Razvoj mobilne igre u Unity okruženju

Cica, Filip

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

Razvoj mobilne igre u Unity okruženju

Završni rad

Filip Cica

Osijek, 2022.

SADRŽAJ

1. UVOD

U današnjem svijetu gdje gotovo svi posjeduju nekakav oblik računala, industrija računalnih igara postaje iz godine u godinu sve vrjednija i naprednija. Ta industrija 2022. godine vrijedi 300 milijardi američkih dolara [1]. Dio te industrije su neovisne (engl. *Independent*, *Indie*) igre. *Indie* igra je računalna igra koju razvija pojedinac ili manja skupina ljudi bez financijske podrške nekog velikog izdavača.

U radu će biti istraženo što sve ulazi u razvoj jedne ležerne *indie* mobilne igre. Ležerna igra (engl. *casual game*) je jednostavna igra koju je lako naučiti igrati te koju se povremeno igra na kratko vrijeme [2]. Igra će biti razvijena na takav način da je spremna za tržište ali i dalje otvorena za nadogradnju i preradu. Naravno, nemoguće je izmisliti potpuno novi i nikad prije viđen koncept kad se uzme u obzir broj *indie* igara dostupnih na internetu. U radu će se vidjeti kako nadograditi i spojiti već postojeće koncepte i modele postojećih igara te njihovim spojem postići novu ležernu *indie* mobilnu igru koja će zabaviti i očarati igrača.

Za početak će se definirati zahtjevi na programsko rješenje (igru) korisničkim pričama kojima će se iz perspektive igrača odrediti što je to što će najviše zabaviti i oduševiti igrača. Nakon toga će se napraviti osvrt na poznatije tehnologije korištene za razvoj *indie* igara te će se detaljnije opisati Unity, razvojno okruženje za razvoj igara [3] i reći će se zašto je baš ta tehnologija odabrana za implementaciju prethodno određenih zahtjeva. Nadalje će biti prikazan proces implementacije igre gdje će se vidjeti razne mogućnosti koje Unity razvojno okruženje nudi te će biti opisane specifične funkcionalnosti koje su postignute programskim rješenjem. Provesti će se funkcionalno testiranje igre te će se popraviti pronađeni nedostatci. Nakon čega će igra biti dana na testiranje grupi korisnika kako bi mogući nedostatci bili pronađeni i popravljeni. Na kraju će se razmotriti i opisati koraci potrebni da se igra plasira na tržište.

1.1. Zadatak završnog rada

Zadatak završnog rada je prikazati tijek izrade ležerne *indie* mobilne igre korištenjem Unity razvojnog okruženja. Potrebno je dokumentirati inspiraciju za izradu igre i zahtjeve na programsko rješenje korisničkim pričama. Prikazati grafički i arhitekturni dizajn cjelokupne igre te implementirati igru.

2. PREGLED PODRUČJA RADA

Razmotrit će se nekoliko programskih rješenja sličnih onome što se želi od igre opisane u ovom radu. U ovom slučaju to su konkretne mobilne igre dostupne u Play Trgovini na bilo kojem Android uređaju. Izabrane su tri trodimenzionalne (3D) igre koje sadrže sličnu mehaniku koja će biti implementirana u igri koja se opisuje ovim radom. Tema tih igara je mini golf što znači da sve funkcioniraju na sličan način tj. loptica se mora dovesti do rupe u što manje pokreta prelazeći prepreke. Uz navedene 3D igre odabrane su još dvije dvodimenzionalne (2D) igre kod kojih se neće obraćati pozornost samo na njihovu mehaniku nego i na grafički dizajn koji je jednostavan i ugodan. Dakle cilj je spojiti mehaniku navedenih 3D igara sa grafičkim dizajnom navedenih 2D igara kako bi se dobila 2D mini golf igra sa jednostavnim i ugodnim grafičkim dizajnom.

2.1. 3D mobilne igre

Tri igre koje su izabrane se mogu naći u Play Trgovini pod nazivima *Mini Golf King*, *Vista Golf* i *Mini Golf*. Mini golf igre su jako stare te postoji veliki broj sličnih računalnih i mobilnih igara. Navedene tri su izabrane jer su najpopularnije i najbolje prikazuju što se želi postići u programskom rješenju igre opisane ovim radom. Iako jako slične, svaka od njih sadrži neke specifičnosti. To su 3D igre, svaka sa više od 10 milijuna preuzimanja što znači da se može reći da su relativno popularne igre. Sve tri su jako slične i postoji više klonova takvih igara.

Mini golf igre uobičajeno imaju različite terene od kojih grupa igrača odlučuje na kojem će se terenu igrati i onaj tko je u najmanje poteza pogodio rupu je pobjednik. Slične funkcionalnosti pruža i *Mini Golf King* igra navedena gore. U igri se igra protiv drugog igrača jedan na jedan i pobjednik dobiva nekakve bodove. Igra sadrži zanimljiv sustav nagrađivanja igrača gdje se od igrača traži da tijekom šutanja lopte do cilja skuplja kristale i na temelju količine skupljenih kristala biva nagrađen [4]. Kod definiranja zahtjeva igre, ugrađivanje sistema nagrađivanja igrača je jedna od poželjnih funkcionalnosti.

U *Vista Golf* igri postoji konačan broj razina koje se igraju samostalno, svaka razina je teža od prethodne [5]. Igra posjeduje jednostavan grafički dizajn i obilježavaju je kremaste boje, također igra nije vrlo mehanički zahtjevna no i dalje je vrlo zabavna za igrati. Što se tiče ležernih igara, prelaženje jednostavnih razina od kojih je svaka teža od prethodne čini se kao nešto što svaka ležerna igra mora imati.

U igri nazvanoj *Mini Golf* može se birati između igranja sa više igrača i igranja samostalno prolazeći kroz razine igre. Ima malo kompleksniju mehaniku sa raznim vrstama dinamičkih prepreka i jako je lijepo grafički dizajnirana. Igra sadrži jako puno različitih prepreka od kojih neke imaju posebna svojstva a neke su i dinamične [6]. Imati puno različitih prepreka u igri obilježje je koje bi omogućilo postepeno povećavanje razinu težine i nešto što bi igrača zabavljalo duže vremena.

2.2. 2D mobilne igre

U nastavku će biti navedene dvije 2D igre potpuno različite od prethodno opisanih 3D igara. Dvije igre koje su izabrane su *Color Switch* i *I Love Hue Too*. Igre su jednostavno grafički dizajnirane, korištenjem jednostavnih boja i tekstura što su karakteristike koje će se implementirati i u igru napravljenu u okviru ovog rada jer igre sa previše efekata, animacija, zvukova i blještavih boja mogu izmoriti igrača.

Color Switch je 2D igra sa više od 10 milijuna preuzimanja na Play Trgovini, karakteriziraju je šaren dizajn, jednostavni zvučni efekti i jednostavna mehanika. Zanimljiva funkcionalnost kod *Color Switch* je to da ima više načina (modova) igranja (engl. *game mode*) od kojih je osnovni mod igre beskonačna razina. Na beskonačnoj razini uobičajeno postoji konačan broj različitih elemenata koji su uglavnom prepreke koji se nasumično pojavljuju pred igračem [7]. Osim što je poželjno da igra opisana u radu bude jednostavna i da bude 2D kao *Color Switch,* bilo bi jako zanimljivo i poželjno implementirati beskonačni mod u mini golf igri jer bi rezultat bio jedinstvena, neobična i zanimljiva funkcionalnost.

I Love Hue To je vizualno zadovoljavajuća 2D igra u kojoj se slaže slagalica. Igrač dobiva sliku na kojoj su prikazane palete boja koje se pobrkaju i tada je zadaća igrača da ih posloži [8]. U ovoj igri najzanimljivija stvar je prikaz paleta boja na potpuno crnoj pozadini sa bijelim linijama. *I Love Hue To* poslužila je kao inspiracija za dizajn igre izrađene u okviru ovog rada. Navedena igra također vodi igrača kroz priču, što je vrlo zanimljiv aspekt naoko vrlo obične igre i što bi se moglo uzeti u obzir kod dizajniranja i definiranja zahtjeva igre u ovom radu.

3. ZAHTJEVI NA PROGRAMSKO RJEŠENJE

Od mnogih načina da se zahtjevi na programsko rješenje definiraju najviše se ističu: funkcionalni i nefunkcionalni zahtjevi, dijagrami slučajeva korištenja (engl. *use case diagram*), umne mape i korisničke priče (engl. *user story*). Kod definiranja zahtjeva za igru čini se najbolje odabrati korisničke priče jer olakšavaju planiranje implementacije. Iz korisničkih priča se vidi kojim redom je najefikasnije implementirati pojedine elemente igre. Sa korisničkim pričama se dizajner igre stavlja u perspektivu igrača i kroz pisanje korisničkih priča odmah se može zamisliti kako će igra izgledati i kako će se ponašati što je vrlo korisno kod dizajniranja izgleda i ponašanja igre. Korisničke priče nisu egzaktne tako da se mogu interpretirati na više načina i daju dizajneru više slobode pri dizajniranju igre. Igra se može opisati s nekoliko velikih korisničkih priča. Jedna veća korisnička priča se zove ep. Dobra praksa je rastaviti epove na manje korisničke priče gdje sam ep služi kao naslov iz kojeg se vidi na što se odnose korisničke priče dobivene iz tog epa. Kolekcija epova se naziva inicijativa, inicijativa predstavlja cjelokupni cilj. Inicijativa ovih epova bi bila napraviti mobilnu ležernu *indie* igru. Kod implementacije igre, korisničke priče će služiti kao upute. Korisničke priče su uglavnom složene onim redoslijedom kojim će se implementirati elementi igre.

Epovi i korisničke priče su prikazane tablicama kako bi bilo lakše referencirati ih u poglavljima koja slijede. Tablice neće biti dodatno objašnjenje jer su same po sebi razumljive.

Ep 1	„Kao igrač želim da igra bude 2D mini golf.“
Priča 1	„Kao igrač želim da igra ima teren preko cijelog zaslona.“
Priča 2	„Kao igrač želim da se teren može povećavati i smanjivati potezima prsta po zaslonu.“
Priča 3	„Kao igrač želim da na donjoj strani terena bude loptica a na gornjoj rupa.“
Priča 4	„Kao igrač želim da igra ima prepreke.“
Priča 5	„Kao igrač želim da ako ne postignem pogodak u zadani broj udaraca da izgubim.“
Priča 6	„Kao igrač želim da ako postignem pogodak u zadani broj udaraca da pobijedim.“
Priča 7	„Kao igrač želim da mi se u slučaju pobijede ili gubitka prikaže prozor sa mojim rezultatima“

Tablica 3.1. Ep 1

Ep 2	„Kao igrač želim da igra ima Glavni izbornik.“
Priča 1	„Kao igrač želim da glavni izbornik ima gumb koji vodi na izbornik kolekcija.“
Priča 2	„Kao igrač želim da glavni izbornik ima gumb koji vodi na beskonačnu razinu.“
Priča 3	„Kao igrač želim da izbornik ima gumb koji otvara postavke“
Priča 4	„Kao igrač želim da glavni izbornik ima gumb koji otvara prikaz mojih podataka i rezultata“

Tablica 3.2. Ep 2

Ep 3	„Kao igrač želim da igra ima izbornik kolekcija.“
Priča 1	„Kao igrač želim da svaka razina prati dizajn opisan u prvom epu“
Priča 2	„Kao igrač želim da razine budu grupirane u kolekcije.“
Priča 3	„Kao igrač želim da gumbovima prikazanim na izborniku kolekcija otvaramo kolekcije razina.“
Priča 4	„Kao igrač želim da sve razine budu zaključane osim prve.“
Priča 5	„Kao igrač želim da ostvarivanjem pobjede na prvoj razini otključavamo drugu i tako nadalje.“
Priča 6	„Kao igrač želim da svaka razina bude teža od prethodne“

Tablica 3.3. Ep 3

EP 4	„Kao igrač želim da igra ima beskonačnu razinu“
PRIČA 1	„Kao igrač želim da se teren može povećavati i smanjivati potezima prsta po zaslonu.“
PRIČA 2	„Kao igrač želim da na donjoj strani terena bude loptica a na gornjoj rupa.“
PRIČA 3	„Kao igrač želim da igra ima prepreke.“
PRIČA 4	„Kao igrač želim da se pri početku igranja otvaranju beskonačne razine generira nasumični teren“

Priča 5	„Kao igrač želim da se pri kliku gumba sa glavnog izbornika koji vodi na beskonačnu razinu se pokaže izbornik sa gumbom koji pokreće generiranje razine“
PRIČA 6	„Kao igrač želim da igra u slučaju pogotka u zadani broj koraka na mjestu rupe generira lopticu a iznad da generira novi nasumični teren i novi broj koraka “
PRIČA 7	„Kao igrač želim da svaki novi nasumični teren bude malo teži od prethodnog.“
PRIČA 8	„Kao igrač želim da u slučaju da ne ostvarim pogodak u zadani broj koraka da izgubim.“
PRIČA 9	„Kao igrač želim da mi se kad izgubim prikaže zaslon sa mojim rezultatom i da mi se da mogućnost da ponovo igram.“
PRIČA 10	„Kao igrač želim da mi se rezultat računa prema broju koraka koji su mi ostali i prema vremenu u kojemu sam ostvario pogodak.“

Tablica 3.4. Ep 4

EP 5	„Kao igrač želim da igra ima Glazbu i zvučne efekte“
Priča 1	„Kao igrač želim da se u pozadini čuje [1]lofi pjesma koja će se vrtjeti u krug.“
Priča 2	Kao igrač želim da se tijekom igranja igre, kod sudaranja loptice sa preprekama čuju zvučni efekti.“
Priča 3	„Kao igrač želim da kod interakcije sa UI elementima igre čujemo zvučne efekt.“

Tablica 3.5. Ep 5

EP 6	„Kao igrač želim da igra ima jednostavan grafički dizajn.“
PRIČA 1	„Kao igrač želim da teren i pozadine izbornika budu crne.“
Priča 2	„Kao igrač želim da loptica bude bijele boje.“
Priča 3	„Kao igrač želim da svaki gumb bude drugačije kremaste boje.“
Priča 4	„Kao igrač želim da prepreke budu raznih boja i oblika.“

Tablica 3.6. Ep 6

[1] Lofi je žanr glazbe kod kojeg se uobičajeno čuju smetnje i čiji su zvukovi uglavnom malo izobličeni

Ostale korisniče priče	
Priča 1	„Kao igrač želim imati mogućnost autentifikacije mailom tako da mi napredak u igri ostane spremljen tj. ako izbrišem igru pa je ponovo instaliram da se napredak ne izbriše.
Priča 2	„Kao igrač želim da se prije otvaranja glavnog izbornika, izbornika razine i beskonačne razine privremeno prikaže zaslon za učitavanje koji će prikazivati jednostavnu animaciju.“
Priča 3	„Kao igrač želim imati mogućnost mijenjanja teme igre. Neka bude nekoliko mogućih izbora. Neka se nove teme otključavaju prelaženjem razina ili kupuju bodovima.“
Priča 4	„Kao igrač želim da igra ima nekakav sistem nagrađivanja igrača“
Priča 5	„Kao igrač želim imati mogućnost mijenjanja izgleda loptice.“
Priča 6	„Kao igrač želim da se igra može igrati samo u portret položaju mobilnog uređaja.“
Priča 7	Kao igrač želim imati mogućnost otvaranja postavki gdje ću moći mijenjati različite stavke igre kao što su glasnoća glazbe, tema, izgleda loptice itd.“
Priča 8	„Kao igrač želim da igra ima nekakvu priču iza sebe. Neka mi se prije prelaženja svake razine prikaže dijalog koji će me uvesti u nekakvu priču.“
Priča 9	„Kao igrač želim da sav tekst u igri bude na engleskom jeziku.“
Priča 10	„Kao developer želim imati mogućnost dodavati nove razine bez velikih promjena u kodu.“
Priča 11	„Kao developer želim imati mogućnost jednostavno dodati nove vrste prepreka bez velikih promjena u kodu.“
Priča 12	„Kao developer želim moći izmjenjivati UI bez velikih promjena u kodu.“

Tablica 3.7. Ostale korisničke priče

4. TEHNOLOGIJE ZA IZRADU IGARA

U ovom poglavlju će biti opisane tehnologije najčešće korištene za razvoj *indie* mobilnih igara. Tehnologije koje će biti opisanu su: GameMaker studio 2, Unreal Engine, Godot, Unity i Firebase. Posebno će biti opisani Unity i Firebase jer su oni korišteni za implementaciju igre opisanu u ovom radu.

4.1. Razvojna okruženja za 2D igre

Za razvoj 2D mobilnih *indie* igara se najčešće koriste: GameMaker studio 2, Unreal Engine, Godot i Unity. Svi od navedenih nude tehnologije za uređivanje grafike, za pravljenje animacija i uređivanje fizike igre.

GameMaker studio 2 je multi platformsko i multi žanrovsko razvojno okruženje najčešće korišteno za razvoj 2D igara. Implementaciju ponašanja nudi vizualnim skriptiranjem svojim GML Vizualom (engl. *Game Maker Language Visual*, GML *Visual*) [9]. GML Vizual omogućuje programiranje ponašanja bez znanja programiranja korištenjem akcijskih blokova za sastavljanje logike igre na vizualan i intuitivan način [10] . GameMaker studio 2 je besplatan za svrhe učenja i ima veliku količinu video materijala s uputama (engl. *tutorial*) i uputa što olakšava rad . Navedeno okruženje je jako dobro za razvojne programere početnike, no za iskusnije postoje bolje opcije.

Unreal Engine je multi platformsko razvojno okruženje koje se primarno koristi za razvoj 3D igara, no može se koristiti i za razvoj 2D igara. Koristi vizualno skriptiranje bazirano na čvorovima (engl. *node based visual scripting*) i C++ za implementaciju ponašanja igre [11]. Za njega postoji velika količina videa, uputa, tečajeva za učenje, trgovina imovinom (engl. *asset*) i ima veliku zajednicu razvojnih inženjera na forumima. Uz sve navedeno, iako se može koristiti za razvoj 2D igara, primarno je namijenjen za razvoj 3D igara i ne nudi najbolje mogućnosti za 2D igre u usporedbi s konkurentima.

Godot je multi platformsko razvojno okruženje otvorenog koda koje se koristi za razvoj 2D i 3D igara. Daje mogućnost implementacije ponašanja u C#, C++, GD Skriptu i vizualnim skriptiranjem. GD skript je jezik sličan Phytonu napravljen za Godot. Vizualno skriptiranje u Godotu je namijenjeno ljudima koji ne znaju programirati kako bi ga postepeno naučili [12]. Postoji online priručnik za korištenje nazvan Godot Docs i jako dobra zajednica developera koja radi na Godotu i sa Godotom.

Svi navedeni alati za razvoj igara nude izvrsne mogućnosti razvoja mobilnih 2D igara. Alat koji će se koristiti za izradu igre je Unity razvojno okruženje. Zašto je baš Unity odabran, po čemu se razlikuje od ostalih alata i zašto je najpovoljniji za korištenje će se objasniti u sljedećem poglavlju

4.1.1. Unity

Unity je prvobitno napravljen kao shader/kompajler i tek je kasnije postao razvojno okruženje koje je službeno objavljeno od strane tvrtke Unity Technologies. Unity Technologies su prvu verziju objavili 2005. godine na Appleovoj konferenciji developera. Danas nakon puno verzija i poboljšanja se smatra jednim od najboljih alata za izradu igara bez većih mana, koji uvelike olakšava implementaciju ideja na brz i kvalitetan način. Omogućuje razvoj raznih vrsta igara: 2D, 2.5D, 3D, VR (engl. *virtual reality*, virtualna stvarnost), AR (engl. *augmented reality*, proširena stvarnost) itd. Također se koristi u inženjerstvu za simulaciju sustava, arhitekturi za izradu modela i u filmskoj industriji za izradu animacija [3].

Unity nudi integrirane alate za razvoj već ugrađene u okruženje. Također nudi razne mogućnosti poput: pristupa assetima, skriptama, vizualno skriptiranje, monetizaciju, analitiku itd. bez ikakve potrebe za vanjskim softverima. Unity je jako dobro dokumentiran, ima jako velik broj sudionika na Unity forumima gdje se uvijek može potražiti pomoć jer je velika mogućnost da se neko već susreo sa identičnim problemom, također ima Unity Asset Store [13] gdje se može kupiti razna imovina koja olakšava implementaciju igre i gdje također postoji veliki broj imovine dostupan besplatno na korištenje u projektima.

Ponašanje igre u Unityju se implementira u jeziku C# korištenjem programskog okruženja Microsoft Visual Studio. Unityjev API za skriptiranje je jako dobro dokumentiran i jako čitak, također je vrlo jednostavan za koristiti i savladati.

4.2. Firebase

Firebase je platforma za razvoj aplikacija i igara koju je razvila tvrtka Google. Firebase je BaaS (engl. *Backend-as-a-Service*) tj. nudi cijelu pozadinsku (engl. *backend*) infrastrukturu i jednostavno sučelje za korištenje usluga kako ne bi bilo potrebno modelirati i implementirati bazu podataka na posebnom poslužitelju [14]. Firebase nudi mnoge usluge od kojih će se koristiti Firebase autentifikacija i Firebase baza podataka u stvarnom vremenu (engl. *Realtime database*).

Firebase nudi NoSQL bazu podataka gdje se podatci strukturirano upisuju u dokumente koji se spremaju na poslužitelj i kasnije kad dođe do potrebe za spremljenim podatcima, poziva se na taj dokument. NoSQL znači da je to baza podataka koja sprema podatke u dokumente, u formatu sličnom JSON-u. Navedeno znači da nije potrebno razvijati posebnu bazu podataka i raspisivati SQL upite kako bi se dobili podatci iz baze nego je potrebno reći Firebase bazi podataka koje podatke iz kojeg dokumenta se želi koristiti, što Firebase omogućuje jednostavnim skupom funkcija koji se mogu naći u dokumentaciji. Firebase sadrži poprilično čitljivu dokumentaciju i rastući broj video materijala s uputama koji će olakšati implementaciju baze podataka i autentifikacije u igru. Firebase baza podataka u stvarnom vremenu daje sposobnost spremanja podataka o igraču kao što su: korisničko ime, podatci o završenim razinama, najbolji rezultat na beskonačnoj razini itd. u slučaju da igrač želi spremiti svoj napredak u igri.

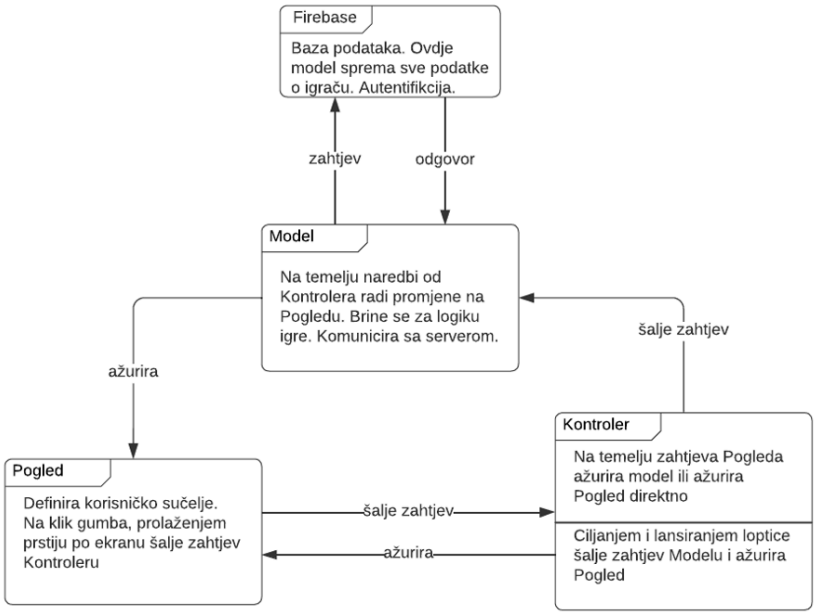
Firebase podržava autentifikacije lozinkom, telefonskim brojem, Google računom, Facebook računom itd. Firebase autentifikacija će omogućiti igraču prijavu u igru sa Google računom u slučaju da igrač želi biti prijavljen i zajedno sa bazom podataka će nakon prijave prikazivati podatke igrača u igri.

5. DIZAJN IGRE

U ovom poglavlju bit će opisan dizajn igre izrađene u okviru ovog rada. Dizajn će se definirati prateći korisničke priče napisane u poglavlju 3. Prvo će biti opisan dizajn visoke razine (engl. *High Level Design*, HLD) igre korištenjem MVC (engl. *Model-View-Controller*, Model-Pogled-Upravljač) arhitekturnog obrasca. Nadalje će biti opisan grafički dizajn igre gdje će se naglasiti koja korisnička priča je korištena za koji dio grafičkog sučelja igre.

5.1. Dizajn visoke razine

Dizajn visoke razine je generalni dizajn sustava. HLD-om treba biti prikazana generalna arhitektura sustava tj. glavni dijelovi sistema ukratko opisani i odnos među tim dijelovima tj. tok podataka kroz sustav. Za HLD igre koristit će se jedan od arhitekturnih obrazaca. Arhitekturni obrazac koji će se koristiti je MVC. MVC je arhitekturni obrazac kojim se odvaja korisničko sučelje od logike igre



Slika 5.9. MVC igre

Na slici 5.9. je prikazan MVC kojim je objašnjeno kako će igra funkcionirati tj. kako će se MVC koristiti u kontekstu igre. Igra je podijeljena na Pogled, Model, Upravljač i bazu podataka. Igra će se podijeliti na takav način kako bi bilo lakše raditi moguće potrebne izmjene nakon što je igra implementirana, ovakav pristup također ubrzava proces implementacije te pomaže pri organizaciji projekta. Dalje će detaljnije biti objašnjeno koja je funkcija svakog dijela.

Pogled predstavlja korisničko sučelje igre. Korisničko sučelje se može podijeliti na dva dijela: izbornici na kojima se nalaze uglavnom gumbi i kontrola igre (npr. touch, tipkovnica, džojstik). Izrada prvog djela je vrlo jednostavna u Unityju. Stvaranje UI objekata u Unityju je vrlo jednostavno, stvore se 2D objekti kojima se određuju varijable kao što su: visina, širina i boja. Svakom UI objektu u Unityju se može dodati gumb komponenta koja sadrži funkciju koja se poziva kad se klikne na objekt (engl. *on click function*, funkcija na klik), u funkciji na klik u Unityju se može referencirati bilo koja funkcija sa bilo kojeg objekta koja će se onda prilikom klika na UI element izvršiti. Od ovog svega jedino što je potrebno implementirati je funkcija referencirana u funkciji na klik, sve ostalo se postavlja u Unityjevom sučelju. Tako da će se Pogled u kontekstu igre samo brinuti o definiranju funkcija koje se referenciraju u funkciji na klik, a te funkcije će definirati koji UI dijelovi će biti prikazani na zaslonu uređaja. Za drugi dio korisničkog sučelja tj. za kontrolu igre će biti direktno odgovoran Upravljač.

Upravljač u igri omogućava razdvajanje korisničkog sučelja i logike igre. Većina interakcije Pogleda i Modela će ići preko Upravljača.. Rečeno je da će kontrola igre tj. način na koji se igra samu igru ići direktno od Upravljača prema Modelu. To se radi na takav način kako bi u slučaju da je potrebno promijeniti na koji način se loptica kontrolira (npr. ne želi se više koristiti dodir zaslona nego džojstik) bilo lakše napraviti takvu izmjenu tj. kako ne bi trebalo raditi veće komplikacije i promjene u programskom kodu aplikacije, nego samo promjenu tog jednog dijela.

Model će na temelju zahtjeva od Upravljača ažurirati Pogled. Model se također brine za ponašanje igre. O svim događajima u igri npr. sudaranje lopte sa preprekama, ulazak lopte u rupu, brojanje preostalih udaraca, brojanje rezultata, spremanje rezultata u bazu, spremanje razine do koje je igrač došao u bazu podataka se brine Model.

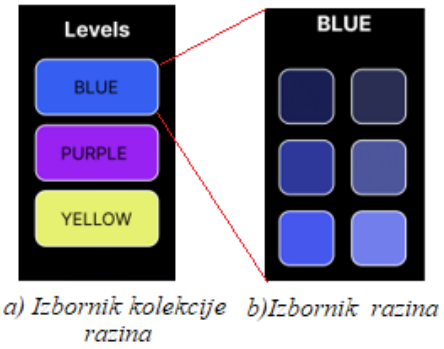
Dio dijagrama koji nije MVC na slici 5.7. je Firebase baza podataka. Nije potrebno opisivati bazu podataka detaljno jer je Firebase baza podataka oblik usluge u kojoj nije potrebno brinuti se o strukturi podataka nego se programskim kodom strukturirano spremaju podatci, kojima se kasnije pod određenim uvjetima pristupa.

5.2. Grafički dizajn igre

Za izradu grafičkog dizajna se koristila figma. Figma je alat koji služi za izradu prototipa grafičkog sučelja web ili mobilne aplikacije, odnosno igre u ovom slučaju [15]. Grafičko sučelje slično onome prikazanom u ovom poglavlju će biti u finalnom proizvodu, te će znatno olakšati izradu grafičkih elemenata igre pri implementaciji igre.

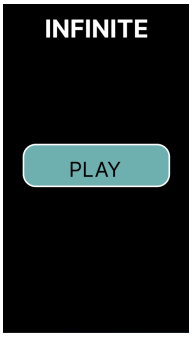
Slika 5.1. Glavni izbornik igre

Na slici 5.1. prikazan je potencijalni izgled glavnog izbornika igre. Izgled je rađen po epu iz tablice 3.6. gdje piše da pozadina izbornika treba biti crne boje i da svaki gumb treba biti drugačije kremaste boje.



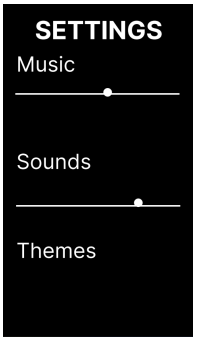
Slika 5.2. Izbornici razina

Na slici 5.2. su prikazani potencijalni izgledi izbornika razina. Slika a) prikazuje izbornik kolekcije razina a slika b) prikazuje izgled izbornika razina odabrane kolekcije. Izgled je rađen po epovima iz tablica 3.3. i 3.6. gdje je napisano da razine istog tipa budu grupirane u kolekcije. Klikom na gumb „*Blue*“ izbornika na slici a) otvara se izbornik sa slike b). Pozadina mora biti crne boje kao i kod svih ostalih zaslona.



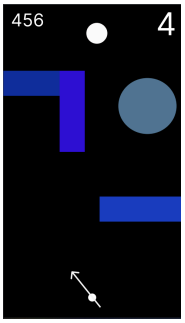
Slika 5.3. Pokretač beskonačne razine

Na slici 5.3. je prikazan potencijalni izgled izbornika koji će pokretati beskonačnu razinu. Izgled je rađen po epu iz tablice 3.6. i priči 5 iz tablice 3.4. u kojima je napisano da pozadina treba biti crna, gumb obojen i da se klikom na gumb „*Infinite*“ koji je prikazan na slici 5.1. otvara grafičko sučelje prikazano na slici 5.3.



Slika 5.4. Postavke

Na slici 5.4. je prikazan potencijalni izgled postavki igre. Izgled je rađen po priči 1 iz tablice 3.6 i priči 7 iz tablice 3.7. gdje piše da pozadina mora biti crna i da igrač želi imati mogućnost otvaranja postavki gdje će moći mijenjati različite stavke igre.

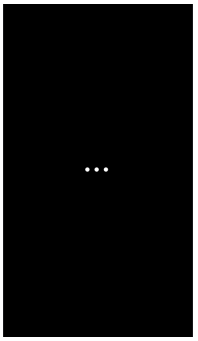


Slika 5.5. Igra

Na slici 5.5. je prikazan potencijalni izgled igre opisan epovima iz tablica 3.1. i 3.6. U epu iz tablice 3.1. su navedeni osnovni elementi igre a to su: loptica, rupa i prepreke. Iz epa 3.6. uzet je raspored boja tj. crna pozadina, bijela loptica i prepreke raznih boja.

Slika 5.6. Izbornik nakon završetka razine

Slika 5.6. prikazuje izbornik nakon završetka razine opisan pričom 7 iz tablice 3.1. i pričom 9 iz tablice 3.4. kojima je napisano da nakon završetka igre se mora pojaviti izbornik koji nudi mogućnost ponovnog igranja ili vraćanja na izbornik.



Slika 5.7. Zaslon očitavanja

Slika 5.7. prikazuje zaslon učitavanja opisan pričom 2 iz tablice 3.7. gdje piše da zaslon za učitavanje mora prikazivati nekakvu animaciju. Animacija na slici 5.7. bi bila valno pomicanje točkica gore dolje.

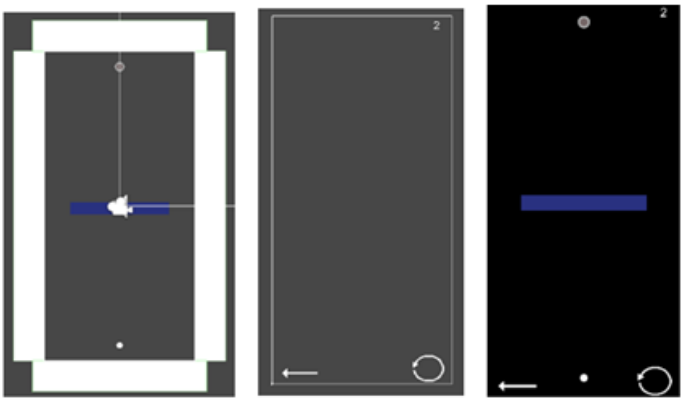
6. IMPLEMENTACIJA IGRE

U ovom poglavlju biti će opisana implementacija igre opisane ovim radom. Za svaki element igre će biti objašnjeno kako je izveden. Igra se implementirala po korisničkim pričama iz poglavlja 3. S obzirom na prirodu Unity razvojnog okruženja implementacija igre u ovom radu može se podijeliti na sljedeće dijelove: mehanika igre, grafičko korisničko sučelje i glazba i zvučni efekti.

Svaki objekt u Unityu se sastoji od najmanje jedne komponente. Komponente su kao unaprijed programirane skripte od Unitya koje se mogu postaviti na objekt unutar Unity editora. Nakon što su postavljene objekt će dobiti određena svojstva komponente koja je postavljena na njega. Svaki objekt sadrži Preobrazba (engl. *Transform)* komponentu koja sadrži podatke o poziciji, rotaciji i dimenzijama objekta. C# skripta je komponenta u kojoj programer može sam isprogramirati željeno ponašanje određenog objekta. Svaka C# skripta je klasa koja nasljeđuje Unityjevu MonoPonašanje (engl. *MonoBehaviour*) klasu koja sadrži Ažuriraj (engl. *Update*) funkciju koja se poziva jednom po okviru i Počni (engl. *Start*) funkciju koja se poziva jednom na početku scene. Neke od komponenti koje je bitno spomenuti su: Prikazivač Slike (engl. *Sprite Renderer*), Izvor Zvuka (engl. *Audio Source*), 2D sudarač (engl. *2D Colider*) i Gumb (engl. *Button*). Prikazivač Slike je komponenta koja prikazuje objekt na zaslonu uglavnom na temelju slike koje mu je dana. Izvor Zvuka je komponenta koja služi kao izvor glazbe u pozadini scene. 2D Sudarač je komponenta koja detektira kad neki drugi objekt dodiruje objekt na kojeg je komponenta postavljena. Gumb je komponenta koju se može postaviti na UI elemente. Klikom na neki UI element koji ima Gumb komponentu bi se pokrenula funkciju koja je prethodno definirana u nekoj skripti. Ta prethodno definirana funkcija kako bi sve funkcioniralo ispravno treba biti referencirana u Gumb komponenti. UI objekt na kojemu su zakačeni i na kojem se prikazuju svi ostali UI elementi zove se Platno (engl. *Canvas)*. Glavna Kamera (engl. *Main Camera*) je objekt kojeg svaka scena ima i koji projicira sve što se događa u sceni na zaslon.

Igre u Unityju se sastoje od scena. Jedna scena predstavlja jedan zaslon ili razinu igre na slici 6.1. pod a) vidimo isječak iz scene. U scenama instanciramo 2D ili 3D objekte koji međusobno djeluju na slici 6.1. pod a) vidimo objekte igre a to su: zidovi, loptica, rupa, prepreka i Glavna Kamera. Međusobno djelovanje objekata je definirano C# skriptama i gore navedenim komponentama. U slučaju ove igre napravljena je skripta koja sa 2D sudarač komponentom rupe detektira kad ju 2D sudarač lopte dodiruje. Nakon što je dodir detektirana pozvati će se funkcija koja na zaslonu prikazuje „You Won“. Objekt Platno se uvijek nalazi između objekta Glavna

Kamera i objekata igre što se vidi na slici 6.1. Pod c) se vidi gledište kamere, pod b) Platno, a pod a) objekti igre. Na objekt Platno se stavljaju svi UI objekti (gumbi, tekst, slike, klizači itd.). Ako je potrebno kontrolirati korisničko sučelje mora se u C# skripti definirati što se događa kad se djeluje na UI objekte korisničkog sučelja. Igra se radila na takav način da se prvo implementiralo ponašanje tj. mehanika jedne razine i nakon toga su se dodali UI elementi kojima se prelazi iz scene u scenu i koji prikazuju broj preostalih udaraca lopte kao što se vidi na slici 6.1.



Slika 6.1. Elementi igre

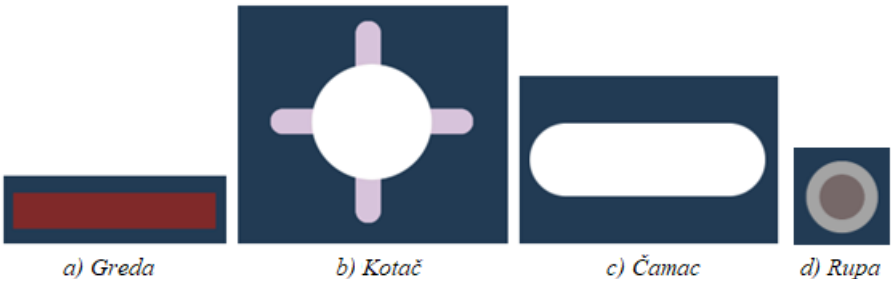
6.1. Mehanika igre

Kao što je gore navedeno mehanika igre se sastoji od objekata koji međusobno djeluju i skripte kojom je propisano međusobno djelovanje i kontroliranje igre. Iz Epa 1 iz tablice 3.1. i Epa 4 iz tablice 3.4. vide se objekti koje je potrebno imati u igri. Objekti su lopta, prepreke, rupa i zidovi koji čine polje igre (Glavna Kamera je izostavljena jer je nevidljiva igraču).

Lopta koja se vidi na slici 6.2. se može smatrati igračem jer se nju pomiče kroz teren. Kako bi se slijedio MVC obrazac iz poglavlja 5, kontrola igrača definirana je apstraktnim razredom UpravljačLopte (engl. *BallControler*) koja sadrži virtualnu funkciju Pucaj (engl. *Shoot*). Funkciju Pucaj potrebno je pri kodiranju konkretnog upravljača potrebno prepisati tj. definirati na koji način će se lopta kontrolirati. U slučaju ove igre bilo je potrebno isprogramirati kontrole za zaslon na dodir. Na slici 6.2. vidi se ciljanje lopte dodirom na zaslon i povlačenjem prsta prema dolje. Micanjem prsta po zaslonu mijenja se smjer i dužina bijele trake vidljive na slici 6.2. Bijela traka je indikator smjera i snage kretanja lopte nakon što se podigne prst sa zaslona.

Slika 6.2. Ciljanje lopte

U igri postoje dinamičke i statičke prepreke. Kako bi se olakšalo dodavanje prepreka sa raznim ponašanjima, napravljena je apstraktna klasa Prepreka (engl. *Obstacle*) koja sadrži virtualnu funkciju PomakniSe (engl. *Move*). Funkciju PomakniSe svaki konkretni razred treba prepisati i u njoj definirati svoj način kretanja. To je rađeno na takav način kako bi se sve vrste prepreka mogle držati u jednoj listi u Modelu i pozivati njihove PomakniSe funkcije grupno tj. u jednoj petlji. U igri se koriste tri različite vrste prepreka: greda, čamac i kotač. Greda na slici 6.3. pod a) je stacionarna prepreka kojoj se određuju dimenzije pri dizajniranju razine. Čamac na slici 6.3. pod b) je prepreka koja je isprogramirana na takav način da se kreće s jedne strane zaslona na drugu. Kotač na slici 6.3. pod c) je prepreka koje se okreće oko svoje osi.



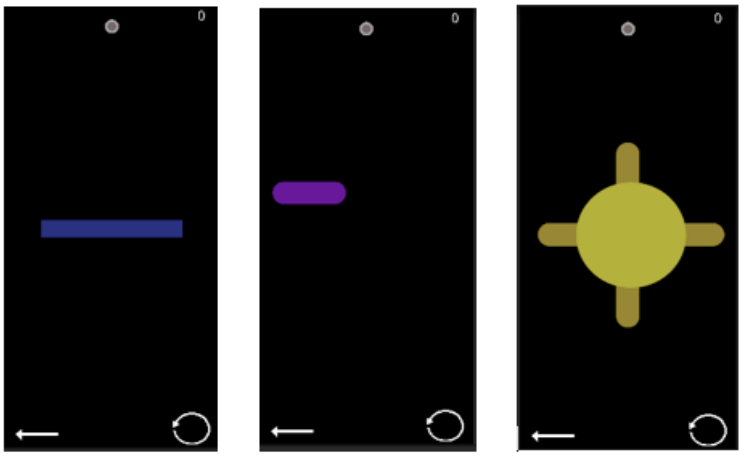
Slika 6.3. Objekti igre

Zidovi su elementi igre koji se ne vide ne zaslonu jer je skriptom određeno da će se s obzirom na veličinu zaslona pomaknuti na rubove tako da budu nevidljivi i da igra isto izgleda na svakom uređaju tj. da se rubovi zidova ne vide na zaslonu ali da se i dalje može loptica odbijati od njih. Prikaz kako zidovi izgledaju u Unityju nalazi se na slikama 6.1. i 6.3. Zadnji objekt je rupa prikazana na slici 6.3. pod d) , to je objekt koji detektira kad ga loptica dodirne preko komponente 2D Sudarač i poziva funkciju koja prikazuje na zaslonu tekst „You Won“.

6.1.1. Normalne razine

Svaka obična razina sadrži zidove, loptu, prepreke i rupu. Model skripta je postavljena na objekt Glavna Kamera, Pogled skripta je postavljena na UI objekt Platno a Upravljač skripta na loptu (igrača). Upravljač, Pogled, Rupa i sve prepreke su referencirane u Modelu. Model određuje kako će se igra ponašati koristeći navedene objekte.

U Modelu se pokreću prepreke pozivanjem funkcije PomakniSe prepreka koje su u sceni. Upravljač je instanciran u Modelu i pozivanjem njegovih funkcija se pokreće ciljanje i lansiranje lopte. Model također sadrži podatke o broju dozvoljenih pokušaja i u slučaju da broj pokušaja padne na nulu zaustavlja igru. Broj pokušaja se prikazuje na zaslonu tako što Model šalje broj pokušaja koji treba prikazati Pogledu. U slučaju da igrač postigne pogodak, na zaslonu se prikazuje „You Won“. Model se također brine za glazbu i zvučne efekte, no to će se objasniti u poglavlju 6.3. Na slici 6.4. vide se razine sa gore navedenim preprekama. Vidi se da su prepreke drugačijih boja nego na slikama gore radi estetike igre.

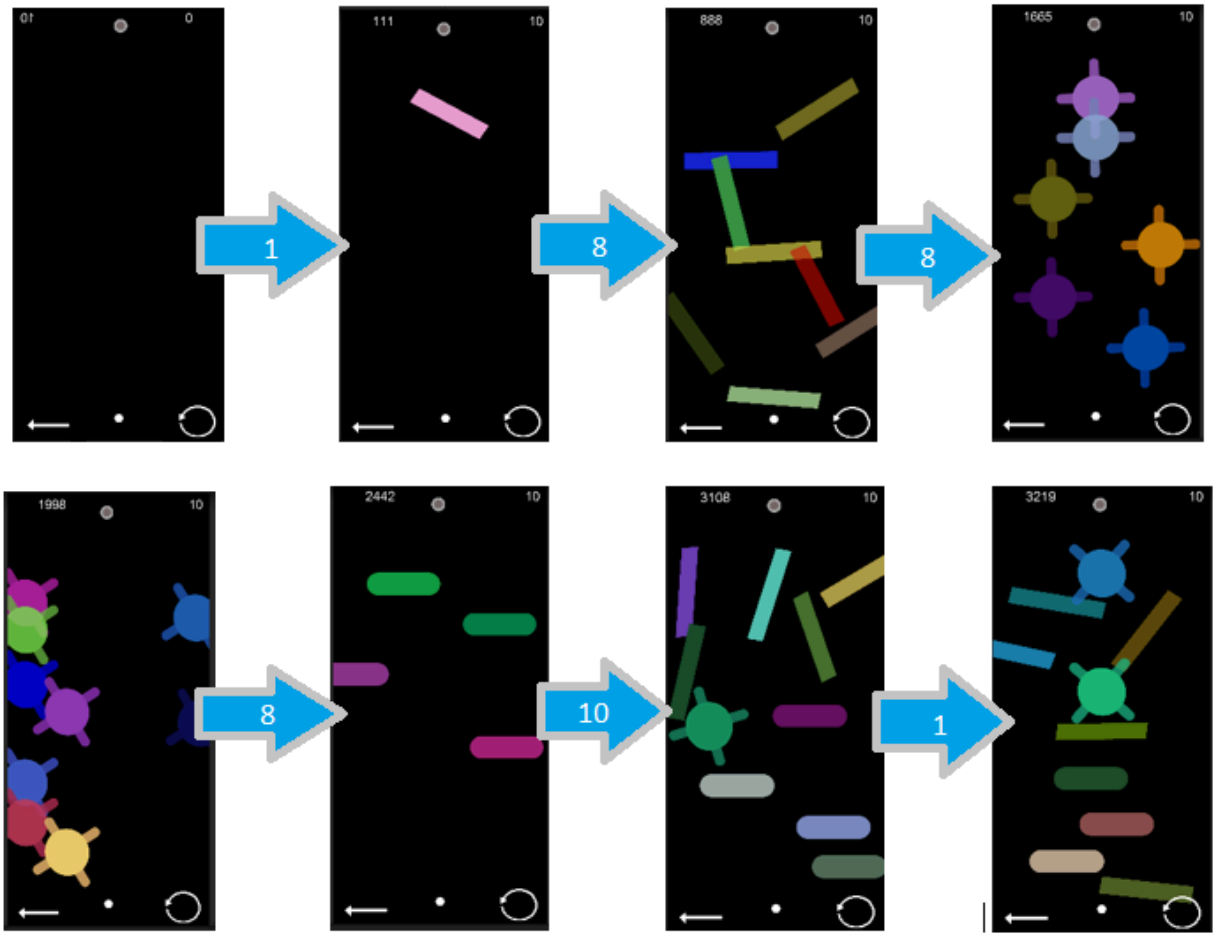


a) Razina sa gredom *b) Razina sa čamcem* *c) Razina sa kotačem*

Slika 6.4. Razine sa tipovima prepreka

6.1.2. Beskonačna razina

Beskonačna razina je razina gdje se nakon svakog pogotka rupe automatski generira malo teža razina. Kod beskonačne razine su prikazana dva broja na vrhu a to su broj bodova koji je postignut i broj preostalih udaraca. U beskonačnoj razini ako broj udaraca dođe do nula igra se završava. Beskonačno generiranje se postiže postavljanjem određenih prepreka u scenu na pseudo nasumične pozicije zaslona. Skripta Beskonačni generator (engl. *Infinite generator*) sadrži funkciju koja u scenu instancira prepreku na pseudo nasumičnu poziciju. Pomoću te funkcije u Modelu beskonačne razine razvijen je algoritam koji instancira prepreke u scenu. Što se više pogodaka postigne, to se više prepreka instancira. Na slici 6.5. vidi se da se prvo pojavljuju samo grede pa se, nakon što je postignuto nekoliko pogodaka, pojavljuju samo kotači pa samo čamci i na kraju mješavina raznih prepreka. Algoritam za generiranje prepreka je jednostavan za izmijeniti i potrebna je znatna količina testiranja da se usavrši, no postignuto je generiranje opisano korisničkim pričama i algoritam funkcionira dovoljno dobro da igru čini zabavnom za igranje. Na slici 6.5. vidi se napredak kroz razine. Brojevi u strelicama predstavljaju koliko je pogodaka postignuto od prethodne slike.



Slika 6.5. Prolazak kroz beskonačnu razinu

6.2. Grafičko korisničko sučelje

UI objekti koji su u sceni potpuno su odvojeni i nevezani za objekte igre, osim ako se drugačije ne odredi kroz C# skriptu. Može se reći da se grafičko sučelje prikazuje preko scene. Grafičko sučelje koje je implementirano je poprilično slično dizajnu koji je prikazan u poglavlju 5.2., napravljene su manje izmjene i prilagodbe.

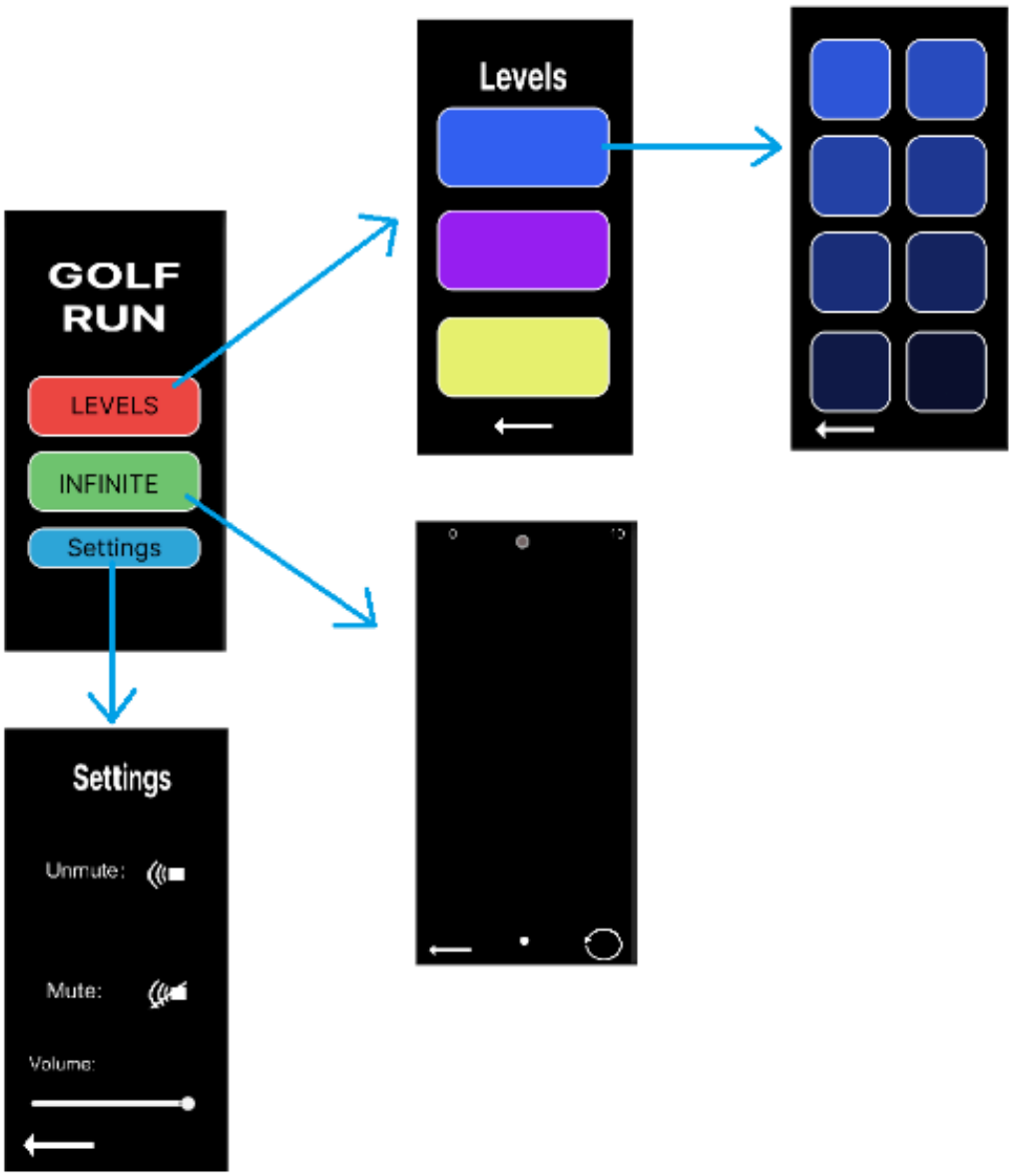
6.2.1. Korisničko sučelje na razinama

Grafičko sučelje sa slike 6.6. je na svakoj od razina i sadrži sljedeće elemente: prikaz preostalih udaraca, gumb za nazad, gumb za ponovno počinjanje razine i u slučaju da se ostvari pogodak nam se prikazuje zaslon preko razine koji govori „You Won“ i gumb koji vodi na sljedeću razinu

Slika 6.6. „You Won“ zaslon

6.2.2. Navigacija do razina i postavke

Na slici 6.7. se vidi slijed navigacije kroz korisničko sučelje igre. Kad se otvori igra prikaže se scena koja prikazuje naslov i gumbe koji vode na ostale scene tj. dijelove korisničkog sučelja. Gumb Razine (engl. *Levels*) vodi na scenu sa kolekcijama razina. Kolekcije razina je podijeljena po bojama. Plavi gumb otvara izbornik sa plavim razinama koje sadrže kao prepreke samo grede. Žuti gumb otvara izbornik sa žutim razinama koje sadrže prepreke samo sa čamcima. Ljubičasti gumb otvara izbornik s ljubičastim razinama koje sadržavaju samo kotače kao prepreke. Gumb Beskonačna (engl. *Infinite*) vodi na scenu na kojoj se odmah može krenuti sa igranjem beskonačne razine. Gumb Postavke (engl. *Settings*) vodi na postavke glazbe i glasnoće.



Slika 6.7. Navigacija kroz korisničko sučelje igre

6.2.3. Razlike između prvotnog i krajnjeg grafičkog dizajna

Pri definiranju grafičkog dizajna određeno je da će taj dizajn služiti kao predložak te da će krajnji proizvod vjerojatno biti nešto drugačiji. U ovom poglavlju će se proći kroz razlike prvotnog i krajnjeg dizajna.

Kod grafičkog dizajna Glavnog izbornika izbačen je gumb Statistike (engl. *Statistics*) i promijenjen je naziv igre iz 2D Golf u GOLF RUN. Ne postoji veliki broj statistika za pokazati, a s obzirom na vrstu igre (ležerna) nije čak potrebno ni prikazivati statistike.

Izbornik kolekcije razina i izbornik razina određene kolekcije su vrlo slični prvotnom dizajnu, samo što je u implementaciji uklonjen nepotreban tekst. Pokretač beskonačne razine je uklonjen jer je potpuno nepotreban zato što stvara dodatan korak kod igranja beskonačne razine i jer stvara dodatnu scenu za implementirati.

Izbornik nakon završetka razine je potpuno promijenjen. Novi izbornik čestita igraču na pobjedi i nudi ili da nastavi na sljedeću razinu ili da se vrati na izbornik razina. Na novom dizajnu ne postoje ogromni gumbi već strelice, što čini dizajn čišćim. Također, novi izbornik je potpuno proziran što dodaje estetici igre.

Kod grafičkog sučelja same igre dodana su dva gumba u donje kutove, jedan gumb (strelica) vraća igrača nazad na izbornik a drugi gumb (zaobljena strelica) resetira razinu.

6.3. Glazba i zvučni efekti

Sviranje glazbe u sceni se postiže stvaranjem praznog objekta na kojeg se postavi komponenta Izvor Glazbe u kojoj se referencira odabrana audio datoteka te je na komponenti potrebno podesiti postavku „Play On Awake“ koja počinje svirati glazbu pri očitavanju scene i postavku „Loop“ koja glazbu svira u petlji, bez prestanka. Kod puštanja zvuka u igri nastaje problem ako postoji instanca objekta koji pušta zvuk u prvoj sceni a potrebno je učitati drugu scenu. Pri učitavanju nove scene svi objekti se uništavaju u svrhu efikasnijeg trošenja memorije i glazba u drugoj sceni prestaje. Iz tog je razloga napravljena skripta koja naređuje sustavu da objekt koji pušta glazbu ne uništi pri očitavanju ostalih scena.

Zvučni efekti se namještaju tako što se komponenta Izvor Glazbe prikvači na objekte igre i UI objekte. Napravljen je skripta Puštač zvuka (engl. *Sound player*) koja se koristi u Modelu kako bi se pustio zvuk jednom svaki put kad loptica dodirne neku prepreku zid ili rupu. A kod UI elemenata podesi se Gumb komponenta da svaki put kad se dodirne gumb uz to što promjeni scenu, pusti i zadani zvuk

6.4. Korisničke priče koje nisu implementirane

Ep 1 Priča 2: „Kao igrač želim da se teren može povećavati i smanjivati potezima prsta po zaslonu.“ nije implementirana jer tereni koji su dizajnirani kao dio igre nisu toliko veliki da je potrebno uvećavati i smanjivati zaslon kako bi vidjeli što se događa

Ep 2 Priča 4: „Kao igrač želim da glavni izbornik ima gumb koji otvara prikaz mojih podataka i rezultata“ nije implementirana jer ne postoji dovoljna količina podataka i rezultata za prikazati.

Mogao bi se prikazivati najveći postignuti broj bodova u beskonačnoj razini, no postoje druga mjesta gdje se takav broj može prikazati i samo to nije dovoljno za dodatnu scenu.

Ep 4 Priča 5: „Kao igrač želim da se pri kliku gumba sa glavnog izbornika koji vodi na beskonačnu razinu se pokaže izbornik sa gumbom koji pokreće generiranje razine“ u poglavlju 6.2.3. je objašnjen razlog zašto navedena priča nije implementirana.

Ep 7 Priča 1: „Kao igrač želim imati mogućnost autentifikacije mailom tako da mi napredak u igri ostane spremljen tj. ako izbrišem igru pa je ponovo instaliram da se napredak ne izbriše.“ Jednostavno ne postoji dovoljno podataka koji bi se spremali u bazu podataka kako bi se implementacija autentifikacije Gmailom isplatila.

7. ZAKLJUČAK

Kroz ovaj rad je pokazano kako kombinacijom različitih postojećih koncepata napraviti novu i pomalo drugačiju ležernu igru koristeći Unity razvojno okruženje. Dvodimenzionalna šarolika igra s beskonačnom razinom na način da nakon svakog pogotka se generira nova malo teža razina je igra čiji klon ili barem nešto slično njoj je teško naći na internetu. Ovim radom je prikazana formula za stvaranje novih i zanimljivih igara tj. prikazano je kako kombiniranjem već postojećih igara na neobičan način i sa malo kreativnosti rezultira unikatnom igrom koja nudi novo iskustvo igranja.

U radu su korištene već postojeće tehnologije koje olakšavaju realizaciju ideja, kao što su korisničke priče kojima se točno odredilo kako se igra mora ponašati, kako bi trebala izgledati i kakav osjećaj bi trebao imati korisnik dok igra igru i time je olakšan proces dizajniranja kako grafičkog dizajna tako i dizajna visoke razine. Pri dizajniranju dizajna visoke razine je bio fokus na tome da igra bude laka za nadogradnju što se implementacijom MVC obrasca i postiglo. Olakšalo se dodavanje novih razina, novih vrsta prepreka i modifikacija algoritma koji je odgovoran za pseudo nasumično, beskonačno generiranje razine. Također je pokazano koliko je korisno razviti grafički dizajn koji će se pratiti i nadograđivati tijekom implementacije.

Pokazano je kako ideju razraditi i bolje definirati. Prvotno, potrebno je potražiti inspiraciju u sličnim uradcima. Potrebno je točno odrediti što ideja je i što je krajnji cilj. Nakon toga slijedi dizajniranje kako bi se postigla čišća slika krajnjeg izgleda. Na kraju ostaje implementacija kroz koju blisko prateći dizajn se riješi nepotrebnih stvar i dobije proizvod gotovo spreman za tržište.

Igra koje je razvijena ima neke manje nedostatke što se tiče učinkovitosti. Također, nisu sve korisničke priče implementirane zbog kratkog vremenskog okvira dostupnog za implementaciju i složenosti iste. Bitnije priče koje nisu implementirane, a bilo bi ih dobro implementirati su: mogućnost mijenjanja izgleda loptice i boja grafičkog dizajna kako korisnik želi i sustav za nagrađivanje igrača. Cilj je također bio da igra koristi bazu podataka, no trenutno je obujam podataka premalen da bi koji bi se spremali te bilo potrebno implementirati bazu podataka. Moguće je da će pri daljnjem razvoju nastati potreba da se u igru implementira i baza podataka.

LITERATURA

[1] J, Wise, How much is the Gaming Industry worth in 2022?, Earthweb, 2022. [online], dostupno na: https://earthweb.com/how-much-is-the-gaming-industry-worth/ [14. 7. 2021]

[2] I, Winter, What is a casual game?, informa, 5 Howick Pl, London SW1P 1WG, Ujedinjeno Kraljevstvo, 2011. [online], dostupno na: https://www.gamedeveloper.com/design/what-is-a-casual-game [14. 7. 2022.]

[3] Unity, Unity Technologies, [online], dosutpno na: https://unity.com/ [14.7.2021]

[4] Google Play: Mini Golf King [online], dostupno na:

https://play.google.com/store/apps/details?id=com.pnixgames.minigolfking&hl=hr&gl=US [16. 5. 2022.]

[5] Google Play: Vista Golf, [online], dostupno na:

https://play.google.com/store/apps/details?id=com.sg.vistagolf&hl=hr&gl=US [16. 5. 2022.]

[6] Google Play: Mini Golf 3D Multiplayer Rival, [online], dostupno na:

https://play.google.com/store/apps/details?id=com.MobileSportsTime.MiniGolfEldoradoGolf [16. 5. 2022.]

[7] Google Play: Color Switch, [online], dostupno na:

https://play.google.com/store/apps/details?id=com.colorswitch.switch2&hl=en_US&gl=US [17. 5. 2022]

[8] Google Play: i love hue too, [online], dostupno na:

https://play.google.com/store/search?q=i%20love%20hue%20too&c=apps&hl=en_US&gl=US [17. 5. 2022]

[9] GameMaker studio 2,YoYo Games, [online], dostupno na:

https://gamemaker.io/en/gamemaker [25. 5. 2022.]

[10] GameMaker Manual YoYo Games , [online], dostupno na:

https://manual.yoyogames.com/Drag_And_Drop/Drag_And_Drop_Index.htm [25. 5. 2022.]

[11] Unreal Engine, Epic Games, [online], dostupno na: https://www.unrealengine.com/en-US [25. 5. 2022.]

[12] Godot Engine, [online], dostupno na: https://godotengine.org/ [25. 5. 2022.]

[13] Unity Asset Store, [online], dostupno na: https://assetstore.unity.com/ [25. 5. 2022.]

[14] Firebase, Google, [online], dostupno na: https://firebase.google.com/ [25. 5. 2022.]

[15] Figma, [online], dostupno na: https://www.figma.com/ [29. 5. 2022.]

SAŽETAK

U ovom završnom radu opisuje se postupak izrade 2D *indie* ležerne igre pomoću Unity razvojnog okruženja. U radu se prikazalo kako se kombinacijom već postojećih koncepata računalnih igara može postići nova igra zanimljiva igraču. To je postignuto kombinacijom 2D prikaza, šarolikosti i jednostavnosti modernih ležernih igara s već dugo postojanim konceptom mini golf igara. Zahtjevi na navedenu igru predstavljeni su korisničkim pričama te je na temelju njih razvijen dizajn visoke razine i grafički dizajn igre. Na temelju navedenog dizajna igra je implementirana pomoću Unity razvojnog okruženja te je tako dobiven radni prototip na temelju kojeg se vidi budući potencijal igre.

Ključne riječi: indie igra, ležerna igra, mini golf, Unity, 2D mobilna igra

ABSTRATCT

Development of a mobile game in Unity environment

This thesis paper demonstrates how to create a casual 2D *indie* game using the Unity development environment. This study demonstrates how an unique and engaging game may be made using preexisting computer game concepts. The 2D depiction, color, and simplicity of contemporary casual games were combined with the preexisting notions of minigolf games to create a novel idea. User stories were used to explain the game's requirements, and a high-level design was created based on those user stories. The game was developed using the Unity development environment based on the above mentioned design, and as a result, a functioning prototype was achieved on the basis of which the future potential of the game is easily visible.

Keywords: casual game, indie game, mini golf, Unity, 2D mobile game