

Elemental Clash

Kortspel på pekskärm med realtidsavläsning och
-interaktion med fysiska kort.

Cardgame on a touchscreen with realtime
scanning and interaction of physical cards.

Emil Larsgärde
Ludwig Boge
Jonatan Ebenholm
Gayathri Naranath
Gustaf Kronholm
Armen Abedi
Mirijam Björn

Examinator: Daniel Jönsson

Sammanfattning

Projektet ”Elemental Clash” syftar till att undersöka användningen av fysiska kort i ett digitalt kortspel som spelas på en stor pekskärm. Detta projekt genomfördes av en grupp studenter vid Linköpings universitet. Projektets mål var att undersöka hur fysiska kort påverkar spelets tempo och användarens upplevda stressnivå i jämförelse med digitala kort. Spelet, som utvecklades i *Unity*, använder både fysiska och digitala kort för att skapa monster och *spells*, vilket ger en unik blandning av fysisk och digital interaktion. Temat för spelet bestämdes att vara elementära varelser som slåss mot varandra liknande *Clash Royale*. Designen av spelets grafik inspirerades av en tecknad stil.

Frågeställningarna för projektet fokuserar på hur spelets design och tekniska lösningar påverkar spelets tempo och därmed stressnivån hos spelaren, samt hur användarupplevelsen skiljer sig när fysiska kort används jämfört med digitala. Avgränsningarna specificerade att spelet skulle optimeras för att spelas på en 50-tums pekskärm och med en Logitech C930e webbkamera, som var tillgängliga på Visualiseringscenter C i Norrköping.

De fysiska korten designades med ArUco-markörer på baksidan och relevant spelinformation på framsidan. Bildbehandlingbiblioteket *OpenCV* användes för att läsa av markörerna från kortens baksida och hämta kortets position på spelplanen. Detta gjordes med hjälp av en kamera placerad rakt ovanför pekskärmen som korten placerades på. De digitala korten implementerades parallellt för att möjliggöra jämförelse med de fysiska korten.

Spelets logik och mekanismer, såsom karaktärernas rörelser, stridssystem och användning av mana, implementerades med hjälp av Unitys AI Navigation-paket och andra inbyggda funktioner. En central del av spelets design var att skapa en balans mellan snabbhet och strategi, där spelarna behövde fatta snabba beslut samtidigt som de hanterade resurser som mana för att spela sina kort effektivt.

Användartester genomfördes med 12 deltagare som spelade spelet både med fysiska och digitala kort. Genom användartester visades att designval som till exempel flera steg för att spela ett kort, ett kostnadssystem med ”mana” och korta, tidsbegränsade spelrundor ökade tempot och stressnivån. Pekskrmen och de fysiska korten bidrog till en högre stressnivå på grund av fler nödvändiga steg för att göra ett drag. Användarna upplevde att trots att de digitala korten var enklare och snabbare att spela med, så gav de fysiska korten en bättre användarupplevelse då den ökade stressnivån gjorde spelet roligare.

Innehåll

Sammanfattning	i
Figurer	v
Tabeller	1
1 Introduktion	1
1.1 Bakgrund	1
1.2 Syfte	1
1.3 Frågeställningar	1
1.4 Avgränsningar	2
2 Relaterat arbete	3
2.1 Bildigenkänning	3
2.2 Stress kopplat till beslutstagande	4
2.3 Stress i kortspel	4
3 Implementation och metod	5
3.1 Unity	5
3.1.1 Animation	5
3.2 OpenCV	6
3.3 Kort	6
3.3.1 Fysiska kort	6
3.3.2 Digitala kort	8
3.4 Speldesign	8
3.5 Grafik	9
3.5.1 3D-modellering	9
3.5.2 Design	10
3.5.3 Val av kameraperspektiv	12
3.6 Spellogik	12
3.6.1 Databas	13

3.6.2	Rörelse	14
3.6.3	Combat	14
3.6.4	Spawner	15
3.6.5	Mana	15
3.6.6	Spells	16
3.6.7	Runor	16
3.7	Ljud	17
3.8	Användargränssnitt	18
3.9	Användartester	18
4	Resultat	20
4.1	Spel	20
4.2	Användartester	23
5	Analys och diskussion	25
5.1	Implementation och metod	25
5.2	Resultat	26
5.2.1	Spel	26
5.2.2	Användartester	26
5.3	Källkritik	27
5.4	Etisk och samhällelig reflektion	27
6	Slutsatser	28
6.1	Frågeställningar	28
6.2	Vidareutveckling av systemet	29
A	Reflektion över systemutvecklingsprocessen	32
A.1	Armen Abedi	33
A.2	Emil Larsgärde	33
A.3	Gayathri Naranath	33
A.4	Gustaf Kronholm	34
A.5	Jonatan Ebenholm	34
A.6	Ludwig Boge	35
A.7	Mirijam Björn	35
B	Individuella bidrag	36
B.1	Armen Abedi	36
B.2	Emil Larsgärde	36
B.3	Gayathri Naranath	36

B.4	Gustaf Kronholm	36
B.5	Jonatan Ebenholm	37
B.6	Ludwig Boge	37
B.7	Mirijam Björn	37
B.8	Karljohan Lundin Palmerius	37
B.9	Daniel Jönsson	37
C	Resultat från användartester	38
C.1	Användartest 1	38
C.2	Användartest 2	41

Figurer

2.1	Hur OpenCV identifierar ArUco-markörer [14].	3
3.1	Kameran för projektet.	7
3.2	Kalibrering i spelet.	7
3.3	De digitala korten.	8
3.5	Träden i spelet med olika material	10
3.4	Olika 3D-modeller med toon-shadern applicerad i Unity.	10
3.6	Första iden för designen av monster (<i>spirits</i>)	11
3.7	Vänster: Design för spelplan. Höger: Design för kort	11
3.8	<i>Concept art</i> för olika monster och spells	12
3.9	Samtliga monster i spelet	13
3.10	Fyra olika prototyper för kamerans perspektiv	14
3.11	Vad användare föredragit i en studie om hur ljud påverkar inlevelse [5].	17
3.12	Runor i olika tillstånd	18
4.1	Spelet under speltiden.	20
4.2	Några spelfunktioner.	21
4.3	Några spelfunktioner.	21
4.4	Startskärmen när spelet startats.	21
4.5	Menyn som kommer upp när man trycker på options.	21
4.6	Slutskärmen för när en spelare har vunnit.	22
4.7	Olika spells som har olika typer av effekter.	22
4.8	Digitala korten	22
4.9	Fysiska spelkortet.	23
C.1	1:Långsamt/Lågt tempo, 5:Snabbt/Högt tempo	38
C.2	1:Mycket lugn, 5:Mycket stressad	39
C.3	1:För kort, 5:För lång	39
C.4	1:Mycket obalanserad, 5:Bra balanserad	40
C.5	1:Mycket obalanserad, 5:Bra balanserad	40
C.6	1:Långsamt/Lågt tempo, 5:Snabbt/Högt tempo	41

C.7	1:Mycket lugn, 5:Mycket stressad	41
C.8	1:För kort, 5:För lång	42
C.9	1:Mycket obalanserad, 5:Bra balanserad	42
C.10	1:Mycket obalanserad, 5:Bra balanserad	43

Kapitel 1

Introduktion

Denna rapport innehåller den kompletta utredningen för projektet *Elemental Clash* i kursen TNM094. Följande kapitel adresserar bakgrund, syfte och centrala frågeställningar som projektet avser att utreda och besvara. Motiveringen bakom projektet är att undersöka vilken typ av påverkan fysiska kort och en pekskärm kan ha på det upplevda tempot av spelet samt undersöka hur användarupplevelsen påverkas då fysiska kort används gentemot digitala.

1.1 Bakgrund

Spelmarknaden idag är större än någonsin tidigare, och tekniken bakom spelen har blivit mer varierade. En typ av teknik som blivit allt mer tillgänglig är virtuell verklighet, där spelare kan interagera direkt med spelvärlden. Kortspel genren är dock väldigt uniform när det kommer till hur korten spelas, och därför finns det möjlighet till att innovera. Skulle fysiska spelkort kunna tillämpas i ett spel, där de blir inlästa och bidrar till spelmekaniken?

1.2 Syfte

Målet med projektet är att utreda hur fysiska kort, spelade på en stor pekskärm, kan ha en påverkan på spelets tempo och användarens upplevda stress, i jämförelse med digitala kort. Projektet ska även utreda hur användarupplevelsen påverkas av att använda de fysiska korten jämfört med digitala korten. För att utreda detta kommer ett spel att designas med ett högre tempo i åtanke, sedan kommer spelet att skapas i *Unity*.

1.3 Frågeställningar

- Hur ska spelet designas och vilka tekniska lösningar ska användas för att öka tempot i spelet och därmed stressnivån hos spelaren?
- Hur kan en pekskärm och fysiska ArUco-markerade spelkort bidra till det upplevda tempot och stressnivån hos spelaren?
- Hur påverkas användarupplevelsen av att spela samma digitala kortspel med spelkort i verkligheten (fysiska) gentemot spelkort i spelet (digitala)?

1.4 Avgränsningar

Spelet är anpassat för att spelas på en 50 tum pekskärm som finns tillgänglig i Visualiseringscentrum i Norrköping och en Logitech C930e webbkamera. Spelet är optimerat specifikt för denna skärm och kamera, och är ursprungligen utvecklat för att besvara vetenskapliga frågor. Därför är spelet inte avsett att användas utanför testmiljöer som syftar till att utforska dessa vetenskapliga frågeställningar.

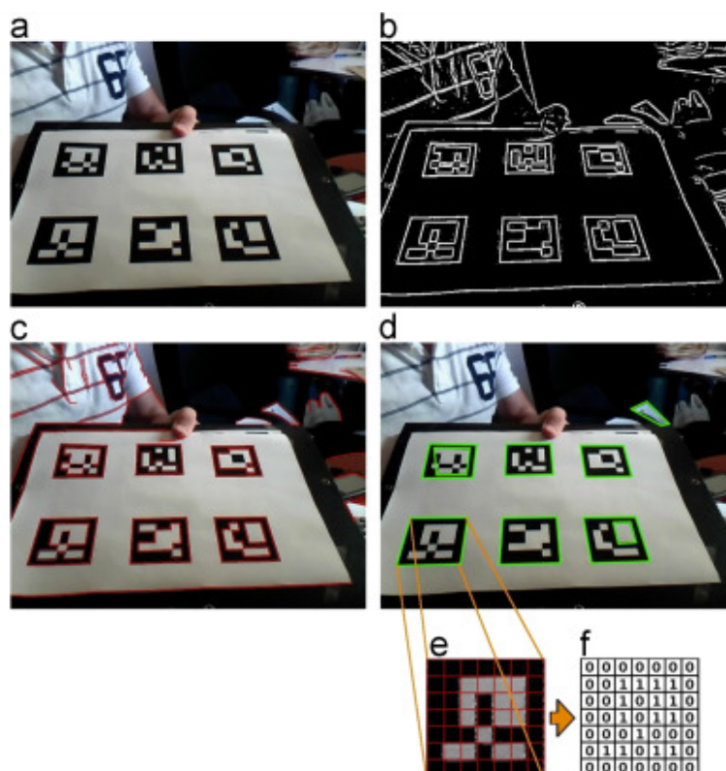
Kapitel 2

Relaterat arbete

2.1 Bildigenkänning

Det finns flera metoder för att läsa in fysiska kort i spelet, men det beslutades att användning av ArUco-markörer var det mest lämpade i detta fall. Detta är då ArUco är en typ av binär markör som endast innehåller ett ID, vilket gör den lättare att identifiera än till exempel QR-koder då den innehåller mindre information. De är även optimerade så att skillnaden mellan de olika markörerna är så stora som möjligt, vilket tillåter systemet att förstå vilken markör det är trots att några bits läses av fel [14].

ArUco tillhör biblioteket OpenCV, vilket är ett gratis *open-source* bibliotek för bildbehandling och datorseende. Det innehåller många funktioner för flera användningsområden, men i detta projekt behövdes endast de för ArUco-markörer.



Figur 2.1: Hur OpenCV identifierar ArUco-markörer [14].

2.2 Stress kopplat till beslutstagande

När spelet utvecklades var ett av målen att spelet skulle upplevas stressigt och ha ett högt tempo. Genom att spelarna måste fatta många beslut på kort tid, begränsar det möjligheten hos spelaren att göra välövervägda val jämfört med om de hade haft mer tid.

Studien *Decision making under stress: A selective review* [15] belyser att stress påverkar beslutsfattande avsevärt, vilket påverkar både vardagliga val och kritiska, livsförändrande beslut. Den påpekar att individer under stress kan välja alternativ som ger omedelbara belöningar, men som kan ha negativa långsiktiga konsekvenser. Denna tendens tyder på att stress kan leda till mer kortsiktiga beslut.

2.3 Stress i kortspel

För att kunna besvara frågeställningarna har spelet tagit inspiration av andra digitala kortspel. Kortspel som *Hearthstone*[1], *Gwent*[11] och *Legends of Runeterra*[3] är digitala *Collectable Card Games* (CCG) som liknar varandra i det att det finns två spelare som spelar mot varandra och som spelar endast på sin egen runda. Detta är ett sätt att skapa kortspel på, men det tillbringar oftast inte stress eller högt tempo och spelen har därför en timer för hur lång tid en runda är så att spelare inte tar för lång tid på sig att agera. Ett spel som detta projekt i stället har tagit stor inspiration av är *Clash Royale*, som också är digitalt kortspel, men som inte har några rundor som varsin spelare spelar på. *Clash Royale* går ut på att förstöra sin motståndares torn innan de förstör ens egna torn. Det finns också en timer som avslutar spelet om spelarna inte förstört varandras torn när timern är slut. Om en spelare förstör deras motståndares tre torn eller förstör fler torn än deras motståndare när timern är slut så vinner dem.

I stället för rundor, som *Hearthstone*, *Gwent* och *Legends of Runeterra* har, så spelar båda spelarna i *Clash Royale* samtidigt. För att tempot inte ska bli för högt så finns en resurs som heter *elixir*, vilket liknar *mana* från andra spel. *Mana* är en resurs som används för att spela kort och genererar passivt över spelets gång, vilket bestämmer hastigheten av kort per sekund som kan spelas. För att uppmuntra till att spelare inte ska lagra en stor mängd *mana* och spela alla sina kort med en gång så finns det en gräns på hur mycket *mana* en spelare kan ha, och om spelarna inte spelar kort medan gränsen är uppnådd så får de ingen passiv *mana* och därför slösar *mana* de annars kunde spendera. Detta leder till en god balans mellan att vänta till det perfekta tillfället att spela sina kort, men också att inte vänta för länge, vilket ger spelet en god rytm och leder till ett gott tempo.

De mekaniker som gåtts igenom angående *Clash Royale* är de mekaniker *Elemental Clash* använder sig av för att balansera ett tempo som inte är för högt men ändå är högt nog att tillbringa stress hos spelaren och deras val i spelet.

Kapitel 3

Implementation och metod

Spelet består av många olika funktioner som implementerats. Detta kapitel förklarar hur alla olika delar av spelet implementerats.

3.1 Unity

För att implementera spelet används *Unity* som är en plattform för spelutveckling och en spelmotor som används för att skapa interaktiva 2D- och 3D-spel samt andra digitala upplevelser. Unity har flera funktioner och verktyg för att hjälpa utvecklarna med att uppfylla sina visioner och för att underlätta utvecklingsprocessen som verktyg för att bygga användargränssnitt och hantera input från tangentbord, mus, touchskärmar, eller spelkontroller och en inbyggd fysikmotor som möjliggör realistiska rörelser och kollisioner [7]. Unity används för att skapa scener, hantera interaktioner mellan objekt och implementation av kod under detta projekt.

För att undvika konflikter i projektet där flera utvecklare jobbar parallellt med varandra har *prefabs* använts. Enligt Rauber [10] kan prefabs hjälpa till med att undvika merge-konflikter genom att det delas upp spelets objekt i olika filer vilket tillåter utvecklare att arbeta separat från varandra. Spelobjekt såsom monster, torn och spelplan har därför brytits ut i egna prefabs för att undvika merge-konflikter. Prefabs används också då det är ett kraftfullt verktyg för att effektivisera och strukturera arbetsflödet i Unity. De tillåter utvecklare att återanvända och hantera spelobjekt på ett enkelt och effektivt sätt.

Det finns två scener i projektet, en för startskärmen och en för själva spelet. Där spelet startar på startskärmen där spelarna kan kalibrera kameran, fixa inställningar och klicka sig vidare till spelet.

3.1.1 Animation

Genom att animera karaktärersmodellerna blir spelet mer intuitivt och spelaren får visuell återkoppling på vilket läge karaktärerna är i under spelets gång. För att uppnå detta används dels programmet *Adobe Mixamo*. Mixamo är ett program som kan användas för att automatiskt rigga eget skapade 3D-Modeller. Genom Mixamo har modellerna i spelet riggats, vilket är nödvändigt för att kunna animera dem. I Mixamo finns även animationer som kan hämtas och appliceras på ens egna karaktärer, och det är här som de tre animationerna som har använts i detta spel har hämtats ifrån. Dessa animationer är en *walking*, *attacking* och en *death* animation, vilket är de tre lägena som en karaktär i detta spel kan ha.

I själva spelet används Unity:s inbyggda *Animation Controllers* för att möjliggöra animering av karaktärerna i olika lägen. Detta görs med hjälp av en så kallad *Animator* komponent med en kopplad

Animation State Controller samt ett skript som appliceras på alla spelkaraktärer. I karaktärernas *Animation State Controller* definieras tre variabler av typen *boolean* (alltså kan sättas som sanna eller falska) som heter *isWalking*, *isAttacking*, och *isDead*. Alla dessa booleans är kopplade till varsin animation. Skriptet kollar vilket läge karaktären är i och ändrar värdet på dessa booleans i *Animation Controller* komponenten. Exempelvis om karaktären är i ett läge där den attackerar en annan karaktär sätter skriptet variabeln *isAttacking* till "true" i *Animation Controller* komponenten, och därmed sätter komponenten karaktärens animation till den attackerande animationen som är kopplad till denna variabel.

3.2 OpenCV

Då *OpenCV* är ett bibliotek som framförallt stöder språk som *C++*, *Java*, *Python* och *Matlab*, behövdes *OpenCV plus Unity*[16] som hämtades från Unity:s *Asset Store*. Denna asset är en version av *OpenCV* som är optimerat för *C#* och kan användas direkt i *Unity*. Då denna asset inte kontinuerligt uppdateras så har den framför allt huvudfunktionerna av *OpenCV*, men inte de nyaste versionerna. Dessa funktioner var dock tillräckligt och gjorde det vi behövde; att hämta ett ID från en *ArUco*-markör, hämta positionen på markören, och skapa en rektangel av tre markörer för kalibreringen.

I skriptfilen *arucoreader* används alla dessa funktioner. Först identifieras ID:t som sparas i en lista. Kameran läser av ID:t och positionen hämtas med hjälp av markörens hörn där ett medelvärde räknas av hörnen för att få fram mitten på markörens position. Dessa positioner kan sedan användas på tre förbestämda ID:n för att skapa en rektangel mellan dessa se figur 3.1 och 3.2. Denna rektangel är till för att berätta för systemet var i bilden spelplanen befinner sig. Måtten på rektangeln sparas för att sedan användas som ett internt koordinatsystem och räkna ut en relativ position i rektangeln för att positionen man lägger kortet på skärmen ska stämma överens med spelets koordinater. Då själva ID:t endast håller information om vilket monster kortet representerar och inte vilket lag det tillhör, bestäms detta också med hjälp av positionen att kolla vilket sida av planen kortet befinner sig på.

3.3 Kort

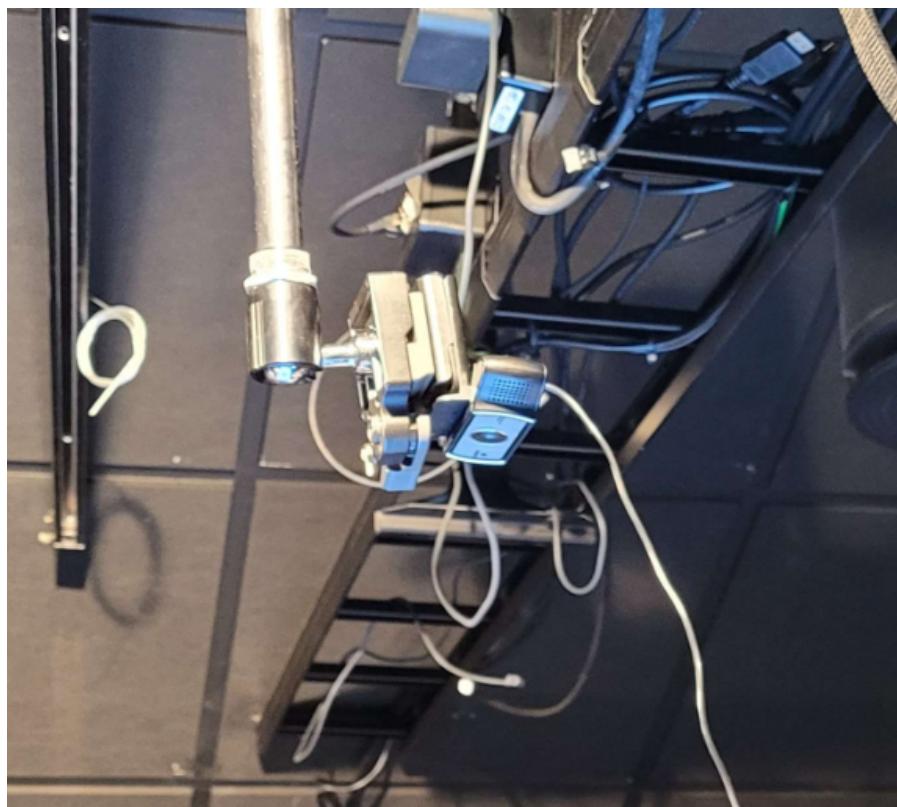
I detta kapitel diskuteras hur fysiska och digitala kort har implementerats i spelet för att undersöka deras inverkan på spelupplevelsen. Digitala kort implementerades parallellt för att möjliggöra en jämförelse med de fysiska korten och mäta skillnader i tempo och stressnivå hos spelaren.

3.3.1 Fysiska kort

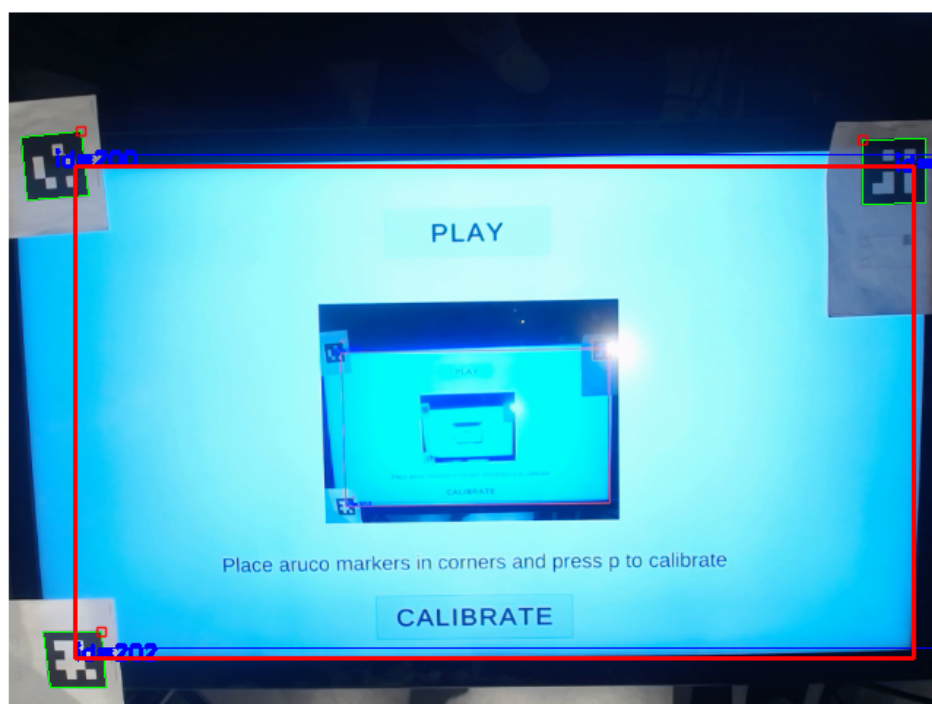
För att kunna spela spelet med förståelse om vad som händer samt besvara frågeställningarna behövs fysiska kort. Korten har *ArUco*-koderna på baksidan för att framkalla monsterna och information om monstertyp, liv, resurskostnad och en liten text om monstret på framsidan, se fig. 3.7.

Då det finns sex olika element som monsterna kan ha och tre typer av monster per element behöver hela 18 kort designas, samt hitta en storlek på dessa kort som kameran kan hitta. Därför har temporära kort använts för att utveckla projektet medan en design för korten tagits fram. Dessa koder är 10x10 cm stora för att kameran med säkerhet ska kunna identifiera koderna och är lätta att generera vid behov.

När användartester skulle hållas skrevs de slutgiltiga korten ut. Då de ska gå att hålla i handen utan att vara för stora behövde även *ArUco*-koderna vara mindre, men fortfarande tillräckligt stora för att



Figur 3.1: Kameran för projektet.



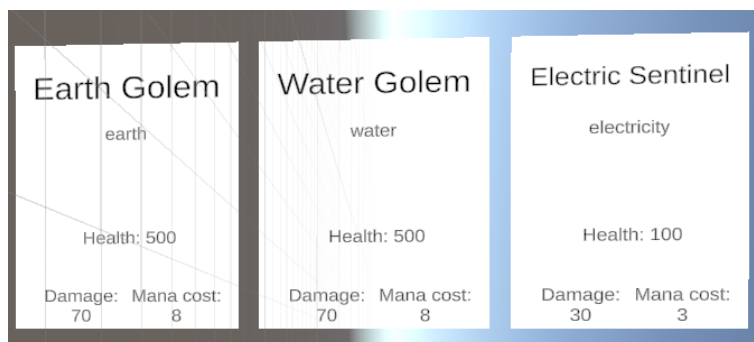
Figur 3.2: Kalibrering i spelet.

kunna läsas av. Därför gjordes de så stora som möjligt vilket resulterade i 6x6 cm för att få plats på korten.

3.3.2 Digitala kort

För att undersöka hur fysiska kort påverkar spelet behövdes något att jämföra med, därför implementerades digitala kort. Det första som gjordes var att skapa kortobjektet. Detta gjordes genom att skapa en *Prefab* som bestod av olika UI-element. Ett bildobjekt användes som bakgrund till kortet och flera textobjekt skapades ovanpå bakgrunden för att visa information såsom namn, liv och skada. De digitala kortens design kan ses i figur 3.3.

För att kontrollera de digitala korten skapades en skriptfil *DigitalCardHandler* som ansvarar för touch-input till korten, att generera nya kort då gamla spelas och att skapa rätt monster när kort spelas. För att hålla koll på vilket monster som korten ska skapa har en lista med par av datatyperna *Gameobject* och *int* skapats för varje lag. *Gameobject* är kortobjektet i spelet och *int* är det id som tilldelats kortet vid generering. För att generera ett kort med ett slumpmässigt id från databasen kallades medlemsfunktionen *AssignRandomToDigitalCard* som skapades i *spawn*-skriptet som beskrivs i 3.6.4. Denna medlemsfunktion hämtar ett slumpmässigt objekt från databasen och skriver dess information till kortobjektets text, sedan returneras det id som valts och sparas i ett par av nämnd typ. För att skapa ett monster då kortet placeras kallas samma funktion som används för de fysiska korten i *spawn*-skriptet men med position som fås av touch-input och id som fås från det markerade kortet. För att hålla koll på vilket kort som markerats av spelarna skapades en *selectedCard* variabel av typ *Gameobject* för varje lag.



Figur 3.3: De digitala korten.

3.4 Speldesign

Ett av syftena med projektet var att undersöka hur spelet med tillhörande fysiska kort kan utformas för att höja speldynamiken och spelarengagemanget bland spelare, och spelet har därmed utvecklats med detta i åtanke.

Som nämnt i 2.3 så kommer spelet att likna Clash Royale, vilket enligt definitionen given av [8], inte är ett *casual video game* (CVG). Ett CVG enligt recensionen är ett spel karaktäriserat av låga kognitiva belastningar och korta tidskrav. Medan exempelvis Clash Royale har korta tidskrav så behöver spelare använda sin fulla kognitiva förmåga för att vinna. Detta är dels för att spelet är ett flerspelare spel och spelare tävlar mot varandra för att vinna, där endast en av spelarna faktiskt vinner, men det är också eftersom att spelet ger uppmuntran till snabba beslut med många strategier, som varierar beroende på motståndarens ageranden.

Recensionen förklarar också att CVGs ofta kan ge upphov till ett så kallat *flow-state*, vilket först beskrevs av Csikszentmihalyi [2] och kan som beskrivet i tidigare nämnda recensionen förklaras som ett tillstånd där en person finner en jämvikt mellan individens färdigheter och spelets regler, mål och utmaningar. Flow-state är inte vad en stressad person upplever, utan är istället ett tillstånd där spelaren skulle känna sig vid lugn men fortfarande vara engagerad. Spelet gruppen utvecklar kommer därför försöka göra spelet svårt nog att vara kul men också alltid ge spelaren för lite tid att känna sig bekväm, för att ge upphov till stress.

Spelet har utvecklats för att spelas på en stor pekskärm, och skärmen används för att utföra flera saker i spelet. När en spelare vill använda ett kort för att generera en spelkaraktär eller en *spell* på spelplanen måste de trycka på en *Spawn* knapp som finns på pekskärmen. På pekskärmen finns även en så kallad *minimap*, vilket är en förminskad version av hela spelplanen som finns på varje spelares sida av skärmen. Om spelaren använder digitala kort i stället för fysiska måste spelaren interagera med denna minimap för att bestämma var en spell ska genereras. Genom att använda en pekskärm gentemot en extern enhet som till exempel en mus eller kontroll kommer gruppen undersöka om spelupplevelsen blir mer intuitiv på grund av den direkta interaktionen mellan spelaren och spelet. Hoppet är att detta gör att spelaren kan utföra saker i spelet snabbare, vilket bidrar till att hålla tempot högt under spelomgången och förbättrar speldynamiken.

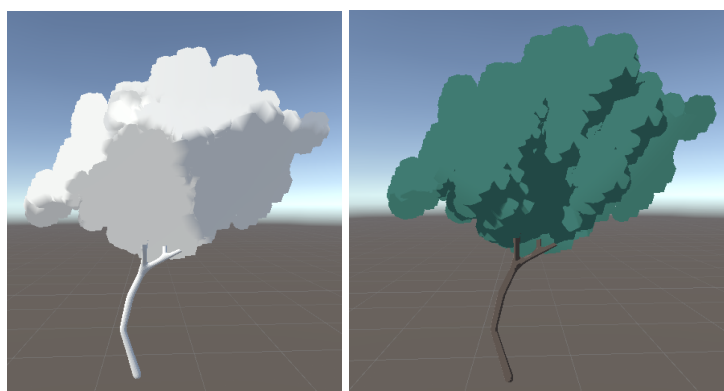
Något som är tänkt att höja spelarengagemanget är spelarens stressnivå, och därför har spelet utvecklats med flera stressfaktorer i spelet. Varje spelomgång är tänkt att vara kort, under 5 minuter, vilket gör att spelaren måste ta snabba beslut gällande vad de ska göra för att vinna mot motståndaren. Båda spelare kan hela tiden se hur mycket *health* alla torn och karaktärer har kvar. Förutom att detta är en bra visuell återkoppling till spelaren blir det också en stressfaktor om man exempelvis ser att ett av ens torn håller på att förstöras eller att ens spelkaraktär tar mycket skada när den blir attackerad av motståndaren. Förutom att ta snabba beslut måste även besluten vara genomtänkta, då varje kort har en viss kostnad för att kunna användas. Att använda ett kort som gör mer skada kommer att kosta mer, så spelaren behöver vara strategisk och planera sina attacker mot motståndaren. Det kan leda till stress om situationer uppstår exempelvis där spelaren använt upp all sin mana vilket släpper genom motståndarattacker lättare, eller om flera torn håller på att förstöras och spelaren har lågt med mana och behöver bestämma vilket kort som ska användas vart, och så vidare.

3.5 Grafik

Grafiksektionen fokuserar på den visuella designen och implementeringen av spelets grafiska element. Den behandlar 3D-modellering, designval och kameraperspektiv som används för att skapa en engagerande och intuitiv spelupplevelse. Valet av kameraperspektiv diskuterades ingående för att säkerställa att spelplanen är tydlig och lätt att navigera oavsett spelarens position framför pekskärmen.

3.5.1 3D-modellering

I spelet finns det en blandning av hämtade och egenskapade *assets*. En egenskapad asset är träden, som är gjorda i *Blender* och har en *shader* som är gjord direkt i Unity genom en hlsl-fil och en skriptfil. Då grafiken inte är fokuset av projektet så planerades det att tiden för dess implementation skulle hållas till de första sprintarna och inte jobbas på utav hela sprintteamet. Därför valdes det att grafikerna skulle vara *cartoon-inspererade*, det vill säga att grafiken ser tecknad ut. Därför gjordes träden och andra modeller med ett lågt antal polygoner då det först är enklare att modellera och sen också gör att prestandan av spelet blir bättre. Materialet som nämndes tidigare är en variant av *Blinn-Phong* shading modellen. Shadern skapades med hjälp av en guide [12] som går igenom både teorin, inspirationen

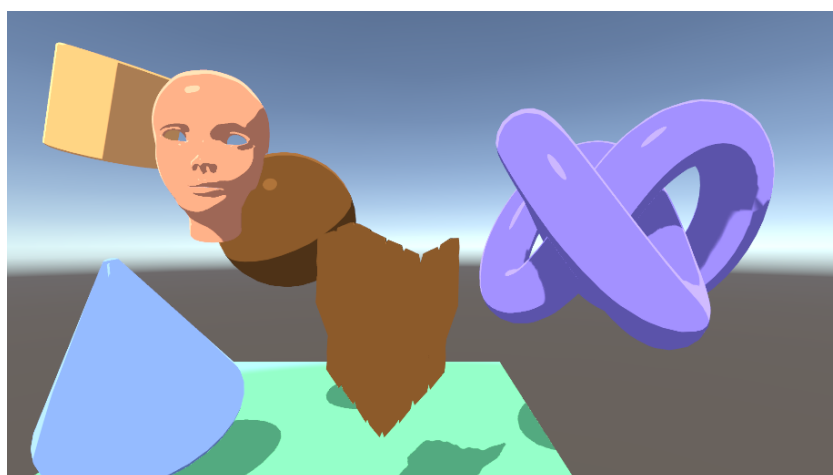


(a) Träd utan material.

(b) Träd med toon-shader.

Figur 3.5: Träden i spelet med olika material

och implementationen av shadern. Blinn-Phong modellen är bra då det är en billig *lighting model* som har simpel logik bakom sig och välkänd, och har därför mycket dokumentation online samt i litteratur. Ett exempel på hur shadern ser ut på ett visst antal objekt kan ses i 3.4. Som det syns så ser avrundade objekt bäst ut då det utnyttjar den spekulära shadingen. Träden utan materialet applicerat syns i 3.5a och med materialet applicerat i 3.5b. Träden är skapade med hjälp av Blenders partikelsystem för löven, som är sfäriska, och sedan använder en *decimate-modifier* för att göra om sfärerna till en form som innehåller färre polygon men som behåller formen något.



Figur 3.4: Olika 3D-modeller med toon-shadern applicerad i Unity.

3.5.2 Design

Temat för spelet bestämdes gemensamt under *Scrum*-mötena i början av projektet. Det bestämdes då att spelet skulle handla om *wizards* i torn som slåss mot varandra genom att kalla på *spirits* och *elementals* som slåss åt dem. Från början var det tänkt att alla monster i spelet skulle vara *spirits*/magiska manifestationer av de olika elementen. Detta ändrades sedan till att de skulle vara *elementals*, eftersom de har en mer fysisk form snarare än andlig, vilket skulle underlätta när 3D modeller för monstren skulle skapas. Det ansågs även tematiskt mer troligt att en *wizard* skulle kunna skapa *elementals* som lyder deras order än att de skulle kunna övertyga naturliga *spirits* att slåss för dem. De olika elementen hos monsterna tillät att ett ”sten sax påse” system kunde inkluderas där vissa element är starkare mot vissa element och svagare mot andra. Det bestämdes att det skulle finnas sex olika element i spelet

och att det skulle finnas tre unika typer av monster inom varje element, vilket gör att det finns totalt 18 unika monster. De tre typerna är *golem*, *wyvern*, och *sentinel*. *Golems* har mer *health* och högre *attack damage* än de andra typerna, men har lägre *attack speed* och *walk speed*. De är alltså stora och starka, men är väldigt långsamma. *Wyverns* kan röra sig och attackera mycket snabbare än *golems*, men har inte lika mycket *health* och gör inte lika mycket skada. *Sentinels* har mycket högre *attack range* än de andra, vilket gör att de kan stå och attackera på håll medan de andra måste gå fram till dem för att kunna attackera tillbaka. På grund av detta gör de inte så mycket skada som de andra och har inte så mycket *health*.

För att underlätta inför skapandet av 3D modellerna skapades det *concept art* för de tre olika monsterna. *Concept arten* avslutades dock inte helt eftersom mer fokus behövde läggas på koden till spelet. Sedan mot slutet av projektet insåg gruppen att det inte fanns tillräckligt med tid att skapa alla unika 3D modeller som hade planerats. Planen ändrades då till att skapa tre modeller, en för varje typ av monster, som sedan med hjälp av texturer kunde ändras för att visa vilket element de har. Det skapades även *concept art* för designen för hur spells, de fysiska korten, och spelplanen skulle se ut.



Figur 3.6: Första iden för designen av monster (*spirits*)



Figur 3.7: Vänster: Design för spelplan. Höger: Design för kort



Figur 3.8: *Concept art* för olika monster och spells

Då tiden inte fanns för att modellera samtliga 3D-modeller och torn har 5 av 6 modeller hämtats ifrån *Unity asset store* och *TurboSquid* som kan ses i 3.9. De modeller som har hämtats till monstren *Sentinel*, *Shambler*, *Golem* och *Crusher* samt modellen till tornen. Både *Unity asset store* och *TurboSquid* tillåter användning av *assets* som är gratis eller köpta för kommersiellt bruk så länge de inte är tillgängliga för användaren i produkten eller säljs vidare. Detta är något *Unity* sköter automatiskt när produkten byggs.

3.5.3 Val av kameraperspektiv

En av de första besluten som behövde göras angående spelets grafik var vilket kameraperspektiv som skulle användas. Olika spelplansprototyper skapades i *Blender* med olika typer av kameraperspektiv för att sedan testas på pekskärmen, dessa kan ses i figur 3.10. Först testades ett perspektiv som visar spelplanen från sidan ner som syns i figur 3.10a. Detta testades för att undersöka om perspektivet ser bra ut oavsett om man står längs den breda sidan av skärmen eller den korta. Perspektivet gjorde det svårt för spelarna att se spelet och därför testades 3.10b som ser spelplanen rakt ovanifrån. I 3.10c undersöktes det om två kameror med ortografiskt perspektiv kunde användas för att ge spelaren en känsla av att hen står bakom sina torn. Den ortografiska kameran användes så att ingen perspektivförvrängning skedde och så att de två kamerorna kunde perfekt mötas upp i mitten av spelplanen. Den sista och slutgiltiga idéen syns i 3.10d. För att spelplanen ska se bra ut för så många som möjligt användes en kamera som tittar rakt ner på spelplanen. För att minska perspektivförvrängningen av tornen placerades kameran väldigt högt upp men med en hög brännvidd.

3.6 Spelllogik

I detta kapitel beskrivs spelets logik och mekanismer som styr hur karaktärer rör sig, interagerar och påverkar varandra. Rörelsesystemet för karaktärerna implementerades med *Unity:s* *AI Navigation*-paket, och stridssystemet definierar hur karaktärer attackerar och tar skada. Kapitlet täcker även specifika spell-funktioner och deras unika beteenden, samt hur spells kan placeras strategiskt på spelplanen.

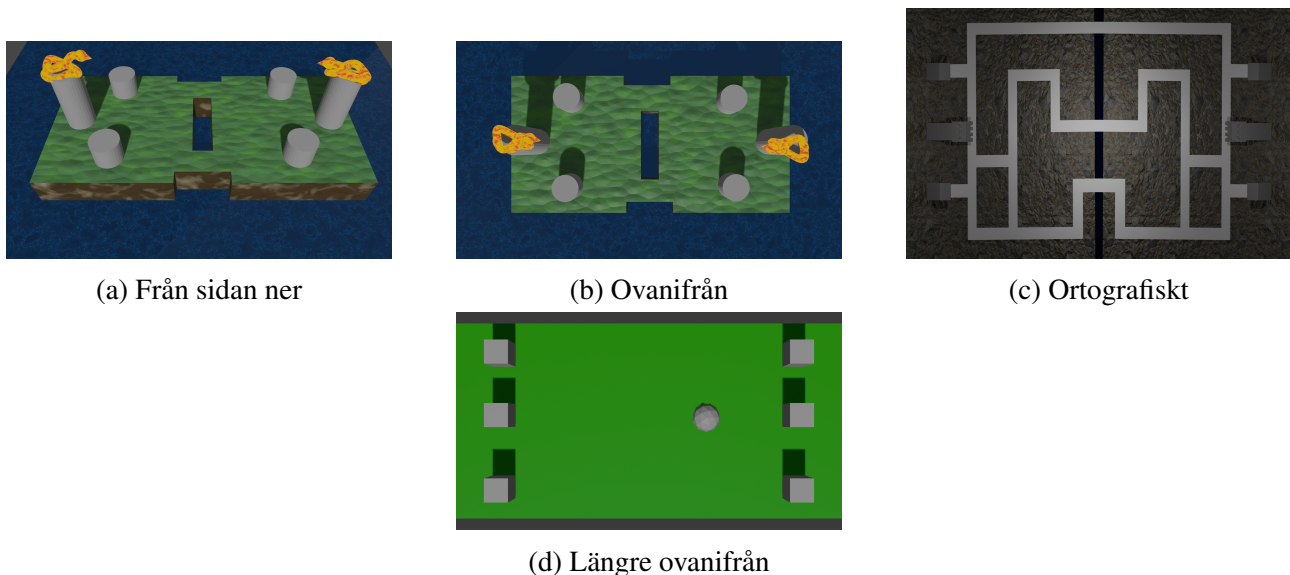


Figur 3.9: Samtliga monster i spelet

3.6.1 Databas

All information kopplade till korten är samlat i en databas som är uppbyggd i en JSON-fil. Databasen tillhandahåller följande information om korten:

- Index
- Namn
- Health
- Mana cost
- Base Damage
- Attack range
- Detect range
- Movementspeed
- Attackspeed
- Element typ
- Lag



Figur 3.10: Fyra olika prototyper för kamerans perspektiv

Skriptet *reader* innehåller kod för att kunna läsa av JSON-filen för att sedan skapa en *CardList* vilket är en lista av samtliga kort som sedan kan användas för att hämta väsentlig information till karaktärerna. Reader-skriptet kallas direkt när spelet startar och skapar då en instans av *CardList* som kan redigeras och ändras på utan att vi ändrar i originallistan.

3.6.2 Rörelse

För att få karaktärerna att röra sig över spelplanen användes Unity:s *AI Navigation* paket. Paketet låter användaren skapa ett *NavMesh Surface* objekt som baserat på inställningar bakar en *NavMesh* över all objekt i scenen. Den bakade NavMeshen avgör var karaktärer i scenen kan röra sig och *Pathfinda* till. För att få ett spelobjekt att röra sig på NavMeshen läggs en komponent *NavMesh Agent* på spelobjektet. Spelobjekten som skulle röra sig i detta fall var våra monster, en *NavMesh Agent* komponent lades därmed till på monster prefaben.

Skriptfilen som ansvarar för monsters rörelse är "AI-Locomotion". Logiken för monsters rörelse beskrivs i detalj i 3.6.3 och är beroende av attacklogiken. Vad som i slutändan avgör hur ett monster rör sig är medlemsvariabeln *enemy*. Om *enemy* är ett annat spelobjekt sätts *agent.destination* till *enemy:s position*. Vad *agent.destination* gör är att hitta den kortaste banan från agenten själv till *enemy* längs NavMeshen, detta får monstret att röra sig.

3.6.3 Combat

Stridssystemet i spelet, alltså det som bestämmer hur karaktärer attackerar varandra och hur de tar skada, definieras mestadels i skriptet "AI-Combat" och "AI-Locomotion", två skript som appliceras på alla spelobjekt i scenen (tornen och karaktärerna). I AI-combat skriptet definieras följande variabler hos spelobjekten:

- Nuvarande health
- Maximal health
- Grundläggande skada som objektet tillfogar andra objekt

- Elementtyp (gäller inte på torn)
- Lag (vänster eller höger)

Ytterligare två variabler som har med stridssystemet definieras i skriptet AI-Locomotion:

- Detekteringsavstånd
- Attack-avstånd

Alla dessa variabler är faktorer som bestämmer om två objekt i spelet attackerar varandra samt hur mycket skada varje attack ger.

En karaktär i spelet har som grundläggande mål att röra sig mot det motståndartorn som är närmast och attackera den. Om karaktären under färdens gång detekterar en motståndarkaraktär inom sitt detekteringsavstånd börjar den i stället röra sig mot motståndaren. När karaktären har rört sig tills den är på attack-avstånd till motståndaren stannar den och börjar attackera motståndaren med sin grundläggande skada. Förutom grundläggande skada ger karaktären även en extra skada som bestäms av vilken elementtyp båda karaktärerna har. Exempelvis, om karaktären är en vatten-typ och motståndaren är en eld-typ ger detta mycket extra skada (eftersom vatten släcker ut eld), men om motståndaren i stället är en vind-typ ger detta ingen extra skada (eftersom vatten inte har någon speciell effekt mot vind). Karaktärerna attackerar varandra tills en av karaktärernas health blir noll, vilket betyder att denna karaktär ”dör”. Den vinnande karaktären fortsätter då röra sig mot sitt närmaste motståndartorn. Motståndartornen har en mycket större maximal health, samt en grundläggande skada som den tillfogar en attackerande karaktär (så att karaktären inte ska kunna attackera tornet för evigt).

3.6.4 Spawner

Skriptet för att *spawna* monster (att få ett monster att dyka upp på spelplanen) och aktivera spells får in ett ID från *ArUco* koden på de fysiska korten. Det får även in positionen som kortet har på spelplanen. ID:t används för att hitta kortet i databasen. I början av *spawner*-skriptet finns ett test som kallar på databasen för att se om kortet som spelats är ett monster-kort eller ett spell-kort. Ifall kortet som spelats är ett spell-kort kallas skriptet för spells, se 3.6.6, annars kör *spawner*-skriptet vidare. På *gameobjectet* som *spawner*-skriptet ligger på i scenen finns också en lista som innehåller *prefabs* för alla de olika monster som kan *spawnas*. Dessa ligger i samma ordning som de ligger i databasen, vilket gör att deras ID:n matchar. Alla monster *prefabs* har redan alla skript de behöver applicerade på sig. När monstret ska *spawnas* skapas en ny instans av monster-*prefaben* med ID:t från kortet. Monstret görs sedan till ett barn av en *prefab* som ligger i scenen, vilket applicerar viss skalning. Sedan tas alla monstrets *stats* från databasen med hjälp av ID:t och appliceras på rätt ställen i skripten som ligger på monstret. Detta gör att när ett monster-kort aktiveras dyker det upp på rätt ställe med rätt värden för den typen av monster.

3.6.5 Mana

För att kunna spela ett monster och kasta *spells* behövs en resurs för att begränsa antal kort som kan spelas samtidigt men också för att addera en strategisk aspekt. Implementationen av en resurs betyder att en kostnad tillhör varje kort, vilket får spelarna att tänka över vilket kort de vill använda. Valutan som valts är mana och spelarna kan ha som mest 10 mana-enheter på samma gång för att undvika situationen där en spelare konstant läser in kort och lägger ut dem på spelplanen. När spelet börjar är varje spelare given 5 mana-enheter.

Under spelets gång används funktionen *RegenerateMana* för att kontinuerligt återhämta mana-enheter. En viss mängd mana återhämtas för varje bilduppdatering i spelet, och om en spelare har nått max-gränsen av mana återhämtar man ingen mer mana tills man är under gränsen igen.

En *ManaBar* som finns på varsin planhalva visar hur mycket mana som varje spelare har kvar uppdateras genom funktionen *UpdateManaUI*, som tar antal mana-enheter som en spelare har och gör det till en visuell *slider*, och ändras i längd beroende på hur många mana-enheter som spelaren har kvar.

3.6.6 Spells

Spells är den andra korttypen, som skiljer sig från monster då spells inte använder sig av *stats* och varje spell har annorlunda beteende respektive till de andra spellsen. Spells hämtar fortfarande information från databasen, vilket beskrivs i 3.6.1, men eftersom de till skillnad från monster är mer unika jämte varann så sparas majoriteten av dess stats i olika spell-funktioner. En spell-funktion är en funktion som ger spells deras beteende beroende på vilken typ av spell det är. Till exempel så ger gräs-spellen en läkande effekt på monstren som tillhör samma lag som den som spelade spellen, medan en blix-spell gör skada på motståndaren monster.

Identifikationen av att ett kort är ett spell-kort beskrevs i 3.6.4. Efter att kortet är identifierat som sådant så kallar spawner-skriptet på *SpellCaster*-skriptet vilket har funktionen *SpellPlacer* som har in argument för spellens typ, vilket lag den tillhör och dess position. I funktionen bestäms vilken spell det är med hjälp av typen, och sedan kallar *spellPlacer* på den individuella spellen som ska utföras med dess position och lag.

Positionen hos spells fungerar annorlunda till monster, då ens egna monster alltid borde spawna på ens egen sida. Detta är inte lämpligt för spells då de ibland är bäst att använda när ens monster blir anfallna av motståndare på andra sidan spelplanen. Om detta är fallet så vill vi först och främst att spells ska kunna placeras på vilken position som helst på spelplanen, men det fungerar inte bra att använda samma metod som för monster då spelare inte ska behöva springa runt 50 tum skärmen varje gång de vill sikta med en spell. Därför finns det en *minimap* på varsin sida av spelplanen, som gör att man kan sikta spellen över hela spelplanen men från en nerskalad yta, vilket löser problemet med positionering hos spells. Funktionaliteten för minimappen är i *LookForPress*-skriptet som håller koll på alla fingrar som är placerade på skärmen och ger återkoppling om att spelet ser ens finger genom att spawna ett partikelsystem under fingret, samt att det spawnar en indikator till vart spellen siktas mot någonstans när man håller över minimappen.

3.6.7 Runor

Runsystemet är ett system som utvecklades under den andra sprinten men som i projektets skede togs bort då det inte interagerade med de andra spelmekaniker som gruppen skapat på ett lämpligt sätt. Funktionen är dock färdigutvecklad och en stor del av arbetsinsatsen gick till den, samt att den var planerad att vara med i slutspelet fram tills den sista sprinten och har därför sitt eget avsnitt. Runor är mönster som ritas med partikelsystem i *RuneMaker*-skriptet. Idén är att spelarna måste följa mönstret med sina fingrar och på så sätt rita runan för att spawna ett monster- eller spell-kort. I 3.12a syns en runa som spelaren inte har rört än, och i 3.12b syns det hur den ser ut efter att spelaren följt runan en liten bit. Runan är som nämnt uppbyggd av flera partikelsystem, och om spelaren rör vid ett av dem så färgas de om från blå till röd. Om spelaren får alla partikelsystem i en runa att bli röda så skulle ett monster spawna eller en spell bli använd. Funktionens var tänkt att lägga till ett stressmoment i spelet där spelaren behövde rita en runa snabbt för att ta del på sina kort, men detta gjorde i stället så att det blev för många steg för att spela ett kort. Därför valdes funktionen bort efter möte med kunden som föreslog att just denna funktion var det som borde tas bort.

3.7 Ljud

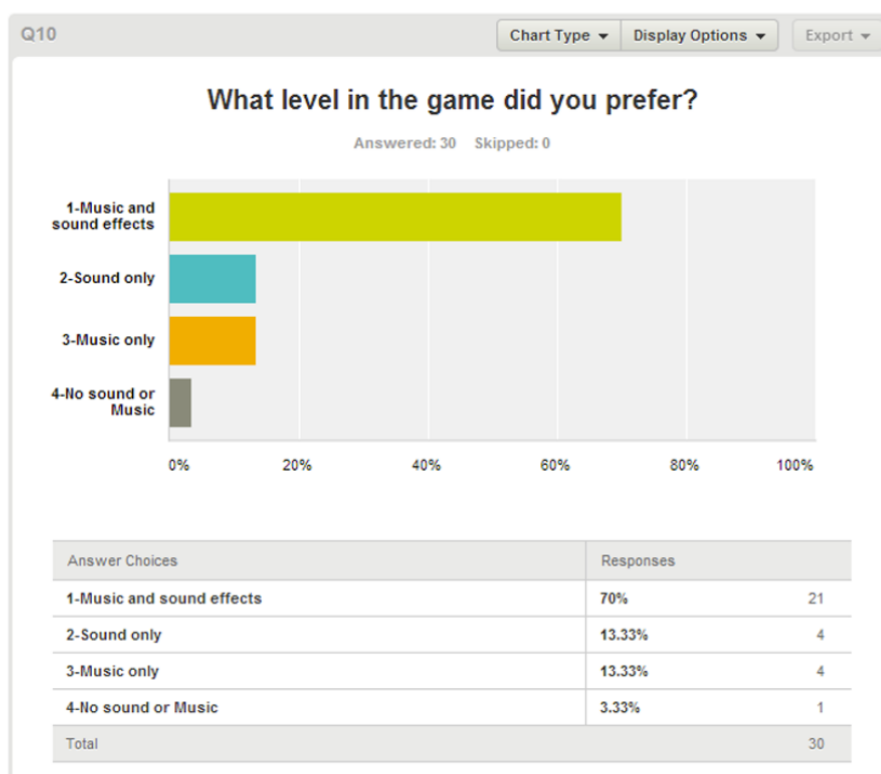
För att få mer inlevelse i spelet lades ljud till, både i form av musik och effekter. Tre olika låtar valdes till olika moment i spelet där en lugn låt spelades i menyn, en intensiv låt när spelet spelas, och en gladare låt efter matchens slut för att indikera att någon har vunnit. Att gå från en lugn låt till en intensiv låt när spelet startar ger användaren en tydligare känsla av att man är i en kamp mot en annan spelare [9].

Ljudeffekterna är till för att rikta spelarens uppmärksamhet och hjälper användaren att tydligare förstå när något händer. Dessa ljudeffekter är när man trycker på en knapp, när det är mindre än tio sekunder kvar på matchen, när ett monster *spawns*, och när man har för lite mana för att använda ett kort [9].

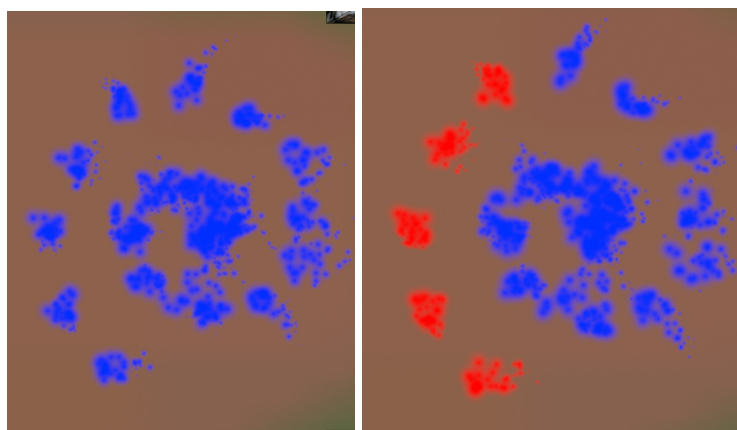
Under första användartestet tyckte vissa användare att det var otydligt varför det inte gick att lägga ut ett kort, då det både kunde varit problem med avläsningen eller att man inte hade tillräckligt med mana. Därför lades en indikator till i form av ljud som spelas när ett kort kostar mer mana än vad användaren har. Detta indikerar att något gått fel och riktar användarens uppmärksamhet mot dess mana.

Ljudet för när monster *spawns* lades till då monstret skapas under kortet som läggs ut och därför inte ser monstret direkt. Ljudet indikerar då att det har skett och spelaren kan ta bort sitt kort från planen.

För att upplysa spelarna om att det är lite tid kvar av matchen spelas ett tickande ljud. Detta är ofta ett ljud som är associerat med att man behöver skynda sig och gör därför slutet av matchen mer stressig, samtidigt som man har lite tid kvar att göra skillnad i matchen. Dessutom är rytmiskt repeterande ljud ett vanligt sätt inom ljud och musik att skapa spänning och stress [6].



Figur 3.11: Vad användare föredragit i en studie om hur ljud påverkar inlevelse [5].



(a) Oberörd runa

(b) Berörd runa

Figur 3.12: Runor i olika tillstånd

3.8 Användargränssnitt

För att låta användaren interagera med spelet behövde ett användargränssnitt implementeras. För att användaren skulle kunna välja olika inställningar för spelet skapades en startmeny som syns i figur 4.4 och 4.5. Denna låter användaren välja om digitala kort ska användas eller inte och låter användaren kalibrera skärmen för de fysiska korten.

För att indikera lag och hur mycket liv ett monster eller torn har implementerades en *healthbar* som syns i figur 4.2a. Denna healthbar innehåller en rektangel som krymper på längden då objektets health går ner. För att göra det tydligare hur mycket health objektet har ändras healthbarens färg från grön till röd då health sjunker. Dessa färger brukar användas för att indikera fara och säkerhet och förstärker därmed indikatorn.

För att visa att spelet tagit slut skapades en slutskärm som syns i figur 4.6. Denna slutskärm dyker upp då ett av lagen inte har några torn kvar eller då spelet har pågått i 5 minuter. Slutskärmen visar olika text för de två spelarna beroende på vem som vann eller om det blev oavgjort. För att öka stressen hos spelarna lades det till en synlig timer och en tickande ljudeffekt då det var mindre än 10 sekunder kvar av matchen. Den tickande ljudeffekten står ut från resten av spelet och ger en signal till spelaren att det är ont om tid.

3.9 Användartester

Då frågeställningarna handlar om hur stress och tempo påverkas av olika faktorer i spelet utfördes användartester för att undersöka detta. Användartesterna bestod av ett alfa och beta-test, och användarna fick därefter uttrycka sina åsikter om testerna genom att svara på en enkät med utvalda frågor som berörde testerna. Användartesterna är baserade på tabell 1 från artikeln *The Validity and Effectiveness of a Business Game Beta Test*[4]. Där testerna utfördes genom att låta två utomstående personer inom målgruppen spela spelet mot varandra, först en runda med digitala kort och sedan en runda med fysiska kort. En relativt omfattande introduktion till spelets regler gavs, men den var inte alltför detaljerad eftersom en del av undersökningen handlade om att bedöma spelets intuitivitet och se om användarna själva kunde lista ut hur man spelade. Efteråt fick användarna fylla i ett formulär med följande frågor:

- Hur upplevde du tempot av spelet? (välj 1-5)

- Hur stressad kände du dig under spelet? (välj 1-5)
- Om du kände dig stressad, vad var det som bidrog till den känslan? (Svarsfält)
- Kände du dig mer stressad när du använde de fysiska korten eller de digitala? Varför? (Svarsfält)
- Var de fysiska spelkorterna enklare att använda än de digitala? Vad kan förbättras? (Svarsfält)
- Hur upplevde du längden på timern? (1-5)
- Upplevde du någon stress/tidspress från timern? Om ja, på vilket sätt? (Svarsfält)
- Hur upplevde du balanseringen för monster vs monster? (1-5 samt svarsfält för kommentarer)
- Hur upplevde du balanseringen för monster vs torn? (1-5 samt svarsfält för kommentarer)
- Övriga åsikter

Under användartesterna observerade utvecklingsgruppen både spelet och spelarna noggrant, och gjorde detaljerade anteckningar. Gruppen fokuserade på att bedöma spelarnas stressnivåer under spelet, notera om spelarna blev alltför stressade eller om de fick vänta för länge och blev uttråkade. Dessutom dokumenterades eventuella buggar som upptäcktes under spelomgångarna.

Den första omgången av användartesterna visade att spelet hade många buggar, vilket var en oönskad stressfaktor. Efter att buggarna åtgärdades genomfördes en ny omgång användartester.

Kapitel 4

Resultat

Det finns två huvudsakliga resultat för projektet. Det första är spelet, som är till grund för att kunna besvara frågeställningarna. Det andra är användartester som ger möjlighet till att besvara frågeställningarna.

4.1 Spel

Ett av resultaten är spelet som producerades i Unity. Spelet har fungerande 3D-karaktärer med AI för dess rörelse och attack. Karaktärerna har även fungerande animationer då de rör sig och attackerar motståndare. OpenCV för Unity används för att läsa av ArUco-kort som placeras på skärmen och återger dess kort-id och position på spelplanen. Som alternativ till fysiska kort skapades digitala kort som kan användas i stället för de fysiska genom touch. Ett mana-system används som resurs för implementation av ett kontinuerligt villkor under spelets gång för att läsa in karaktärer och att använda spells. Dessa funktioner syns i figurerna nedan.

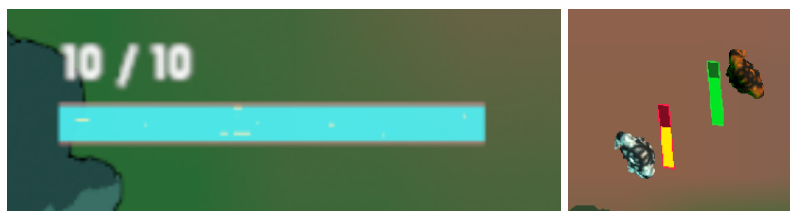


Figur 4.1: Spelet under speltiden.



(a) Ett torn med healthbar (b) Minimap med spawnknapp för kort

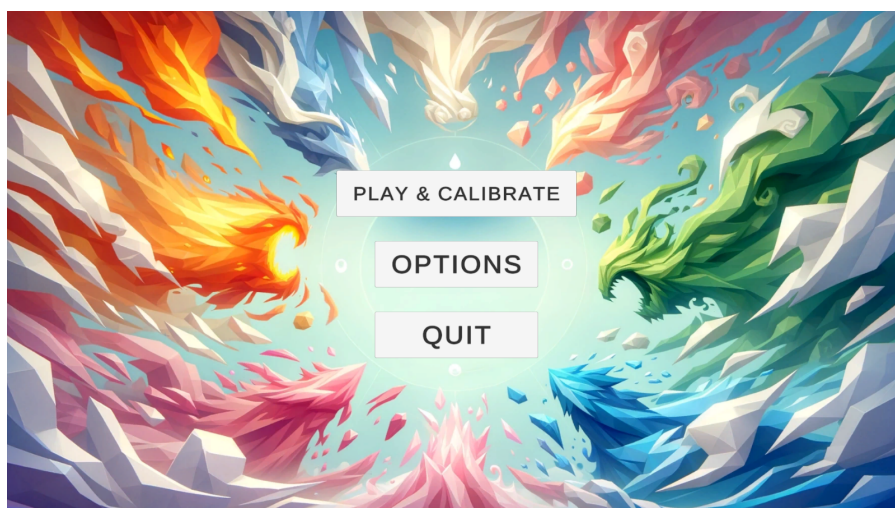
Figur 4.2: Några spelfunktioner.



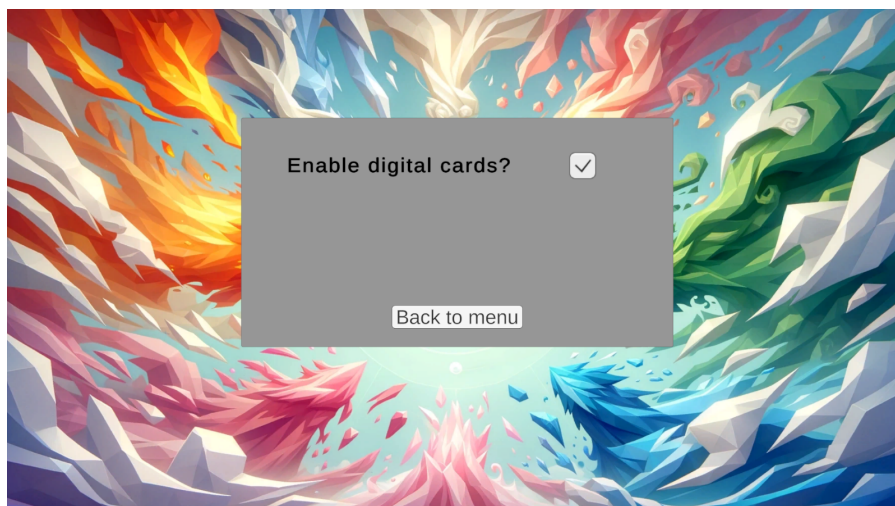
(a) Användarens mana

(b) Combat

Figur 4.3: Några spelfunktioner.



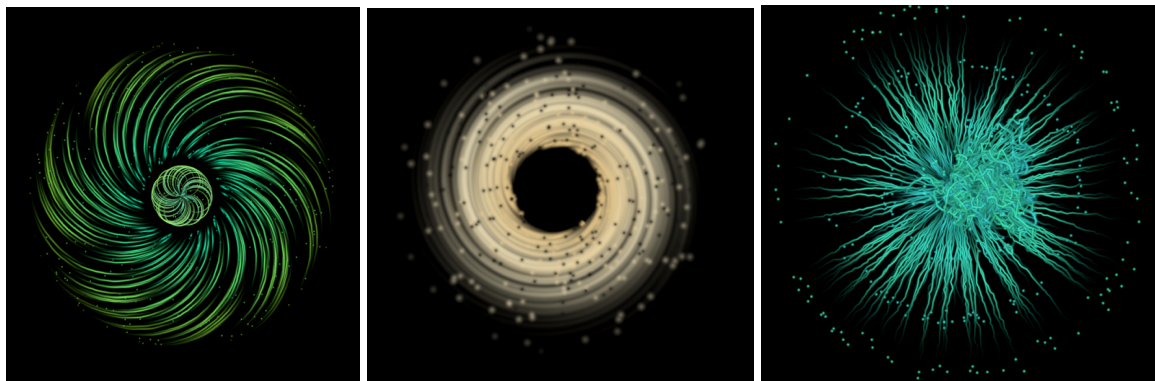
Figur 4.4: Startskärmen när spelet startats.



Figur 4.5: Menyn som kommer upp när man trycker på options.



Figur 4.6: Slutskärmen för när en spelare har vunnit.

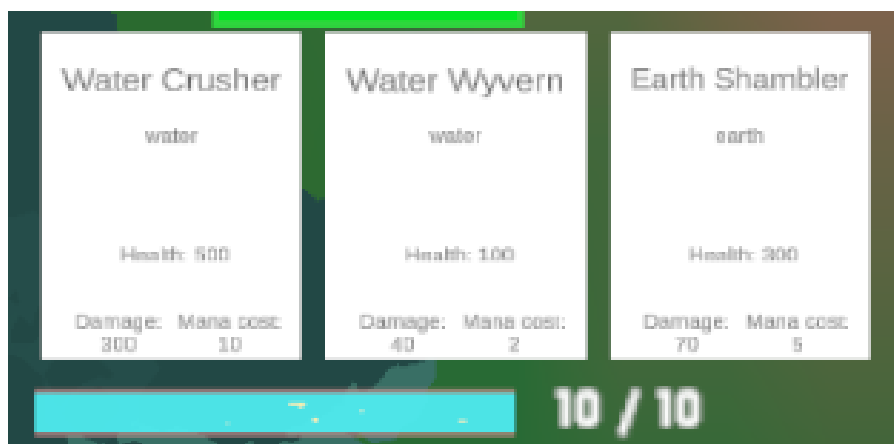


(a) Naturspell

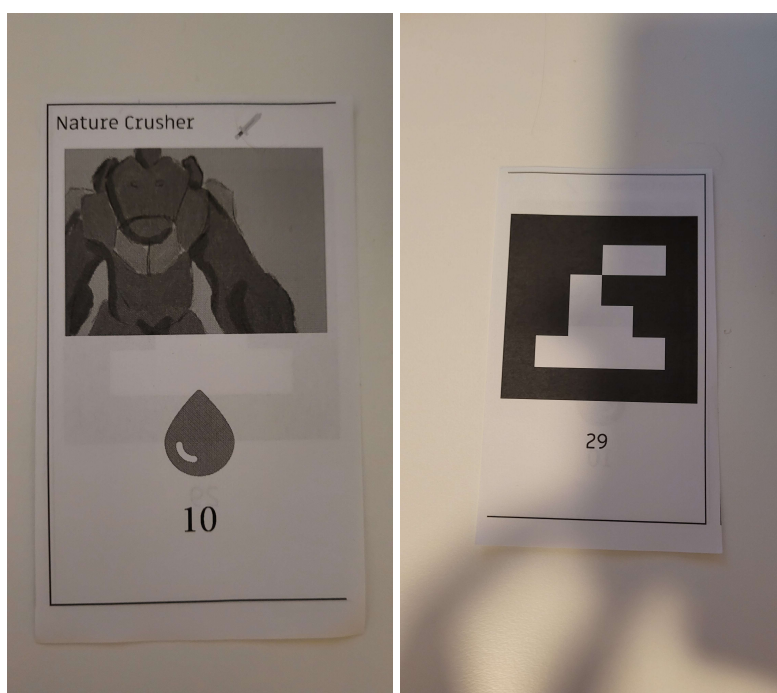
(b) Vindspell

(c) Blixtspell

Figur 4.7: Olika spells som har olika typer av effekter.



Figur 4.8: Digitala korten



(a) Framsidan av ett kort.

(b) Baksidan av samma kort.

Figur 4.9: Fysiska spelkort.

4.2 Användartester

Testerna utfördes på 12 olika människor som gav varsitt svar på formuläret och visas i [C](#).

De flesta upplevde att spelet hade ett högt tempo. I den första omgången av användartester var medelvärdet av betyget för hur stressigt spelet upplevdes en trea. Stressen för källan var däremot buggarna i spelet, vilket var negativ stress och gruppen fick ingen tydlig bild av hur själva speldesignen påverkade användarnas stressnivå. I de senare användartesterna upplevde användarna spelet som mer stressigt än i de tidigare testerna. Användarna svarade här att det var på grund av att tyckte spelet hade ett högt tempo och att de kände att de behövde spela kort hela tiden, och det påpekades även att det var ”bra” stress, alltså att det var spännande att spela. Det var dock några som påpekade att det var lite stressigt när de fysiska korten inte spawnades, och en som inte upplevde spelet som stressigt alls.

När det gäller åsikten angående de två olika korttypen framkom det under den första omgången av användartester att de fysiska korten ofta upplevdes som mer stressande än de digitala. Användarna tyckte att det var svårare att läsa av mana-kostnaden och hantera korten samtidigt som de behövde vända på dem och lägga ner dem korrekt för avläsningens skull. De digitala korten gjorde spelet snabbare, vilket ökade stressnivån, men upplevdes som mer användarvänliga och mindre komplicerade att hantera.

I de senare användartesterna var resultaten ganska lika, alltså att de fysiska korten var mer stressframkallande än de digitala. Användarna påpekade att det var tidskrävande och mer utmanande att hantera och spela med de fysiska korten, men trots den ökade stressen tyckte vissa att spelet var roligare och mer engagerande med de fysiska korten. Stressen med de fysiska korten berodde alltså på behovet av att blanda och hitta rätt kort, lägga ner dem med baksidan upp och behöva trycka på Spawn-knappen, till skillnad från de digitala korten där man bara behövde klicka en gång. Timern upplevdes som lagom lång i båda användartesterna.

Balanseringen mellan spelobjekt upplevdes okej för det mesta. Under de första användartesterna tyckte de flesta spelarna att det var dålig balansering mellan olika monster, men till de senare användar-

testerna tyckte spelarna att både monster-monster balanseringen och monster-torn balanseringen var helt okej eller bra. Det var några kommentarer om specifika spelobjekt som spelarna tyckte var för stark eller för svaga, men det var också många som upplevde att de inte kunde kommentera på balanseringen eftersom de inte hade tid att kolla vilka monsterkort de spelade utan bara la ner ett kort för att hinna attackera/försvara så fort som möjligt.

I övrigt så tyckte spelarna att det var ett roligt spel (förutom buggarna under det första användartestet). Det fanns några kommentarer på hur spelet hade kunnat förbättras på andra fronter, men dessa kommentarer är inte relevanta till undersökningen i denna rapport.

Kapitel 5

Analys och diskussion

I detta kapitel kommer en analys och diskussion om implementationen och resultatet, samt en etisk och samhällsreflektion att hållas.

5.1 Implementation och metod

I kapitel 3 förklaras det hur spelets funktionalitet implementerades. En av funktionerna som implementerades men inte användes var runor. Mycket tid lades ner på denna funktion bara för att senare ta bort den. Detta var på grund av att funktionen inte passade ihop med de andra spelmekanikerna. Om projektet hade genomförts igen skulle det kunna skapas ett *Game Design Document* (GDD) som presenterat i [13] för att bestämma spelets speldesign innan utvecklingen börjat. På detta sätt skulle endast de funktioner som finns med i GDD utvecklas och på så sätt undvika att spendera tid på onödiga funktioner.

En viktig del som gruppen inte tänkte på innan spelmekaniker utvecklades var hur många steg det tog från att ett kort placerades till att ett monster spawnat. Först måste användaren välja ett kort som de vill spela, om det är en spell eller monster så kommer siktandet vara annorlunda vilket lägger till ett extra steg, sedan måste de trycka på en knapp för att spawna kortet. Detta är redan många steg, och runsystemet lade till en extra och stor del som gjorde att det tog för lång tid från att användaren skulle välja ett kort till att de fått en effekt från det kortet. Om antal steg hade diskuterats i mer detalj och dokumenterats hade det kunnat synas tidigare att runsystemet inte fick plats i systemet för kort och tiden hade kunnat spenderats på annat som är dels enklare och mer meningsfullt.

I början av projektet då gruppen planerade och tog fram idén kring projektet så fanns det stora ambitioner och förväntningar över hur spelet skulle se ut och fungera. Nu när projektet är slut så finns det funktioner som inte har hunnits implementeras. Det hade varit bra att från en början skapa en realistisk plan som är flexibel för att senare under projektets gång kunna vidareutveckla produkten.

Under början av projektet lades mycket tid på att bland annat skapa *concept art* för monster och spells. Mot slutet av projektet behövde dessa designer ändras på grund av tidsbrist, så att i stället för att varje monster skulle ha en unik 3D-modell så skapades det i stället tre olika modeller, en för varje typ av monster. För att representera vilket element ett monster har används i stället texturer. Detta innebär att all tid som lades på att skapa *concept art* inte ledde till något, eftersom den inte kunde användas. Det hade i stället varit bättre att lägga den tiden från början på att utveckla funktioner som skulle hjälpa gruppen att svara på frågeställningarna för projektet.

Som tidigare nämnt i 3.3.1 har temporära fysiska kort använts under utvecklingen för att en design för alla kort inte funnits. Detta fungerar bra men kanske inte är hållbart när de permanenta korten väl designats. Då de temporära koderna är mycket större än koderna på de permanenta korten kan

det betyda att systemet kanske kräver så stora koder för att kunna identifiera vilket ID som ligger på planen. Dessutom är de fysiska korten en central del av detta projekt, dels för att kunna spela spelet och ha användartester, men framför allt för att kunna undersöka frågeställningarna.

5.2 Resultat

5.2.1 Spel

Både gruppens och kundens vision av spelet har för mestadels uppfyllts i spelets funktion och design. Spelet har utvecklats för att likna spelet *Clash Royale* som beskrevs i 2.3, med användning av digitala kort som läggs på en spelplan för att förstöra motståndarens bas. Detta tycker gruppen har utförts väl, med tillägget av de fysiska ArUco-korten som är ett alternativ för digitala korten. Där visionen av spelet i jämförelse med resultatet skiljer sig åt är den större ambitionen som fanns för spelet, innan det sågs att det skulle krävas tid och arbete på andra delar av projektet. Ett exempel på det är på karaktärerna som skulle implementeras. Vid projektets början var tanken att det skulle finnas en karaktär som vänder sig och springer ifrån motståndare när karaktären har förlorat en viss mängd health, men idén skrotades när det insågs att mer arbete behövde läggas på *combat*-systemet.

Ett annat exempel på en del av utvecklingen som behövde minimeras var på designen av karaktärerna som används i spelet. Tanken var mer ambitiös och varje unik karaktär skulle också ha en unik *sprite*, men denna idé lades undan på grund av att mer arbete behövde läggas på kodning i stället.

5.2.2 Användartester

Användartesterna var ett värdefullt verktyg för att kunna hitta och fixa buggar i spelet samt besvara frågeställningarna. Trots att noggranna beslut hade fattats kring speldesignen för att skapa ett spel med högt tempo och hög stressnivå, var användartesterna avgörande för att utvärdera om dessa designval uppnådde sina avsedda syften. Efter användartesterna på den buggiga versionen av spelet fick gruppen en mycket god idé om de buggar som fanns i spelet, och testarna gav även sina förslag på förbättring. Gruppen kunde därför omarbete spelet och skapa en ny version som var näst intill fritt från buggar, och användartesterna som utfördes på denna version av spelet gav mycket goda och positiva resultat. De valda frågorna i enkäten visade sig vara välformulerade och användbara för att besvara våra frågeställningar.

Trots detta hade det varit mer fördelaktigt att genomföra flera interna tester först för att kunna ha upptäckt och åtgärdat buggarna innan användarna stötte på dem i testerna. Detta hade eliminerat behovet av att utföra två omgångar av användartester, vilket både hade sparat tid och också resulterat i en större mängd användbar data.

Vad som kan ses i användartesten är att de fysiska korten upplevdes som mer stressande än de digitala. Ett intressant resultat från testerna är att användare upplevde att spelet blev snabbare med de digitala korten men inte lika stressigt som med de fysiska korten. Detta betyder det som avgör stressen mellan de två kort-alternativen i spelet är användarvänligheten. De fysiska korten var inte lika användarvänliga och gjorde spelet stressigare men användarna tyckte att detta var roligare. Vanligtvis är mindre användarvänlighet en dålig sak men i detta fall bidrog det till en bättre användarupplevelse genom ökat tempo och stress hos användaren.

5.3 Källkritik

Gruppen har använt sig av vetenskapliga artiklar där det har funnits möjlighet, men för spelutveckling så finns det brister då tekniken för spel ständigt förbättras. Därför har källor så som Unitys egna dokumentation använts för att utveckla en del spelmekaniker. Unity har inga incitament att ge falsk information om hur deras funktioner går att använda då de som företag vill att användare av spelmotorn ska kunna åsamka de resultat som de vill uppnå, och därför har Unity orsak att ge klar och koncis dokumentation, utan fel och uppdaterat till nya standarder. För specifika problem med C#-scripten så har exempelvis forum, *development-loggar* eller video-guider kunnat varit till hjälp. Forum kom till hjälp då andra användare av spelmotorn har kunnat ställt de frågor som gruppen har och fått dem besvarade. Detta är till stor hjälp i mån om att projektet rör sig snabbt, och det kan ta för lång tid att ställa en fråga på ett forum för att sedan vänta på ett svar som löser just det problemet. Devloggar är bra för att ge insikt till varför vissa steg görs för olika funktioner i ett spel och är ofta bra för att ge nyckelord att undersöka vidare på. Videor på hemsidor, som exempelvis *YouTube*, har varit till hjälp för specifika problem och har ofta fungerat som inspiration för hur ett problem som projektgruppen har kan utredas.

5.4 Etisk och samhällelig reflektion

För avläsning av ArUco-koder används en kamera för att läsa av koderna. Denna kamera pekas ner på pekskärmen men kan ändå plocka upp känslig information såsom ansikten. På grund av detta är det viktigt att data från kameran hanteras korrekt för att undvika att data läcks till tredje parter. Det är viktigt att spelet inte sparar denna data och att kameran stängs ner då den inte används. Eftersom spelet bestämmer när kameran är aktiv är det upp till utvecklarna att garantera att kameran stängs ner när den borde.

Under spelets gång sparas enbart temporär data som krävs för att spelet ska köra. När spelet sedan är över återställs all tillbaks till sina standardvärden. Det sparas alltså ingen data om själva användarna, och ingen data sparas efter att matchen är slut.

Kortens design kan skapa tillgänglighetsdilemman för personer med olika funktionsnedsättningar. Korten kan exempelvis vara svåra att använda för spelare med synnedsättning eller dyslexi. Under designfasen är det viktigt att beakta dessa aspekter och försöka skapa en universell design som skulle kunna tänkas att passa till så bred publik som möjligt.

Spelet kompletteras med fysiska kort, och vid tillverkningen av dessa kort är det viktigt att beakta miljöaspekterna genom att välja material och tillverkningsprocesser som har minimal negativ påverkan på miljön. Gruppens val av programvaror och andra resurser som använts under för spelutvecklingen har alla varit kostnadsfria, och detta har dels varit för att kunna använda projektets budget till miljövänliga alternativ vid tillverkning av korten. Korten kan exempelvis tillverkas av återvunnet material i stället för plast eller andra icke-biologiska material, även om dessa alternativ är billigare.

Kapitel 6

Slutsatser

6.1 Frågeställningar

Syftet med detta projekt var att utreda hur fysiska kort, spelade på en stor pekskärm, kan ha en påverkan på spelets tempo och användarens upplevda stress, i jämförelse med digitala kort samt utreda hur användarupplevelsen påverkas av dessa två alternativ. Detta har gjorts genom att utveckla ett spel i *Unity*.

Frågeställningarna besvaras som följande:

- **Hur ska spelet designas och vilka tekniska lösningar ska användas för att öka tempot i spelet och därmed stressnivån hos spelaren?**

Genom att designa spelet så att användaren behöver välja rätt kort, lägga det upp och ner på bordet, trycka på en knapp och sedan plocka bort kortet för att spela ett kort ökas användarens stressnivå då det krävs flera steg för att spela ett kort. Även genom att ha ett kostnadssystem för korten med "mana" som fylls på med en viss tidsfördröjning ökas stressen då användare behöver vänta på att få tillräckligt med mana för att kunna spela ett kort. Genom att också ha olika kort som kostar mer eller mindre beroende på styrkan hos det monster som kortet representerar kan användare använda olika speltaktiker, som kräver att de behöver tänka efter vilka kort de spelar för att kunna vinna en spelomgång. Dessutom behöver de göra allt detta under de korta och tidsbegränsade spelrundorna som detta spel har. Till sist har element såsom visuell återkoppling på spelobjekts hälsa genom *healthbars*, stressande musik, tickande timer när spelet går mot sitt slut och liknande rapporterats ge en ökad känsla av stress hos spelaren. Allt detta kommer ge ett spel med högt tempo och stressade spelare som behöva väga många olika beslut snabbt för att vinna. Att dessa designval bidrar med ett högt tempo och ökad stressnivå har visats genom användartesterna.

- **Hur kan en pekskärm och fysiska ArUco-markerade spelkort bidra till det upplevda tempot och stressnivån hos spelaren?**

För att besvara den andra frågeställningen behövde användartesten utföras. Dessa visade att tempot och stressnivån hos spelaren ökade då en pekskärm och fysiska spelkort användes. Detta var på grund av att pekskärmen och de fysiska korten ökade antalet steg som krävdes för att användaren skulle kunna göra ett drag i spelet. Det krävdes att användaren läste av sina kort, placerade ett på spelplanen, tryckte på pekskärmen och sedan plockade bort kortet från skärmen.

- **Hur påverkas användarupplevelsen av att spela samma digitala kortspel med spelkort i verkligheten (fysiska) gentemot spelkort i spelet (digitala)?**

För att besvara den tredje och sista frågeställningen behövde även här användartesten utföras. Testen visade att användare upplevde användarupplevelsen som bättre då fysiska kort användes i stället för digitala. Detta var för att de tyckte att den ökade stressnivån som de fick av att spela med de fysiska korten gjorde spelet roligare.

6.2 Vidareutveckling av systemet

I projektet så ledde tidsbrist till att en del lösningar till problem inte blev som planerat. En av dessa problem som skulle vara bra att lösa om projektet vidareutvecklades är att spelaren inte kan ha mer än tre kort på handen när det gäller de digitala spelkort. Implementationen av fler kort hann inte skapas och eftersom projektgruppen ville jämföra digitala och fysiska kort så spelades spelet endast med tre fysiska kort på handen för att göra de två kort alternativen så lika som möjligt. Det resultat som användartesterna gav var då att de fysiska korten var stressigare att använda, men detta kan vara eftersom spelaren ständigt behövde tänka på att dra ett nytt kort efter de spelat ett kort samt att de behövde ta bort det från spelplanen. Detta gjordes automatiskt med de digitala korten. Detta la till extra steg för de fysiska korten och gav spelaren lite tid att tänka på deras korts möjligheter. Med fler kort på handen, till exempel trettio, och inga högar att dra nya kort från så kanske spelet blir mindre stressigt med de fysiska korten.

Litteraturförteckning

- [1] Blizzard. Hearthstone. <https://hearthstone.blizzard.com/en-us>. hämtad: 2024-05-30.
- [2] M Csikszentmihalyi. Beyond boredom and anxiety. Jossey-Bass., 2000.
- [3] Riot Games. Legends of runeterra. <https://playruneterra.com/en-us/>. hämtad: 2024-05-30.
- [4] Steven C. Gold and Joseph Wolfe. The validity and effectiveness of a business game beta test. *Simulation & Gaming*, 43(4):481–505, 2012.
- [5] Stephen Gormanley. Audio immersion in games—a case study using an online game with background music and sound effects. *The Computer Games Journal*, 2:103–124, 2013.
- [6] Sylvie Hébert, Renée Béland, Odrée Dionne-Fournelle, Martine Crête, and Sonia J Lupien. Physiological stress response to video-game playing: the contribution of built-in music. *Life sciences*, 76(20):2371–2380, 2005.
- [7] J. Hocking. *Unity in Action, Third Edition: Multiplatform Game Development in C#*. Manning, 2022.
- [8] Mantovani F Pallavicini F, Pepe A. Commercial off-the-shelf video games for reducing stress and anxiety: Systematic review. URL:<https://mental.jmir.org/2021/8/e28150>, Aug 2021.
- [9] Peter Peerdeman. Sound and music in games. *Amsterdam: VrijeUniversiteit*, pages 2–3, 2010.
- [10] Manuel Rauber. Merge conflicts in unity - how to avoid them? <https://manuel-rauber.com/2023/01/25/merge-conflicts-in-unity-how-to-avoid-them>. hämtad: 2024-05-30.
- [11] CD Projekt RED. Gwent. <https://www.playgwent.com/en/>. hämtad: 2024-05-30.
- [12] Roystan. Toon shader. <https://roystan.net/articles/toon-shader>. hämtad: 2024-05-30.
- [13] Mario Gonzalez Salazar, Hugo A Mitre, Cuauhtémoc Lemus Olalde, and José Luis González Sánchez. Proposal of game design document from software engineering requirements perspective. In *2012 17th International Conference on Computer Games (CGAMES)*, pages 81–85. IEEE, 2012.
- [14] R Schouten, Omür Arslan, and Aykut Isleyen. Accuracy of single camera pose estimation with aruco fiducial markers. 2022.
- [15] Katrin Starcke and Matthias Brand. Decision making under stress: A selective review, 2012.

- [16] Paper Plane Tools. Opencv plus unity. <https://assetstore.unity.com/packages/tools/integration/opencv-plus-unity-85928>. hämtad: 2024-05-30.

Bilaga A

Reflektion över systemutvecklingsprocessen

Om vi jämför hur den faktiska systemutvecklingsprocessen har varit i jämförelse med vår initiala planering av den tycker vi i helhet att det har gått bra och för det mesta som vi planerade. Vi genomförde vårt projekt med en balanserad användning av Scrum-metoden. Vår projektplanering inkluderade dagliga Standup-möten som skulle vara ca 15 minuter långa, ett sprintplaneringsmöte i början av varje sprint, en sprintgenomgång i slutet av varje sprint tillsammans med kunden, samt en sprintutvärdering. När vi väl satte igång med sprintarna märkte vi, på grund av projektets storlek, att dagliga möten ibland var onödiga och vi gick i stället över till att ha 2-3 möten i veckan. Detta fungerade bra, förutom att vi sällan höll det inom 15-minuter. Detta var både en bra och dålig grej; bra då vi kunde diskutera vårt arbete och ta hjälp av varandra då ingen var expert på sin arbetsuppgift och oftast behövde rådgivning, men det kunde även vara negativt då tanken var att mötena skulle vara korta när de egentligen tog upp till en timme de flesta gångerna. De andra mötesprinciperna följde vi som vi planerade, förutom att vi inte hade ett specifikt sprintutvärderingsmöte utan oftast diskuterade det som detta möte skulle behandla under sprintgenomgången. Vi har också inte haft lika många kundmöten som tänkt, men detta har mestadels varit på grund av att vi inte alltid haft stora visuella framsteg i spelet efter varje sprint. Därav hade vi kundmöten när vi kände att det fanns något nytt att visa upp i stället för att göra det i slutet på varje sprint.

Dessa modifieringar av Scrum-metoden gjorde vi då det passade vårt specifika projekt och den nivån som både projektet och gruppmedlemmarna var på, och var ett väldigt bra beslut som hjälpte oss. Att använda Scrum gav oss struktur, vilket var värdefullt eftersom detta var vår första erfarenhet av att använda en systemutvecklingsmetodik och hålla på med ett riktigt systemutvecklingsprojekt. Genom att regelbundet genomföra sprintmöten och följa upp våra aktiviteter kunde vi hålla oss på rätt spår och göra justeringar där det behövdes. Vi var alla flexibla under utvecklingsgången, vilket Scrum tillät oss att vara och vilket också var en bra grej med tanke på ett av de större problemen vi hade under utvecklingsgången. Vid varje planeringsmöte fördelade vi jämnt ut inlägg från sprintbackloggen inom teamet, men det blev ofta att vissa uppgifter var mer omfattande än andra vilket resulterade i en ojämn arbetsbelastning för vissa gruppmedlemmar. Vissa behövde mycket längre tid än tänkt för att göra sina uppgifter, medan andra blev klara snabbt och blev utan arbetsuppgifter. Detta märktes av ganska sent så detta problem blev olöst, men tack vare att gruppmedlemmarna var flexibla hoppade man in där det behövdes och hjälpte till så att allt blev gjort och så arbetsbelastningen på enstaka personer inte blev för hög. Att dela upp inläggen från sprintbackloggen så att varje inlägg hade ungefär samma arbetsbelastning hade varit det vi hade gjort annorlunda om vi fick göra om det.

I de första sprintarna märkte vi också av att många inlägg inte var specifika nog med den arbetsuppgiften de beskrev. Ett exempel på detta är ett inlägg från sprintbackloggen för den första sprinten som hette ”Implementera spelmekanik”, vilket inte var specifikt nog och då det inte fanns en tydlig definition på när detta ansågs vara färdig implementerat kunde vi inte bocka av den. Detta problem var dock något vi identifierade, och under resterande sprintar var vi mer specifika när vi gjorde nya

inlägg i backloggen.

De verktyg som vi planerade att använda för systemutvecklingen var spelmotorn Unity, IDEn Visual Studio, versionshanteringsverktyget Git tillsammans med den grafiska Git-klienten Fork, samt projekthanteringsverktyget Jira Software. Förutom några enstaka fall där vi fick problem med Git och versionshanteringen (som var ganska enkla att lösa) har alla dessa verktyg fungerat väldigt bra och förenklat den faktiska utvecklingsprocessen för oss i gruppen, då de flesta av dessa verktyg har varit verktyg som gruppmedlemmarna har haft vana vid sen innan. Jira hjälpte oss att använda Scrum på ett effektivt sätt så det fanns mallar för sprintar, backloggar med mera. Att använda Unity som spelmotor var ett mycket bra val, då vi tog mycket hjälp av online-resurser såsom till exempel YouTube-videor där folk förklarade hur man utförde vissa saker i just denna spelmotor, vilket inte hade varit möjligt i samma utsträckning om vi hade valt en annan spelmotor som inte var lika populär som Unity är. Vi anser därför att dessa val av verktyg varit ideala för oss.

A.1 Armen Abedi

Min roll i projektet har varit som utvecklare. Jag har arbetat med databasen, ArUco och kamerasytemet samt mana-systemet. För mig har det varit en rimlig arbetsbelastning och det har varit bra att arbeta på olika delar av koden. Vid projektets början var jag endast satt på ArUco och kamera-systemet, men efter några sprintar behövdes arbetet fördelas till andra områden, och då bytte jag över till databas och mana-system.

Användningen av Scrum gick bra till mestadels, det har varit ett väldigt bra sätt att strukturera upp vad som ska göras och när, samt att delegera arbete när det behövs. Den nackdelen som jag kände från Scrum låg mest på okunnigheten då det var gruppens första användning av metoden, och därför blev det stressigt när det till exempel behövdes byta över utvecklare till andra kod områden.

A.2 Emil Larsgärde

Mina roller i detta projekt har varit Scrum-mästare samt utvecklare. Som Scrum-mästare har jag ansvarat om gruppens produktivitet genom att använda ramverket för scrum. Som Scrum-mästare har jag agerat ordförande i samtliga möten och säkerställt att alla Scrumaktiviteter utförs. I utvecklings-teamet har jag arbetat med databasen, ArUco och kamerasytemet, mana-systemet och hjälpt till att koppla samman alla delar in i *spawn*-funktionen.

Då detta för första gången för samtliga med arbetsmetoden Scrum så har det varit en inlärningsprocess för samtliga för att förstå och uppfatta arbetsprocessen. Jag tycker att med tiden har Scrum fungerat bättre och bättre då alla anpassar sig mer och mer till arbetsmetoden. Till en början var alla aktiviteter lite för diffusa och det var svårt att veta vad som skulle göras för att kunna anpassa det till spelet. Vissa aktiviteter har tagit mer tid än andra (till exempel ArUco) vilket har lett till att medlemmar som är klara med sin aktivitet inte vet vad de ska göra, men jag känner också att vi alltid har haft grejer att göra såsom 3D-modellering av karaktärer och torn.

Men överlag är jag väldigt nöjd med hur vi har jobbat och nu till slutet har det fungerat riktigt bra.

A.3 Gayathri Naranath

I detta projekt har jag varit en del av spelutvecklingsteamet, alltså har jag suttit mestadels med att implementera spellogiken och spelelement i olika skript och i Unity. Detta bestämdes innan vi började

med projektarbetet och tanken var att jobba med spelutvecklingen under de flesta sprintar, och i någon sprint där det inte fanns mycket att jobba med på den fronten skulle jag i stället jobba med grafiska element som till exempel. 3D-Modellering av spelobjekten.

För de mesta har detta upplägg fungerat ganska bra. Däremot har det inte varit så lätt att dela upp arbetet som behövde göras på ett bra sätt, vilket gjorde att jag kände att det inte fanns något att göra under flera tillfällen. Vi följde Scrum-metoden där varje inlägg i sprintbackloggen för varje sprint fick en ansvarig, men flera av dessa inlägg var inte specifika nog. Detta gjorde att det blev lite förvirring ibland kring vad exakt som skulle göras för att dessa inlägg skulle kunna anses färdiga. Men när dessa aspekter har fungerat så har hela grupparbetet också fungerat väldigt bra. Att satsa på att verkligen följa Scrum-metoden i vårt arbete var ett jättebra val som enligt mig är det som gjorde att utvecklingsarbetet gick så bra.

A.4 Gustaf Kronholm

Min roll i detta projekt har varit utvecklare. Detta har huvudsakligen varit för kamerasystemet och användningen av ArUco, men jag har även hjälp till med databasen. Det har fungerat bra för mig då jag alltid haft något att göra och aldrig jobbat med denna typ av bildbehandling tidigare vilket var både kul och utmanande att testa på. Däremot hade det varit både roligt och nyttigt för min personliga utveckling som programmerare att utveckla även andra delar av projektet som till exempel spelmekaniken.

Upplägget har däremot till mestadels fungerat bra. Vi har arbetat utifrån Scrum vilket i teorin betyder att alla borde ha något att göra då vi delade upp ansvarsområden inom varje sprint för det som behövdes göras. Om man blev klar tidigare kunde man hjälpa någon annan om möjligt.

Däremot tog det ett tag att vänja sig vid detta arbetssätt och därför blev vissa stundtals utan arbete, men ju längre projektet höll på desto mer bekväma blev vi med att jobba med Scrum och arbetet flöt på bättre. Då implementeringen av ArUco tog längre tid än tänkt gick mina ansvarsområden från backloggen över mellan sprintar då de inte blev klara i tid. Därför blev det naturligt att jag fortsatte tills det var klart. Då jag vid detta stadiet hade skaffat mig mest kunskap om ArUco blev det lätt att jag hamnade på de ansvarsområdena som var kvar om ArUco.

Trots detta är jag ändå väldigt nöjd med hur gruppen har arbetat och tycker att vi har arbetat så effektivt vi kan med den kunskap och erfarenhet vi hade om Scrum som utvecklingsmetodik.

A.5 Jonatan Ebenholm

Mina roller i projektet har varit utvecklare samt produktägare. Som produktägare har jag hållit kontakt och bokat möten med kunden. Som utvecklare har jag jobbat både på grafik och spelmekaniker. Jag skapade träden och dess toon-shader samt spelplanens utseende. Några spelmekaniker jag jobbat med är runorna och spellsen.

Medan jag anser att varje sprint har fungerat bra och sprintmötena i helhet har gynnat gruppen så tycker jag i efterhand att vi borde ha markerat hur stora de individuella sprintuppgifterna var. Ett uppgift i sprinten kunde vara att skapa ett träd, vilket tog mig mindre än en dag medan runsystemet tog mig en vecka, men de såg lika stora ut i Jira där vi planerade sprinten. Detta ledde till att jag tog stora tasks utan att förbereda mig för att de kunde ta veckor, och tog på mig dem själv, fast än de kanske hade lämpat sig bättre till att göras av två utvecklare.

Jag tycker också att ju längre in i projektet vi kom, ju bättre jobbade vi med Scrum metodiken och

började prioritera de viktigare delarna för att uppnå projektmålen. Att ha fysiska standup möten några gånger i veckan har enligt mig varit en succé då projektet samt arbetssätten kan diskuteras och ändras på vid behov under sprintens gång.

A.6 Ludwig Boge

Min roll i detta projekt har varit som utvecklare. Jag har arbetat på många olika delar av projektet efter behov. Detta har fungerat bra för mig då jag känt att jag alltid har haft något att jobba på. Däremot har det fungerat mindre bra då det känts som att man måste själv hitta saker som behöver göras då det ibland inte fanns tillräckligt med saker i sprintbackloggen. En förbättring vore att kontinuerligare under sprinten uppdatera sprintbackloggen när saker blev klara för att undvika att folk inte har något att göra. De dagliga scrummöten har fungerat bra då det gett mycket transparens mellan alla i projektet men det fanns saker som var mindre bra med dem. Detta var att det lätt blev att mötena gick över en kvart då folk visade vad dem hade gjort eller fastnat med. I stället för att gå in och undersöka kod på dessa möten skulle ett annat möte direkt efter ha bokats med de relevanta deltagarna.

A.7 Mirijam Björn

Min roll i projektet var utvecklare, och jag var del av grafik teamet. Under början av projektet skapade jag *concept art* för spelplanen, kortens design, spells, och monster. All *concept art* avslutades dock inte eftersom gruppen insåg att mer fokus behövde läggas på koden. Jag gick då över till att skapa en funktion för att *spawn* monster. Mot slutet av projektet gick jag tillbaka mer till grafiken och jobbade med 3D-modellering.

Jag tycker att Scrum-mötena har fungerat bra och har hjälp till att få en helhets blick över hela spelet och vad alla jobbat med. Produktbackloggen underlättade att dela upp arbetsuppgifter mellan gruppmedlemmarna, fast vissa av inläggen var ganska breda och hade kunnat delas upp i mindre bitar. Detta hade gjort att arbetet hade kunnat fördelas mer jämnt i gruppen. Det hade även varit bra att lägga mindre fokus i början av projektet på *concept arten*. Jag lade ner ganska mycket tid på den eftersom jag tänkte att den skulle underlätta när 3D-modellerna skulle skapas, men på grund av tidsbrist behövde designerna för monsterna förenklas. Det hade i stället varit bättre att lägga den tiden på koden för spelet och sedan mot slutet av projektet skapat *concept art*, eftersom vi då hade kunnat undvika att slösa tid på något som sedan inte användes.

Bilaga B

Individuella bidrag

En lista över vilka som varit involverade i projektet, inom teamet och utanför, med en beskrivning av någonting som de, genom individuellt, självständigt arbete, bidragit med.

B.1 Armen Abedi

- Avläsning av ArUco-markörer.
- Grundimplementering av Mana-System och Mana-UI.
- Databasstruktur och kort-information.

B.2 Emil Larsgärde

- Databas och databasstruktur för kort-informationen
- Färdigställt mana hantering och mana UI
- Scrummästare

B.3 Gayathri Naranath

- Spellogik för detektion och attack mellan karaktärer och torn.
- Infrastruktur för varierande attackstyrka beroende på båda karaktärers element.
- Animationer och animationskontroller.
- Spelobjektens livmätare och livmätarnas grafiska utseende.

B.4 Gustaf Kronholm

- Avläsning av ArUco-Markörer.
- Kalibreringsfunktion för att ställa in spelplanens storlek.

B.5 Jonatan Ebenholm

- Produktägare.
- Runsystemet.
- Grafik för spelplanen.
- Spells.
- Toon-Shadern.
- Touch grafiker.
- Minimap för att sikta med spells.

B.6 Ludwig Boge

- Rörelsefunktionalitet för monster
- Refaktorering och omskrivning av rörelse och attack-logik
- Digitala kort
- Startmeny och slutmeny + vinst-krav

B.7 Mirijam Björn

- *Concept art* för spelplan, kort, spells, och monster.
- Funktion för att *spawna* monster med rätt data från databasen.
- 3D-modell för wyverns.

B.8 Karljohan Lundin Palmerius

Kund för projektet.

B.9 Daniel Jönsson

Examinator för kursen i kandidatarbetet.

Bilaga C

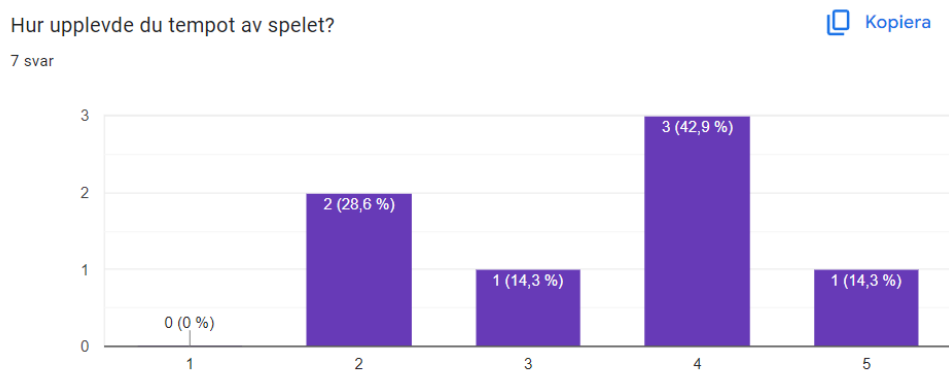
Resultat från användartester

I denna bilaga presenteras svaren från de två användartester som utfördes under projektet.

Frågeformulären för de två testen innehöll samma frågor förutom en fråga om stress relaterat till timern som lades till för andra testet.

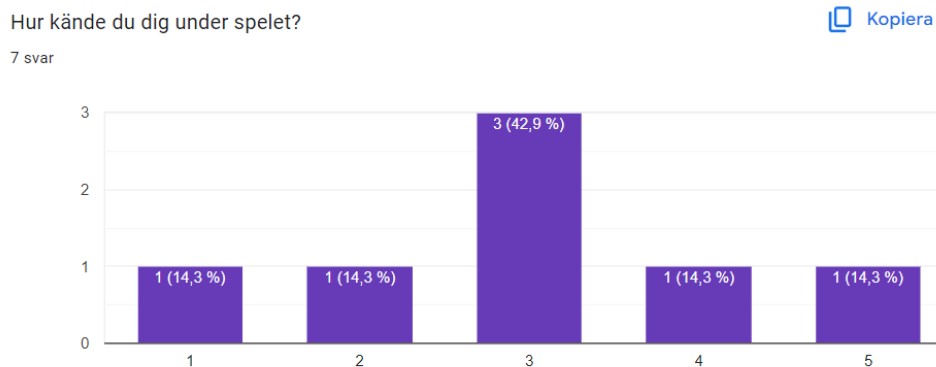
C.1 Användartest 1

Fråga 1: Hur upplevde du tempot av spelet?



Figur C.1: 1:Långsamt/Lågt tempo, 5:Snabbt/Högt tempo

Fråga 2: Hur kände du dig under spelet?



Figur C.2: 1:Mycket lugn, 5:Mycket stressad

Fråga 3: Om du kände dig stressad, vad var det som bidrog till den känslan?

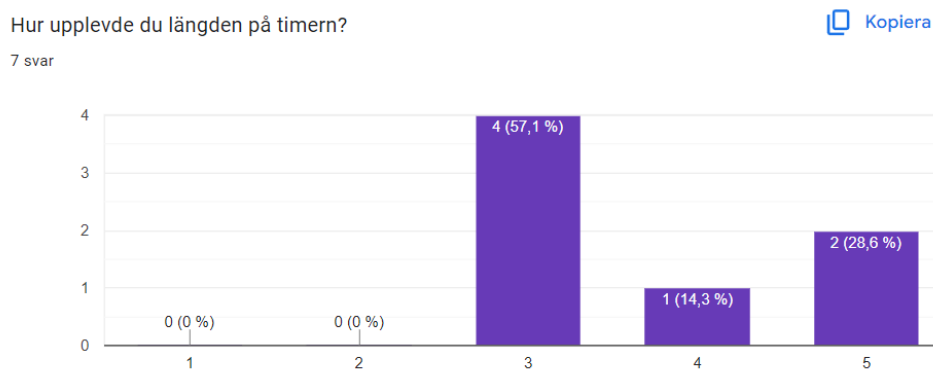
De flesta av svaren säger att den stress de kände kom från de buggar som fanns under testerna. Andra svar nämnde att de upplevde stress över att de kände att de behövde vara snabbare än motståndaren, att de behövde vara uppmärksamma över sin mana för att kunna spela kort så snabbt som möjligt, samt när de blev attackerade.

Fråga 4: Kände du dig mer stressad när du använde de fysiska korten eller de digitala? Varför?

Majoriteten av svaren anger att de upplevde mer stress av de fysiska korten, eftersom det krävdes fler steg för att spela dem. Det nämndes även att de fysiska korten gav mer stress eftersom de inte alltid fungerade som de skulle. En person svarade att de upplevde mer stress från de digitala korten eftersom de fick spelet att gå snabbare.

Fråga 5: Var de fysiska spelkorterna enklare att använda än de digitala? Vad kan förbättras?

Alla svaren, utom ett som var orelaterat till frågan, anger att de tyckte de digitala korten var lättare att använda. De anledningar som gavs varför de digitala korten upplevdes som lättare att använda var att det var lättare att läsa av vad korten gjorde och vad de kostade. Flera spelare svarade även att det hade varit bättre om kortens ArUco-kod var på samma sida som kortets "stats". Vissa tyckte även att de digitala korten var bättre eftersom det fanns buggar som gjorde att de fysiska korten inte alltid fungerade som de skulle.

Fråga 6: Hur upplevde du längden på timern?

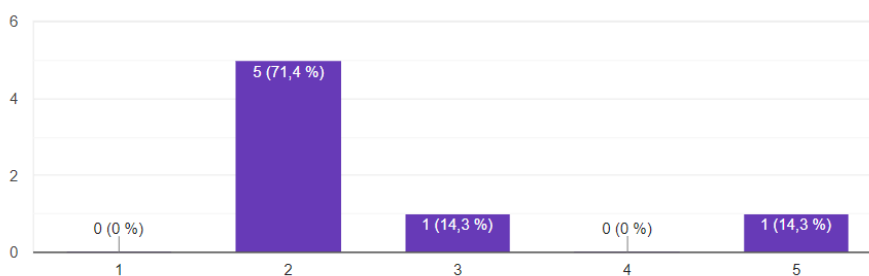
Figur C.3: 1:För kort, 5:För lång

Fråga 7: Hur upplevde du balanseringen för monster vs monster?

Hur upplevde du balanseringen för monster vs monster?

 Kopiera

7 svar



Figur C.4: 1:Mycket obalanserad, 5:Bra balanserad

Fråga 8: Om du tyckte det var obalanserat, på vilket sätt?

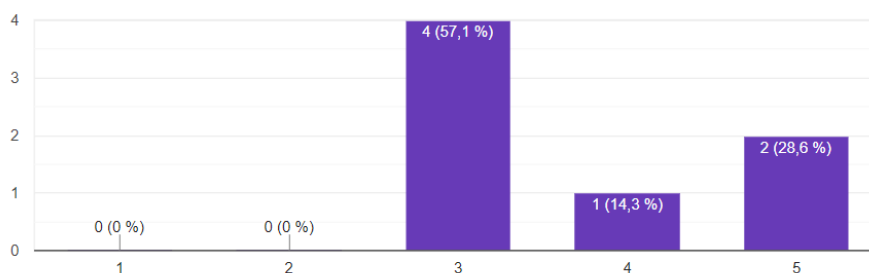
Flera personer svarade att de tyckte att de billigare monstren var för starka i förhållande till vad de kostade att spela. En person svarade även att det dyraste monstret (Crusher) var alldeles för starkt jämfört med de andra korten. Spelets buggar nämndes även som en anledning.

Fråga 9: Hur upplevde du balanseringen för monster vs torn?

Hur upplevde du balanseringen för monster vs torn?

 Kopiera

7 svar



Figur C.5: 1:Mycket obalanserad, 5:Bra balanserad

Fråga 10: Om du tyckte det var obalanserat, på vilket sätt?

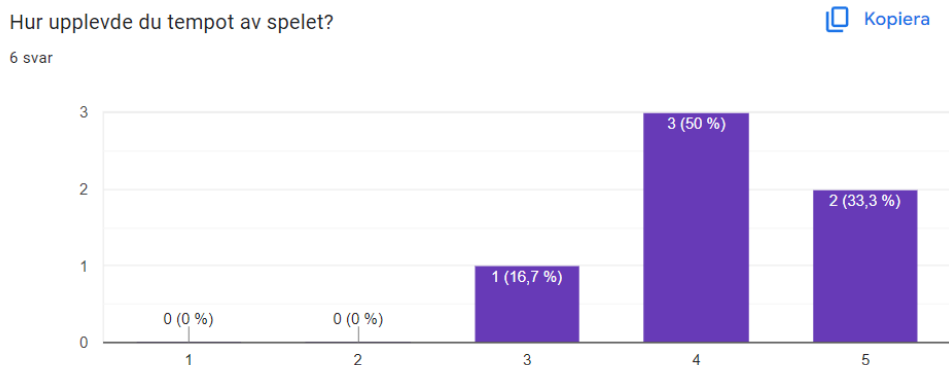
Spelets buggar anges som anledning. En person svarade även att de inte märkte att tornen attackerade monster.

Fråga 11: Övriga åsikter?

Över lag svarade de som testade spelet att de tyckte det var roligt och hade god potential. Flera personer gav även förslag på hur spelet skulle kunna förbättras. En person skrev att de hade velat ha musik i spelet. En person skrev att det hade varit bra om texten som säger vem som vann dök upp på kortsidorna riktat mot spelarna så att det blev lättare för spelarna att läsa. En annan person svarade att det hade underlättat om det fanns något sätt att urskilja vilka monster som hörde till vilket lag, samt att det hade varit bra om det fanns ett ställe på skärmen att placera kortleken så att man inte behövde vända sig om varje gång man skulle ta ett nytt kort.

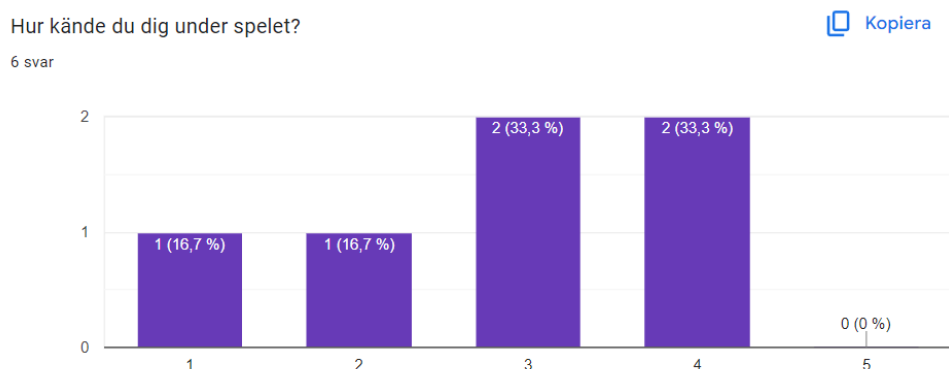
C.2 Användartest 2

Fråga 1: Hur upplevde du tempot av spelet?



Figur C.6: 1:Långsamt/Lågt tempo, 5:Snabbt/Högt tempo

Fråga 2: Hur kände du dig under spelet?



Figur C.7: 1:Mycket lugn, 5:Mycket stressad

Fråga 3: Om du kände dig stressad, vad var det som bidrog till den känslan?

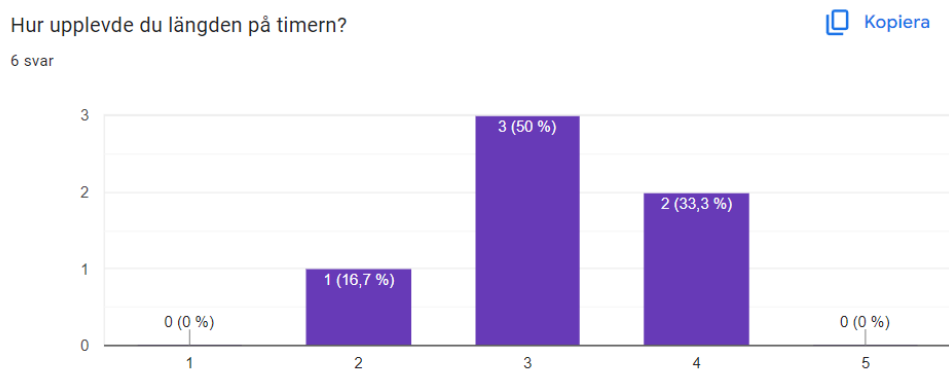
De som kände stress upplevde den främst från spelets höga tempo samt att de kände att de inte ville slösa mana, vilket gjorde att de försökte spela kort så snabbt det gick. En person kände även stress över att de fysiska korten inte alltid lästes av korrekt.

Fråga 4: Kände du dig mer stressad när du använde de fysiska korten eller de digitala? Varför?

De som testade spelet svarade nästan alla att de upplevde mer stress från de fysiska korten. Detta var främst eftersom det krävdes fler steg för att få ut monster i spelet och att de fysiskt behövde flytta korten fram och tillbaka. En person svarade att de inte upplevde någon stress, men de tyckte ändå att de fysiska korten bidrog till ett högre tempo.

Fråga 5: Var de fysiska spelkorterna enklare att använda än de digitala? Vad kan förbättras?

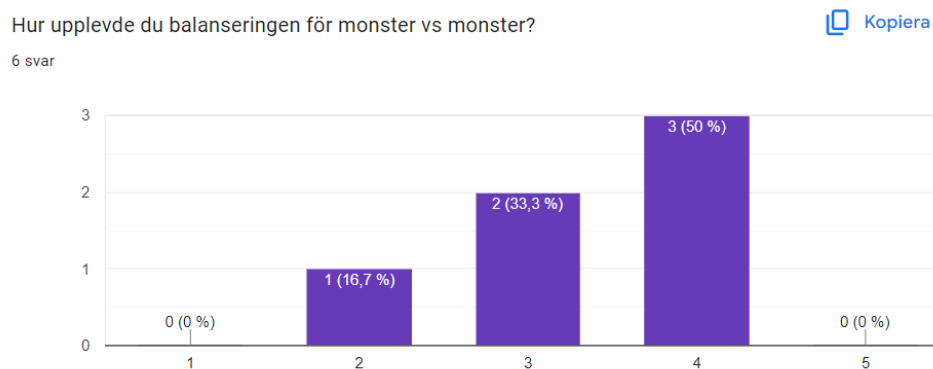
Majoriteten svarade att de tyckte att de digitala korten var lättare att använda eftersom de krävde färre steg för att spela. Några tyckte även att de digitala korten var enklare att använda eftersom de fysiska ibland inte fungerade korrekt. En person svarade att de tyckte att de digitala och de fysiska korten var lika lätta att använda.

Fråga 6: Hur upplevde du längden på timern?

Figur C.8: 1:För kort, 5:För lång

Fråga 7: Upplevde du någon stress/tidspress från timern? Om ja, på vilket sätt?

Ingen av de som svarade upplevde att timern bidrog till stress.

Fråga 8: Hur upplevde du balanseringen för monster vs monster?

Figur C.9: 1:Mycket obalanserad, 5:Bra balanserad

Fråga 9: Om du tyckte det var obalanserat, på vilket sätt?

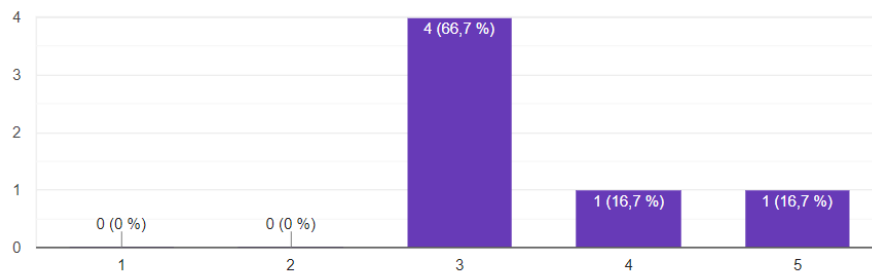
Flera personer svarade att det dyraste monstret (Crushers) behövde ändras. De tyckte både att den hade för lite health och därför dog för snabbt, samt att den gjorde för mycket skada till andra monster. Svaren var väldigt delade, där vissa tyckte att Crushers var för starka och andra tyckte att de inte var tillräckligt starka.

Fråga 10: Hur upplevde du balanseringen för monster vs torn?

Hur upplevde du balanseringen för monster vs torn?

 Kopiera

6 svar



Figur C.10: 1:Mycket obalanserad, 5:Bra balanserad

Fråga 11: Om du tyckte det var obalanserat, på vilket sätt?

En person svarade att de tyckte att tornen borde kunna attackera monster längre bort. En annan person svarade att de tyckte tornen hade för lite health och dog för snabbt.

Fråga 12: Övriga åsikter?

De flesta som svarade tyckte att spelet var kul. De som hade varit med under det första användartestet tyckte även att spelet hade förbättrats sen dess. En av testpersonerna tyckte att det var bra att "manabaren" hade flyttats till det nedre vänstra hörnet, vilket gjorde den lättare att se. De tyckte dock att den borde flyttas till höger sida när de fysiska korten används så att den är närmre spawn-knappen. De tyckte även att det var bra att musik hade lagts till. En person tyckte att de fick tillbaka mana för snabbt och det hade varit bättre om det i stället var halva hastigheten. De skrev även att de hade fördragit om de fysiska korten hade ArUco-koden och monstrets "stats" på samma sida, samt att de hade velat ha fler "stats" synliga på kortet. En person svarade igen att de tycker att det hade varit bättre om det fanns ett område på skärmen att placera korten så att man inte behövde vända sig om till ett bord hela tiden. Till sist svarade en person att en bug som gör att två kort med samma id inte kan spelas efter varandra behöver fixas.