

Liepājas Valsts Tehnikums

Datorspēle “ScoreStorm”

Kvalifikācijas eksāmena praktiskās daļas dokumentācija

Profesionālā kvalifikācija Programmēšanas tehniķis

Grupas nosaukums 4PT

Projekta izstrādātājs Gustavs Narvils

Eksāmena datums 2024. gada 20. jūnijs

Liepāja 2024

# Saturs

[Ievads 4](#_Toc164695519)

[1. Uzdevuma formulējums 5](#_Toc164695520)

[2. Programmatūras prasību specifikācija 7](#_Toc164695521)

[2.1. Produkta perspektīva 7](#_Toc164695522)

[2.2. Sistēmas funkcionālās prasības 7](#_Toc164695523)

[2.2.1. Vispārīgās prasības 7](#_Toc164695524)

[2.2.2. Iestatījuma loga prasības 9](#_Toc164695525)

[2.2.3. Pirmspēles loga prasības 11](#_Toc164695526)

[2.2.4. Spēles loga prasības 12](#_Toc164695527)

[2.3. Sistēmas nefunkcionālās prasības 20](#_Toc164695528)

[2.3.1. Valoda 20](#_Toc164695529)

[2.3.2. Saskaņotība 20](#_Toc164695530)

[2.3.3. Vizuālais izskats 20](#_Toc164695531)

[2.3.4. Datorspēles veiktspējā 20](#_Toc164695532)

[2.3.5. Spēles skaņas 20](#_Toc164695533)

[2.4. Gala lietotāja raksturiezīmes 21](#_Toc164695534)

[3. Izstrādes līdzekļu, rīku apraksts un izvēles pamatojums 22](#_Toc164695535)

[3.1. Izvēlēto risinājuma līdzekļu un valodu apraksts 22](#_Toc164695536)

[3.2. Iespējamo risinājuma līdzekļu un valodu apraksts 23](#_Toc164695537)

[4. Sistēmas struktūras modelēšana un projektēšana 26](#_Toc164695538)

[4.1. Sistēmas struktūras modelis 26](#_Toc164695539)

[4.2. Klašu diagramma / ER diagramma 27](#_Toc164695540)

[4.3. Funkcionālais un dinamiskais sistēmas modelis 28](#_Toc164695541)

[4.4. Aktivitāšu diagramma 29](#_Toc164695542)

[4.5. Lietotjumgadījuma diagramma 34](#_Toc164695543)

[4.6. Sistēmas moduļu apraksts un algoritmu shēmas 35](#_Toc164695544)

[5. Lietotāju ceļvedis 37](#_Toc164695545)

[5.1. Galvenā izvēlne 37](#_Toc164695546)

[5.2. Iestatījumi 37](#_Toc164695547)

[5.3. Pirmspēles logs 38](#_Toc164695548)

[5.4. Spēle 38](#_Toc164695549)

[6. Testēšanas dokumentācija 40](#_Toc164695550)

[6.1. Izvēlētais testēšanas metodes, rīku apraksts un pamatojums 40](#_Toc164695551)

[6.2. Testpiemēru kopa 42](#_Toc164695552)

[6.3. Testēšanas žurnāls 43](#_Toc164695553)

[7. Secinājumi 44](#_Toc164695554)

[8. Lietoto terminu un saīsinājumu skaidrojumi 45](#_Toc164695555)

[9. Literatūras un informācijas avotu saraksts 46](#_Toc164695556)

[Pielikums 47](#_Toc164695557)

# Ievads

“ScoreStorm” ir datorspēle, kas ir izstrādāta priekš Windows operētājsistēmas datoriem, kur viens no izveidošanas iemesliem bija ka Latvijā nav daudz spēļu veidošanas uzņēmumu vai grupas, kuras izstrādā augstas kvalitātes spēles. Manuprāt, ja Latvijā būtu vairāki spēļu veidotāji, tad tas palīdzētu attīstīt Latvijas IT jomas sektoru un to uzstatīsit pazīstamāku pasaulē, kā arī ienest vairāk biznesu Latvijai. Jo vismaz pagaidām Latvijā nav vel īsti nekāda spēļu veidošanas uzņēmums vai grupu, kura ir ļoti pazīstama visā pasaulē. Tādēļ, manuprāt, šī spēle ir nepieciešama, jo tā varētu attīstīt Latvijas spēļu veidošanas jomu, jo ir iespējams, ka šī spēle var palikt ļoti populāra un tā rezultātā padarītu Latviju zināmāku un pievilcīgāku citiem IT uzņēmumiem gan arī citiem spēles veidošanas uzņēmumiem.

Šī spēle arī, manuprāt, ir nepieciešama, jo personīgi pats esmu ieraudzījis, ka vismaz pagaidām nav neviena spēlē tirgū, kura ir līdzīgi mana spēles idejai, jo viņas visas parasti ir citā spēles žanrā vai arī viņas nav pilnībā realizētas. Piemēram, ir spēlēs, kur galvenais mērķis ir spēlēt vienu un to pašu līmeni, lai dabūtu cik vien augstu rezultātu spēlētājs var, bet gandrīz visas šīs spēles ir citos žanros kā ritmu žanrā vai arī tajās spēlēs, kuras ir šaušanas kategorijas bieži neļauj tev kustēties, vai arī spēles gaita ir ļoti vienkārša un tādā rezultāta nav jautri tās spēlēt atkārtoti. Tādēļ ar savējo spēli es cenšos uzstatīsit to žanrā, kur nav tik daudz šādas veida spēles.

Vel viens iemesls kāpēc šī spēle, manuprāt, ir nepieciešama, jo es zinu, ka šāda veida spēlēm ir liels pieprasījums. Kā ir iepriekš minēts, eksistē spēles, kur spēlētājam vaig dabūt visvairāk punktus, bet viņi ir parasti citos žanros, bet vienalga šīs spēles ir ļoti populāras. Šīm veidu spēlēm parasti ir ļoti aktīvi spēlētāji, kuri sacenšas ar viens pret otru, lai dabūtu visslābāko rekordu. Tādēļ ja ir uztaisīta spēle, kura ir pievilcīga tāda veida spēlētājiem, spēle paliks populāra tāda veida spēlētājiem un arī vienmēr būs aktīvi spēlētāji, kuri centīsies pārspēt cita spēlētāja labāko rekordu.

Pēdējais iemesls kāpēc šī spēle ir nepieciešam ir tāpēc, ka daudzas šaušanas spēlēs pārsvarā ir domātas priekš pieaugušiem tādēļ tajās spēles ir bieži ļoti daudzas lietas, kas nav piemēroti visiem vecumiem, bet ar savējo spēli es izveidošu spēli, ko visi cilvēki vienalga no kāda vecuma var spēlēt un izbaudīt. Tādēļ šī spēle ir nepieciešama, jo tā ir arī piemērota tādiem cilvēkiem, kuri nespēlē tāda veida žanra spēles, jo viņas satur lietas, kas priekš viņiem nav piemērotas. Tas nenozīmē, ka šī spēle ir domāta pilnībā priekš maziem bērniem, bet tas nozīmē, ka spēle ir domāta cilvēkiem pusaudžu vecumā, kuri ir vieni no aktīvākajiem datorspēles spēlētājiem.

# 1. Uzdevuma formulējums

“ScoreStorm” ir datorspēle, kura vismaz pagaidām ir izstrādāta tikai priekš Windows operētājsistēmas, kur spēles galvenais mērķis ir dabūt cik vien daudz punktus fiziski spēles spēlētājs var. Spēles laikā spēlētājs redz savu cilvēku no trešās personas skata un var kontrolē spēlēs cilvēku izmantojot datora tastatūru, un pelīti, kur spēlētājs var skatīties, un staigāt apkārt, gan arī mērķēt un šaut vienu no viņu ieročiem. Pirms spēles sākuma spēlētājam ir dota iespēja izvēlēties vienu no diviem spēlējamiem cilvēkiem, kur katram ir doti divi ieroči, kur katrs strādā savādāk un ir savi plusi un mīnusi. Spēlētājam ir arī dota iespēja izvēlēties starp diviem spēles laukiem, kur atšķirība ir spēles lauka izskats un izkārtojums.

Spēles laikā spēlētājam ir dotas vairākas iespējas, kā viņš varētu iegūt punktus. Pirmkārt, spēlētājs var dabūt punktus nošaujot uz spēles lauka esošos pretiniekus, kuri tev vienmēr uzbruks un arī, kuri vienmēr nejaušās vietās parādīsies uz lauka, kad kāds cits pretinieks tiks nošauts. Spēlē ir divu veidu pretinieki parastie, kurus ir viegli nošaut un nedara tik daudz bojājumus tev spēles laikā un ir īpašie pretinieki, kur viņiem ir kaut kāda speciāls atribūts, kurš pataisa viņus stiprākus, bet ja spēlētājs var vienu nošaut, tad viņš dabūs lielu punktu bonusu. Viens no īpašiem pretiniekiem ir ļoti stiprs, kur viņu ir grūti nošaut un ja viņš tev trāpa tad uzreiz spēle beidzās, bet viņš ļoti lēni kustās, kamēr otrs ir ļoti ātri kustās, bet viņu ir viegli nošaut. Otrkārt, ja spēlētājs spēles laikā grib dabūt daudz punktus raundā, spēlētājs spēles laikā var dabūt reizinātāja kapsulas, kur uz īsu laiku spēlētāja iegūtais punktu daudzums tajā laikā tiks reizināts. Pēdējais veids kā spēlētājs var iegūt punktus spēles laikā ir atrast reti sastopamus priekšmetus, kur ja spēlētājs to atradīs un savāks, tad viņš dabūs lielu punktu bonusu.

Spēles laikā spēlētājam ir arī dots, laiks līdz spēles beigšanai, kur ja laiks beidzās tad spēle automātiski beidzas. Spēles laikā spēlētājam ir doti divi veidi, kā viņš varētu laiku pagarināt, pirmkārt, atrodot laika kapsulas, kuras atrodas apkārt spēles laukam un, otrkārt, nošaujot pretinieku, kur ja nošauj parasto pretinieku, tad laiks tikai pa mazlietiņ palielinās, bet ja nošauj īpašo tad laiks tiek pa vairāk palielināts.

Spēles laikā spēlētājam ir arī dots “kombo” skaitītājs, kur ja spēlētājs var nošaut divus pretiniekus īsā noteiktā laikā tad tas palielināsies, bet ja spēlētājs nenošauj pretinieku noteiktajā laikā, tad tas pazūd. Šis skaitītājs ir svarīgs, jo spēles beigās spēlētajā iegūtais augstākais “kombo” skaitītāja augstākais skaitlis tiek reizināts ar spēlētāja rezultātu un tas nozīmē spēlētājs var iegūt ļoti daudz punktus.

Spēle var beigties dēļ trīs iemesliem, spēles laiks beidzās, kāds pretinieks tevi nošāva vai arī spēlētājs nošāva pilnībā visus pretiniekus, kas ir piedāvāti uz spēles lauka. Kad spēle ir beigusies uz ekrāna tiek parādīts spēlētāja beigās iegūtais rezultāt un arī pasaka vai tas ir jauns rekords vai nē. Kad spēlētājs izvēlas iet atpakaļ uz spēles sākumekrāna viņa rezultāts tiks saglabāts ja tas bija jauns rekords un tad tiks parādīts pirmspēles ekrānā. Spēlētājam sākumekrānā ir poga, kas viņu aizvedīs uz iestatījuma lapas, kur spēlētājs var mainīt daudzas lietas saistībā ar viņu spēli, kā ar kurām pogām kontrolēt spēli, ekrāna rezolūciju, skaņās skaļumu un peles ātrumu.

# 2. Programmatūras prasību specifikācija

Šajā nodaļa tiks aprakstītas visas lietas saistītas par spēlēs perspektīvu, funkcionālajām prasībām, nefunkcionālas prasības un arī kāda ir spēles spēlētāja raksturiezīmes.

## 2.1. Produkta perspektīva

Datorspēle “ScoreStorm” ir trešās personas šaušanas spēle, kur galvenais mērķis tev ir dabūt cik vien daudz punktus tu vari. Šī spēle ir unikāla ar to, ka šī spēle tu var kustēties apkārt pa spēles lauku ar savu spēles tēlu, bet citās spēles tu parasti paliec uz vietas.

Pagaidām datorspēlei visas galvenās funkcijas un prasības ir jau izstrādāt, bet ir vel dažas lietas, kuras būtu vel vēlams izstrādāt vēlāk pēc projekta pabeigšanas. Pirmais ir izstrādāt valodas opcijas, kur lietotājs varēs pārmainīt spēles lietotāja interfeisa valodu uz citu valodu. Pirmā valoda, kura būtu izstrādāta ir angļu, bet pēc tam varētu būtu vācu, spāņu, franču utt. Otrais būtu izstrādāt vairāk spēles tēlus un arī līmeņus, kur katram ir savi unikāli ieroči un arī katram līmeni ir arī savi unikāli pretinieki. Varbūt vel pēc tam izstrādātu spēlei līmeņa veidotāju, kurš ļautu spēlētājiem izstrādāt pašam savus līmeņus, iestatījumus un varbūt vel pat pielikt klāt pašus savus modeļus un tekstūras. Trešā lietā, ko es spēlei gribētu nākotnē izstrādāt ir kaut kāds daudz spēlētāju režīms, kur divi vai vairāk spēlētāji varētu spēlēt vienlaikus vienā līmenī. Pēdējā lieta, ko es gribētu izstrādāt priekš šīs spēles ir atbalsts ar vairākām perifērijas ierīcēm, jo pagaidām vienīgais veids, kā spēlēt šo spēli ir ar peli un klaviatūru, bet nākotnē es gribēt arī dot iespēju spēlēt spēli ar vairākām ierīcēm. Viena ierīce, ko es gribētu atbalstīt nākotnē ir “Xbox” un “PlayStation” spēles pultis, jo ja spēle atbalstītu tiešu tās ierīces, tad ir iespējams nākotnē šo spēli pat uzlikt uz spēļu konsolēm. Otrā ierīce, ko es gribētu atbalstīt ir skārienjutīgie ekrāni, jo nākotnē es šo spēli gribētu uzlikt uz telefoniem tādēļ būtu jāizstrādā priekš šīs spēles virtuālās pogas, lai spēlētājs varētu spēlēt spēli izmantojot pirkstus.

## 2.2. Sistēmas funkcionālās prasības

Šajā nodaļa tiks aprakstītas visas datorspēles “ScoreStorm” funkcionālās prasības.

### 2.2.1. Vispārīgās prasības

**P.1. Datorspēles “ScoreStorm” aizvēršana**

Mērķis:

Šīs funkcijas mērķis ir nodrošināt iespēju spēlētājam aizvērt spēli izmantojot pogu.

Ievaddati:

Peles klikšķis uz pogu “Iziet uz darbvirsmu”.

Apstrāde:

1. Tiek pārbaudīts vai spēlētājs atrodas galvenā izvēlne.
2. Tiek pārbaudīts vai spēlētājs nospieda pogu.

Izvaddati:

Tiek spēlēta pogas skaņa un pēc vienas sekundes spēle tiks aizvērta.

**P.2. Datorspēles “ScoreStorm” iestatījuma loga atvēršana**

Mērķis:

Šīs funkcijas mērķis ir nodrošināt iespēju spēlētājam atvērt iestatījuma logu izmantojot pogu.

Ievaddati:

Peles klikšķis uz pogu “Iestatījumi”.

Apstrāde:

1. Tiek pārbaudīts vai spēlētājs atrodas galvenā izvēlne.
2. Tiek pārbaudīts vai spēlētājs nospieda pogu.

Izvaddati:

Tiek spēlēta pogas skaņa un pēc vienas sekundes spēlētājs tiek pārvietots uz iestatījuma logu.

**P.3. Datorspēles “ScoreStorm” pirmspēles loga atvēršana**

Mērķis:

Šīs funkcijas mērķis ir nodrošināt iespēju spēlētājam atvērt pirmspēles logu izmantojot pogu.

Ievaddati:

Peles klikšķis uz pogu “Spēlēt”.

Apstrāde:

1. Tiek pārbaudīts vai spēlētājs atrodas galvenā izvēlne.
2. Tiek pārbaudīts vai spēlētājs nospieda pogu.

Izvaddati:

Tiek spēlēta pogas skaņa un pēc vienas sekundes spēlētājs tiek pārvietots uz pirmspēles logu.

**P.4. Datorspēles “ScoreStorm” galvenās izvēlnes atgriešanās**

Mērķis:

Šīs funkcijas mērķis ir nodrošināt iespēju spēlētājam atgriezties atpakaļ galvenās izvēlnes logā izmantojot pogu.

Ievaddati:

Peles klikšķis uz pogu ar apzīmējumu “X”.

Apstrāde:

1. Tiek pārbaudīts vai spēlētājs atrodas pirmspēles vai iestatījuma logā.
2. Tiek pārbaudīts vai spēlētājs nospieda pogu.

Izvaddati:

Tiek spēlēta pogas skaņa un pēc vienas sekundes spēlētājs tiek pārvietots uz galvenās izvēlnes logu.

### 2.2.2. Iestatījuma loga prasības

**P.5. Datorspēles “ScoreStorm” skaņas skaļuma maiņa**

Mērķis:

Šīs funkcijas mērķis ir nodrošināt iespēju spēlētājam mainīt spēles skaņas skaļumu ar bīdni.

Ievaddati:

1. Peles kreisā taustiņa turēšana uz skaņas bīdni.
2. Peli kustina pa labi vai pa kreisi.

Apstrāde:

1. Funkcija pārbauda vai spēlētājs kustina peli.
2. Funkcija pārbauda vai spēlētājs ir sasniedzis spēles maksimālo skaņas skaļuma limitu.
3. Funkcija pārbauda vai spēlētājs ir sasniedzis spēles minimālo skaņas skaļuma limitu.

Izvaddati:

1. Ja spēlētājs bīda bīdni uz kreiso pusi, spēles skaņas skaļums samazināsies.
2. Ja spēlētājs bīda bīdni uz labo pusi, spēles skaņas skaļums palielinās.

**P.6. Datorspēles “ScoreStorm” mūzikas skaļuma maiņa**

Mērķis:

Šīs funkcijas mērķis ir nodrošināt iespēju spēlētājam mainīt spēles mūziku skaļumu ar bīdni.

Ievaddati:

1. Peles kreisā taustiņa turēšana uz mūzikas bīdni.
2. Peli kustina pa labi vai pa kreisi.

Apstrāde:

1. Funkcija pārbauda vai spēlētājs kustina peli.
2. Funkcija pārbauda vai spēlētājs ir sasniedzis spēles maksimālo mūzikas skaļuma limitu.
3. Funkcija pārbauda vai spēlētājs ir sasniedzis spēles minimālo mūzikas skaļuma limitu.

Izvaddati:

1. Ja spēlētājs bīda bīdni uz kreiso pusi, spēles mūzikas skaļums samazināsies.
2. Ja spēlētājs bīda bīdni uz labo pusi, spēles mūzikas skaļums palielinās.

**P.7. Datorspēles “ScoreStorm” rezolūcijas maiņa**

Mērķis:

Šīs funkcijas mērķis ir nodrošināt iespēju spēlētājam mainīt spēles ekrāna rezolūciju ar izvēlni.

Ievaddati:

1. Spēlētājs nospiež peles kreiso taustiņu virsū uz rezolūcijas izvēlni.
2. Spēlētājs nospiež peles kreiso taustiņu virsū uz viņa izvēlēto rezolūciju.

Apstrāde:

1. Funkcija pārbauda vai spēlētājs ir nospiedis peles kreiso taustiņu.
2. Funkcija pārbauda spēlētāja ekrāna maksimālo rezolūciju.
3. Funkcija pārbauda vai spēlētājs nav izvēlējies tādu pašu rezolūciju kā tagad.
4. Funkcija pārbauda vai spēlētājs ir nospiedis peles kreiso taustiņu uz izvēlēto rezolūciju.

Izvaddati:

Spēlētājs ekrāna izšķirtspēja mainās uz to, ko viņš bija izvēlējies.

**P.8. Datorspēles “ScoreStorm” peles kursora ātruma maiņa**

Mērķis:

Šīs funkcijas mērķis ir nodrošināt iespēju spēlētājam mainīt peles kursora ātrumu ar bīdni.

Ievaddati:

1. Peles kreisā taustiņa turēšana uz peles kursora ātruma bīdņa.
2. Peles kustināšana pa labi vai pa kreisi.

Apstrāde:

1. Funkcija pārbauda vai spēlētājs kustina peli.
2. Funkcija pārbauda vai spēlētājs ir sasniedzis peles kursora maksimālo ātruma limitu.
3. Funkcija pārbauda vai spēlētājs ir sasniedzis peles kursora minimālo ātrumu limitu.

Izvaddati:

1. Ja spēlētājs bīda bīdni uz kreiso pusi, peles kursora ātrums samazinās.
2. Ja spēlētājs bīda bīdni uz labo pusi, peles kursora ātrums palielinās.

**P.9. Datorspēles “ScoreStorm” darbības pogas maiņa**

Mērķis:

Šīs funkcijas mērķis ir nodrošināt iespēju spēlētājam mainīt visas spēles darbības aktivizācijas pogas. Šīs darbības iekļauj staigāšanu, skriešana, bloķēšana, ieroča pārlādēšana, saskare un ieroča maiņa.

Ievaddati:

1. Peles kreisā taustiņa klikšķis uz darbības maiņa ievades lauka.
2. Nospiež jebkuru klaviatūras taustiņu.

Apstrāde:

1. Funkcija pārbauda vai spēlētājs ir nospiedis klaviatūras taustiņu
2. Funkcija pārbauda vai spēlētājs jau nav izvēlējies tādu pašu pogu priekš citas darbības.

Izvaddati:

1. Ja spēlētājs nav izvēlējies tādu pašu pogu, kā citai darbībai tad tā poga tiek iestatīta, kā darbības aktivizācijas poga.
2. Ja spēlētājs ir iestatījis tādu pašu pogu, kā citai darbībai tad nekas nenotiek.

### 2.2.3. Pirmspēles loga prasības

**P.10. Spēles tēla izvēle**

Mērķis:

Mērķis ar šo funkciju ir dot iespēju spēlētājam izvēlēties spēles tēlu ar ko grib spēlēt.

Ievaddati:

Spēlētājs noliek peli virs tēla, ko viņš grib spēlēt un nospiež peles kreiso taustiņu.

Apstrāde:

1. Spēle pārbauda vai spēlētājs ir nospiedis peles kreiso taustiņu.
2. Spēle pārbauda vai spēlētājs ir ieguvis rekordus ar izvēlēto tēlu uz katra līmeņa.

Izvaddati:

Tiek atskaņota izvēlēšanās skaņa un tiek parādīts spēles līmeņa izvēlne kā arī zem katra līmeņa tiek parādīts, katra līmeņa rekords ar izvēlēto spēlētāju.

**P.11. Spēles līmeņa izvēle**

Mērķis:

Mērķis ar šo funkciju ir dot iespēju spēlētājam izvēlēties spēles līmeni, ko viņš grib spēlēt.

Ievaddati:

Spēlētājs noliek peli virs līmeņa, ko viņš grib spēlēt un nospiež peles kreiso taustiņu.

Apstrāde:

Spēle pārbauda vai spēlētājs ir nospiedis peles kreiso taustiņu.

Izvaddati:

Tiek atskaņota izvēlēšanās skaņa un spēlētājs tiek aiznests uz spēles loga.

### 2.2.4. Spēles loga prasības

**P.12. Datorspēles “ScoreStorm” spēles uzsākšana**

Mērķis:

Mērķis ar šo funkciju ir dot spēlētājam iespēju sākt spēlēt.

Ievaddati:

Spēlētājs uzspiež ar peles kreiso klikšķis uz līmeņa, ko viņš gribētu spēlēt.

Apstrāde:

1. Spēle pārbauda vai spēlētājs ir izvēlējies vienu no spēles tēliem.
2. Spēle pārbauda vai spēlētājs ir izvēlējies vienu no spēlēs līmeņiem.

Izvaddati:

1. Spēlētājs tiek pārvietots uz spēles logu.
2. Sākas 3 sekunžu laika atskaita, pēc kuras ir dota lietotājam kontrolē par spēles tēlu un arī sāk skaitīt laiku.
3. Tiek spēlēta tēla stāvēšanas animācija.

**P.13. Datorspēles “ScoreStorm” spēles tēla kustināšana uz priekšu**

Mērķis:

Mērķis ar šo funkciju ir dot spēlētājam iespēju pakustināt viņa spēles tēlu uz priekšu.

Ievaddati:

Spēlētājs tur iešanas uz priekšu pogu, kura pēc noklusējuma ir “W”.

Apstrāde:

Spēle pārbauda vai lietotājs ir nospiedis iešanas uz priekšu taustiņu.

Izvaddati:

1. Spēlētāja tēls tiek lēnām kustināts uz priekšu.
2. Tiek spēlēta atbilstošā animācija.

**P.14. Datorspēles “ScoreStorm” spēles tēla kustināšana uz atpakaļ pusi**

Mērķis:

Mērķis ar šo funkciju ir dot spēlētājam iespēju pakustināt viņa spēles tēlu uz atpakaļ pusi.

Ievaddati:

Spēlētājs tur iešanas uz atpakaļ pogu, kura pēc noklusējuma ir “S”.

Apstrāde:

Spēle pārbauda vai lietotājs ir nospiedis iešanas uz atpakaļ taustiņu.

Izvaddati:

1. Spēlētāja tēls tiek lēnām kustināts uz atpakaļ pusi.
2. Tiek spēlēta atbilstošā animācija.

**P.15. Datorspēles “ScoreStorm” spēles tēla kustināšana pa labi**

Mērķis:

Mērķis ar šo funkciju ir dot spēlētājam iespēju pakustināt viņa spēles tēlu pa labi.

Ievaddati:

Spēlētājs tur iešanu pa labi pogu, kura pēc noklusējuma ir “D”.

Apstrāde:

Spēle pārbauda vai lietotājs ir nospiedis iešanu pa labi taustiņu.

Izvaddati:

1. Spēlētāja tēls tiek lēnām kustināts uz labo pusi.
2. Tiek spēlēta atbilstošā animācija.

**P.16. Datorspēles “ScoreStorm” spēles tēla kustināšana pa kreisi**

Mērķis:

Mērķis ar šo funkciju ir dot spēlētājam iespēju pakustināt viņa spēles tēlu pa kreisi.

Ievaddati:

Spēlētājs tur iešanu pa kreisi pogu, kura pēc noklusējuma ir “A”.

Apstrāde:

Spēle pārbauda vai lietotājs ir nospiedis iešanu pa kreisi taustiņu.

Izvaddati:

1. Spēlētāja tēls tiek lēnām kustināts uz kreiso pusi.
2. Tiek spēlēta atbilstošā animācija.

**P.17. Datorspēles “ScoreStorm” spēles skatīšanās**

Mērķis:

Mērķis ar šo funkciju ir dot spēlētājam iespēju skatīties apkārt spēles laukā izmantojot peli.

Ievaddati:

Spēlētājs pakustina datorpeli uz jebkuru virzienu.

Apstrāde:

1. Spēle pārbauda vai kamera nav sasniegusi viņa maksimālo vai minimālo Y pozīciju.
2. Spēle pārbauda vai lietotājs kustina peli.

Izvaddati:

Spēles kamera tiks pabīdīta uz to pusi, kur spēlētājs ir novietojis peli.

**P.18. Datorspēles “ScoreStorm” ieroča mērķēšana**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju ļaut spēlētājam mērķēt spēles ieroci.

Ievaddati:

Spēlētājs tur peles labo taustiņu.

Apstrāde:

Spēle pārbauda vai lietotājs tur peles labo taustiņu.

Izvaddati:

1. Spēles kamera tiek pietuvināta.
2. Tiek spēlēta mērķēšanas animācija.
3. Ierocis tiks mērķēts tik ilgi kamēr spēlētājs atlaidīs vaļā peles labo taustiņu.

**P.19. Datorspēles “ScoreStorm” ieroča šaušana**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju ļaut spēlētājam šaut spēles ieroci.

Ievaddati:

1. Spēlētājs tur peles labo taustiņu.
2. Spēlētājs nospiež peles kreiso taustiņu.

Apstrāde:

1. Spēle pārbauda vai lietotājs tur peles labo taustiņu.
2. Spēle pārbauda vai lietotājs ir nospiedis peles kreiso taustiņu.
3. Spēle pārbauda kāds ierocis ir spēlētājam.
4. Spēle pārbauda vai ierocī ir lodes.

Izvaddati:

1. Ja spēlētājam ir lodes tad tiek spēlēts atbilstošais ieroča šaušanas skaņā, tiek spēlēta šaušanas animāciju un ieroča lode tiek izšauta.
2. Ja spēlētājam nav lodes tad tiek spēlētā tukša ieroča skaņas efekts.

**P.20. Datorspēles “ScoreStorm” ieroča pārlādēšana**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju spēlētājam pārlādēt ieroci ja ir beigušās lodes.

Ievaddati:

Spēlētājs nospiež pārlādēšanas pogu, kura pēc noklusējuma ir “R”.

Apstrāde:

1. Spēle pārbauda vai lietotājs ir nospiedis pārlādēšanas pogu.
2. Spēle pārbauda vai lietotājam magazīnā ir par vismaz 1 lodi mazāk cik ir maksimums.
3. Spēlē pārbauda vai lietotājam ir vel lodes.

Izvaddati:

Ja spēlētājam ieroča magazīns ir tukšs un spēlētājam ir lodes ar, ko to pielādēt tad spēlē pārlādēšanas animāciju un spēlētāja magazīns pielādējas ar pieejamām papildu lodēm.

**P.21. Datorspēles “ScoreStorm” skriešana**

Mērķis:

Mērķis ar šo funkciju ir dot iespēju lietotājam skriet.

Ievaddati:

1. Lietotājs ir jātur jebkurš no iešanas virziena taustiņiem.
2. Lietotājs tur skriešanas pogu, kura pēc noklusējuma ir “Left Shift”.

Apstrāde:

1. Spēle pārbauda vai lietotājs staigā.
2. Spēle pārbauda vai lietotājs tur skriešanas pogu.

Izvaddati:

Spēles spēlētājs sāk skriet un tiek atskaņota skriešanas animācija.

**P.22. Datorspēles “ScoreStorm” bloķēšana**

Mērķis:

Mērķis ar šo funkciju ir dot iespēju lietotājam bloķēt

Ievaddati:

Spēlētājs nospiež bloķēšanas pogu, kura pēc noklusējuma ir “Space”

Apstrāde:

1. Spēle pārbauda vai spēles tēls nepārlādē ieroci vai arī dara trāpīšanas animāciju.
2. Spēle pārbauda vai lietotājs ir nospiedis bloķēšanas pogu

Izvaddati:

Spēles tēls izdara bloķēšanas animāciju, kura laikā ja pretinieks trāpa spēlētājam zaudētās dzīvības daudzums samazinās par 50%.

**P.23. Datorspēles “ScoreStorm” šaušana par pretinieku**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju spēlētājam šaut par pretinieku.

Ievaddati:

Spēlētājs nospiež peles kreiso taustiņu virs pretinieka.

Apstrāde:

1. Spēle pārbauda vai vietu kur lietotājs mērķē atrodas virsū uz pretinieku.
2. Spēle pārbauda vai pretinieku dzīvības vērtība pēc trāpīšanas ir lielāka par 0.

Izvaddati:

1. Ja pretiniekam trāpa spēlētājs un viņa dzīvības vērtība ir lielāka par 0, tad tiek atskaņota trāpīšanas skaņas efekts un animācija, kā arī tiek atņemtas attiecībā daudz dzīvības atkarībā no ieroča ar, ko trāpīja.
2. Ja pretiniekam trāpa spēlētājs un viņa dzīvības vērtība ir mazāka par 0, tad tiek atskaņota miršanas skaņas efekts un animācija, kā arī tiek spēlētājam pieskaitītas 5 sekundes pie laika, un tiek pieskaitīts +1 spēlētāja “kombo” rādītājam.

**P.24. Datorspēles “ScoreStorm” pretinieku uzbrukšana**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju, ka spēles laikā pretinieki uzbruks spēlētājam.

Ievaddati:

Spēlētājs sāk spēli

Apstrāde:

1. Spēle pārbauda vai lietotājs atrodas tuvi pretiniekiem.

Izvaddati:

Ja spēlētājs atrodas tuvi pretiniekiem, tad viņi sāks lēnam uzbrukt, bet ja neatrod tad viņi stāvēs uz vietas.

**P.25. Datorspēles “ScoreStorm” pretinieku parādīšanās**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju, ka spēles laikā citi pretinieki paradīsies, ja tiks nošauti pietiekami daudzi pretinieki.

Ievaddati:

Spēlētājs nošauj pretinieku.

Apstrāde:

Spēle pārbauda vai uz lauka esošā pretinieku skaits ir mazāks par 10.

Izvaddati:

Uz spēles lauka nejaušā vieta parādīsies pretinieks.

**P.26. Datorspēles “ScoreStorm” dzīvības zaudēšana**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju, ka spēles laikā spēlētājs varēs zaudēt dzīvības, ja pretinieki trāpīs.

Ievaddati:

Spēlētājam trāpa pretinieks.

Apstrāde:

1. Spēle pārbauda vai lietotājam trāpīja.
2. Spēle pārbauda vai lietotājs bloķē.
3. Spēle pārbauda vai lietotājam dzīvības daudzums pēc trāpīšanas nav mazāks par 0.

Izvaddati:

1. Ja spēlētājam trāpa pretinieks un viņa dzīvības vērtība ir lielāka par 0, tad tiek atskaņota trāpīšanas skaņas efekts un animācija, kā arī tiek atņemtas dzīvības. Ja spēlētājs bloķēja dzīvības zaudēšanas laikā, tad atņemtās dzīvības daudzums samazinās par 50%.
2. Ja spēlētājam trāpa pretinieks un viņa dzīvības vērtība ir mazāka par 0, tad tiek atskaņota miršanas animācija un spēlētājs tiek aizvests uz spēles beigu logu.

**P.27. Datorspēles “ScoreStorm” laika papildīšanas kapsulas savākšana**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju spēlētājam palielināt spēles laiku savācos kapsulas, kas atrodas uz laukuma.

Ievaddati:

* 1. Spēlētājs skatās tieši virsū uz laika kapsulu.
  2. Spēlētājs nospiež saskares pogu, kura pēc noklusējuma ir “E”.

Apstrāde:

1. Spēle pārbauda vai spēlētājs ir pietiekami tuvi laika kapsulai.
2. Spēle pārbauda vai lietotājs ir nospiedis saskares pogu.

Izvaddati:

Spēlētāja spēles laikam tiek pieskaitītas 30 sekundes

**P.28. Datorspēles “ScoreStorm” lodes savākšana**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju spēlētājam savākt lodes priekš ieročiem spēles laikā.

Ievaddati:

1. Spēlētājs skatās tieši virsū uz lodes kastes.
2. Spēlētājs nospiež saskares pogu, kura pēc noklusējuma ir “E”.

Apstrāde:

1. Spēle pārbauda vai spēlētājs ir pietiekami tuvi lodes kastei.
2. Spēle pārbauda vai lietotājs ir nospiedis saskares pogu.
3. Spēle pārbauda kāda veida ierocim pieskaitīt lodes.

Izvaddati:

Spēlētāja lodēm tiek pieskaitīts klāt 30 lodes.

**P.29. Datorspēles “ScoreStorm” “kombo” skaitītājs**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju spēlētājam aktivizēt “kombo” skaitītāju, kurš spēles beigās dos lielu bonusu.

Ievaddati:

Spēlētājs nošauj pretinieku.

Apstrāde:

Spēle pārbauda vai lietotājs ir nošāvis divus pretiniekus vismaz 10 sekunžu laika intervālā.

Izvaddati:

Spēlētājam uz ekrāna paradās “kombo” skaitītājs.

**P.30. Datorspēles “ScoreStorm” “kombo” skaitītāja pazušana**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju spēlētājam pazaudēt iegūto “kombo”.

Ievaddati:

Spēlētājs nenošauj pretiniekus.

Apstrāde:

1. Spēle pārbauda vai ir pagājušas 10 sekundes bez pretinieka nomiršanas.
2. Spēle pārbauda vai iegūtā “kombo” vērtība ir lielāka par spēlē iegūto lielāko “kombo” vērtību.

Izvaddati:

Spēlētājam pazūd “kombo” skaitītājs un ja tā vērtība ir lielāka par tagadējo spēlē iegūto “kombo” tad šī vērtība pārraksta otru un tiek saglabāta.

**P.31. Datorspēles “ScoreStorm” spēles beigšana**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju spēlei beigties.

Ievaddati:

Spēles laiks beidzas vai arī tiek nošauti visi spēlē pieejamie pretinieki, vai spēlētājs ir nomirs.

Apstrāde:

1. Spēle pārbauda vai ir beidzies spēles laiks.
2. Spēle pārbauda vai ir nošauti visi pretinieki.
3. Spēle pārbauda vai ir nomiris spēlētājs.

Izvaddati:

Parādās spēles beigu ekrāns un beidzās spēle.

**P.32. Datorspēles “ScoreStorm” iziet uz sākumekrāna poga**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju spēlētājam pārlādēt ieroci ja ir beigušās lodes.

Ievaddati:

Spēlētājs nospiež uz pogu “Iziet uz sākumekrānu”

Apstrāde:

1. Spēle pārbauda vai lietotājs ir nospiedis pogu.
2. Spēle pārbauda vai lietotājs atrodas beigu ekrānā.

Izvaddati:

Spēlētājs tiek aizvests uz sākumekrānu.

**P.33. Datorspēles “ScoreStorm” punktu saskaitīšana**

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju ka iegūtie punkti spēles laikā tiks saskaitīti.

Ievaddati:

Spēlētājs pabeidz spēli.

Apstrāde:

1. Spēle pārbauda vai lietotājs beidz spēli.
2. Spēle saskaita punktus izmantojot formulu: Punkti = (atlikušais laiks sekundēs + nošauto pretinieku skaits) \* lielākais spēle iegūtais kombo.
3. Spēlē pārbauda vai lietotāja iegūtais punktu daudzums ir jauns rekords uz tā līmeņa un ar to cilvēku.

Izvaddati:

Tiek parādīti spēlē iegūtie punkti un ja bija iegūts jauns rekords, tad spēle to arī parādā, kā arī to saglabā.

## 2.3. Sistēmas nefunkcionālās prasības

Šajā nodaļa tiks aprakstītas visas spēles nefunkcionālās prasības.

### 2.3.1. Valoda

Datorspēlei ir jābūt izstrādātai Latvijas Valsts Republikas oficiālā valodā.

### 2.3.2. Saskaņotība

Datorspēlei ir jābūt viegli saprotamai un izmantot tā, lai spēlētājām nebūtu problēmas vai jautājumi datorspēles spēlēšanas laikā. Datorspēles kontrolēm ir jābūt ērtām, loģiskām un ergonomiskām

### 2.3.3. Vizuālais izskats

Datorspēlei ir jābūt vienkāršam vizuālam izskatam, kur nav nekārtības. Vizuālam dizainā pamatā būs divas krāsas balts ar zilu.

### 2.3.4. Datorspēles veiktspējā

Datorspēlei ir jābūt tik optimizētai ka, tā varētu vismaz strādāt uz datoriem, kuriem ir 7 gadu vecs aprīkojums. Spēle tiek uzskatīta, ka tā labi iet ja uz dotā datora tā iet ar ātrumu vismaz 60 kadri sekundē.

### 2.3.5. Spēles skaņas

Spēles mūzikai pēc žanra ir jābūt ātrai un enerģiskai. Spēles skaņas efektiem ir jābūt reālistiskam un atbilstošām spēles objektam, kuram piemīt tā skaņa.

## 2.4. Gala lietotāja raksturiezīmes

Datorspēles “ScoreStorm” galvenais spēlētāju grupa ir jaunieši no 15 līdz 20 gadu vecumam. Šī vecuma grupa tika izvēlēta, jo parasti šīs vecuma grupas cilvēki visbiežāk spēlē trešās personas šaušanas spēles. Šīs datorspēles galvenais spēlētāju grupa arī iekļauj cilvēkus, kuriem ir jau pieredze spēlējot šāda veida spēles un arī cilvēkiem, kuriem ir pieredze spēlējot šāda veida punktu krāšanas spēles.

Šī spēle ir izstrādāta tā, lai tā varētu turēt spēlētāju interesi uz ilgu laiku un likt viņiem gribēt spēlēt spēli ilgu laiku ar spēles punktu sistēmu. Spēle ir arī izstrādāta saprotama un ērta priekš spēlētājiem tā, lai viņiem nerastos problēmas kaut ko saprast, kamēr spēlējot spēli.

# 3. Izstrādes līdzekļu, rīku apraksts un izvēles pamatojums

Šajā nodaļā tiks aprakstīti izstrādes rīki, kuri tika izmantoti datorspēles izstrādes laikā, kā arī šajā nodaļā ir aprakstīts, kādi vel izstrādes rīki varēja būt izmantoti izstrādes laikā nevis tie kuri tika izmantoti.

## 3.1. Izvēlēto risinājuma līdzekļu un valodu apraksts

Izstrādājot šo datorspēli tika izmantoti daudzi līdzekļi kuri padarīja izstrādes procesu daudz vieglāku.

**Spēļu dzinis – Unity:**

Projekts tika izstrādāts izmantojot Unity spēļu dzini, kura versija izstrādes laikā bija tā laika ilgtermiņa atbalstu versija 2022.3.16f1. - Šo spēļu dzini galvenokārt izvēlējās projektu izstrādātājs, jo jau ir bijusi pieredze ar šo spēles dzini. Vel viens iemesls kāpēc tieši šis spēļu dzinis tika izvēlēts bija, jo Unity piedāvā Unity Asset Store, kur spēļu izstrādātāji var publicēt gan arī lejupielādēt spēļu modeļus, skaņas un lietotāja interfeisa elementus. Šos spēles elementus pēc tam ir iespējams viegli importēt iekšā eksistējošos projektos un arī rediģēt ja ir vajadzīgs. Pēdējais iemesls kāpēc tika izvēlēts tieši šī spēļu izstrādes vide bija, jo Unity salīdzinot ar citiem spēļu dziņiem ir daudz vieglāk ejams uz mazāk jaudīgiem datoriem nekā citi spēļu dziņi.

**Programmēšanas valoda – C#:**

Tāpēc, ka tika izmantots Unity spēļu dzinis tas nozīmē, ka programmēšanas valoda, kura tika izmantota datorspēles izstrādei bija C# programmēšanas valoda. C# ir objektorientēta programmēšanas valoda, kas nozīmē, ka tā ir ļoti labi piemērota priekš datorspēļu izstrādes, jo datorspēles tiek veidotas, pamatojoties uz objektiem un to mijiedarbību. Tādēļ tika izmantota šī valoda, jo datorspēle ir ļoti objektorientēta tādēļ bija vajadzīgs izvēlēties valodu, kas ir objektorientēta.

**Izstrādes vide – Visual Studio:**

Projekts tika izstrādāts izmantojot Visual Studio izstrādes vide. Viens no galvenajiem iemesliem, kāpēc šī vide tika izvēlēta, jo to Unity piedāvā, kā vienu no divām izstrādes vidēm, ko izvēlēties veidojot projektu. Tā kā izstrādes vidi piedāvā Unity tas nozīmē, ka tam arī būs laba savietojamība ar Unity projektiem un tas nozīmē, ka būs mazāk problēmas starp Unity un Visual Studio sazināties nekā izmantojot izstrādes vidi, kura varbūt neatbalsta Unity. Pēdējais iemesls kāpēc tika izvēlēts Visual Studio bija tāpēc, ka salīdzinot ar citām izstrādes vidēm Visual Studio ir daudz vairāk pielāgošanas iespējas nekā ar citām izstrādes vidēm.

**Dokumentu rakstīšanas un testpiemēru rakstīšanas rīks - Microsoft Office:**

Projekta dokumentācija, kā arī testpiemēri tika rakstīti izmantojot Microsoft Office komplektu. Projekta dokumentācija tika rakstīta izmantojot Microsoft Word un testpiemēra tika rakstīti izmantojot Microsoft Excel. Vienīgais iemesls kāpēc šie rīki tika izvēlēti un nevis citi bija tāpēc, ka šiem rīkiem ir daudz labākas un vairākas tekstu formatēšanas opcijas, kuras bija obligāti jāaizmanto, jo dokumentācijai bija obligāts teksta formatēšanas standarts, kurš bija jāievēro.

**Projektu failu glabātuve – GitHub:**

Vietne kura tika izvēlēta priekš projektu failu glabāšanas un versonēšanas bija GitHub. Tika izvēlēta šī vietne, jo GitHub ir tieši paredzēts priekš programmēšanas projektu glabāšanas un tas nozīmē, ka tas atbalsta Unity projektus un ļauj tos versonēt. GitHub arī tika izvēlēts, jo salīdzinot ar citiem projektu failu glabātuvēm, augšupielādēt, lejupielādēt un versonēt projektu ir ļoti viegli un vienkārši izdarīt salīdzinot ar citām vietnēm. Vel viens iemesls kāpēc tika izvēlēts GitHub bija tāpēc, ka GitHub neglabā pilnībā visus projekta failus, jo GitHub glabā tikai nepieciešamo. Kad atver pirmo reizi projektu, Unity arī izveido vairākus failus, kuri tikai eksistē, ja Unity varētu savienoties ar projektu. Bet, GitHub šos failus nesaglabā samazinot projektu failu apjomu serverim, piemēram, datorspēles “ScoreStorm” projektu faili ir 5 GB lieli, bet projektu faili, kas tiek glabāti GitHub ir tikai 100 MB, jo vietne tikai glabā svarīgo. Šis ir labi priekš lieliem projektiem, jo katri reizi, kad lietotājs grib lejupielādēt vai augšupielādēt projektu viņām nav visu laiku jāaugšupielādē vairāku gigabaitu liels fails.

**Diagrammas veidošanas rīks – Draw.io:**

Rīks, kurs tika izmantots priekš diagrammu veidošanas bija tiešsaistes rīks draw.io. Vienīgais galvenais iemesls kāpēc šis rīks tika izvēlēts bija, jo projektu izstrādātajam jau ir pieredze ar šo rīku un šis rīks jau dara visi vajadzīgo, ko viņam vajadzētu priekš projektu izstrādes, Bet viena ļoti laba funkcija, kas ir draw.io ir tas, ka pēc diagrammu uzzīmēšanas ir pēc tam iespējams pārbīdīt un mainīt augšupielādējot diagrammu uz draw.io.

## 3.2. Iespējamo risinājuma līdzekļu un valodu apraksts

Izstrādājot datorspēli, projekta izstrādātajam bija arī liela izvēle ar alternatīviem rīkiem, ko izmantot datorspēles izstrādei, bet šiem rīkiem arī bija daži iemesls kāpēc tieši tie tika neizvēlēti.

**Spēļu dzinis – Unreal Engine:**

Veidojot projektu varēja tikt izmantots Unreal Engine spēļu dzinis Unity vietā, bet bija divi iemesli kāpēc projektu izstrādātājs šo dzini neizvēlējās. Pirmkārt, Unreal izstrādātas spēles prasa daudz vairāk datora resursus nekā Unity. Viena spēles prasība bija tā ka spēlei vajadzētu iet uz datoriem, kuriem nav visjaunākie datora komponenti. Piemēram, Unity minimālais vajadzīgais RAM, lai palaistu uz datora ir 8 gigabaiti, kamēr Unreal ir 16 gigabaiti. Otrais iemesls, kāpēc netika izvēlēts Unreal bija tāpēc, ka projektu izstrādātajam nebija nekādas pieredzes izmantojot Unreal, kamēr bija pieredze ar Unity. Tā kā bija dots termiņš projektu izstrādei, projektu izstrādātājs negribēja riskēt iespēju, ka visas prasības netiks realizētas, jo nezinātu kā visu izdarīt.

**Spēļu dzinis – Godot:**

Veidojot projektu varēja arī tikt izmantots Godot spēļu dzinis nevis Unity bet bija divi iemesli kāpēc projektu izstrādātājs negribēja izvēlēties šo spēļu dzinis. Pirmais iemesls, kāpēc netika izvēlēta šis spēļu dzinis, bija tāds pats iemesls, kāpēc Unreal Engine netika izmantots, un tas bija tāpēc, ka projektu izstrādātājam nebija pieredze ar šo spēļu dzinēju. Otrais iemesls kāpēc projektu izstrādātājs neizvēlējās šo spēļu dzini bija tāpēc ka salīdzinot ar Unity vai pat ar Unreal ir tas ka Godot ir daudz jaunāks. Tas ir slikti, jo, ja projektu izstrādātājam ir problēmas ar Godot vai viņš nezina, kā kaut ko realizēt, interneta resursi, kas palīdzētu to salabot, būs daudz mazāki nekā ar Unity vai Unreal. Tā rezultātā varētu rasties situācija, ka projektu izstrādātājs nevarētu pabeigt projektu.

**Izstrādes vide – MonoDevelop:**

Kad izveidot Unity projektu tiek automātiski izvēlēta Visual Studio izstrādes vide, bet ir iespējas izvēlēties arī MonoDevelop kā izstrādes vidi projektu iestatījumos, bet bija divi iemesli kāpēc netika izvēlēta. Pirmais koda atkļūdošanai ir daudz sliktāk MonoDevelop nekā Visual Studio. Jo Unity ir vairāk integrēts ar to nekā ar MonoDevelop, jo MonoDevelop nav tik aktīvi uzstūrēts nekā Visual Studio. Otrais iemesls kāpēc MonoDevelop netika izvēlēts bija, jo salīdzināt ar Visual Studio, MonoDevelop ir daudz mazāk personalizācijas opcijas nekā ar Visual Studio.

**Programmēšanas valoda – C++:**

Tika izvēlēts C# programmēšanas valoda, jo tika izmantots Unity spēļu dzinis, bet ja projektu izstrādātājs būtu izvēlējies Unreal vai Godot tad būtu ticis izmantots C++. Projekts varēja tikts izstrādāts C++ programmēšanas valodā ar tām pati funkcijām, kas ir C# Unity versijā, bet beigās netika izvēlēts, jo Unity to tik labi neatbalsta.

**Dokumentu rakstīšanas un testpiemēru rakstīšanas rīks – Libre Office:**

Dokumentācija un testpiemēri tika rakstīti izmantojot Microsoft Office, bet netika izmantojot Libre Office dēļ viena iemesla. Iemesls kāpēc netika izvēlēts Libre Office bija jo dokumentācijai un testpiemēriem bija svarīgas formatējuma prasības, kuras nevarēja būt realizētas Libre Office, jo tur nav tik daudz formatēšanas rīki.

**Projektu failu glabātuve – GitLab:**

Veidojot projektu varēja tikt izmantots GitLab nevis GitHub priekš projektu failu glabāšanas, bet netika izvēlēts priekš viena iemesla. Iemesls kāpēc netika izvēlēts GitLab bija, jo salīdzinot ar GitHub ir daudz primitīvāks un nav tikt daudzas funkcijas. Un iemesls kāpēc GitLab nav tik daudz funkcijas kā GitHub ir, jo GitLab ir mazāk populārāks nekā GitHub.

**Diagrammas veidošanas rīks – Figma:**

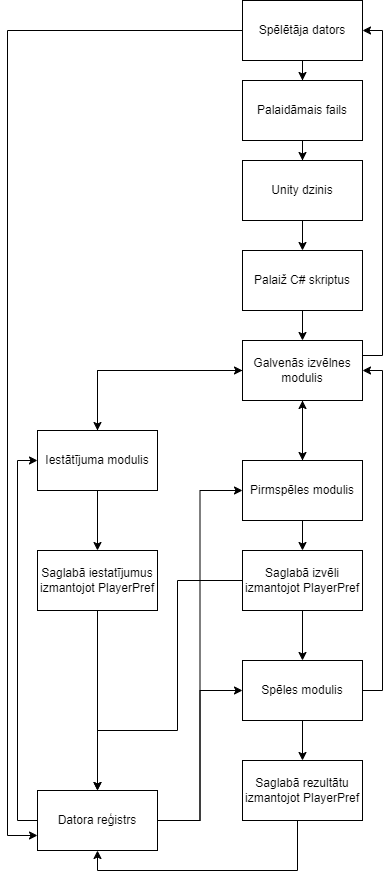
Figma varēja tikt izmantota Draw.io vietā, bet beigās netika izmantota tāpēc, ka Figma nav tik daudzi diagrammu zīmēšanas rīki nekā Draw.io. Tas nozīmētu, ka būtu bijis daudz grūtāk uzstatīsit diagrammas izmantojot Figma nevis Draw.io.

# 4. Sistēmas struktūras modelēšana un projektēšana

Šajā nodaļa atrodas visas diagrammas saistītas ar projektu.

## 4.1. Sistēmas struktūras modelis

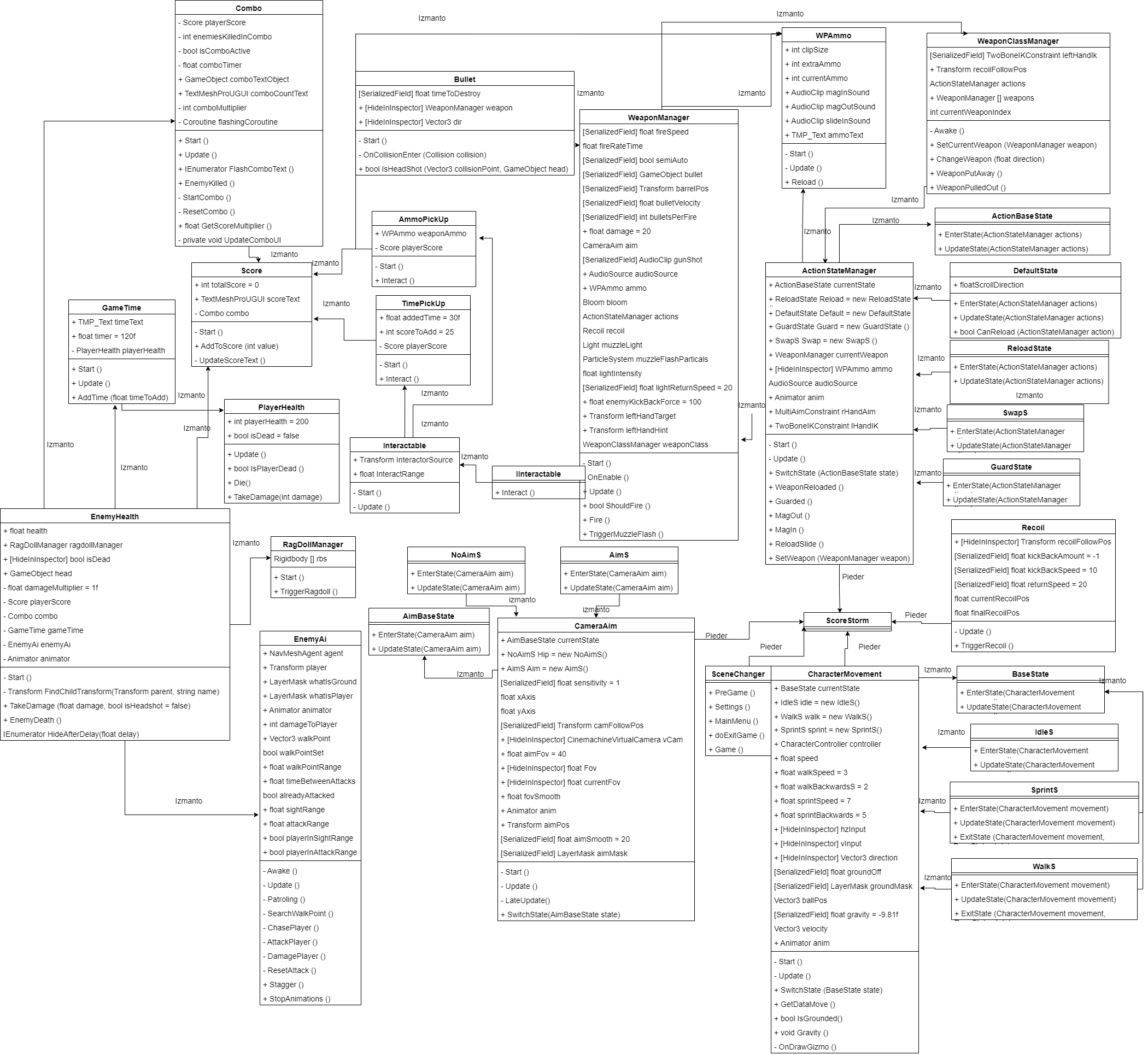
Šajā diagramma var apskatīties kāda ir datorspēles sistēmas darbības princips. Diagramma var apskatīties, kādas darbības notiek, kad spēlētājs atver spēli uz savu datoru, kas notiek ka pārej starp moduļiem un arī kā notiek vajadzīgās informācijas saglabāšana, kā arī kur to lasa.



**1.attēls Sistēmas struktūras modelis**

## 4.2. Klašu diagramma / ER diagramma

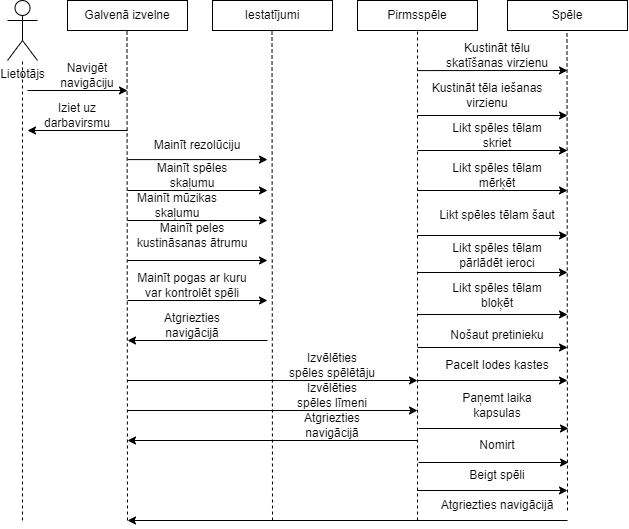
Šajā diagramma var apskatītas kādu ir saistība starp datorspēlēs klasēm šajā klases diagrammā.



**2.attēls klašu diagramma**

## 4.3. Funkcionālais un dinamiskais sistēmas modelis

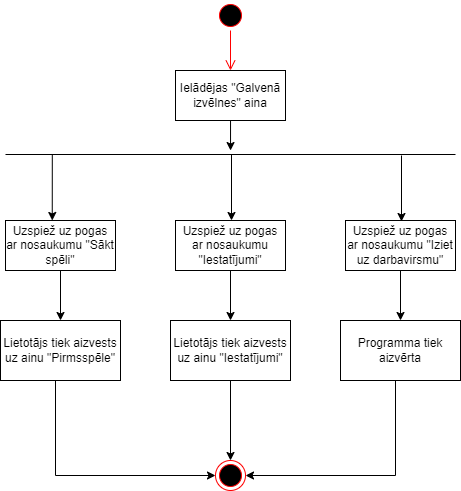
Šajā diagramma var apskatīt kāda izskatās datorspēles darbība un kādas ir lietotāja iespējas sekvenču diagrammā.



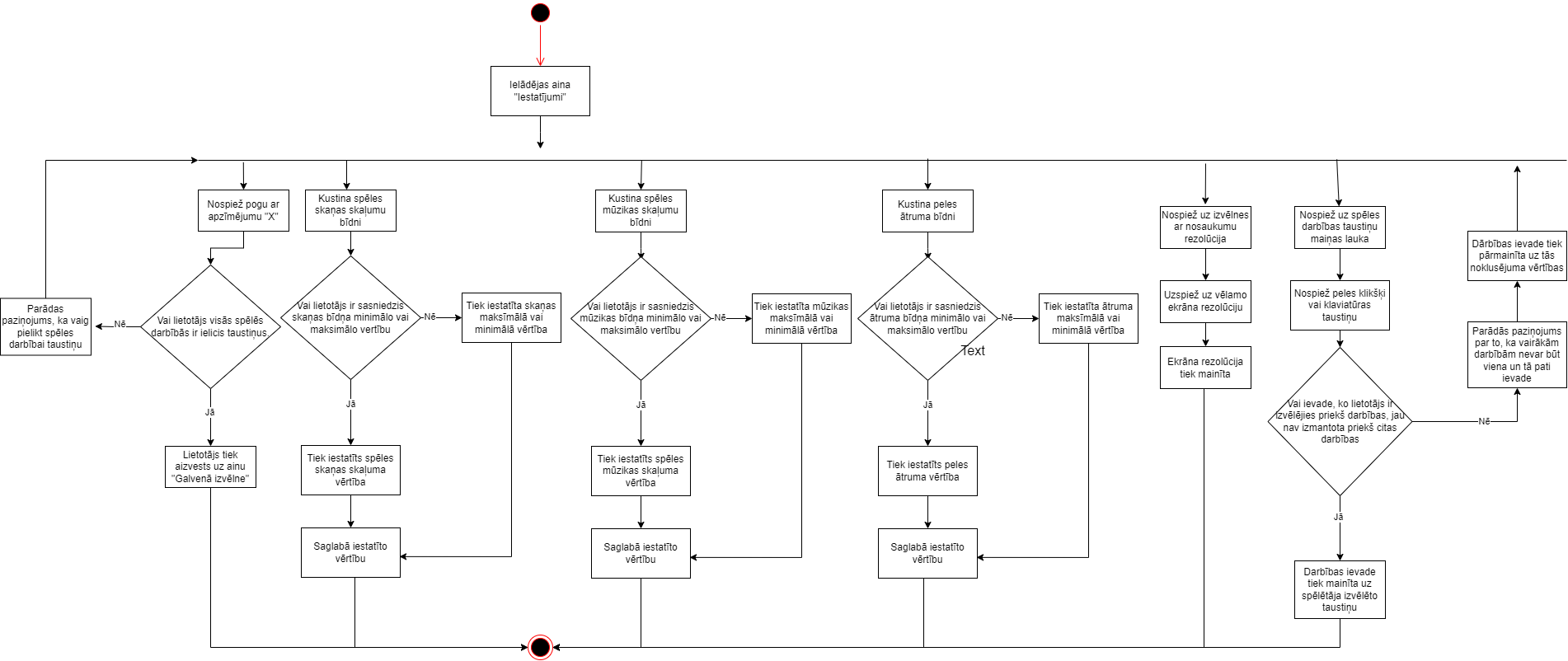
**3.attēls Sekvenču diagramma**

## 4.4. Aktivitāšu diagramma

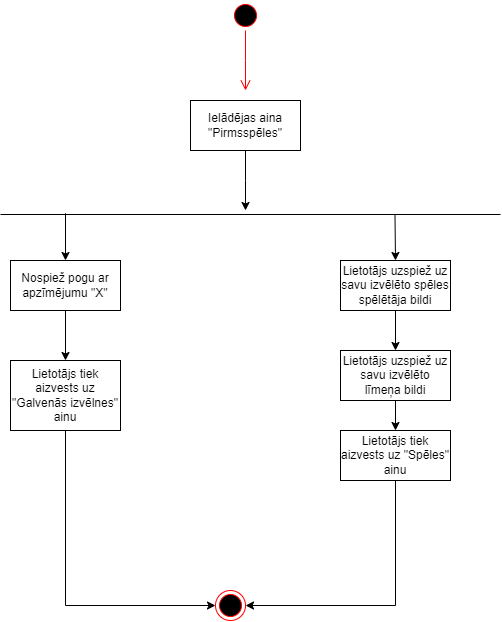
Šajā nodaļa var apskatīt aktivitātes diagrammas priekš katriem programmas moduļiem.



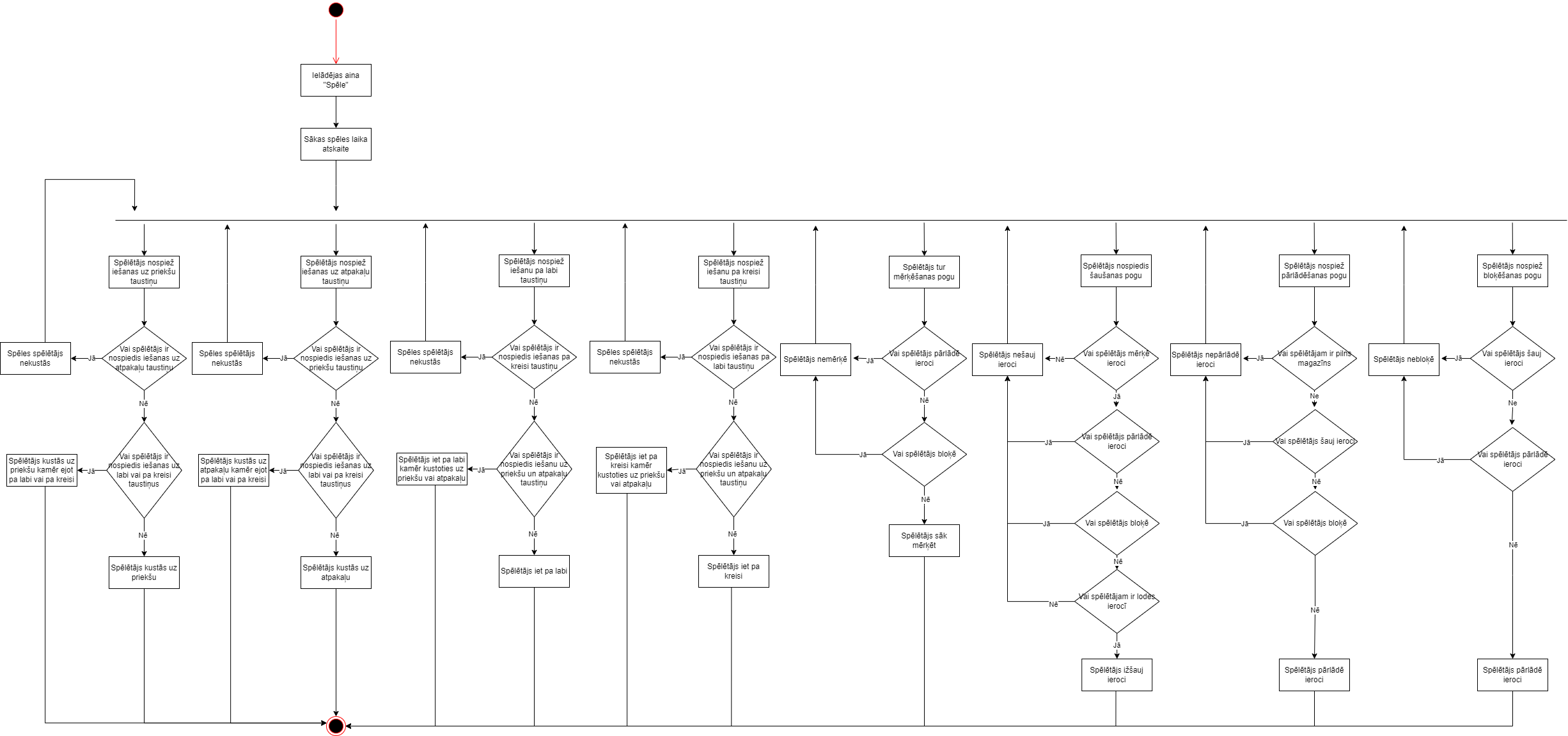
**4.attēls Aktivitātes diagramma galvenai izvēlne**



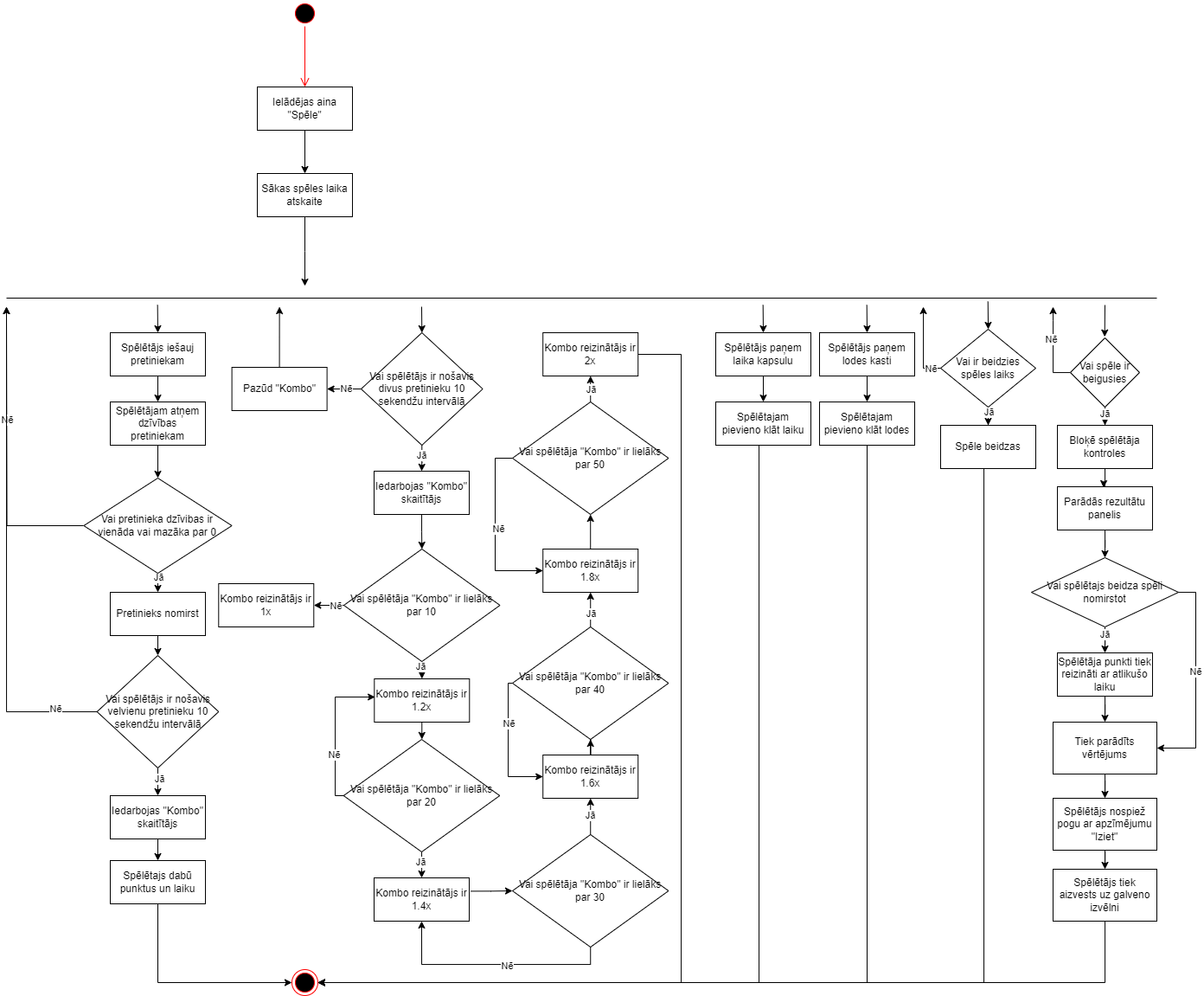
**5.attēls Aktivitātes diagramma iestatījumiem**



**6.attēls aktivitātes diagramma pirmspēles logam**

****

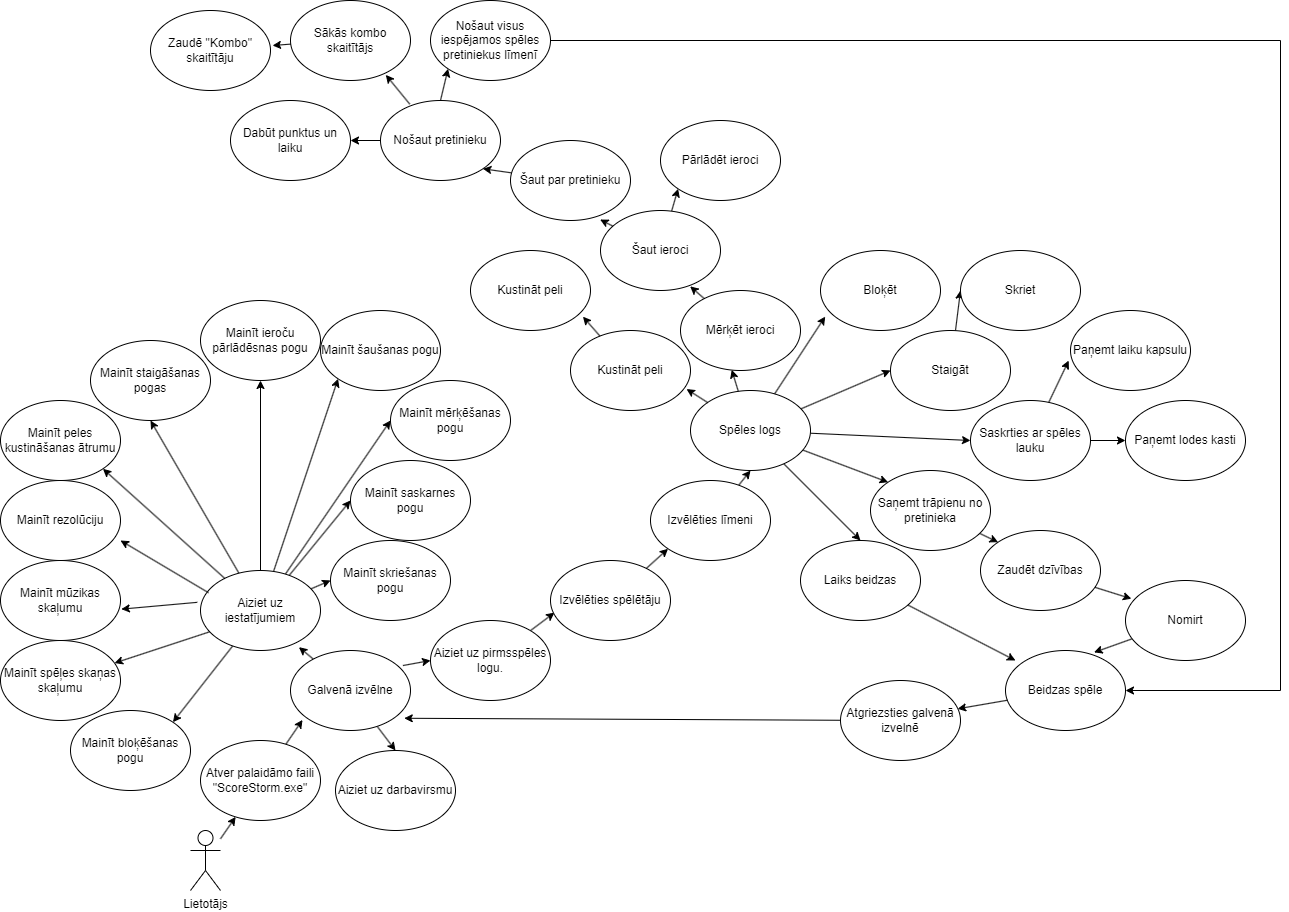
**7.attēls aktivitātes diagramma spēles logam pirmā daļa**

****

**8.attēls aktivitātes diagramma spēles logam otrā daļa**

## 4.5. Lietotjumgadījuma diagramma

Šajā diagramma var apskatīt kādas ir iespējamas spēlētāju darbības spēlējot spēli.



**9. attēls Lietotjumgadījuma diagramma**

## 4.6. Sistēmas moduļu apraksts un algoritmu shēmas

Sistēma sastāv no četriem programmas moduļiem, kuri ir galvenā izvēlne, iestatījumi, pirmspēles un spēles modulis. Kopā ir četri moduļi, kur katram ir speciāls iemesls un katrs modulis ir sadalīts kā aina programmā.

Galvenā izvēlnē ir viss vienkāršākais modulis, tāpēc ka tās funkcija ir būt kā navigācijai un arī tur atrodas tikai trīs pogas, kuru funkcijas ir pārvest lietotāju starp ainām vai aizvērt ciet programmu. Divas pogas, kuras aizved lietotāju uz citām ainām ir “Sākt spēli” un “Iestatījuma” pogas, kuras pēc nospiešanas aizvedīs lietotāju uz pirmspēles un iestatījuma ainām attiecīgi. Ainu pāreja notiek izmantojot iebūvēto Unity funkciju, kuru sauc par SceneManager, kura pēc aktivizēšanas ielādēs izvēlēto ainu un izdzēsīs iepriekšējo ainu. Pēdējā poga modulī ir “Iziet uz darbvirsmu” poga, kuru pēc nospiešanas izdara Unity iebūvēto funkciju Quit, kura uz reiz aizver ciet programmu.

Iestatījuma moduļa galvenais uzdevums ir nodrošināt lietotājam iespēju mainīt spēles iestatījumus izmantojot pogas, bīdņus un izvēlnes. Lietas, kuras lietotājam ir atļautas mainītas šajā modulī ir spēles skāņas efektu skaļums, spēles mūzikas skaļumu, peles kursora ātrumu, spēles ekrāna rezolūciju, kā arī mainīti visas spēles darbības ievades ar citām pogām. Visi iestatījumi saglabā savu vērtību izmantojot Unity PlayerPrefs funkciju. Šī funkcija ir laba tāpēc kā tā ļauj saglabāt int, string, float datu tipu vērtības ne tikai caur ainām, bet arī aizverot ciet programmu vērtības paliks uz lietotāju datoru. Šī funkcija saglabā vērtības uz lietotāja datoru izmantojot datora reģistru, jo programma saglabā vērtības kā reģistru atslēgu. Pēc tam, kad vērtības tiek saglabātas uz lietotāja datora, tajā laikā, kad lietotājs sāks spēli visas saglabātās vērtības tiks iestatītas spēlē un mainīs attiecīgo.

Pirmspēles moduļa galvenā funkcija ir dot lietotājam iespēju izvēlēties spēles spēlētāju un spēles līmeni, kur grib spēlēt. Modulī pagaidām atrodas tikai četras pogas, divas priekš spēlētāju izvēles un divi priekš līmeņu izvēles, bet nākotnes versijās tās būs vairākas. Šīs pogas izmanto to pašu funkciju, kuras izmanto pogas iestatījumos un tas ir PlayerPrefs. Kur kad lietotājs izvēlās spēlētāju tā vērtība tiek saglabāta un tā pat ir ar līmeņu izvēli. Kad spēle tiek sākta, tad spēle pārbauda, kādas vērtības tika izvēlētas un tad ielādē attiecīgos spēles objektus spēlē.

Spēles modulis ir viss lielākais modulis programmā, jo tur atrodas spēles galvenā daļa. Spēles spēlētājs var staigāt, skriet, noklusējuma ,mērķēt, pārlādēt, bloķēt, ieroča maiņa un nomirt, visas šīs darbības ir sadalītas stāvokļos, kur spēles tēls atrodas, kad lietotājs izdara kaut kādu komandu. Šie stāvokļi dara katrs dara savu funkciju, bet visiem stāvokļiem ir daži ierobežojumi. Staigāšanas stāvoklis ļauj lietotājam kustēties, skriešanas stāvoklis ļauj lietotājam kustēties, bet ātrāk. Noklusējuma stāvoklis ir stāvoklis, kur lietotājs ienāk, ja neko nedara. Mērķēšanas stāvoklis dod iespēju lietotājam šaut ar ieroci. Pārlādēšanas stāvoklis ļauj lietotājam pārlādēt ieroci, bet šajā stāvoklī ir aizliegts bloķēt, mainīt ieroci un šaut. Bloķēšanas stāvoklis ļauj bloķēt, bet aizliedz šaut, mainīt ieroci un pārlādēt. Ieroča maiņas stāvoklis maina lietotāja ieroci uz citu un aizliedz šaut, bloķēt un pārlādēt. Nomiršanas stāvoklis neļauj lietotājam neko darīt. Kad lietotājs ieiet ikvienā stāvoklī tad arī uzreiz tiks spēlēta attiecīgā animācija un lietotājs izies no stāvokļa, kad lietotājs izdarīs konkrētu darbību vai arī tiks automātiski iziets no stāvokļa pēc darbības pabeigšanas. Lietotājs arī var saskarties ar spēles dažiem objektiem, kā laika kapsulas un lodes kastes izmantojot interactable funkciju un spēlētāja sadursmi ar grīdu un sienu izmanto slāņus, lai saprastu, kam var iet cauri un kam nevar.

Stāvokļa sistēma tā pati strādā priekš pretiniekiem, bet viņiem ir tikai trīs stāvokļi un tie ir meklēšanas, ķeršanas, un arī uzbrukšanas stāvoklis. Meklēšanas stāvoklī pretinieks staigā par noteikto zonu. Ja pretinieks atrod lietotāju vai lietotājs šauj, tad pretinieks ieiet ķeršanas stāvoklī, kur pretinieks skrien pakaļ lietotājam. Ja pretinieks ir ticis noteiktā attālumā no spēlētāja, tad pretinieks ieiet uzbrukšanas stāvoklī, kur pretinieks centīsies uzbrukt lietotājam, bet ja spēlētājs būs izgājis no uzbrukšanas distances, tad pretinieks ieies atpakaļ ķeršanas stāvoklī. Pretinieks var zināt, kur jāiet izmantojot NavMesh komponentu priekš Unity, kas ļauj pretiniekiem saprasti, kas ir zeme un, kas ir objekts, kuram ir vajadzīgs apiet apkārt. Ja pretinieks tiek nošauts, tad tiek aktivētas pretinieka modeļa ragdoll un pēc 5 sekundēm tas pazūd. Kad tiek nošauts pretinieks sākas pārbaude, kas pārbauda vai divi pretinieki tika nošauti 10 sekunžu intervālā, ja ir tad sākas “kombo” skaitītājs, kura laikā lietotāja iegūtie punkti tiek reizināti, bet ja nē tad skaitītājs pazūd.

Kad beidzas spēles tiek apskatīts vai spēle beidzās tāpēc ka lietotājs nošāva visus iespējamos pretiniekus uz lauka, vai arī bija beidzies laiks, vai spēlētājs arī nomira spēles laikā. Ja spēlētājs nošāva visus pretiniekus tad iegūtais punktu daudzums spēles laikā tiek summēts ar atlikušo laiku sekundēs, bet ja nē tad rezultāts ir cik daudz punkti tika iegūti spēles laikā. Kad spēle ir beigusies tiek paradīts panelis, kur ir parādīti spēles rezultāti. Tiek parādīts beigu rezultāts, tiek pateikts vai tas ir bijis jauns rekords un arī tiek parādīts vērtējums par to cik labi lietotājs ir spēlējis. Iegūtais rezultāts un vērtējums tiek saglabāts tikai tad, kad lietotājs ir ieguvis jaunu rekordu, bet ja nav tad rezultāts tiks pazaudēts. Spēles beigu panelī atrodas arī poga pēc kuras nospiešanas lietotājs tiek aizvests atpakaļ uz galveno izvēlni.

# 5. Lietotāju ceļvedis

Šajā nodaļā ir aprakstīts lietotāja ceļvedis priekš datorspēles “ScoreStorm”, kur ir aprakstīts kā lietotājs var izmantot programmu un arī, ko katra poga vai lauks dara.

## 5.1. Galvenā izvēlne

Galvenā izvēlne ir programmas sadaļa, ko lietotājs ieraudzīs katru reizi ka atvērs vaļā programmu. Galvenā izvēlne sastāv no datorspēles nosaukuma, programmas izstrādātāja vārds un trīs pogas, kur katrai ir sava funkcija. Pirmā poga ir poga ar nosaukumu “Sākt spēli”, kad lietotājs nospiež šo pogu viņš tiks aizvests uz programmas pirmspēles logu. Otrā poga galvenā izvēlne ir poga ar nosaukumu “Iestatījumi”, kur pēc nospiešanas lietotājs tiks aizvests uz iestatījumu logu. Pēdējā poga ir ar nosaukumu “Iziet uz darbvirsmu”, kur pēc nospiešanas programma tiks aizvērta.



**10.attēls galvenās izvēlnes logs**

## 5.2. Iestatījumi

Iestatījumi ir programmas sadaļa, kur lietotājs var mainīt vairākas spēles iestatījumus, lai lietotājs varētu personalizēt savu spēles gaitu. Pirmkārt iestatījuma logā atrodas trīs bīdņi ar, ko lietotājs var kontrolēt spēles mūzikas skaļumu, spēles skaņas skaļumu, kā arī spēles kameras ātrumu. Ja lietotājs kustinās bīdni uz leju tad skaņas vai ātrumu vērtība tiks samazināta, bet ja bīdni kustinās uz augšu tad tā tiks palielināta. Iestatījuma logā vel arī atrodas izvēles lauciņš, kur spēlētājs var izvēlēties spēles rezolūciju. Visas rezolūcijas opcijas, būs ņemta, no lietotāja monitora atbalstītām rezolūcijām. Pēdējā lieta, ko lietotājs var izdarīt iestatījuma logā ir mainīt spēles darbību ievades pogas. Lietotājs var mainīt iešanas, skriešanas, šaušanas, mērķēšanas, pārlādēšanas, ieroča maiņas, bloķēšanas un arī vides saskares pogas. Lietotājs var mainīt darbības ievades uzspiežot uz pogas lauka un pēc tam nospiest uz tastatūras vai peles vienu pogu, lai iestatītu to, kā ievadi. Lietotājam ir arī iespēja atgriezties galvenā izvēlnē uzspiežot uz pogas loga augšējā kreisajā stūrī ar apzīmējumu “X”.

**11.attēls iestatījumi**

## 5.3. Pirmspēles logs

Pirmspēles logs ir programmas sadaļa, kur lietotājs var sākt spēli. Pirmspēles loga apakšas vidū atrodas bildes ar spēles tēliem, kur zem tiem atrodas rekords, ko lietotājs ir ieguvis ar to spēles spēlētāju konkrētā līmenī. Lai izvēlētos līmeni lietotājam ir jānospiež vienām no divām līmeņu bildēm. Kad lietotājs ir uzlicis peles kursoru virs līmeņu bildes lapas apakšā mainīsies lietotāja iegūtais rekords tieši uz to līmeni. Kad spēlētājs ir izvēlējies līmeni, tad viņam ir dota opcija izvēlēties spēles tēlu, kura strādā tā pati, kā līmeņu izvēle. Kad lietotājs izvēlās spēles spēlētāju, tad lietotājs tiks pārnests uz spēles logu un sāksies spēle.

## 5.4. Spēle

Spēles loga lietotājs var redzēt informāciju, kas palīdzēs lietotājam spēlēt spēli. Loga augšā kreisajā stūri var redzēt spēlē iegūtos punktus, vidū var redzēt spēles atlikušo laiku un labajā stūrī var redzēt “kombo” skaitītāju, ja tas ir aktīvs. Loga apakšā kreisajā stūrī var redzēt spēlētāja atlikušās dzīvības, kuras ir parādītas kā josla un labajā stūrī lietotājs var redzēt, kādu ieroci ir izvēlējies, kā arī cik lodes tagad ir ieroča magazīnā un cik lodes kopā ir tam ierocim. Loga pašā centrā arī ir redzams maziņš punktiņš, kurš tikai ir aktīvs, tad kad lietotājs nemērķē.

Lietotājs var kontrolēt savu spēles spēlētāju izmantojot klaviatūras vai peles taustiņus visas darbības kontroles lietotājam bija iespējams mainīt iestatījumos, bet programma arī piedāvā noklusējuma ievades pogas. Lietotājs var kustināt spēlētāju par spēles lauku turot kādu no staigāšanas pogām, kuras pēc noklusējuma ir “W” uz priekšu, “S” uz atpakaļ, “A” pa kreisi un “D” pa labi. Ja lietotājs staigājot arī tur skriešanas pogu, kura pēc noklusējuma ir “Shift”, tad spēles spēlētājs sāks skriet un viņa ātrums palielināsies. Lietotājs var kontrolēt kameru izmantojot peli un to kustot. Ja lietotājs grib mērķēt ieroci, tad viņam ir jātur mērķēšanas poga, kura pēc noklusējuma ir peles labais taustiņš. Kamēr lietotājs mērķē ieroci viņam arī ir iespēja šaut ieroci nospiežot pēc noklusējuma peles kreiso taustiņu. Ja lietotājs ir izšāvis ieroci un arī vel ir lodes lai pārlādētu ieroci, tad viņš var nospiest pārlādēšanas pogu, kura pēc noklusējuma ir “R”. Lietotājs arī var samazināt pretinieka izdarīto bojājumu nospiežot bloķēšanas pogu, kura pēc noklusējuma ir “Space”. Ja lietotājs spēles laikā atrod lodes kastes vai laika kapsulas, tad nospiežot saskares pogu, kura pēc noklusējuma ir “E”, tad lietotājs to var pacelt un sev iedot mazliet punktus, kā arī vairāk lodes vai laiku. Spēles galvenais mērķis ir nošaut cik vien daudz iespējami pretiniekus cik lietotājs var. Lietotājs šaujot par pretiniekiem var šaut par jebkuru ķermeņa daļu, kur grib, bet ja šaus par pretinieku galvu, tad tas izdarīs viņiem vairāk bojājumus nekā par jebkuru citurieni, kā arī dot iespēju apstulbināt pretinieku uz īsu laiku, kur viņi nekustēsies. Ja spēlētājs nošauj divus pretiniekus 10 sekunžu intervālā, tad sāksies “Kombo” skaitītājs, kura laikā spēlētāja punkti tiks reizināti, atkarībā cik liels ir skaitītājs. Ja lietotājs nenošauj pretinieku 10 sekundēs, tad tas pazūd un punkti netiek reizināti. Lietotājam arī ir dzīvības, ko pretinieki var atņemt ja viņi izdara veiksmīgu uzbrukumu pret spēlētāju, ja dzīvības beigsies, tad spēlētājs nomirs un arī spēle beigsies.

Lietotājs var beigt spēli trīs veidos, pirmais, nošaut visus iespējamos pretiniekus, otrais, ļaut laikam beigties, trešais, nomirt dēļ pretiniekiem. Kad spēle beidzas tiek parādīts rezultātu logs, kur tiek parādīts iegūtais punktu daudzums, gan arī vērtējums, kas parāda cik labi lietotājs ir spēlējis, kā arī paziņojumu par jaunu rekordu, ja ir jauns rekords. Kad lietotājs ir apskatījis savu rezultātu viņš var nospiest pogu ar apzīmējumu “Iziet”, kur pēc nospiešanas lietotājs tiks aizvests atpakaļ uz galveno izvēlni.

# 6. Testēšanas dokumentācija

Šajā nodaļa aprakstīts kā notika datorspēles testēšana, kā arī parāda datorspēles testēšanas rezultātus.

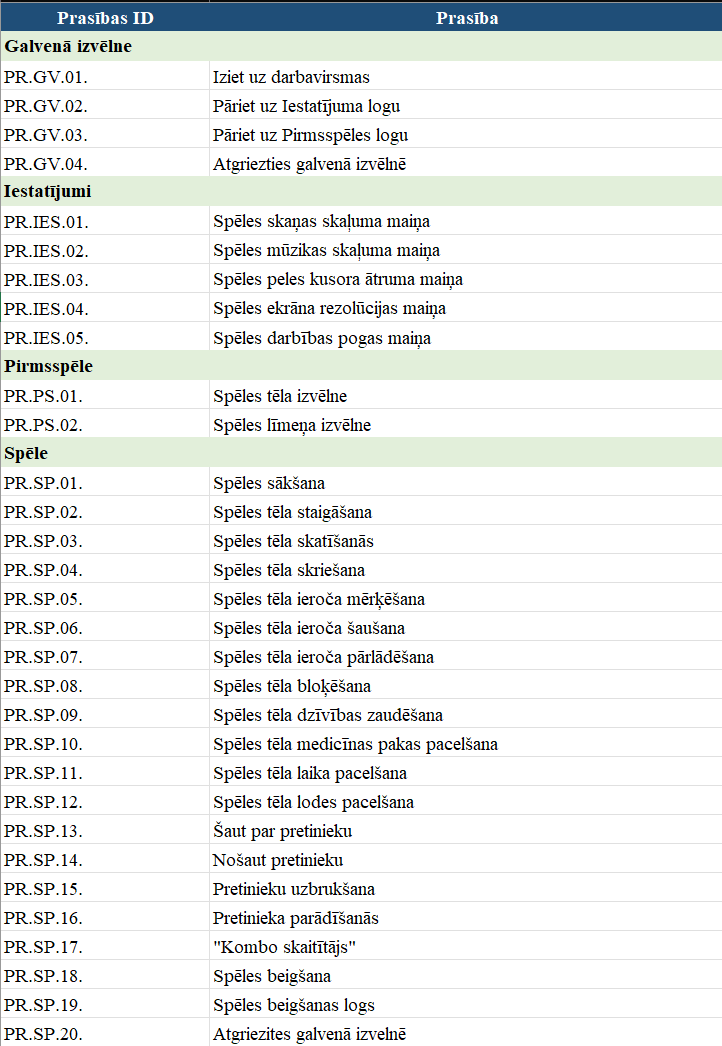
## 6.1. Izvēlētais testēšanas metodes, rīku apraksts un pamatojums

Testējot spēli, tika veikta divu veidu testēšana, melnās kastes testēšana un baltās kastes testēšana. Atšķirība starp melno un baltās kastes testēšanu ir tā ka melnās kastes testēšanā neizmanto kodu, kamēr baltās kastes testēšanā izmanto kodu, lai pārbaudītu vai programma veic visas savas vajadzīgās komandas priekš funkcijas.

Melnās kastes testēšanā nereti netiek izmantoti programmatūras rīki, jo šādā veidā tiek pārbaudīta sistēma bez jebkādas iekšējas kodu struktūras. Tādēļ, šāda testēšana var notikt bez jebkāda rīka atbalsta, un tāpēc nekādi rīki netiek izmantoti.

Baltās kastes testēšanas laikā tika izmantots iebūvētais Visual Studio atkļūdošanas rīks. Šis rīks ļāva projekta izstrādātajam sadalīt kodu daļās un ļāva pārbaudīt vai tika izpildītas visas vajadzīgās darbības priekš funkcijas izstrādes.

Kad tika veikta programmas testēšana visa informācija par testēšanu tika pierakstīta testēšanas žurnāla, kura ir parādīta nākošās divās apakšnodaļās un termini, kas tika izmantoti testēšanā var apskatīt 8. nodaļas 1. tabulā. Testēšanas žurnālā tiek arī izmantoti prasības saīsinājumi, kurus var apskatīt 12. attēlā.

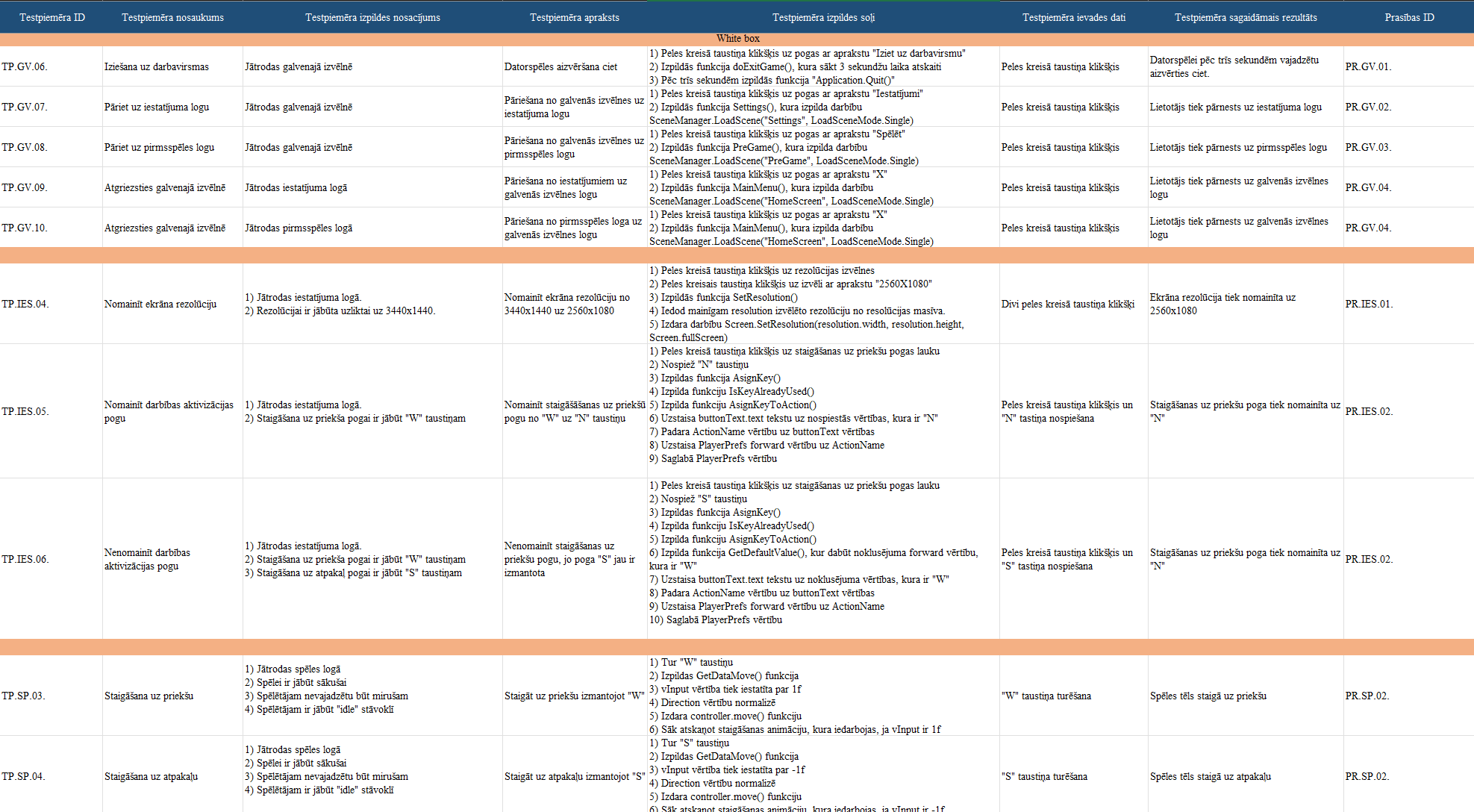


**12.attēls prasības testēšanai**

## 6.2. Testpiemēru kopa

Attēls, kurā ir teksts, fonts, cipars, rinda

Apraksts ģenerēts automātiski**13.attēls melnās kastes testpiemēri**

**14.attēls baltās kastes testpiemēri**

## 6.3. Testēšanas žurnāls

**15.attēls melnās kastes testpiemēri**

# 7. Secinājumi

Secinājumā, datorspēle “ScoreStorm”, manuprāt, bija izstrādāta līdz manu pašu spēju ietvariem un visi, ko es gribēju izveidot priekš projekta, manuprāt, tika izstrādāts ļoti labā stāvoklī. Visas prasības un funkcijas, kuras tika aprakstītas šajā dokumentā tika realizētas un pilnīgi ieviestas pēdējā datorspēles versijā.

Datorspēles izstrādes process man ir iemācījis vairākas lietas par to, kā viss labāk rakstīti kodu tā lai tā ir viegli pārlasāma, kā arī viegli attīstāma. Datorspēļu izstrādes process mani ir iemācījis, kā vislabāk pārvaldīt projektus šādā apjomā: organizēt savu laiku, organizēt visus nepieciešamos projekta failus un resursus, un veidot lielus projektus bez pārmērīga stresa.

Tā kā es apguvu šīs lietas, kamēr veidojot projektu, tās izstrāde bija viegls bez lielas pārslogošanas un stresa. Projektu izstrādes laikā es strādāju gandrīz katru dienu, kur es censtos izveidot vismaz vienu mazu funkciju priekš programmas vai arī mazlietiņ papildināt dokumentāciju, katru dienu. Tā kā es šo darīju katru dienu es projekta funkcionalitāti izstrādāju daudz ātrāk nekā es pats iedomājos un tā rezultātā, man beigās palika tikai dokumentācijas rakstīšana, programmas testēšana, kā arī programmas vizuālā izskata veidošana un uzlabošana.

# 8. Lietoto terminu un saīsinājumu skaidrojumi

**1. tabula**

**Termini un to skaidrojumi**

|  |  |
| --- | --- |
| **Termins** | **Skaidrojums** |
| Unreal | Saīsinājums Unreal Engine |
| RAM | Datora operatīvā atmiņa |
| Debugging | Atkļūdošana |
| Ragdoll | Spēļu modeļu gravitācijas fizika. |
| PR | Prasība |
| TP | Testpiemērs |
| GV | Galvenā izvēlne |
| IES | Iestatījumi |
| PS | Pirmspēle |
| SP | Spēle |

# 9. Literatūras un informācijas avotu saraksts

Kā uzstatīsit prātnieka patstāvīgā navigāciju:

<https://docs.unity3d.com/Packages/com.unity.ai.navigation@2.0/manual/>

Kā izmantot PlayerPrefs funkciju Unity, lai varētu saglabāt informāciju:

<https://docs.unity3d.com/ScriptReference/PlayerPrefs.html>

Spēlētāja kustības palīdzība:

<https://docs.unity3d.com/520/Documentation/Manual/class-CharacterController.html>

Ragdoll fizikas palīdzība:

<https://docs.unity3d.com/520/Documentation/Manual/class-Rigidbody.html>

Palīdzība par to kā skaņās strādā Unity:

<https://docs.unity3d.com/Manual/Audio.html>

# Pielikums