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

"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, manuprāt, ir nepieciešama, jo esmu pamanījis, ka tirgū pašlaik nav nevienas spēles, kas līdzinātos manai spēles idejai. Esošās spēles parasti ir citos žanros vai arī tās nav pilnībā realizētas. Piemēram, ir spēles, kur galvenais mērķis ir spēlēt vienu un to pašu līmeni, lai iegūtu pēc iespējas augstāku rezultātu, taču lielākā daļa šo spēļu pieder ritmu žanram. Savukārt šaušanas kategorijas spēles bieži vien neļauj spēlētājam kustēties, kā mērķēšanu trenētājos vai arī spēles gaita ir ļoti vienkārša, padarot tās mazāk interesantas atkārtotai spēlēšanai. Tāpēc es cenšos izveidot spēli žanrā, kurā nav tik daudz šāda veida spēļu.

Vēl viens iemesls, kāpēc šī spēle, manuprāt, ir nepieciešama, ir tas, ka es zinu, ka šāda veida spēlēm ir liels pieprasījums. Kā jau iepriekš minēts, pastāv spēles, kurās spēlētājam jāmēģina iegūt pēc iespējas vairāk punktu, taču tās parasti ir citos žanros. Neskatoties uz to, šīs spēles ir ļoti populāras. Šāda veida spēlēm parasti ir ļoti aktīvi spēlētāji, kuri sacenšas savā starpā, lai uzstādītu labāko rezultātu. Tādēļ, ja tiek izveidota spēle, kas ir pievilcīga šādiem spēlētājiem, tā kļūs populāra un vienmēr būs aktīvi spēlētāji, kuri centīsies pārspēt citu spēlētāju labākos rezultātus.

Pēdējais iemesls kāpēc šī spēle, manuprāt, ir nepieciešam ir tāpēc, ka daudzas šaušanas spēlēs pārsvarā ir domātas priekš pieaugušajiem, kura rezultātā daudzas lietas, kas tiek attēlotas tajās spēlēs nav domātas priekš spēlētājiem, kuriem nav 18 gadi. Bet ar savu spēli es gribu izveidot spēli, kuru arī var spēlēt cilvēki, kuri nav pilngadīgi, piemēram, pusaudži. Tādēļ es domāju mana spēle ir nepieciešama, jo tā varētu piesaistīt lielu spēlētāju bāzi, kas sastāv no jaunākiem spēlētājiem, kuri meklē drošu un jautru spēles pieredzi bez nevēlama satura. Šī spēle ir piemērota pusaudžiem, jo tajā nav lietas, kas varētu nebūt piemērotas viņu vecumiem, piemēram, asinis vai šausmīgas ainas.

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 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. Viens ir automātiskais vieglais ierocis, kamēr otrs ir vienšāviena smagais ierocis. Pirmajam spēlētājam ir lielāka mobilitāte, bet mazāk dzīvības kamēr otram spēlētajam ir sliktāka mobilitāte, bet vairāk dzīvības. Spēlētājam ir arī dota iespēja izvēlēties starp diviem spēles laukiem, kur atšķirība ir spēles lauka izskats un izkārtojums.

Spēles spēlētājs arī var mainīt, kā viņš grib spēlēt spēli spēles iestatījumos. Spēles iestatījumos spēlētājs var mainīt spēles ekrāna rezolūciju, kā arī spēles skaņas skaļumu, kā arī mūzikas skaļumu. Spēlētājam arī ir iespēja nomainīt spēles peles ātrumu, kas mainīs cik viegli spēlētājs varēs pakustināt spēles kameru. Pēdējā lieta, ko spēlētājs var darīt iestatījumos ir nomainīt gandrīz visas spēles darbības izpildes kontroles un uzlikt tās uz jebkuru klaviatūras pogu uz, ko grib.

Spēles laikā spēlētājam ir dotas viens veids, kā viņš var iegūt punktus. Spēlētājs var dabūt punktus nošaujot uz spēles lauka esošos pretiniekus, kuri sākumā staigās uz vietas, bet spēlei progresējoties viņi vienmēr sekos spēlētājam, un paliks arī ātrāki. Pretinieki arī paradīsies nejaušās vietās uz spēles lauka, katru reizi, ka pretinieks tiks nošauts. Spēlē ir trīs veidu pretinieki, kur katram ir sava speciāla priekšrocībā. Pirmais pretinieku veids ir parastie pretinieki, kuri uzbrūk pieejot, klāt pie spēlētāja un viņam iesitot. Otrais pretinieku veids ir burvju pretinieks, kuri met uguns bumbas spēlētājam, kad ir pietiekami tuvi. Pēdējais pretinieku veids ir stiprie pretinieki, kuri strādā līdzīgi parastiem pretiniekiem, tikai viņiem ir daudz vairākas dzīvības un arī nogalina spēlētāju vienā sitienā. Spēlētājs var arī palielināt iegūto punktu daudzumu savācot reizinātāja kapsulas, kuru savācot reizinās spēlētāja iegūtos punktus ar divi uz nākošām trīsdesmit sekundēm. Vel viens veids kā spēlētājs var palielināt iegūto punktu daudzumu ir dabūt un uzturēt spēles "kombo". Spēlētājs dabū "kombo" nošaujot divus pretiniekus desmit sekunžu intervālā un ja spēlētājs nenošauj vel vienu pretinieku desmit sekunžu intervālā tas pazūd. Ja spēlētājs var nošaut desmit pretiniekus "kombo" intervālā, tad punkti lēnām sāks reizināties ar 1,2 un katrus desmit pretiniekus tas palielinās ar 0,2 līdz punkti tiek reizināti ar divi. Ja spēlētājs zaudēs "kombo" tad punktu reizinātājs pazudīs un spēlētājam, tas būs jāsakrāj no jauna.

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 pagarināt spēles laiku palielināt, lai spēle nebeigtos. Pirmais veids ir savācot laika kapsulas, kuras atrodas apkārt spēles laukam un savācot vienu palielina spēles laiku par sešdesmit sekundēm. Otrais veids ir nošaujot pretinieku, kur par katru nošauto pretinieku, laiks palielinās par piecām sekundēm.

Spēle var beigties dēļ trīs iemesliem, spēles laiks beidzās, kāds pretinieks tevi nogalina vai arī spēlētājs nošāva pilnībā visus pretiniekus, kur var būt maksimāli simts katru spēli. Ja spēle beidzās dēļ laika vai spēlētāja miršanas dēļ, tad parādīsies ekrāns, kas pateikts kā spēle beidzās un ļaus spēlētājam aiziet uz spēles galveno izvēlni. Ja spēle beidzas, kad spēlētājs nošauj visus pretiniekus, tad tiek parādīti spēlētāja iegūtie punkti, atlikušo laiku un kopējo iegūto punktu skaitu, kuru iegūst, saskaitot atlikušo laiko ar punktiem. Tāpat tiek parādīts vērtējums par spēlētāja sniegumu, kur vērtējumi var būt: F, D, C, B, A vai S.

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:

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:

Tiek pārbaudīts vai spēlētājs nospieda pogu.

Izvaddati:

Tiek spēlēta pogas skaņa un 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:

Tiek pārbaudīts vai spēlētājs nospieda pogu.

Izvaddati:

Tiek spēlēta pogas skaņa un 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:

Tiek pārbaudīts vai spēlētājs nospieda pogu.

Izvaddati:

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

P.5. Datorspēles “ScoreStorm” fona mūzikas laika saglabāšana

Mērķis:

Šīs funkcijas mērķis ir saglabāt fona mūzikas vērtību, kura tiek spēlēta galvenās izvēlnes, iestatījumu un pirmspēles ainā, lai mūzika nesāktos no jauna, kad pāriet starp ainām.

Ievaddati:

Spēlētājs pāriet starp ainām.

Apstrāde:

Tiek pārbaudīts, cik ilgi mūzika ir spēlēta.

Izvaddati:

Mūzikas laiks tiek saglabāts un kad spēlētājs pāriet starp ainām mūzika sāks atskaņoties no tās vietas.

2.2.2. Iestatījuma loga prasības

P.6. 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.
4) Funkcija pārbauda vai lietotājs atrodas iestatījumu ainā.

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.
3) Ja spēlētājs atrodas iestatījuma logā tad spēles skaņas skaļuma vērtība tiks uzreiz atjaunota nevis pēc ainu maiņas.

P.7. 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.
4) Funkcija pārbauda vai spēlētājs atrodas iestatījumu ainā.

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.
3) Ja spēlētājs atrodas iestatījuma logā tad mūzikas skaļuma vērtība tiks uzreiz atjaunota nevis pēc ainu maiņas.

P.8. 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 rezolūciju, ko spēlētājs bija izvēlējies.

P.9. 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.10. 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.11. 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 abiem spēlētājiem.

P.12. 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.

P.13. Palīdzības loga atvēršana

Mērķis:

Mērķis ar šo funkciju ir dot iespēju spēlētājam atvērt palīdzības logu, kur ir dota informācija par to kā spēlēt spēli.

Ievaddati:

Peles klikšķis uz pogu ar apzīmējumu “i”.

Apstrāde:

Spēle pārbauda vai spēlētājs ir nospiedis pogu.

Izvaddati:

Tiek atskaņota pogas nospiešanas skaņa un paradās palīdzības logs ar pirmo lapu.

P.14. Palīdzības loga lapas pāreja

Mērķis:

Mērķis ar šo funkciju ir dot iespēju spēlētājam pāriet starp palīdzības loga trīs lapām.

Ievaddati:

1) Peles klikšķis uz pogu ar apzīmējumu "→".
2) Peles klikšķis uz pogu ar apzīmējumu "←".

Apstrāde:

1) Spēle pārbauda vai spēlētājs ir nospiedis pogu.
2) Spēle pārbauda uz kuras lapa puses spēlētājs atrodas.

Izvaddati:

1) Tiek atskaņota pogas nospiešanas skaņa.
2) Ja spēlētājs nospiedīs pogu ar apzīmējumu "→", tad parādīsies nākamā lapas puse.
3) Ja spēlētājs nospiedīs pogu ar apzīmējumu "←", tad parādīsies iepriekšējā lapas puse.

P.15. Palīdzības loga aizvēršana

Mērķis:

Mērķis ar šo funkciju ir dot iespēju spēlētājam aizvērt palīdzības logu.

Ievaddati:

Peles klikšķis uz pogu ar apzīmējumu "X".

Apstrāde:

Spēle pārbauda vai spēlētājs ir nospiedis pogu.

Izvaddati:

Tiek atskaņota pogas nospiešanas skaņa un palīdzības logs ir aizvērts.

2.2.4. Spēles loga prasības

P.16. 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 kādu spēlētāju spēlētājs ir izvēlējies.
2) Spēle pārbauda kādu līmeni spēlētājs ir izvēlējies.

Izvaddati:

1) Spēlētājs tiek pārvietots uz spēles logu.
2) Parādās atbilstošais spēles tēls.
3) Parādās atbilstošais spēles līmenis.
4) Sākas laika atskaita.
5) Pazūd peles kursors.
6) Tiek atskaņota atbilstošā spēlētāju mūzika.
7) Parādās pirmie 15 pretinieki.

P.17. 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.18. 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.19. 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.20. 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.21. Datorspēles "ScoreStorm" spēles skatīšanās

Mērķis:

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

Ievaddati:

Spēlētājs pakustina datora peli 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.22. 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) Parādās sarkanais mērķēšanas punkts.
4) Ierocis tiks mērķēts tik ilgi kamēr spēlētājs atlaidīs vaļā peles labo taustiņu.

P.23. 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 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.24. 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.25. 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.26. 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:

1) Spēles tēls izdara bloķēšanas animāciju.
2) Ja spēlētājam animācijas laikā uzbrūk pretinieks sitot atņemto dzīvības daudzums samazinās par pusi.
3) Ja spēlētājam animācijas laikā uzbrūk pretinieks metot uguns bumbas, tad nekādas dzīvības netiks atņemtas.

P.27. Datorspēles "ScoreStorm" objektu pacelšana

Mērķis:

Mērķis ar šo funkciju ir dot iespēju spēlētājam pacelt spēles objektus.

Ievaddati:

Spēlētājs nospiež saskares pogu, kura pēc noklusējuma ir "E".

Apstrāde:

1) Spēle pārbauda vai spēlētājs atrodas tuvu paceļamam objektam.
2) Spēle pārbauda vai objekts ir lodes kaste, pirmā aptieciņa, laiku reizinātājs vai laiku pagarinātājs.

3) Spēle pārbauda vai lietotājs ir nospiedis saskares pogu.
4) Spēle pārbauda vai spēlētājs nav sasniedzis maksimālo dzīvības vērtību.

Izvaddati:

1) Tiek spēlēto atbilstošā pacelšanas skaņa.
2) Paceltais objekts pazūd.
3) Ja spēlētājs pacēla zaļo lodes kasti, tad dos 60 lodes spēlētāja maz kalibra ierocim.
4) Ja spēlētājs pacēla pelēko lodes kasti, tad dos 7 lodes spēlētāja liel kalibra ierocim.
5) Ja spēlētājs pacēla pirmo aptieciņu, tad dos 50 dzīvības klāt spēlētājam, ja viņš nav sasniedzis maksimālo dzīvības vērtību.
6) Ja spēlētājs pacēla laika kapsulu, tad spēles laiks tiek pagarināts par 60 sekundēm.
7) Ja spēlētājs pacēla reizinātāju kapsulu, tad spēles uz 30 sekundēm reizinās visus nākošos iegūtos punktus ar 2 un uz ekrāna parādīsies 2x pie punktiem.
8) Pēc skaņas beigšanas objekts tiek iznīcināts.

P.28. 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.
3) Spēle pārbauda vai pretiniekam netrāpīja par galvu.
4) Spēle pārbauda kāda veida pretinieks tas ir.

Izvaddati:

1) Ja pretiniekam trāpa spēlētājs un viņa dzīvības vērtība ir lielāka par 0, tad tiek atņemta atbilstošā dzīvības daudzuma no pretinieka, spēlēts iešaušanas skaņas efekts, kā arī pretinieks paliek agresīvs un visu laiku sekos spēlētājam.
2) Ja pretiniekam trāpa spēlētājs un viņa dzīvības vērtība ir mazāka par 0, tad tiek atskaņota miršanas skaņas efekts un aktivēts pretinieka objekta ragdoll, kā arī tiek spēlētājam pieskaitītas 5 sekundes pie laika, un tiek pieskaitīts +1 spēlētāja "kombo" rādītājam, un pēc piecām sekundēm pretinieka objekts tiek iznīcināts.
3) Ja spēlētājs trāpa galvā, tad ir 50% iespēja, ka pretinieks uz brīdi nekustēsies un izdarīs īsu animāciju.

P.29. Datorspēles "ScoreStorm" pretinieku staigāšana uz vietas

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju, ka spēles laikā pretinieks staigās uz vietas.

Ievaddati:

Spēlētājs sāk spēli.

Apstrāde:

1) Spēle pārbauda vai spēlētājs nav iešāvis pretiniekam.
2) Spēle pārbauda vai spēlētājs nav pretinieku sekošanas attālumā.
3) Spēle pārbauda vai spēlētājs nav nošāvis 10 pretinieks.
4) Spēle pārbauda vai nākošais iešanas punkts atrodas uz navigācijas zonas.

Izvaddati:

Pretinieks staigā uz vietas noteiktajā rādiusā.

P.30. Datorspēles "ScoreStorm" pretinieku sekošana

Mērķis:

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

Ievaddati:

Spēlētājs atrodas spēlē.

Apstrāde:

1) Spēle pārbauda vai spēlētājs atrodas pretinieku redzes zonā.
2) Spēle pārbauda vai spēlētājs ir iešāvis pretiniekam.
3) Spēle pārbauda vai spēlētājs ir nošāvis kopā 10 pretiniekus.

Izvaddati:

Pretinieks sekos spēlētājam izmantojot tuvāko ceļu

P.31. Datorspēles "ScoreStorm" pretinieku uzbrukšana

Mērķis:

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

Ievaddati:

Spēlētājs atrodas spēlē.

Apstrāde:

1) Spēle pārbauda vai lietotājs atrodas pretinieka uzbrukšanas zonā.
2) Spēle pārbauda vai pretinieks var izmantot burvestības.

Izvaddati:

1) Ja pretinieks nevar izmantot burvestības, tad pretinieks izdarīs fizisku uzbruku, kur tiks spēlēta animācija, uzbrukšanas skaņas efekts, kā tiks atņemtas atbilstoši spēlētāju dzīvības, no tuvas distances.
2) Ja pretinieks var izmantot burvestības, tad pretinieks izdarīs mešanas uzbrukumu, kur tiks spēlēta animācija, uzbrukšanas skaņas efekts, kā izmesta uguns bumba, kura ja savienojās ar spēlētāju atņems viņam dzīvības.

P.32. 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 sāk spēli.

Apstrāde:

1) Spēle pārbauda vai uz spēles lauka kopā atrodas 15 pretinieki.
2) Spēle pārbauda cik pretiniekus spēlētājs ir nošāvis.
3) Spēle pārbauda vai nav sasniegts maksimālais pretinieku skaits.

Izvaddati:

1) Ja nav sasniegts limits, tad uz lauka parādīsies pretinieks pretinieku staigāšanas zonās.
2) Ja ir sasniegts limits, tad uz lauka neparadīsies pretinieks.
3) Ja uz lauka ir kopā bijuši 25, 50, 75, 98, 99 pretinieki, tad nākošais pretinieks, kas parādies būs stiprais pretinieks.
4) Ja nē tad ir 50% iespēja, ka nākošais pretinieks būs parastais pretinieks vai burvju pretinieks.

P.33. Datorspēles "ScoreStorm" dzīvības zaudēšana

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 atbilstošas dzīvības daudzums. Ja spēlētājs bloķēja dzīvības zaudēšanas laikā, tad atņemtās dzīvības daudzums samazinās par 50%, no fiziskiem uzbrukumiem un 100% no mešanas uzbrukumiem.
2) Ja spēlētājam trāpa pretinieks un viņa dzīvības vērtība ir mazāka par 0, tad tiek atskaņota miršanas animācija, kur spēlētāja kontrole ir atņemta, un pēc animācijas parādās spēles zaudēšanas logs.

P.34. Datorspēles "ScoreStorm" punktu iegūšana

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju spēlētājam iegūt punktus spēles laikā.

Ievaddati:

Spēlētājs nošauj pretinieku.

Apstrāde:

1) Spēle pārbauda vai ir aktīvs punktu reizinātājs, no savācamā objekta.
2) Spēle pārbauda vai ir aktīvs punktu reizinātājs, no "kombo" skaitītāja.

Izvaddati:

1) Ja spēlētājam nav nekāds reizinātājs aktīvs, tad iegūs parasto punktu daudzumu no pretiniekiem.
2) Ja spēlētājam ir aktīvs punktu reizinātājs no savācamā objekta, tad iegūtie punkti tiks reizināti ar 2.
3) Ja spēlētājam ir aktīvs "kombo" skaitītājs un ja tā vērtība ir 10, tad spēlētāja iegūtie punkti tiks reizināti ar 1,2.
4) Ja spēlētājam ir aktīvs "kombo" skaitītājs un ja tā vērtība ir 20, tad spēlētāja iegūtie punkti tiks reizināti ar 1,4.
5) Ja spēlētājam ir aktīvs "kombo" skaitītājs un ja tā vērtība ir 30, tad spēlētāja iegūtie punkti tiks reizināti ar 1,6.
6) Ja spēlētājam ir aktīvs "kombo" skaitītājs un ja tā vērtība ir 40, tad spēlētāja iegūtie punkti tiks reizināti ar 1,8.
7) Ja spēlētājam ir aktīvs "kombo" skaitītājs un ja tā vērtība ir 50, tad spēlētāja iegūtie punkti tiks reizināti ar 2.
8) Ja ir aktīvi abi divi reizinātāji, tad iegūtie punkti tiks reizināti ar abu reizinātāju skaitļu summu.

P.35. 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:

1) Spēle pārbauda vai lietotājs ir nošāvis divus pretiniekus vismaz 10 sekunžu laika intervālā.
2) Spēle pārbauda cik liela ir "kombo" vērtība.

Izvaddati:

1) Spēlētājam uz ekrāna paradās "kombo" skaitītājs.
2) Ja spēlētāja "kombo" vērtība ir lielāka par 10, tad uz ekrāna parādīsies reizinātāja vērtība pie skaitītāja un tiks aktivizēts reizinātājs.
3) "Kombo" atlikušā laika līnija tiek iestatīta uz 10.

P.36. Datorspēles "ScoreStorm" "kombo" skaitītāja pazušana

Mērķis:

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

Ievaddati:

Spēlētājs nenošauj pretiniekus.

Apstrāde:

Spēle pārbauda vai ir pagājušas 10 sekundes bez pretinieka nomiršanas.

Izvaddati:

Spēlētājam pazūd "kombo" skaitītājs, kā arī reizinātājs tiek izslēgts un iestatīs uz 1.

P.37. Datorspēles "ScoreStorm" zaudēšanas ekrāns

Mērķis:

Mērķis ar šo funkciju ir nodrošināt ka parādās ekrāns, kad spēlētājs zaudē spēli.

Ievaddati:

Spēles laiks beidzas vai spēlētājs ir nomirs.

Apstrāde:

1) Spēle pārbauda vai ir beidzies spēles laiks.
2) Spēle pārbauda vai ir nomiris spēlētājs.

Izvaddati:

1) Parādās spēles zaudēšanas ekrāns.

2) Tiek atskaņota zaudēšanas skaņas efekts.
3) Parādās peles kursors
4) Ja spēle beidzās dēļ laika, tad uz ekrāna parādās teksts “Laiks ir beidzies”.
5) Ja spēle beidzās dēļ spēlētāja nāves, tad uz ekrāna parādās teksts “Jūs esat miruši!”.

P.38. Datorspēles “ScoreStorm” uzvaras ekrāns

Mērķis:

Mērķis ar šo funkciju ir nodrošināt ka parādās ekrāns, kad spēlētājs uzvara spēli.

Ievaddati:

Spēlētājs nošauj visus pretiniekus.

Apstrāde:

1) Spēle pārbauda vai spēlētājas ir nošāvis visus 100 pretiniekus.
2) Pārbauda esošo spēlētāju.
3) Pārbauda iepriekšējo rezultātu uz esošā līmeņa ar esošo spēlētāju.

Izvaddati:

1) Parādās spēles uzvaras ekrāns.
2) Tiek atskaņota uzvaras skaņas efekts.
3) Parādās peles kursors
4) Parādās attiecošā spēlētāju bilde.

P.39. Datorspēles “ScoreStorm” iziet uz sākumekrāna poga

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju atgriezties uz galveno izvēlni pēc spēles.

Ievaddati:

Spēlētājs nospiež uz pogu “Iziet uz sākumekrānu”

Apstrāde:

Spēle pārbauda vai lietotājs ir nospiedis pogu.

Izvaddati:

Spēlētājs tiek aizvests uz galveno izvēlni un tiek pogas atskaņota skaņa.

P.40. 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:

1) Tiek parādīti spēlē iegūtie punkti.
2) Ja bija iegūts jauns rekords, tad pie punktiem parādās zīmīte, ka ir jauns rekords.
3) Tiek parādīts vērtējums.
4) Ja rezultāts bija mazāks par 2000, tad spēlētājs tiek vērtēts kā F.
5) Ja rezultāts bija mazāks par 5000, tad spēlētājs tiek vērtēts kā D.
6) Ja rezultāts bija mazāks par 10000, tad spēlētājs tiek vērtēts kā C.
7) Ja rezultāts bija mazāks par 15000, tad spēlētājs tiek vērtēts kā B.
8) Ja rezultāts bija mazāks par 20000, tad spēlētājs tiek vērtēts kā A.
9) Ja rezultāts bija vairāk par 20000, tad spēlētājs tiek vērtēts kā S.

P.41. Datorspēles "ScoreStorm" ieroča maiņa

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju spēlētājam mainīt ieroci spēles laikā.

Ievaddati:

Spēlētājs nospiež maiņas pogu, kura pēc noklusējuma ir "Q".

Apstrāde:

1) Spēle pārbauda vai lietotājs ir nospiedis maiņas pogu.
2) Spēle pārbauda vai spēlētājs nepārlādē ieroci.
3) Spēle pārbauda vai spēlētājs bloķē.

Izvaddati:

Spēlētājs pārmaina ieroci un spēlē atbilstošo skaņas efektu.

P.42. Datorspēles "ScoreStorm" pretinieku patriņāšanās

Mērķis:

Mērķis ar šo funkciju ir nodrošināt iespēju ka spēles laikā pretinieki paliks ātrāki

Ievaddati:

Spēlētājs nošauj pretiniekus

Apstrāde:

1) Spēle pārbauda spēlētājs ir nošāvis 50 pretiniekus

2) Spēle pārbauda vai spēlētājs ir nošāvis 75 pretiniekus.

Izvaddati

1) Ja spēlētājs ir nošāvis 50 pretiniekus, tad viņu ātrums tiks uzlikts uz 6.5f.
2) Ja spēlētājs ir nošāvis 75 pretiniekus, tad viņu ātrums tiks uzlikts uz 7.5f.

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:

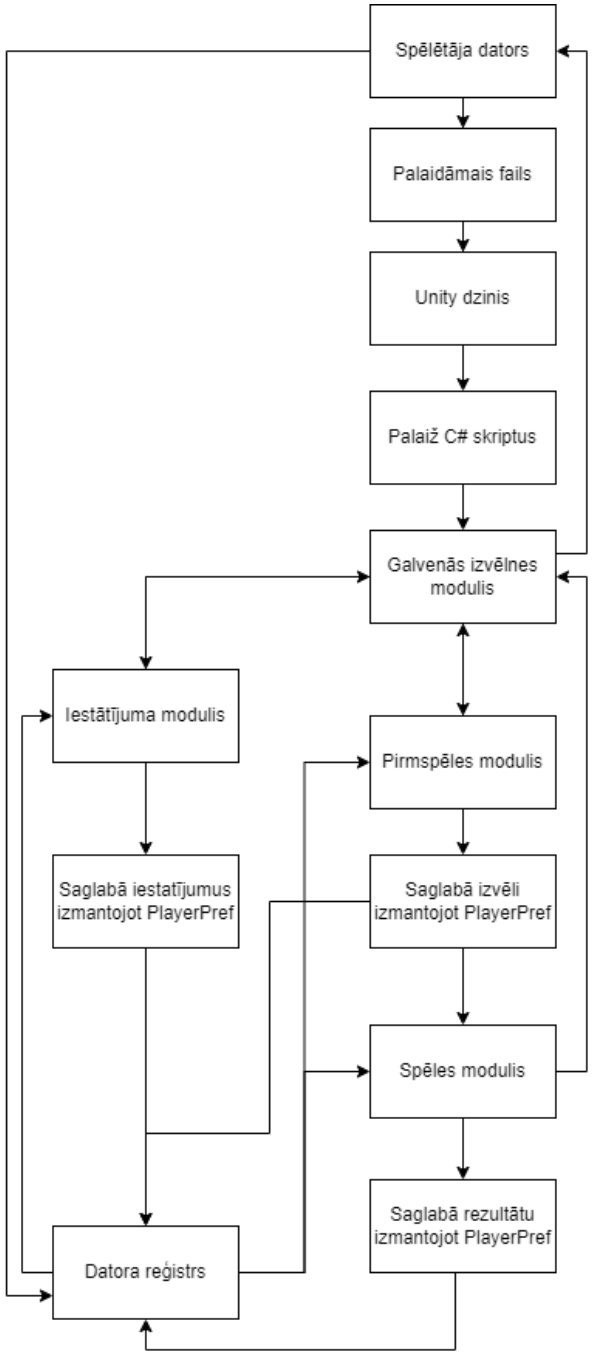
Figma varēja tikt izmantota Draw.io vietā, bet beigās netika izmantota tāpēc, ka Figma nav tik daudzi diagrammu zīmēšanas rīki nekā Draw.io. Tas nozīmētu, ka būtu bijis daudz grūtāk uzstatīsit diagrammas izmantojot Figma nevis Draw.io.

4. Sistēmas struktūras modelēšana un projektēšana

Šajā nodaļa atrodas visas diagrammas saistītas ar projektu.

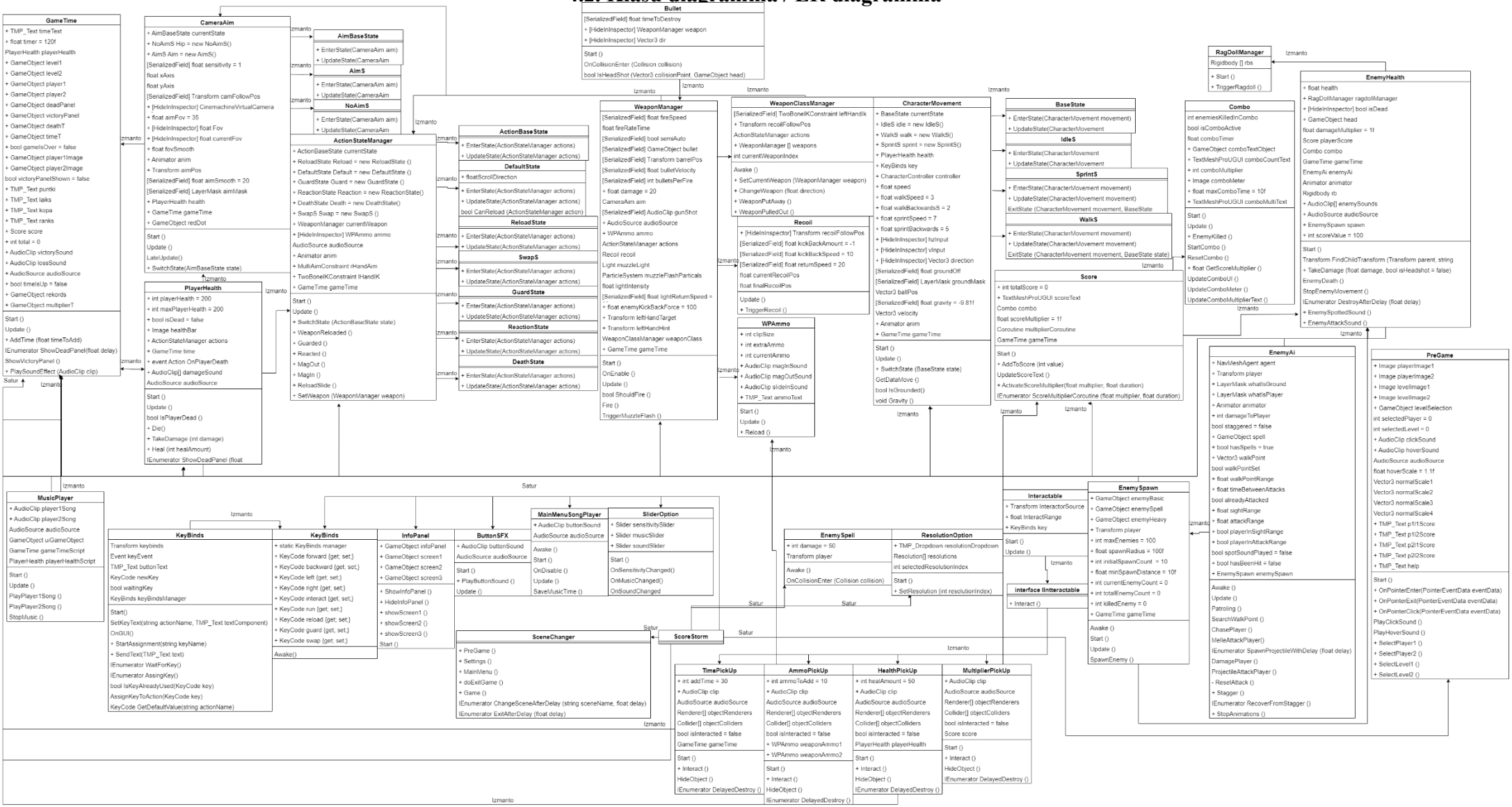
4.1. Sistēmas struktūras modelis

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

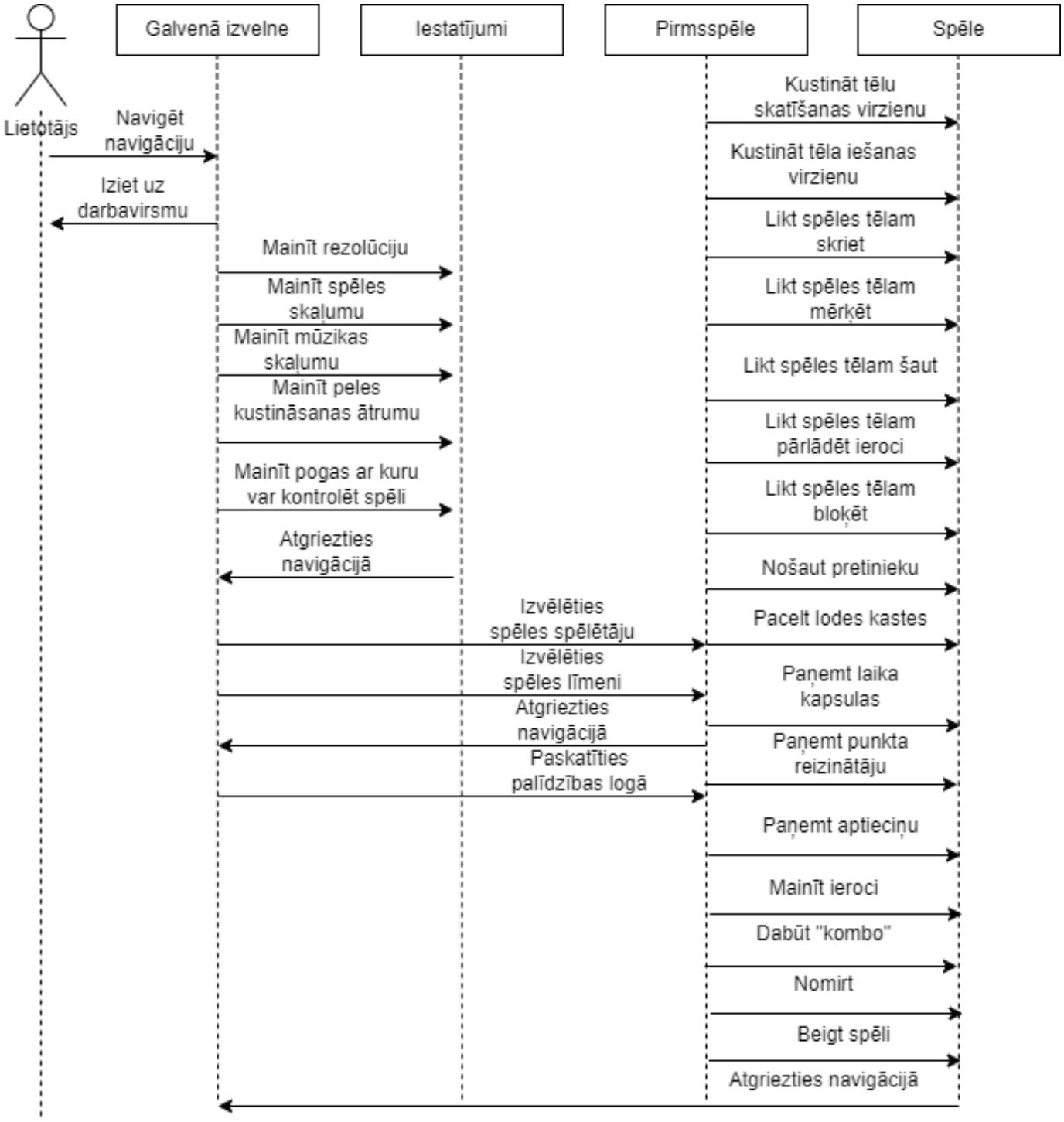
4.2. Klašu diagramma / ER diagramma



2.attēls klašu diagramma

4.3. Funkcionālais un dinamiskais sistēmas modelis

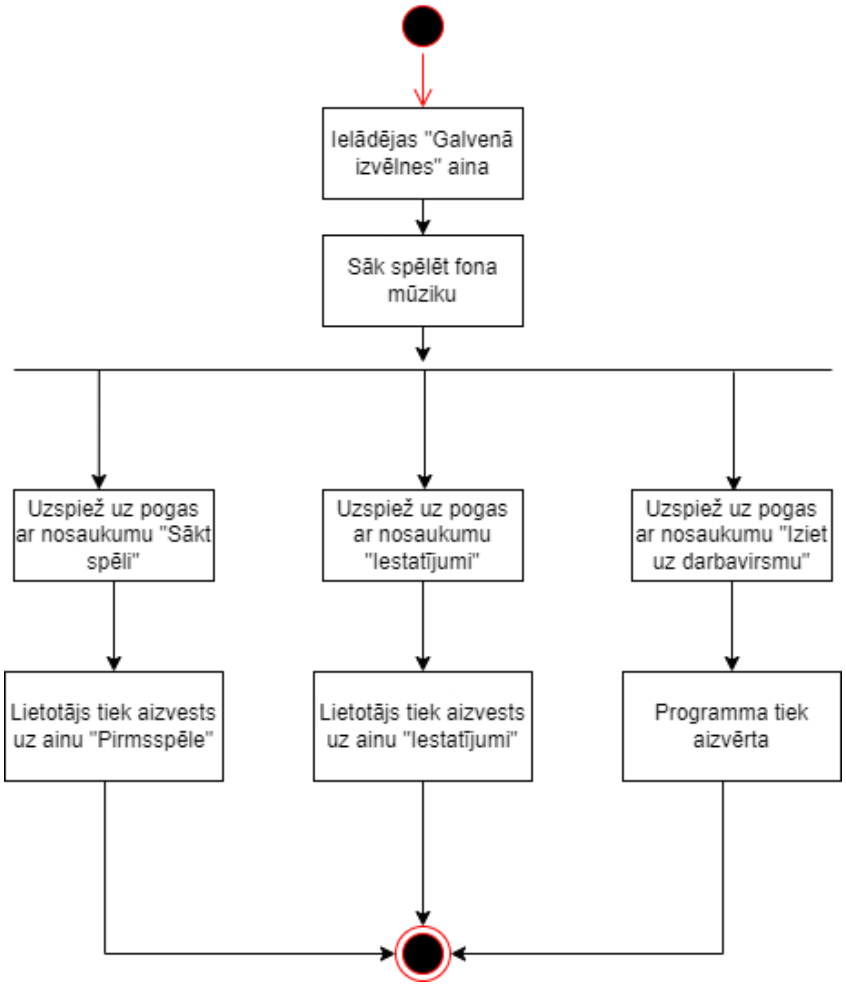
Šajā diagramma var apskatīt kāda izskatās datorspēles darbība un kādas ir lietotāja iespējas sekvenču diagrammā.



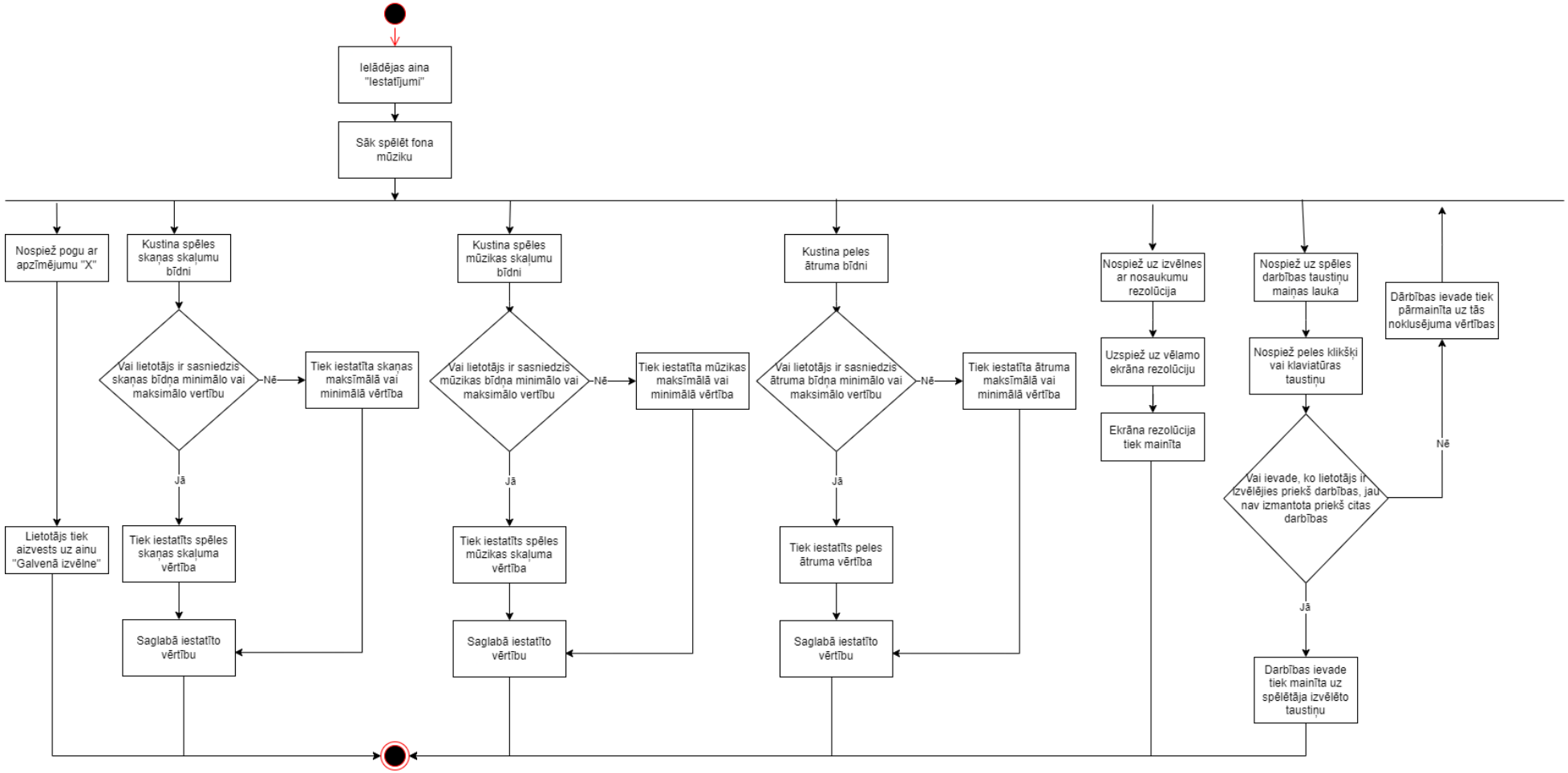
3.attēls Sekvenču diagramma

4.4. Aktivitāšu diagramma

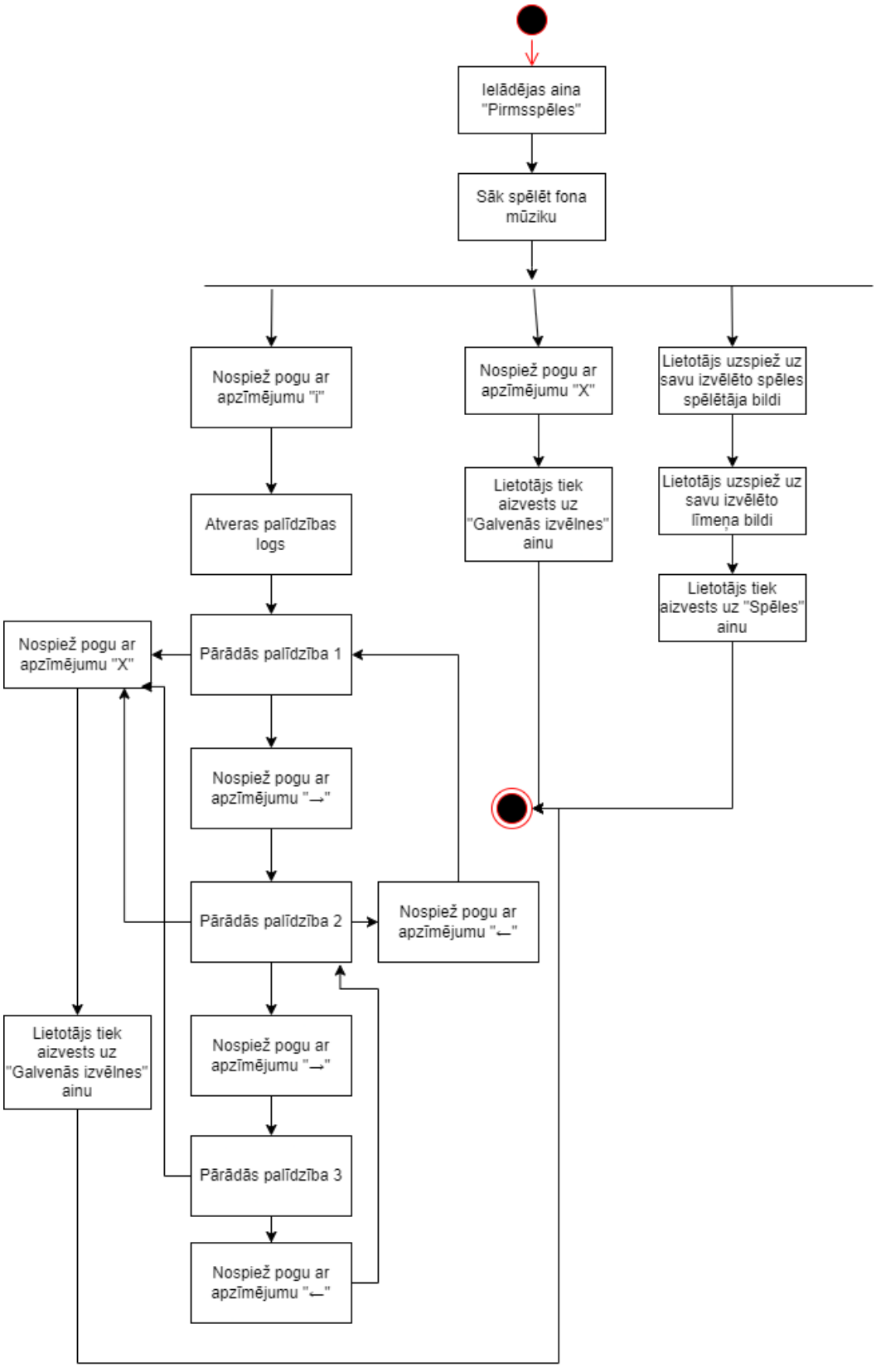
Šajā nodaļa var apskatīt aktivitātes diagrammas priekš katriem programmas moduļiem.



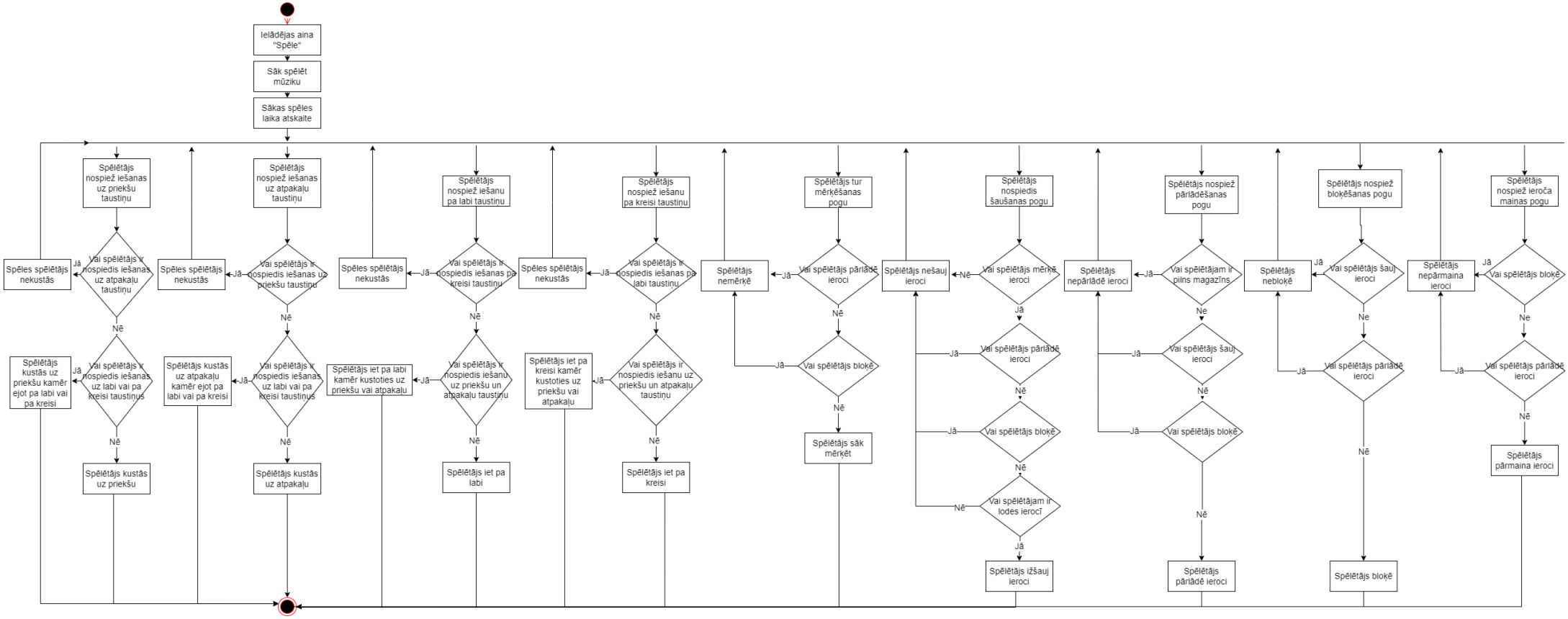
4.attēls Aktivitātes diagramma galvenai izvēlne



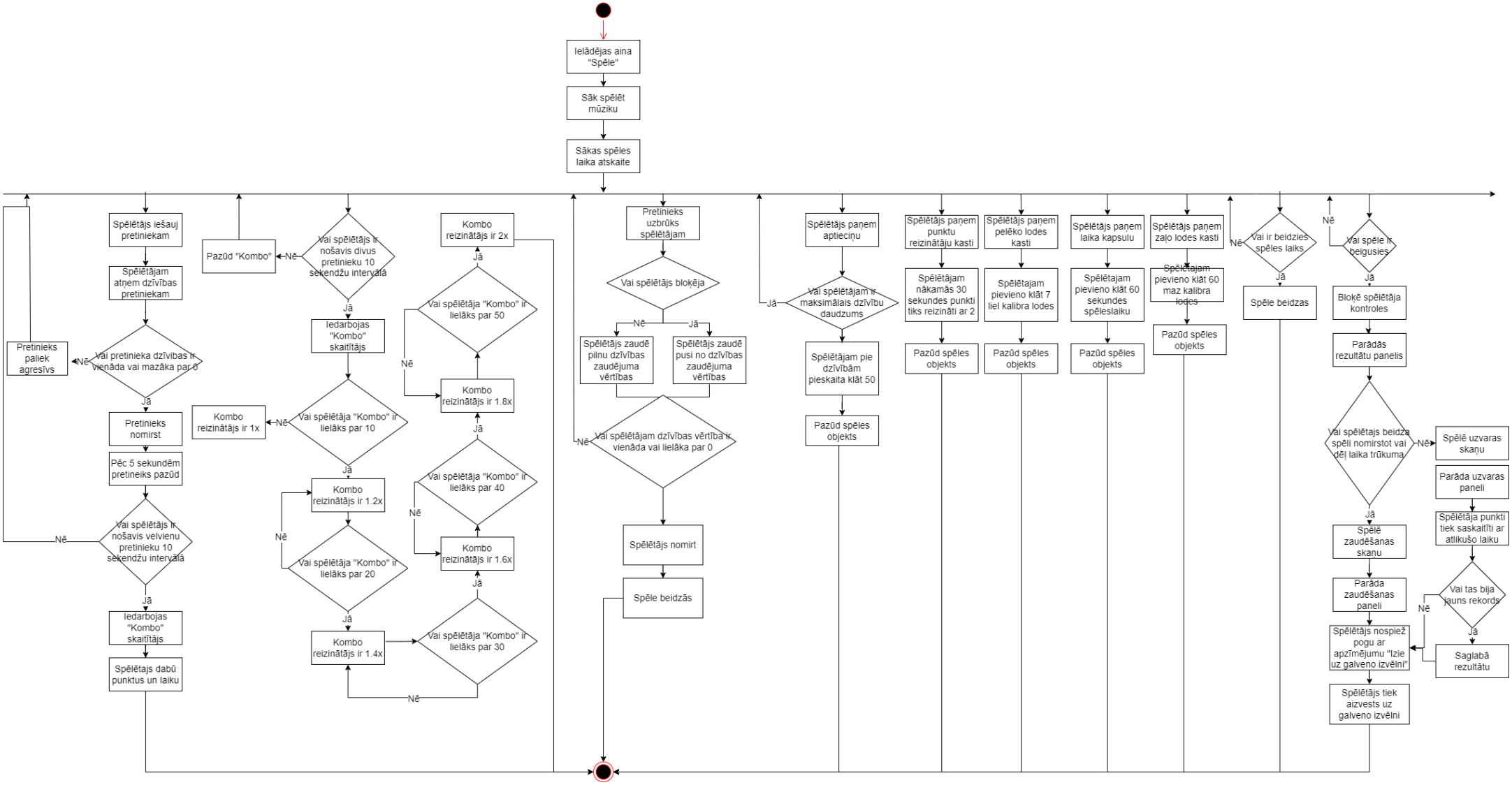
5.attēls Aktivitātes diagramma iestatījumiem



6.attēls aktivitātes diagramma pirmspēles logam



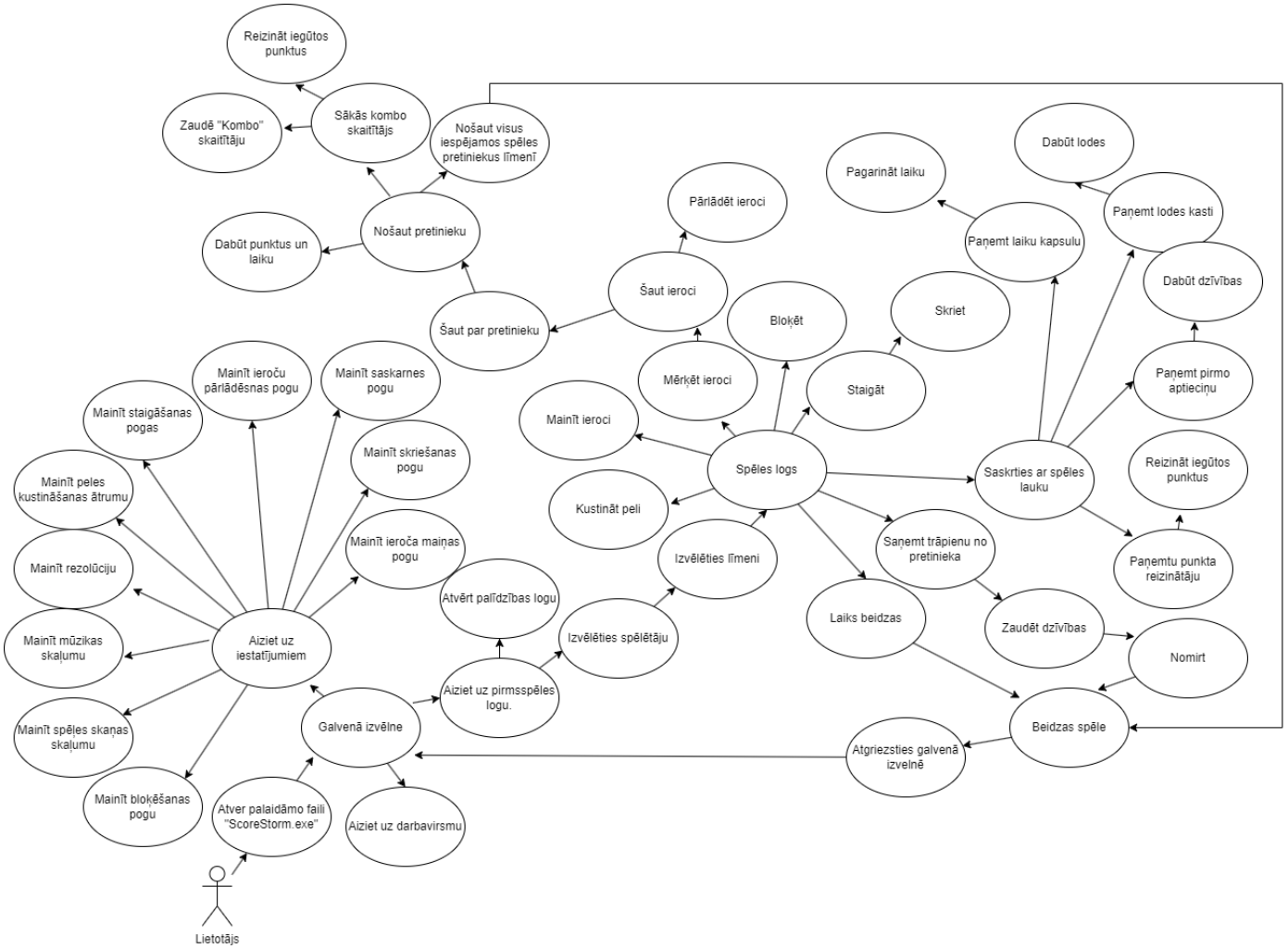
7.attēls aktivitātes diagramma spēles logam pirmā daļa



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. Pirmspēles logā arī atrodas palīdzības logs, kurš lietotajam pastāsta, kā spēlēt spēli.

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 fiziski vai metot objektu, 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 vispār netiek rādīts un glabāts. 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

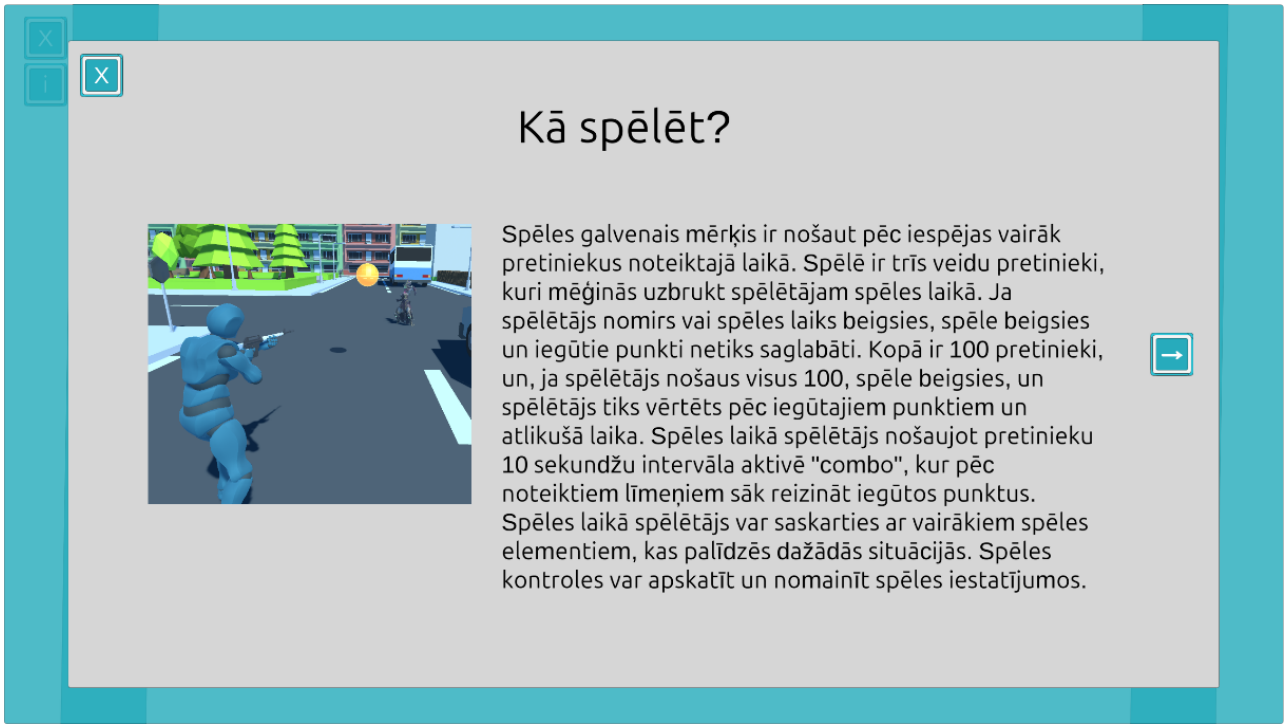
Pirmspēles logs ir programmas sadaļa, kur lietotājs var sākt spēli. Spēlētājs var atgriezties galvenajā izvēlnē izmantojot pogu “x” lapas kreisajā augšējā stūrī. Pirmspēles loga apakšas vidū atrodas bildes ar spēles tēliem, kur kad spēlētājs ir nolicis kursoru virs attēla parādīsies apraksts par spēlētāju. Lai izvēlētos līmeni lietotājam ir jānospiež vienām no divām līmeņu bildēm.

12.attēls spēlētāju izvēlne

Kad lietotājs ir izvelējies spēlētāju parādīsies līmeņu izvēlne, kur zem katra līmeņa bildes parādīsies rekord ar iegūto rezultātu. Kad lietotājs izvēlas līmeni spēle automātiski sāksies.

13.attēls līmeņa izvēlne

Pirmspēles logā arī atrodas poga ar apzīmējumu “i”, kas atvērs vaļā palīdzības loga, kur spēlētājam ir dots izskaidrojums, kā spēlēt spēli. Spēlētājs arī var pāriet uz citu padomju lapu izmantojot pogas ar bultiņu apzīmējumiem abos stūros. Ja spēlētājs grib aizvērt lapu ciet viņš var izmantot pogu ar apzīmējumu “x” kreisajā augšējā stūrī.



14.attēls pirmais palīdzības logs

15.attēls otrais palīdzības logs

16.attēls trešais palīdzības logs

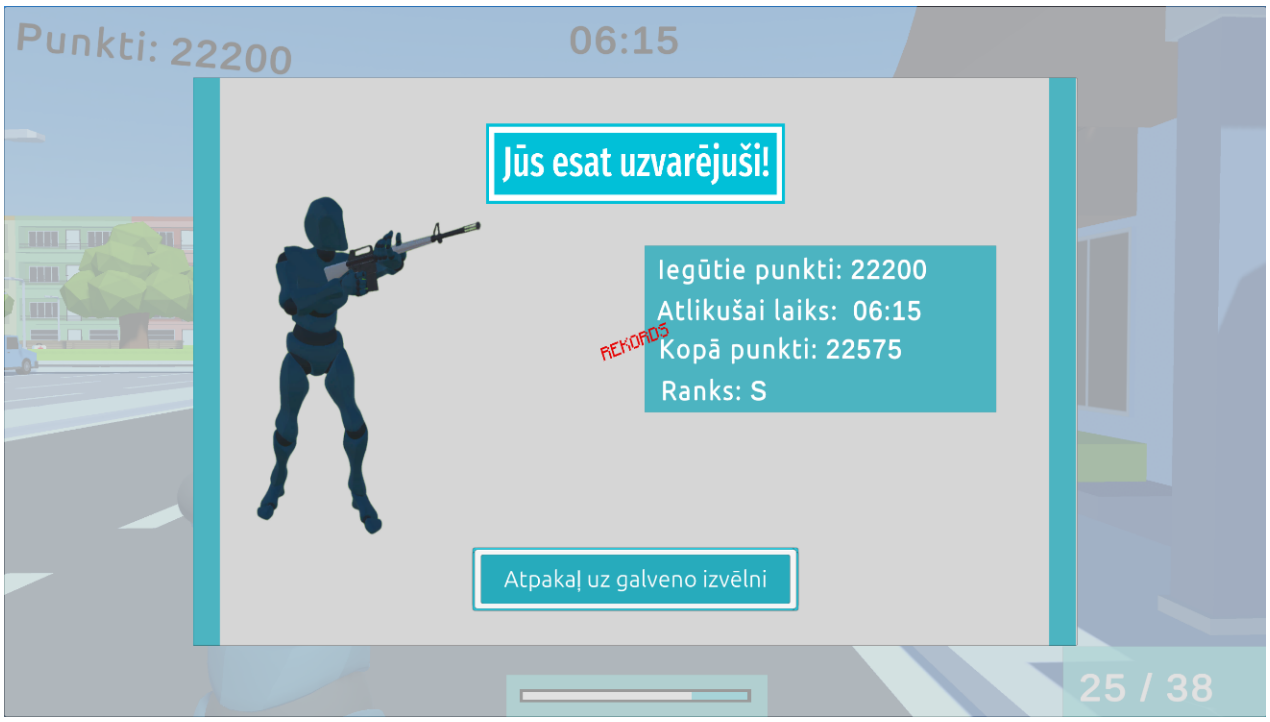
5.4. Spēle

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 sarkans punktiņš, kurš tikai ir aktīvs, tad kad lietotājs mērķē.

17.attēls spēles logs

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 "Left 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 mazu bonusu, kā laiku, reizināt punktus, dzīvības vai lodes. Spēlētājs arī var mainīt savu ieroci izmantojot maiņas pogu, kura pēc noklusējuma ir "Q". 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. Ja spēle beidzas nošaujot visus pretiniekus, tad 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 "Atpakaļ uz galveno izvēlni", kur pēc nospiešanas lietotājs tiks aizvests atpakaļ uz galveno izvēlni.



18.attēls uzvaras logs

Ja lietotājs pabeidz spēli nomirstot vai laikam beidzoties, tad parādīsies zaudēšanas ekrāns ar atbilstošo zaudēšanas tekstu.

19.attēls laika beigšanās logs

20.attēls miršanas logs

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 4. tabulā. Testēšanas žurnālā tiek arī izmantoti prasības saīsinājumi, kurus var apskatīt 1. tabulā.

6.2. Testpiemēru kopa

1. tabula

Testpiemēru prasības

Prasības ID	Prasība
Galvenā izvēlne	
PR.GV.01.	Iziet uz darbavirsmas
PR.GV.02.	Pāriet uz Iestatījuma logu
PR.GV.03.	Pāriet uz Pirmsspēles logu
PR.GV.04.	Atgriezties galvenā izvēlnē
Iestatījumi	
PR.IES.01.	Spēles ekrāna rezolūcijas maiņa
PR.IES.02.	Spēles darbības pogas maiņa
PR.IES.03.	Spēles peles kusora ātruma maiņa
PR.IES.04.	Spēles skaņas skaļuma maiņa
PR.IES.05.	Spēles mūzikas skaļuma maiņa
Pirmsspēle	
PR.PS.01.	Spēles tēla izvēlne
PR.PS.02.	Spēles līmeņa izvēlne
PR.PS.03.	Palīdzības logs
Spēle	
PR.SP.01.	Spēles tēla staigāšana
PR.SP.02.	Spēles tēla skatīšanās
PR.SP.03.	Spēles tēla skriešana
PR.SP.04.	Spēles tēla ieroča mērķēšana
PR.SP.05.	Spēles tēla ieroča šaušana
PR.SP.06.	Spēles tēla ieroča pārlādēšana
PR.SP.07.	Spēles tēla ieroča maiņa
PR.SP.08.	Spēles tēla bloķēšana
PR.SP.09.	Spēles tēla dzīvības zaudēšana
PR.SP.10.	Spēles pacēļamie objekti
PR.SP.11.	Nošaut pretinieku
PR.SP.12.	Pretinieku uzvedība
PR.SP.13.	Pretinieka parādīšanās
PR.SP.14.	"Kombo skaitītājs"
PR.SP.15.	Spēles beigšana logs

2. tabula

Testpiemēri

Testpiemēra ID	Testpiemēra nosaukums	Testpiemēra izpildes nosacījums	Testpiemēra apraksts	Testpiemēra izpildes soļi	Testpiemēra ievades dati	Testpiemēra sagaidāmais rezultāts	Prasības ID
Black Box							
TP.GV.01.	Iziešana uz darbvirsmas	Jāatrodas galvenajā izvēlnē	Datorspēles aizvēršana ciet	1) Peles kreisā taustiņa klikšķis uz pogas ar aprakstu "Iziet uz darbavirsmu"	Peles kreisā taustiņa klikšķis	Datorspēlei pēc trīs sekundēm vajadzētu aizvērties ciet.	PR.GV.01.
TP.GV.02.	Pāriet uz iestatījuma logu	Jāatrodas galvenajā izvēlnē	Pāriešana no galvenās izvēlnes uz iestatījuma logu	1) Peles kreisā taustiņa klikšķis uz pogas ar aprakstu "Iestatījumi"	Peles kreisā taustiņa klikšķis	Lietotājs tiek pārnests uz iestatījuma logu	PR.GV.02.
TP.GV.03.	Pāriet uz pirmspēles logu	Jāatrodas galvenajā izvēlnē	Pāriešana no galvenās izvēlnes uz pirmspēles logu	1) Peles kreisā taustiņa klikšķis uz pogas ar aprakstu "Spēlēt"	Peles kreisā taustiņa klikšķis	Lietotājs tiek pārnests uz pirmspēles logu	PR.GV.03.
TP.GV.04.	Atgriezties galvenajā izvēlnē	Jāatrodas iestatījuma logā	Pāriešana no iestatījumiem uz galvenās izvēlnes logu	1) Peles kreisā taustiņa klikšķis uz pogas ar aprakstu "X"	Peles kreisā taustiņa klikšķis	Lietotājs tiek pārnests uz galvenās izvēlnes logu	PR.GV.04.
TP.GV.05.	Atgriezties galvenajā izvēlnē	Jāatrodas pirmspēles logā	Pāriešana no pirmspēles loga uz galvenās izvēlnes logu	1) Peles kreisā taustiņa klikšķis uz pogas ar aprakstu "X"	Peles kreisā taustiņa klikšķis	Lietotājs tiek pārnests uz galvenās izvēlnes logu	PR.GV.04.
TP.IES.01.	Nomainīt ekrāna rezolūciju	1) Jāatrodas iestatījuma logā. 2) Rezolūcijai ir jābūt uzliktai uz 3440x1440.	Nomainīt ekrāna rezolūciju no 3440x1440 uz 2560x1080	1) Peles kreisā taustiņa klikšķis uz rezolūcijas izvēlnes 2) Peles kreisais taustiņa klikšķis uz izvēli ar aprakstu "2560X1080"	Divi peles kreisā taustiņa klikšķi	Ekrāna rezolūcija tiek nomainīta uz 2560x1080	PR.IES.01.

TP.IES.02.	Nomainīt darbības aktivizācijas pogu	1) Jāatrodas iestatījuma logā. 2) Staigāšana uz priekša pogai ir jābūt "W" taustiņam	Nomainīt staigāšanas uz priekšu pogu no "W" uz "N" taustiņu	1) Peles kreisā taustiņa klikšķis uz staigāšanas uz priekšu pogas lauku 2) Nospiež "N" taustiņu	Peles kreisā taustiņa klikšķis un "N" taustiņā nospiešana	Staigāšanas uz priekšu poga tiek nomainīta uz "N"	PR.IES.02.
TP.IES.03.	Nenomainīt darbības aktivizācijas pogu	1) Jāatrodas iestatījuma logā. 2) Staigāšana uz priekša pogai ir jābūt "W" taustiņam 3) Staigāšana uz atpakaļ pogai ir jābūt "S" taustiņam	Nenomainīt staigāšanas uz priekšu pogu, jo poga "S" jau ir izmantota	1) Peles kreisā taustiņa klikšķis uz staigāšanas uz priekšu pogas lauku 2) Nospiež "S" taustiņu	Peles kreisā taustiņa klikšķis un "S" taustiņā nospiešana	Staigāšanas uz priekšu poga tiek nomainīta uz "N"	PR.IES.02.
TP.PS.01.	Spēle tēla izvēle	Jāatrodas pirmspēles logā.	Izvēlēties pirmo spēlētāju	1) Peles kreisā taustiņa klikšķis uz pirmā spēlētāja attēla	Peles kreisā taustiņa klikšķis	Tiek spēlēta izvēles skaņa un paradās līmeņu izvēle	PR.PS.01.
TP.PS.02.	Spēle līmeņa izvēle	Jāatrodas pirmspēles logā.	Izvēlēties otro līmeni	1) Peles kreisā taustiņa klikšķis uz pirmā spēles līmeņa attēla	Peles kreisā taustiņa klikšķis	Tiek spēlēta izvēles skaņa un paradās spēles logs, tam ir jābūt izvēlētam līmenim	PR.PS.02.
TP.PS.03.	Spēle līmeņa izvēle	Jāatrodas pirmspēles logā.	Izvēlēties otro līmeni	1) Peles kreisā taustiņa klikšķis uz otrā spēles līmeņa attēla	Peles kreisā taustiņa klikšķis	Tiek spēlēta izvēles skaņa un paradās spēles logs, tam ir jābūt izvēlētam līmenim	PR.PS.02.
TP.PS.04.	Palīdzības loga atvēršana	Jāatrodas pirmspēles logā.	Atvērt vaļā palīdzības logu	1) Peles kreisā taustiņa klikšķis uz pogas ar apzīmējumu "i"	Peles kreisā taustiņa klikšķis	Tiek spēlēta pogas skaņa un paradās palīdzības logs ar pirmo lapaspusi	PR.PS.03.
TP.PS.05.	Palīdzības loga aizvēršana	1) Jāatrodas pirmspēles logā. 2) Jābūt atvērtam palīdzības logam.	Aizvērt ciet palīdzības logu	1) Peles kreisā taustiņa klikšķis uz pogas ar apzīmējumu "X"	Peles kreisā taustiņa klikšķis	Tiek spēlēta pogas skaņa un tiek aizvērts palīdzības logs	PR.PS.03.

TP.SP.01.	Staigāšana uz priekšu	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies	Staigāt uz priekšu izmantojot "W"	1) Tur "W" taustiņu	"W" taustiņa turēšana	Spēles tēls staigā uz priekšu	PR.SP.01.
TP.SP.02.	Staigāšana uz atpakaļ	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies	Staigāt uz atpakaļ izmantojot "S"	1) Tur "S" taustiņu	"S" taustiņa turēšana	Spēles tēls staigā uz atpakaļ	PR.SP.01.
TP.SP.02.	Skatīšanās apkārt	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies	Skatīties apkārt	1) Kustina peli	Datorspēles kustināšana	Spēles kamera tiek pakustināta	PR.SP.02.
TP.SP.04.	Skriešana uz priekšu	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies	Skriet uz priekšu	1) Tur "W" taustiņu 2) Tur "Left Shift" taustiņu	"W" un "Left Shift" taustiņu turēšana	Spēlētājs skrien uz priekšu	PR.SP.03.
TP.SP.05.	Ieroča mērķēšana	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies	Mērķēt ieroci	1) Tur peles labo taustiņu	peles labā taustiņa turēšana	Spēlētājs mērķē ieroci un uz ekrāna parādās sarkans punkts	PR.SP.04.
TP.SP.06.	Ieroča pārlādēšana	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies 5) Spēlētājam jābūt vel vismaz viena pārpalikusi lode 6) Spēlētājam jābūt izšautai vismaz viena lode	Pārlādēt ieroci	1) Nospiež taustiņu "R"	Nospiež taustiņu "R"	Spēlētājs pārlādē ieroci un ierocim ir jābūt visas lodes ielādētas.	PR.SP.06.
TP.SP.07.	Pretinieku sekošana	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies	Pretinieks seko spēlētājam	1) Spēlētājs nošauj pretinieku	Spēlētājs nošauj pretinieku	Tiek spēlēta ieraudzīšanas skaņa un visi pretinieki uz laukuma sāk	PR.SP.11.

		5) Spēlētājam jānošauj 9 pretinieki kopā				sekot spēlētājam	
White box							
TP.GV.06.	Iziešana uz darbvirsmas	Jāatrodas galvenajā izvēlnē	Datorspēles aizvēršana ciet	1) Peles kreisā taustiņa klikšķis uz pogas ar aprakstu "Iziet uz darbvirsmu" 2) Izpildās funkcija doExitGame(), kura sākt 3 sekunžu laika atskaiti 3) Pēc trīs sekundēm izpildās funkcija "Application.Quit()"	Peles kreisā taustiņa klikšķis	Datorspēlei pēc trīs sekundēm vajadzētu aizvērties ciet.	PR.GV.01.
TP.GV.07.	Pāriet uz iestatījuma logu	Jāatrodas galvenajā izvēlnē	Pāriešana no galvenās izvēlnes uz iestatījuma logu	1) Peles kreisā taustiņa klikšķis uz pogas ar aprakstu "Iestatījumi" 2) Izpildās funkcija Settings(), kura izpilda darbību SceneManager.LoadScene("Settings", LoadSceneMode.Single)	Peles kreisā taustiņa klikšķis	Lietotājs tiek pārnests uz iestatījuma logu	PR.GV.02.
TP.GV.08.	Pāriet uz pirmspēles logu	Jāatrodas galvenajā izvēlnē	Pāriešana no galvenās izvēlnes uz pirmspēles logu	1) Peles kreisā taustiņa klikšķis uz pogas ar aprakstu "Spēlēt" 2) Izpildās funkcija PreGame(), kura izpilda darbību SceneManager.LoadScene("PreGame", LoadSceneMode.Single)	Peles kreisā taustiņa klikšķis	Lietotājs tiek pārnests uz pirmsspēles logu	PR.GV.03.
TP.GV.09.	Atgriezties galvenajā izvēlnē	Jāatrodas iestatījuma logā	Pāriešana no iestatījumiem uz galvenās izvēlnes logu	1) Peles kreisā taustiņa klikšķis uz pogas ar aprakstu "X" 2) Izpildās funkcija MainMenu(), kura izpilda darbību SceneManager.LoadScene("HomeScreen", LoadSceneMode.Single)	Peles kreisā taustiņa klikšķis	Lietotājs tiek pārnests uz galvenās izvēlnes logu	PR.GV.04.
TP.GV.10.	Atgriezties galvenajā izvēlnē	Jāatrodas pirmspēles logā	Pāriešana no pirmspēles loga uz galvenās izvēlnes logu	1) Peles kreisā taustiņa klikšķis uz pogas ar aprakstu "X" 2) Izpildās funkcija MainMenu(), kura izpilda darbību SceneManager.LoadScene("HomeScreen", LoadSceneMode.Single)	Peles kreisā taustiņa klikšķis	Lietotājs tiek pārnests uz galvenās izvēlnes logu	PR.GV.04.

TP.IES.04.	Nomainīt ekrāna rezolūciju	1) Jāatrodas iestatījuma logā. 2) Rezolūcijai ir jābūt uzliktai uz 3440x1440.	Nomainīt ekrāna rezolūciju no 3440x1440 uz 2560x1080	1) Peles kreisā taustiņa klikšķis uz rezolūcijas izvēlnes 2) Peles kreisais taustiņa klikšķis uz izvēli ar aprakstu "2560X1080" 3) Izpildās funkcija SetResolution() 4) Iedod mainīgam resolution izvēlēto rezolūciju no rezolūcijas masīva. 5) Izdara darbību Screen.SetResolution(resolution.width, resolution.height, Screen.fullScreen)	Divi peles kreisā taustiņa klikšķi	Ekrāna rezolūcija tiek nomainīta uz 2560x1080 un parādās paziņojums konsoles logā.	PR.IES.01.
TP.IES.05.	Nomainīt darbības aktivizācijas pogu	1) Jāatrodas iestatījuma logā. 2) Staigāšana uz priekša pogai ir jābūt "W" taustiņam	Nomainīt staigāšanas uz priekšu pogu no "W" uz "N" taustiņu	1) Peles kreisā taustiņa klikšķis uz staigāšanas uz priekšu pogas lauku 2) Nospiež "N" taustiņu 3) Izpildās funkcija AsignKey() 4) Izpilda funkciju IsKeyAlreadyUsed() 5) Izpilda funkciju AsignKeyToAction() 6) Uztaisa buttonText.text tekstu uz nospiestās vērtības, kura ir "N" 7) Padara ActionName vērtību uz buttonText vērtības 8) Uztaisa PlayerPrefs forward vērtību uz ActionName 9) Saglabā PlayerPrefs vērtību	Peles kreisā taustiņa klikšķis un "N" taustiņā nospiešana	Staigāšanas uz priekšu poga tiek nomainīta uz "N"	PR.IES.02.
TP.IES.06.	Nenomainīt darbības aktivizācijas pogu	1) Jāatrodas iestatījuma logā. 2) Staigāšana uz priekša pogai ir jābūt "W" taustiņam 3) Staigāšana uz atpakaļ pogai ir jābūt "S" taustiņam	Nenomainīt staigāšanas uz priekšu pogu, jo poga "S" jau ir izmantota	1) Peles kreisā taustiņa klikšķis uz staigāšanas uz priekšu pogas lauku 2) Nospiež "S" taustiņu 3) Izpildās funkcija AsignKey() 4) Izpilda funkciju IsKeyAlreadyUsed() 5) Izpilda funkciju AsignKeyToAction() 6) Izpilda funkcija GetDefaultValue(), kur dabūt noklusējuma forward vērtību, kura ir "W" 7) Uztaisa buttonText.text tekstu uz noklusējuma vērtības, kura ir "W" 8) Padara ActionName vērtību uz buttonText vērtības 9) Uztaisa PlayerPrefs forward vērtību uz ActionName 10) Saglabā PlayerPrefs vērtību	Peles kreisā taustiņa klikšķis un "S" taustiņā nospiešana	Staigāšanas uz priekšu poga tiek nomainīta uz "N"	PR.IES.02.

TP.IES.07.	Nomainīt peles kustināšanas ātrumu	1) Jāatrodas iestatījuma logā.	Uzlikt peles ātrumu uz mazāko iespējamo vērtību	1) Peles kreisā taustiņa turēšana uz bīdņa 2) Peles kustināšana uz kreiso pusi 3) Izpildas OnSensitivityChanged() 4) Saglabā bīdņa vērtību PlayerPrefs Sensitivity atslēgai	Peles kreisā taustiņa turēšana un kustināšana pa kreisi	Peles ātrumam ir jābūt 100	PR.IES.03.
TP.IES.08.	Nomainīt mūzikas skaļumu	1) Jāatrodas iestatījuma logā.	Uzlikt mūzikas skaļumu uz mazāko iespējamo vērtību	1) Peles kreisā taustiņa turēšana uz bīdņa 2) Peles kustināšana uz kreiso pusi 3) Izpildās OnMusicChanged() 4) Saglabā bīdņa vērtību PlayerPrefs Music atslēgai	Peles kreisā taustiņa turēšana un kustināšana pa kreisi	Mūzikas skaļumam ir jābūt 0 un nevajadzētu skanēt vairs.	PR.IES.05.
TP.IES.09.	Nomainīt spēles skaņas skaļumu	1) Jāatrodas iestatījuma logā.	Uzlikt mūzikas skaļumu uz mazāko iespējamo vērtību	1) Peles kreisā taustiņa turēšana uz bīdņa 2) Peles kustināšana uz kreiso pusi 3) Izpildas OnSoundChanged() 4) Saglabā bīdņa vērtību PlayerPrefs Sound atslēgai	Peles kreisā taustiņa turēšana un kustināšana pa kreisi	Skaņas skaļumam ir jābūt 0 un nevajadzētu skanēt vairs.	PR.IES.04.
TP.SP.08.	Staigāšana uz priekšu	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlētājam ir jābūt "idle" stāvoklī 5) Spēlei nav jābūt beigusies	Staigāt uz priekšu izmantojot "W"	1) Tur "W" taustiņu 2) Izpildās GetDataMove() funkcija 3) vInput vērtība tiek iestatīta par 1f 4) Direction vērtību normalizē 5) Izdara controller.move() funkciju 6) Sāk atskaņot staigāšanas animāciju, kura iedarbojas, ja vInput ir 1f	"W" taustiņa turēšana	Spēles tēls staigā uz priekšu	PR.SP.01.
TP.SP.09.	Staigāšana uz atpakaļ	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlētājam ir jābūt "idle" stāvoklī 5) Spēlei nav jābūt beigusies	Staigāt uz atpakaļ izmantojot "S"	1) Tur "S" taustiņu 2) Izpildas GetDataMove() funkcija 3) vInput vērtība tiek iestatīta par -1f 4) Direction vērtību normalizē 5) Izdara controller.move() funkciju 6) Sāk atskaņot staigāšanas animāciju, kura iedarbojas, ja vInput ir -1f	"S" taustiņa turēšana	Spēles tēls staigā uz atpakaļ	PR.SP.01.
TP.SP.10.	Spēles tēla ieroča šaušana	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlētājam ir jābūt "aim" stāvoklī 5) Spēlei nav jābūt beigusies 6) Spēlētājam jātur peles labais taustiņš	Šaut spēles ieroci	1) Nospiež peles kreiso taustiņu 2) currentAmmo vērtība viens tiek atņemts	Peles kreisā taustiņa nospiešana	1) Spēlēta izšaušanas skaņa 2) currentAmmo vērtībai atņemts 1 3) Parādās lodes spēles objekts 4) Aktivējas	PR.SP.05.

		7) Spēlētājam ir jābūt vismaz vienai lodei				kameras atsitums 5) Ekrāna teksts parāda jauno currentAmmo vērtību	
TP.SP.11.	Spēles tēla ieroča maiņa	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlētājam ir jābūt "idle" stāvoklī 5) Spēlei nav jābūt beigusies 6) Spēlētājas nevar būt "Reload" un "Guard" stāvokļos	Pārmainīt spēles ieroci	1) Nospiež "Q" taustiņu 2) Maina stāvokli uz Switch 3) Vērtībai lHandIk svaru iestata 0 4) Vērtībai rHandAim svaru iestata 0 5) Konsoles logā parāda paziņojumu Swap	"Q" taustiņa nospiešana	1) Tiek spēlēta maiņas animācija 2) Tiek spēlēta maiņas skaņa 3) Ierocis tiek nomainīts uz citu 4) Spēlētājs aiziet uz "Default" stāvokli	PR.SP.07.
TP.SP.12.	Spēles tēla ieroča bloķēšana	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlētājam ir jābūt "idle" stāvoklī 5) Spēlei nav jābūt beigusies 6) Spēlētājas nevar būt "Reload" un "Swap" stāvokļos	Spēles tēla bloķēšana	1) Nospiež "Space" taustiņu 2) Maina stāvokli uz Guard 3) Vērtībai lHandIk svaru iestata 0 4) Vērtībai rHandAim svaru iestata 0 5) Konsoles logā parāda paziņojumu Guard 6) Visi bojājumi spēlētājam tiek samazināti uz 50% procentiem 7) Dod iespēju bloķēt uguns bumbas	"Space" taustiņa nospiešana	1) Tiek spēlēta bloķēšanas animācija 2) Spēlētājs aiziet uz "Default" stāvokli	PR.SP.08.
TP.SP.13.	Spēles paceļamie objekti	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies	Pacelt laika kapsulu	1) Nospiež "E" taustiņu 2) Tiek spēlēta pacelšanas skaņa 3) Tiek objekts paslēpts 4) Tiek izpildīta AddTime() funkcija 5) Pēc skaņas beigas objekts tiek iznīcināts	"E" taustiņa nospiešana	Tiek pieskaitīts klāt 60 sekundes pie laika	PR.SP.10.
TP.SP.14.	Spēles tēla dzīvības zaudēšana	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies	Pretinieks atņem spēlētājam dzīvības	1) Parastais pretinieks iesit spēlētājam 2) Spēlētājs ieiet reaģēšanas stāvoklī 3) Tiek izpildīta funkcija TakeDamage() 4) Vizuālā dzīvības līnija tiek samazināta par 50 vērtībām	Pretinieks uzbrūk spēlētājam	1) Spēlētājam atņem 50 dzīvības. 2) Spēlētājs izdara	PR.SP.09.

						reaģēšanas animāciju	
TP.SP.15.	Nošaut pretinieku	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies	Spēlētājs nošauj pretinieku	1) Spēlētājs iešauj pretiniekam 2) Pretinieka EnemyHealth vērtība tiek iestatīta uz 0 3) Pretinieks izdara miršanas skaņu 4) Pretiniekam iedarbojas ragdoll 5) Sāk 5 sekunžu atskaiti	Spēlētājs nošauj pretinieku	1) Pretinieks nokrīt uz zemes 2) Pretinieka spēles objekts pēc piecām sekundēm tiek iznīcināts	PR.SP.11.
TP.SP.16.	Pretinieka parādīšanās	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies	Pretinieku parādīšanās uz kartes	1) Nošauj pretinieku 2) Izdara funkcija spawnEnemy() 3) Atrod parādīšanas vietu	Nošauj pretinieku	Parādīsies pretinieks uz kartes nejaušā vietā.	PR.SP.13.
TP.SP.17.	"Kombo skaitītājs"	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies 5) Kombo jābūt iesāktam vismaz 1	Gaidīt kamēr kombo skaitītājs pazūd	1) Pagaida 10 sekundes neko citu nedarot 2) Tiek izdarīta funkicja ResetCombo() 3) Tiek izdarīta funkcija UpdateComboMultiplierText()	Gaida 10 sekundes	Pazūd kombo skaitītājs	PR.SP.14.
TP.SP.18.	Spēles beigšana logs	1) Jāatrodas spēles logā 2) Spēlei ir jābūt sākušai 3) Spēlētājam nevajadzētu būt mirušam 4) Spēlei nav jābūt beigusies 5) Vērtībai killedEnemy ir jābūt 99	Spēles beigšanas loga paradīšanas	1) Spēlētājs nošauj pretinieku 2) Izpildās funkcija showVictoryPanel 3) Spēles objekts victoryPanel ir iestatīts kā true 4) Dabū tagadējo līmeni un spēlētāju 5) Iestata tekstiem punkti, laiks, kopa, ranks vērtības. 6) Parāda peles kursoru	Spēlētājs nošauj pretinieku	1)Parādās uzvares panelis 2)Spēlēta uzvaras skaņa 3)Parādās spēlētāja bilde	PR.SP.15.

6.3. Testēšanas žurnāls

3. tabula

Testēšanas žurnāls

Testēšanas ID	Datums	Testpiemēra ID	Testpiemēra nosaukums	Testētājs	Statuss	Kļūdas ziņojums	Kļūdas ziņojuma Nr.
Black Box							
TZ.01.B.	25.04.2024.	TP.GV.01.	Iziešana uz darbvirsmas	Gustavs Narvils	Veiksmīgs		
TZ.02.B	25.04.2024.	TP.GV.02.	Pāriet uz iestatījuma logu	Gustavs Narvils	Veiksmīgs		
TZ.03.B	25.04.2024.	TP.GV.03.	Pāriet uz pirmspēles logu	Gustavs Narvils	Veiksmīgs		
TZ.04.B	25.04.2024.	TP.GV.04.	Atgriezties galvenajā izvēlnē	Gustavs Narvils	Veiksmīgs		
TZ.05.B	25.04.2024.	TP.GV.05.	Atgriezties galvenajā izvēlnē	Gustavs Narvils	Veiksmīgs		
TZ.06.B.	26.04.2024.	TP.IES.01.	Nomainīt ekrāna rezolūciju	Gustavs Narvils	Veiksmīgs		
TZ.07.B.	26.04.2024.	TP.IES.02.	Nomainīt darbības aktivizācijas pogu	Gustavs Narvils	Veiksmīgs		
TZ.08.B.	26.04.2024.	TP.IES.03.	Nenomainīt darbības aktivizācijas pogu	Gustavs Narvils	Veiksmīgs		
TZ.09.B.	03.06.2024.	TP.PS.01.	Spēle tēla izvēle	Gustavs Narvils	Veiksmīgs		
TZ.10.B.	03.06.2024.	TP.PS.02.	Spēle līmeņa izvēle	Gustavs Narvils	Veiksmīgs		
TZ.11.B.	03.06.2024.	TP.PS.03.	Spēle līmeņa izvēle	Gustavs Narvils	Veiksmīgs		
TZ.12.B.	03.06.2024.	TP.PS.04.	Palīdzības loga atvēršana	Gustavs Narvils	Neveiksmīgs	Paradījās otrā lapaspuse nevis pirmā.	KZ.02
TZ.13.B.	03.06.2024.	TP.PS.04.	Palīdzības loga atvēršana	Gustavs Narvils	Veiksmīgs		
TZ.14.B.	03.06.2024.	TP.PS.05.	Palīdzības loga aizvēršana	Gustavs Narvils	Veiksmīgs		

TZ.15.B.	26.04.2024.	TP.SP.01.	Staigāšana uz priekšu	Gustavs Narvils	Veiksmīgs		
TZ.16.B.	26.04.2024.	TP.SP.02.	Staigāšana uz atpakaļu	Gustavs Narvils	Veiksmīgs		
TZ.17.B.	04.06.2024.	TP.SP.03.	Skatīšanās apkārt	Gustavs Narvils	Veiksmīgs		
TZ.18.B.	04.06.2024.	TP.SP.04.	Skriešana uz priekšu	Gustavs Narvils	Veiksmīgs		
TZ.19.B.	04.06.2024.	TP.SP.05.	Ieroča mērķēšana	Gustavs Narvils	Veiksmīgs		
TZ.20.B.	04.06.2024.	TP.SP.06.	Ieroča pārlādēšana	Gustavs Narvils	Veiksmīgs		
TZ.21.B.	04.06.2024.	TP.SP.07.	Pretinieku sekošana	Gustavs Narvils	Veiksmīgs		
White Box							
TZ.01.W.	27.04.2024.	TP.GV.06.	Iziešana uz darbvirsmas	Gustavs Narvils	Veiksmīgs		
TZ.02.W.	27.04.2024.	TP.GV.07.	Pāriet uz iestatījuma logu	Gustavs Narvils	Veiksmīgs		
TZ.03.W.	27.04.2024.	TP.GV.08.	Pāriet uz pirmspēles logu	Gustavs Narvils	Veiksmīgs		
TZ.04.W.	27.04.2024.	TP.GV.09.	Atgriezties galvenajā izvēlnē	Gustavs Narvils	Veiksmīgs		
TZ.05.W.	27.04.2024.	TP.GV.10.	Atgriezties galvenajā izvēlnē	Gustavs Narvils	Veiksmīgs		
TZ.06.W.	27.04.2024.	TP.IES.04.	Nomainīt ekrāna rezolūciju	Gustavs Narvils	Neveiksmīgs	Neparadījās konsoles loga paziņojums.	KZ.01
TZ.07.W.	27.04.2024.	TP.IES.04.	Nomainīt ekrāna rezolūciju	Gustavs Narvils	Veiksmīgs		
TZ.08.W.	27.04.2024.	TP.IES.05.	Nomainīt darbības aktivizācijas pogu	Gustavs Narvils	Veiksmīgs		
TZ.09.W.	27.04.2024.	TP.IES.06.	Nenomainīt darbības aktivizācijas pogu	Gustavs Narvils	Veiksmīgs		
TZ.10.W.	04.06.2024.	TP.IES.07.	Nomainīt peles kustināšanas ātrumu	Gustavs Narvils	Veiksmīgs		
TZ.11.W.	04.06.2024.	TP.IES.08.	Nomainīt mūzikas skaļumu	Gustavs Narvils	Veiksmīgs		

TZ.12.W.	04.06.2024.	TP.IES.09.	Nomainīt spēles skaņas skaļuimu	Gustavs Narvils	Veiksmīgs		
TZ.13.W.	27.04.2024.	TP.SP.08.	Staigāšana uz priekšu	Gustavs Narvils	Veiksmīgs		
TZ.14.W.	28.04.2024.	TP.SP.09.	Staigāšana uz atpakaļ	Gustavs Narvils	Veiksmīgs		
TZ.15.W.	04.06.2024.	TP.SP.10.	Spēles tēla ieroča šaušana	Gustavs Narvils	Veiksmīgs		
TZ.16.W.	04.06.2024.	TP.SP.11.	Spēles tēla ieroča maiņa	Gustavs Narvils	Veiksmīgs		
TZ.17.W.	04.06.2024.	TP.SP.12.	Spēles tēla ieroča bloķēšana	Gustavs Narvils	Veiksmīgs		
TZ.18.W.	04.06.2024.	TP.SP.13.	Spēles paceļamie objekti	Gustavs Narvils	Veiksmīgs		
TZ.19.W.	04.06.2024.	TP.SP.14.	Spēles tēla dzīvības zaudēšana	Gustavs Narvils	Veiksmīgs		
TZ.20.W.	04.06.2024.	TP.SP.15.	Nošaut pretinieku	Gustavs Narvils	Veiksmīgs		
TZ.21.W.	04.06.2024.	TP.SP.16.	Pretinieka parādīšanās	Gustavs Narvils	Veiksmīgs		
TZ.22.W.	04.06.2024.	TP.SP.17.	"Kombo skaitītājs"	Gustavs Narvils	Veiksmīgs		
TZ.23.W.	04.06.2024.	TP.SP.18.	Spēles beigšana logs	Gustavs Narvils	Veiksmīgs		

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. Manuprāt beigās es uztaisīju spēli, kura man ne tikai liekās par interesantu spēlēt, bet arī tādu, kura man likās interesantā un tā rezultātā es varēju izdārdat visas prasības, ko es biju piešķīris datorspēlei. Ja nākotnē vēlētos pievienot jaunus elementus šai spēlei, tas būtu viegli izdarāms, jo es visu izveidoju tā, lai tas būtu saprotami un salīdzinoši viegli testējams.

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. Kad es sāku taisīt projektu es domāju, ka lielākā daļa kods būs grūti man saprotam, bet pēc projekta pabeigšanas gandrīz viss kods ir loģiski salasāms un saprotams, pat ja neprot C# programmēšanas valodu vai Unity dzini. Manu izstrādāto kodu var apskatīties pielikumā, kur es esmu ielicis manuprāt es esmu ielicis svarīgākās programmas daļas. 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.

Grūtāka daļa, manuprāt, veidojot projektu bija testēt un pārliecinoties, ka viss iet bez nekādām problēmām, tāpēc ka projekts bija liels. Jo gandrīz visas spēles funkcijas un spēles daļas sarunājās ar viens otru, tādēļ būtu ļoti daudzas reizes, kur es domātu, ka funkcija man ir gatava un strādājoša, bet tad citā funkcija dēļ jaunās funkcijas nobruktu un tad man būtu tā jālabo. Tad pēc labošanas man būtu pēc tam jāiziet cauri visai programmai un jāpārmaina visas, tā lai tās strādātu. Bija arī grūti testēt lietas kā pretinieka parādīšanās, tāpēc ka man bija visu laiku jāpārbauda, ja pretinieks neparādās iekšā kaut kādā objektā, jo spēlē bija ļoti daudzu objekti. Bija arī grūti testēt tādus aspektus kā pretinieka parādīšanās, jo man visu laiku bija jāpārliecinās, ka pretinieks neparādās kādā objektā iekšā. Spēlē bija ļoti daudz objektu, un šī pārbaude prasīja daudz laiku.

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

4. 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
WhiteBox	Baltās kastes testpiemērs
BlackBox	Melnās kastes testpiemērs
W	WhiteBox
B	BlackBox
Spēlētājs	Lietotājs

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

Par to, kā uzstatīsit, lai uz objektiem pretinieki nevarētu staigāt:

https://docs.unity3d.com/2020.1/Documentation/Manual/class-NavMesh-ModifierVolume.html

Pielikumi

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.AI;

public class EnemySpawn : MonoBehaviour
{
    public GameObject enemyBasic; //Parastais pretinieks
    public GameObject enemySpell; //Burvju pretinieks
    public GameObject enemyHeavy; //Stiprais pretinieks
    public Transform player; //Spēlētajs
    public int maxEnemies = 100; //Maksimālais pretinieku daudzums
    public float spawnRadius = 100f; //Parādīšanas rādius
    public int initialSpawnCount = 10; //Cik pretinieki parādās sākumā
    public float minSpawnDistance = 10f; //Minimālais parādīšanas distance

    public int currentEnemyCount = 0; //Cik tagad ir pretinieki uz laukuma
    public int totalEnemyCount = 0; //Cik ir bijuši kopā pretinieki
    public int killedEnemy = 0; //Cik pretinieki ir nogalināti

    public GameTime gameTime;
    private void Awake()
    {
        //Dabū spēlētaju
        int selectedPlayer = PlayerPrefs.GetInt("SelectedPlayer", 1);
        GameObject player1Object = GameObject.Find("Player_1");
        GameObject player2Object = GameObject.Find("Player_2");
        if (selectedPlayer == 1)
        {
            if (player1Object != null)
            {
                player = player1Object.transform;
            }
        }
        else if (selectedPlayer == 2)
        {
            if (player2Object != null)
            {
                player = player2Object.transform;
            }
        }

    }
    void Start()
    {
        //Dabū UI objektu, lai dabūtu gameTime skriptu
        GameObject UI = GameObject.Find("UI");
        if (UI != null)
        {
            gameTime = UI.GetComponent<GameTime>();
            if (gameTime == null)
            {
                Debug.LogError("GameTime skripts neatrasts");
            }
        }
        //Parādās sākuma pretinieki
        for (int i = 0; i < initialSpawnCount; i++)
        {
            SpawnEnemy();
        }
    }
```

21.attēls pretinieku parādīšanas skripta pirmā daļa

```
void Update()
{
    //Ja ir nošauti visi pretinieki spēle beidzās
    if (killedEnemy == maxEnemies)
    {
        gameTime.gameIsOver = true;
    }
    //Pārbauda vai parādītie pretinieku skaits lielāks par maksimālu
    if (totalEnemyCount < maxEnemies)
    {
        //Cik pretinieki vel ir jāparāda
        int enemiesToSpawn = initialSpawnCount - currentEnemyCount;

        if (enemiesToSpawn > 0)
        {
            //Parāda vajadzīgos pretiniekus
            for (int i = 0; i < enemiesToSpawn; i++)
            {
                SpawnEnemy();
            }
        }
    }
}
// Parādās pretinieks
void SpawnEnemy()
{
    Vector3 spawnPosition;
    do
    {
        spawnPosition = Random.insideUnitSphere * spawnRadius;
        spawnPosition += transform.position;
        spawnPosition.y = Terrain.activeTerrain.SampleHeight(spawnPosition);
    } while (Vector3.Distance(spawnPosition, player.position) < minSpawnDistance);

    GameObject enemyToSpawn;
    //Stiprais pretinieks pārādas tikai, kā 25, 50, 75, 99 un 100 pretinieks
    if (totalEnemyCount == 25 || totalEnemyCount == 50 || totalEnemyCount == 75 || totalEnemyCount == 98 || totalEnemyCount == 99)
    {
        enemyToSpawn = enemyHeavy;
    }
    else
    {
        //Ja neparādās stiprais pretinieks tad ir 50% iespēja, ka parādīsies parastais vai burvju pretinieks.
        float randomNumber = Random.Range(0f, 1f);
        if (randomNumber <= 0.5f)
        {
            enemyToSpawn = enemyBasic;
        }
        else
        {
            enemyToSpawn = enemySpell;
        }
    }

    Instantiate(enemyToSpawn, spawnPosition, Quaternion.identity);
    currentEnemyCount++;
    totalEnemyCount++;

    if (totalEnemyCount >= maxEnemies)
    {
        Debug.Log("Maksimālais pretinieku daudzums sasniekts");
    }
}
```

22.attēls pretinieku parādīšanas skripta otrā daļa

```
using JetBrains.Annotations;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.AI;
using UnityEngine.ProBuilder;

public class EnemyAi : MonoBehaviour
{

    public NavMeshAgent agent;

    public Transform player; //Spēlētajs

    public LayerMask whatIsGround, whatIsPlayer;

    public Animator animator;

    public int damageToPlayer; // Cik daudz dzīvības atņems spēlētājam

    private bool staggered = false; //Vai pretinieks ir apstādināts

    public GameObject spell; //Uguns bumba
    public bool hasSpells = true; //Vai var mest uguns bumbas

    //Kad pretinieks nav atradis spēlētāji viņš staigā uz riņķa notiektajā rādijusā
    public Vector3 walkPoint;
    bool walkPointSet;
    public float walkPointRange;

    public float timeBetweenAttacks; // Laiks starp uzbrukumiem
    bool alreadyAttacked; // Vai nav jau uzbrucis

    public float sightRange, attackRange; // Attālums, kur pretinieks redz spēlētāju un attālums, kur pretinieks var uzbrukt.
    public bool playerInSightRange, playerInAttackRange; //Vai spēlētāju redzu un vai var viņam uzbrukt

    private bool spotSoundPlayed = false; //Vai spēlētāja redzēšanas skaņa bija spēlēta

    public bool hasBeenHit = false; //Vai spēlētājs ir iešāvis pretiniekam

    public EnemySpawn enemySpawn;

    private void Awake()
    {
        //Dabū spēlētāja objektu, kā arī parādīšanās komponentu
        int selectedPlayer = PlayerPrefs.GetInt("SelectedPlayer", 1);
        GameObject selectedPlayerObject = null;

        if (selectedPlayer == 1)
        {
            selectedPlayerObject = GameObject.Find("Player_1");
        }
        else if (selectedPlayer == 2)
        {
            selectedPlayerObject = GameObject.Find("Player_2");
        }
        player = selectedPlayerObject.transform;
        agent = GetComponent<NavMeshAgent>();
        agent.speed = Random.Range(3f, 7.5f);

        GameObject enemiesObject = GameObject.Find("Enemies");
        enemySpawn = enemiesObject.GetComponent<EnemySpawn>();



    }
```

23.attēls pretinieku uzvedības skripta pirmā daļa

```
private void Update()
{
    //Ja pretinieks ir apstādināts izej ārā
    if (staggered)
        return;

    PlayerHealth playerHealth = player.GetComponent<PlayerHealth>();
    //Ja spēlētajs ir nomiris, tad beidz viņam sekot
    if (playerHealth != null && playerHealth.isDead)
    {
        Patroling();
        animator.SetBool("Walk", true);
        animator.SetBool("Attack", false);
        return;
    }

    //Ja ir nošauti  10 pretinieki, tad viņi visu laiku sekos spēlētājam
    if (enemySpawn != null && enemySpawn.killedEnemy >= 10)
    {
        hasBeenHit = true;
    }
    //Ja ir nošauti 50 pretinieki, viņu ātrums palielināsies uz gandrīz maksimālo iešanas ātrumu
    else if (enemySpawn != null && enemySpawn.killedEnemy >= 50) {
        agent.speed = 6.5f;
    }
    //Ja ir nošauti 75 pretinieki, viņu ātrums palielināsies uz maksimālo iešanas ātrumu
    else if (enemySpawn != null && enemySpawn.killedEnemy >= 75)
    {
        agent.speed = 7.5f;
    }



    //Pārbauda redzi un uzbrukšanas attālumu
    playerInSightRange = Physics.CheckSphere(transform.position, sightRange, whatIsPlayer);
    playerInAttackRange = Physics.CheckSphere(transform.position, attackRange, whatIsPlayer);
    //Ja spēlētāju neredz un nav uzbrukšanas attāluma, tad staigā uz vietas
    if (!playerInSightRange && !playerInAttackRange || !playerInAttackRange && !hasBeenHit)
    {
        Patroling();
        animator.SetBool("Walk", true);
        animator.SetBool("Attack", false);
    }
    //Ja spēlētāju redz, bet nevar uzbrukt sāc sekot
    if (playerInSightRange && !playerInAttackRange || !playerInAttackRange && hasBeenHit)
    {
        ChasePlayer();
        animator.SetBool("Walk", true);
        animator.SetBool("Attack", false);
        if (!spotSoundPlayed)
        {
            EnemyHealth enemyHealth = GetComponent<EnemyHealth>();
            if (enemyHealth != null)
            {
                //Spēlē skaņu
                enemyHealth.EnemySpottedSound();
                spotSoundPlayed = true;
            }
        }

    }
    //Uzbrūc spēlētājam ja redz un ir tuvu
    if (playerInSightRange && playerInAttackRange || hasBeenHit && playerInAttackRange)
    {
        //Ja var mest uguns bumbas, tad met, bet ja nē tad uzbrūc fiziski
        if (hasSpells)
        {
            ProjectileAttackPlayer();
        }
        else
        {
            MeleeAttackPlayer();
        }
        animator.SetBool("Walk", false);
        animator.SetBool("Attack", true);
    }
}
```

24.attēls pretinieku uzvedības skripta otrā daļa

```
//Staigā uz vietas noteiktā rādiusā
private void Patroling()
{
    if (!walkPointSet) SearchWalkPoint();
    if (walkPointSet) agent.SetDestination(walkPoint);

    Vector3 distanceToWalkPoint = transform.position - walkPoint;

    if (distanceToWalkPoint.magnitude < 1f) walkPointSet = false;
}
//Meklē nākošo ceļu, kamēr iet uz vietas
private void SearchWalkPoint()
{
    float randomZ = Random.Range(-walkPointRange, walkPointRange);
    float randomX = Random.Range(-walkPointRange, walkPointRange);

    walkPoint = new Vector3(transform.position.x + randomX, transform.position.y, transform.position.z + randomZ);

    if (Physics.Raycast(walkPoint, -transform.up, 2f, whatIsGround)) walkPointSet = true;
}
//Seko spēlētājam
private void ChasePlayer()
{
    animator.SetBool("Reaction", false);
    agent.SetDestination(player.position);
}
//Uzbrūc fiziski spēlētājam
private void MeleeAttackPlayer()
{
    agent.SetDestination(transform.position);
    transform.LookAt(player);

    if (!alreadyAttacked)
    {
        EnemyHealth enemyHealth = GetComponent<EnemyHealth>();
        if (enemyHealth != null)
        {
            enemyHealth.EnemyAttackSound();
        }
        animator.SetTrigger("Attack");

        float animationDuration = 1f;
        Invoke(nameof(DamagePlayer), animationDuration);

        alreadyAttacked = true;
        Invoke(nameof(ResetAttack), timeBetweenAttacks);
    }
}
//Parāda ugunsbumbu, pēc īsa laika
private IEnumerator SpawnProjectileWithDelay(float delay)
{
    yield return new WaitForSeconds(delay);

    Vector3 spawnPosition = transform.position + transform.forward * 1.5f + transform.up * 1.5f;
    Rigidbody rb = Instantiate(spell, spawnPosition, Quaternion.identity).GetComponent<Rigidbody>();
    rb.velocity = Vector3.zero;
    Vector3 projectileDirection = transform.forward + Vector3.up * 0.25f;
    float forceMagnitude = 15f;
    rb.AddForce(projectileDirection * forceMagnitude, ForceMode.Impulse);

}
```

26.attēls pretinieku uzvedības skripta trešā daļa

```
    //Uzbrūc spēlētājam izmantojot ugunsbumbas
    private void ProjectileAttackPlayer()
    {
        agent.SetDestination(transform.position);
        transform.LookAt(player);

        if (!alreadyAttacked)
        {
            EnemyHealth enemyHealth = GetComponent<EnemyHealth>();
            if (enemyHealth != null)
            {
                enemyHealth.EnemyAttackSound();
            }
            animator.SetTrigger("Attack");

            StartCoroutine(SpawnProjectileWithDelay(0.5f));
            alreadyAttacked = true;
            Invoke(nameof(ResetAttack), timeBetweenAttacks);
        }
    }

    //Atņem dzīvības spēlētājam
    private void DamagePlayer()
    {
        if (playerInAttackRange)
        {
            PlayerHealth playerHealth = player.GetComponent<PlayerHealth>();
            ActionStateManager actions = player.GetComponentInChildren<ActionStateManager>();
            if (playerHealth != null)
            {
                if (actions.currentState == actions.Guard) playerHealth.TakeDamage(damageToPlayer / 2);
                else playerHealth.TakeDamage(damageToPlayer);
            }
        }
    }
    //Restartē uzbrukumu
    private void ResetAttack()
    {
        alreadyAttacked = false;
    }
    //Apstādina pretinieku
    public void Stagger()
    {
        agent.isStopped = true;
        staggered = true;
        animator.SetBool("Walk", false);
        animator.SetBool("Attack", false);
        animator.SetBool("Reaction", true);
        Debug.Log("Enemy staggered");

        StartCoroutine(RecoverFromStagger());
    }
    //Pēc apstādināšanas sāk atkal iet
    private IEnumerator RecoverFromStagger()
    {
        yield return new WaitForSeconds(1.2f);
        EnemyHealth enemyHealth = GetComponent<EnemyHealth>();
        if (enemyHealth != null && !enemyHealth.isDead)
        {
            staggered = false;
            agent.isStopped = false;
            animator.SetBool("Reaction", false);
        }
    }
    //Apstādina animācijas pretinieka
    public void StopAnimations()
    {
        animator.SetBool("Walk", false);
        animator.SetBool("Attack", false);
        animator.SetBool("Reaction", false);
    }


}
```

27.attēls pretinieku uzvedības skripta ceturtā daļa