LATVIJAS UNIVERSITĀTE

DATORIKAS FAKULTĀTE

APMĀCĪBU RĪKS BĒRNIEM

KVALIFIKĀCIJAS DARBS

Autors: Lauris Ločmelis

Stud. apl. nr.: LL09290

Darba vadītājs: Dr. dat. Jānis Zuters

RĪGA 2017

ANOTĀCIJA

Apmācību rīks bērniem ir paredzēt ar mērķi attīstīt bērnu domāšanu un reakcijas ātrumu. Rīks piemērots, lai bērns patstāvīgi spētu uzlabot skolā iegūtās zināšanas un attīstīt izzinātos skaitļus un burtus. Moderno tehnoloģiju laikā ir būtiski atvieglots bērnu pašizzināšanās un pašizglītošanās process, līdz ar to, šis rīks var kalpot par lielisku veidu, kā uzlabot savas zināšanas arī ārpus skolas laika, atvieglojot turpmāko mācību procesu.

Atslēgas vārdi: rīks, matemātika, anagramma.

ABSTRACT

LEARNING TOOL FOR KIDS

Learning tool for kids is made with aim to develop thinking and reaction time for kids. This tool is made for kids to get better with their knowledge which they have gathered in school and also to get better with numbers and letters. In this modern technology era, it is very easy for the kids to ease their own process of self-learning. And this tool can serve as a great way to improve the skills also not only in school, but anywhere, by easing the further study process.

Keywords: tool, math, anagram.

SATURS

IEVADS

Kvalifikācijas darba tēmu esmu izvēlējies, jo uzskatu, ka moderno tehnoloģiju laikmets ir devis ļoti pozitīvu ietekmi uz cilvēkiem, un var kalpot ne tikai kā izklaides līdzeklis, bet gan arī kā rīks, lai uzlabotu savas zināšanas. Bērniem augot ir nepieciešams nemitīgi pilnveidoties un apgūt arvien daudz jaunas zināšanas, taču laiks ir ierobežots skolas solā.

Tāpēc manis radītais rīks atvieglos bērniem labāk izprast matemātikas un latviešu valodas priekšmetus. Šis rīks var kalpot gan iemaņu uzlabošanai, gan arī kā līdzeklis bērniem pašiem pārbaudīt savas zināšanas. Neapšaubāmi patīkami ir arī fakts, ka tiek fiksēti rezultāti, kas var rosināt bērnos veselīgu konkurenci mācīties labāk un apzinīgāk.

Tā kā matemātika un valodas ir ļoti nozīmīgas, tad izvēlējos iekļaut šo abu priekšmetu pamatelementus – burtus un ciparus uzdevumu veidošanai.

1. PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA

1.1. Ievads

1.1.1. Dokumenta nolūks

Programmatūras prasību specifikācijā ir aprakstītas prasības apmācību rīka bērniem izgatavošanai. Apmācību rīks bērniem tiks izstrādāts vadoties pēc šīs specifikācijas, kura ir paredzēta programmas izstrādātājam.

1.1.2. Darbības sfēra

Apmācību rīka mērķis ir piedāvāt iespēju bērniem attīstīt savas zināšanas latviešu valodā un matemātikā, kā arī uzlabot savas koncentrēšanās prasmes un veiklību. Programmai jābūt viegli saprotamai, jo mērķauditorija ir bērni. Ar šī rīka palīdzību bērniem būs iespēja pārbaudīt savas zināšanas un veiklību.

1.1.3. Definīcijas un saīsinājumi

varchar – simbolu virkne;

int – vesels skaitlis;

ID – identifikācijas numurs;

1.1.4. Saistība ar citiem dokumentiem

Šis dokuments ir sastādīts vadoties pēc standarta LVS 68:1996 “Programmatūras prasību specifikācijas ceļvedis”, kurā aprakstītas vadlīnijas programmatūras prasību specifikācijas izveidē.

1.1.5. Pārskats

Ievads satur dokumenta pamatinformāciju – dokumenta nolūku, darbības sfēru, definīcijas un saīsinājumus, saistību ar citiem dokumentiem, kā arī dokumenta pārskatu.

Vispārējais apraksts apraksta projekta funkcijas, lietotāja raksturiezīmes, kā arī sistēmas prasības.

Funkcionālās prasības aprasta funkciju mērķus un darbības ko tām būtu jāveic.

Ārējās saskarnes prasības apraksta lietotāja un sakaru saskarni.

Nefunkcionālās prasības iekļauj informāciju par datu drošību.

1.2 Vispārējais apraksts

1.2.1. Projekta funkcijas

1. Administratora autorizācija;
2. Anagrammu pārvaldīšana:
    a) Attēlošana;
    a) Pievienošana;
    b) Dzēšana.
3. Rezultātu pārvaldīšana:
    a) Attēlošana;
    b) Atlasīšana;
    c) Dzēšana.
4. Administratoru pārvaldīšana:
    a) Attēlošana;
    b) Atlasīšana;
    c) Dzēšana;
    d) Datu maiņa;
    e) Pievienošana.
5. Spēles uzsākšana;
6. Spēles pārtraukšana;
7. Jautājumu atbildēšana.

1.2.2. Lietotāja raksturiezīmes

Rīks, kā jau minēts nosaukumā, paredzēts lietošanai bērniem. Līdz ar to tam jābūt viegli saprotamam, ar vienkāršu saskarni. Administratora tiesības iespējams piešķirta ar jau esoša administratora pieeju.

1.2.3. Sistēmas prasības

Lietotājam ir jābūt pieejai internetam, kā arī uzinstalētai interneta pārlūkprogrammai.

1.2.4. Datubāzes prasības

Iespējai piekļūt datubāzei ir jābūt administratoram, kā arī jānodrošina iespējā vienlaicīgi izmantot datubāzes datus apstrādei vairākiem lietotājiem. Datubāzei jāatbalsta UTF-8 simbolu kodējums.

1.3. Funkcionālās prasības

1.3.1. Administratora autorizācija

Mērķis: Veikt administratora identifikāciju.

Ievade: Lietotājvārds un parole.

Apstrāde: Tiek pārbaudīts, vai administrators ar tādu lietotājvārdu eksistē un pārbauda vai parole ir pareiza.

Izvade: Veiksmīgas autorizācijas gadījumā tiek atvērts administratora panelis.

1.3.2. Anagrammu attēlošana

Mērķis: Attēlot datubāzē esošos anagrammu ierakstus

Ievade: Anagrammu pogas nospiešana izvēlnes sadaļā.

Apstrāde: Tiek izsaukts pieprasījums uz datubāzi, kur atlasa anagrammu datus.

Izvade: Tiek attēlots saraksts ar anagrammām.

1.3.3. Anagrammu pievienošana

Mērķis: Pievienot sarakstam jaunu vārdu anagrammu spēlei.

Ievade: Jāievada jaunais anagrammas vārds.

Apstrāde: Tiek papildināta datubāze ar nepieciešamo informāciju.

Izvade: Tiek attēlots atjaunots anagrammu saraksts.

1.3.4. Anagrammu dzēšana

Mērķis: Izdzēst no anagrammu saraksta tur esošu ierakstu.

Ievade: Dzēšanas pogas nospiešana pie attiecīgās anagrammas.

Apstrāde: No datubāzes tiek izdzēsts attiecīgais anagrammas ieraksts.

Izvade: Tiek attēlots atjaunots anagrammu saraksts.

1.3.6. Rezultātu attēlošana

Mērķis: Rezultātu attēlošana pēc attiecīgā spēles tipa un grūtības pakāpes.

Ievade: Rezultātu pogas nospiešana izvēlnes sadaļā.

Apstrāde: Tiek izsaukts pieprasījums uz datubāzi, kur atlasa rezultātu datus.

Izvade: Tiek attēlots saraksts ar rezultātiem.

1.3.6. Rezultātu atlasīšana

Mērķis: Konkrēta rezultāta, detalizēta apraksta atlasīšana.

Ievade: Atvēršanas pogas nospiešana pie vēlamā rezultāta no rezultātu saraksta.

Apstrāde: Tiek izsaukts pieprasījums uz datubāzi, kur atlasa konkrētā rezultāta datus.

Izvade: Tiek attēlots konkrētā rezultāta detalizēts saraksts.

1.3.7. Rezultātu dzēšana

Mērķis: Izdzēst no rezultātu saraksta tur esošu ierakstu.

Ievade: Dzēšanas pogas nospiešana pie attiecīgi atlasītā rezultāta.

Apstrāde: No datubāzes tiek izdzēsts attiecīgais rezultāta ieraksts.

Izvade: Tiek attēlots atjaunots rezultātu saraksts.

1.3.8. Administratoru attēlošana

Mērķis: Administratoru saraksta attēlošana.

Ievade: Lietotāju pogas nospiešana izvēlnes sadaļā.

Apstrāde: Tiek izsaukts pieprasījums uz datubāzi, kur atlasa administratoru datus.

Izvade: Tiek attēlots saraksts ar administratoriem.

1.3.9. Administratoru atlasīšana

Mērķis: Konkrēta administratora, detalizēta apraksta atlasīšana.

Ievade: Atvēršanas pogas nospiešana pie vēlamā administratora no administratoru saraksta.

Apstrāde: Tiek izsaukts pieprasījums uz datubāzi, kur atlasa konkrētā administratora datus.

Izvade: Tiek attēlots konkrētā administratora detalizēts saraksts.

1.3.10. Administratoru dzēšana

Mērķis: Izdzēst no administratoru saraksta tur esošu ierakstu.

Ievade: Dzēšanas pogas nospiešana pie attiecīgi atlasītā administratora.

Apstrāde: No datubāzes tiek izdzēsts attiecīgais administratora ieraksts.

Izvade: Tiek attēlots atjaunots administratoru saraksts.

1.3.11. Administratoru paroles maiņa

Mērķis: Veikt konkrēti atlasīta administratora datu nomaiņu.

Ievade: Lietotājvārds, vārds, parole un atkārtota parole.

Apstrāde: Tiek pārbaudīts, vai administrators ar tādu lietotājvārdu eksistē un pārbauda vai paroles sakrīt. Atjauno administratoru sarakstu datubāzē.

Izvade: Tiek paziņots par veiksmīgu pievienošanu, un attēlots konkrētais atlasītais administrators.

1.3.12. Administratoru pievienošana

Mērķis: Pievienot sistēmā jaunu administratoru.

Ievade: Lietotājvārds, vārds, parole, un atkārtota parole.

Apstrāde: Tiek pārbaudīts, vai administrators ar tādu lietotājvārdu eksistē un pārbauda vai paroles sakrīt. Pievieno administratoru sarakstam datubāzē jaunu ierakstu.

Izvade: Tiek paziņots par veiksmīgu pievienošanu, un attēlots konkrētais atlasītais administrators.

1.3.13. Spēles uzsākšana

Mērķis: Uzsākt attiecīgi izvēlēto spēli no saraksta.

Ievade: Lietotājvārds, izvēlēta attiecīgā spēle un grūtības līmenis, un pogas „sākt spēli” nospiešana.

Apstrāde: Tiek atlasīti jautājumi pēc konkrētajiem izvēlētajiem parametriem

Izvade: Tiek attēlots pirmā jautājuma atbildēšanas skats.

1.3.14. Spēles pārtraukšana

Mērķis: Pārtraukt uzsākto spēli vēl pirms spēles laika beigām.

Ievade: Pārtraukšanas pogas nospiešana izvēlnes sadaļā.

Apstrāde: Tiek pārtraukta jautājumu atbildēšana.

Izvade: Tiek atvērta sākuma izvēlne.

1.3.15. Jautājumu atbildēšana

Mērķis: Veikt konkrētu jautājumu atbildēšanu.

Ievade: a) anagrammas atbildes vārda ievadīšana anagrammu gadījumā;

b) pareizās pogas (viena no trim) nospiešana salīdzināšanas gadījumā;

c) pareizā secībā pogu nospiešana kārtošanas gadījumā;

Apstrāde: Tiek pārbaudīts vai sniegtā atbilde ir korekta.

Izvade: Tiek pievienoti punkti pie rezultāta.

1.4 Ārējās saskarnes prasības

1.4.1. Lietotāja saskarne

Lietotāja saskarne tiek veidota vienkārša, lai būtu viegli saprast nepieciešamās darbības, kuras ir jāveic, jo mērķauditorija ir bērni.

1.4.2. Sakaru saskarne

Paredzēta darboties uz HTTPS protokola.

1.5 Nefunkcionālās prasības

1.5.1. Datu drošība

Paroles tiek glabātas šifrētas hash šifrā. Spēle, spēlētāja vārds un vecums tiek pieglabāti *cookies*. Vēlams lietot HTTPS.

1.6 Papildus prasības

Nepieciešams nodrošināt rīkam 3 dažādu grūtības līmeņu jautājumus atkarībā no vecuma grupas katrai spēlei.

Anagrammas:

Līmenis	Apraksts
1. Līmenis (viegls)	Anagrammā iekļautā vārda garums ne garāks par 5 burtiem.
2. Līmenis (vidējs)	Anagrammā iekļautā vārda garums ne garāks par 8 burtiem.
3. Līmenis (sarežģīts)	Anagrammā iekļautā vārda garums neierobežots.

Salīdzināšana:

Līmenis	Apraksts
1. Līmenis (viegls)	Tiek salīdzināti tikai veseli skaitļi intervālā [0..20].
2. Līmenis (vidējs)	Tiek salīdzināti veseli skaitļi, jeb izteiksmes ar saskaitīšanas un atņemšanas darbībām intervālā [-50..50].
3. Līmenis (sarežģīts)	Tiek salīdzināti veseli skaitļi, jeb izteiksmes ar saskaitīšanas, atņemšanas, reizināšanas un dalīšanas darbībām intervālā [-100..100].

Kārtošana:

Līmenis	Apraksts
1. Līmenis (viegls)	Tiek kārtoti secībā tikai veseli skaitļi intervālā [0..20].
2. Līmenis (vidējs)	Tiek kārtoti secībā veseli skaitļi, jeb izteiksmes ar saskaitīšanas un atņemšanas darbībām intervālā [-50..50].
3. Līmenis (sarežģīts)	Tiek kārtoti secībā veseli skaitļi, jeb izteiksmes ar saskaitīšanas, atņemšanas, reizināšanas un dalīšanas darbībām intervālā [-100..100].

2. PROGRAMMATŪRAS PROJEKTĒJUMA APRAKSTS

2.1. Ievads

2.1.1. Dokumenta nolūks

Programmatūras projektējuma apraksta nolūks ir aprakstīt, kā pēc programmatūras prasību specifikācijas tiks veidots attiecīgais rīks, lai atvieglotu programmatūras izstrādi tā veikšanas brīdī

2.1.2. Darbības sfēra

Apmācību rīks ir paredzēts bērniem, lai varētu attīstīt savas zināšanas latviešu valodā un matemātikā, kā arī uzlabot savas koncentrēšanās prasmes un veiklību. Rīks paredzēts brīvai lietošanai, kā arī ar iespējām veikt uzlabojumus (administratora darbības).

2.1.3. Definīcijas un saīsinājumi

varchar – simbolu virkne;

int – vesels skaitlis;

ID – identifikācijas numurs;

2.1.4. Saistības ar citiem dokumentiem

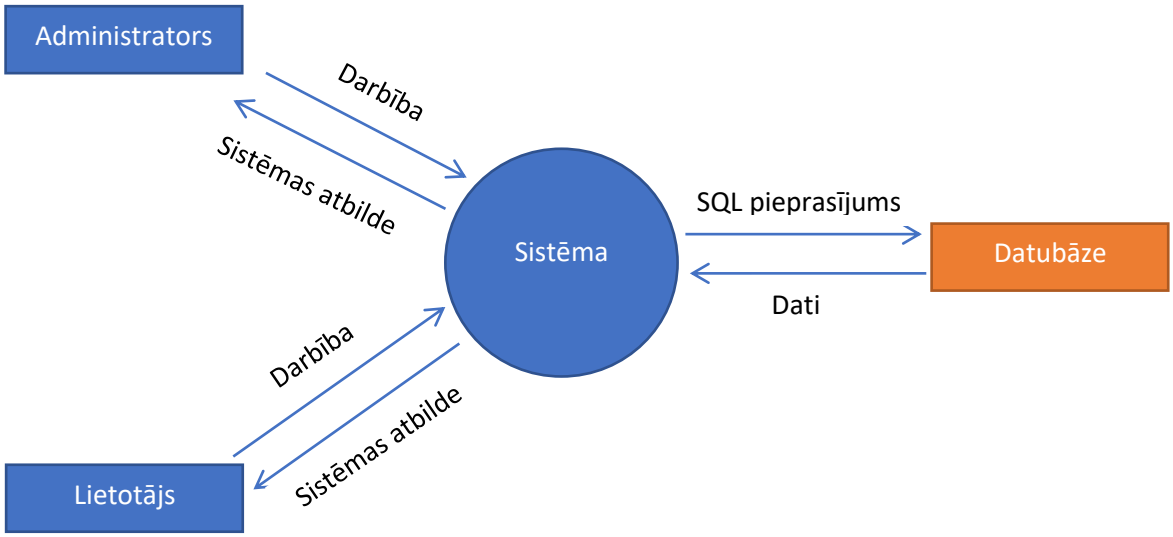
Šis dokuments ir sastādīts vadoties pēc standarta LVS 72:1996 “Ieteicamā prakse programmatūras projektējuma aprakstīšanai”, kurš apraksta vadlīnijas programmatūras projektējuma apraksta izveidē.

2.2. Moduļu dekompozīcija

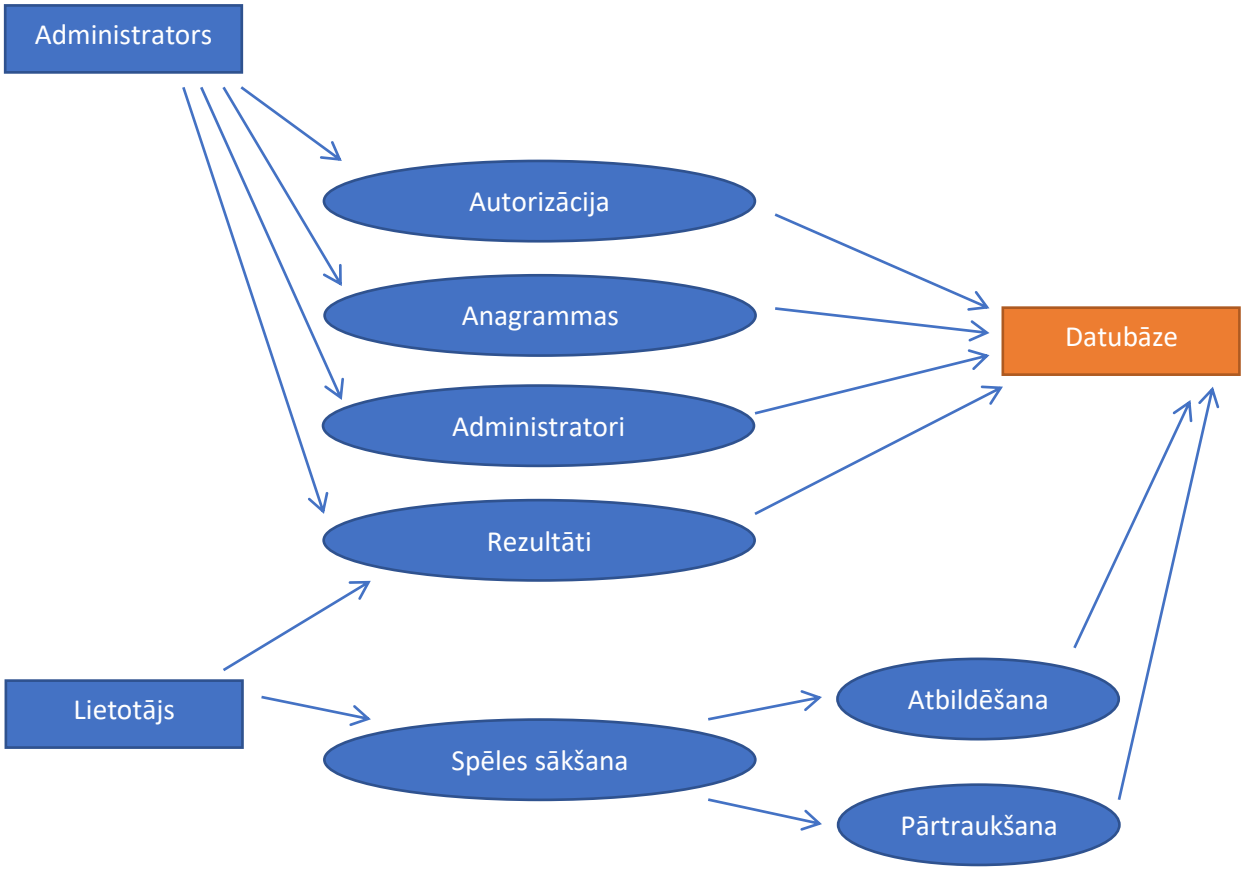
Modulis	**Funkcijas**	**Izmantotās DB tabulas**
Administratoru pārvaldīšana	Autorizācija, paroles maiņa, administratoru pievienošana/dzēšana	*admins*
Rezultātu pārvaldīšana	Rezultātu attēlošana, dzēšana	*answers, users, scores*
Anagrammu pārvaldīšana	Anagrammu pievienošana, dzēšana	*anagrams*
Spēles uzsākšana	Rezultāta skaitīšanas uzsākšana, lietotāja datu iegūšana	*users, scores*
Spēles pārtraukšana	Rezultāta fiksēšana	*scores*
Jautājumi	Jautājumu atbildēšana	*scores, answers, anagrams*

2.3. Starpprocesu atkarības (datu plūsmu diagrammas)

2.3.1. Datu plūsmu diagrammas 0.līmenis



2.3.1. Datu plūsmu diagrammas 1.līmenis



2.4. Datu bāzes detalizēts apraksts

2.4.1. Datubāzes tabulu detalizēts apraksts

Tabula *admins*						
Nosaukums	Datu tips	Garums	Primārā atslēga	NULL	Unikāla vērtība	Skaidrojums
id	int	6	Jā	Nē	Jā	Unikāls identifikators
username	varchar	32	Nē	Nē	Jā	Unikāls lietotājvārds
password	varchar	128	Nē	Nē	Nē	Parole šifrētā veidā
salt	varchar	10	Nē	Nē	Nē	Paroles šifrēšanas papildlauks
name	varchar	50	Nē	Nē	Nē	Vārds, uzvārds

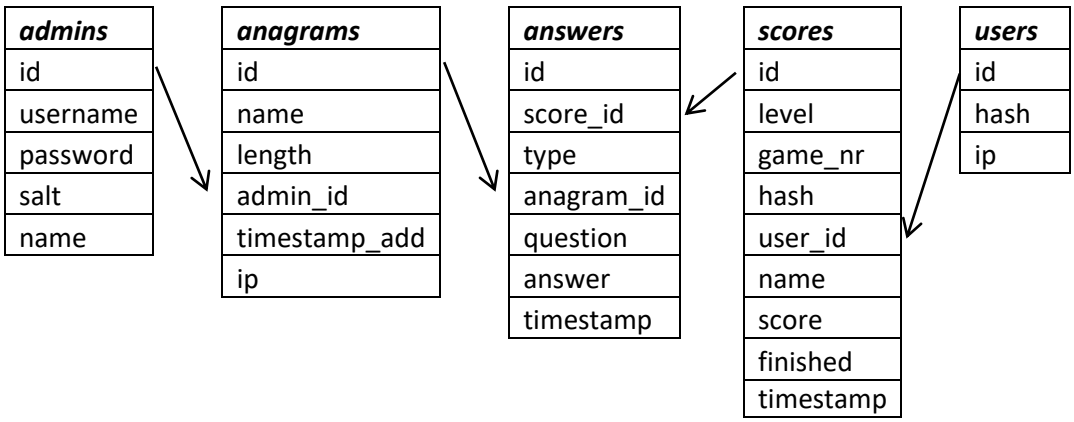
Tabula *anagrams*						
Nosaukums	Datu tips	Garums	Primārā atslēga	NULL	Unikāla vērtība	Skaidrojums
id	int	6	Jā	Nē	Jā	Unikāls identifikators
name	varchar	50	Nē	Nē	Nē	Anagrammas vārds
length	int	3	Nē	Nē	Nē	Vārda garums
admin_id	int	6	Nē	Nē	Nē	Administratora identifikators
timestamp_add	int	10	Nē	Nē	Nē	Pievienošanas laiks
ip	varchar	15	Nē	Nē	Nē	IP adrese

Tabula *users*						
Nosaukums	Datu tips	Garums	Primārā atslēga	NULL	Unikāla vērtība	Skaidrojums
id	int	6	Jā	Nē	Jā	Unikāls identifikators
hash	varchar	32	Nē	Nē	Nē	Lietotāja šifrējums
ip	varchar	15	Nē	Nē	Nē	IP adrese

Tabula *scores*						
Nosaukums	Datu tips	Garums	Primārā atslēga	NULL	Unikāla vērtība	Skaidrojums
id	int	6	Jā	Nē	Jā	Unikāls identifikators
level	int	3	Nē	Nē	Nē	Grūtības pakāpe
game_nr	int	1	Nē	Nē	Nē	Spēles veids
hash	varchar	32	Nē	Nē	Nē	Lietotāja šifrējums
user_id	int	6	Nē	Nē	Nē	Lietotāja identifikators
name	varchar	50	Nē	Nē	Nē	Lietotāja vārds
score	int	6	Nē	Nē	Nē	Rezultāts
finished	enum	(false, true)	Nē	Nē	Nē	Pabeigtas spēles noteikšanas lauks
timestamp	int	10	Nē	Nē	Nē	Laiks

Tabula *answers*						
Nosaukums	Datu tips	Garums	Primārā atslēga	NULL	Unikāla vērtība	Skaidrojums
id	int	6	Jā	Nē	Jā	Unikāls identifikators
score_id	int	6	Nē	Nē	Nē	Rezultāta identifikators
type	int	1	Nē	Nē	Nē	Spēles tips
anagram_id	int	6	Nē	Nē	Nē	Anagrammas identifikators
question	varchar	200	Nē	Nē	Nē	Jautājums
answer	varchar	200	Nē	Nē	Nē	Atbilde
timestamp	int	10	Nē	Nē	Nē	Laiks

2.4.1. Datubāzes ER modelis



3. TESTĒŠANAS DOKUMENTĀCIJA

Projekta izstrādes beigu posmā tika veikta testēšana funkcijām, kuru izpildīšanā tika nepieciešams ievadīt ievaddatus.

Administratora autorizācija:

Ievaddati	Paredzētā darbība	Rezultāts
Tiek atstāts tukšs ievadlauks	Tiek atkārtoti ielādēts autorizācijas logs.	Izpildās
Tiek ievadīta nepareiza informācija	Tiek atkārtoti ielādēts autorizācijas logs.	Izpildās
Tiek ievadīti pareizi autorizācijas dati	Tiek ielādēts administratora panelis.	Izpildās

Anagrammu pievienošana:

Ievaddati	Paredzētā darbība	Rezultāts
Tiek atstāts tukšs ievadlauks	Tiek atkārtoti ielādēts anagrammu saraksts bez izmaiņām.	Izpildās
Tiek ievadīta skaitļu virkne	Tiek atkārtoti ielādēts anagrammu saraksts ar papildināto ierakstu.	Izpildās

Administratora pievienošana un labošana:

Ievaddati	Paredzētā darbība	Rezultāts
Tiek atstāts tukšs ievadlauks	Tiek paziņots kļūdas paziņojums par neaizpildītajiem laukiem.	Izpildās
Tiek ievadīta eksistējoša administratora lietotājvārds	Tiek paziņots kļūdas paziņojums par jau eksistējošu lietotājvārdu.	Izpildās

Tiek ievadītas paroles, kuras nesakrīt	Tiek paziņots kļūdas paziņojums par jau eksistējošu lietotājvārdu.	Izpildās
Ievadīta parole, kas īsāka par 6 simboliem, vai garāka par 20 simboliem	Tiek paziņots kļūdas paziņojums par nekorektu paroli.	Izpildās

Jautājumu (anagrammu) atbildēšana:

Ievaddati	Paredzētā darbība	Rezultāts
Tiek atstāts tukšs ievadlauks	Tiek paziņots par kļūdu un samazinās spēles laiks.	Izpildās
Tiek ievadīta pareiza atbilde	Tiek ielādēta jauna anagramma.	Izpildās
Tiek ievadīta nepareiza atbilde	Tiek paziņots par kļūdu un samazinās spēles laiks.	Izpildās

Jautājumu (salīdzināšana) atbildēšana:

Ievaddati	Paredzētā darbība	Rezultāts
Tiek ievadīta pareiza atbilde (nospiesta pareizā poga)	Tiek ielādēts jauns salīdzināšanas jautājums.	Izpildās
Tiek ievadīta nepareiza atbilde (nospiesta nepareizā poga)	Tiek paziņots par kļūdu un samazinās spēles laiks.	Izpildās

Jautājumu (kārtošana) atbildēšana:

Ievaddati	Paredzētā darbība	Rezultāts
Tiek ievadīta pareiza atbilde (pareiza secība)	Poga iekrāsojas zaļa, un visu pareizu atbilžu gadījumā tiek ielādēts jauns kārtošanas jautājums.	Izpildās
Tiek ievadīta nepareiza atbilde (nepareiza secība)	Tiek paziņots par kļūdu un samazinās spēles laiks	Izpildās

4. PROJEKTA ORGANIZĀCIJA

Projekta izstrādes sākumposmā tika izlemta projekta tēma un, atsaucoties uz to, tika meklēta informācija. Nācās izpētīt aptuveno bērnu zināšanu līmeni pēc attiecīgajiem vecumposmiem, lai varētu izprast uzdevumu sarežģītības pakāpi.

Tika izstrādāta programmatūras prasību specifikācija un programmatūras projektējuma apraksts ar pēc iespējas precīzāk detalizētu vēlamo rezultātu.

Tika veidots programmatūras kods, paralēli sekojot līdzi padarītajam, kā arī nepieciešamības gadījumā tika precīzāk nodefinēta programmatūras prasību specifikācija.

Pēc rīka izstrādes, tas tika testēts uz ievaddatiem, lai novērstu projekta izstrādes laikā radušās kļūdas un neprecizitātes.

Beigu posmā tika izdarīti secinājumi un novērtēti darba rezultāti.

5. KVALITĀTES NODROŠINĀJUMS

Kvalitātes nodrošināšanas nolūkos sākotnēji tika izstrādāta programmatūras prasību specifikācija un programmatūras projektējuma apraksts. Šie dokumenti tika veidoti saskaņā ar LVS standartiem LVS 68:1996 „Programmatūras prasību specifikācijas ceļvedis" un LVS 72:1996 „Ieteicama prakse programmatūras projektējuma aprakstīšanai".

Kods ir rakstīts saskaņā ar labu programmēšanas praksi, veicot tajā skaidri saprotamus komentārus.

Lai nodrošinātu pēc iespējas augstāku kvalitāti programmas beigu posmā tika veikta testēšana, lai pārliecinātos par iespējamajām pieļautajām kļūdām. Kā arī kļūdu gadījumā tika veikta kļūdu labošana un atkārtota testēšana.

6. DARBIETILPĪBAS NOVĒRTĒJUMS

Darbietilpības aprēķināšanai tika izmantots vienkāršs tiešsaistē pieejams COCOMO 81 kalkulators, un aprēķinam izmantoti „Organic" (Relatīvi maziem projektiem ar mazu darbaspēka atbalstu un pietiekami labām zināšanām) koeficienti, kas uzskaitīti zemāk:

	a_b	b_b	c_b	d_b
Organic	2,4	1,05	2,5	0,38

Aptuvenais koda rindiņu skaits projektā: 1800, jeb 1.8KLOC (rindiņu tūkstoši)

E – nepieciešamais laiks (personmēnešos);

D – izstrādes laiks (mēneši);

E= a_b (KLOC)^ b_b =2,4*1,8^1,05=4,45 personmēneši;

D= c_b (E)^ d_b=2,5x4,45^0,38=4,41 mēneši.

7. REZULTĀTI UN SECINĀJUMI

Izveidotais rīks atbilst sākumā izvirzītajam mērķim un var kalpot par palīgu bērniem savu iemaņu attīstīšanai ārpus skolas. Uzskatu, ka ar izveidotā rīka palīdzību ir iespējams veikt sākotnēji pieprasītās funkcijas un pārvaldīt datus.

Secinu, ka ērtākais veids kā veikt darba izstrādi šādam rīkam būtu pēc Agile principa, jo, manuprāt līdz galam precīzi nodefinēt nepieciešamās funkcijas ir ļoti grūti, tāpēc atviegloti ir, ja klientam ir iespēja redzēt paveikto atsevišķos laika posmos.

Varu secināt, ka esmu izvēlējies savām prasmēm atbilstošu projektu, kurš nav īpaši sarežģīts, tāpat arī varu secināt, ka rīku ir iespējams uzlabot vēl vairāk izprotot bērnu attīstību pirmsskolas un sākumskolas vecumposmā, lai varētu precīzāk nodefinēt spēļu sarežģītību.

8. PROGRAMMATŪRAS KODS

8.1. Admin.php kontrolieris

```
class Controller_Admin extends Controller_Template {

	public $template = 'template/admin';

	public function before() {
		parent::before();
	}
	/*** Pārbauda vai ir ielogojies ***/
	private function checkLogin() { 65
		if( Session::instance()->get('logged_in',false) == false ) {
HTTP::redirect('admin'); };
	}
	/*** Autorizācijas logs ***/
	public function action_login()
	{
		// Pārbauda vai nav ielogojies
		if( Session::instance()->get('logged_in',false) == true ) {
HTTP::redirect('admin/anagram'); };

		if(!empty($_POST) && isset($_POST['inputUser']) && $_POST['inputUser']
!= "" && isset($_POST['inputPassword']) && $_POST['inputPassword'] != "") {

			// Meklējam lietotāju
			$user = new Model_ORM_Admin;
			$user->where('username', '=', strtolower($_POST['inputUser']))-
>find();

			// Chekojam paroli
			if($user->loaded() && $user->login($_POST['inputPassword']) ==
true) {
				// Veiksmīga autorizācija
```

```
                        HTTP::redirect('admin/anagram');
                    };
                    // Kļūda autorizējoties
                    $this->template->failed = true;
                }
                $this->template->login = true;
            }

            /*** Anagrammu saraksts ***/
            public function action_anagram()
            {
                $this->checkLogin();

                $center = View::factory('admin/anagram');

                    $aModel = Model::factory('Game');

                    $center->anagrams = $aModel->getAnagrams();

                    if(isset($_POST['anagramName']) && trim($_POST['anagramName'])
!= '') {
                        $aModel->addAnagram($_POST['anagramName']);
                        HTTP::redirect('admin/anagram');
                    }
                    else if(isset($_GET['delete']) &&  (int)$_GET['delete'] > 0) {
                        $aModel->removeAnagram($_GET['delete']);
                        HTTP::redirect('admin/anagram');
                    }
                $this->template->active = 'anagram';
                $this->template->content = $center->render();
            }
            /*** Highscore saraksts ***/
            public function action_highscore()
```

```
{
	$this->checkLogin();
	$center = View::factory('admin/highscore/list');
		$aModel = Model::factory('Game');
		$level = $this->request->param('level');
		$game = $this->request->param('game');

		if((int)$level == 0) { $level = 1; }
		if((int)$game == 0) { $game = 1; }

		$center->level = $level;
		$center->game = $game;
		$center->highscores = $aModel->getHighscores($level, $game, true);

		if(isset($_GET['delete']) &&  (int)$_GET['delete'] > 0) {
			$aModel->removeHighscore($_GET['delete']);
			HTTP::redirect('admin/highscore/' . $level . '/' . $game);
		}

	$this->template->active = 'highscore';
	$this->template->content = $center->render();
}
/*** Highscore saraksts ***/
public function action_highscore_details()
{
	$this->checkLogin();
		$center = View::factory('admin/highscore/details');

		$aModel = Model::factory('Game');

		$id = $this->request->param('id');

		$center->details = $aModel->getHighscoreDetails($id);
```

```
                if(isset($_GET['delete']) &&  (int)$_GET['delete'] > 0) {
                        $aModel->removeHighscore($id);
                        HTTP::redirect('admin/highscore');
                }
        $this->template->active = 'highscore';
        $this->template->content = $center->render();
}
        /*** Administratoru saraksts ***/
        public function action_administrators()
        {
                $this->checkLogin();

                $center = View::factory('admin/administrators/list');

                        // Meklējam lietotāju
                        $admins = new Model_ORM_Admin;
                        $center->administrators = $admins->find_all();

                $this->template->active = 'administrators';
                $this->template->content = $center->render();
        }
        /*** Administratoru forma ***/
        public function action_administrators_form()
        {
                $this->checkLogin();

                $center = View::factory('admin/administrators/form');

                        $openID = $this->request->param('id');

                        $userModel = new Model_ORM_Admin;

                        // Ielādē uz labošanu
```

```
if($openID != 'add' && (int)$openID > 0) {

    $userModel->where('id','=',(int)$openID)->find();

    // Lietotāju neatradu
    if(!isset($userModel->id) || $userModel->id == 0) {
HTTP::redirect('admin/administrators'); };
    $center->userData = $userModel;
};
if(isset($_POST['save'])) {
    // Validation

    $post = Validation::factory($_POST)

        ->rule('userUsername', 'not_empty')
        ->rule('userUsername', array($userModel,
'unique_username'), array(':value', (int)$openID));

    if($openID == 'add') {
        $post
            ->rule('userPassword', 'not_empty')
            ->rule('userConfirm', 'not_empty');
    };

    $post
        ->rule('userPassword', 'min_length',
array(':value', 6))
        ->rule('userPassword', 'max_length',
array(':value', 20))
        ->rule('userConfirm', 'matches',
array(':validation', ':field', 'userPassword'))
        ->rule('userName', 'not_empty');

    if($post->check()) {
        $userModel->username = $post['userUsername'];
```

```
                    if(isset($post['userPassword']) &&
$post['userPassword'] != "") {
                        $userModel->salt = $userModel->random();
                        $userModel->password = $userModel-
>hash($post['userPassword'], $userModel->salt);
                    }

                    $userModel->name = $post['userName'];
                    $userModel->save();

                    Session::instance()->set('success', true);
                    HTTP::redirect('admin/administrators/' . $userModel-
>id);
                }
                else {
                    $center->post = $_POST;
                $errors = array(
                'userUsername_not_empty'=>'Lietotājvārds nav aizpildīts',
                'userUsername_unique_username'=>'Tāds lietotājvārds jau
pastāv',
                'userPassword_not_empty'=>'Parole nav aizpildīta',
                'userConfirm_not_empty'=>'Atk. parole nav aizpildīts',
                'userPassword_min_length'=>'Parolei jābūt vismaz 6 simbolus
garai',
                'userPassword_max_length'=>'Parolei īsākai par 20 simboliem',
                'userConfirm_matches'=>'Parole nesakrīt',
                'userName_not_empty'=>'Vārds nav aizpildīts',
                    );

                    foreach($post->errors() as $k=>$v) {
                        $center->error = $errors[$k.'_'.$v[0]];
                        break;
                    };
                };
```

```
                }
                if(isset($_POST['delete']) &&  (int)$openID > 0) {
                        $userModel->delete();
                        HTTP::redirect('admin/administrators');
                }


        if(Session::instance()->get('success', false) == true) {
                $center->success = true;
                Session::instance()->delete('success');
        };

        $center->my_id = Session::instance()->get('user_id');
        $this->template->active = 'administrators';
        $this->template->content = $center->render();
    }
    /*** Izlogošanās ***/
    public function action_logout()
    {
        Session::instance()->destroy();
        HTTP::redirect('admin');
    }
}
```

8.1. Game.php modelis

```
class Model_Game {

	public function getQuestion($game_id, $type, $level) {

		if($type == 4) { $type = rand(1,3); }

		$data = array('type'=>$type,'level'=>$level);

		$anagram_id = NULL;

		if($type == 1) {
			// Anagrammas
			$types = array(
				1=>array('min'=>1,'max'=>5),
				2=>array('min'=>1,'max'=>8),
				3=>array('min'=>1,'max'=>100),
			);

			$thisType = $types[$level];

			$rec = DB::select('id','name')
					->from('anagrams')
					->where('length','>=',$thisType['min'])
					->and_where('length','<=',$thisType['max'])
					->order_by(DB::expr('rand()'))
					->execute()->current();

			$data['answer'] = $rec['name'];
			$data['parts'] = array();

			for($n=0;$n<mb_strlen($rec['name']);$n++) {
```

```
                $data['parts'][] = mb_substr($rec['name'],$n,1);
            };

            $shuffled = $data['parts'];

            do { shuffle($shuffled); }
            while (implode('', $shuffled) == $rec['name']);

            $data['shuffled'] = implode(' ', $shuffled);

            $anagram_id = $rec['id'];
            $question = $data['shuffled'];
            $answer = $data['answer'];

        }
        else if($type == 2) {
            // Salīdzināšana
            $first = $this->getNumber($level);
            $second = $this->getNumber($level);
            $oper = ($first['value'] > $second['value']) ? '>' : (($first['value'] <
$second['value']) ? '<' : '=');

            $data['record'] = array('first'=>$first,'second'=>$second,'oper'=>$oper);
            $question = $first['number'] . ' un ' . $second['number'];
            $answer = $oper;

        }
        else if($type == 3) {
            // Kārtošana
            $question = '';
            $answer = '';

            $record = array();
```

```
                for($n=1;$n<=5;$n++) {
                        do { $number = $this->getNumber($level); }
                        while (isset($record[$number['value']]));

                        $record[$number['value']] = $number;
                        $question .= $number['number'] . '; ';
                };

                $keys = array_keys($record);
                asort($keys);
                $n=1;
                foreach($keys as $k) {
                        $answer .= $record[$k]['number'] . '; ';
                        $record[$k]['place'] = $n++;
                }

                $data['record'] = array_values($record);
        }

        DB::insert('answers',array('score_id','type','anagram_id','question','answer',
'timestamp'))
                ->values(array($game_id, $type, $anagram_id, $question, $answer,
time()))->execute();

        return $data;
    }

    private function getNumber($level) {

        $types = array(
                1=>array('min'=>0,'max'=>20,'difficulty'=>1),
                2=>array('min'=>-50,'max'=>50,'difficulty'=>3),
                3=>array('min'=>-100,'max'=>100,'difficulty'=>5),
        );
```

```
		$thisType = $types[$level];
		$thisOperator = rand(1,$thisType['difficulty']);

		if($thisOperator == 1) {
			// Skaitlis
			$thisValue = rand($thisType['min'],$thisType['max']);
		}
		else if($thisOperator == 2) {
			// Saskaitīšana
			$thisValue = rand($thisType['min'],$thisType['max']);

			$first = rand($thisType['min'],$thisType['max']-1);
			$second = $thisValue - $first;
		}
		else if($thisOperator == 3) {
			// Atņemšana
			$thisValue = rand($thisType['min'],$thisType['max']);

			$second = rand($thisType['min'],$thisType['max']);
			$first = $thisValue + $second;
		}
		else if($thisOperator == 4) {
			// Reizināšana
			$first = rand($thisType['min'],$thisType['max']);
			$maxSecond = abs($thisType['min']) > abs($thisType['max']) ?
abs($thisType['min']) : abs($thisType['max']);
			$maxSecond = floor($maxSecond / $first);
			$second = rand(((-1) * $maxSecond), $maxSecond);

			$thisValue = $first * $second;
		}
		else if($thisOperator == 5) {
```

```
                // Dalīšana
                $thisValue = rand($thisType['min'],$thisType['max']);
                $second = rand($thisType['min'],$thisType['max']);
                $first = $thisValue * $second;
            }

            $operators = array(1=>'number',2=>'+',3=>'-',4=>'*',5=>'/');
            return array('value'=>$thisValue,'number'=>(($thisOperator == 1) ? $thisValue
: $first . ' '. $operators[$thisOperator] . ' ' . $second));

        }
        /*** Pievieno anagrammas ierakstu ***/
        public function addPlayer($hash) {
            DB::insert('users',array('hash','ip'))
            ->values(array($hash, $_SERVER['REMOTE_ADDR']))->execute();
        }
        public function startGame($time, $hash, $name, $level, $game) {
            $player = Cookie::get('player_id', 0);
            $row_id =
DB::insert('scores',array('level','game_nr','hash','user_id','name','score','finished','timestamp'))
            ->values(array($level, $game, $hash, $player, $name, 0, 'false', $time))-
>execute();

            // Atgriež spēles ID
            return $row_id[0];
        }
        /*** Iegūst spēles tipu ***/
        public function getGameData($id, $hash) {
            return DB::select('id','game_nr','level')
                                ->from('scores')
                                ->where('id','=',$id)
                                ->and_where('hash','=',$hash)
                                ->and_where('finished','=','false')
                                ->execute()->current();
```

```
	}
	/*** Finišē spēli ***/
	public function finishGame($id, $hash) {
		DB::update('scores')->set(array('finished'=>'true'))->where('id','=',$id)->and_where('hash','=',$hash)->execute();
	}
	/*** Iegūst highscores tabulu ***/
	public function getHighscores($level,$game) {
		return DB::select('id','name','score','timestamp','user_id')
					->from('scores')
					->where('level','=',$level)
					->and_where('game_nr','=',$game)
					->and_where('finished','=','true')
					->execute()->as_array();
	}
	/*** Iegūst highscores detalizāciju ***/
	public function getHighscoreDetails($id) {
		return DB::select('id','type','question','answer','timestamp')
					->from('answers')
					->where('score_id','=',$id)
					->execute()->as_array();
	}
	/*** Pievieno punktus highscore ***/
	public function addHighscore($id, $points) {
		DB::update('scores')->set(array('score'=>DB::expr(`score` + ' . $points)))->where('id','=',$id)->execute();
	}
	/*** Dzēš highscores ierakstu ***/
	public function removeHighscore($id) {
		DB::delete('scores')->where('id','=',$id)->execute();
		DB::delete('answers')->where('score_id','=',$id)->execute();
	}
	/*** Iegūst anagrammu tabulu ***/
	public function getAnagrams() {
```

```
            return DB::select('id','name','length')
                                    ->from('anagrams')
                                    ->order_by('name','ASC')
                                    ->execute()->as_array();
        }
        /*** Pievieno anagrammas ierakstu ***/
        public function addAnagram($name) {
            $admin_id = Session::instance()->get('user_id', 0);
            DB::insert('anagrams',array('name','length','admin_id','timestamp_add','ip'))
            ->values(array(mb_strtoupper($name), mb_strlen($name), $admin_id, time(),
$_SERVER['REMOTE_ADDR']))->execute();
        }
        /*** Dzēš anagrammas ierakstu ***/
        public function removeAnagram($id) {
            DB::delete('anagrams')->where('id','=',$id)->execute();
        }
}
```

IZMANTOTĀ LITERATŪRA

PHP funkciju aprakstu krājums [tiešsaiste] – **https://php.net/**

Info par anagrammām [tiešsaiste] - **https://lv.wikipedia.org/wiki/Anagramma**

Informācija par COCOMO [tiešsaiste] - **https://en.wikipedia.org/wiki/COCOMO**

COCOMO aprēķins un informācija [tiešsaiste] - **http://groups.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/baker/cocomo.html**

Bootstrap 4. versijas dokumentācija [tiešsaiste] - **https://v4-alpha.getbootstrap.com/**

LVS 68:1996 „Programmatūras prasību specifikācijas ceļvedis”

LVS 72:1996 „Ieteicama prakse programmatūras projektējuma aprakstīšanai”

DOKUMENTĀRĀ LAPA

Kvalifikācijas darbs „***Apmācību rīks bērniem***” izstrādāts Latvijas Universitātes Datorikas fakultātē.

Ar savu parakstu apliecinu, ka darbs izstrādāts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: ***Lauris Ločmelis*****_____ ___.05.2017**.

Rekomendēju darbu aizstāvēšanai

Darba vadītājs: ***Dr.dat Jānis Zuters*** **_____ ___.05.2017.**

Recenzents: ***M.dat. Dace Gobleja***

Darbs iesniegts 29.05.2017.

Kvalifikācijas darbu pārbaudījumu komisijas sekretārs: *Darja Solodovņikova*_____

Darbs aizstāvēts kvalifikācijas darbu pārbaudījuma komisijas sēdē

___.06.2017. prot. Nr. _____

Komisijas sekretārs(-e): _____