

LATVIJAS UNIVERSITĀTE
EKSAKTO ZINĀTŅU UN TEHNOLOĢIJU FAKULTĀTE

LAIMES RATA SPĒLES SERVISS

KVALIFIKĀCIJAS DARBS

Autors: Raimonds Eisaks

Studenta apliecības Nr.: re22011

Darba vadītājs Mg. energ. Sergejs Bogdanovs

RĪGA 2025

ANOTĀCIJA

Laiques rata spēles serviss piedāvā lietotājiem aizraujošu un dinamisku spēlēšanas pieredzi, kur viņi var izvēlēties dažādus galdus un likt likmes uz dažādiem sektoriem. Lietotne nodrošina reāllaika spēles pieredzi, ļaujot spēlētājiem skatīties, kā mainās laimes rats un kā tiek sadalīti laimesti atkarībā no veiktajām likmēm. Serviss rada arī spēles datus citiem servisiem datu analīzei izmantojot Kafka klasterus.

Papildus spēlei lietotājiem tiek piedāvāta iespēja sekot līdzi savai statistikai. Šī statistika ļauj spēlētājiem analizēt savu sniegumu, redzēt tendences un pieņemt pamatotus lēmumus nākamajās spēlēs. Lietotne arī nodrošina dažādus galdus, kas ļauj dažādiem lietotājiem spēlēt vienlaikus, kā arī izveidot savu privāto laimes ratu, kur piekļūt var tikai ar paroli.

Atslēgvārdi: Web, Scala, Akka Actors, Kafka, React.js, Websocket, Apache Derby
SQL

ABSTRACT

Spinning wheel game service offers users an engaging and dynamic gaming experience, where they can choose different tables and place bets on various sectors. The app provides a real-time gaming experience, allowing players to watch how the wheel spins and how winnings are distributed based on the placed bets. The service also generates game data for other services to analyze using Kafka clusters.

In addition to the game, users are provided with the ability to track their statistics. This statistics feature allows players to analyze their performance, identify trends, and make informed decisions for future games. The app also supports multiple tables, enabling different users to play simultaneously.

Keywords: Web, Scala, Akka Actors, Kafka, React.js, Websocket, Apache Derby
SQL

SATURA RĀDĪTĀJS

ANOTĀCIJA	2
ABSTRACT	3
SATURA RĀDĪTĀJS	4
APZĪMĒJUMU UN TERMINU SARAKSTS.....	7
IEVADS.....	8
Nolūks	8
Darbības sfēra.....	8
Saistība ar citiem dokumentiem	8
Pārskats.....	8
VISPĀRĒJAIS APRAKSTS.....	10
Esošā stāvokļa apraksts	10
Pasūtītājs	10
Produkta perspektīva	10
Darījumphrasības	10
Sistēmas lietotāji	11
Vispārējie ierobežojumi	12
Pieņēmumi un atkarības.....	12
PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA.....	13
2.1 Konceptuālais datubāzes apraksts	13
2.2 Funkcionālās prasības.....	13
2.2.1 Funkciju sadalījums pa moduļiem/komponentiem.....	13
2.2.2 Lietotāja modulis	16
2.2.3 Laimes ratu konfigurācijas modulis.....	25
2.2.4 Laimes rata modulis.....	28
2.2.5 Statistikas modulis	38
2.2.6 Spēles dzinēja modulis	40
2.2.7 Konta atlikuma modulis.....	44
2.2.8 Kafka modulis	46
2.3 Nefunkcionālās prasības.....	48
2.3.1 Pieejamība.....	48
2.3.2 Lietojamība	48
2.3.3 Veiktspēja.....	48

2.3.4 Uzturamība	48
2.3.5 Drošība.....	48
2.3.6 Mērogojamība	48
3.1. Datubāzes projektējums	49
3.1.1. Datubāzes loģiskais modelis.....	49
3.1.2 Datubāzes fiziskais modelis	50
3.2. Daļējs funkciju projektējums	53
3.2.1. Projektējums funkcijai “Izveidot lietotāja kontu”	53
3.2.2. Projektējums funkcijai “Dzēst lietotāja kontu”	54
3.2.3. Projektējums funkcijai “Pieteikšanās sistēmā”	54
3.2.4. Projektējums funkcijai “Atteikšanās no sistēmas”	55
3.2.5. Projektējums funkcijai “Rediģēt lietotāja vārdu”	55
3.2.6. Projektējums funkcijai “Rediģēt e-pastu”	56
3.2.7. Projektējums funkcijai “Nomainīt paroli”	56
3.2.8. Projektējums funkcijai “Iegūt lietotāja datus”	57
3.2.9. Projektējums funkcijai “Nosūtīt e-pastu klientu atbalstam”	57
3.2.10. Projektējums funkcijai “Izveidot pieprasījumu pievienoties spēles ratam”	58
3.2.11. Projektējums funkcijai “Izveidot privāto spēles ratu”	59
3.2.12. Projektējums funkcijai “Iziet no spēles rata”	59
3.2.13. Projektējums funkcijai “Pievienoties laimes ratam”	60
3.2.14. Projektējums funkcijai “Uzzināt spēles rata pieejamību”	61
3.2.15. Projektējums funkcijai “Nosūtīt ziņu tērztavā”	61
3.2.16. Projektējums funkcijai “Pievienot likmi”	62
3.2.17. Projektējums funkcijai “Dzēst likmi”	63
3.2.18. Projektējums funkcijai “Uzstādīt spēles fāzi – likmju fāze”	63
3.2.19. Projektējums funkcijai “Uzstādīt spēles fāzi – likmes uzliktas”	64
3.2.20. Projektējums funkcijai “Uzstādīt spēles fāzi – laimes rata griežšanās fāze”	64
3.2.21. Projektējums funkcijai “Uzstādīt spēles fāzi – likmju rezultātu fāze”	64
3.2.22. Projektējums funkcijai “Uzstādīt spēles fāzi – spēles kārtas beigu fāze”	65
3.2.22. Projektējums funkcijai “Iegūt spēlētāja statistiku”	65
3.2.23. Projektējums funkcijai “Iegūt laimes rata servisa statistiku”	66
3.2.24. Projektējums funkcijai “Aprēķināt laimes kārtas lietotāja laimestu”	66
3.2.25. Projektējums funkcijai “Aprēķināt likmes koeficientu”	67
3.2.26. Projektējums funkcijai “Spēles rata fāžu kontrolieris”	67
3.2.27. Projektējums funkcijai “Spēles rata fāžu kontrolieris”	68
3.2.28. Projektējums funkcijai “Papildināt konta atlikumu”	68

3.2.29. Projektējums funkcijai “Konta atlikuma spēle (react komponente)”	69
3.2.30. Projektējums funkcijai “Parsēt klases JSON formātā”	70
3.2.31. Projektējums funkcijai “Datu nosūtīšana uz Kafka severi”	70
3.3 Daļējs lietotāja saskarņu projektējums	71
3.3.1 Sākumskats	72
3.3.2 Reģistrācijas skats	73
3.3.3 Pieteikšanās skats	73
3.3.4 Vestibila skats	74
3.3.5 Spēles rata skats	75
3.3.6 Profila rediģēšanas skats	75
3.3.7 Statistikas skats	76
4. PROGRAMMATŪRAS TESTĒŠANA	77
4.1 Testpiemēri	77
4.1.1 Lietotāja modulis	77
4.1.2 Laimes ratu konfigurācijas modulis.....	79
4.1.3 Laimes rata modulis.....	79
4.1.4 Statistikas modulis	80
4.1.5 Spēles dzinēja modulis	81
4.1.6 Konta atlikuma modulis.....	82
4.2 Testēšanas žurnāls.....	82
4.2.1 Lietotāja modulis	82
4.2.2 Laimes ratu konfigurācijas modulis.....	83
4.2.3 Laimes rata modulis.....	84
4.2.4 Statistikas modulis	84
4.2.5 Spēles dzinēja modulis	85
4.2.6 Konta atlikuma modulis.....	85
5. PROJEKTA PĀRVALDĪBA	86
5.1 Projekta organizācija	86
5.2 Kvalitātes nodrošināšana.....	86
5.3 Konfigurāciju pārvaldība	87
5.4 Darbietilpības novērtējums	87
NOBEIGUMS.....	89
IZMANTOTĀ LITERATŪRA UN AVOTI.....	90
PIELIKUMI.....	91

APZĪMĒJUMU UN TERMINU SARAKSTS

1. Kafka – datu straumēšanas apstrādes līdzeklis
2. PK – primārā atslēga datubāzes modelī
3. FK – ārēja atslēga datubāzes modelī
4. Git – versiju kontroles sistēma
5. Scala – programmēšanas valoda
6. React.js - programmēšanas valoda

IEVADS

Nolūks

Šis dokuments ir radīts, lai aprakstītu laimes rata spēles servisa izstrādes prasības, funkcionālās un nefunkcionālās prasības, funkcijas, projektējumu, ierobežojumus, testu projektējumus un izpildes žurnālus. Darbs paredzēts lietotājiem, izstrādātājiem, projekta īpašniekiem un komisijai.

Darbības sfēra

Laimes rata spēles serviss ir tīmekļa lietotne, kas ļauj lietotājiem likt likmes uz sektoriem. Lietotne izmanto websocket īsziņas lai nodrošinātu saziņu starp serveri un lietotāju. Lietotnē lietotāji var iepazīties ar spēles likumiem, izvēlēties ratu un spēlēt kopā ar citiem lietotājiem.

Saistība ar citiem dokumentiem

Dokumenta noformēšanā izmantotas standartu LVS 68:1996 [1] un LVS 72:1996 [2] prasības.

Pārskats

Dokumenta struktūra ir sadalīta 5 daļās:

1. Vispārīgs apraksts, kurā atrodas esošā stāvokļa apraksts, produkta perspektīva, darījumprasības un vispārējie ierobežojumi.
2. Programmatūras prasību specifikācija - funkcionālās prasības un nefunkcionālās prasības.
3. Programmatūras projektējuma apraksts - funkciju projektējums, saskarnes projektējums, serveru projektējums.
4. Programmatūras testēšanas projektēšana dokumentācija - testēšanas piemēri un testu žurnāls)

5. Projekta pārvaldība - projekta organizācija, kvalitātes nodrošināšana, konfigurāciju pārvaldība, darbietilpības novērtējums un secinājumi.

VISPĀRĒJAIS APRAKSTS

Esošā stāvokļa apraksts

Daudzi ir spēlējuši laimes rata spēles, kuras balstās uz nejaušu iznākumu un kurās ir iespējams likt likmes, taču pārsvarā tajos likmju koeficienti ir mazi, likmju sektoru daudzums ir neliels un nav pieejama statistika un likmju vēsture, kas ļauj spēlētējiem balstoties uz savu spēles stilu analizēt savas likmes. Laimes rata spēles serviss piedāvā lietotājiem aizraujošu un interaktīvu spēles pieredzi, ļaujot likt likmes uz dažādiem sektoriem un sekot līdzi spēles norisei reāllaikā un redzēt savu un citu statistiku.

Tirgū ir pieejami analogi – ruletes tipa spēles kuras piedāvā dažādi pakalpojumu sniedzēji (Evolution un citi), taču tajos ir klasiskās ruletes likumi bez sesijas analīzes, kas nav tik aizraujoši, kā arī nav tieša iespēja pieslēgties pie viena spēļu galda – laimes rata. Laimes rata serviss iekļauj sevī šo funkcionalitāti.

Pasūtītājs

Sistēma aprakstīta pēc studenta iniciatīvas kvalifikācijas darba ietvaros.

Produkta perspektīva

Tīmekļa lietotne ar trīs apakšserveriem un vairākiem apakšservisiem.

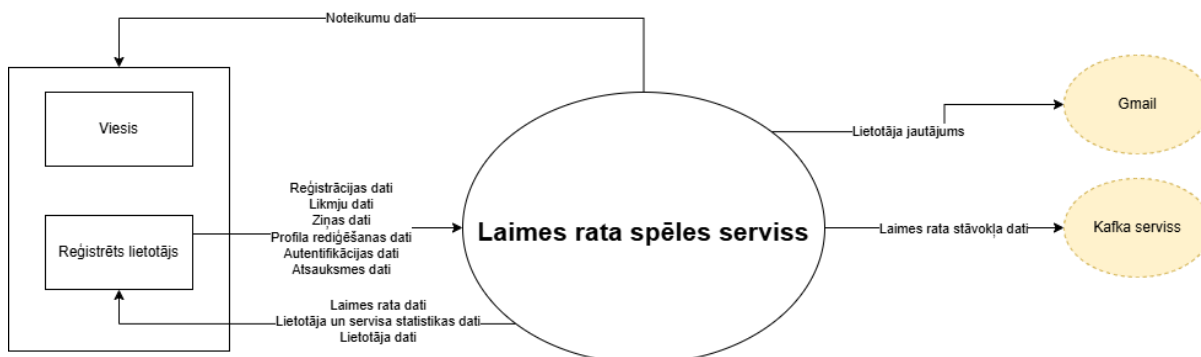
Darījumphrasības

- Iespēja pieslēgties spēles serverim
- Iespēja izvēlēties spēles ratu
- Iespēja likt likmes
- Iespēja apskatīt spēlētāju skaitu laimes ratos
- Reģistrēties serverī
- Dzēst kontu
- Iespēja rakstīt ziņas laimes rata sarakstē
- Iespēja dzēst likmi
- Iespēja redzēt spēles rata stāvokli
- Iespēja iepazīties ar spēles likumiem

- Iespēja spēlēt dažādiem lietotājiem pie viena laimes rata
- Iespēja apskatīt sava profila datus
- Iespēja rediģēt profilu
- Iespēja izveidot privāto spēles galdu
- Iespēja papildināt konta atlikumu
- Iespēja nosūtīt ziņojumu klientu atbalstam

Sistēmas lietotāji

Lietotāju grupa	Apzīmējums	Apraksts
Viesis	VS	Lietotājs, kurš var apskatīt tikai laimes rata spēles servisa likumus, pāriet uz servisa koda repozitoriju, iepazīties ar dokumentāciju un izveidot profilu.
Reģistrēts lietotājs	RL	Lietotājs, kurš var pieteikties sistēmā, rediģēt savus profila datus, pievienoties publiskajam vai privātajam laimes ratam (var izveidot to), sūtīt ziņas tērzētavā, likt likmes.



Attēls 1.1. 0. līmeņa datu plūsmas diagramma

Vispārējie ierobežojumi

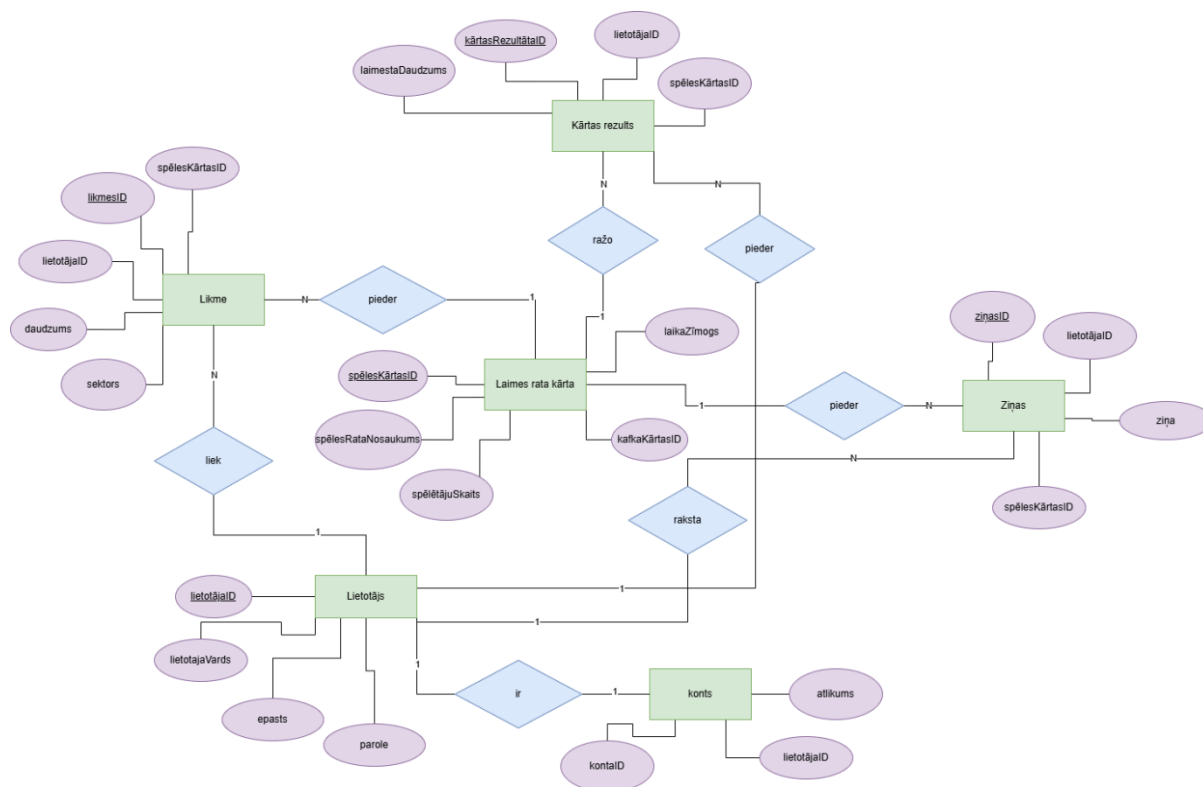
Dators ar jaunāko pārlūkprogrammas versiju (google chrome, safari u.c) un interneta pieslēgumu.

Pieņēmumi un atkarības

- Tīmekļa vietnei jābūt strādājošai vismaz 4 populārākajās pārlūkprogrammās – safari, google chrome, microsoft edge un firefox.
- Lietotājam ir jābūt pieejamam interneta pieslēgumam.
- Serveris un datu bāze jābūt nepārtraukti pieejama visiem lietotājiem

PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA

2.1 Konceptuālais datubāzes apraksts



Attēls 2.1. Datubāzes konceptuālais modelis

Attēlā 2.1 ir attēlots programmatūras datubāzes konceptuālais modelis. Modelī ir 6 entitijas, kurās glabājas sistēmai svarīgi dati – dati par lietotāju, spēles un lietotāja ražoti dati (likmes, ziņas u.t.t.).

2.2 Funkcionālās prasības

Sistēmā ir 7 moduļi, kuros ir funkciju kopums ar mērķi izpildīt noteiktus uzdevumus veiksmīgai sistēmas darbībai. Moduļi ir atkarīgi viens no otra.

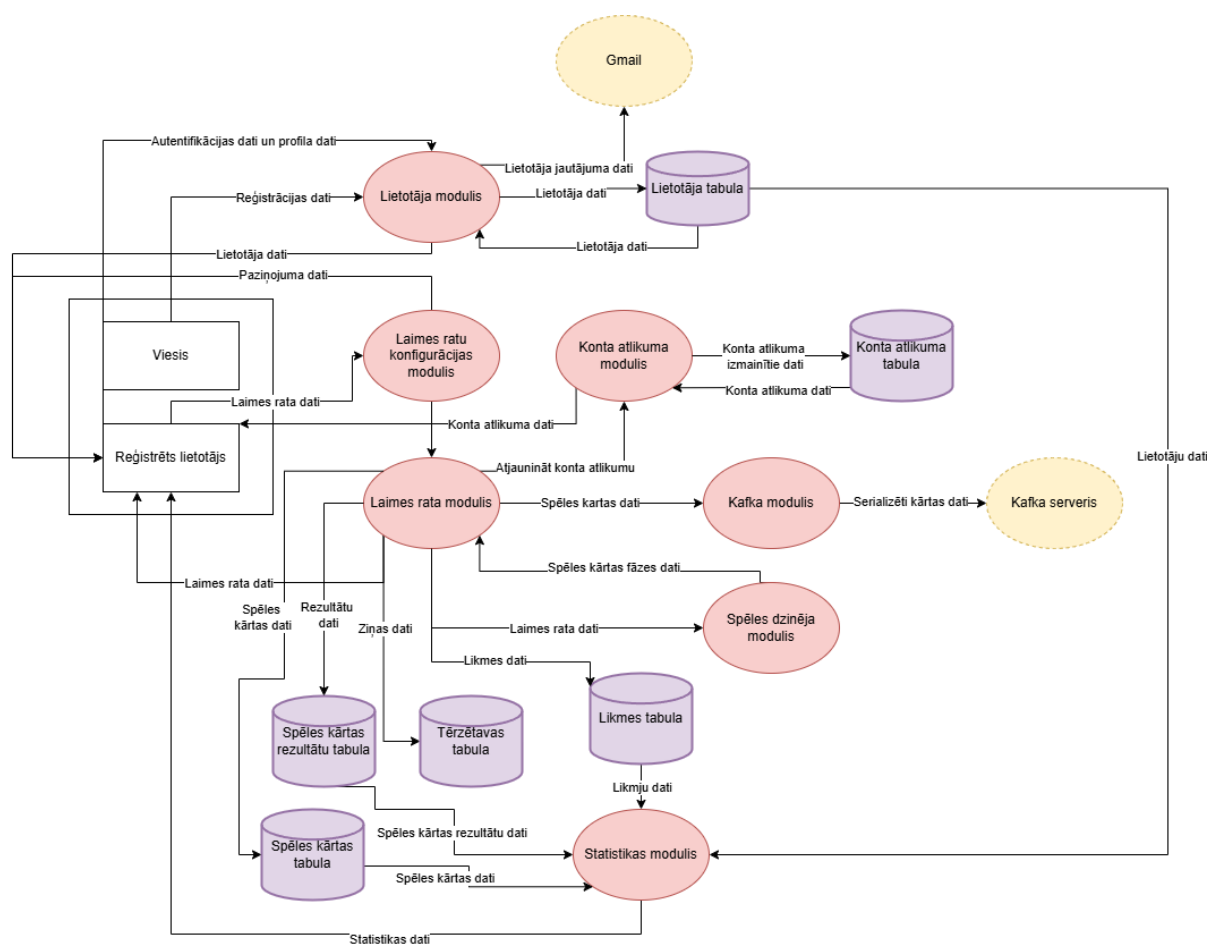
2.2.1 Funkciju sadalījums pa moduļiem/komponentiem

Lietotnes funkcijas sadalītas moduļos

Modulis	Funkcija	Identifikators
Lietotāja modulis	Izveidot lietotāja kontu	USER001
	Dzēst lietotāja kontu	USER002

	Pieteikšanās sistēmā	USER003
	Atteikšanās no sistēmas	USER004
	Rediģēt lietotāja vārdu	USER005
	Rediģēt e-pastu	USER006
	Nomainīt paroli	USER007
	Iegūt lietotāja datus	USER008
	Nosūtīt e-pastu klientu atbalstam	USER009
Laiemes ratu konfigurācijas modulis	Izveidot pieprasījumu pievienoties laimes ratam	TM001
	Izveidot privāto laimes ratu	TM002
Laiemes rata modulis	Iziet no laimes rata	SPW001
	Pievienoties laimes ratam	SPW002
	Uzzināt laimes rata pieejamību	SPW003
	Nosūtīt ziņu tērztētavā	SPW004
	Pievienot likmi	SPW005
	Dzēst likmi	SPW006
	Uzstādīt spēles fāzi – likmju fāze	SPW007
	Uzstādīt spēles fāzi – likmes uzliktas	SPW008
	Uzstādīt spēles fāzi – laimes rata griežšanās fāze	SPW009
	Uzstādīt spēles fāzi – likmju rezultātu fāze	SPW010
	Uzstādīt spēles fāzi – spēles kārtas beigu fāze	SPW011
Statistikas modulis	Iegūt spēlētāja statistiku	ST001
	Iegūt laimes rata servisa statistiku	ST002
Spēles dzinēja modulis	Aprēķināt laimes kārtas lietotāja laimestu	GE001

	Aprēķināt likmes koeficientu	GE002
	Laimes rata stāvokļa nomaiņa	GE003
	Laimes rata laimīga cipara ģenerēšana	GE004
Konta atlikuma modulis	Papildināt konta atlikumu	BN001
	Konta atlikuma spēle (react komponente)	BN002
Kafka modulis	Parsēt klases JSON formātā	KFK001
	Datu nosūtīšana uz Kafka severi	KFK002



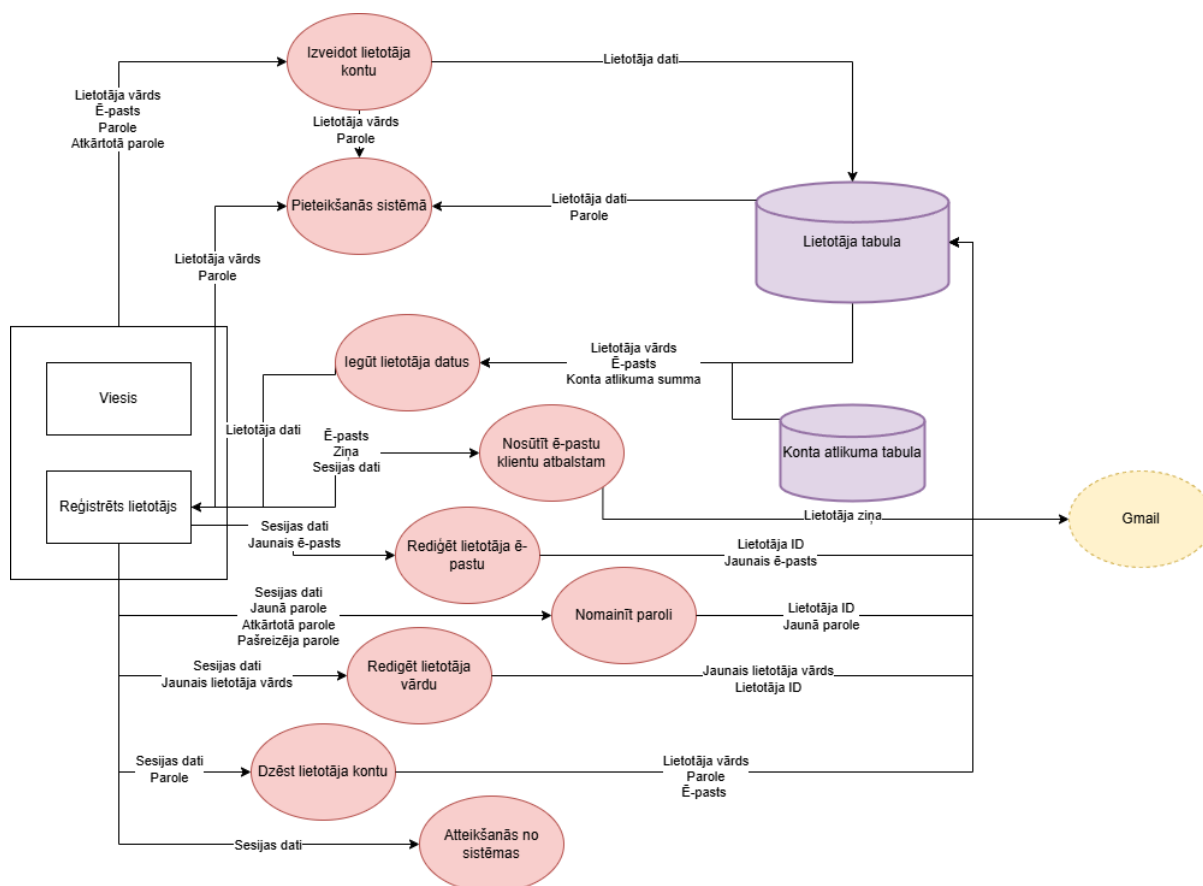
Attēls 2.2 1.līmeņa datu plūsmas diagramma

Attēlā 2.2 redzama 1.līmeņa datu plūsmas diagramma kurā ir 7 moduļi. Viesim ir piekļuve tikai lietotāja moduļim, kur tas var reģistrēties serverī. Reģistrēts lietotājs jau piekļūst gandrīs visiem moduļiem izņemot spēles dzinēja un kafka moduli. Redzams, ka laimes rata modulis visvairāk saistīts ar datubāžu ierakstu veidošanu. Spēles dzinēja un kafka moduli izmanto tikai laimes rata modulis. Spēles dzinēja modulis definē laimes rata moduļa secīgu funkciju darbību, kafka modulis glabā datus par laimes rata moduļa stāvokļiem.

Statistikas modulis agregē datubāzes tabulu datus un nosūta reģistrētam lietotājam datu savu un laimes rata servisa statistiku. Laimes rata konfigurācijas modulis veido jaunus laimes ratus un sūta pieprasījums pievienoties esošiem laimes ratiem.

Konta atlikuma moduli izmanto reģistrēts lietotājs un laimes rata modulis. Laimes rata modulis veic izmaiņas kontā, kad spēlētājs iziet no laimes rata (noņem vai pievieno laimestu), savukārt, reģistrēts lietotājs spēlē spēli, kurā tam ir iespēja uzvarēt konta papildinājumu.

2.2.2 Lietotāja modulis



Attēls 2.3 Lietotāja moduļa 1.līmeņa datu plūsmas diagramma

Funkcija “Izveidot lietotāja kontu”

Nosaukums	Izveidot lietotāja kontu
Identifikators	USER001
Apraksts	
Funkcija paredzēta jauna lietotāja konta izveidei.	
Ievaddati	
<ol style="list-style-type: none"> 1. Lietotājvārds – obligāts lauks 2. Ē-pasts – obligāts lauks 3. Parole – obligāts lauks 4. Atkārtotā parole – obligāts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. Tiek pārbaudīts ievaddatu korektums ar datubāzes atribūtu nosacījumiem: (vārds un ē-pasts jābūt unikāliem laukiem) 2. Tiek pārbaudīt vai lietotāja vārds atbilst fromātam (nav atstarpju un īpašo simbolu) 3. Tiek pārbaudīts vai parole ir 8 simbolus gara un sakrīt ar atkārtoto paroli 4. Lietotājam tiek piešķirts unikāls identifikators; 5. Tiek izveidots jauns profils (datubāzē izveido jaunu ierakstu) 6. Lietotājam tiek izveidota sesija 	
Izvaddati	
Lietotājs tiek pāradresēts uz vestibila skatu	
Kļūdu paziņojumi	

1. “Tāds e-pasts jau eksistē”
2. “Tāds lietotājvārds jau eksistē”
3. “Parole un atkārtotā parole nesakrīt”
4. “Lietotāja vārds neatbilst formātam”

Funkcija “**Dzēst lietotāja kontu**”

Nosaukums	Dzēst lietotāja kontu
Identifikators	USER002
Apraksts	
Funkcija paredzēta lietotāja konta dzēšanai.	
Ievaddati	
<ol style="list-style-type: none"> 1. Lietotāja sesijas dati – automātiski aizpildīts obligāts lauks 2. Parole – obligāts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. Lietotāja dati tiek dzēsti no datubāzes 2. Lietotāja sesija tiek pārtraukta 	
Izvaddati	
Lietotājs tiek pārraidīts uz sākumskatu.	
Kļūdu paziņojumi	
<ol style="list-style-type: none"> 1. “Nepareiza parole” 	

2. "Nav aizpildīti visi obligātie lauki"

Funkcija “**Pieteikšanās sistēmā**”

Nosaukums	Pieteikšanās sistēmā
Identifikators	USER003
Apraksts	
Funkcija paredzēta reģistrēta lietotāja pieteikšanās sistēmai .	
Ievaddati	
<ol style="list-style-type: none">1. Lietotāja vārds - obligāts lauks2. Parole - obligāts lauks	
Apstrāde	
<ol style="list-style-type: none">1. Tiek pārbaudīts vai lietotāja vārds eksistē datubāzē2. Tiek pārbaudīts vai parole sakrīt ar datubāzē esošo3. Tiek pārbaudīts vai lietotājs ir jau pieteicies sistēmā4. Lietotājam izveido sesiju	
Izvaddati	
Lietotājs tiek pāadresēts uz vestibila skatu	
Kļūdu paziņojumi	
<ol style="list-style-type: none">1. “Nepareiza parole vai lietotājvārds”2. "Nav aizpildīti visi obligātie lauki"	

Funkcija “**Atteikšanās no sistēmas**”

Nosaukums	Atteikšanās no sistēmas
Identifikators	USER004
Apraksts	
Fukcija ļauj reģistrētam lietotājam atteikties no sistēmās	
Ievaddati	
1. Lietotāja sesijas dati – – automātiski aizpildīts obligāts lauks	
Apstrāde	
1. Lietotāja actor modelis tiek dzēsts 2. Lietotājs tiek izņemts no sesijas tabulas un tiek dzēsti sesijas dati	
Izvaddati	
Lietotājs tiek pāradresēts uz sākumskatu.	

Funkcija “**Rediģēt lietotāja vārdu**”

Nosaukums	Rediģēt lietotāja vārdu
Identifikators	USER005
Apraksts	
Funkcija paredzēta lietotāja vārda izmaiņai.	

Ievaddati
<ol style="list-style-type: none"> 1. Lietotāja jaunais vārds - obligāts lauks 2. Lietotāja sesijas dati – automātiski aizpildīts obligāts lauks
Apstrāde
<ol style="list-style-type: none"> 1. Tiek pārbaudīts vai tāds lietotāja vārds eksistē 2. Datubāzē lietotājam nomaina lietotāja vārdu
Izvaddati
Lietotājam tiek paziņots par lietotāja vārda nomaiņu
Kļūdu paziņojumi
<ol style="list-style-type: none"> 1. “Nav aizpildīts lauks” 2. “Lietotāja vārds neatbilst formātam” 3. “Lietotāja vārds jau eksistē”

Funkcija “**Rediģēt e-pastu**”

Nosaukums	Rediģēt e-pastu
Identifikators	USER006
Apraksts	
	Funkcija paredzēta lietotāja epasta nomaiņai.
Ievaddati	
	<ol style="list-style-type: none"> 1. Lietotāja e-pasts - obligāts lauks

2. Lietotāja sesijas dati – automātiski aizpildīts obligāts lauks
Apstrāde
<ol style="list-style-type: none"> 1. Tiek pārbaudīts vai ēpasts atbilst ē-pasta formātam 2. Tiek pārbaudīts vai tāds ē-pasts eksistē 3. Datubāzē lietotājam nomaina e-pastu
Izvaddati
Lietotājam tiek paziņots par e-pasta nomaiņu
Kļūdu paziņojumi
<ol style="list-style-type: none"> 1. “Nav aizpildīts lauks” 2. “Neatbilst ē-pasta formātam” 3. “Tāds ē-pasts jau eksistē”

Funkcija “Nomainīt paroli”

Nosaukums	Nomainīt paroli
Identifikators	USER007
Apraksts	
Funkcija paredzēta lietotāja paroles nomaiņai.	
Ievaddati	
<ol style="list-style-type: none"> 1. Pašreizēja parole – obligāts lauks 2. Jaunā parole – obligāts lauks 3. Atkārtota jaunā parole – obligāts lauks 	

4. Lietotāja sesijas dati – automātiski aizpildīts obligāts lauks
Apstrāde
<ol style="list-style-type: none"> 1. Tiek pārbaudīts, vai jaunā un atkārtotā parole sakrīt, nav tukši lauki un jābūt vismaz 8 simbolus garai 2. Tiek pārbaudīts vai pašreizējā parole sakrīt ar datubāzē esošo 3. Ja pārbaudes tika pārietas veiksmīgi tad datubāzē tiek nomainīta parole
Izvaddati
Lietotājam tiek paziņots par paroles nomaiņu
Kļūdu paziņojumi
<ol style="list-style-type: none"> 1. “Atkārtotā un jaunā parole nesakrīt” 2. “Nav aizpildīti lauki” 3. “Jaunā un esoša parole sakrīt” 4. “Jaunai parolei jābūt vismaz 8 simbolus garai” 5. “Pašreizējā parole ir nepareiza”

Funkcija “**Iegūt lietotāja datus**”

Nosaukums	Iegūt lietotāja datus
Identifikators	USER008
Apraksts	
Funkcija paredzēta lietotāja datu iegūšanai.	
Ievaddati	

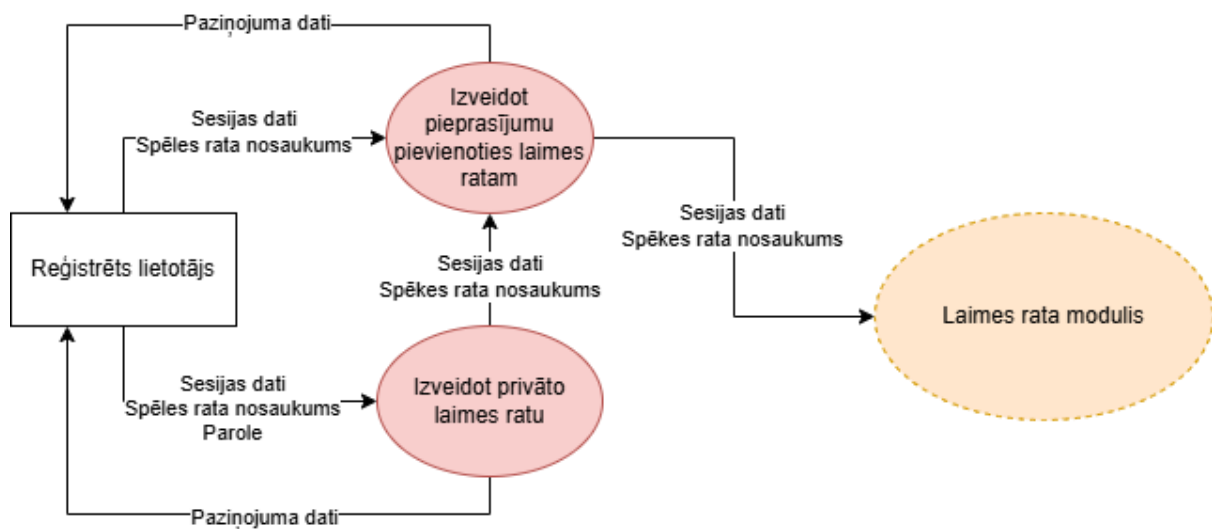
1. Lietotāja vārds - aizpildīts automātiski, lietotājam neredzams
Apstrāde
1. Tiek veikts pieprasījums datubāzē un atgriezti dati par lietotāju
Izvaddati
Saraksts ar lietotāja datiem
Kļūdu paziņojumi
1. “Lietotājs ar tādu lietotāja vārdu neeksistē”

Funkcija “Nosūtīt e-pastu klientu atbalstam”

Nosaukums	Nosūtīt e-pastu klientu atbalstam
Identifikators	USER009
Apraksts	
	Funkcija paredzēta ē-pasta nosūtīšanai klientu atbalstam
Ievaddati	
	<ol style="list-style-type: none"> 1. Lietotāja sesijas dati – automātiski aizpildāms obligāts lauks 2. Lietotāja ē-pasts - automātiski aizpildāms obligāts lauks 3. Ziņa – obligāts lauks
Apstrāde	
	1. Tiek nosūtīts ē-pasts no lietotāja vārda klientu atbalstam

Izvaddati
Lietotājam paziņo, ka ē-pasts tika nosūtīts
Kļūdu paziņojumi
1. “Ē-pasts netika nosūtīts”

2.2.3 Laimes ratu konfigurācijas modulis



Attēls 2.4. Spēles ratu konfigurācijas 2. līmeņa datu plūsmas diagramma

Funkcija “Izveidot pieprasījumu pievienoties laimes ratam”

Nosaukums	Izveidot pieprasījumu pievienoties laimes ratam
Identifikators	TM001
Apraksts	
Funkcija paredzēta izveidot pieprasījumu pievienoties spēles ratam	
Ievaddati	

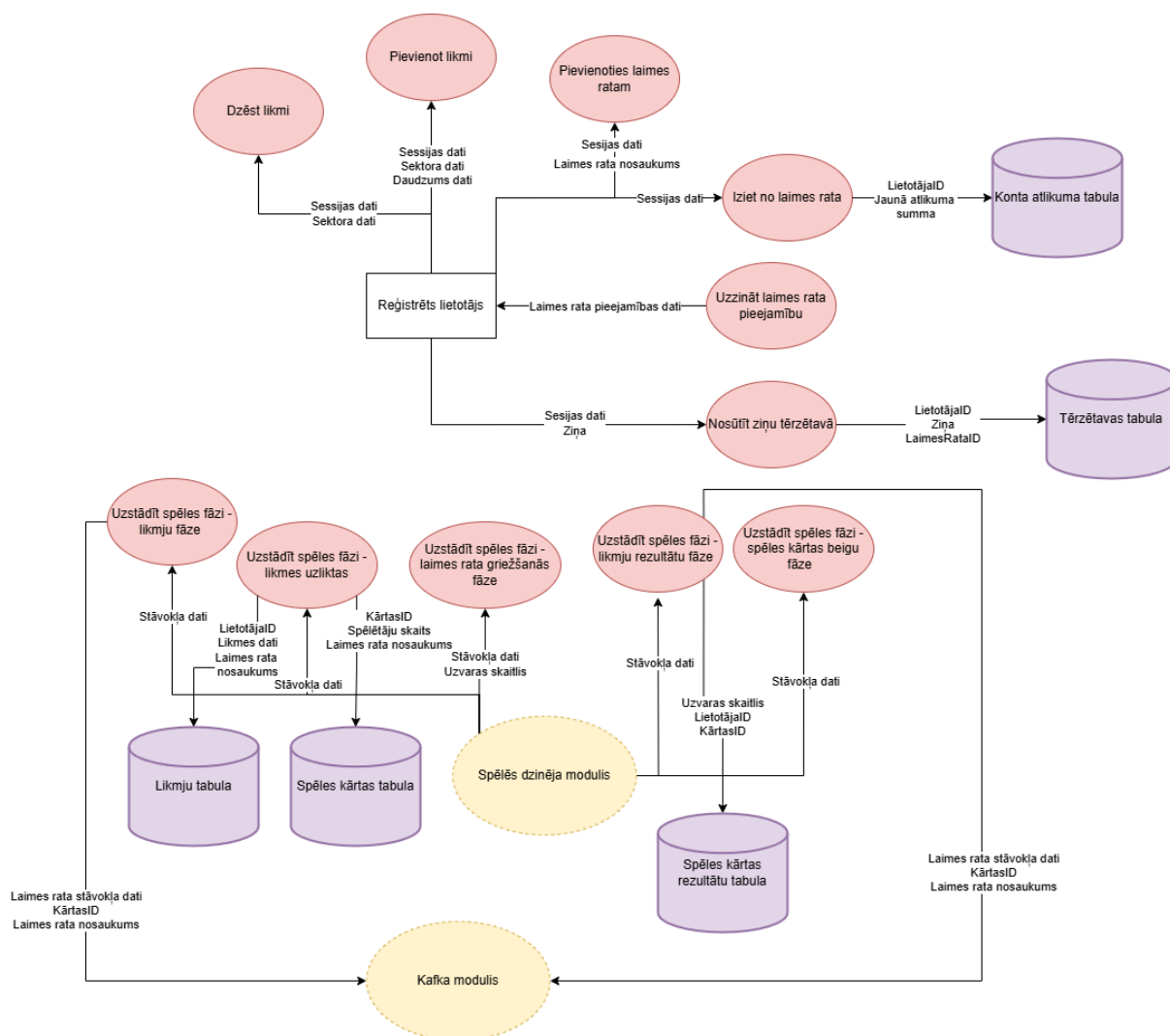
<ol style="list-style-type: none"> 1. Lietotāja sesijas dati – automātiski aizpildīts obligāts lauks 2. Spēles rata nosaukums – obligāts lauks 3. Spēlētāju sesijas pārvaldnieks - automātiski aizpildīts obligāts lauks
Apstrāde
<ol style="list-style-type: none"> 1. Pārbauda spēles rats ar tādu nosaukumu eksistē 2. Pārbauda, vai spēlētājs jau spēlē kādā no laimes ratiem 3. Nosūta pieprasījumu pieteikties laimes ratam
Izvaddati
Pieprasījums spēles ratam
Kļūdu paziņojumi
<ol style="list-style-type: none"> 1. “Spēlētājs jau spēlē kādā no laimes ratiem” 2. “Laimes rats ar tādu nosaukumu neeksistē”

Funkcija “**Izveidot privāto laimes ratu**”

Nosaukums	Izveidot privāto laimes ratu
Identifikators	TM002
Apraksts	
Funkcija paredzēta izveidot privāto spēles ratu	
Ievaddati	
<ol style="list-style-type: none"> 1. Spēles rata nosaukums – obligāts lauks 2. Parole – neobligāts lauks 	

3. Sesijas dati – automātiski aizpildīts lauks
Apstrāde
<ol style="list-style-type: none"> 1. Pārbauda spēles rats ar tādu nosaukumu eksistē 2. Pārbauda, vai parole nav tukša simbolu virkne un ir vismaz 8 simbolus gara 3. Izveido jaunu spēles rata sesiju un pievieno spēles rata sesijas tabulā 4. Izdara pieprasījumu pievienoties izveidotajam spēles ratam
Izvaddati
Lietotājs tiek pāradresēts uz spēles rata skatu.
Kļūdu paziņojumi
<ol style="list-style-type: none"> 1. “Laimes rats ar tādu nosaukumu jau eksistē”

2.2.4 Laimes rata modulis



Attēls 2.5. Laimes rata moduļa 2. līmeņa datu plūsmas diagramma

Funkcija “Iziet no laimes rata”

Nosaukums	Iziet no laimes rata
Identifikators	SPW001
Apraksts	
	Funkcija atļauj lietotājam iziet no laimes rata
Ievaddati	

<ol style="list-style-type: none"> 1. Spēles rata sesijas dati – automātiski aizpildīts lauks 2. Lietotāja sesijas dati – automātiski aizpildīts lauks
Apstrāde
<ol style="list-style-type: none"> 1. Spēlētāja sesijas datus noņem spēles rata datus 2. Spēlētājs tiek izņemts no spēles rata spēlētāju saraksta
Izvaddati
Lietotājs tiek pāradresēts uz vestibila skatu.

Funkcija “Pievienoties laimes ratam”

Nosaukums	Pievienoties laimes ratam
Identifikators	SPW002
Apraksts	
Funkcija atļauj lietotājam pievienoties laimes ratam	
Ievaddati	
<ol style="list-style-type: none"> 1. Lietotāja sesijas dati – automātiski aizpildīts lauks 2. Laimes rata sesijas dati – obligāts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. Spēlētāja sesijas datus pievieno spēles rata sesijas datus 2. Spēlētājs tiek pievienots spēles rata spēlētāju sarakstam 3. Ja spēlē tiek izsaukta funkcijas, kura sāk jaunu spēles kārtu 	

Izvaddati
Lietotājs tiek pāradresēts uz spēles rata skatu.

Funkcija “Uzzināt spēles rata pieejamību”

Nosaukums	Uzzināt spēles rata pieejamību
Identifikators	SPW003
Apraksts	
Funkcija atļauj uzzināt cik brīvu vietu ir spēles ratā	
Ievaddati	
<ol style="list-style-type: none"> 1. Spēles rata nosaukums – automātiski aizpildīts lauks 2. Lietotāja sesijas dati - automātiski aizpildīts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. Spēlēs rata spēlētāju tabulā tiek saskaitīts cik spēlētāju spēlē 2. Teik aprēķināts skaitlis, cik spēlētāju var spēlēt (pašlaik visos spēles ratus maksimālais spēlētāju skaits ir 100) 	
Izvaddati	
Skatlis, kurš parāda cik ir brīvu vietu	
Kļūdu paziņojumi	
<ol style="list-style-type: none"> 1. “Spēles rats ar tādu nosaukumu neeksistē” 	

Funkcija “Nosūtīt ziņu tērzētavā”

Nosaukums	Nosūtīt ziņu tērzētavā
Identifikators	SPW004
Apraksts	
Funkcija atļauj lietotājam nosūtīt ziņu tērzētavā	
Ievaddati	
<ol style="list-style-type: none"> 1. Spēles rata sesijas dati – automātiski aizpildīts lauks 2. Lietotāja sesijas dati – automātiski aizpildīts lauks 3. Ziņa – obligāts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. Tiek iegūti visu spēles rata spēlētāju sesijas dati 2. Katram spēlētājam tiek nosūtīta ziņa 	
Izvaddati	
Jaunā ziņa parādās tērzētavas nodaļā, spēles rata skatā	
Kļūdu paziņojumi	
<ol style="list-style-type: none"> 1. “Ziņa nav nosūtīta” 	

Funkcija “Pievienot likmi”

Nosaukums	Pievienot likmi
------------------	------------------------

Identifikators	SPW005
Apraksts	
Funkcija atļauj lietotājam likt likmi spēles kārtā	
Ievaddati	
<ol style="list-style-type: none"> 1. Spēles rata sesijas dati – automātiski aizpildīts lauks 2. Lietotāja sesijas dati – automātiski aizpildīts lauks 3. Likmes sektors – obligāts lauks 4. Daudzums – obligāts lauks 5. Konta atlikuma daudzums – automātiski aizpildīts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. Tiek validēti likmju ievades lauki – (Likmes sektors – skaitlis no 1 līdz 100 un daudzums no 1 līdz konta atlikuma daudzumam) 2. Pārbauda vai spēles rata fāze ir likmju fāze 3. Pārbauda vai konta atlikuma daudzums ir pietiek likmes likšanai 4. Ja validācija ir veiksmīga un ir likmju fāze, tad likme tiek pievienota spēles rata likmju tabulai. 	
Izvaddati	
Likmju nodaļā lietotājam parādās jauna likme.	
Kļūdu paziņojumi	
<ol style="list-style-type: none"> 1. “Likmes sektoram jābūt skaitlim no 1 līdz 100” 2. “Nepietiek līdzekļu” 	

Funkcija “**Dzēst likmi**”

Nosaukums	Dzēst likmi
Identifikators	SPW006
Apraksts	
Funkcija atļauj lietotājam likt likmi spēles kārtā	
Ievaddati	
<ol style="list-style-type: none"> 1. Spēles rata sesijas dati – automātiski aizpildīts lauks 2. Lietotāja sesijas dati – automātiski aizpildīts lauks 3. Likmes sektors – obligāts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. Pārbauda vai spēles rata fāze ir likmju fāze 2. Ja lietotājam eksistē likme uz doto likmes sektoru, tad likme tiek dzēsta no likmju tabulas. 	
Izvaddati	
Likmju nodaļā lietotājam pazūd dzēstā likme.	
Kļūdu paziņojumi	
<ol style="list-style-type: none"> 1. “Nevar izdzēst likmi – likmju fāzē beidzās” 	

Funkcija “Uzstādīt spēles fāzi – likmju fāze”

Nosaukums	Uzstādīt spēles fāzi – likmju fāze
Identifikators	SPW007

Apraksts
Funkcija nomaina spēles rata fāzi
Ievaddati
1. Spēles rata sesijas dati – automātiski aizpildīts lauks
Apstrāde
<ol style="list-style-type: none"> 1. Spēles rata stāvoklis tiek nomainīts uz - likmju faze 2. Likmju tabulā visas iepriekšējās likmes tiek dzēstas 3. Spēles rata kārtai tiek ģenerēts jauns identifikators 4. Tiek nosūtīti dati uz kafka servisu
Izvaddati
Lietotājs spēles fāžu nodaļā redz paziņojumu, ka likmju fāze ir sākusies

Funkcija “**Uzstādīt spēles fāzi – likmes uzliktas**”

Nosaukums	Uzstādīt spēles fāzi – likmes uzliktas
Identifikators	SPW008
Apraksts	
Funkcija nomaina spēles rata fāzi	
Ievaddati	
1. Spēles rata sesijas dati – automātiski aizpildīts lauks	

Apstrāde
<ol style="list-style-type: none"> 1. Spēles rata stāvoklis tiek nomainīts uz - likmju faze beizdās 2. Spēles rata kārtas likmes tiek ierakstītas tabulā
Izvaddati
Lietotājs spēles fāžu nodaļā redz paziņojumu, ka likmju fāze ir beigusies

Funkcija “**Uzstādīt spēles fāzi – spēles rats griežās**”

Nosaukums	Uzstādīt spēles fāzi – spēles rats griežās
Identifikators	SPW009
Apraksts	
	Funkcija nomaina spēles rata fāzi
Ievaddati	
	<ol style="list-style-type: none"> 1. Spēles rata sesijas dati – automātiski aizpildīts lauks 2. Uzvaras skaitlis – obligāts lauks
Apstrāde	
	<ol style="list-style-type: none"> 1. Spēles rata stāvoklis tiek nomainīts uz – spēles rats griežas
Izvaddati	
	Lietotājs spēles fāžu nodaļā redz paziņojumu, ka spēles rats griežas

Funkcija “**Uzstādīt spēles fāzi – likmju rezultātu fāze**”

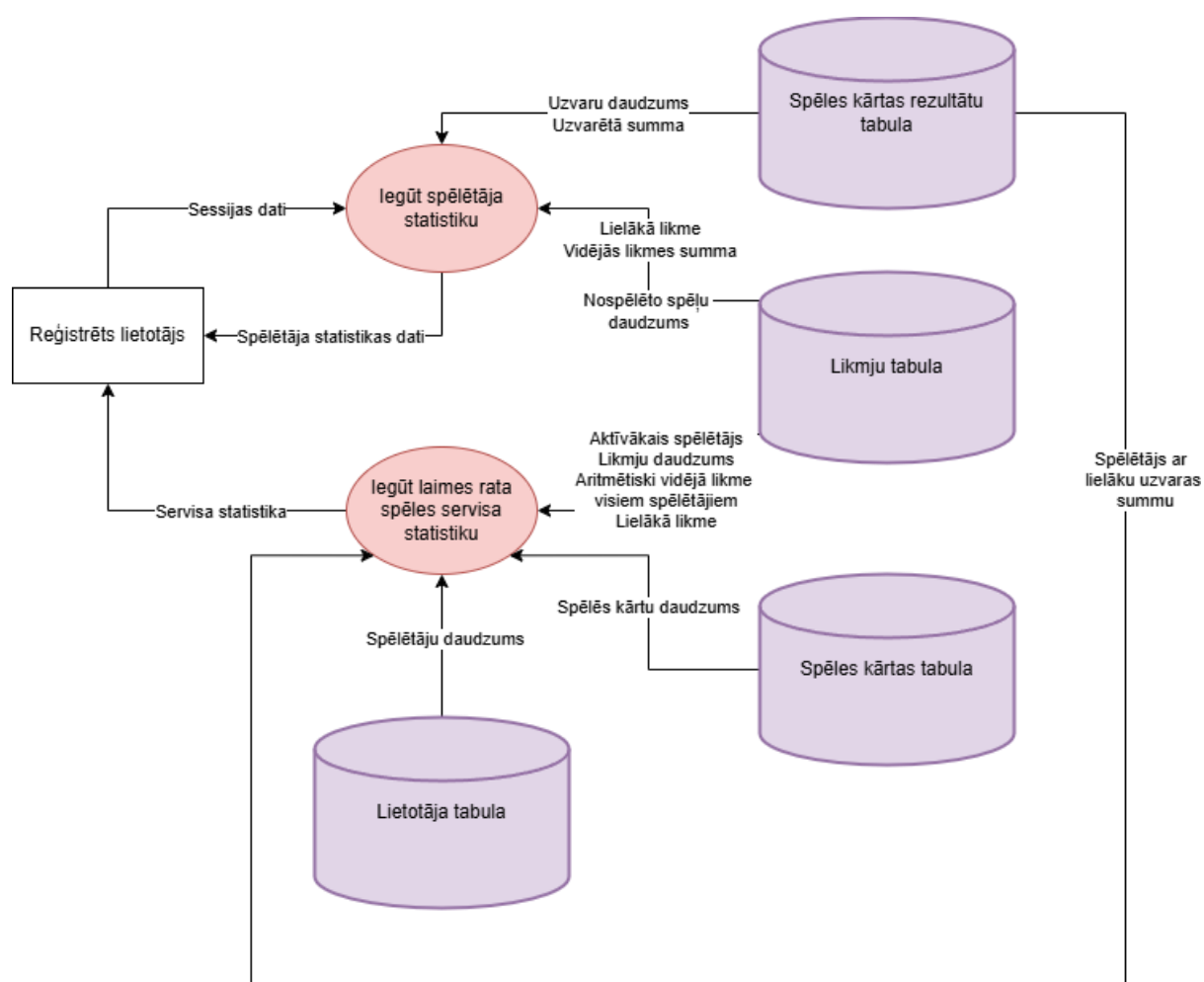
Nosaukums	Uzstādīt spēles fāzi – likmju rezultātu fāze
Identifikators	SPW010
Apraksts	
Funkcija nomaina spēles rata fāzi	
Ievaddati	
<ol style="list-style-type: none"> 1. Spēles rata sesijas dati – automātiski aizpildīts lauks 2. Uzvaras skaitlis – obligāts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. Lietotājam tiek izskaitļota summa, kuru tas uzvarēja vai zaudēja 2. Dati par spēles stāvokli tiek nosūtīti un kafa servisu 3. Laimes rata likmes tabulā tiek dzēstas visas likmes 	
Izvaddati	
Lietotājs redz vizuāli laimīgo skaitli un kārtas nodaļā redz uzvarētu/zaudēto summu.	

Funkcija “Uzstādīt spēles fāzi – spēles kārtas beigu fāze”

Nosaukums	Uzstādīt spēles fāzi – spēles kārtas beigu fāze
Identifikators	SPW011
Apraksts	
Funkcija nomaina spēles rata fāzi	

Ievaddati
1. Spēles rata sesijas dati – automātiski aizpildīts lauks
Apstrāde
<ol style="list-style-type: none"> 1. Spēles rata stāvoklis tiek nomainīts uz – spēles kārtā beidzās 2. Ja spēles rats ir privāts, tad tiek pārbaudīts vai kāds spēlē 3. Ja neviens nespēlē privātais laimes rats tiek dzēsts 4. Ja spēlē tiek izsaukta funkcijas, kura sāk jaunu spēles kārtu
Izvaddati
Lietotājs spēles fāžu nodaļā redz paziņojumu, ka spēles kārtā beidzās

2.2.5 Statistikas modulis



Attēls 2.6. Statistikas moduļa 2. līmeņa datu plūsmas diagramma

Funkcija “Iegūt spēlētāja statistiku”

Nosaukums	Iegūt spēlētāja statistiku
Identifikators	ST001
Apraksts	
	Funkcija nomaina spēles rata fāzi
Ievaddati	

1. Lietotāja vārds – automātiski aizpildīts lauks
Apstrāde
1. Tiek pārbaudīts vai lietotājs ar doto lietotājvārdu eksistē 2. Tiek veikti vairāki pieprasījumi datubāzē un atlasīti aktuālie dati par lietotāju
Izvaddati
Lietotāja statistikas dati
Kļūdu paziņojumi
1. “Lietotājs neeksistē”

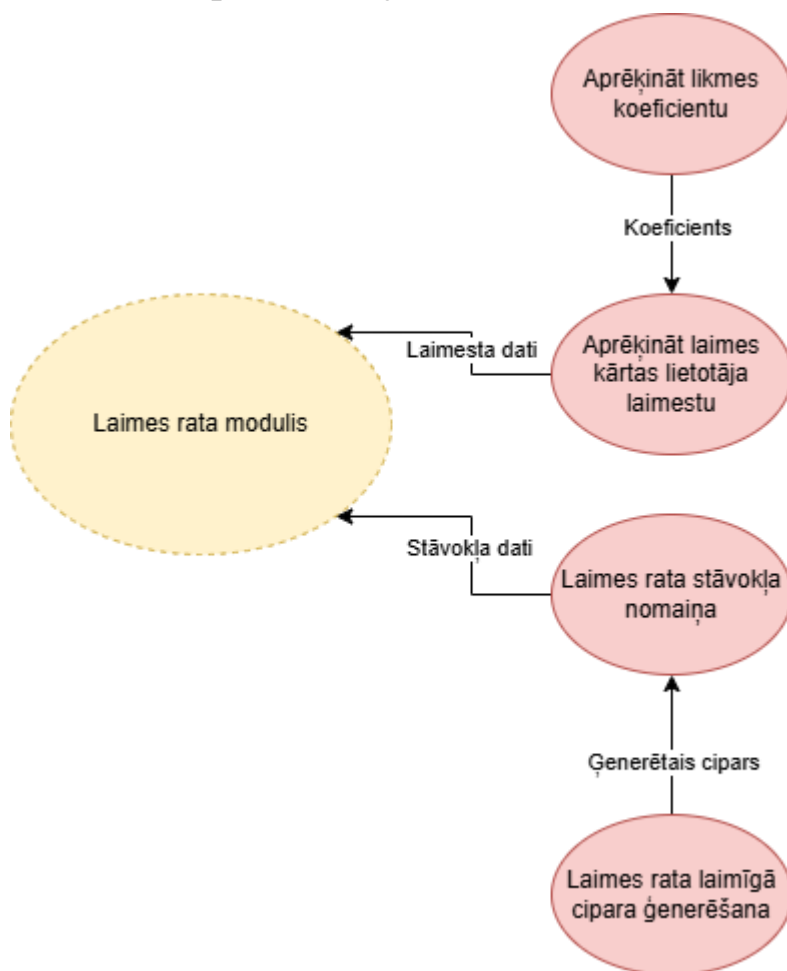
Funkcija “**Iegūt laimes rata servisa statistiku**”

Nosaukums	Iegūt laimes rata servisa statistiku
Identifikators	ST002
Apraksts	
	Funkcija nomaina spēles rata fāzi
Ievaddati	
	-
Apstrāde	
	1. Tiek veikti vairāki pieprasījumi datubāzē un atlasīti aktuālie dati par servisu

Izvaddati

Servisa statistikas dati

2.2.6 Spēles dzinēja modulis



Attēls 2.7. Spēles dzinēja 2. līmeņa datu plūsmas diagramma

Funkcija “Aprēķināt laimes kārtas lietotāja laimestu”

Nosaukums	Aprēķināt spēles kārtas lietotāja laimestu
Identifikators	GE001
Apraksts	

Funkcija atļauj aprēķināt laimestu
Ievaddati
<ol style="list-style-type: none"> 1. Likmju tabula – automātiski aizpildīts lauks 2. Spēlētāja sesijas data – automātiski aizpildīts lauks 3. Uzvaras skaits – obligāts lauks
Apstrāde
<ol style="list-style-type: none"> 1. Katram lietotājam balstoties uz kārtas likmēm un uzvaras skaitli aprēķina cik lielu daudzumu ir uzvarējis/zaudējis.
Izvaddati
Saraksts ar spēlētāja sesijas datiem un summu, kuru ir uzvarējis/zaudējis.

Funkcija “**Aprēķināt likmes koeficientu**”

Nosaukums	Aprēķināt likmes koeficientu
Identifikators	GE002
Apraksts	
Funkcija atļauj aprēķināt likmes koeficientu	
Ievaddati	
<ol style="list-style-type: none"> 1. Likmes daudzums – obligāts lauks 2. Uzvaras skaits – obligāts lauks 	
Apstrāde	

<ol style="list-style-type: none"> 1. Ja uzvaras skaits ir 100, koeficients ir 50 2. Ja uzvaras skaits ir pāra skaits, tad koeficients ir 3 3. Ja uzvaras skaits ir nepāra skaits, tad koeficients ir 4. Likmes daudzums tiek reizināts ar koeficientu
Izvaddati
Atgriež uzvaras koeficientu, ar kuru reizina likmes daudzumu

Funkcija “Spēles rata fāžu kontrolieris”

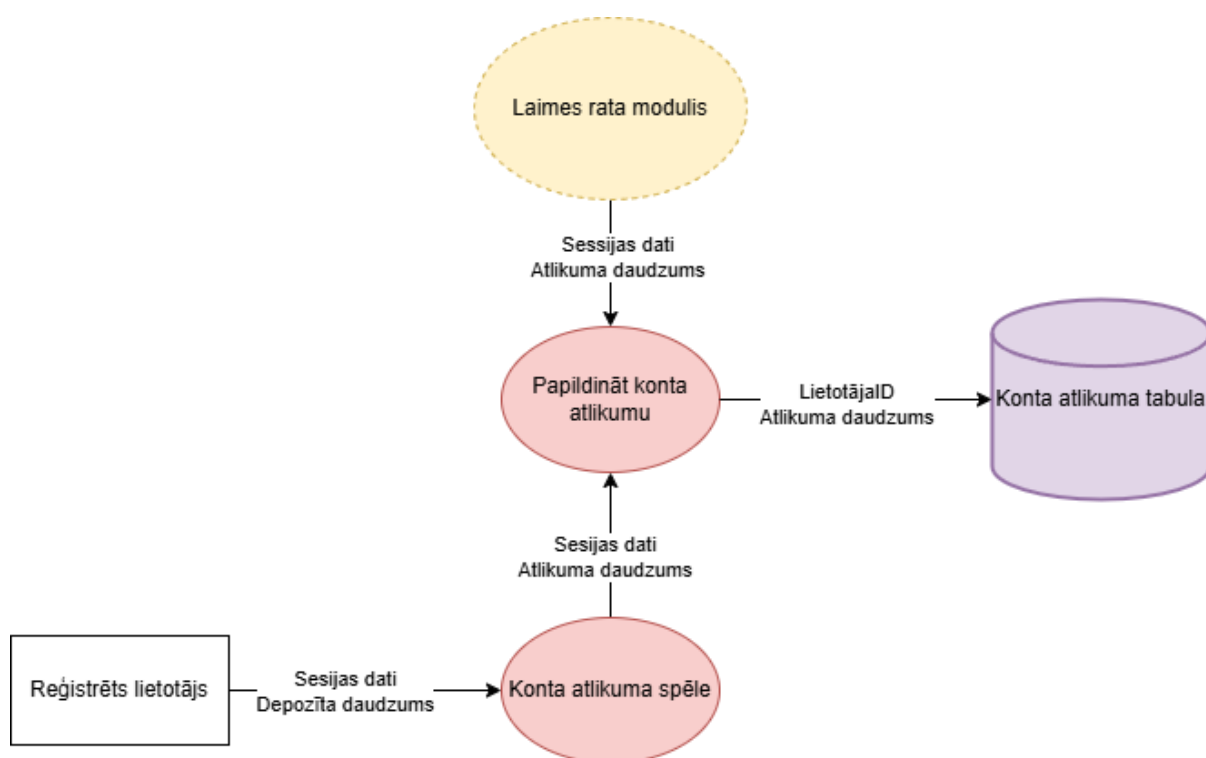
Nosaukums	Spēles rata fāžu kontrolieris
Identifikators	GE003
Apraksts	
Funkcija atļauj aprēķināt likmes koeficientu	
Ievaddati	
<ol style="list-style-type: none"> 1. Laimes rata sesijas dati – automātiski aizpildīts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. Funkcija aizsūta pieprasījumu spēles fāzes nomaiņai (likmju fāze) laimes ratam 2. Funkcija nogaida 10 sekundes 3. Funkcija aizsūta pieprasījumu spēles fāzes nomaiņai (likmes uzliktas) laimes ratam 4. Funkcija nogaida 2 sekundes 5. Tiek ģenerēts gadījumskaits 6. Funkcija aizsūta pieprasījumu spēles fāzes nomaiņai (laimes rata griežšanās fāze) laimes ratam 	

7. Funkcija nogaida 5 sekundes 8. Funkcija aizsūta pieprasījumu spēles fazes nomaiņai (likmju rezultātu fāze) laimes ratam 9. Funkcijas nogaida 3 sekundes 10. Funkcija aizsūta pieprasījumu spēles fazes nomaiņai (spēles kārtas beigu fāze) laimes ratam
Izvaddati
-

Funkcija “Laimes rata laimīga cipara ģenerēšana”

Nosaukums	Spēles rata laimīga cipara ģenerēšana
Identifikators	GE004
Apraksts	
Funkcija atļauj aprēķināt likmes koeficientu	
Ievaddati	
1. Nav	
Apstrāde	
1. Funkcija ģenerē gadījumskaitli no 1 līdz 100	
Izvaddati	
Atgriež skaitli diapazonā no 1 līdz 100	

2.2.7 Konta atlikuma modulis



Attēls 2.8. Konta atlikuma moduļa 2. līmeņa datu plūsmas diagramma

Funkcija “Papildināt konta atlikumu”

Nosaukums	Papildināt konta atlikumu
Identifikators	BN001
Apraksts	
Funkcija atļauj lietotājam likt likmi spēles kārtā	
Ievaddati	
1. Lietotāja sesijas dati – automātiski aizpildīts lauks 2. Atlikuma daudzums – obligāts lauks	
Apstrāde	

<ol style="list-style-type: none"> 1. Pārabuda, vai lietotājs eksistē 2. Datubāzē nomaina lietotāja konta atlikuma daudzumu
Izvaddati
Likmju nodaļā lietotājam pazūd dzēstā likme.

Funkcija “Konta atlikuma spēle”

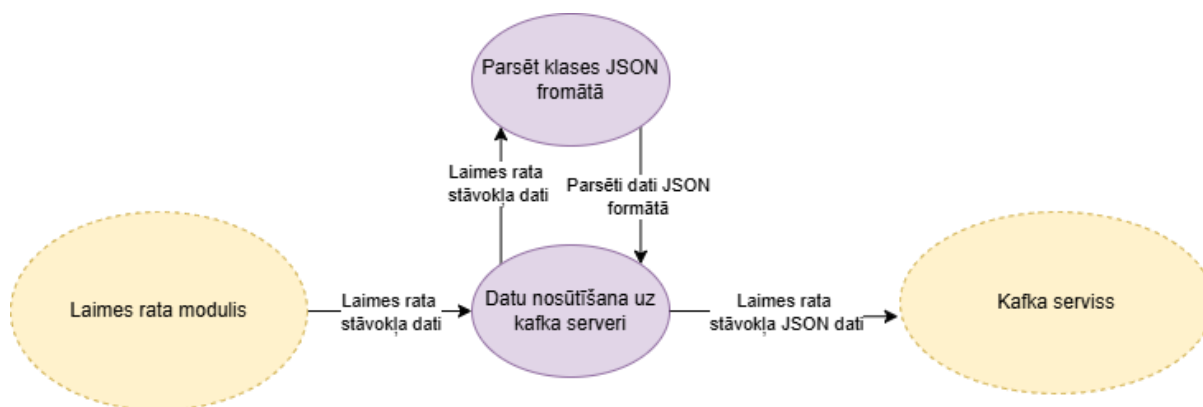
Nosaukums	Konta atlikuma spēle
Identifikators	BN002
Apraksts	
Funkcija atļauj lietotājam likt likmi spēles kārtā (tiks implementēta kā react komponente)	
Ievaddati	
<ol style="list-style-type: none"> 1. Lietotāja sesijas dati – automātiski aizpildīts lauks 2. Depozīta summa – obligāts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. Pārbauda, vai laiks ir no 12:00 līdz 12:30 2. Pārbauda vai depozīta skaitlis ir skaitli lielāks par 0 3. Tiek aprēķināts depozīta koeficients balstoties uz depozīta summu 4. Lietotājam tiek attēlotas 3 kārtis 5. Balstoties uz lietotāja izvēli lietotājam tiek papildināts konta atlikums, ja tiek izvēlēta veiksmīgā kārts 	
Izvaddati	

Lietotājam parāda vai konts tika papildināts

Kļūdu paziņojumi

1. “Deposit amount must be number greater than 0”

2.2.8 Kafka modulis



Attēls 2.9. Spēles dzinēja 2. līmeņa datu plūsmas diagramma

Funkcija “Parsēt klases JSON formātā”

Nosaukums	Parsēt klases JSON formātā
Identifikators	KFK001
Apraksts	
Funkcija ļauj algebrisku datu tipa konvertēt JSON datos	
Ievaddati	
<ol style="list-style-type: none"> 1. Klase, kurai ir pieejams JSON kodētājs – obligāts lauks 	
Apstrāde	

1. Pārveido klasi, izmantojot daļējo kodētāju, JSON datos
Izvaddati
Klase JSON formātā

Funkcija “**Datu nosūtīšana uz Kafka severi**”

Nosaukums	Datu nosūtīšana uz Kafka severi
Identifikators	KFK002
Apraksts	
Funkcija atļauj nosūtīt JSON datus uz Kafka serveri	
Ievaddati	
<ol style="list-style-type: none"> 1. JSON dati – automātiski aizpildīts lauks 2. Kafka producenta dati – automātiski aizpildīts lauks 3. Kafka tēmas nosaukums – automātiski aizpildīts lauks 	
Apstrāde	
<ol style="list-style-type: none"> 1. JSON datus nosūta uz kafka servisu 	
Izvaddati	
Likmju nodaļā lietotājam pazūd dzēstā likme.	

2.3 Nefunkcionālās prasības

2.3.1 Pieejamība

Sistēmai jābūt spējīgai nodrošināt pieejamību visiem lietotājiem, kas ir cik iespējams tuvu 100% līmenim.

2.3.2 Lietojamība

Sistēma atbalsta visa veida operētājsistēmas pārlūkprogrammas (Chrome, Firefox, Safari u.c) un dažādu datoru ekrānu izmērus.

2.3.3 Veiktspēja

Sistēmai jāuztur, jāglabā datus par 10000 lietotājiem. Sistēmai jānodrošina vienlaicīgu darbību 1000 lietotājiem. Sistēmai ir jānodrošina saskarnes nesastingšana pie intensīvām darbībām, kā arī ātri atbildēt lietotājam uz pieprasījumu (līdz 5s).

2.3.4 Uzturamība

Programmatūras projekta arhitektūra nodrošina drošu sistēmas komponentu atjaunināšanu un uzturēšanu.

2.3.5 Drošība

Sistēmai ir jānodrošina datu aizsardzība, lai lietotāju sensitīva informācija būtu drošībā un netiktu 3.personu rokās. Lietotāji var apskatīt savus datus. Administratori var redzēt lietotāju datus.

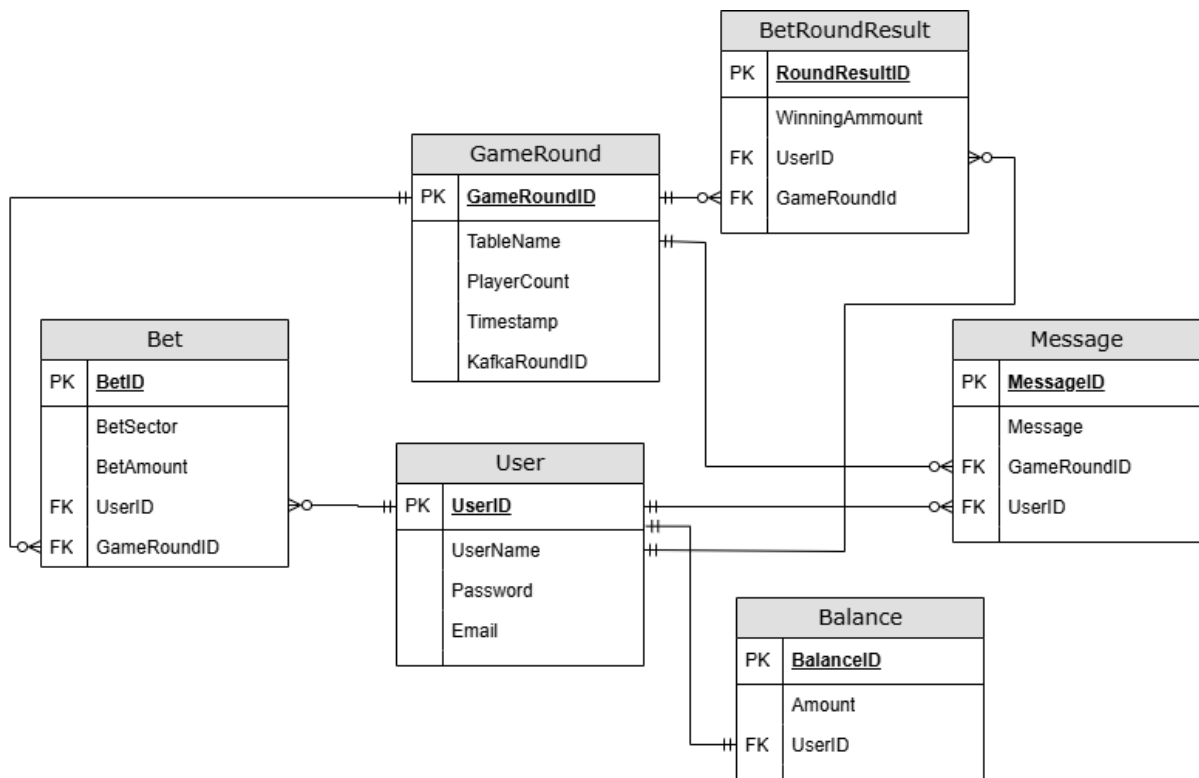
2.3.6 Mērogojamība

Sistēmai ir jānodrošina nepārtrauktu piekļuvi un veiktspēju pieaugošam lietotāju skaitam. Sistēmas arhitektūra ļauj viegli mainīt veiktspējas parametrus, kas nodrošinās efektīvu mērogojamību.

3. PROGRAMMATŪRAS PROJEKTĒJUMA APRAKSTS

3.1. Datubāzes projektējums

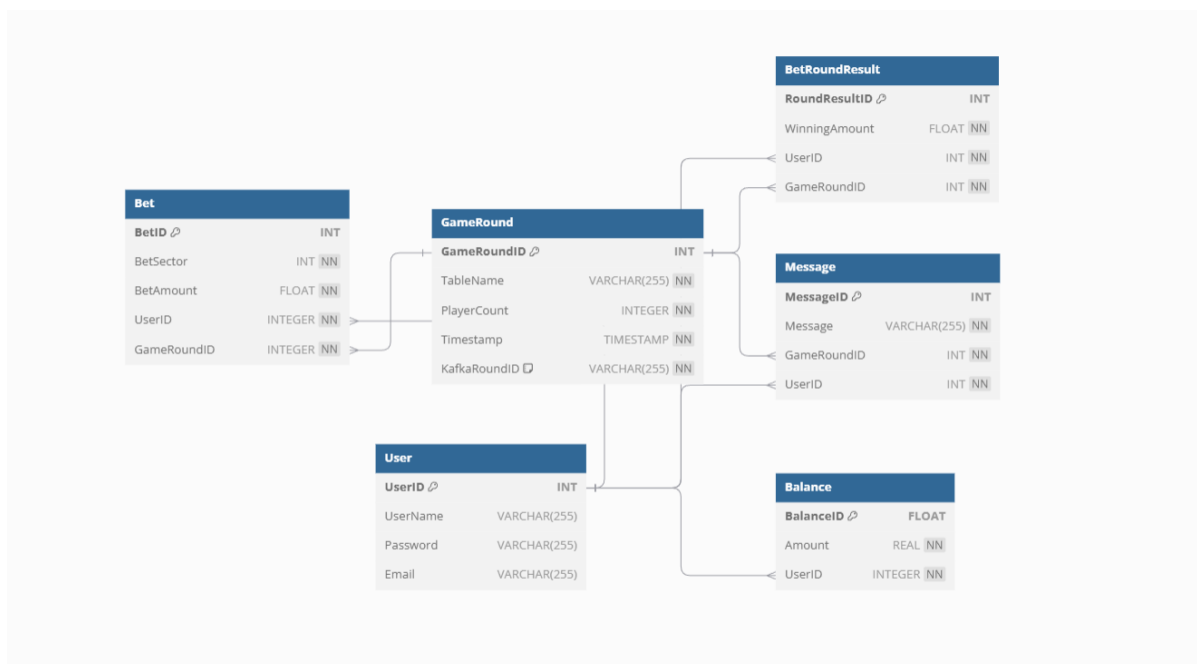
3.1.1. Datubāzes loģiskais modelis



Attēls 3.1. Datubāzes loģiskais modelis

Attēlā 3.1. ir redzams datubāzes loģiskais modelis, veidots “vārnu kāju” sintaksē balsotties uz attēlā 2.1 attēloto konceptuālo modeli un citiem iepriekš dokumentā minētiem faktiem. Pārveidojot no konceptuālā uz loģisko modeli, entītijas tika pārveidotas par tabulām, un tām tiek parādītas ID vērtības, PK un FK.

3.1.2 Datubāzes fiziskais modelis



Attēls 3.2. Datubāzes fiziskais modelis

Attēlā 3.2. attēlots datubāzes fiziskais modelis, kas satur 6 tabulas un ir projektēts atbilstoši konceptuālajam un loģiskajam modelim, izmantojot Apache Derby Sql.

3.1.3. Datubāzes tabulu apraksti

Tabula 3.1 User

Nosaukums	Datu tips	Apraksts	Pieļauj NULL vērtību	Atslēga
UserID	Int	Automātiski ģenerēts unikāls lietotāja identifikators	Nē	PK
UserName	Varchar(255)	Lietotāja vārds	Jā	
Password	Varchar(255)	Lietotāja parole	Jā	
Email	Varchar(255)	Lietotāja ēpasts	Jā	

Tabula 3.2 Bet

Nosaukums	Datu tips	Apraksts	Pieļauj NULL vērtību	Atslēga
BetID	Int	Automātiski ģenerēts unikāls likmes identifikators	Nē	PK
BetSector	Int	Likmes sektors	Nē	
BetAmount	Float	Likmes summa	Nē	
UserID	Int	Lietotāja identifikators	Nē	FK
GameRoundID	Int	Spēles kārtas identifikators	Nē	FK

Tabula 3.3 GameRound

Nosaukums	Datu tips	Apraksts	Pieļauj NULL vērtību	Atslēga
GameRoundID	Int	Automātiski ģenerēts unikāls spēles kārtas identifikators	Nē	PK
TableName	Varchar(255)	Spēles rata nosaukums	Nē	
PlayerCount	Int	Spēlētāju skaits	Nē	
Timestamp	Timestamp	Spēles kārtas datums	Nē	
KafkaRoundID	Varchar(255)	Kafka spēlēs kārtas identifikators	Nē	

Tabula 3.4 BetRoundResult

Nosaukums	Datu tips	Apraksts	Pieļauj NULL vērtību	Atslēga
RoundResultID	Int	Automātiski ģenerēts unikāls spēles kārtas rezultāta identifikators	Nē	PK
Winning ammount	Float	Lietotāja uzvarētā/zaudētā summa	Nē	
UserID	Int	Lietotāja identifikators	Nē	FK
GameRoundID	Int	Spēles kārtas identifikators	Nē	FK

Tabula 3.5 Message

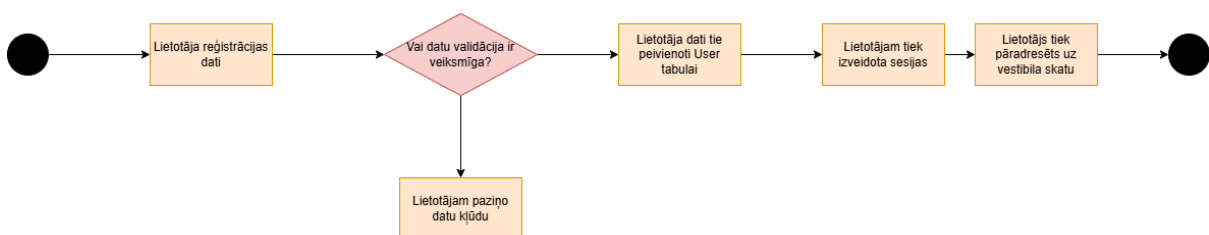
Nosaukums	Datu tips	Apraksts	Pieļauj NULL vērtību	Atslēga
MessageID	Int	Automātiski ģenerēts unikāls ziņas identifikators	Nē	PK
Message	Varchar(255)	Lietotāja ziņa	Nē	
GameRoundID	Int	Spēles kārtas identifikators	Nē	FK
UserID	Int	Lietotāja identifikators	Nē	FK

Tabula 3.6 Message

Nosaukums	Datu tips	Apraksts	Pieļauj NULL vērtību	Atslēga
-----------	-----------	----------	-------------------------	---------

BalanceID	Int	Automātiski ģenerēts unikāls konta atlikuma identifikators	Nē	PK
Amount	Float	Atlikums	Nē	
UserID	Int	Lietotāja identifikators	Nē	FK

3.2. Daļējs funkciju projektējums



Attēls 3.3. Aktivitāšu diagramma funkcijai “Izveidot lietotāja kontu”

3.2.1. Projektējums funkcijai “Izveidot lietotāja kontu”

ID: USER001
Ievaddati <ul style="list-style-type: none"> Lietotājvārds – string Ē-pasts – string Parole – string Atkārtotā parole - string
Obligātie lauki <ul style="list-style-type: none"> Lietotājvārds Ē-pasts Parole Atkārtotā parole
Citas pārbaudes: <ul style="list-style-type: none"> Ievadītais ēpasts atbilst ē-pasta formātam Parole un atkārtotā parole ir vienādas Paroles minimālais garums – 8 simboli

<ul style="list-style-type: none"> • Lietotāja vārds un ēpasts ir unikāli lauki (nav sakrītošo ierakstu User tabulā) • Lietotāja vārds atbilst formātam (nav atstarpju un īpašo simbolu , / . u.c) (izmanojot Regex)
<p>Apstrāde</p> <ul style="list-style-type: none"> • Lietotāja dati tiek ierakstīti datubāzē (tabulā User) • Lietotājam tiek izveidota sessija (tiek izsaukta funkcija <i>pieteikšanās sistēmā</i>) • Lietotājs tiek novirzīts uz “/lobby” skatu (navigate(“/lobby”))

3.2.2. Projektējums funkcijai “Dzēst lietotāja kontu”

ID: USER002
<p>Ievaddati</p> <ul style="list-style-type: none"> • Sesijas dati – klase Player • Parole - string
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Lietotāja sesijas dati • Parole
<p>Citas pārbaudes:</p> <ul style="list-style-type: none"> • Ievadītā parole sakrīt ar tabulā User esošo
<p>Apstrāde</p> <ul style="list-style-type: none"> • Lietotāja dati (UserName , Email, Password) tabulā tiek User nomainīti uz Null (loģiskā datu dzēšana) • Tiek izsaukta funkcija “Atteikšanās no sistēmas”

3.2.3. Projektējums funkcijai “Pieteikšanās sistēmā”

ID: USER003
<p>Ievaddati</p> <ul style="list-style-type: none"> • Lietotājvārds – string • Parole - string
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Lietotājvārds • Parole

<p>Citas pārbaudes:</p> <ul style="list-style-type: none"> • Lietotājvārds eksistē datubāzē • Ievadītā parole sakrīt ar datubāzē esošo
<p>Apstrāde</p> <ul style="list-style-type: none"> • Lietotājam izveido jaunu actor modeli – ActorRef • Lietotājam izveido sesijas datu klasi - Player • Lietotāju pievieno tiešsaistes spēlētāju datu struktūrai ref[IO, Map[Player[Option[ActorRef]]]] • Lietotājs tiek pāradresēts uz “/lobby” skatu

3.2.4. Projektējums funkcijai “Atteikšanās no sistēmas”

ID: USER004
<p>Ievaddati</p> <ul style="list-style-type: none"> • Lietotāja sesijas dati – Player
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Lietotāja sesijas dati
Citas pārbaudes: Nav
<p>Apstrāde</p> <ul style="list-style-type: none"> • Lietotājam tiek dzēsti sesijas dati, izņemti no ref[IO, Map[Player[Option[ActorRef]]]] datu struktūras un dzēsts actor modelis actorRef • Lietotājam tiek dzēst tā actor modelis – actorRef ! poisonPill • Lietotājs tiek novirzīts uz “/welcome” skatu

3.2.5. Projektējums funkcijai “Rediģēt lietotāja vārdu”

ID: USER005
<p>Ievaddati</p> <ul style="list-style-type: none"> • Jaunais lietotājvārds – string • Lietotāja sesijas dati – Player
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Jaunais lietotājvārds • Lietotāja sesijas dati

Citas pārbaudes:

- Vai jaunais lietotājvārds eksistē datubāzē
- Lietotāja vārds atbilst formātam (nav atstarpju un īpašo simbolu , / . u.c) (izmanojot Regex)

Apstrāde

- Lietotājam tabulā User tiek nomainīts lauks UserName ar jauno lietotājvārdu
- Lietotāja sesijas datos Player lauks playerID : String tiek nomainīts uz jauno lietotājvārdu
- Lietotājam tiek paziņots, ka lietotājvārds tika nomainīts

3.2.6. Projektējums funkcijai “Rediģēt e-pastu”

ID: USER006

Ievaddati

- Jaunais ē-pasts – string
- Lietotāja sesijas dati – Player

Obligātie lauki

- Jaunais ē-pasts
- Lietotāja sesijas dati

Citas pārbaudes:

- Vai jaunais ē-pasts eksistē datubāzē
- Vai ē-pasts atbilst ē-pasta formātam (validācijai izmanto regex)

Apstrāde

- Iziet visas validācijas pārbaudes
- Lietotājam tabulā User tiek nomainīts lauks Email ar jauno ē-pastu
- Lietotājam tiek paziņots, ka lietotājvārds tika nomainīts

3.2.7. Projektējums funkcijai “Nomainīt paroli”

ID: USER007

Ievaddati

- Pašreizējā parole - string
- Jaunā parole – string

<ul style="list-style-type: none"> • Atkārtota jaunā parole - string • Lietotāja sesijas dati – Player
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Pašreizējā parole • Jaunā parole • Atkārtota jaunā parole • Lietotāja sesijas dati
<p>Citas pārbaudes:</p> <ul style="list-style-type: none"> • Pašreizējā parole sakrīt ar datubāzē esošo • Jaunā parole jābūt 8 simbolus gara • Tiek pārbaudīts, vai jaunā un atkārtotā parole sakrīt
<p>Apstrāde</p> <ul style="list-style-type: none"> • Lietotāju datubāzē User tabulā lietotājam nomaina paroles lauku (password) • Lietotājam tiek paziņots, ka parole tika nomainīta

3.2.8. Projektējums funkcijai “Iegūt lietotāja datus”

ID: USER008
<p>Ievaddati</p> <ul style="list-style-type: none"> • Lietotāja sesijas dati – Player
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Lietotāja sesijas dati
Citas pārbaudes: Nav
<p>Apstrāde</p> <ul style="list-style-type: none"> • Tiek iegūts lietotāja datubāzes ID (UserID) izmantojot funkciju getPlayerId • Lietotāju datubāzē User tiek atlasīti dati balstoties uz UserID par lietotāju (UserName, Email) • Konta atlikuma datubāzē Balance tiek atlasīts konta atlikuma daudzums (Amount) balstoties uz UserID

3.2.9. Projektējums funkcijai “Nosūtīt e-pastu klientu atbalstam”

ID: USER009

Ievaddati
<ul style="list-style-type: none"> • Lietotāja sesijas dati – Player • Lietotāja ē-pasts – string • Ziņa - string
Obligātie lauki
<ul style="list-style-type: none"> • Lietotāja sessijas dati
Citas pārbaudes:
<ul style="list-style-type: none"> • Pārbauda vai ziņa nav tukša simbolu virkne
Apstrāde
<ul style="list-style-type: none"> • Izveido gmail sesiju – Session.getInstance, kurā notiek autentifikācija • Uztāda ē-pasta datus: message.setSubject, message.setText, message.setFrom, message.setRecipients • Nosūta ziņu klientu atbalstam Transport.send(message) • Paziņo lietotājam par nosūtītu ziņu MessageToPlayer(“email-sent”)

3.2.10. Projektējums funkcijai “Izveidot pieprasījumu pievienoties spēles ratam”

ID: TM001
Ievaddati
<ul style="list-style-type: none"> • Lietotāja sessijas dati – Player • Spēles rata nosaukums – string • Spēlētāju sesijas pārvaldnieks - OnlinePlayerManagerTrait
Obligātie lauki
<ul style="list-style-type: none"> • Lietotāja sesijas dati • Spēles rata nosaukums • Spēlētāju sesijas pārvaldnieks
Citas pārbaudes:
<ul style="list-style-type: none"> • Eksistējošs spēles rata nosaukums • Pārbauda vai spēlētājs jau spēlē kādā no spēles ratiem
Apstrāde
<ul style="list-style-type: none"> • Ja pārbaudes tika veiksmīgi izietas, spēlētājs izmantojot savu actor (player.actorRef) modeli nosūta pieprasījumu (JoinTable) pievienoties spēles ratam

(table) pieprasījumā nododot spēlētāja sesijas, spēles rata actorRef un spēlētāju sesijas pārvaldnieka datus

3.2.11. Projektējums funkcijai “Izveidot privāto spēles ratu”

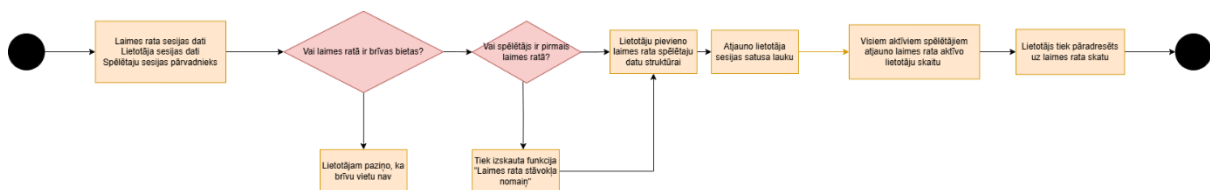
ID: TM002
<p>Ievaddati</p> <ul style="list-style-type: none"> • Spēles rata nosaukums – string • Parole – string • Lietotāja sesijas dati - Player
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Spēles rata nosaukums • Parole • Lietotāja sesijas dati
<p>Citas pārbaudes:</p> <ul style="list-style-type: none"> • Tiek apskatīts vai spēles rats ar tādu nosaukumu eksistē privāto spēles ratu datu struktūrā (privateTableRef datu struktūra)
<p>Apstrāde:</p> <ul style="list-style-type: none"> • Tiek izveidots jauns actor modelis spēles ratam system.actorOf(tableProps, tableName) • Jaunais actor modelis tiek pievienots privāto spēles ratu datu struktūrā - privateTableRef • Tiek izsaukta funkcija “Izveidot pieprasījumu pievienoties spēles ratam”

3.2.12. Projektējums funkcijai “Iziet no spēles rata”

ID: SPW001
<p>Ievaddati</p> <ul style="list-style-type: none"> • Spēles rata sesijas dati – ActorRef • Lietotāja sesijas dati – Player • Spēlētāju sesijas pārvaldnieks - OnlinePlayerManagerTrait
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Spēles rata sesijas dati

<ul style="list-style-type: none"> Lietotāja sesijas dati Spēlētāju sesijas pārvaldnieks
Citas pārbaudes: Nav
<p>Apstrāde:</p> <ul style="list-style-type: none"> Lietotāja sesijas datus (Player) izņem ārā no laimes rata lietotāju datu struktūras (playingPlayer) Visiem aktīviem spēlētājiem atjauno laimes rata aktīvo lietotāju skaitu (broadcastAvailability()) Tiešsaistes lietotāju datu struktūrā lietotājam atjauno laimes rata statusa lauku (player -> None) Lietotājs tiek pāradresēts uz “lobby” skatu

3.2.13. Projektējums funkcijai “Pievienoties laimes ratam”



Attēls 3.4. Aktivitāšu diagramma funkcijai “Pievienoties laimes ratam”

ID: SPW002
<p>Ievaddati</p> <ul style="list-style-type: none"> Spēles rata sesijas dati – ActorRef Lietotāja sesijas dati – Player Spēlētāju sesijas pārvaldnieks - OnlinePlayerManagerTrait
<p>Obligātie lauki</p> <ul style="list-style-type: none"> Spēles rata sesijas dati Lietotāja sesijas dati Spēlētāju sesijas pārvaldnieks
<p>Citas pārbaudes:</p> <ul style="list-style-type: none"> Pārbauda vai laimes ratā ir pieejamas brīvas vietas Ja lietotājs ir pirmais laimes ratā, tad tiek izsaukta funkcija “Laiemes rata stāvokļa nomaiņa”

Apstrāde:

- Lietotāja sesijas datus (Player) pievieno laimes rata lietotāju datu struktūras (playingPlayer)
- Tiešsaites lietotāju datu struktūrā lietotājam atjauno laimes rata statusa lauku (player -> Some(actorRef))
- Visiem aktīviem spēlētājiem atjauno laimes rata aktīvo lietotāju skaitu (broadcastAvailability())
- Lietotājs tiek pāradresēts uz “table” skatu

3.2.14. Projektējums funkcijai “Uzzināt spēles rata pieejamību”

ID: SPW003
Ievaddati <ul style="list-style-type: none">• Spēles rata sesijas dati – ActorRef
Obligātie lauki <ul style="list-style-type: none">• Spēles rata sesijas dati
Citas pārbaudes: Nav
Apstrāde: <ul style="list-style-type: none">• Tiek aprēķinātas brīvas vietas atņemot no 100 tiešaites laimes rata lietotāju datu struktūras izmēru (100 - playingPlayers.size)• Rezultāts tiek nosūtīts lietotājam

3.2.15. Projektējums funkcijai “Nosūtīt ziņu tērzētavā”

ID: SPW004
Ievaddati <ul style="list-style-type: none">• Spēles rata sesijas dati – ActorRef• Lietotāja sesijas dati – Player• Ziņa - string
Obligātie lauki <ul style="list-style-type: none">• Spēles rata sesijas dati• Lietotāja sesijas dati• Ziņa

<p>Citas pārbaudes:</p> <ul style="list-style-type: none"> • Tiek pārbaudīts vai ziņa nav tukša simbolu virkne
<p>Apstrāde:</p> <ul style="list-style-type: none"> • Ziņas dati tiek pievienoti tabulā Message (Message, UserID, GameRoundID) • Izmantojot tiešsaistes laimes rata spēlētāju datu struktūru katram spēlētājam tiek aizsūtīta lietotāja ziņa, izmantojot katra spēlētāja sesijas datus

3.2.16. Projektējums funkcijai “Pievienot likmi”

ID: SPW005
<p>Ievaddati</p> <ul style="list-style-type: none"> • Spēles rata sesijas dati – ActorRef • Lietotāja sesijas dati – Player • Likmes sektors – Int • Daudzums – float • Konta atlikuma daudzums – float
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Spēles rata sesijas dati • Lietotāja sesijas dati • Likmes sektors – int • Daudzums – float • Konta atlikuma daudzums
<p>Citas pārbaudes:</p> <ul style="list-style-type: none"> • Tiek pārbaudīts vai likmes sektors ir skaitlis no 1 līdz 100 • Tiek pārbaudīts vai daudzums ir skaitlis lielāks par 0 • Tiek pārbaudīts vai daudzums ir lielāks par konta atlikuma daudzumu
<p>Apstrāde:</p> <ul style="list-style-type: none"> • Likmei tiek izveidots likmes objekts Bet • Likme tiek pievienota laimes rata likmju datu struktūrai (tableOfBets) • Lietotājam tiek paziņots, ka likme tika veiksmīgi uzlikta

3.2.17. Projektējums funkcijai “Dzēst likmi”

ID: SPW006
Ievaddati <ul style="list-style-type: none">• Spēles rata sesijas dati – ActorRef• Lietotāja sesijas dati – Player• Likmes sektors – Int
Obligātie lauki <ul style="list-style-type: none">• Spēles rata sesijas dati• Lietotāja sesijas dati• Likmes sektors
Citas pārbaudes: <ul style="list-style-type: none">• Tiek pārbaudīts vai likmes sektors ir skaitlis no 1 līdz 100• Tiek pārbaudīts vai lietotājam eksistē likme ar doto sektoru• Tiek pārbaudīts vai laimes rata kārtas fāze ir likmju fāze
Apstrāde: <ul style="list-style-type: none">• Likme tiek noņemta no laimes rata likmju datu struktūras (tableOfBets)• Lietotājam tiek paziņots, ka likme tika veiksmīgi dzēsta

3.2.18. Projektējums funkcijai “Uzstādīt spēles fāzi – likmju fāze”

ID: SPW007
Ievaddati <ul style="list-style-type: none">• Spēles rata sesijas dati – ActorRef
Obligātie lauki <ul style="list-style-type: none">• Spēles rata sesijas dati
Citas pārbaudes: Nav
Apstrāde: <ul style="list-style-type: none">• Tiek nomainīta laimes rata fāze roundState = BetsStartState• Spēles rata kārtai tiek ģenerēts jauns identifikators (roundId = new Id)• Tiek nosūtīti dati uz kafka servisu izmantojot “Datu nosūtīšana uz Kafka severi” funkciju

3.2.19. Projektējums funkcijai “Uzstādīt spēles fāzi – likmes uzliktas”

ID: SPW008
Ievaddati <ul style="list-style-type: none">• Spēles rata sesijas dati – ActorRef
Obligātie lauki <ul style="list-style-type: none">• Spēles rata sesijas dati
Citas pārbaudes: Nav
Apstrāde: <ul style="list-style-type: none">• Laimes rata stāvoklis tiek nomainīts (roundState = BetsEndState)• Visas likmes kuras ir laimes rata likmju datu struktūrā (tableOfBets) ieraksta likmes datubāzes tabulā (Bet)

3.2.20. Projektējums funkcijai “Uzstādīt spēles fāzi – laimes rata griežšanās fāze”

ID: SPW009
Ievaddati <ul style="list-style-type: none">• Spēles rata sesijas dati – ActorRef• Uzvaras skaits – Int
Obligātie lauki <ul style="list-style-type: none">• Spēles rata sesijas dati• Uzvaras skaits
Citas pārbaudes: Nav
Apstrāde: <ul style="list-style-type: none">• Laimes rata stāvoklis tiek nomainīts (roundState = GameStartState)

3.2.21. Projektējums funkcijai “Uzstādīt spēles fāzi – likmju rezultātu fāze”

ID: SPW010
Ievaddati <ul style="list-style-type: none">• Spēles rata sesijas dati – ActorRef• Uzvaras skaits – Int

Obligātie lauki
<ul style="list-style-type: none"> • Spēles rata sesijas dati • Uzvaras skaits
Citas pārbaudes: Nav
<p>Apstrāde:</p> <ul style="list-style-type: none"> • Laimes rata stāvoklis tiek nomainīts (roundState = GameResultState) • Katram lietotājam, kurš lika likmes tiek izskaitļota summa, kura tika uzvarēta/zaudēta. (izmanto funkcija “Aprēķināt laimes kārtas lietotāja laimestu”) • Lietotājiem tiek nosūtīta informācija par laimes skaitli un zaudēto/uzvarēto summu. • Tiek nosūtīti dati uz kafka servisu izmantojot “Datu nosūtīšana uz Kafka serveri” funkciju

3.2.22. Projektējums funkcijai “Uzstādīt spēles fāzi – spēles kārtas beigu fāze”

ID: SPW011
<p>Ievaddati</p> <ul style="list-style-type: none"> • Spēles rata sesijas dati – ActorRef
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Spēles rata sesijas dati
Citas pārbaudes: Nav
<p>Apstrāde:</p> <ul style="list-style-type: none"> • Laimes rata stāvoklis tiek nomainīts (roundState = EooundEndState) • Lietotājiem tiek paziņots par spēles kārtas beigām • Ja laimes rata spēlētāju nav un ja tā ir privātais laimes rats self.path.name.startsWith(“Private”), tad šis laimes rats tiek dzēsts un izņemts no privāto laimes ratu datu struktūras (privateTableRef) • Ja laimes ratā ir spēlētāji, tad tiek izsaukta funkcija (“Laimes rata stāvokļa nomaiņa”)

3.2.22. Projektējums funkcijai “Iegūt spēlētāja statistiku”

ID: ST001

Ievaddati
<ul style="list-style-type: none"> Lietotāja vārds - String
Obligātie lauki
<ul style="list-style-type: none"> Lietotāja vārds
Citas pārbaudes:
<ul style="list-style-type: none"> Vai lietotājs ar tādu lietotāja vārdu eksistē User tabulā
Apstrāde:
<ul style="list-style-type: none"> Tiek veikti 6 SQL pieprasījumi (query) Apache Derby datubāzē, kuros kā galvenais atlases atribūts ir UserID Iegūtie pieprasījuma rezultāti tiek pārveidoti vienā simbolu virknē izmantojot interpolāciju un nodoti lietotājam (s"\$ {totalBetsPlayer}")

3.2.23. Projektējums funkcijai “Iegūt laimes rata servisa statistiku”

ID: ST002
Ievaddati: Nav
Obligātie lauki: Nav
Citas pārbaudes: Nav
Apstrāde:
<ul style="list-style-type: none"> Tiek veikti 6 SQL pieprasījumi (query) Apache Derby datubāzē, kuros tiek ņemti vispārīgi dati par tabulām Iegūtie pieprasījuma rezultāti tiek pārveidoti vienā simbolu virknē izmantojot interpolāciju un nodoti lietotājam (s"\$ {totalPlayerCount}")

3.2.24. Projektējums funkcijai “Aprēķināt laimes kārtas lietotāja laimestu”

ID: GE001
Ievaddati
<ul style="list-style-type: none"> Likmju tabula – TableOfBets Spēlētāja sesijas data – actorRef Uzvaras skaits – Int
Obligātie lauki

<ul style="list-style-type: none"> • Likmju tabula • Spēlētāja sesijas data • Uzvaras skaitlis
Citas pārbaudes: Nav
<p>Apstrāde:</p> <ul style="list-style-type: none"> • Ja lietotājs lika likmes spēles kārtā, tad izmantot .map tiek izziets cauri visām lietotāja likmēm un aprēķināta uzvarētā/zaudētā summa. • Ja likmes sektors <code>bet.betCode == winningNum</code>, tad tiek izsaukta funkcija “Aprēķināt likmes koeficientu” un laimests tiek pieskaitīts klāt, ja ne, tad atņemts.

3.2.25. Projektējums funkcijai “Aprēķināt likmes koeficientu”

ID: GE002
<p>Ievaddati</p> <ul style="list-style-type: none"> • Likmes daudzums - float • Uzvaras skaitlis – Int
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Likmju daudzums • Uzvaras skaitlis
Citas pārbaudes: Nav
<p>Apstrāde:</p> <ul style="list-style-type: none"> • Ja uzvaras skaitlis ir 100, tad likmes daudzums * 50 • Ja uzvaras skaitlis ir pāra skaitlis, tad likmes daudzums * 3 • Ja uzvaras skaitlis ir nepāra skaitlis, tad likmes daudzums * 2

3.2.26. Projektējums funkcijai “Spēles rata fāžu kontrolieris”

ID: GE003
<p>Ievaddati</p> <ul style="list-style-type: none"> • Laimes rata sesijas dati - actorRef
<p>Obligātie lauki</p> <ul style="list-style-type: none"> • Laimes rata sesijas dati
Citas pārbaudes: Nav

Apstrāde:

- Funkcija aizsūta pieprasījumu spēles fāzes nomaiņai (likmju fāze) laimes ratam
- Funkcija nogaida 10 sekundes (IO.sleep(10.seconds))
- Funkcija aizsūta pieprasījumu spēles fāzes nomaiņai (likmes uzliktas) laimes ratam
- Funkcija nogaida 2 sekundes (IO.sleep(2.seconds))
- Tiek izsaukta funkcija “Laimes rata laimīga cipara ģenerēšana”
- Funkcija aizsūta pieprasījumu spēles fāzes nomaiņai (laimes rata griežšanās fāze) laimes ratam //(actorRef ! GameStart(number)) tā tiek sūtīts pieprasījums
- Funkcija nogaida 5 sekundes (IO.sleep(5.seconds))
- Funkcija aizsūta pieprasījumu spēles fāzes nomaiņai (likmju rezultātu fāze) laimes ratam
- Funkcijas nogaida 3 sekundes (IO.sleep(3.seconds))
- Funkcija aizsūta pieprasījumu spēles fāzes nomaiņai (spēles kārtas beigu fāze) laimes ratam

3.2.27. Projektējums funkcijai “Spēles rata fāžu kontrolieris”

ID: GE004
Ievaddati: Nav
Obligātie lauki: Nav
Citas pārbaudes: Nav
Apstrāde: <ul style="list-style-type: none">• Funkcija ģenerē gadījumskaitli no 1 līdz 100 un atgriež to

3.2.28. Projektējums funkcijai “Papildināt konta atlikumu”

ID: BN001
Ievaddati: <ul style="list-style-type: none">• Lietotāja sesijas dati – Player• Atlikuma daudzums - Int
Obligātie lauki: <ul style="list-style-type: none">• Lietotāja sesijas dati• Atlikuma daudzums

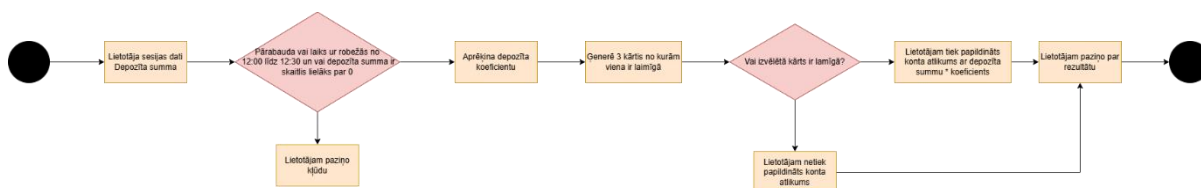
Citas pārbaudes:

- Pārbauda vai lietotājs eksistē User tabulā (datubāzē)

Apstrāde:

- Datubāzes tabulā Balance lietotājam ar UserID tiek nomainīts atlikuma daudzums (Amount)

3.2.29. Projektējums funkcijai “Konta atlikuma spēle (react komponente)”



ID: BN001

Ievaddati:

- Lietotāja sesijas dati – Player
- Depozīta summa – float
- Izvēlēta kārts – event (react.js)

Obligātie lauki:

- Lietotāja sesijas dati
- Depozīta summa
- Izvēlēta kārts

Citas pārbaudes:

- Pārbauda vai lokālais laiks ir no 12:00 līdz 12:30
- Pārbauda vai depozīta summa ir skatlis lielāks par 0

Apstrāde:

- Tiek aprēķināts koeficients balstoties uz depozīta summu
- Lietotājam parādās trīs kārtis (vienai no tām ir ģenerēts laimīgais indekss `setLuckyCard()` `setGameShow(true)` un nomaina spēles stāvokli `setGameOver(false)`)
- Lietotājam izvēloties kārti tiek pārbaudīts: ja tika izvēlēta laimīgā kārts, tad konta atlikums tiek papildināts (tiek izsaukta funkcija “Papildināt konta atlikumu”).
- Lietotājam norāda, vai tika papildināts konta atlikums vai ne

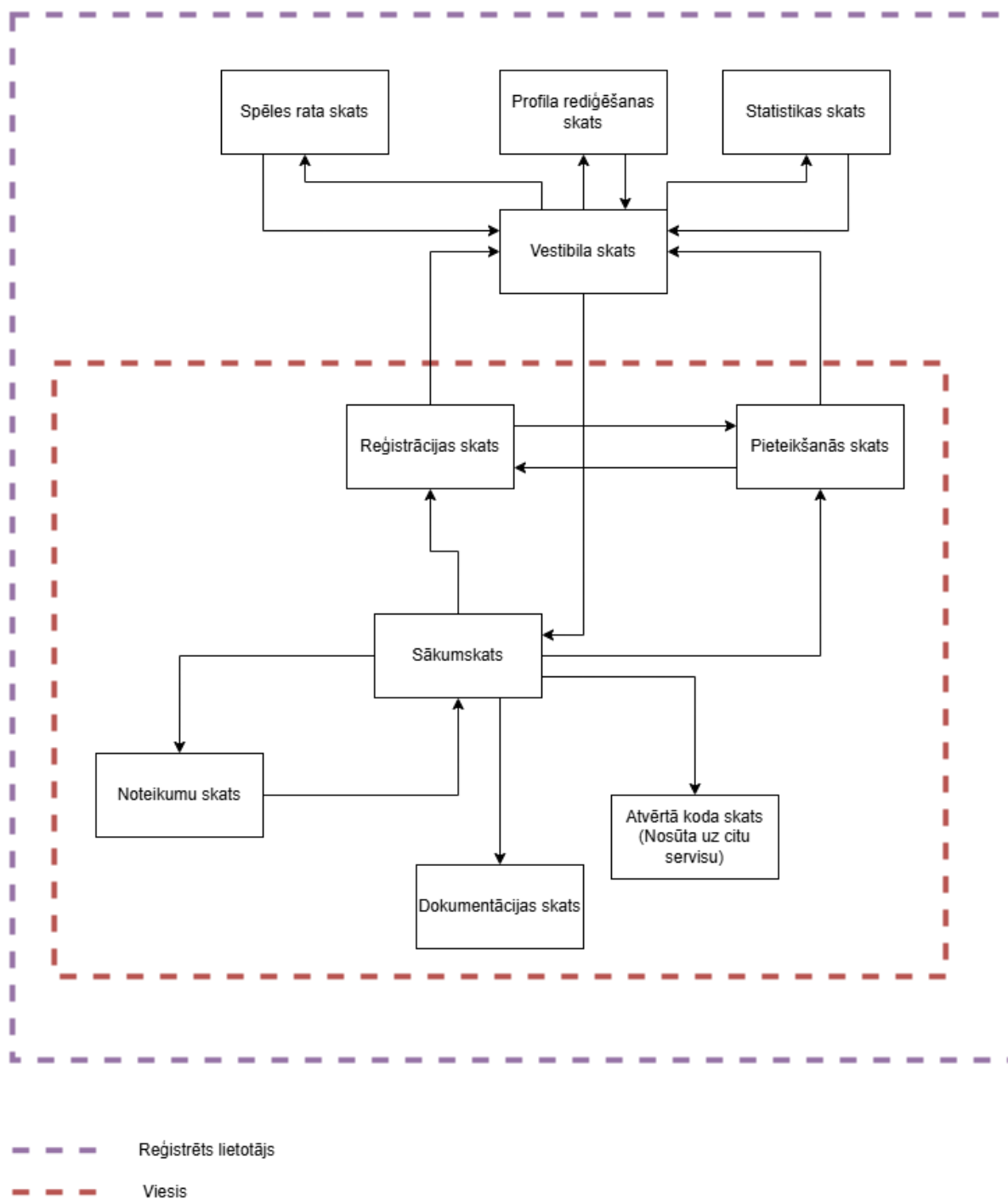
3.2.30. Projektējums funkcijai “Parsēt klases JSON formātā”

ID: KFK001
Ievaddati: <ul style="list-style-type: none">• Klase, kurai ir circe JSON semi-auto kodētājs – scala class
Obligātie lauki: <ul style="list-style-type: none">• Klase
Citas pārbaudes: Nav
Apstrāde: <ul style="list-style-type: none">• Izmantojot circe metodi .asJson klase tiek kodēta JSON formātā

3.2.31. Projektējums funkcijai “Datu nosūtīšana uz Kafka severi”

ID: KFK002
Ievaddati: <ul style="list-style-type: none">• JSON dati – string• Kafka producenta dati – KafkaProducer[String, String]• Kafka tēmas nosaukums – string
Obligātie lauki: <ul style="list-style-type: none">• JSON dati – string• Kafka producenta dati• Kafka tēmas nosaukums
Citas pārbaudes: Nav
Apstrāde: <ul style="list-style-type: none">• JSON datus nosūta Kafka serveri uz pareizu sekciju balstoties uz kafka tēmas nosaukumu (producerSend())

3.3 Daļējs lietotāja saskarņu projektējums

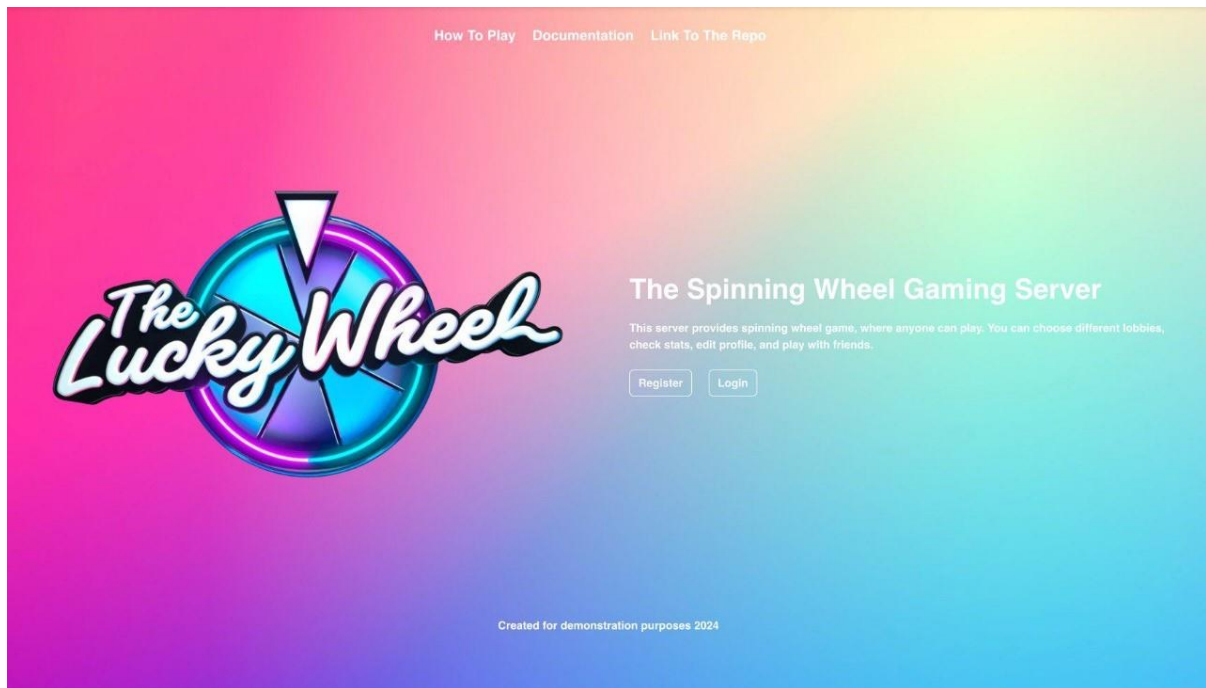


3.3.1. attēls Ekrānplūsmas diagramma visām lietotāju grupām

Laires rata servīsā ir 10 skati. Viesis var iepzīties ar servisa likumiem un apskatīt demonstrējošu video pamācību. Viesis var iepazīties ar servisa dokumentāciju, doties uz atvērtā koda repozitoriju, kā arī reģistrēties vai pieteikties sistēmai. Reģistrētam lietotājam ir

papildus četri skati, kuros ietilpst pašas spēles skats, lietotāja profila datu skats, kā arī statistikas skats.

3.3.1 Sākumskats




Attēls 3.1.2. Sākumskats

Sākumskata augšējā daļā redzama josla kur lietotājam piedāvā apskatīt servera noteikumus, apskatīt projekta dokumentāciju, kā arī iepazīties ar programmatūras kodu.

Skata vidusdaļā lietotājam piedāvā pieteikties sistēmai vai izveidot jaunu profilu.

3.3.2 Reģistrācijas skats




Create an Account

Already have an account? [Login here](#)

Attēls 3.1.3. Reģistrācijas skats

Šajā skatā lietotājam ir iespēja izveidot profile servisā. Veiksmīgas reģistrācijas rezultātā lietotājs dodas uz vestibila skatu.

3.3.3 Pieteikšanās skats



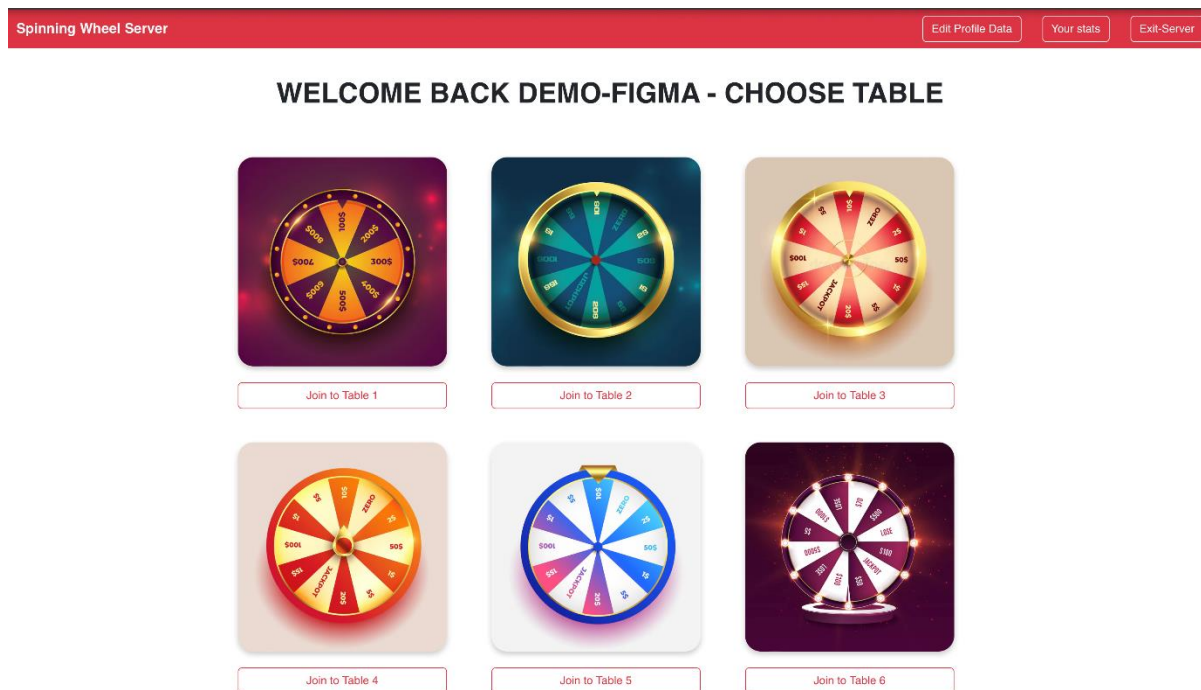
Join the gaming service

Don't have an account? [Make it here](#)

Attēls 3.1.4. Pieteikšanās skats

Šajā skatā lietotājam dota iespēja pietiekties sistēmā. Veiksmīgas reģistrācijas rezultātā lietotājs dodas uz vestibila skatu.

3.3.4 Vestibila skats



Attēls 3.1.5. Vestibila skats

Šis ir vestibila skats, kurā lietotājs var pievienoties publiskajam spēles ratam, apskatīt to pieejamību, kā arī pievienoties vai izveidot privāto spēles ratu.

3.3.5 Spēles rata skats

Spinning Wheel Server

Exit-Table

The Spinning Wheel Game

43444546474849505152535455565758

Control panel of wheel

Add Bet

Choose sector:

Enter bet code

Write bet:

Enter Amount

Submit Bet

State Of Wheel

Round started. Place your bets!

Playing players : 1

Your current balance : \$100.00

Your Round History

Dashboard

List of bets

New round started. Place your bets!

Chat

Attēls 3.1.6. Spēles rata skats

Šajā skatā lietotājs var apskatīt spēles rata stāvokli, likt likmes apskatīt savu spēles kārtu rezultātus, kā arī rakstīt īsziņas tērētavā.

3.3.6 Profila rediģēšanas skats

Spinning Wheel Server

Back To The Lobby

Update User Data

User Profile

Name

Your current name : demo-figma

Update Profile

Email

Your current email : demo-figma@gamil.com

Update Profile

Delete profile

Current password

Delete Profile

Change Password

Current Password

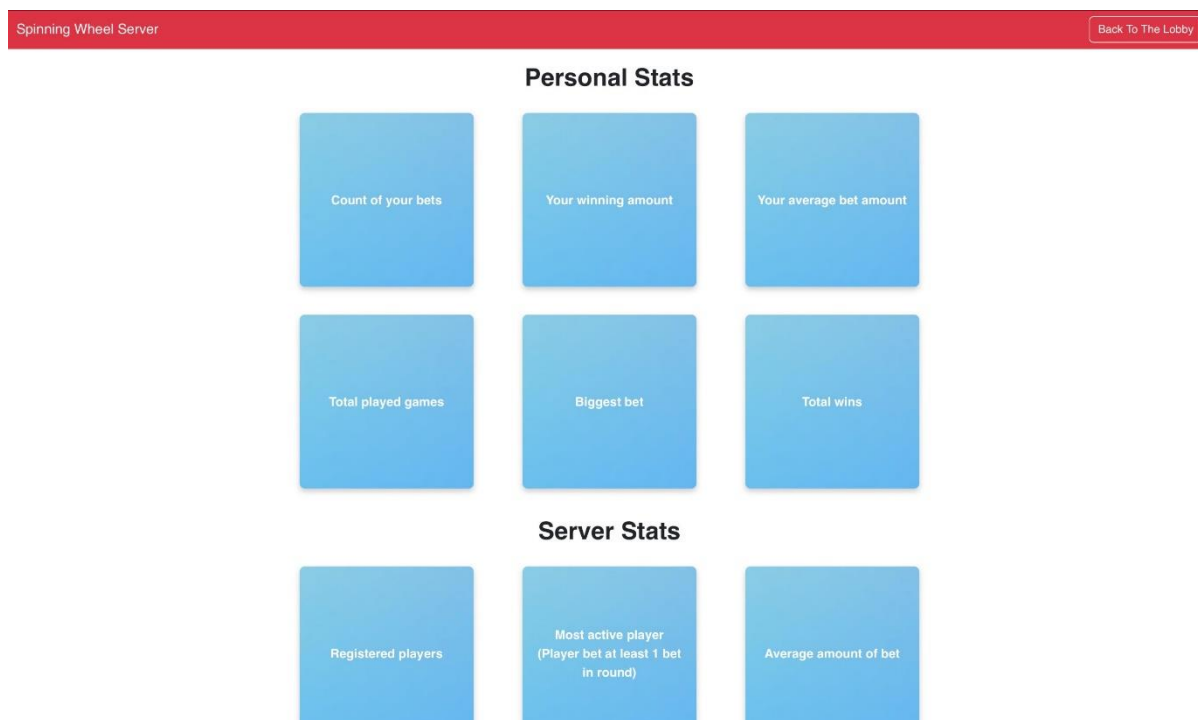
New Password

Confirm New Password

Attēls 3.1.7. Profila rediģēšanas skats

Šajā skatā lietotājam ir iespēja rediģēt savus datus (ē-pastu, lietotājvārdu, paroli), papildināt konta atlikumu nospēlējot to kā spēli, kā arī nosūtīt ē-pasta ziņu klieru atbalsta servisam

3.3.7 Statistikas skats



Attēls 3.1.8. Statistikas skats

Šajā skatā lietotājs var apkatīt savu un laimes rata spēles servisa statistiku, kura tiek atjaunota katru reizi, kad tiek apmeklēts šis skats

4. PROGRAMMATŪRAS TESTĒŠANA

Produkta testēšana tika veikta izstrādes procesā, pārbaudot, vai izveidotā funkcionalitāte atbilst prasībām. Testēšanas dokumentēšana tika veikta sistēmas izstrādes beigās. Tika izmantotas gan manuālās, gan automātiskās testēšanas metodes.

Automātiskā testēšana tika veikta funkcijām kam bija nepieciešami un nebija nepieciešami ievaddati no pārlūkprogrammas. Testēšana tika veikta arī actor un asinhrona tipa funkcijām.

Manuālā testēšana tika veikta funkcijām, kurām bija nepieciešami ievaddati no pārlūkprogrammas

- **Ierīce, uz kuras tika veikta testēšana, konfigurācija:**
 - Ierīce: **Apple Macbook Pro 16**
 - Operētājsistēma: macOS Sonoma
 - Izmantojamās pārlūkprogrammas: Safari, Google Chrome

4.1 Testpiemēri

4.1.1 Lietotāja modulis

4.1.1.1 Manuālie testi

Testa identifikators	Testējamā funkcija	Apraksts	Sagaidāmais rezultāts
TUSER001	USER001	Pārbauda vai var izveidot lietotāju ar netbilstošu formātam lietotāja vārdu	Lietotāja vārds neatbilst formātam
TUSER002	USER001	Pārbauda vai var izveidot lietotāju ar esošu lietotājvārdu	Lietotāja vārds jau eksistē
TUSER003	USER001	Pārbauda vai var izveidot lietotāju ar eksistējošu ē-pastu	Ē-pasts jau eksistē

TUSER004	USER001	Visi ievadītie dati ir korekti	Lietotāja profils tika izveidots un lietotājs tika pāradresēts uz vestibila skatu
TUSER005	USER003	Pārbauda vai var pieteikties sistēmā ar neeksistējošiem lietotāja datiem	Lietotāja vārds vai parole ir nepareizi
TUSER006	USER003	Pārbauda vai var pieteikties sistēmā, ja kāds jau ir pieteicies	Lietotāja vārds vai parole ir nepareizi
TUSER007	USER003	Tiek ievadīti pareizi dati	Lietotājs tiek pāradresēts uz vestibila skatu
TUSER008	USER005	Tiek ievadīts neatbilstošs formātam lietotāja vārds	Lietotāja vārds neatbilst formātam
TUSER009	USER005	Tiek ievadīts eksistējošs lietotāja vārds	Lietotāja vārds jau eksistē
TUSER010	USER005	Tiek ievadīti korekti dati	Lietotāja vārds tika nomainīts
TUSER012	USER006	Tiek ievadīts eksistējošs e-pasts	E-pasts jau eksistē
TUSER012	USER006	Tiek ievadīti korekti dati	E-pasts tika nomainīts
TUSER013	USER007	Tiek ievadīta nepareiza pašreizējā parole	Pašreizējā parole ir nepareiza
TUSER014	USER007	Ievadīti korekti dati	Parole tika nomainīta

TUSER015	USER009	Tiek uzrakstīta korekta ziņa	Ē-pasts tika nosūtīts
TUSER016	USER009	Tiek nosūtīta tukša ziņa	Tukša ziņa nevar būt nosūtīta

4.1.2 Laimes ratu konfigurācijas modulis

4.1.2.1 Manuālie testi

Testa identifikators	Testējamā funkcija	Apraksts	Sagaidāmais rezultāts
TTM001	TM001	Tiek ievadīts neeksistējošs laimes rata nosaukums	Nepareizs laimes rata nosaukums
TTM002	TM001	Tiek ievadīti korekti dati	Tiek nosūtīts pieprasījums pievienoties laimes ratam
TTM003	TM002	Tiek ievadīts eksistējošs laimes rata nosaukums	Laimes rata nosaukums jau eksistē
TTM004	TM002	Tiek ievadīti korekti dati	Laimes rats tika izveidots un lietotājs tiek pāradresēts uz laimes rata skatu

4.1.3 Laimes rata modulis

4.1.3.1 Manuālie testi

Testa identifikators	Testējamā funkcija	Apraksts	Sagaidāmais rezultāts
----------------------	--------------------	----------	-----------------------

TSPW001	SPW002	Meģināt pievienoties neeksistējošam laimes ratam	Laimes rats neeksistē
TSPW002	SPW002	Meģināt pievienoties eksistējošam laimes ratam	Lietotājs tiek pievienots laimes ratam un pāradresēts uz laimes rata skatu
TSPW003	SPW005	Pievieno likmi ar neeksistējošu likmes sektoru	Likmes sektoram jābūt skaitlim no 1 līdz 100
TSPW004	SPW005	Pievieno likmi ar korektiem datiem (likmes sektors 12, likme 5.23)	Lietotājam vizuāli parāda, ka likme tika pievienota
TSPW005	SPW005	Pievieno likmi spēles fāzē – likmju fazes beigās	Likmes var likt tikai likmju fāzē
TSPW006	SPW006	Dzēst likmi lietotājam ar citu sektoru (23)	Likme neeksistē
TSPW007	SPW006	Dzēst likmi ar eksistējošu sektoru	Lietotāja likme tika dzēsta

4.1.4 Statistikas modulis

4.1.4.1 Manuālie testi

Testa identifikators	Testējamā funkcija	Apraksts	Sagaidāmais rezultāts
TST001	ST001	Tiek savākta statistika par tikko reģistrētu lietoāju	Visiem vaicājumiem jābūt 0

TST002	ST002	Tiek ievadīts neeksistējoša lietotāja vārds	Lietotājs neeksistē
--------	-------	---	---------------------

4.1.5 Spēles dzinēja modulis

4.1.5.1 Automātiskie testi

Testa identifikators	Testējamā funkcija	Apraksts	Sagaidāmais rezultāts
TGE001	GE001, GE002	Pārbauda, ja spēlētājam ir kaut viena likme atgriež summu	Summa - Some(50)
TGE002	GE001, GE002	Atgriež korektu summu, ja lietotājam ir uzvarējošā un zaudējošās likmes	Summa – Some(-10)
TGE003	GE001, GE002	Atgriež none, ja lietotājam nav likmju	Summa - None
TGE004	GE001, GE002	Lietotājam ir tikai zaudējošās likmes	Summa – Some(-30)
TGE005	GE001, GE002	Lietotājam ir likme uz 100 sektoru, kura ir uzvarējošā	Summa – Some(2500)
TGE006	G003, GE004	Tiek pārbaudīts vai funkcija aizsūt pieprasījumus spēles ratam par stāvokļa nomaiņu un ģenerē korektu gadījumskaitli	Tika saņemti visi pieprasījumu korektajā laika intervālā

4.1.6 Konta atlikuma modulis

4.1.6.1 Manuālie testi

Testa identifikators	Testējamā funkcija	Apraksts	Sagaidāmais rezultāts
TBN001	BN001	Tiek ievadīts neeksistējošs lietotājs	Lietotāja konta atlikums neeksistē
TBN002	BN001	Tiek ievadīta korekta summa (24.65)	Lietotāja konta atlikums tagad ir 24.65
TBN003	BN002	Dati tiek ievadīti 15:30	Spēle ir pieejama tikai no 12:00 līdz 12:30
TBN004	BN002	Tiek ievadīts skaitlis -5	Depozīta summai jābūt skaitlim lielākam par 0
TBN005	BN002	Tiek ievadīti korekti dati	Lietotājam piedāva izvēlēties kārtis un aprēķina koeficientu

4.2 Testēšanas žurnāls

4.2.1 Lietotāja modulis

4.2.1.1 Manuālie testi

Testa identifikators	Datums	Rezultāts	Komentārs
TUSER001	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER002	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER003	05.01.2024	OK	Izpildās abās pārlūkprogrammās

TUSER004	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER005	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER006	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER007	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER008	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER009	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER010	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER011	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER012	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER013	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER014	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER015	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TUSER016	05.01.2024	OK	Izpildās abās pārlūkprogrammās

4.2.2 Laimes ratu konfigurācijas modulis

4.2.2.1 Manuālie testi

Testa identifikators	Datums	Rezultāts	Komentārs
TTM001	05.01.2024	OK	Izpildās abās pārlūkprogrammās

TTM001	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TTM001	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TTM001	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TTM001	05.01.2024	OK	Izpildās abās pārlūkprogrammās

4.2.3 Laimes rata modulis

4.2.3.1 Manuālie testi

Testa identifikators	Datums	Rezultāts	Komentārs
TSPW001	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TSPW002	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TSPW003	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TSPW004	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TSPW005	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TSPW006	05.01.2024	OK	
TSPW007	05.01.2024	OK	Izpildās abās pārlūkprogrammās

4.2.4 Statistikas modulis

4.2.4.1 Manuālie testi

Testa identifikators	Datums	Rezultāts	Komentārs
TST001	05.01.2024	OK	Izpildās abās pārlūkprogrammās

TST002	05.01.2024	OK	
--------	------------	----	--

4.2.5 Spēles dzinēja modulis

4.2.5.1 Automātiskie testi

Testa identifikators	Datums	Rezultāts	Komentārs
TGE001	05.01.2024	OK	
TGE002	05.01.2024	OK	
TGE003	05.01.2024	OK	
TGE004	05.01.2024	OK	
TGE005	05.01.2024	OK	

4.2.6 Konta atlikuma modulis

4.2.6.1 Manuālie testi

Testa identifikators	Datums	Rezultāts	Komentārs
TBN001	05.01.2024	OK	
TBN002	05.01.2024	OK	
TBN003	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TBN004	05.01.2024	OK	Izpildās abās pārlūkprogrammās
TBN005	05.01.2024	OK	Izpildās abās pārlūkprogrammās

5. PROJEKTA PĀRVALDĪBA

5.1 Projekta organizācija

Darba izstrādē tika izmantota Agile pieeja. Projekta sākumā tika izvirzītas vispārīgas prasības un sākās projekta izstrāde. Projekts tika sadalīts domēnos un katram domēnam jeb modulim turpmāk tika precizēta dokumentācija un tika palielināta funkcionalitāte.

Sākumā tika plānots izveidot serveri, kurā būs tikai viens spēles rats, pie kura varētu pieslēgties spēlētāji, bez reģistrācijas un pārējās funkcionalitātes. Projektam attīstoties tika pievienota dažādu spēles ratu sesijas, tērztava, lietotāja reģistrācija un cita funkcionalitāte. Bija arī sarežģīti posmi, kuros bija jāpieņem svarīgi lēmumi. Piemēram, webscoket servera komunikācijas protokola izstrāde un tam lietderīgas tehnoloģijas. Tika izveidots Kafka modulis, kas atļauj projekta partneriem analizēt spēles rata datus biznesa nolūkos, kā arī samazina risku nepareizu datu veidošanai. Kopumā tas viss palielināja izstrādes termiņu.

Projekts tika izstrādāts prakses ietvaros, programmatūra tika veidota mācību ietvaros. Tika definētas vispārīgas prasības, kuras turpmāk tika papildinātas un precizētas. Programmatūras arhitektūras, dizana, projektējuma un pārējās izstrādes aktivitātes veicu pats.

5.2 Kvalitātes nodrošināšana

Kods ir strukturēts tā, ka tas tika sadalīts dažādos domēnos (komponentēs) balstoties uz biznesa prasībām. Tas nodrošina labu koda pārskatamību, ļauj nošķirt dažādas komponentes, kas atvieglo koda izstrādi un turpmāko testēšanu. Serviss balstās uz asinhrono programmēšanu, lai nodrošinātu spēles funkcionalitāti un korektu saziņu starp lietotāju un serveri. Lai nodrošinātu kvalitatīvu servera asinhrono datu apstrādi tika izmantota akka actor[5] tehnoloģija, kas kontrolē kolīziju rašanos un atļauj veidot dažādus laimes ratus ar vienu funkcionalitāti. Sistēmā arī svarīgās struktūrās tika izmantota variable immutability tehnika (šī servisa gadījumā tika izmanta cats effect bibliotēka), kas nodrošina datu konsistenci un dažādu komponentu stāvokļu neatkarību. Servera daļas sistēmas kods ir balstīts uz funkcionālo programmēšanu.

Izstrādes procesā tika veikta testēšana, lai pārliecinātos, ka lietotne atbilst izvirzītajām prasībām. Vienkāršākās funkcijas tika testētas ar automatizētiem testiem, savukārt pārējās – manuāli. Pēc katras izstrādātās funkcionalitātes tika veikti vienībtesti, kuros tika identificētas dažādas kļūmes. Detalizēts testēšanas apraksts tika izveidots izstrādes noslēgumā. Testēšanas rezultāti norādīja, ka lietotne atbilst prasībām.

5.3 Konfigurāciju pārvaldība

Programmatūras konfigurācijas pārvaldībai tika izmantots Git. Tika izveidots GitHub repozitorijs[6], kurš nodrošināja versiju kontroli. Katru reizi pēc jauna moduļa izstrādes un testēšanas kods tika veidots jauns git zars, kuru apskatīja darba vadītājs. Veiksmīga risinājuma gadījumā kods tika pievienots galvenjam zaram.

Dokumentācija tika glabāta google docs mākonī, kurš nodrošināja dokumenta pieejamību dažādiem lietotājiem un iespēju veidot dokumenta versijas. Diagramu izstrādē tika izmantots rīks – draw.io[3], kurš lokāli veidoja datnes versijas. Saskarnes izstādei tika izmantota figma[7].

5.4 Darbietilpības novērtējums

Darbietilpības novērtēšanas laikā tika ņemta vērā arī pieredze dokumentācijas veidošanā programminženierijas kursā kā arī pieredze git versiju kontrolē. Lai aprēķinātu darbietilpību, tika izmantota 3 punktu metode:

$$\underline{pesimistiskais + optimistiskais + 4 \cdot reālistiskais}$$

6

Tabula 5.1. Darbietilpības novērtējuma tabula (dienās)

Novērtējamais	Optimistiskais novērtējums	Reālistiskais novērtējums	Pesimistiskais novērtējums	Rezultējošais novērtējums
Lietotāja modulis	5	7	9	7
Laires ratu konfigurācijas modulis	2	3	5	3,17
Laires rata modulis	7	9	12	9,17
Statistikas modulis	2	3	4	3
Spēles dzinēja modulis	4	5	6	5

Konta atlikuma modulis	2	3	4	3
Kafka modulis	1	2	3	2
Dokumentācija	18	23	25	22,5
Saskarne	8	10	11	9,83
Kopā	49	63	79	63,33

Tabula 5.1. redzams darbietilpības novērtējums sadalīts pa daļām. Kopumā darbietilpības novērtējums ir 63,33 dienas, kas, konvertējot uz cilvēkmēnešiem – 1 cilvēkmēnesī ir 18.33 dienas - sanāk 3.46 cilvēkmēneši.

Vēl var pārbaudīt projekta darbietilpību ar citām metrikām, piemēram, izmantojot QSM novērtējuma tabulas [4].

Kodā ir aptuveni 4122 rindiņas, no kurām aptuveni 300 ir import rindiņas – automātiski ģenerētas. Tātad kopumā ir aptuveni 3822 paša rakstītas rindiņas.

Biznesa sistēmu implementāciju tabulas datos ir minēts, ka vidēji 3.2 mēnešu ilgs projekts sastāv no 1889 koda rindiņām. Tā kā kodā ir divas reizes vairāk rindiņu, tad var noteikti secināt, ka pēc QSM tabulu datiem, šī projekta apjoms pārsniedz 3.2 mēnešu ilgu projektu.

NOBEIGUMS

Darbam ir pievienoti 3 pielikumi ar lietotnes pirmkoda daļām.

1. Pielikums 1 - Konta atlikuma spēle (react komponente) (BN002)
2. Pielikums 2 - – Laimes rata stāvokļa nomaina (GE003) un laimes rata laimīga cipara ģenerēšanas (GE004) funkcija
3. Pielikums 3 - Spēlētāju sesijas pārvaldnieks (PlayerManager), kurā ietilpst lietotāja moduļa funkcijas

Tā kā dokuments un programmatūra tika izstrādās vienlaicīgi, bija biežāk jāmaina kods vai dokumentācija, taču arī bija skaidrākas prasības, ko var izstrādāt.

Darba izstrādes gaitā tika apgūtas jaunas tehnoloģijas – scala programmēšanas valoda, akka tehnoloģiju klāsts (akka actor, akka https), reacts.js programmēšanas valoda, kā arī kafka tehnoloģiju saime.

Lietotne ir izstrādāta – tā atbilst prasībām un tā ir lietojama, taču, lai būt konkurentsējīga reālā tirgu ir nepieciešama papildu funkcionalitāte un uzlabots saskarnes dizains.

IZMANTOTĀ LITERATŪRA UN AVOTI

- [1] LVS 68:1996 “Programmatūras prasību specifikācijas ceļvedis”, Latvijas Nacionālais standartizācijas un metroloģijas centrs, 1996.
- [2] LVS 72:1996 “Ieteicamā prakse programmatūras projektējuma aprakstīšanai”, Latvijas Nacionālais standartizācijas un metroloģijas centrs, 1996.
- [3] JGraph Ltd, “draw.io” [Tiešsaiste]. Available: <https://app.diagrams.net/>
- [4] QSM, “QSM Benchmark tables” [Tiešsaiste]. Available: <https://www.qsm.com/resources/qsm-benchmark-tables>
- [5] Akka, “Akka technologies” [Tiešsaiste]. Available: <https://akka.io/>
- [6] GitHub, “Laimis rata spēles servisa atvērtais pirmkods” [Tiešsaiste]. Available: <https://github.com/raimondseisaks/SpinningWheelService>
- [7] Figma, «Figma design tool» [Tiešsaiste]. Available: <https://www.figma.com/>
- [8] Apache Kafka, “Apache Kafka documentation” [Tiešsaiste]. Available: <https://kafka.apache.org/documentation/>

PIELIKUMI

Pielikums 1 - Konta atlikuma spēle (react komponente) (BN002)

```

1 import React, { useState } from "react";
2 import { useWebSocket } from "../WebSocketHandler";
3 import "../BalanceGameStyle.css";
4
5 const BalanceContainer = () => { Show usages  ⚡ raimondseisaks
6   const { userData, sendMessage } = useWebSocket();
7   const [showGame : boolean, setShowGame] = useState( initialState: false);
8   const [depositAmount : string, setDepositAmount] = useState( initialState: "");
9   const [bonusMessage : string, setBonusMessage] = useState( initialState: "");
10  const [luckyCard, setLuckyCard] = useState( initialState: null);
11  const [timeRestrictedMessage : string, setTimeRestrictedMessage] = useState( initialState: "");
12  const [multiplier : number, setMultiplier] = useState( initialState: 0);
13  const [gameOver : boolean, setGameOver] = useState( initialState: false);
14
15  // Function to check if the current time is within the allowed time range (12:00 PM to 12:30 PM)
16  const isWithinAllowedTime = () => {...};
17
18  // Function to calculate multiplier based on deposit
19  const calculateMultiplier = (deposit) : number => { Show usages  ⚡ raimondseisaks
20    const cappedDeposit : number = Math.min(deposit, 25); // Cap the deposit at 25
21    const multiplierValue : number = 3 - (cappedDeposit / 25) * 3;
22    return parseFloat(multiplierValue.toFixed( fractionDigits: 2));
23  };
24
25  // Function to start the mini-game
26  const startMiniGame = (e) : void => { Show usages  ⚡ raimondseisaks
27    e.preventDefault();
28    if (!isWithinAllowedTime()) {
29      setTimeRestrictedMessage( value: "The game is only available between 12:00 PM and 12:30 PM.");
30      return;
31    } else {
32      setTimeRestrictedMessage( value: "");
33    }
34
35    if (depositAmount && parseFloat(depositAmount) > 0) {
36      const multiplierValue : number = calculateMultiplier(depositAmount);
37      setMultiplier(multiplierValue);
38    }
39  }
40
41  // ...
42
43  // ...
44
45  // ...
46
47  // ...
48
49  // ...
50
51  // ...
52
53  // ...
54
55  // ...
56
57  // ...
58
59  // ...
60
61  // ...
62
63  // ...
64
65  // ...
66
67  // ...
68
69  // ...
70
71  // ...
72
73  // ...
74
75  // ...
76
77  // ...
78
79  // ...
80
81  // ...
82
83  // ...
84
85  // ...
86
87  // ...
88
89  // ...
90
91  // ...
92
93  // ...
94
95  // ...
96
97  // ...
98
99  // ...
100
101  // ...
102
103  // ...
104
105  // ...
106
107  // ...
108
109  // ...
110
111  // ...
112
113  // ...
114
115  // ...
116
117  // ...
118
119  // ...
120
121  // ...
122
123  // ...
124
125  // ...
126
127  // ...
128
129  // ...
130
131  // ...
132
133  // ...
134
135  // ...
136
137  // ...
138
139  // ...
140
141  // ...
142
143  // ...
144
145  // ...
146
147  // ...
148
149  // ...
150
151  // ...
152
153  // ...
154
155  // ...
156
157  // ...
158
159  // ...
160
161  // ...
162
163  // ...
164
165  // ...
166
167  // ...
168
169  // ...
170
171  // ...
172
173  // ...
174
175  // ...
176
177  // ...
178
179  // ...
180
181  // ...
182
183  // ...
184
185  // ...
186
187  // ...
188
189  // ...
190
191  // ...
192
193  // ...
194
195  // ...
196
197  // ...
198
199  // ...
200
201  // ...
202
203  // ...
204
205  // ...
206
207  // ...
208
209  // ...
210
211  // ...
212
213  // ...
214
215  // ...
216
217  // ...
218
219  // ...
220
221  // ...
222
223  // ...
224
225  // ...
226
227  // ...
228
229  // ...
230
231  // ...
232
233  // ...
234
235  // ...
236
237  // ...
238
239  // ...
240
241  // ...
242
243  // ...
244
245  // ...
246
247  // ...
248
249  // ...
250
251  // ...
252
253  // ...
254
255  // ...
256
257  // ...
258
259  // ...
260
261  // ...
262
263  // ...
264
265  // ...
266
267  // ...
268
269  // ...
270
271  // ...
272
273  // ...
274
275  // ...
276
277  // ...
278
279  // ...
280
281  // ...
282
283  // ...
284
285  // ...
286
287  // ...
288
289  // ...
290
291  // ...
292
293  // ...
294
295  // ...
296
297  // ...
298
299  // ...
300
301  // ...
302
303  // ...
304
305  // ...
306
307  // ...
308
309  // ...
310
311  // ...
312
313  // ...
314
315  // ...
316
317  // ...
318
319  // ...
320
321  // ...
322
323  // ...
324
325  // ...
326
327  // ...
328
329  // ...
330
331  // ...
332
333  // ...
334
335  // ...
336
337  // ...
338
339  // ...
340
341  // ...
342
343  // ...
344
345  // ...
346
347  // ...
348
349  // ...
350
351  // ...
352
353  // ...
354
355  // ...
356
357  // ...
358
359  // ...
360
361  // ...
362
363  // ...
364
365  // ...
366
367  // ...
368
369  // ...
370
371  // ...
372
373  // ...
374
375  // ...
376
377  // ...
378
379  // ...
380
381  // ...
382
383  // ...
384
385  // ...
386
387  // ...
388
389  // ...
390
391  // ...
392
393  // ...
394
395  // ...
396
397  // ...
398
399  // ...
400
401  // ...
402
403  // ...
404
405  // ...
406
407  // ...
408
409  // ...
410
411  // ...
412
413  // ...
414
415  // ...
416
417  // ...
418
419  // ...
420
421  // ...
422
423  // ...
424
425  // ...
426
427  // ...
428
429  // ...
430
431  // ...
432
433  // ...
434
435  // ...
436
437  // ...
438
439  // ...
440
441  // ...
442
443  // ...
444
445  // ...
446
447  // ...
448
449  // ...
450
451  // ...
452
453  // ...
454
455  // ...
456
457  // ...
458
459  // ...
460
461  // ...
462
463  // ...
464
465  // ...
466
467  // ...
468
469  // ...
470
471  // ...
472
473  // ...
474
475  // ...
476
477  // ...
478
479  // ...
480
481  // ...
482
483  // ...
484
485  // ...
486
487  // ...
488
489  // ...
490
491  // ...
492
493  // ...
494
495  // ...
496
497  // ...
498
499  // ...
500
501  // ...
502
503  // ...
504
505  // ...
506
507  // ...
508
509  // ...
510
511  // ...
512
513  // ...
514
515  // ...
516
517  // ...
518
519  // ...
520
521  // ...
522
523  // ...
524
525  // ...
526
527  // ...
528
529  // ...
530
531  // ...
532
533  // ...
534
535  // ...
536
537  // ...
538
539  // ...
540
541  // ...
542
543  // ...
544
545  // ...
546
547  // ...
548
549  // ...
550
551  // ...
552
553  // ...
554
555  // ...
556
557  // ...
558
559  // ...
560
561  // ...
562
563  // ...
564
565  // ...
566
567  // ...
568
569  // ...
570
571  // ...
572
573  // ...
574
575  // ...
576
577  // ...
578
579  // ...
580
581  // ...
582
583  // ...
584
585  // ...
586
587  // ...
588
589  // ...
590
591  // ...
592
593  // ...
594
595  // ...
596
597  // ...
598
599  // ...
600
601  // ...
602
603  // ...
604
605  // ...
606
607  // ...
608
609  // ...
610
611  // ...
612
613  // ...
614
615  // ...
616
617  // ...
618
619  // ...
620
621  // ...
622
623  // ...
624
625  // ...
626
627  // ...
628
629  // ...
630
631  // ...
632
633  // ...
634
635  // ...
636
637  // ...
638
639  // ...
640
641  // ...
642
643  // ...
644
645  // ...
646
647  // ...
648
649  // ...
650
651  // ...
652
653  // ...
654
655  // ...
656
657  // ...
658
659  // ...
660
661  // ...
662
663  // ...
664
665  // ...
666
667  // ...
668
669  // ...
670
671  // ...
672
673  // ...
674
675  // ...
676
677  // ...
678
679  // ...
680
681  // ...
682
683  // ...
684
685  // ...
686
687  // ...
688
6
```

```

43     setShowGame( value: true); // Show the mini-game
44     setBonusMessage( value: "");
45     setLuckyCard( value: Math.floor( x: Math.random() * 3) + 1); // Random lucky card (1, 2, or 3)
46     setGameOver( value: false);
47   } else {
48     alert("Please enter a valid deposit amount!");
49   }
50 };
51
52 const handleCardSelection = (card) : void => { Show usages  raimondseisaks
53   const deposit : number = parseFloat(depositAmount);
54
55   if (card === luckyCard) {
56     const total : number = deposit * multiplier;
57     setBonusMessage(
58       value: `Congratulations! You picked the lucky card! You received a bonus of $$${total.toFixed( fractionDigits:
59     )};
60     userData[2] = (parseFloat(userData[2]) + total).toFixed( fractionDigits: 2);
61     sendMessage(`update-balance-amount ${userData[2]}`);
62   } else {
63     setBonusMessage(
64       value: `Bad Luck! You don't get a bonus.`
65     );
66   }
67   setShowGame( value: false); // Hide the game after the result
68   setGameOver( value: true); // Set the game as over
69 };
70
71 return (
72   <div className="balance-container">
73     <div className="form-group">
74       <h2>Current Balance</h2>
75       <h3 className="balance-text">Your current balance: ${userData[2]}</h3>
76       {!gameOver && ( // Only show Add Balance button if game is not over
77         <form onSubmit={startMiniGame}>
78           <input
79             type="number"
80             placeholder="Enter deposit amount"
81             className="form-control"
82             value={depositAmount}

```

```

83         onChange={(e : ChangeEvent<HTMLInputElement> ) : void => setDepositAmount(e.target.value)}
84         required
85     />
86     <button className="btn btn-outline-light mt-3" type="submit">
87         Calculate Multiplier
88     </button>
89 </form>
90 }}
91 {timeRestrictedMessage && (
92     <p className="text-light mt-3">{timeRestrictedMessage}</p>
93 )}
94
95 {/* Show multiplier only after clicking "Add Balance" */}
96 {showGame && depositAmount && (
97     <div className="mt-3">
98         <p className="text-light">Your multiplier: {multiplier}</p>
99     </div>
100 )}
101 </div>
102
103 {/* Show mini-game when ready */}
104 {showGame && (
105     <div className="mini-game">
106         <h3 className="text-light">Pick a Card to See Your Bonus!</h3>
107         <div className="cards">
108             <button className="card-game" onClick={() : void => handleCardSelection( card: 1)}>
109                 Card 1
110             </button>
111             <button className="card-game" onClick={() : void => handleCardSelection( card: 2)}>
112                 Card 2
113             </button>
114             <button className="card-game" onClick={() : void => handleCardSelection( card: 3)}>
115                 Card 3
116             </button>
117         </div>
118     </div>
119 )}

```

```

120
121     {/* Show bonus message */}
122     {bonusMessage && <p className="text-light fw-bold mt-3">{bonusMessage}</p>}
123 </div>
124 );
125 };
126
127 export default BalanceContainer; Show usages  raimondseisaks
128

```

Pielikums 2 – Laimes rata stāvokļa nomaina (GE003) un laimes rata laimīga cipara ģenerēšanas (GE004) funkcija

```

1 package reisaks.FinalProject.ServerSide.GameLogic
2
3 import akka.actor.ActorRef
4 import cats.effect.IO
5 import reisaks.FinalProject.ServerSide.AkkaActors.TableActorMessages._
6 import scala.concurrent.duration._
7 import scala.util.Random
8
9 object SpinningWheel { @ raimondseisaks *
10
11     //Generate random number from 1 to 100
12     private def generateWinningNumber(): IO[Int] = IO { @ raimondseisaks
13         Random.nextInt(100) + 1
14     }
15
16     //Change state of spinning wheel actor and send winning number
17     def stateChanger(tableRef: ActorRef): IO[Unit] = { @ raimondseisaks
18         def loop: IO[Unit] = for {
19             _ <- IO(tableRef ! BetsStart)
20             _ <- IO.sleep(10.seconds)
21             _ <- IO(tableRef ! BetsEnd)
22             _ <- IO.sleep(2.seconds)
23             number <- generateWinningNumber()
24             _ <- IO(tableRef ! GameStart(number))
25             _ <- IO.sleep(5.seconds)
26             _ <- IO(tableRef ! GameResult(number))
27             _ <- IO.sleep(3.seconds)
28             _ <- IO(tableRef ! RoundEnd)
29         } yield ()
30         loop
31     }
32 }
33
34

```

Pielikums 3 – Spēlētāju sesijas pārvaldnieks (PlayerManager), kurā ietilpst lietotāja moduļa funkcijas

```

1 package reisaks.FinalProject.DomainModels
2
3 import akka.actor.TypedActor.dispatcher
4 import akka.actor.{ActorRef, ActorSystem, PoisonPill}
5 import cats.effect.unsafe.implicit.global
6 import cats.effect.{IO, Ref}
7 import reisaks.FinalProject.DomainModels.SystemMessages.{ExistingID, GameError}
8 import reisaks.FinalProject.ServerSide.AkkaActors.PlayerActor
9 import reisaks.FinalProject.ServerSide.AkkaActors.TableActorMessages.{AddBetToTable, DeleteBet, LeaveTable, SendChat}
10 import reisaks.FinalProject.DomainModels.SqlDatabase._
11 import reisaks.FinalProject.ServerSide.AkkaActors.PlayerActorMessages.MessageToPlayer
12 import scala.util.Success
13 import java.util.Properties
14 import javax.mail._
15 import javax.mail.internet._
16
17 case class Player(var playerId: String, actorRef: ActorRef) ⚡ raimondseisaks
18
19 trait PlayerManagerTrait { ⚡ raimondseisaks
20   def createSession(id: String, password: String): IO[Either[GameError, Player]] ⚡ raimondseisaks
21   def endPlayerSession(player: Player): IO[Unit] ⚡ raimondseisaks
22   def isPlayerPlaying(player: Player): IO[Boolean] ⚡ raimondseisaks
23   def addToTable(player: Player, actorRef: ActorRef): IO[Unit] ⚡ raimondseisaks
24   def leftTable(player: Player): IO[Unit] ⚡ raimondseisaks
25   def requestAddBet(player: Player, betCode: String, amount: String): IO[Unit] ⚡ raimondseisaks
26   def deleteBet(player: Player, betCode: String): IO[Unit] ⚡ raimondseisaks
27   def sendChatMessage(player: Player, message: String): IO[Unit] ⚡ raimondseisaks
28   def deleteUser(player: Player, password: String): Unit ⚡ raimondseisaks
29   def sendSupportMessage(player: Player, playerEmail: String, messageText: String): Unit ⚡ raimondseisaks
30 }
31
32 class PlayerManager(system: ActorSystem, ref: Ref[IO, Map[Player, Option[ActorRef]]]) extends PlayerManagerTrait {
33
34   def createSession(id: String, password: String): IO[Either[GameError, Player]] = {...}
35
36   def endPlayerSession(player: Player): IO[Unit] = { ⚡ raimondseisaks
37     isPlayerPlaying(player).flatMap { result =>
38       if (result) {
39         IO(player.actorRef ! MessageToPlayer("Please leave the table"))
40       } else {
41         for {

```

```

54         _ <- IO(player.actorRef ! PoisonPill)
55         _ <- ref.update(existingPlayerIds => existingPlayerIds - player)
56     } yield ()
57 }
58 }
59 }
60
61  def isPlayerPlaying(player: Player): IO[Boolean] = {  raimondseisaks
62     ref.get.map { playerMap =>
63         playerMap.get(player) match {
64             case Some(None) => false
65             case Some(_) => true
66         }
67     }
68
69  def leftTable(player: Player): IO[Unit] = {  raimondseisaks
70     ref.update { playersMap =>
71         playersMap.get(player) match {
72             case Some(Some(table)) =>
73                 table ! LeaveTable(player)
74                 playersMap + (player -> None)
75         }
76     }
77 }
78
79  def addToTable(player: Player, actorRef: ActorRef): IO[Unit] = {  raimondseisaks
80     ref.update { playersMap =>
81         playersMap.get(player) match {
82             case Some(None) =>
83                 val updated = playersMap + (player -> Some(actorRef))
84                 updated
85         }
86     }
87 }
88
89  def requestAddBet(player: Player, betCode: String, amount: String): IO[Unit] =
90     ref.get.flatMap(w => w.get(player) match {
91         case Some(Some(table)) => IO {

```







```

92     Bet.create(betCode, amount) match {
93         case Right(bet) => table ! AddBetToTable(player, bet)
94         case Left(error) => player.actorRef ! MessageToPlayer(error.message)
95     }
96 }
97 })
98
99  def deleteBet(player: Player, betCode: String): IO[Unit] = {  raimondseisaks
100     ref.get.flatMap(w => w.get(player) match {
101         case Some(Some(table)) => IO {
102             table ! DeleteBet(player, betCode)
103         }
104     })
105 }
106
107  def sendChatMessage(player: Player, message: String): IO[Unit] =  raimondseisaks
108     ref.get.flatMap(w => w.get(player) match {
109         case Some(Some(table)) => IO {
110             table ! SendChatMessage(message, player)
111         }
112         case _ => IO.unit
113     })
114

```

```

115  def deleteUser(player: Player, password: String): Unit = {  raimondseisaks
116     deleteUserById(player, password)
117     endPlayerSession(player).unsafeToFuture()
118 }
119
120  def sendSupportMessage(player: Player, playerEmail: String, messageText: String): Unit = {
121     val supportEmail = "eisaks83@gmail.com" // Support email
122     val host = "smtp.gmail.com" // Gmail SMTP server
123     val username = "eisaks83@gmail.com"
124     val password = "xmge opci phpi jrsm"
125
126     val properties = new Properties()
127     properties.put("mail.smtp.auth", "true")
128     properties.put("mail.smtp.starttls.enable", "true")
129     properties.put("mail.smtp.host", host)
130     properties.put("mail.smtp.port", "587")
131
132     // Authenticate and get the session
133     val session = Session.getInstance(properties, new Authenticator() {
134  override protected def getPasswordAuthentication: PasswordAuthentication = {
135         new PasswordAuthentication(username, password)
136     }
137     })
138
139     try {
140         val message = new MimeMessage(session)
141         message.setFrom(new InternetAddress(playerEmail))
142         message.setRecipients(Message.RecipientType.TO, supportEmail)
143         message.setSubject("Player Support Request")
144         message.setText(s"Player Email: $playerEmail\n\n$messageText")
145
146         Transport.send(message)
147         player.actorRef ! MessageToPlayer("email-sent")
148     } catch {
149         case e: MessagingException =>
150             e.printStackTrace()
151             player.actorRef ! MessageToPlayer("email-not-sent")
152     }
153 }
154 }

```

Kvalifikācijas darbs ***Laimes rata spēles serviss*** izstrādāts Latvijas Universitātes Eksakto zinātņu un tehnoloģiju fakultātē, Datorikas nodaļā.

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

Autors: ***Raimonds Eisaks*** _____ **06.01.2025.**

Rekomendēju darbu aizstāvēšanai

Darba vadītājs: ***M.energ, Sergejs Bogdanovs*** _____ **06.01.2025.**

Recenzents: ***Egons Šolmanis***

Darbs iesniegts **06.01.2025.**

Kvalifikācijas darbu pārbaudījumu komisijas sekretārs (elektronisks paraksts)