēt, identifikatoru | int | FK | Uzdevumi | No | No |
| Lietotaja\_ID | Norāde uz lietotāja, kuram piešķirtas tiesības redzēt uzdevumu, kurš norādīts laukā “Uzdevums”, identifikatoru | int | FK | Lietotaji | No | No |

#### 3.1.3.8 Lietotāju Lomu tabula

Šī kodifikatoru tabula glabā informāciju par lietotāju lomām. Būs divas lietotāju lomas: “Reģistrēts lietotājs” un “Administrators”.

*3.8. tab.* **Lietotāju lomu tabula**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nosaukums datu bāzē | Apraksts un speciālās prasības | Datu tips | Atslēga | Nullable | Unikāls |
| ID | Unikāls identifikators | int | PK | No | Yes |
| Loma | Glabā informāciju par lietotāja lomām.  Visas vērtības:   * Reģistrēts * Administrators | nvarchar(20) |  | No | Yes |

#### 3.1.3.9 Programmēšanas valodu tabula

Šī kodifikatoru tabula glabā informāciju par sistēmā atbalstītajām programmēšanas valodām. Jāglabā vērtības “C++”, “Java”, “Python” un “C#”.

*3.9. tab.* **Programmēšanas valodu tabula**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nosaukums datu bāzē | Apraksts un speciālās prasības | Datu tips | Atslēga | Nullable | Unikāls |
| ID | Unikāls identifikators | int | PK | No | Yes |
| Valoda | Glabā informāciju par sistēmā atbalstītām programmēšanas valodām | nvarchar(30) |  | No | Yes |

#### 3.1.3.10 Rezultātu tipu tabula

Šī kodifikatoru tabula glabā informāciju par iesniegto rezultātu tipiem. Jāglabā vērtības “Pareizs rezultāts”, “Pārsniegts laika limits”, “Pārsniegts atmiņas limits”, “Nepareizs rezultāts”, “Izpildlaika kļūda”, “Kompilācijas kļūda”.

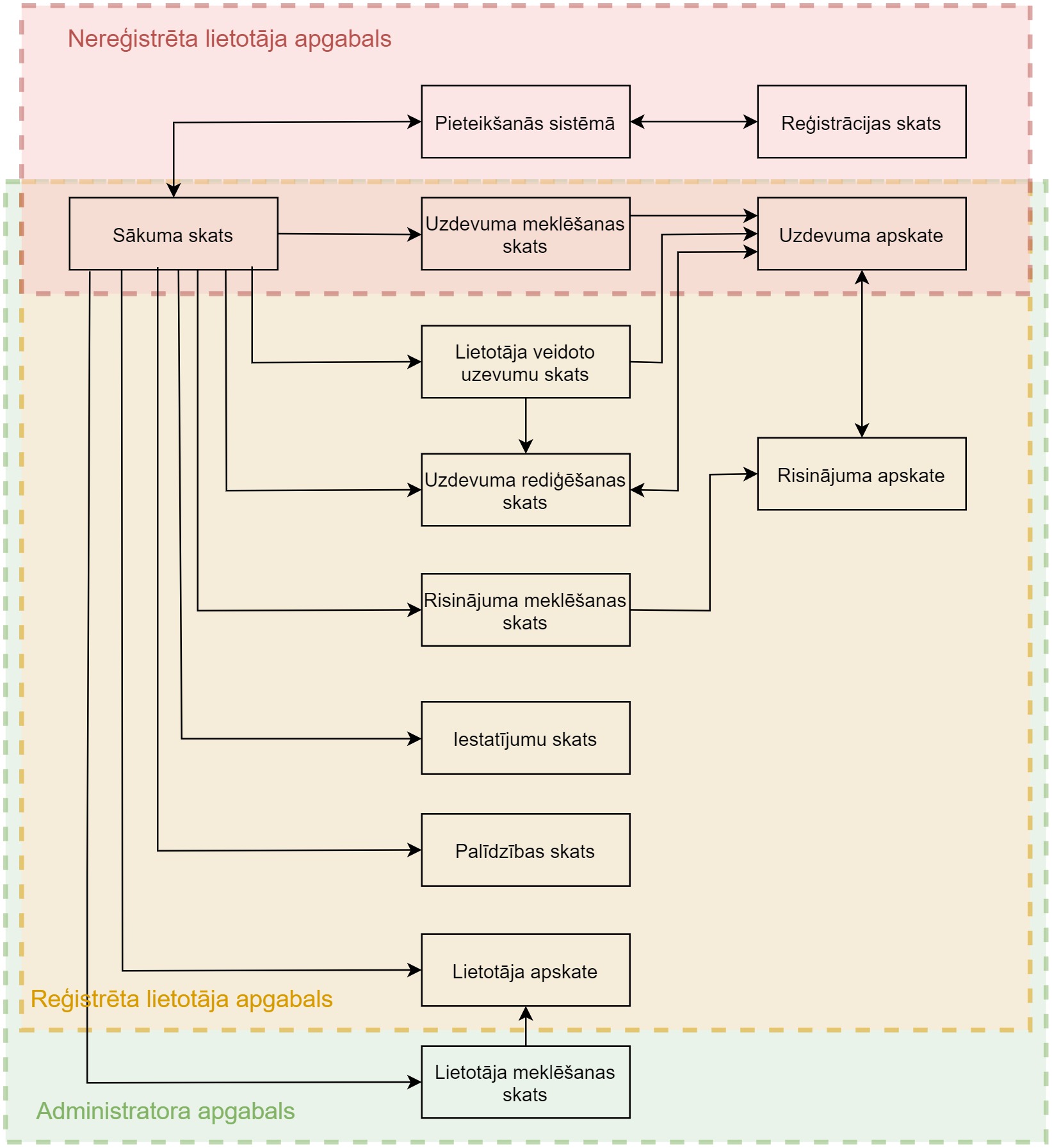
*3.10. tab.* **Rezultātu tipu tabula**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nosaukums datu bāzē | Apraksts un speciālās prasības | Datu tips | Atslēga | Nullable | Unikāls |
| ID | Unikāls identifikators | int | PK | No | Yes |
| Tips | Glabā informāciju par iesniegto risinājumu rezultātu tipiem | nvarchar(30) |  | No | Yes |

## 

## 3.2 Daļējs funkciju projektējums

### 3.2.1 Ekrānskatu plūsmu projektējums

****

*3.3. att.* **Ekrānskatu plūsmu diagramma**

Ekrānskatu plūsmu diagramma attēlo saistības starp skatiem un dažādos veidus, kā lietotājiem pie šiem skatiem nokļūt.

Diagramma sadalīta 3 apgabalos - nereģistrēta lietotāja, reģistrēta lietotāja un administratora. Katrā apgabalā iekļautie skati redzami tikai attiecīgajai lietotāju grupai. Apgabali var arī pārklāties. Tie skati, kas pieejami ar vienu no soli no sākuma skata un ir lietotājam redzami, ir atrodami galvenajā izvēlnē, un šajos ekrānskatos var nokļūt jebkurā laikā no jebkura cita skata.

Nereģistrēts lietotājs var piekļūt tikai pieteikšanās, reģistrācijas, uzdevumu meklēšanas un uzdevuma apskates skatiem. Reģistrēšanās skatā var nokļūt tikai no pieteikšanās skata. Pēc pieteikšanās/reģistrēšanās, lietotājs tiek novirzīts uz sākuma skatu un atrodas reģistrētā lietotāja apgabalā vai administratora apgabalā.

Detalizētāk aprakstīti sarežģītākie skati (uz kuriem ved vairāk nekā viena bultiņa).

Uzdevuma apskates skatam var piekļūt:

* No uzdevumu meklēšanas skata, uzklikšķinot uz attiecīgā uzdevuma;
* No lietotāja veidoto uzdevumu skata, uzklikšķinot uz attiecīgā publiskotā uzdevuma;
* No uzdevumu rediģēšanas skata pēc veiksmīgas uzdevuma publiskošanas;
* No iesniegta risinājuma apskates, uzklikšķinot uz uzdevuma.

Uzdevuma apskates skats pieejams tikai tiem uzdevumiem, kuri ir publiskoti, savukārt uzdevuma rediģēšanas skats pieejams nepubliskotiem un ierobežotā variantā arī publiskiem uzdevumiem.

Uzdevuma rediģēšanas skatam var piekļūt:

* Galvenajā izvēlnē izvēloties “Izveidot uzdevumu”, tiks izveidots jauns tukšs uzdevums un lietotājs tiks novirzīts uz uzdevuma rediģēšanas skatu;
* No lietotāja veidoto uzdevumu skata, uzklikšķinot uz attiecīgā nepubliskotā uzdevuma;
* No publiskota uzdevuma apskates, uzklikšķinot uz pogas “Rediģēt”, kas atvērs ierobežoto rediģēšanas skatu, kurā atļauts tikai pievienot un noņemt lietotājus, kuriem uzdevums redzams (uzdevuma nosacījumus un testpiemērus publiskotam uzdevumam rediģēt nedrīkst).

Risinājuma apskates skatam iespējams piekļūt:

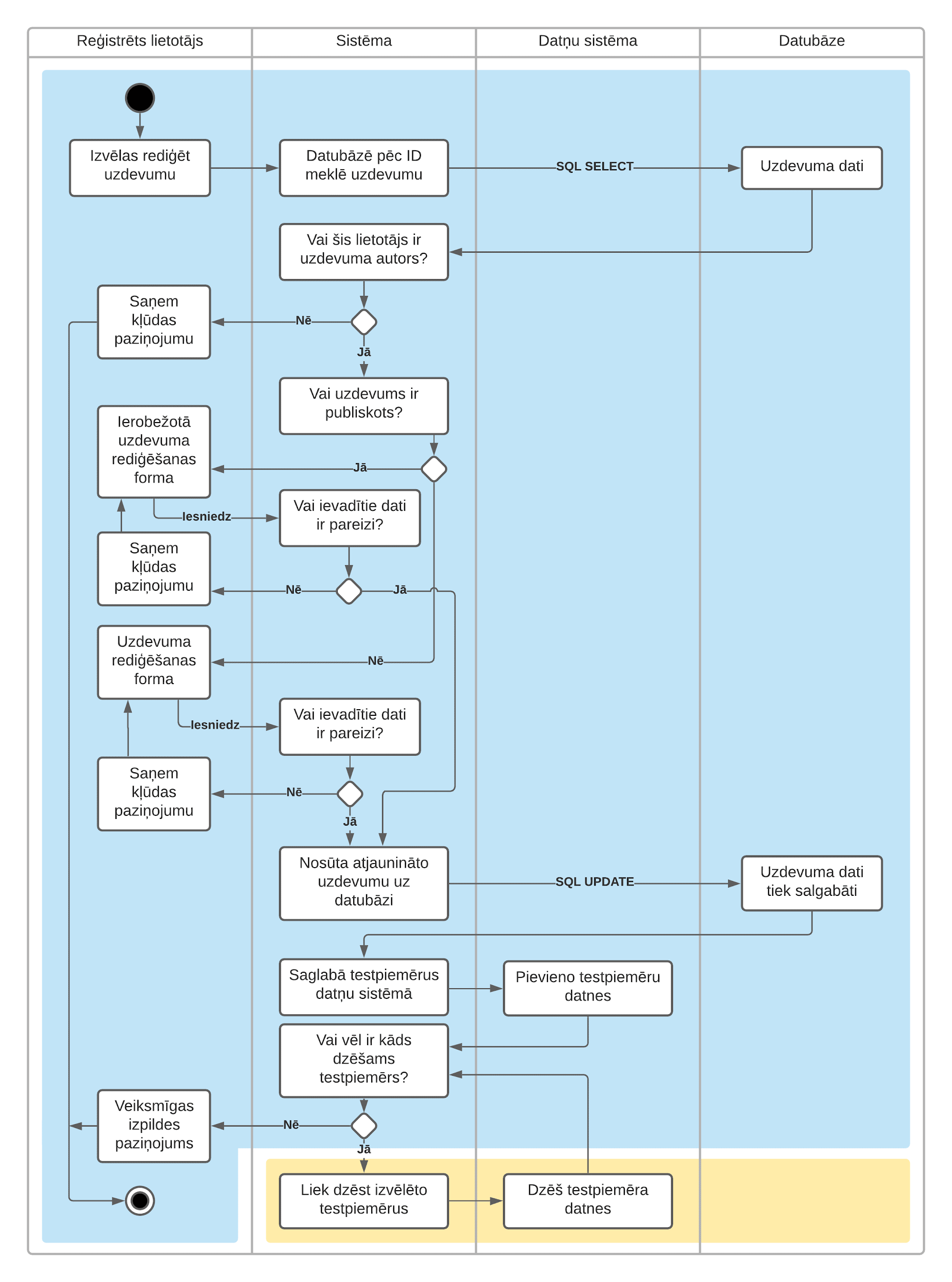
* No uzdevuma apskates skata, iesniedzot uzdevuma risinājumu. Pēc risinājuma testēšanas beigām lietotājs tiks novirzīts uz risinājuma apskati;
* No risinājumu meklēšanas skata, kurā redzami visi lietotāja iesniegtie risinājumi, uzklikšķinot uz attiecīgā risinājuma.

Lietotāja apskates skatam iespējams piekļūt:

* Lietotājam izvēlnē uzklikšķinot uz pogas “Profils”;
* Administratoram no lietotāju meklēšanas skata, uzklikšķinot uz attiecīgā lietotāja.

Lietotājs drīkst apskatīt tikai savu profilu, savukārt administrators drīkst apskatīt jebkura lietotāja profilu.

### 3.2.2 Uzdevuma rediģēšanas un testpiemēra dzēšanas projektējums

****

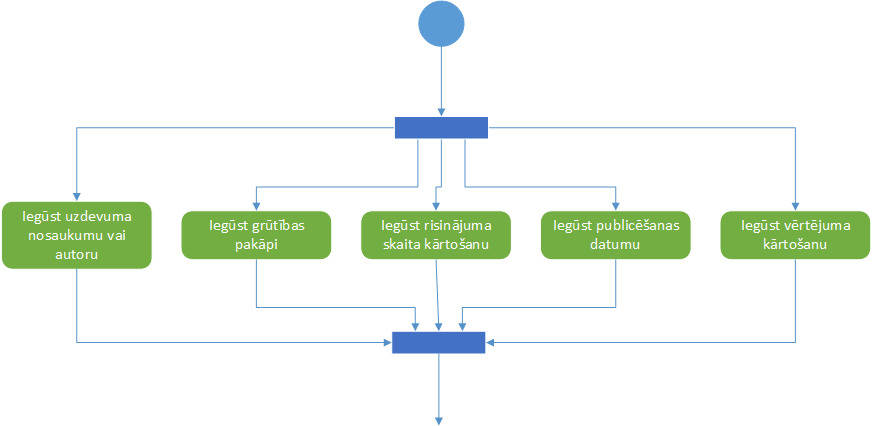
*3.4. att.* **Uzdevuma rediģēšanas un testpiemēra dzēšanas peldceliņu diagramma**

Peldceliņu diagrammā ar zilu iekrāsota funkcionalitāte, ko veic uzdevuma rediģēšanas funkcija, bet ar dzeltenu — funkcionalitāte, kuru veic testpiemēra dzēšanas funkcija. Diagrammā nodalītas četras daļas: lietotāja pusē redzamais un veicamais, sistēmā notiekošais, darbības ar datņu sistēmu un darbības ar datubāzi. Uzdevumu rediģēšanu un testpiemēru dzēšanu var veikt tikai reģistrēts lietotājs (t.sk. administrators), kas parādās diagrammas kolonas nosaukumā.

Uzdevumu drīkst rediģēt tikai lietotājs, kurš ir uzdevuma autors. Atkarībā no tā, vai uzdevums ir publiskots vai nē, lietotājam tiek attēlota pilnā uzdevuma rediģēšanas saskarne vai ierobežotā, kas ļauj tikai mainīt sarakstu ar lietotājiem, kam uzdevums pieejams.

Kad forma pareizi aizpildīta un iesniegta, uzdevuma dati datubāzē tiek atjaunināti, tiek saglabātas testpiemēru datnes datņu sistēmā un par katru dzēšamo testpiemēru tiek izsaukta testpiemēra dzēšanas funkcija, kas tos pa vienam dzēš no datņu sistēmas.

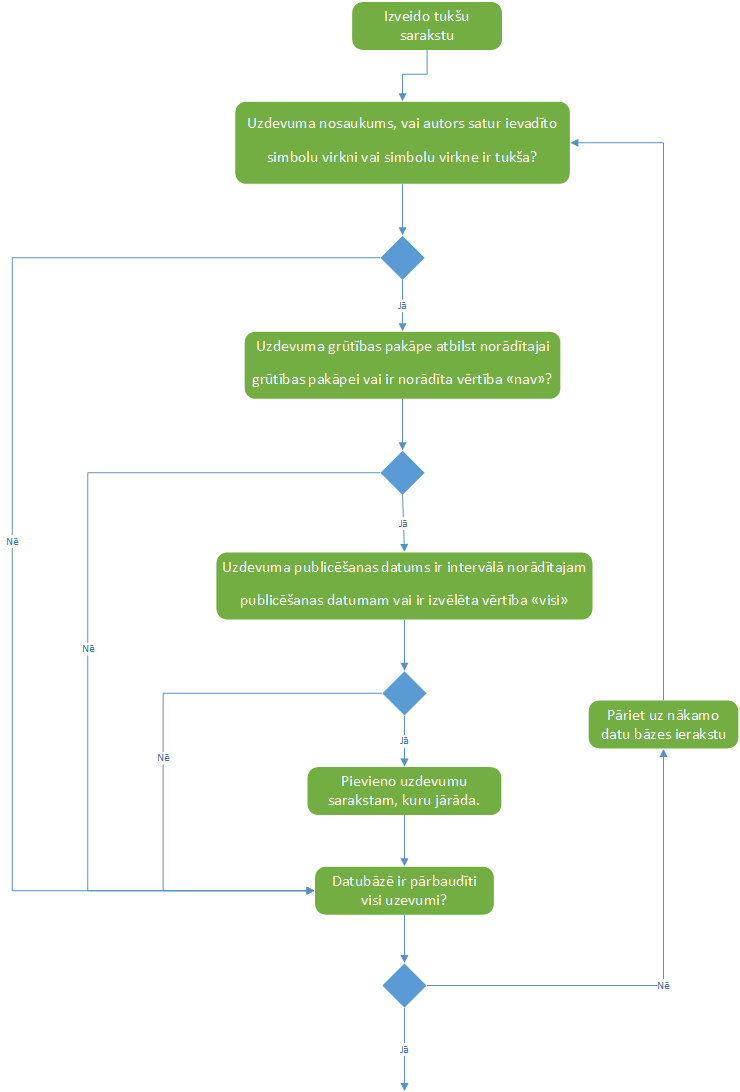
### 3.2.3 Uzdevumu meklēšanas projektējums

****

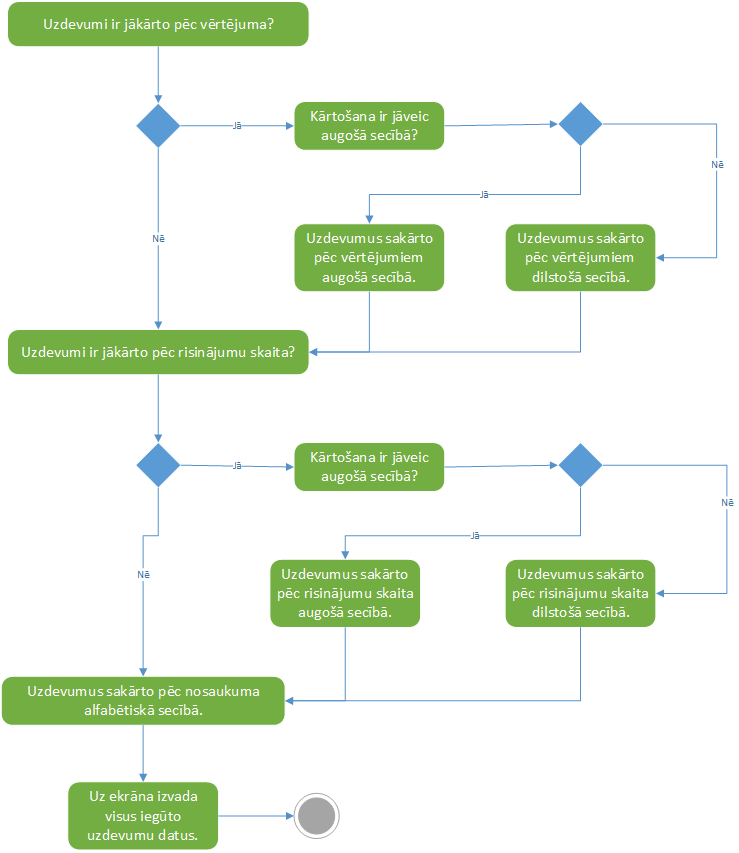
*3.5. att.* **Uzdevumu meklēšanas secību diagramma (ievaddatu iegūšana)**

Šajā diagrammas daļā ir redzams, ka tiek iegūta informācija par ievadītajiem datiem uzdevumu meklēšanas filtrā [3.3.2]. Iegūtie dati tiek fiksēti un funkcija sāk uzdevumu datu iegūšanu no datubāzes [3.6 att.].

Tiek izveidots tukšs saraksts, kurā tiks glabāti uzdevumu dati. Vispirms pārbauda, vai uzdevuma nosaukums vai uzdevuma autora vārds satur meklēšanas laukā ierakstīto simbolu virkni. Ja šo simbolu virkni satur, vai arī meklēšanas laukā nekas nav ierakstīts, tad tiek pārbaudīta uzdevuma grūtības pakāpe. Ja tā atbilst filtrā norādītajai vērtībai, vai arī filtrā ir izvēlēta vērtība “nav”, tad tiek pārbaudīts uzdevuma publicēšanas datums. Ja publicēšanas datums atbilst norādītajam intervālam, vai arī izvēlētais intervāls ir visi datumi, tad uzdevums atbilst visiem kritērijiem un tas tiek pievienots sarakstam. Šī funkcijas daļa turpinās tik ilgi, kamēr ir iziets cauri visiem uzdevumiem. Tad sāk uzdevumu kārtošanu.

****

*3.6. att.* **Uzdevumu meklēšanas secību diagramma (uzdevumu datu iegūšana no datubāzes)**

****

*3.7. att.* **Uzdevumu meklēšanas secību diagramma (uzdevumu kārtošana)**

Uzdevumus var kārtot pēc diviem kritērijiem - augstākā vērtējuma un risinājumu skaita. Vispirms tiek pārbaudīts, vai ir jākārto pēc kāda no šiem kritērijiem. Ja ir, tad jāpārbauda, vai jākārto augoši vai dilstoši un tad visus uzdevumus sakārto atbilstošā secībā. Prioritātes kārtošanai ir sekojošā secībā: pēc vērtējuma, pēc risinājumu skaita, pēc uzdevuma nosaukuma alfabētiskā secībā. Ja nav jākārto pēc augstākā vērtējuma vai risinājumu skaita, tad pēc noklusējuma visus uzdevumus sakārto pēc nosaukuma alfabētiskā secībā.

### 3.2.4 Risinājumu iesniegšanas un testēšanas projektējums

## 

*3.8. att.* **Risinājumu iesniegšanas un testēšanas aktivitāšu diagramma**

Šī diagramma attēlo kā lietotājs iesniedz savu risinājumu kādam noteiktam uzdevumam, un kā šis uzdevuma risinājums tiek attiecīgi testēts.

Sistēma pārbauda vai, iesniedzot risinājumu, aktīvais lietotājs ir pieteicies sistēmā, ja lietotājs nav pieslēdzies sistēmai, tad viņš tiek novirzīts uz pieslēgšanās lapu. Pēc tam seko nepieciešamo ievaddatu pārbaude, t.i. tiek pārbaudīts vai tika norādīta risinājuma programmēšanas valoda, un vai risinājumam tika pievienots programmkoda fails, ja trūkst kādi no ievaddatiem, tad tiek izvadīts kļūdas paziņojums.

Pēc risinājuma pārbaudes datubāzē tiek ievietota informācija par šo risinājumu [3.1.3.5] un uz servera tiek saglabāts iesūtītā risinājuma programmkods.

Tad notiek programmkoda kompilēšana, ja tā notiek veiksmīgi tad risinājums tiek pārbaudīts ar uzdevumam piesaistītajiem testpiemēriem un, ja nav programmas izpildes laikā radušās kļūdas, tad tiek saglabāta izvaddatu datne serverī un informācija par testiem datubāzē [3.1.3.4].

Ja programmkods nekompilējas vai ja testpiemēru izpildes laikā ir radušās kļūdas tiek izvadīts kļūdas paziņojums un attiecīgā informācija par kļūdu tiek saglabāta datubāzē [3.1.3.4].

## 

### 3.2.5 Uzdevuma publiskošanas un koplietošanas projektējums

### 

*3.9. att.* **Uzdevuma publiskošanas un koplietošanas komunikāciju diagramma**

Šajā diagrammā ir 2 objekti: uzdevums un datubāze. Pirmā ziņa “izveidot\_uzdevumu()” paziņo, ka ir izveidots uzdevums. Pēc tam var rediģēt šo uzdevumu. Ja lietotājs negrib, lai uzdevums ir redzams visiem, tad ir nepieciešams atlasīt lietotājus kuri var pildīt uzdevumu. Visbeidzot uzdevumu pievieno datubāzei un to publicē [2.2.4.4]. Ja tika atlasīti lietotāji, tad uzdevumu arī koplieto [2.2.4.3].

## 3.3 Daļējs lietotāja saskarņu projektējums

### 3.3.1 Par saskarnēm

## 

*3.8. att.* **Saskarnes šablons**

Saskarnēm pamatstruktūra ir redzama 3.3. att. Pašā lapas augšā ir Nosaukums, tad seko izvēlnes rīkjosla ar šādām izvēlnēm:

* “Sākumlapa”, kas aiziet atpakaļ uz sākumlapu;
* “Uzdevumi”, kas aiziet uz uzdevumu meklēšanas lapu [3.3.2.];
* “Palīdzība”, kas aiziet uz statisku palīdzības lapu, kur ir rakstīti visbiežāk uzdotie jautājumi, risinājumu rezultātu atšifrējumi, u. tml.;
* Ja lietotājs ir administrators, tad papildus šīm vēl redzama opcija “Lietotāji”, kas aiziet uz lietotāju meklēšanas lapu.

Lapas labajā pusē ir logi ar uzdevumiem, kas sakārtoti pēc norādītā kritērija. Pavisam šeit katrā logā varēs redzēt 10 vienumus, kas iegūti, izmantojot uzdevumu meklēšanas funkciju [2.2.4.7] pēc šādiem kritērijiem:

* Jaunākās augšupielādes - kārtojums pēc jaunākā datuma;
* Aktuālākie pēdējā nedēļā - kārtojums dilstoši pēc vērtējuma, “publicēšanas datums” ir *pēdējā nedēļā*;
* Visvairāk izpildītie - kārtojums dilstoši pēc risinājumu skaita.

Katra saraksta apakšā ir saite uz uzdevumu meklēšanas lapu[3.3.2], kur filtra iestatījumi ir koriģēti pēc attiecīgā kritērija. Šie logi ar uzdevumiem nav sastopami šādās saskarnēs:

* Uzdevumu meklēšana;
* Uzdevuma apskate.

Pieslēdzoties sistēmā labajā augšējā stūrī parādīsies ikona ar cilvēciņu. Uzklikšķinot uz šīs ikonas parādīsies izvēlne. Izvēlnes opcijas:

* Profils - aiziet uz lietotāja profilu;
* Iestatījumi - aiziet uz iestatījumu lapu;
* Izveidot uzdevumu - izveido jaunu uzdevumu un aiziet uz uzdevuma rediģēšanas lapu [3.3.4];
* Mani uzdevumi - aiziet uz skatu, kurā redzami lietotāja izveidotie uzdevumi;
* Mani risinājumi - aiziet uz lietotāja iesniegto uzdevumu risinājumu apskates lapu[3.3.5].

### 

### 3.3.2 Uzdevumu meklēšana

## 

*3.9. att.* **Uzdevumu meklēšanas saskarne**

Šeit tiek attēlots, kāda uzdevuma informācija tabulā tiek attēlota un filtrs, kurā var norādīt papildus meklēšanas kritērijus. Grupā “Kārtošana” abiem kritērijiem ir izvēles “nav”, kas nekārto, “Augoši”, kas kārto datus augošā secībā un “Dilstoši”, kas kārto datus dilstošā secībā. “Publicēšanas datums” ir pieejamas šādas izvēles:

* Pēdējās 24h;
* Pēdējā nedēļā;
* Pēdējā mēnesī;
* Pēdējā gadā;
* Specifisks.

Ja tiek izvēlēts “Specifisks”, tad parādās opcijas “Sākuma datums” un “Beigu datums”. Ja netiek izvēlēts “Specifisks”, tad šo lauku nav.

Uzdevumu tabulā kolonnā vērtējums pirmais skaitlis norāda starpību starp vērtējumu “patika” un “nepatika” skaitu. Iekavās ir rakstīts iesniegto vērtējumu skaits. Risinājumu kolonna ir norādīts visu iesniegto risinājumu skaits. Visi uzdevumi ir sakārtoti alfabētiskā secībā.

### 3.3.3 Uzdevuma apskate

## 

*3.10. att.* **Uzdevuma apskates saskarne**

Šeit tiek attēlots, kā uzdevuma informācija tiek reprezentēta.

Sākumā ir nosaukums, pēc kura seko uzdevuma nosacījumi. Tad ir informācija par testpiemēriem.

Pa labi no uzdevuma nosaukuma ir publicēšanas datums un pa labi virs testpiemēru tabulas ir norādīta informācija programmas izpildes laika un atmiņas limiti. Lapas labajā pusē ir iesniegšanas lauks, kurā norāda programmēšanas valodu un poga, ar kuru notiek programmas koda faila augšupielāde. Ja fails ir augšupielādēts, tad zem pievienošanas pogas parādīsies paziņojums ”Ir pievienots ‘FailaNosaukums.paplašinājums’ ’”. Kad fails ir pievienots, tad poga “Iesūtīt”, kas iesniedz risinājumu testēšanai, tiks iespējota.

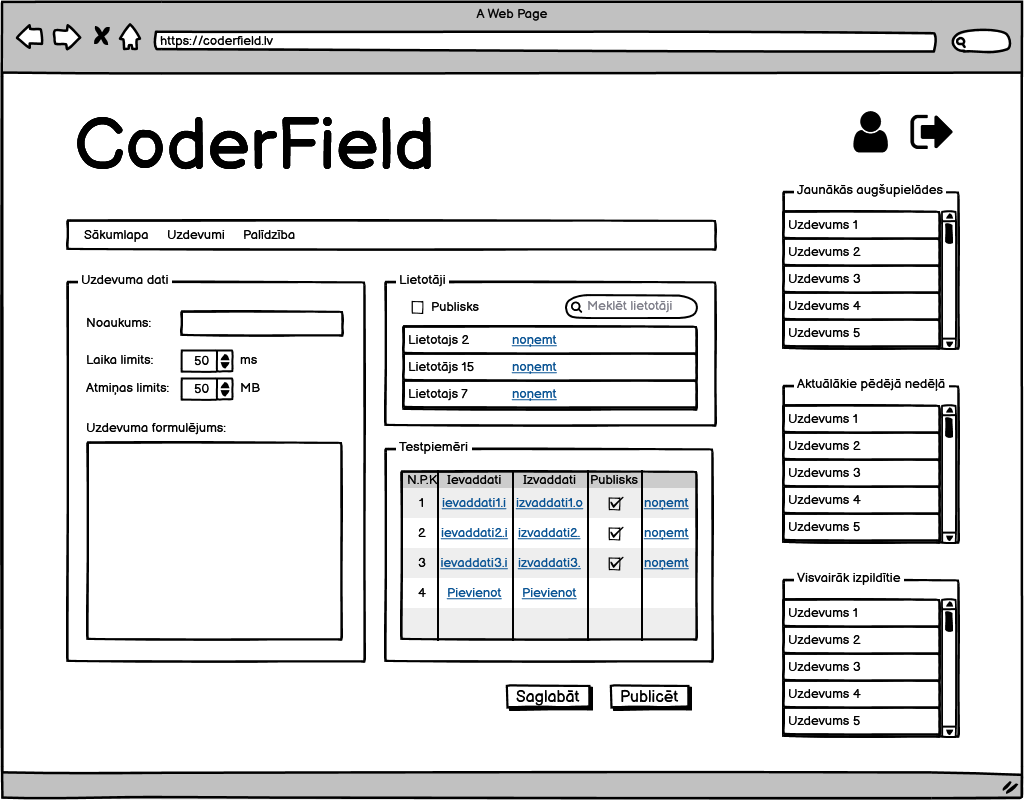
Zem Iesniegšanas lauka ir novērtēšanas lauks ar trim simboliem:

* Īkšķis uz augšu nozīmē, ka uzdevums patika;
* Īkšķis uz leju nozīmē, ka uzdevums nepatika;
* Neitrāla seja nozīmē, ka uzdevums ne patika, ne nepatika.

Ja lietotājs nav iesniedzis nevienu šī uzdevuma risinājumu, tad veicot novērtēšanu redzams kļūdas paziņojums.

Ja lietotājs ir uzdevuma autors, pie uzdevuma redzama poga “Rediģēt uzdevumu”, kas aiziet uz uzdevuma rediģēšanas skatu [3.3.4], un poga “Dzēst uzdevumu”.

### 3.3.4 Uzdevuma rediģēšana

****

*3.11. att.* **Uzdevuma rediģēšanas saskarne**

Šeit tiek attēlots, kā izskatās uzdevuma rediģēšanas lapa. Pavisam ir trīs lauki:

* Uzdevuma dati, kas sevī iekļauj
  + Nosaukumu
  + Izpildes laika limitu
  + Izpildei nepieciešamo brīvpiekļuves atmiņas limitu
  + Formulējumu
* Lietotāji, kurā norāda visus lietotājus, kas uzdevumu var redzēt. Ja lauks “Publisks” ir iespējots, tad saraksts ir atspējots un otrādi.
* Testpiemēri, kurā norāda testpiemēra atbilstošos datus:
  + Ievaddatu failu
  + Izvaddatu failu
  + Publiskumu

Ja rediģējamais uzdevums ir publisks, tad visi lauki, neskaitot lietotāju lauku, ir atspējoti (ja uzdevums ir publisks, tad to rediģēt vairs nedrīkst, drīkst tikai norādīt lietotājus, kas uzdevumu varēs apskatīt).

### 3.3.5 Risinājumu meklēšana

## 

*3.12. att.* **Risinājumu meklēšanas saskarne**

Šeit tiek attēlots, kā lietotājs var pārskatīt savus risinājumus. Lietotājs var ievadīt uzdevuma, kuru viņš meklē, nosaukumu un atrasts atbilstošā uzdevuma risinājumus. Tā kā risinājuma iesūtīšanas datuma laukā ir norādīts tikai gads, mēnesis un diena, tad ir arī lauks “Mēģinājums”, kurā ir norādīts, kurš iesūtījums tas ir. Visi risinājumi ir sakārtoti dilstoši pēc iesniegšanas datuma, kur pirmie rezultāti ir visjaunākie iesniegumi un pēdējie ir visvecākie.

### 3.3.6 Risinājuma apskate

## 

*3.13. att.* **Risinājuma apskates saskarne**

Šeit tiek attēlots, kāda informācija ir pieejama lietotājam par iesūtīto risinājumu. Vispirms ir redzams nosaukums, pēc kura seko risinājuma programmkods. Pēc noklusējuma tas ir aizklāts. Uzklikšķinot uz “Rādīt vēl”, atveras lauks, kurā ir rakstīts viss programmkods. Pēc tam seko testu rezultāti, kur ir parādīti programmas rezultāti katram testpiemēram, kurš ir piesaistīts pie uzdevuma. Var redzēt izpildes rezultātu, izpildes laiku, izpildei nepieciešamo brīvpiekļuves atmiņu un piezīmes. Piezīmēs tiek ierakstīts “Testpiemērs ir izpildīts veiksmīgi”, ja viss ir kārtībā. Ja tā nav, tad ir uzrakstīti atbilstošie paziņojumi par to, kas nogāja greizi. Testpiemēriem, kuri atzīmēti kā “redzami” un kuru izpilde notikusi veiksmīgi (iegūts pareizs vai nepareizs rezultāts), iespējams arī apskatīties iesūtītā risinājuma izdotos izvaddatus.

# IZMANTOTĀ LITERATŪRA UN AVOTI

1. Standarta LVS 68:1996 “Programmatūras prasību specifikācijas ceļvedis” vadlīnijas;
2. Standarta LVS 72:1996 “Ieteicamā prakse programmatūras projektējuma aprakstīšanai” vadlīnijas.
3. CodeWars interneta vietne. Pieejama: <https://www.codewars.com/>
4. CodeForces interneta vietne. Pieejama: <https://codeforces.com/>
5. CleverCode interneta vietne. Pieejama: [https://clevercode.lv/](https://clevercode.lv)
6. Google OAuth 2.0 dokumentācija. Pieejama: <https://developers.google.com/identity/protocols/oauth2> [Aplūkots 20.01.2021]

**Aizstāvēšanas lapa**

**Nosaukums:** Programmēšanas uzdevumu vietne “CoderField”

**Izstrādāja:** Artis Pauniņš, ap19122

Ainārs Dukaļskis, ad19068

Anete Alise Upeniece, au19025

Niks Derums, nd19009

Artis Jasinovičs, aj19090

**Projekta vadītājs:** Docents Dr.dat. Pēteris Paikens

**Aizstāvēšanas datums:** 25.01.2021.

**Komisijas vērtējums:** \_\_\_\_\_\_\_\_\_

**Komisijas sastāvs:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_