

Projekt Brætspil

Udviklingen af Territory Takeover



Uddannelsessted: Vid Gymnasier HTX Grenaa

Vejledere: Kasper Skov Christensen, Magnus Håkon Petersen

Fag: Digitalt Design og Udvikling (DDU)

13/03/23 - 25/04/23

Anders M.M.J. Larsen

Astrid T. Petersen

Mikkel A. Erikstrup

Mikkel R. Blom

Rasmus A. Riis

Abstract

This is a paper about the process of developing an immersive boardgame. The boardgame's physical representation is significantly different than traditional boardgames. Since the boardgame's focus is on immersion, it contains built-in music, lights, and a three-dimensional miniature landscape. Because the boardgame contains these features for improving immersion it is not possible for it to be folded up and stored in a box on the shelf like most traditional boardgames. The board is a stationary table that is meant to be played while standing up. By reading this paper you will gain an understanding of the development process of the boardgame "Territory Takeover". The process spans everything from the idea generation process to the evaluation and conclusion of the project. The group experienced some hiccups during development, primarily regarding time constraints, but was in the end able to produce a product that fulfilled its intended purpose.

Indholdsfortegnelse

Indledning	1
Projektstyring	2
Hvad er SCRUM?	2
Hvordan er SCRUM brugt?	4
Discord til kommunikation	4
Problemformulering	6
Problemanalyse	6
Kravspecifikationer	7
Opgavens krav	7
Gruppens krav	7
Gruppens ønsker	7
Design	9
Idégenerering	9
Forberedt brainstorm	9
Skattespillet	9
Totemspillet	10
Kooperativ lydoplevelse	10
Valg af idé og problemformulering	10
Skift af idé	11
Spildesign	12
Kortdesign	13
Overvejelser ift. spildesign	14
Lyddesign	15
Produktudvikling	16
Produktion af bordet	16
Produktion af landskabet	17
Udformning af flamingolandskabet	17
Produktion af teksturer samt miniaturemodeller	17
Detaljering af landskabet	22
Produktion af brættet	25
Produktion af felter	25
Opsætning af elektriske komponenter	29
Produktion af spillerbrikker	43
Produktion af kort	45
Produktion af lyd	46

Test af spil	48
Første test af skattespillet.....	48
Anden test af skattespillet	50
Quick and dirty test.....	50
Test af Territory Takeover.....	50
Evaluering.....	52
Diskussion	52
Konklusion.....	53
Bibliografi	54
Bilag.....	56

Indledning

Videospil er efterhånden blevet kendt som et af de bedste medier, når det gælder om at skabe fordybelse og indlevelse hos forbrugeren. Én af de klare fordele ift. mere traditionelle interaktive formater som brætspil er det væld af muligheder, de elektroniske elementer medbringer. Desuden er brætspillet også bundet af en kasse, som skal kunne indeholde hele spillet. Er det muligt at skabe en mere interaktiv oplevelse i et brætspil ved at bruge digitale eller elektroniske interaktionselementer? Og hvad hvis brætspillet omtænkes som en stationær oplevelse, der ikke længere er begrænset af spilkassens størrelse?

I dette projekt har der været fokus på netop det. Navnlig at udvikle et brætspil der fremmer indlevelsen hos spilleren, ved bl.a. at gøre brug af digitale og elektroniske elementer. Der er blevet konstrueret et bord med en interaktiv spilleplade der giver spilleren feedback i form af lys samtidig med at trække spillerens bevidsthed ind i spilverdenen ved brug af lyd og en tredimensionel udsmykning af selve spillerpladen. I resten af rapporten beskrives arbejdet og metoderne brugt til udformningen af produktet.

Projektstyring

I starten af projektet blev et enkelt gruppemedlem valgt som projektleder over hele projektperioden. Projektlederen har ansvaret for at holde overblik over processen og sikre at alle i gruppen kan bidrage produktivt til projektet. I gruppen var det vigtigt at projektlederen ikke bestemte over de andre gruppemedlemmer eller havde fuld kontrol over projektet. Projektlederens rolle var udelukkende at holde overblik og lede projektet, ikke styre det. Hvis der skulle forekomme situationer hvor gruppens medlemmer ikke kunne enes over et projektrelateret problem eller beslutning ville det falde til projektlederen at tage den endelige beslutning.

Hvad er SCRUM?

Til at hjælpe projektlederen til at holde overblik blev projektstyringsmetoden SCRUM brugt. SCRUM er defineret som følgende: "*Scrum is a lightweight framework that helps people, teams and organizations generate value through adaptive solutions for complex problems.*" (Schwaber & Sutherland, 2023). SCRUM kan også forklares som en metode der deler et projekt op i mindre bidder, som så efterfølgende kan planlægges, udføres og evalueres (Schwaber & Sutherland, 2023).

SCRUM er udviklet ud fra både empirisme og Lean-tankegangen (Schwaber & Sutherland, 2023). Empirisme er den filosofiske tankegang hvori alt viden om virkeligheden opleves igennem sanserne (Holtug, 2023). Med andre ord handler empirisme om at opnå viden gennem observationer. Lean tankegangen handler om optimering. Den består af konstant eksperimentering og innovation af f.eks. produktet til at skabe en bedre, hurtigere og billigere måde at gøre det på, som også reducerer mængden af spildmateriale. Lean-tankegangen er derfor rigtig god til at tilpasse sig det konstant ændrende arbejdsmiljø (Lean Enterprise Institute, 2023). Derfor er SCRUM altså bygget på principperne om gennemsigtighed, inspektion og tilpasning.

Gennemsigtighedsprincippet er til for at sørge for at alle involveret i udviklingsprocessen kan se hvad der skal laves, om det skal laves nu og lignende informationer som kunne være relevante. I SCRUM er mange vigtige beslutninger nemlig taget på baggrund af de formelle artefakter, og hvis informationen omkring dem er begrænset, bliver beslutningen også mere risikabel, og derved måske også mindre værdifuld (Schwaber & Sutherland, 2023). Artefakterne uddybes senere i dokumentet.

Inspektionsprincippet kommer i forlængelse af gennemsigtighedsprincippet. Hvis der er dårlig gennemsigtighed, kommer der også en inspektion som kan være helt til spild eller endda misledende. Artefakterne og fremskridtet mod målene skal i SCRUM regelmæssigt inspiceres for at der kan opdages eventuelle fejl og mangler. I processen indgår der også tilpasning, hvilket forklares dybere i næste afsnit, så der kan nemt opstå fejl og mangler fra gamle idéer eller opgaver som senere er blevet ændret eller gjort overflødige (Schwaber & Sutherland, 2023).

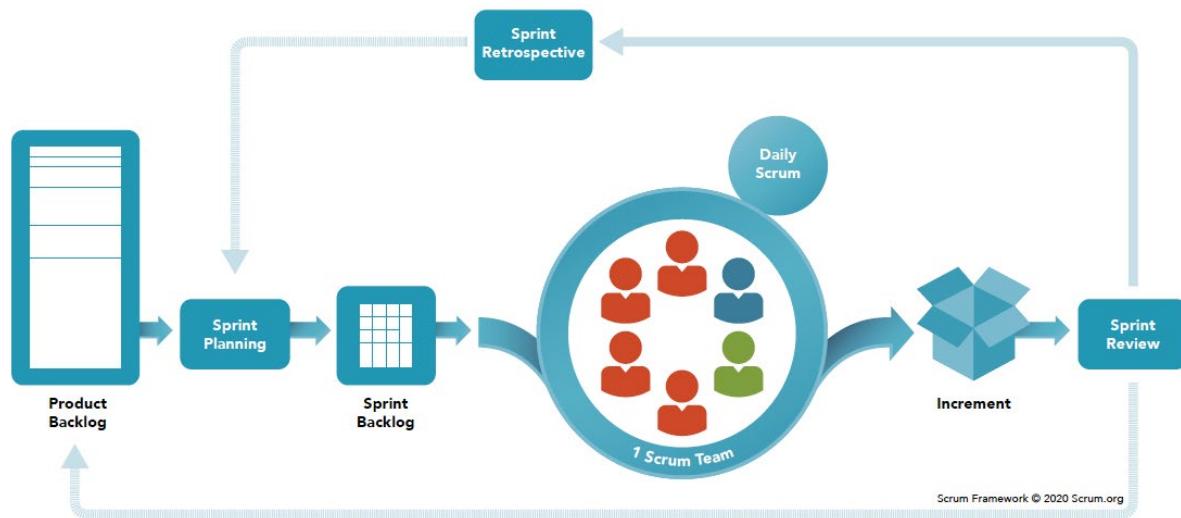
Tilpasningsprincippet et essentielt for at inspektionsprincippet har en værdi, hvis der ikke tilpasses efter en inspektion, kunne inspektionen lige så godt ikke være fundet sted. Tilpasningen er til for at rette på de ting som ikke stemmer overens med projektets mål. Tilpasningerne skal helst forekomme hurtigst muligt for at mindske afvigelsen mest muligt, hvilket også er derfor regelmæssige inspektioner er at foretrække (Schwaber & Sutherland, 2023).

De tre artefakter i SCRUM er følgende; product-backlog, sprint-backlog og inkrement. Product-backloggen indeholder alt det som produktet skal indeholde for at det er færdigt. Det kan beskrives som en slags oversigt over arbejdsopgaver der skal laves for at produktet er færdigt. Sprint-backloggen minder meget om product-backloggen, med den ene forskel at den kun udspænder sig over en kort periode defineret af sprintet og at den også indeholder en plan for hvordan opgaverne skal løses. Sprintfasen uddybes i et senere afsnit. Det sidste artefakt kaldes for inkrement, og

indebærer at der er sket et betydeligt fremskridt mod færdiggørelsen af produktet (Schwaber & Sutherland, 2023).

SCRUM er delt op i 5 faser. Faserne er sprint, sprintplanlægning, SCRUM-møde, sprintvurdering og sprintovervejelser. Sprintet er omdrejningspunktet - det er i sprintet alt arbejdet foregår, og hvor de gældende opgaver for projektet befinner sig. Det er også i sprintet de resterende faser finder sted. Der er mange sprints i et projekt; når ét sprint er ovre starter et nyt ét med det samme. Dette gøres af hensyn til fleksibilitet så der nemt kan laves ændringer, rettelser og tilføjelser til produktet. Et sprint har en fast tidsperiode aftalt på forhånd for at sikre struktur og konsistens. Tidsperioden kan variere fra projekt til projekt, men i samme projekt skal alle sprints helst være af samme længde (Schwaber & Sutherland, 2023).

Sprintplanlægningen er meget selvbeskrivende, det er i denne fase sprintet planlægges med hvilke opgaver der skal løses. SCRUM-mødet er til for at opdatere hele holdet på fremskridtet og hvad planen fremadrettet er. Det er forslægt at holde mødet dagligt, dog kan det i en undervisningssituation være svært da der ikke nødvendigvis arbejdes på projektet dagligt. I de situationer kan der holdes møder, hver dag der arbejdes på projektet. Sprintvurderingen finder sted tæt på slutningen af sprintet og det er her SCRUM-teamet og interesserterne vurderer arbejdet der er blevet udført i sprintet samt evt. ændringer i produktets miljø. Det allersidste i sprintet er sprintovervejelser. Det er her selve sprintet og arbejdsmetoderne analyseres og der bliver foretaget eventuelle ændringer for at øge effektivitet og kvalitet i arbejdet og produktet (Schwaber & Sutherland, 2023). En visuel repræsentation af SCRUM-faserne kan ses på figur 1.



Figur 1 - Model over et SCRUM forløb (scrum.org, 2023)

I et SCRUM-team er der tre roller; en product owner, en SCRUM-master og udviklere. Product ownerens opgave er bl.a. at maksimere værdien af produktet, opsætte målet for produktet samt at lave en product backlog. Det er også product ownerens ansvar at kommunikere deres valg klart og tydeligt ud til resten af SCRUM-teamet. SCRUM-masteren har ansvaret for at SCRUM-metoden kører som planlagt. Det kan f.eks. indebære at fungere som mellemled mellem SCRUM-teamet og evt. interesserter, samt sørge for at alle SCRUM-events forløber når de burde, og som de burde. Udviklerne i et SCRUM-team står for både hvad det gældende sprint skal indeholde, samt selve udførelsen af sprintets opgaver (Schwaber & Sutherland, 2023).

Hvordan er SCRUM brugt?

Projektgruppen har i dette projekt brugt SCRUM til at holde styr på arbejdsopgaver og fremskridtet af disse. Længden af sprints har været en smule uregelmæssig, da det rent lavpraktisk ikke har været muligt at arbejde på projektet i regelmæssige intervaller. Især i starten af projektperioden var arbejdet meget begrænset til de skemalagte timer indenfor faget, hvorimod der i slutningen også blev prioriteret projektarbejde over fritid. Overordnet set har et sprint varet en dag, med enkelte arbejdsopgaver delt ud mellem gruppens medlemmer til at blive udført imellem de skemalagte arbejdsperioder, når det gav mening.

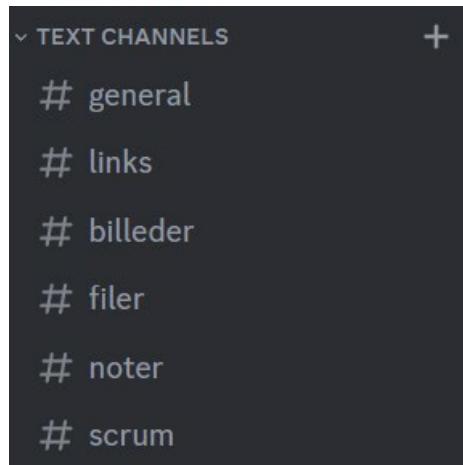
I forhold til rollefordelingen med SCRUM-rollerne blev de uddelt så alle i gruppen var udviklere, alle arbejdede jo på projektet sammen i sprints. SCRUM-master rollen blev givet til projektlederen for overblik og struktur. Product owner rollen blev givet til et gruppemedlem som ikke var projektleder. Dette blev gjort både for at mindske arbejdsbyrden og ansvaret hos projektlederen, samt for at sørge for at hvis projektlederen var syg en dag, ville der stadig være en i gruppen hvis ansvar det var at holde lidt overblik.

I starten af projektperioden, da arbejdet hovedsageligt foregik i skoletiden ud fra det planlagte skema, blev der regelmæssigt holdt SCRUM møder i starten og slutningen af arbejdsdagen. Dette faldt fra mod slutningen af projektet. I starten var der altid et naturligt start- og sluttidspunkt på arbejdsdagen da den fulgte skoledagen. I slutningen af projektet arbejdede gruppen som sagt det meste af fritiden på projektet, og gruppens medlemmer havde private forpligtelser der gjorde at de ofte blev nødt til at komme og gå fra arbejdet på projektet på forskellige tidspunkter. Det gjorde at den naturlige start og slut der før havde været ikke længere var til stede. Det resulterede i at der i de sidste dage af projektperioden ikke blev brugt særligt meget SCRUM.

Discord til kommunikation

Til at holde styr på SCRUM blev kommunikationsproduktet Discord brugt. Discord er et program hvori der nemt kan kommunikeres samt deles filer, på en måde hvor det er nemt at holde overblik. Gruppen har tidligere haft erfaringer med at bruge programmet Trello til SCRUM. Trello er et program hvori der nemt kan laves kanban-tavler, som er gode til at holde overblik over flere processer på en gang. Gruppens erfaring er dog, at selvom Trello er et program, der virker godt til at holde overblik over SCRUM, så gør den detalje at det ikke tilbyder meget i form af kommunikation at det tit og ofte kun vil være projektlederen som bruger det, da det er uafhængigt af alle fildelinger og kommunikationskanaler mellem medlemmer. Trello er også et program som udelukkende bruges til projekter og lign. hvorimod Discord er mere fleksibelt og gruppens medlemmer bruger det privat i dagligdagen. Gruppen valgte derfor at bruge discord til at holde styr på SCRUM da det i forvejen var programmet som blev brugt til kommunikation og derfor ville give mere gennemsigtighed, da alle gruppens medlemmer oftere ville tilgå det.

Discord giver mulighed for at oprette en server. En server fungerer lidt ligesom en gruppechat, hvor adgangen kan kontrolleres så indholdet kan holdes privat. En server har dog den ekstra funktionalitet at der i den også kan oprettes kanaler, så det bliver nemmere at organisere kommunikationen. Det gør det også lettere at finde informationer eller lign. da kommunikationen er opdelt. Gruppens opdeling af kanaler kan ses på figur 2.



Figur 2 - Opsætning af kanaler i programmet Discord

Problemformulering

Til dette projekt har vi valgt følgende problemformulering:

Hvordan kan der designes et digitalt fysisk interaktivt brætspil der kombinerer en 3-dimensionel fysisk konstruktion af spilmiljøet ("spillepladen") og digitale interaktionselementer der udnytter lyd og lys med henblik på at øge indlevelsen?

Problemanalyse

Indlevelse i brætspil er anderledes end indlevelse i andre medier såsom bøger, film eller anden fiktion. Dette skyldes, at der typisk er et fokus på andre elementer, og da det kan være svært at skabe en fiktion i et brætspil, som ikke er rent narrativt. I spil som Twilight Imperium kan der dog ses eksempler på fiktion omkring et brætspil som skaber indlevelse. (Twilight Imperium: Third Edition, n.d.)

Det første trin for at skabe indlevelse i et brætspil er at spilleren har opmærksomhed på spillet. Hvis spilleren ikke er opmærksom, vil de blive hevet ud af spillets fortælling og narrativ. Det næste trin er involvering i spillet. Spilleren bliver nødt til ikke blot at være opmærksom på spillet, spilleren skal også være investeret i spillet. Dette kan til sidst føre til indlevelse, men dette kræver dog både elementer såsom at spilleren har en indgang til spillet i form af f.eks. en spillerkarakter og en fiktion at indleve sig i. (Calleja, 2022)

Det betyder, at der i spillet skal opbygges en fiktion. I dette spil er det valgt at gøre gennem digitale virkemidler og udformning af bræt fremfor ved tekst. Dette giver tomme pladser, som spilleren selv skal udfylde, og derved kan spilleren skabe sin egen fortælling omkring hvad der sker i spillet. Spilbrættet er derved blot stimulerende for indlevelsen i stedet for at kontrollere indlevelsen.

En generel barriere for brætspil er dog regelsættet. For at en spiller kan spille skal der først gå en vis mængde tid med opsætning og at lære regler. I tilfældet med et brætspil med digitale elementer kan denne blokade dog afhjælpes med design i pladen, som hjælper spilleren med hvad der skal ske som det næste. Dette giver også spilleren mulighed for at fokusere mere på indlevelsen i stedet for reglerne, hvilken kan øge indlevelsen, da spillerens mentale ressourcer er begrænsede. Hvis al fokus er på hvordan spillet skal spilles i stedet for at spille spillet, står spilleren udenfor spillet i stedet for inde i spillet. (Calleja, 2022)

Kravspecifikationer

Før gruppen valgte tema blev der opstillet nogle ønsker til hvad projektet skulle indeholde. Dette blev gjort i et forsøg på at opnå et produkt der både ville være sjovt og interessant for gruppen at producere. Gruppens ønsker kan tolkes som kravene for valg af tema. Gruppens ønsker kan ses i højre kolonne i tabel 1. Temaet "Taktile (spil)produkter (fysisk interaktionsdesign)" blev valgt, da det bedst kunne rumme gruppens ønsker for projektet. Med det valgte tema fulgtes nogle krav for produktet. Gruppen opsatte også selv nogle krav. Alle kravene kan ses i venstre kolonne af tabel 1. Tabellen er kun for at give et overblik og gør ikke noget for at uddybe hverken krav eller ønsker. Udvalgte krav og ønsker er dog blevet uddybet, og kan læses i afsnittet under tabel 1.

Krav:	Ønsker:
Passende scope til opgaven	2D-plan robot
Human-Computer Interaction	Droner
Fysisk produkt	RF-signal
Innovative interaktionsredskaber til computer	Sjov
Interesse	Gamification/Inklusion af spil
Realiserbart	Fysisk produkt
	Lyd

Tabel 1 - Krav og ønsker

Opgavens krav

Det overordnede krav for eksamensprojektet er at lave et produkt hvori der indgår interaktion mellem menneske og maskine. Omfanget af interaktionen kan variere fra bred til smal. Udover det overordnede krav er der også nogle mere specifikke krav som kommer af det valgte tema. Det valgte tema for denne opgave er "Taktile (spil)produkter (fysisk interaktionsdesign)". Med det valgte tema skal der arbejdes innovativt med interaktionsredskaber med det formål at udvikle et fysisk produkt, som kan videregive input til en computer, samt modtage feedback derfra. Produktet kan evt. være udviklet til at hænge sammen med et spil eller andet program.

Gruppens krav

Gruppens egne krav som blev opsat til produktet, var mere rettet mod praktiske elementer fremfor noget teknisk eller teoretisk. Meget i stil med ønsket "sjov" havde gruppen et krav om at der skulle være en fælles interesse i at lave produktet. Interessen i produktet skulle gerne være med til at holde fokus i gruppen og sørge for arbejdsmoralen ikke falder for meget, når der evt. kunne forekomme lange arbejdsdage, som også godt kunne være placeret i forlængelse af hinanden.

Helt praktisk havde gruppen også kravene "realiserbart" og "passende scope". Kravene minder meget om hinanden og fungerer bedst i sammenspil med hinanden. Ved at sigte efter et realiserbart produkt mindskes sandsynligheden for stress og forhastet arbejde mod slutningen af arbejdsprocessen. Et passende scope eller omfang kommer i spil i forlængelse af det ved at sikre at projektet ikke er for småt, og hjælper desuden gruppen med at blive udfordret under udviklingen, samt sikrer at der er nok kød på processen til at kunne udnytte hele projekttiden, og selvfølgelig også kunne opfylde rapportkravene, da projektet er en eksamensopgave.

Gruppens ønsker

Eftersom dette projekt er et eksamensprojekt og derfor så også et projekt der vil involvere arbejde i fritid og arbejde over en længere periode, har projektgruppen valgt at opstille nogle ønsker ud fra personlige interesser, for at sikre at arbejdsmoralen kan holdes høj. Ikke alle ønsker endte med at

blive opfyldt, men et forsøg blev gjort på at inkludere så mange som muligt, uden det gik ud over kvaliteten af produktet.

Eftersom et af gruppens medlemmer er stor lyd- og musikentusiast, var der et ønske om at inkludere lyd. Lyd i et produkt kan også være med til at give mere dybde og en bedre helhedsfornemmelse, så længe inklusionen af lyd også giver mening at have med i produktet og tilfører noget værdifuldt i forhold til produktets formål. Der var også et mere teknisk ønske om at projektet indeholdt droner eller en robot som bevægede sig over en 2D-plan og udførte noget arbejde. Gamification af et produkt eller en anden måde at involvere spil var endnu et af gruppens ønsker. Sidst men ikke mindst var gruppens ønske om at det skulle være sjovt. Projektet skulle gerne munde ud i et produkt som er sjovt at interagere med, og som forhåbentlig i sammenhæng med det, har været sjovt at udvikle.

Design

I dette afsnit beskrives hele designprocessen, der leder op til den fysiske produktion. Det inkluderer idégenerering, valg af idé, design af spillets regler og elementer osv.

Idégenerering

Dette afsnit indeholder vores idegenerering, samt en detaljeret beskrivelse af de ideer gruppen har været igennem i denne del af processen.

Forberedt brainstorm

For at starte idégenereringsprocessen, valgte vi i gruppen at lave en såkaldt forberedt brainstorm. Metoden er en variation af den klassiske brainstorm (Jeppesen, Henriksen, & Routhe, 2019), hvor gruppens medlemmer får til opgave hver især at finde på tre idéer hjemmefra, her ud fra den valgte case "Taktile (spil)produkter (fysisk interaktionsdesign)" - denne idégenerering kom forud for valget af problemformulering. Derefter fortsættes brainstormen i gruppens fælles arbejdstid med det samme princip som den almindelige brainstorm: alle idéer er velkomne, og gruppens medlemmer bygger videre på hinandens idéer i stedet for at kritisere og skyde ned.

Denne metode bruges for at minimere spildtiden i brainstorm, hvor gruppen bruger tid på at tænke over idéer, og gør i stedet, at gruppen allerede har et springbræt, som videre diskussion og idégenerering kan foregå ud fra. Selvom det ikke nødvendigvis er én af de forberedte idéer, der bliver valgt, giver det altså stadig et sted at starte, når brainstormen sættes i gang fælles. Det er også en metode, som gruppens medlemmer har god erfaring med fra andre projekter, da den har vist sig at strømline idégenereringsprocessen en del, og gjort at gruppens medlemmer hver især kan sætte deres kreativitet i gang på det tidspunkt og den måde de foretrækker; for nogle kan det være en fordel, da det kan være svært at finde på idéer når man bliver "tvunget" til det i den klassiske brainstorm. De idéer der kom ud af denne proces var:

- Robot der spiller guitar og andre instrumenter ud fra brugerinput
- Kortspil med RFID-kort, der trigger lys og lyd
- Kooperativ lydoplevelse, hvor brugere samarbejder om at lave musik med ukonventionelle "instrumenter"
- Brætspil, der handler om at bevæge sig rundt og frigøre spillerens brikker fra "fængsler" på brættet
- Tetris i fysisk format
- Tetris på et display på en jakke med smart fabrics
- Kampbaseret kortspil a la Magic eller Hearthstone, hvor spillere konstruerer en totem der fungerer som spillerens liv, og dyster mod hinanden
- Brætspil, hvor spillere bevæger sig rundt for at samle skatte - interaktive elementer i form af kort, der trigger faste elementer på brættet

Nogle idéer er udeladt, da de lå for tæt op ad allerede nedskrevne idéer eller ikke inspirerede til videre diskussion. De idéer, der blev overvejet nærmere efter brainstormen, er uddybet herunder.

Skattespillet

Skattespillet var en idé til et spil, hvori tre til fire spillere skulle konkurrere om at samle skatte eller artefakter på et bræt. Idéen var løst baseret på spilserien *Mario Party* (Wikimedia Foundation, 2023). Specifikt blev der draget inspiration fra *Mario Party*-seriens detaljerede og interaktive spillebræt, som bidrager en stor del til spilleroplevelsen. Derudover var det også planlagt, ligesom i *Mario Party*-serien, at have forskellige terninger med forskellige fordelinger og vægtninger af tal, så spilleren kunne have en større indflydelse på deres bevægelse end i traditionelle terningbaserede

bevægelsessystemer som f.eks. Ludo (Wikimedia Foundation, 2023). Udover det var en stor del af idéen at have RFID-kort, som spillerne kunne samle og bruge til at flytte eller aktivere bestemte elementer på brættet, som f.eks. en båd, der kunne flytte spillebrikker fra den ene side af brættet til den anden, eller fælder, der kunne blokere bestemte veje på brættet.

Totemspillet

Totemspillet var en spilidé, der tog udgangspunkt i et regelsæt der drager inspiration fra duelbaserede kortspil som Magic the Gathering eller Hearthstone, hvor spillere dyster om at reducere modstanderens liv til 0 først (Wikimedia Foundation, 2023). Det, der ville adskille denne idé fra andre lignende spil, var at spillernes liv i stedet for en terning eller et tal på papir var baseret på en totem, der fysisk repræsenterede spilleren på bordet (*Figur 3*). Når et slag fra en modstander ramte en spiller, ville den spiller fjerne en brik fra sin totem, og en gavnlig effekt skrevet på undersiden af brikkene ville aktiveres. Dette fungerer også som en form for catch-up mechanic, der gør at spillere, der kommer bagud, vil få en bedre chance for at komme ind i spillet igen (The Thoughtful Gamer, 2017). Det fører formentlig til tættere afgjorte spil, hvor alle spillere kan være med længere. Intentionen var, at spillere selv ville kunne bygge deres totem ud fra en række af brikkene med forskellige effekter, så brikkene bedst muligt gavner deres deck af kort, som de har sammensat.



Figur 3 - Mockup af totemspillet

Kooperativ lydoplevelse

Denne idé var et interaktionsdesign der var baseret på at få deltagere til at skabe musik i samarbejde, ved brug af nogle ukonventionelle "instrumenter" der var sat op i form af et bord eller en anden form for ruminstallation. Her var nogle af idéerne f.eks. en trommemaskine, hvor brugere placerer klodser i bestemte positioner for at få trommer til at spille på de passende takter, en tryksensitiv plade der varierer lyden og intensiteten af musikken, eller et håndtag der kan placeres i forskellige positioner for at spille forskellige akorder. Alle instrumenterne/lydinteraktionerne ville desuden kun have mulighed for at spille toner i den samme toneart, så det er nemmere og mere tilgængeligt for ikke-musikere at lave et samspil, der lyder godt.

Valg af idé og problemformulering

Da dette eksamensprojekt klart er det største teknikfagsprojekt, der hidtil er blevet lavet i denne gruppe, var det vigtigste i udvælgelsesprocessen, at alle medlemmer var engagerede og motiverede af projektets indhold. Derfor blev der i gruppen valgt ikke formelt at vælge idéen ud fra en

kravmatrix eller lignende idéudvælgelsesmetode, men i stedet primært ud fra en diskussion bundet i gruppens interesse for idéerne samt de krav, der opstilledes tidligere.

Den idé, der umiddelbart havde størst interesse hos gruppen, var skattespillet. Kombinationen af brætspil med de elektroniske elementer som RFID-kort, der aktiverede bevægelige elementer, virkede som et interessant designrum, som alle gruppens medlemmer syntes var spændende at arbejde med. Med de elektroniske elementer, der skulle styres af en lille computer, opfyldte idéen kravet om Human Computer-Interaction. Det virkede også som et projekt med et passende omfang, der hverken blev for stort eller småt, og opfyldte flere af gruppens ønsker, som f.eks. inklusion af spil, fysisk produkt og forhåbentlig sjovt produkt. Ønsket om lyd kunne også implementeres ved brug af f.eks. baggrundsmusik og lydeffekter, når bestemte ting skete på brættet.

Efter at have valgt den idé, opstillede vi i gruppen en problemformulering, der skulle sikre at vi ikke kom for langt væk fra kernen af den idé, og hjælpe med at holde fokus på det vigtige i idéen. Problemstillingen blev: *"Hvordan kan der designes et digitalt fysisk interaktivt brætspil der kombinerer en 3-dimensionel fysisk konstruktion af spilmiljøet ("spillepladen") og digitale interaktionselementer der udnytter lyd og lys med henblik på at øge indlevelsen?"*

Skift af idé

Efter at have lavet nogle simple prototyper af skattespillet (*Figur 4*), blev gruppen enige om at spillet i dets nuværende form ikke var sjovt at spille. Det største problem var at der foregik for meget på brættet, og at brættet var for stort og åbent. Det gjorde, at spillerne for det meste undgik hinanden for at tage skatte i hver deres hjørne i stedet for at interagere med hinanden i jagten på skatte, og at der var ikke nogen tydelig core mechanic i spillet, men i stedet en masse små gimmicks.

Dermed stod valget mellem enten radikalt at omtænke spillet, eller finde på et nyt koncept. Efter et gruppemøde blev det besluttet at skifte til en anden idé. Da projektets problemformulering nu var fastlagt, var mange af idéerne fra den tidligere brainstorm ikke længere passende - derfor var gruppen nødt til at skabe et nyt koncept.



Figur 4 - Prototype af skattespillet

Efter endnu en diskussion på gruppen blev det besluttet at ændre spillet til en dyst om at overtage territorier på et bræt. Brættet blev lavet meget mindre, for at opfordre spillerne til at interagere mere med hinanden, og spillerne fik hver tre brikker i stedet for én for at skabe stærkere muligheder

for strategi og planlægning. I stedet for de mange forskellige mechanics spillere kunne interagere med rundt på bordet, blev overtagelse af territorier spillets core mechanic, som også var direkte forbundet til sejr. Bevægelsessystemet med terninger blev ændret til at bruge kort, igen for at tilføje et større strategisk element, der stadig var påvirket af held. Spillet fik navnet *Territory Takeover*.

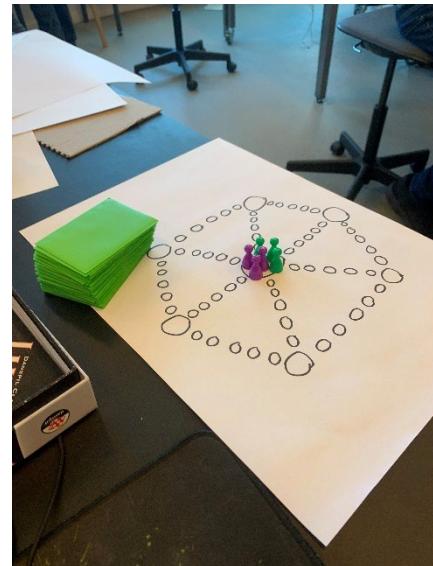
Spildesign

Selvom dette projekt og dets problemformulering mest er baseret på, hvordan indlevelse kan øges i et brætspil ved tilføjelsen af forskellige digitale og fysiske interaktionselementer og udformningen af spilmiljøet, er det også vigtigt at selve spillets design er velovervejet og godt udarbejdet. Derfor er der gået mange tanker ind i hvordan *Territory Takeovers* regelsæt og elementer skulle designes - de tanker bliver beskrevet i dette afsnit.

Det basale koncept er et brætspil designet til 3-4 spillere, hvor spillerne bevæger sig rundt på brættet og dyster om at tage kontrol over seks specielt markerede felter kaldet "arenaer". Spillerne bevæger deres tre brikker hver rundt ved at bruge kort, der trækkes fra en delt bunke på bordet. Alle spillere starter med deres brikker på midten, og der er fem felter mellem midten og hver af arenaerne, som set på prototypen på figur 5. Inden spillet starter, trækker spillere desuden seks kort fra bunken.

Spilloopet består af to faser; den første er planlægningsfasen, hvor hver spiller lægger tre kort med ansigtet nedad fra deres hånd i planlægningszonerne på brættet. Når alle spillere har planlagt tre kort, går spillet til kampfasen. I kampfasen vender spillerne på skift deres planlagte kort ét ad gangen i den rækkefølge de ligger i, og udfører handlingerne beskrevet på kortet, som f.eks. at rykke deres brikker eller trække kort. Hvis en spillerens brik lander på en tom arena, overtager den spiller arenaen, og de får ét point for hver tur de afslutter på arenaen. Hvis en spiller i stedet lander på en arena, hvor en anden spiller står med en brik, starter de to spillere en duel. Når spillere starter en duel, vælger de hver især to kort fra hver deres hånd, der lægges med ansigtet nedad og afsløres på samme tid. Den spiller der har den højeste samlede attack-score, der står på kortene, vinder duellen og overtager arenaen. Taberen rykker sin brik ind til midten. Hvis spillerne står lige, vælger de begge et kort fra deres hånd og afslører det på samme tid; og hvis de stadig står lige, vinder den forsvarende spiller (den brik der stod på arenaen før duellens start). Hvis en spiller på noget tidspunkt i en duel ikke har flere kort tilbage at bruge, tæller det som et tomt kort med en attack-score på 0.

Efter hver kampfase er ovre, trækker alle spillere kort fra bunken til de har seks kort på hånden. Derefter starter en ny planlægningsfase, og spilloopet starter dermed forfra. Spillet fortsætter indtil en spiller har 10 point. Hvis to spillere når 10 point på samme tid, trækker de syv kort og starter en duel. Vinderen af den duel vinder spillet. Turrækkefølgen i kampfasen skifter efter hver planlægningsfase.



Figur 5 - Papirprototype

Kortdesign

Da spillets regelsæt er forholdsvis simpelt, ligger meget af det egentlige design og balancering i kortene, der styrer elementer som bevægelse og at trække kort. En interessant konsekvens af den måde, spillets bevægelsessystem og kampsystem interagerer med hinanden, er at kortene både bliver brugt som en ressource i bevægelse og kamp. Det betyder, at en spiller skal overveje, hvorvidt det er mest værd at bruge et kort til bevægelse eller kamp, og at kort der manipulerer med hvor mange kort en spiller har på hånden i bevægelsesdelen af spillet har en direkte effekt på kampdelen.

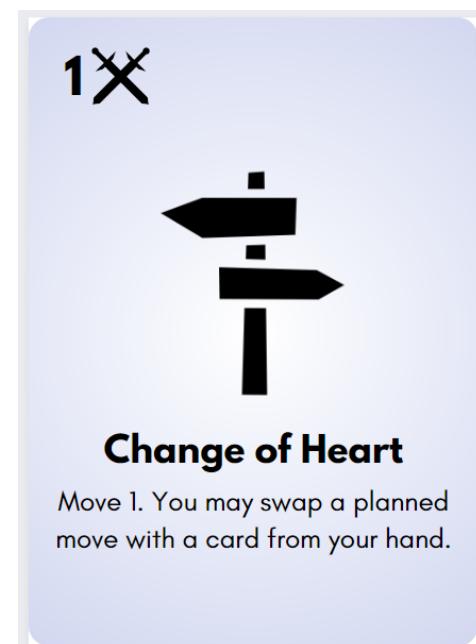
Kortene er designet med den generelle filosofi, at jo sterkere kortets effekt er, desto højere er attack-scoren også. En overvejelse i designprocessen var, hvorvidt kortene med stærke effekter skulle have en lavere attack-score som en form for balancering, men det blev droppet, da det formentlig ville lede til et scenario hvor kortene med stærke effekter kun blev brugt for deres effekt, og de kort med høj attack-score kun ville blive brugt til dueller. I dette system er det i stedet et dilemma; vil spilleren bruge et stærkt kort, eller gemme det til en eventuel duel? Det skaber også et element af held, alt efter hvilke kort spillene trækker, hvilket er godt for et forholdsvis casual brætspil, da det udjævnner niveauforskellen mellem en nybegynder og en øvet spiller lidt - samtidig er det også en form for tilfældighed der er mulig for spillere at respondere på, så de kan justere deres strategi alt efter hvilke kort de trækker (Compton, 2018). Det betyder at det er en knap så frustrerende form for tilfældighed som f.eks. et slag med en terning, der ofte bruges til bevægelse i brætspil - her har spilleren som regel ingen mulighed for at påvirke resultatet, hvilket kan skabe en følelse af at spillerens valg ikke betyder noget.

For at gøre processen med design af kortene nemmere, blev der først opstillet et Excel-ark hvor kortene kunne skrives ind med deres navn, attack-score og effekt (*Figur 7*). Dermed kan der nemmere skabes et overblik og rettes i kortene. De simple kort *Careful Turn* og *Sprint* er i 10 kopier i bunken, mens de andre er i 5, så de sterkere eller mere indviklede kort ikke optræder helt lige så ofte.

Card Name	Attack	Description	Card Amount
Careful Turn	1	Move 1. Draw a card.	10
Sprint	1	Move 2.	10
Leap	2	Move 3.	5
Coordination	1	Move two units 1.	5
Reckless Dash	2	Move 4. Reveal your next move (<i>it doesn't activate till next round</i>).	5
Ambush	3	Move 2. If this starts a battle, you get +2 Attack.	5
Mirror Image	2	Choose another player. Copy their last move.	5
Change of Heart	1	Move 1. You may swap a planned move with a card from your hand.	5
Sabotage	3	Draw a card. Swap another player's planned move with a card from your hand.	5
Blazing Speed	2	Move 2. Discard a card from your hand to move 4 instead.	5
Espionage	3	Move 2. You may look at another player's planned moves.	5

Figur 7 - Excelark over kortene

Efter at have beskrevet kortene i Excel, blev der udarbejdet et design i den webbaserede grafikdesigner Canva (Canva, u.d.). Kortets design består af en simpel baggrund med en gradient, et tal med et sværd bagved der repræsenterer kortets attack-score, et ikon og navn til at identificere kortet nemt for øvede spillere og en beskrivelse (*Figur 6*). Designet er med vilje holdt simpelt og abstrakt, så spilleren selv kan tillægge en mening der passer til den historie de skaber gennem spillet. En anden vigtig overvejelse i kortenes design er at teksten ikke må gå for langt ned i bunden af kortet, da teksten i så fald kan blive blokeret af spillerens hænder, når de holder kortene. Derfor er der med vilje et blankt stykke i kortets bund. Alle spillets ikoner er taget fra Canva's grafikbibliotek og den copyright-fri side Game-Icons.net. (Game-Icons, u.d.)



Figur 6 - Eksempel på kort

Overvejelser ift. spildesign

I processen med at designe spilletts regler og kort, var der selvfølgelig mange ting der blev ændret undervejs. Dette afsnit forsøger at beskrive de vigtigste ændringer og overvejelser, og hvorfor de er blevet lavet.

I den første version af reglerne var maxstørrelsen på spillerens hånd fem kort; dette betød at spilleren kun havde to kort tilbage på hånden i kampfasen, siden hver planlægningsfase tager tre kort hver tur. På dette tidspunkt var der heller ikke særlig mange kort, der lod spilleren trække flere kort. Når spilleren kun havde to kort på hånden til kampfasen, gav det ikke nogen mulighed for at vælge kort til dueller; i stedet var spilleren tvunget til at bruge de to, de havde tilbage på hånden. Dette betød, at det var umuligt at gemme kort til runden efter hvis man indgik i en duel, medmindre spilleren havde et af de få kort, der lod dem trække et ekstra kort, og fjernede derfor et element af planlægning. For at fikse dette problem blev håndstørrelsen til at starte med sat op til syv kort, og kortet *Careful Turn*, som før kun lod en spiller rykke ét felt, fik tilføjet evnen til at trække et ekstra kort. Efter en kort quick and dirty test (se afsnittet "Quick and dirty test") virkede det dog til, at den løsning havde trukket lidt for meget i den anden retning; nu var det ikke lige så vigtigt for spilleren at overveje hvilke stærke kort de ville gemme til kamp, da de typisk stadigvæk ville have nok kort til at have en god chance i dueller, uden at skulle tænke for meget over det. Derfor blev håndstørrelsen til sidst ændret til seks kort som et mellempunkt mellem de to - i de interne tests, der følgende blev lavet i gruppen, virkede det til at være et godt kompromis, da der nu var en passende balance mellem for lidt og for mange kort.

En anden overvejelse vi gjorde i gruppen, var hvorvidt arenaer skulle blive ved med at være under kontrol af en spiller, når den spiller gik ud af arenaen. Det valgte vi at beholde, da spilleren allerede bliver "straffet" ved at andre nemmere kan overtage en arena, hvis de vælger at forlade den. Derfor skaber det en form for risk vs. reward-dynamik (Mullich, 2017) - vil spilleren efterlade en arena sårbar, for måske at kunne overtage en anden, eller holde den de allerede kontrollerer sikker? Samtidig er der lys i hver arena, der lyser farven af den spiller, der kontrollerer den. Det gør, at der ikke er nogle hukommelsesproblemer ift. ejerskab, hvis en spiller forlader en arena, i modsætning til i et traditionelt brætspil - derfor udnytter denne mechanic også det unikke designrum, som spillet er i.

Det sidste vigtige designvalg, var hvor lang tid det gennemsnitligt skulle tage at bevæge sig rundt mellem arenaer. Dette spørgsmål udsprang af designet af brættet, hvor der er fem felter mellem midten og hver arena. Det betyder, at en spiller skal flytte en brik seks felter for at komme fra midten til en arena, eller fra én arena til en anden. Kortene blev designet således, at det praktisk talt er umuligt for en spiller at ramme mere end én arena på en tur, og sørger derfor for at spillet ikke bevæger sig alt for hurtigt. De fleste kort kan rykke brikker enten ét, to eller tre felter; der findes også et par kort, der kan rykke fire felter, men de kræver begge at spilleren ofrer sig på en anden måde - *Blazing Speed* kræver at spilleren ofrer et kort fra deres hånd, mens *Reckless Dash* afslører for modstanderne, hvad spillerens næste træk er.

Lyddesign

Som en del af problemformuleringen har vi også arbejdet med simpelt lyddesign i dette projekt. Oprindeligt var det tænkt, at lydfladen skulle være dynamisk og f.eks. ændre lyden når to spillere kæmpede, når en arena blev overtaget osv., men af hensyn til tid og projektets omfang blev det nødt til at blive ændret. I stedet valgte vi at lave ét lydspor, der blot kører i ring under spillet. Målet med designet af lyden er her, at det ikke skal trække opmærksomheden væk fra spillet, og samtidig skabe en stemning der er passende for spillet. I starten af processen var der tre idéer til temaet og typen af lyd:

- **Krigsmusik:** Horn- og strygeinstrumenter. Stærk rytmefølelse, semihøjt tempo og ”vred” stemning.
- **Lagdelt ambient:** Skiftende lag af synthesizers. Forskellige dele overlapper og bevæger sig rundt i musikken.
- **Minimalistisk:** Klaver, naturlige lyde og evt. simple lag af synthesizer. Langsomt, flydende tempo og rolig, lettere mystisk stemning.

Krigsmusik var umiddelbart den idé med den stærkeste tematiske forbindelse til spillet, idet spillet handler om at kæmpe om territorie mod de andre spillere. Men efter at have produceret en kort demo i det tema, blev det besluttet i gruppen at det gav en for frustrerende stemning og tog for meget fokus væk fra selve spillet, så valget stod i stedet imellem en form for elektronisk ambient musik, og en mere minimalistisk, klaver-orienteret musik. Fordelen ved ambient-idéen er, at synthesizere har mange lydparametre, der kan sættes til flydende at ændre sig over tid, så man som lyddesigner kan få en forholdsvis simpel komposition til at blive ved med at være interessant over længere tid end med traditionel instrumentation. Dog var der delte meninger i gruppen, om hvorvidt det passede til spillets tema og det fantasyagtige visuelle udtryk. Derfor blev det besluttet, at lydsporet skulle laves med noget klaver, naturlyde og evt. nogle simple lag synthesizer.

Produktudvikling

Her skal i beskrive mere teknisk hvordan i har lavet jeres produkt og jeres prototyper. Beskriv alt det arbejde i har lavet, med billeder og diagrammer. Hvis I har lavet 3 forskellige prototyper igennem jeres projekt, så beskriv alle 3. Det er også her i fortæller hvilke programmer i har brugt, samt dokumentere jeres kode. Det giver mening at bruge godt med undertitler. Hvis i har brugt en specifik arbejdspreses så beskriv den med kilder.

Produktion af bordet

Selve bordet (*Figur 8*) er konstrueret af to 12mm MDF-plader, separeret med 200mm. Denne separation giver et hulrum i bordet, hvor det er muligt at placere de elektroniske komponenter der styrer spillet. Et stykke 45x45x200mm imprægneret reglar er placeret i hvert hjørne, for at separere pladerne. Dog er reglarene kun fastmonteret til den øverste plade med en 70mm skrue, samt trælim. Dette er gjort for at gøre det nemmere at tilpasse de elektroniske komponenter senere i processen, hvis det bliver nødvendigt. For at holde den øverste del af bordet fast er fire krydsfinerplader monteret på reglarene med skruer. Pladerne har en højde så de kan skjule siderne af de øverste 100mm flamingo, samtidig med at de overgår kanten af den nederste MDF-plade. På denne måde, kan den øverste del af bordet holdes på plads, samtidig med at det også er muligt nemt at åbne bordet op.



Figur 8 – Rendered 3D-model af siden af bordet

Gruppen valgte at skabe hulrummet af æstetiske grunde. Eftersom bordet skulle kunne stå i et offentligt forum er det vigtigt, at bordet ser præsentabelt ud. Derudover er det også vigtigt at gemme elektronikken væk af to grunde: Den generelle bruger af bordet skal ikke have adgang til elektronikken, da den er finjusteret specifikt til spiloplevelsen, så enhver manipulation af komponenterne kunne forværre spiloplevelsen. Derudover er pointen også ifølge vores problemstilling at øge indlevelsen. Så hvis brugerne er i stand til at se det centrale operationssystem der styrer spillet, kunne det have en negativ effekt på spillerens indlevelse, i det at brugerens fokus kunne blive splittet. Grunden til at hulrummet har en højde på 200mm, er for at give rigeligt med plads til eventuelle spilelementer, eftersom vi startede konstruktionen af bordet før spilideen var udarbejdet færdig. Derudover var det også originalt hensigten at tilføje dynamisk lyd til spillet, hvilket kræver en bærbar computer til håndteringen af lyd. Dog blev denne ide aldrig implementeret grundet tidsmangel.

Samme type reglar som tidligere er brugt til at konstruere benene på bordet (*Figur 8*). Eftersom bordet ikke bærer en væsentlig mængde vægt, er det ikke nødvendigt at benytte metalben til bordet. Derudover er det også hensigten, at spillerne skal stå og gå omkring bordet; derfor en højde af benene på 750mm. Med den yderligere højde på 324mm, giver det en total højde på ca. 1074mm. Yderligere er den detaljerede natur af spillepladen med til at indikere, at det ikke er hensigten at spillerne skal læne sig op ad bordet. På baggrund af dette - og for at spare materialer - har gruppen fravalgt yderligere stabilisering af bordet med krydsbjælker.

Produktion af landskabet

Herunder en beskrivelse af de forskellige processer der indgik i produktionen af landskabet til brætspillet.

Udformning af flamingolandskabet

Landskabets grundstruktur er formet af en 100mm flamingoplade, da det giver mulighed for nem tilpasning af terrænet; desuden er det et let materiale, da det består af 98% luft (Plastindustrien, 2023), hvilket er overensstemmende med gruppens valg ift. konstruktionen af bordet.

Flamingoen er formet i to trin. Først blev den overordnede form af terrænet udgravet med skeer og andre redskaber, hvorefter det blev flatet ud med en varmepistol. Denne proces blev af sikkerhedsmæssige årsager udført udendørs, med bind over mund og næse, grundet de giftige dampes frigivet under opvarmning. Denne proces var nødvendig for at komme tættere på et mere realistisk landskab, end gruppen kunne forme med hånden.

Efterfølgende er landskabet blevet formateret yderligere for at tilpasse komponenterne indlejret i flamingoen. Heriblandt blev en teknik udtaenkt af gruppemedlem Anders, brugt til at udskære et præcist område til at montere LCD-skærmene nede i flamingolandskabet. Teknikken indebar at skære små firkanter ud, som efterfølgende kunne vippes ud, med enden af en hobbykniv. Denne proces blev gentagende gange brugt, pga. de præcise kanter (*Figur 9*).



Figur 9 – Billede af firkant-metoden benyttet til udskæring

Produktion af teksturer samt miniaturemodeller

For at skabe et varieret landskab, blev der produceret en række miniaturer. Dette blev gjort ved en kombination af organiske materialer og ikke-organiske materialer. Der blev produceret klipper, træer, varierende blomster samt svampe. I følgende afsnit vil metoden for disse ting blive beskrevet. Der blev også produceret en række forskellige teksturer, som blev brugt som henholdsvis udsmykning af modeller, samt teksturering af jord.

Teksturer

Til brug af teksturer blev der brugt en række forskellige materialer blandet sammen med maling og trælim. Herunder gælder dette den rå del af en skuresvamp, som blev klippet i små stykker og brugt som mos, samt en blanding af sand, maling og trælim, som blev brugt til varierende ting, dog hovedsageligt som mos. Sandblandingen blev lavet ved først at blande den ønskede farve og efterfølgende tilføje trælimen. Herefter mættes blandingen med sand, til blandingen har den ønskede konsistens, en fugtig bladning, hvor kornene stadig er tydelige.

Yderligere blev der også lavet græs. Dette blev gjort ved at klippe hampereb i små stykker, gerne omkring tre til fem millimeter, og efterfølgende blande langsom op med en lys grøn indtil alle stykker havde taget farve. Efter dette have tørret på et stykke papir for at trække fugten ud af græsset, kunne klumperne smulders ned i beholder, sådan at det var klar til brug senere. Endeligt blev der også brugt sesamfrø og rasp, som blev blandet op med brunlige farver, for at lave yderligere teksturer til jorden.

Klipper

Til produktion af klipper blev der brugt barkstykker som basis for modellen, da bark naturligt har en laglignende tekstur. Yderligere da det er organisk, kan det også have en mere naturligt ujævn overflade end hvis klippen blev skulptureret i f.eks. ler (*Figur 10*). Grunden til at det ikke er sten, som blev brugt til klipperne, er da et miniaturelandskab netop er i en anden skala end virkeligheden, vil teksturen på en sten virke opskaleret i forhold til andre materialer, såsom bark.



Figur 10 - Billede af et stykke bark.

For at minimere skaden på træerne bagen blev taget fra, blev der valgt ældre træer med løstsiddende bark. Der blev ikke indsamlet træer fra døde træer for at undgå insekter i bagen, samt minimere risikoen for rod og svamp. Efterfølgende blev bagen udtørret ved brug af en varmeblæser med en temperatur på 300 grader. Dette blev gjort for at dræbe eventuelle dyr, da meget lidt liv kan overleve denne temperatur i længere perioder. Herefter blev bagen lakeret for at isolere bagen fra oxygen og dermed yderligere rodsikre modellerne.

Efter bagen var tør, blev modellerne grundet med en grå spraymaling, da spraymaling giver en jævn farve og er hurtigt at applikere, samt har en kort tørretid. Dette gav et resultat som kan ses på figur 11.



Figur 11 - Billede af baset bark

For at skabe en illusion af lys og skygger blev indhakkene i klipperne malet med en mørkegrå akrylmaling. Det gav følgende resultat som set på figur 12.



Figur 12 - Billede af bark efter skygger.

Efter dette blev der brugt en "wash" på klipperne. En wash er betegnelsen for en tyndtflydende maling, som har en løbende konsistens. Dette gør at malingen lægger sig i fordybningerne på modellen, hvilket giver en illusion af skygger. Her blev der brugt Army Paints "Flesh wash" til klipperne, hvilket er en mørk brun med rødlige toner, da dette var den eneste wash tilgængelig. For et bedre resultat kunne der i stedet være blevet brugt en wash med mere kølige toner, såsom Army Paints "Strong Tone" eller "Soft Tone".

Herefter blev kanterne og dele af store overflader på klipperne highlightet ved brug af teknikken "drybrushing", hvor efter påførelse af maling på pensel, tørres penslen af for det meste af malingen. Herefter laves tynde lag med lette penselstrøg. Dette giver en blødere kant mellem farverne end almene penselstrøg, samt maler de højeste steder i teksturen mest, hvilket giver en illusion af highlights. Her blev klipperne highlightet med hvid. Dette gav et resultat som set på figur 13.



Figur 13 - Billede af færdig klippe.

Som detalje blev der senere også sat mos på modellen, som beskrevet under afsnittet "Tekstur på basis af sand" med grønne toner. Dette gør klipperne mere organiske, samt gør dem lettere at passe ind i landskabet, grundet at de nu deler farve med jorden. Det gav de endelige klipper, som set på figur 14.



Figur 14 - Billede af klippe med mos.

Træer

Produktionen af træer var todelt. Henholdsvis stammen samt grene og bladkronen. Til produktion af stammen blev ti-tolv stykker af ståltråd af en længde på mellem femten og tyve centimeter brugt. Først blev stængerne viklet om hinanden. Herefter blev grene samt rødder ligeledes formet fra enderne af ståltråden. Når træstammen synes at have en passende form, tages strimler af stof dækket i trælim og vikles om træstammen. Dette giver træerne deres ru tekstur, og når trælimen hærder bliver træstammerne også mere solide. Når trælimen er hærdet, males træet brunt. Herefter bliver skyggerne highlightet med sort, og træstammerne bliver washed med Army Paints "Flesh wash". Som detalje til sidst blev der også tilføjet mos, lavet ved en blanding af skuresvampestykker samt maling.

For at lave bladkronerne, blev skum først skåret ud i den rigtige form, gerne med to til fire stykker skum som udgør bladkronen tilsammen. Dette giver et mere interessant udtryk, da hvert træ ser forskelligt ud i bladkronen. Herefter blev bladene dækket med trælim for at gøre dem lettere at male på, samt for at stoppe en del af sugningen af maling, gørende teksturen mindre hullet når malet på. Efter trælimen var hærdet, blev bladkronen malet i tre forskellige grønne nuancer. En mørk grøn med en anelse rød i, en mørk grøn bestående udelukkende af gul og blå, samt en lys grøn. Forskellige flader efter klipningen af bladkronen blev malet i en af de træfarver, hvor toppen var lysest, og bunden var mørkest og varians i farven resten af stederne. Til sidst blev kanterne af bladkronen dry brushed gule. (Figur 15)



Figur 15 - Billede af færdigt træ.

Blomster og svampe

Blomsterne blev lavet ved brug af korte stængler af ståltråd dyppet i en sandblanding med varierende farver, herefter sat til tørre stående i flamingo. Der blev valgt farverne blå, gul, lilla og rød, hvor alle planterne med undtagelse af de røde blev lavet med farven aflangt ned langs stænglen, inspireret af hvordan blomsterne sidder på lavendler. De røde blev lavet med en knop i toppen, inspireret af roser.

Svampene er lavet ved brug af tegnestifte med et fladt, rundt hoved. De er basset med en hvid med lidt gul i, i dette tilfælde Vallejos farve "Buff" (*Figur 16*).



Figur 16 - Billede af baset knappenål.

Efter to lag med Buff, blev midten af tegnestiften farvet med en orange, og til sidst en rød inde i midten, for at skabe et fade mellem farverne. Bagefter blev hvide prikker sat på svampene. (*Figur 17*).



Figur 17 - Billede med færdige svampe.

Detaljering af landskabet

Efter landskabet var blevet udformet i flamingo, blev der brugt mosaikfiller som et lag ovenpå landskabet. Dette blev gjort for at udjævne landskabets flamingotekstur, og i stedet beholde de store træk fra udformningen af landskabet. Da mosaikfiller hærder gør det også teksturen bedre at male på, uden at der kommer huller i malingen som er svære at udfylde, som der gør ved maling af flamingo. Yderligere blev mosaikfiller også brugt til at dække hullerne omkring felterne og arenaerne samt kortholderne. I tilfælde med større huller som skulle dækkes, blev et stykke gaffatape placeret over hullet, for at skabe et sted fillerne kunne lægges på. Da fillerne hærder og felterne ikke skal kunne holde stor vægt, er strukturen stadig stabil nok til formålet med denne løsning.

Efter dette lag var hærdet blev der lavet nogle yderligere bakker af skumsvampe dækket med mosaikfiller, og herefter var jorden klar til at blive baset. Da LED-erne på dette tidspunkt var sat ind i spillet, var det ikke muligt at base pladen med spray, så dette blev gjort i hånden med en mørk brun med lilla undertoner. Dette blev valgt for at skabe en kontrast med græssets lysegrønne farve.

Da dette var gjort, blev der lavet endnu et lag med grøn, hvor der med en skuresvamp blev duppet brun maling på, for at gøre farvelaget tykkere visse steder. Dette giver også tekstur til brættet. Herefter blev et lag lavet med grøn farve, som på samme vis blev duppet på med en skuresvamp, med et fokus omkring kanterne og fordybningerne i brættet. Dette er for at opbygge highlights omkring fordybningerne, da det gør at formen på pladen står frem. Endnu et lag med en mere gul nuance blev duppet på omkring fordybningerne på kanten, og til sidst blev kanterne dry brushed med en ren gul, for yderligere opbygning af highlightet.

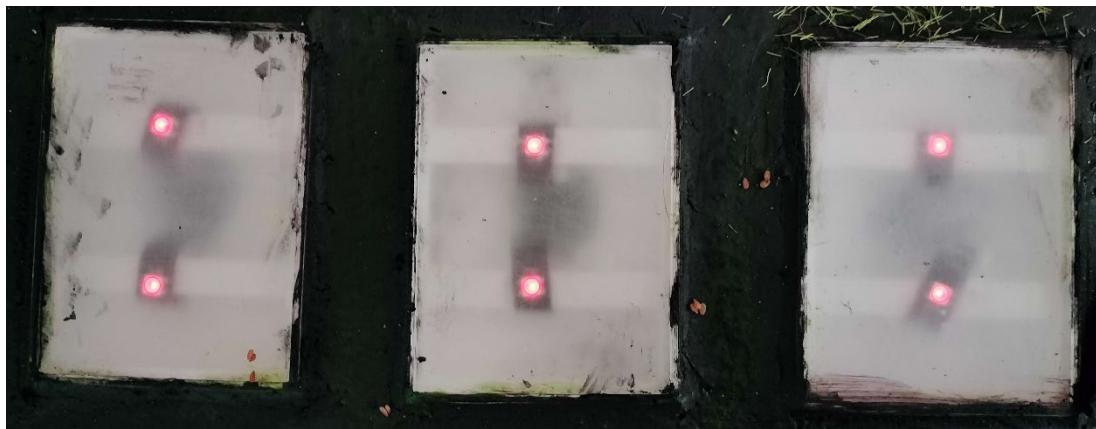
Efterfølgende blev udsmykningen sat på brættet. Først blev klumper af malet rasp og kerner sat på brættet for at skabe yderligere tekstur. De blev sat fast ved først at male med en trælim på bordet, og efterfølgende placere dem de ønskede steder. Herefter blev græsset sat på. Det blev gjort ved at male med en fortyndet trælim, og efterfølgende drysse græsset på brættet i klumpede områder. Der blev valgt ikke at putte græs på hele brættet for ikke at overfylde brættet så det går udover det visuelle udtryk.

Bagefter blev blomster, svampe, træer og sten placeret. Blomster og svampe kunne sættes på brættet ved at stikke dem ned i brættet. Træer og sten er limet fast med trælim. Der er her et fokus på at placere pynt fordelt ligeligt over hele brættet, og træerne er placeret sådan at de blokker mindst muligt for udsynet over brættet. (*Figur 18*)



Figur 18 - Billede af landskab ovenfra.

Som det sidste blev felterne renset for maling, som var kommet på felterne ved et uheld (*Figur 19*). Til dette blev der brugt AK Interactives Paint stripper, som blev påført med en pensel, og efterfølgende vasket af med vand og sæbe (*Figur 20*).



Figur 19 - Billede af felter før rensning.



Figur 20 - Billede af felter efter rensning.

Figur 21 viser billeder af den endelige dekoration af brættet.



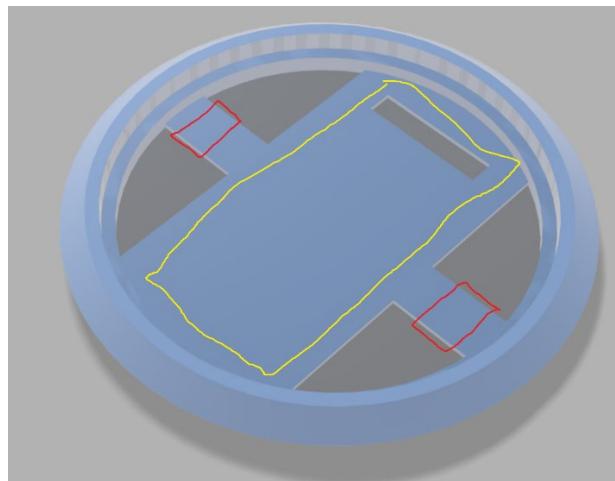
Figur 21 - Kollage med billeder af bræt.

Produktion af brættet

I dette afsnit forklares processen til produktionen af brættet. Dette inkluderer alle de mindre arbejdsopgaver der indgår i det endelige bræt.

Produktion af felter

Arenafelterne er alle 3D printet, og er designet med 3D CAD-softwaren Fusion 360 (Autodesk Fusion 360, 2023). De overordnet design mål for arenafelterne var, at de skulle indeholde en RFID-reader, til at aflæse spillernes RFID-brikker, have to LED lys og have en 4mm uddybning, til en 4mm tyk akrylplade. På nedenstående billede (*Figur 22*) ses 3D modellen af arenafeltet. Den gule firkant repræsenterer RFID-reader'en, med et udskåret hul til dens pins. Dog grundet en komplikation ift. RFID-teknologien brugt, samt en overordnet tidsmangel på konstruktionen af produktet, blev denne dedikeret plads aldrig brugt til at holde RFID-reader'en. En erstatning til RFID-reader'en blev installeret under arenafeltet. De to røde firkanter (*Figur 22*) repræsenterer placeringerne til LED'erne. Eftersom feltet er relativt stort, er det nødvendigt at benytte to LED'er, til at belyse arealet.



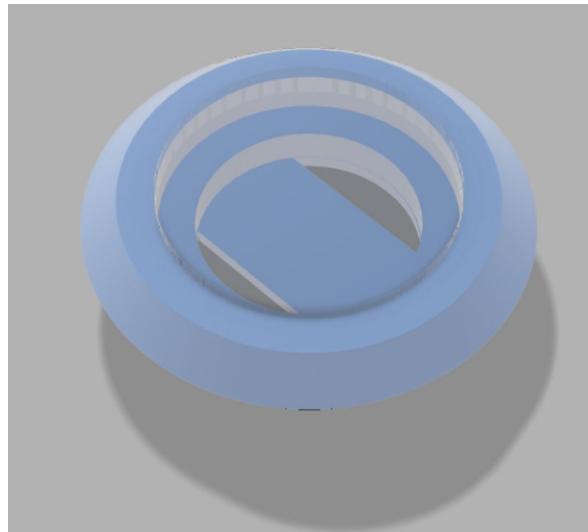
Figur 22 - 3D-model af arenafelt med skitsering til RFID (gul) og LED (rød)

Den blå cirkel (*Figur 22*) er den 4mm akrylplade, der er til for at lyset kan gennemtrænge arenafeltet. For at mindske intensiteten af LED lyset, samtidig med stadig at bibeholde et tydeligt lys, er akrylet blevet behandlet med 800 korn sandpapir. Dette er blevet gjort, for at sprede lyset fra oprindelsespunktet og gøre det mere diffust, hvilket resulterer i et mindre intenst lys, der oplyser et større område. Ifølge en artikel på ledwatcher.com, er der tre hovedmetoder at gøre lys mere diffust, herunder at placere et semi-transparant objekt foran lyskilden (LEDWATCHER, 2020). Dette har resulteret i en mindre optimal spredning af lys, set på nedenstående billede (*Figur 23*).



Figur 23 - Billede af arenafelt efter montering i brættet

Samme specifikationer er også sande ift. de små felter, på nær RFID-reader'en og antallet af LED'er er reduceret til en, pga. størrelsen af det lille felt. Dog er diameteren på den indre cirkel (arealet med akryl plade), reduceret fra 80mm til 25mm, så det passer til diameteren af en spiller brik.



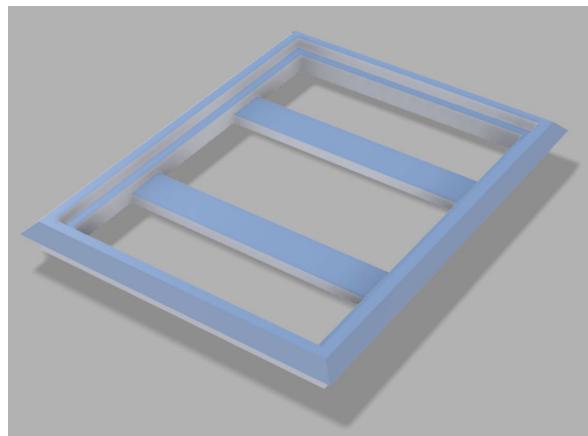
Figur 24 - 3D-model af lille felt

Det lille felt har to cirkelafsnit udskåret i bunden (*Figur 24*), for at gøre plads til enderne af LED'en, kan passe igennem. Dog er spredningen af lys, ligesom arenafelterne ikke optimal (*Figur 25*).



Figur 25 - Billede af lille felt efter montering i brættet

Herunder, modellen for kortholderne, hvor samme metoder er benyttet. Igen, pga. størrelsen, er antallet af LED'er øget til to, placeret på hver deres bjælke (*Figur 26*). Kortholdernes formål er at gøre det muligt at placere kort på brætspilspladen, samtidig med at holderne kan lyse forskelligt lys, for at indikere hvilke kort tilhører hvilke spillere.



Figur 26 - 3D-model af kortholder

Denne model er designet til at være større end kortene, så det altid er muligt at skue lyset bagved kortet, dog er den også præget af problemet med for lidt diffusion.



Figur 27 - Billede af kortholder efter montering i brættet

3D prints processen har været en essentiel del af projektets succes. Alle 3D printet modeller, herunder alle tidligere nævnte modeller, samt spillermodellen, er printet på Creality 3D printerne (Creality, 2023). Grundet den nødvendige præcision af modellerne, pga. de tilsigtede akrylplader, laserskåret til ca. 1/100 af en mm, har lag tykkelsen af 3D printet været nødt til at blive sat til den laveste mulige indstilling. Den mindste indstilling tilladt af slicer programmet Ultimaker Cura (Ultimaker, 2023), er 0.14mm, på både Creality CR-6 SE og Creality Ender-3 S1 printerne. Dog var en konsekvens af denne præcision, at printtiden for komponenterne, var relativt lange. Dette var en hindring ift. konstruktionen af bordet, da det ikke var muligt at fortsætte produktionen af brætter, uden monteringen af de ikke endnu printet komponenter. Derfor var der en periode i processen, hvor monteringen af komponenterne skete i takt med at komponenterne blev færdige med at printe. Denne fejlhåndtering af tid kunne måske være undveget ved at nedsætte præcisionen af 3D printene. Men på tidspunktet, var det gruppens overbevisning, at en nedsætning af kvalitet kunne forsage et yderligere spild af tid, i tilfælde af fejlprint eller print der ikke passer til akrylpladerne. Derfor tog gruppen valget at gå den sikre, men lange vej, ift. printet. Det blev dog også udregnet, at printene var teoretisk muligt at opnå med tiden tilbage, eftersom hvert print havde en printtid på omkring 50 minutter, fordelt på tre 3D printerne.

Dette valg, var også årsagen, til at det ubrugte levn fra den tidligere version af RFID-reader'en aldrig blev fjernet eller opdateret.

Opsætning af elektriske komponenter

Her beskrives de forskellige valg der er taget ift. produktionen af kode til at styre brætspillet's gang. Dette indebærer både vores struktur af kode, samt detaljeret beskrivelser af de forskellige scripts brugt til at styre spillet.

God kodestruktur

Allerede fra starten var det vigtigt at holde koden organiseret. For at opnå en letlæselig kode samt god kodestruktur, opdeles kodens diverse funktioner i forskellige header filer, også kendt som libraries. Det vil sige at f.eks. koden til at håndtere RFID komponenterne, og koden til at håndtere alle LED'erne ikke behøver at stå i samme dokument. Alle header filer bliver dog importeret og brugt i et enkelt ".ino" script, specifikt "Main.ino" under dette projekt. Den vigtigste af alle disse headerfiler er "Variables.h". Specifikt denne fil er vigtig, da den gør det muligt for alle de andre filer at dele data mellem hinanden. Som navnet hentyder til, indeholder denne headerfil kun variabler. I "Main.ino" scriptet, er "Variables.h" også det første library, der bliver importeret, da det ellers ikke ville være muligt for de andre filer at have adgang til alle de andre variabler. Disse andre filer er "Displays.h", "GameSequence.h", "LED.h" og "RFID.h".

Main.ino scriptet

```
bool gameState = false;

//RFID
int RFID_func_call = 0; //1 = enter, 2 = left
int ReaderIndex = 0;
String Player = "";

//Spillere
String yellow[] = {"041866B7DE", "041866D5FD", "041866BC7D"};
String blue[] = {"041866BFF1", "041866C12B", "041866FB20"};
String orange[] = {"041866FF8C", "041866E0BB", "041866D246"};
String purple[] = {"041867035F", "041866EAB7", "041866C12D"};
```

Figur 28 - Udsnit fra variablerne i Variables.h

Dette billede (Figur 28) er et uddrag fra header filen Variables.h. Den første boolean ved navn gameState refererer til om spillet er i gang. De næste tre variabler har alle at gøre med RFID. Den første af disse variabler, ved navn RFID_func_call, eksisterer så RFID.h har en måde at kalde funktioner i Main.ino på. Denne metode består af denne integer, der kan have tre forskellige værdier. Main.ino tjekker konstant denne værdi, og baseret på dens værdi, kalder den forskellige funktioner. Værdien 0 betyder at ingen funktioner skal kaldes, værdien 1 betyder at funktionen PlayerEnter skal kaldes og værdien 2 betyder at funktionen PlayerLeft skal kaldes. Iblandt variablerne er der også fire arrays. Det er disse arrays, som holder styr på hvilke RFID-tags, der tilhører hvilke hold.

Som nævnt før, er Main.ino scriptet brugt til at sammenflette alle header filerne.

```
void setup(){
    //Library setups
    RFID_setup();
    Button_1.setDebounceTime(50);
    Button_2.setDebounceTime(50);

    Serial.begin(9600);
    LED_Setup(true);
    DisplaySetup();
}
```

Figur 29 - setup funktionen fra Main.ino

Funktionen, setup (Figur 29), bliver kørt når Arduinoen starter. Derfor kalder denne funktion de andre header filers setup funktioner.

```
void loop(){
    Button_1.loop();
    Button_2.loop();

    if(!gameState && (Button_1.isReleased() || Button_2.isReleased())){
        gameState = true;
        Serial.println("Starting Game");

        GameStartUpSequence();

        delay(1000);
    }

    if (gameState){
        RFID_loop();
        if(RFID_func_call == 1){
            PlayerEnter();
        } else if (RFID_func_call == 2) {
            PlayerLeft();
        }
        GameLoop();
    }
}
```

Figur 30 - loop funktionen fra Main.ino

Den loop-funktion (Figur 30), der findes i Main.ino, består ikke af særlig meget. Dens formål er at lytte efter om knapperne bliver trykket på, starte spillet, lytte efter RFID_func_call og holde GameLoop funktionen kørende.

```
void PlayerEnter() {
    RFID_func_call = 0;
    ReaderIndex = fix_values(ReaderIndex, 1);

    Serial.print(Player);
    Serial.print(" has entered ");
    Serial.println(ReaderIndex);

    for(int i = 0; i < 3; i++){
        if(Player == blue[i]) {
            switch (ReaderIndex) {
                case 1:
                    LightUpArea(Arena_1, sizeof(Arena_1) / sizeof(int), BRIGHTNESS_ARENA, 0, 0, 255);
                    currentPlayer_1 = 1;
                    break;
                case 2:
                    LightUpArea(Arena_2, sizeof(Arena_1) / sizeof(int), BRIGHTNESS_ARENA, 0, 0, 255);
                    currentPlayer_2 = 1;
                    break;
                case 3:
                    LightUpArea(Arena_3, sizeof(Arena_1) / sizeof(int), BRIGHTNESS_ARENA, 0, 0, 255);
                    currentPlayer_3 = 1;
                    break;
                case 4:
                    LightUpArea(Arena_4, sizeof(Arena_1) / sizeof(int), BRIGHTNESS_ARENA, 0, 0, 255);
                    currentPlayer_4 = 1;
            }
        }
    }
}
```

Figur 31 - Udsnit af PlayerEnter funktionen fra Main.ino

PlayerEnter funktionen (*Figur 31*), bliver kaldt hver gang en ny spillerbrik træder ind på en arena, og den kaldes fra loop funktionen. Arenaernes indekser i koden er ikke stillet op på samme mådes om på det virkelige bræt. Derfor bruges fix_values funktionen i starten af PlayerEnter, til at fikse ReaderIndeks variablen, så den passer med indeksene på det virkelige bræt. Det næste i funktionen er fire for-loops, der indeholder næsten det samme på nær spiller-arrayet i if-statementet, og værdien som currentPlayer variablerne sættes til. Af den grund tages der kun udgangspunkt i det første for-loop her. Pointen med disse for-loops er at opdatere currentPlayer variablerne, så de reflekterer hvilket hold, der opholder den associeret arena. Lysene i arenaen ændres også til holdets farve, så spillerne får feedback, når de overtager en arena. Grunden til der er brugt et langt switch-statement her i stedet for endnu et for-loop, er at der ikke blev fundet en mulig måde at behandle et array med enums.

```
void PlayerLeft() {
    RFID_func_call = 0;
    ReaderIndex = fix_values(ReaderIndex, 2);
    Serial.print(ReaderIndex);
    Serial.println(" is now empty");
}
```

Figur 32 - PlayerLeft funktionen fra Main.ino

Funktionen, PlayerLeft (*Figur 32*), havde originalt mere indhold, men på grund af at reglerne blev skiftet undervejs, var PlayerLeft's funktioner ikke længere relevant. Derfor står den tom, uddover at skrive til serial monitoren at en spillerbrik har forladt en arena, hvilket var meget hjælpsomt under debugging.

RFID-readers'ne

For at forbedre immersion, flow og indlevelse, blev det besluttet at give brættet evnen til selv at tælle point op. Dette betød at brættet skulle have en måde at udregne hvilke spillere, der står på hvilke arenaer. Den oplagte måde at opnå dette på, er ved brug af RFID-readers. Disse RFID-readers kan nemlig sidde nede i bordet og registrere når RFID-knapper er på arenafelterne. Udover blot at tælle point, giver dette også mulighed for at give feedback til spilleren, ved brug af lys i arenaerne. Dette lys vil altså kunne skifte farve til den spillers farve, der er landet på arenaen.

Til dette projekt blev det valgt at bruge Parallax RFID-readers (Parallax, 2023). Ulemper ved at bruge disse RFID-readers frem for andre, er at Parallax RFID-readers bruger en 125kHz frekvens, i modsætning til de fleste Arduino RFID-readers, der en bruger 13,56Mhz frekvens. 125kHz frekvens RFID-readers er en ældre teknologi, og er generelt vurderet til at have dårligere sikkerhed, end de nyere RFID-readers (SEN News, 2018). Grunden til valget af Parallax RFID-readers, er bundet til valget af RFID-knapper. Der var nemlig mange RFID-knapper med 125kHz frekvens til rådighed. Derudover var disse RFID-knapper også 16 millimeter i diameter, hvilket passede perfekt til felternes diameter.

For at kontrollere RFID komponenterne, bruges der en variabeltype kaldt *SoftwareSerial*, der henholdsvis stammer fra et Arduino medfølgende library ved navn SoftwareSerial. Derfor importeres dette library som det allerførste i RFID-koden. Ved at initialisere en SoftwareSerial, giver det evne til at kommunikere med Arduino-tilkoblet komponenter via RX-pins (Arduino, 2023). SoftwareSerial bruge nemlig RX- og TX-pins til at kommunikere. RX-pins kan modtage data og TX-pins kan sende data. Da det ikke er nødvendigt at sende data til RFID-readers'ne, er de ikke forbundet med TX-pins. SoftwareSerial skal dog stadig have en nummeret på en TX-pin for at kunne initialiseres, så i dette tilfælde bliver de givet nummeret på en TX-pin, der ikke er forbundet til noget. Under udførte tests med adskillige RFID-komponenter tilkoblet til samme RX-pin, blev det opdaget at hver RX-pin kun kan håndtere maksimum tre RFID-komponenter. Da der i brætspillet er seks arenaer, der skal kunne tage brug af RFID-kommunikation, blev der taget et valg om at opdele komponenterne, så to RX-pins hver henholdsvis er tilsluttet til tre RFID-komponenter. Derfor er der også brug for at to SoftwareSerials skal initialiseres.

```
#include <SoftwareSerial.h>

//Parallax RFID Reader
int Readers[] = { 2, 3, 4, 5, 6, 7 };

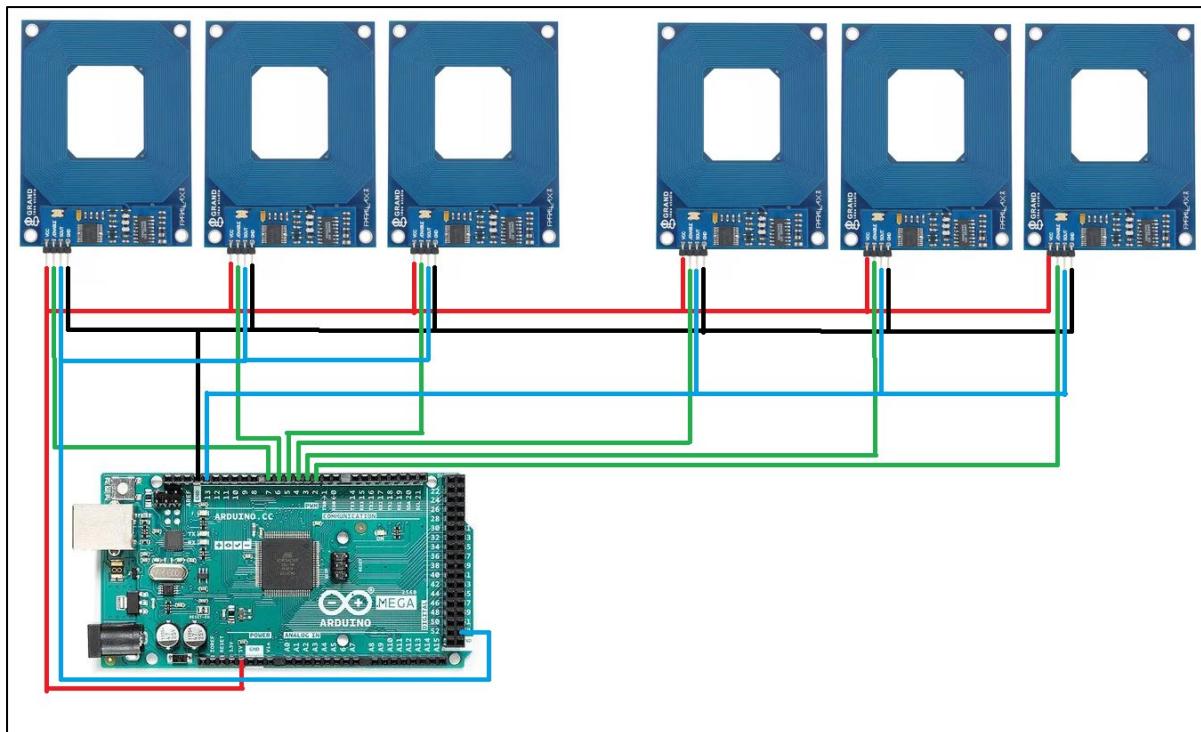
#define RFIDSerialRate 2400

#define RxPin 53
#define TxPin 50
SoftwareSerial RFIDReader(RxPin, TxPin);

SoftwareSerial RFIDReader_y(13, 12);
```

Figur 33 - Uddrag af *RFID.h*, der viser variabler

Udover at initialisere de to SoftwareSerials, oprettes der også en int array (*Figur 33*). Denne int array indeholder værdierne for hvordan /enable-pins'ne fra RFID-readers'ne er tilsluttet til Arduinoen.



Figur 34 - Schematic af RFID modulers opsætning med Arduino

Hver RFID-reader har fire pins, inklusiv en RX-pin til kommunikation med Arduinoen. De har selvfølgelig også en pin til strøm (4,5-5,5V) og en pin til ground. Derudover har hver RFID-reader en "/enable" pin. Opsætningen af alle disse pins kan ses ovenover på figur 34. Ved at sætte denne pin til at være en output pin i Arduino koden, kan de tilkoblet RFID-readers aktiveres og deaktiveres, baseret på mængden af strøm de bliver sendt. I tilfældet af de brugte Parallax RFID-readers, er de aktiveret når deres enable-pin modtager en *LOW* signalstyrke (<1,5V). Ligeledes deaktiveres RFID-readers'ne når deres enable-pin modtager en *HIGH* signalstyrke (5V). Det er vigtigt at kunne aktivere og deaktivere RFID-readers, da en enkelt SoftwareSerial kun virker hvis blot en enkelt af dens tilsluttede komponenter er aktiveret ad gangen. For at undgå dette problem bliver hver RFID-reader aktiveret i et halvt sekund på skift.

```
void RFID_setup()
{
    // RFID reader SOUT pin connected to Serial RX pin at 2400bps
    RFIDReader.begin(RFIDSerialRate);
    RFIDReader_y.begin(RFIDSerialRate);

    // Set Enable pin as OUTPUT to connect it to the RFID /ENABLE pin
    pinMode(2, OUTPUT);
    pinMode(3, OUTPUT);
    pinMode(4, OUTPUT);
    pinMode(5, OUTPUT);
    pinMode(6, OUTPUT);
    pinMode(7, OUTPUT);
}
```

Figur 35 - RFID_setup funktionen fra RFID.h

Funktionen, `RFID_setup` (*Figur 35*), bliver kaldt i `Main.ino`'s egen setup-funktion, hvilket vil sige funktionen bliver kaldt når Arduinoen starter op. Denne funktions funktioner er at starte de to SoftwareSerials, her ved navn "RFIDReader" og "RFIDReader_y". Derudover bliver pin 2 til 7 sat til at være *OUTPUT*. Pin 2 til 7 er de pins, som er tilsluttet til RFID-readers'nes /enable pins. Ved at sætte disse pins til *OUTPUT*, er det muligt at forbinde til hver RFID-readers /enable pin, hvilket henholdsvis gør det muligt at aktivere og deaktivere dem.

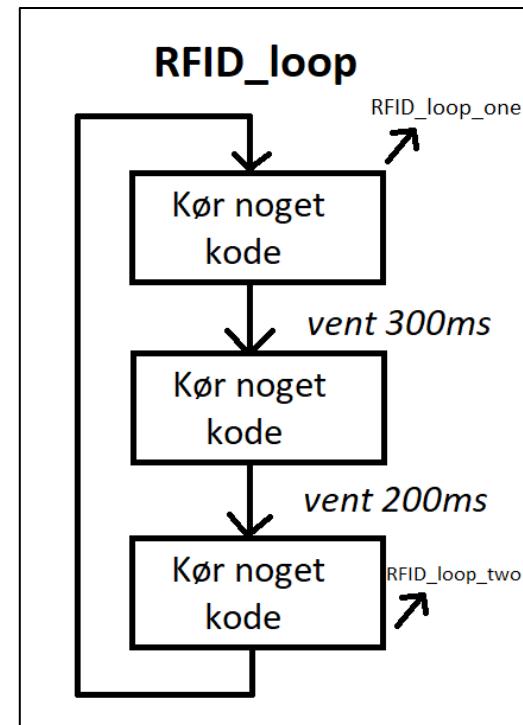
Funktion `RFID_loop` (*Figur 37*) bliver kørt konstant, da den bliver kaldt i `Main.ino`'s loop funktion. `RFID_loop`'s funktion er at hele tiden skifte hvilken RFID-reader, der er aktiv, samtidig med at tjekke om en RFID-knap befinner sig ovenpå en af arenaerne. For at opnå dette, bliver der nødt til at blive brugt noget ventetid i koden, da hver RFID-reader skal have tid til at opfange RFID-signaler. Et diagram over denne `RFID_loop` funktion kan ses på figur 36. Normalvis ville der i sådan en situation tages brug af Arduinos egen delay funktion. Denne delay funktion kan dog ikke bruges her, da alt andet kode bliver sat på pause, ved brug af funktionen. Dette forhindrer knapperne på brættet i at virke, da deres kode tjekker hvert tick om de er blevet trykket på. Af den grund var der brug for en anden metode at introducere ventetid mellem funktionens funktioner.

```
unsigned long LastMilli = 0;
bool firstRun = true;
bool firstRunTwo = true;
void RFID_loop()
{
    if (firstRun) {
        firstRun = false;

        RFID_loop_one();
    }
    if (millis() - LastMilli > 300 && firstRunTwo) {
        firstRunTwo = false;

        for (int i = 1; i < 7; i++) {
            digitalWrite(i + 1, HIGH);
        }
    }
    if (millis() - LastMilli > 500) {
        firstRun = true;
        firstRunTwo = true;
        LastMilli = millis();

        RFID_loop_Two();
    }
}
```



Figur 37 - `RFID_loop` funktionen fra `RFID.h`

Figur 36 - Diagram af `RFID_loop` funktionen

Løsningen til dette problem er at tælle antallet af millisekunder, der er forløbet siden sidste funktionskald. Dette gøres ved at oprette en variabel, i dette tilfælde `LastMilli`, som i slutningen af loopet bliver sat til antal millisekunder siden Arduinoen startede. Antal millisekunder siden Arduinoen startede, kan findes ved at kalde den indbygget Arduino funktion ved navn `millis`. Ved at trække `LastMilli` og `millis` fra hinanden, fås antal tid siden sidste funktionskald. If-statements kan således bruges til at køre kode, når tiden siden sidste funktionskald rammer enten 300 eller 500 millisekunder. To booleans bruges til at sørge for, at denne kode kun bliver kørt en enkelt gang per loop. Koden der køres i denne funktion, er opdelt i to separate funktioner, for at opnå god kodestruktur og læsbarhed. Disse funktioner hedder `RFID_loop_One` og `RFID_loop_Two`.

```
void RFID_loop_One() {
    if (loopNr == 1 || loopNr == 2 || loopNr == 3) {
        RFIDReader.end();
        RFIDReader_y.begin(RFIDSerialRate);
    }
    else if (loopNr == 4 || loopNr == 5 || loopNr == 6) {
        RFIDReader.begin(RFIDSerialRate);
        RFIDReader_y.end();
    }

    for (int i = 1; i < 7; i++) {
        if (i == loopNr) {
            digitalWrite(i + 1, LOW);
        }
        else {
            digitalWrite(i + 1, HIGH);
        }
    }
}
```

Figur 38 - RFID_loop_One funktionen fra RFID.h

RFID_loop_One funktionen (Figur 38) er relativ simpel. Denne funktions formål er at starte den rigtige SoftwareSerial, og derefter gøre så kun den rigtige RFID-reader er aktiv. Dette er alt baseret på en integer variabel ved navn loopNr, der går fra 1 til 6, og skifter ved enden af hvert loop. De to SoftwareSerials bliver skiftet mellem at være tændt og slukket, da der ikke kunne findes nogen anden måde at have to aktive SoftwareSerials i samme projekt.

```
int loopNr = 0;
int LoopTimesSinceLastData[] = { 99, 99, 99, 99, 99, 99, 99 };
String PlayerStandingLastData[] = { "", "", "", "", "", "", "" };
```

Figur 39 - Uddrag af RFID.h, der viser tre variabler

For at vide om en spiller står på en arena, er der brug for at vide om en spiller har trådt ind på arenaen, og om en spiller har forladt arenaen. RFID-readers'ne ved ikke noget af dette. Den eneste information RFID-readers'ne kan give, er hvorvidt de i dette sekund kan oprette forbindelse til en RFID-knap. For at udregne denne information, bliver der nødt til at blive holdt styr på hvor lang tid det var siden en arena sidst modtog spiller data. Derfor bliver int-arrayet LoopTimesSinceLastData oprettet (Figur 39). Dette array holder styr på hvor mange loops der er gået siden arenaerne sidst modtog data. Alt dette betyder, at hvis en arena ikke har modtaget data i lang tid og lige pludselig modtager data igen, er en ny spiller højst sandsynligt nu er trådt ind på arenaen. Det betyder også, at hvis en arena ikke har modtaget data i lang tid, betyder det højst sandsynligt at spilleren på den arena, har forladt arenaen. Der bliver også oprettet et string-array, der holder styr på den sidste spiller, der stod på en arena. Dette array blev skabt, så programmet kan kende forskel, på en spiller der står på en arena, og på en spiller, der hurtigt bliver skiftet ud med en anden spiller på en arena.

```
void RFID_loop_Two() {
    loopNr++;
    if (loopNr > 6) {
        loopNr = 1;
    }

    if (RFIDReader.available() > 0 || RFIDReader_y.available() > 0) {
        if (RFIDReader.available() > 0) {
            ReadSerial(RFIDReader);
        }
        else {
            ReadSerial(RFIDReader_y);
        }
        if (LoopTimesSinceLastData[loopNr] >= 20 || PlayerStandingLastData[loopNr] != TagCode) {
            ReaderIndex = loopNr;
            Player = TagCode;
            RFID_func_call = 1;
        }
        PlayerStandingLastData[loopNr] = TagCode;
        LoopTimesSinceLastData[loopNr] = 0;
    }

    for (int i = 1; i < 7; i++) {
        LoopTimesSinceLastData[i]++;
        if (LoopTimesSinceLastData[i] == 20) {
            RFID_func_call = 2;
            ReaderIndex = loopNr;
        }
    }
}
```

Figur 40 - RFID_loop_Two funktionen fra RFID.h

Funktionen, RFID_loop_Two (*Figur 40*), starter med at skifte variablen loopNr til det næste tal i sekvensen. Derefter bliver der tjekket om der er nyt RFID-data tilgængeligt. Hvis nyt data er tilgængeligt bliver det læst. Der bliver tjekket om en ny spiller er trådt ind på arenaen, og hvis dette er tilfældet, sættes de passende variabler, herunder RFID_func_call. Til sidst går et for-loop gennem alle arenaerne og opdaterer deres tidsvariabler i LoopTimesSinceLastData. Hvis denne variabel er mere end 20, bliver RFID_func_call opdateret, så den reflekterer at en spiller har forladt en arena. ReaderIndeks opdateres også med information om hvilken arena, dette er sket ved.

LED'erne

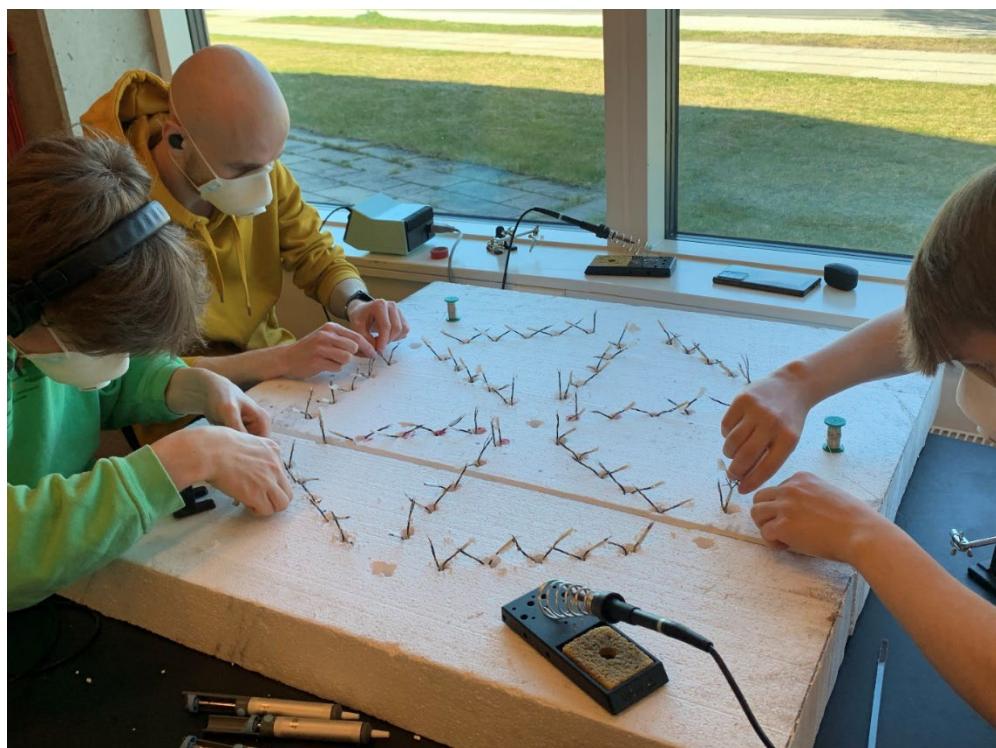
Grundet måden håndteringen af LED'er via library'et " Adafruit_NeoPixel.h" er opsat, er det nødvendigt at angive indekset af den LED der skal tændes. Derfor er alle de forskellige grupperinger af LED segmenter, struktureret af arrays. Dette skyldes også hvordan LED'erne er forbundet med hinanden. For at strømline produktionen af brættet, valgte gruppen at lodde alle LED'erne med samme længde, som efterfølgende kunne loddes sammen på bagsiden af flamingopladen. Dette var gjort, for at reducere den mængde tid der kunne være brugt til at måle alle ledninger før de kunne skæres, afisoleres og loddes. Dog resulterede denne proces i enkelte ledninger der var for korte, men det er af gruppens overbevisning, at en del tid var sparret af denne metode.

Som set nedenfor (*Figur 41*), blev hundredvis af ledninger skåret til den rette længde, afisolert og loddet til en singulær LED. For hver LED, var det nødvendigt at klippe to strøm, jord og DIN (Den pin der videregiver signalet fra tidligere LED) ledninger, hvilket svarer til seks ledninger pr. LED. Da projektet krævede 102 LED'er, svarer det til 612 ledninger. Hver af disse ledninger, skulle derefter loddes sammen, for at skabe en seriell forbindelse fra LED til LED.



Figur 41 - Billede af udskåret ledninger til LED lodning

Som resultat af flamingopladens størrelse, var det ikke muligt at foretage lodningerne under udsugningsstationerne i HackLab lokalet. Derfor var det nødvendigt at flytte loddestationerne til en lokation med nok plads. Dog var en lokation med plads og udsugning ikke tilgængelig, hvilket resulterede i nødvendigheden for at bærer respirationsmasker under lodningen for at undgå den usunde indånding af gasserne udledt fra loddejernet under lodningen (*Figur 42*).



Figur 42 - Billede af LED lodningsprocessen

Dette indikerer også, hvorfor det er nødvendigt at benytte arrays til inddelingen af LED'erne, fordi LED'erne ikke er loddet sammen på baggrund af deres position på LED kæden, men pga. deres beliggenhed ift. hinanden. Derfor er koden struktureret omkring aktivering af arrays af LED'er, frem for individuelle LED'er. Arrays'ne er opbygget på fokusområder, der er relevante at skifte til samme farve. På nedenstående objekt (*Figur 43*), vises et udsnit fra koden, der fremviser opsætningen af disse arrays. Som kan ses, oprettes et array ved navn "Arena_0", og er tildelt LED nr. 20 og 21 i kæden. I dette tilfælde er LED'erne lige ved siden af hinanden, men som det kan ses ved "Arena_1", er LED'erne ikke placeret lige ved siden af hinanden, pga. opsætningen under lodningen af LED'erne.

```
// Arenas
int Arena_0[] = {20, 21};
int Arena_1[] = {30, 36};
int Arena_2[] = {42, 48};
int Arena_3[] = {54, 60};
int Arena_4[] = {66, 72};
int Arena_5[] = {78, 84};
int Arena_6[] = {90, 96};
```

Figur 43 - Udsnit fra array variablerne i "LED.h" scriptet

Grundet brugen af arrays til aktiveringens af LED'erne, er størstedelen af LED.h koden, for det meste, opbygget på én funktion. Funktionen "LightUpArea" modtager et array med LED'er, samt en størrelse på arrayet, en lysstyrke og en RGB-værdi til farven af lyset som parametre til funktionen (Figur 44).

```
void LightUpArea(int array[], int size, int brightness, int r, int g, int b){
    for (int i = 0; i < size; i++) {
        NeoPixel.setBrightness(brightness);
        NeoPixel.setPixelColor(array[i], NeoPixel.Color(r, g, b));
        NeoPixel.show();
    }
}
```

Figur 44 - Udsnit af funktionen "LightUpArea" fra "LED.h" scriptet

I funktionen benyttes et for loop, hvilket er en operation der gentager sig selv x antal gange. Dette specifikke for-loop kører "size" gange, hvor "size" repræsenterer størrelsen af arrayet. På denne måde kan hver indeks i arrayet tilgås. Her bliver lysstyrken kontrolleret som det allerkørste. Dette er for at gøre det muligt at skifte lysstyrken på lyskæden til enhver tid, eftersom NeoPixels ".setBrightness()" funktion ændrer lysstyrken på hele lyskæden. Dog var det ikke muligt at bruge denne funktion optimalt, da systemet havde en mangel på strøm, hvilket gjorde at når LED'erne havde en lysstyrke højere end omkring 1, fik RFID-readers'ne ikke tilført nok strøm, til at kunne læse spillerbrikkerne.

Efter lysstyrken er sat, bliver farven af den specifikke LED sat. Dette er gjort via NeoPixels ".setPixelColor()" funktion. Denne funktion kræver en int, som svarer til indekset af den LED der skal ændres, samt kræver den en farve. Denne farve bliver også tilført via en NeoPixel funktion, da Arduinos kodesprog ikke har en indbygget variabel til farve. Dette er også derfor "LightUpArea" funktionen kræver en RGB-værdi. Derudover bliver arrayet, som er blevet sendt som parametre, tilgået til i dens indeks. Dette er fordi, som tidligere vist, indeholder arrayet indekset til de specifikke LED'er der er tildelt specifikke veje, arenaer, områder, osv.

Til sidst er det påkrævet at bruge NeoPixels ".show()" funktion, til at sende signalet til LED'erne og skifte farven, samt lysstyrken. Denne funktions formål, er at fastlægge ændringerne til LED'erne. På nedenstående objekt (Figur 45), kan det ses hvordan "LightUpArea" funktionen anvendes.

```
LightUpArea(Spiller_1, sizeof(Spiller_1) / sizeof(int), BRIGHTNESS_PLAYER, 0, 0,
255);
LightUpArea(Spiller_2, sizeof(Spiller_2) / sizeof(int), BRIGHTNESS_PLAYER, 150,
0, 139);
LightUpArea(Spiller_3, sizeof(Spiller_3) / sizeof(int), BRIGHTNESS_PLAYER, 255,
65, 0);
LightUpArea(Spiller_4, sizeof(Spiller_4) / sizeof(int), BRIGHTNESS_PLAYER, 255,
175, 0);
```

Figur 45 - Udsnit af eksempel på brugen af "LightUpArea" funktionen fra "LED.h" scriptet

De sidste bemærkselsværdige funktion i LED.h scriptet, er "WinSequence" og "ChangeAllLEDs" - funktionen (*Figur 46*). Disse funktioner styrer hvad der sker, når spillet bliver vundet. "ChangeAllLEDs" funktionen virker på samme måde som den tidligere nævnte "LightUpArea" funktion, dog med nogle få ændringer. Både lysstyrken og arrayet er fjernet som parametre og er i stedet sat som statiske variabler i funktionen. Her bruges AllLEDs arrayet, som indeholder indekset til alle LED'er, samt er lysstyrken sat til 10. Eftersom spillet er ovre, kan lysstyrken øges, da spillet ikke længere er i gang, og det har derfor ikke brug for at læse RFID signaler.

Den overordnet forskel fra "LightUpArea" funktionen, er det delay der er placeret efter for loop'et. Dette delay, gør det muligt at blinke lysene i tilfældet af, at spillet bliver uafgjort. Her er det nødvendigt, at alle spillene som har 10 eller flere points kan vises som del af denne sekvens. Denne sekvens er kontrolleret af "WinSequence" funktionen. Denne funktion er simpel: Den modtager et iterationsantal som parametre, hvorefter den kører et for-loop det antal gange, hvori den kører "ChangeAllLEDs" funktionen med farven til de spillere der har vundet. Dette gør den ved et if-statement, som afgør om en spiller har nok points. Her er benyttet et if-statement frem for et else-if-statement, da det er nødvendigt for funktionen, at alle spilleres lys kan blinke, når funktionen bliver kaldt. Med fire if-statements, bliver hver spillers lys tændt, på skift. Først spiller 1, så 2, osv.

```
void ChangeAllLEDs(int r, int g, int b){  
    for (int i = 0; i < sizeof(AllLEDs) / sizeof(int); i++) {  
        NeoPixel.setBrightness(10);  
        NeoPixel.setPixelColor(AllLEDs[i], NeoPixel.Color(r, g, b));  
        NeoPixel.show();  
    }  
    delay(250);  
}  
  
void WinSequence(int iterations){  
    for (int i = 0; i < iterations; i++) {  
        if (player_1_score >= 10){  
            ChangeAllLEDs(0, 0, 255);  
        }  
        if (player_2_score >= 10){  
            ChangeAllLEDs(150, 0, 139);  
        }  
        if (player_3_score >= 10){  
            ChangeAllLEDs(255, 65, 0);  
        }  
        if (player_4_score >= 10){  
            ChangeAllLEDs(255, 175, 0);  
        }  
    }  
}
```

Figur 46 - Udsnit af "ChangeAllLEDs" og "WinSequence" funktionerne fra "LED.h" scriptet

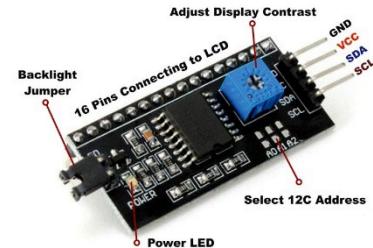
Displays'ne

For at vise hver spillers score i spillet, var det nødvendigt at benytte et display for hver spiller. Displays formål, er både at indikere hvilken spiller tilhører hvilken farve, hvilket også er derfor displays'ne starter med at skrive "Player x" på første linje, hvor x er spiller nummeret. Derudover er det også hensigten, at displayet skal fremvise spillerens score der, som tidligere nævnt, bliver kontrolleret af RFID-readers'ne. Dette bliver gjort på anden linje af displayet, lige under spiller nummeret. Dog var et grundlæggende problem under konstruktionsprocessen, at mange af vores komponenter benyttede såkaldte Rx- og Tx-pins. Dette ville have været et problem, hvis vil tilknyttede displayet direkte til Arduinoen. Men da det ikke var muligt, da mange pins allerede var

brugt af andre komponenter. Derfor benyttede vi os af fire I2C Serial Interface Adapter moduler (Campbell, 2023). Disse moduler gør det muligt at forbinde alle fire displays til én Rx og én Tx pin. Grundet serielforbindelsen, er det muligt for alle fire displays at kommunikere på samme bus. En bus virker, ved at en central kommunikations linje transporterer data fra komponent til komponent. Dette virker på et adresesystem, hvor en master (Arduinoen) sender et signal med en specifik adresse, der fortæller komponenten med den specifikke adresse, hvad den skal gøre.

På denne måde, er det muligt at have fire displays tilsluttet kun to pins, samt strøm og jord. For at ændre adressen på Adapter modulet, er en række af tre paneler (A0, A1 og A2) placeret på siden af modulet (*Figur 47, specifikt "Select 12C Address"*), som kan loddes sammen, til at give op til otte forskellige kombinationer. Hver af disse konfigurerer modulet med en specifik adresse.

Til dette projekt er adresserne 0x27, 0x26, 0x25 og 0x24 valgt. En tabel over de forskellige adressekombinationer kan ses på nedenstående billede (*Figur 48*).



Figur 47 - Diagram af Adapter modulet taget fra (Components101, 2021)

A0	A1	A2	Address
Open	Open	Open	0x27
Jumper	Open	Open	0x26
Open	Jumper	Open	0x25
Jumper	Jumper	Open	0x24
Open	Open	Jumper	0x23
Jumper	Open	Jumper	0x22
Open	Jumper	Jumper	0x21
Jumper	Jumper	Jumper	0x20

Figur 48 - Screenshot af adressetabel til Adapter modulet taget fra (Components101, 2021)

Koden til displaysne er simpel. Først oprettes de fire displays via "LiquidCrystal_I2C.h" library'et. På nedenstående objekt (*Figur 49*), er det muligt at se, hvordan hvert display oprettes med hver deres adresse samt to andre parametre. Disse parametre er antallet af rækker og kolonner. Til projektet har gruppen valgt at benytte standard 16x2 LCD-displays, hvilket er grunden til den størrelse, som er angivet som parametre til hver display.

```
LiquidCrystal_I2C lcd_Spiller_1(0x27, 16, 2);
LiquidCrystal_I2C lcd_Spiller_2(0x26, 16, 2);
LiquidCrystal_I2C lcd_Spiller_3(0x25, 16, 2);
LiquidCrystal_I2C lcd_Spiller_4(0x24, 16, 2);
```

Figur 49 - Udsnit af oprettelsen af displays'ne fra "Displays.h" scriptet

Herefter skal de fire displays sættes op. De har hver en setup sekvens, hvor de aktiveres. Herunder kan der ses de fem linjer kode brugt til at sætte hvert display op (*Figur 50*), som er identiske med undtagelse af navnet på variablen. Funktionen ".begin()" starter displayet op, hvorefter funktionen ".clear()" rydder displayet da scoren fra et tidligere spil, kunne være tilbageværende. Så tændes displayets backlight med ".backlight()" funktionen. Dette er for at gøre det nemmere at læse teksten

på skærmen. På de sidste to liner sættes skrivepositionen til (0,0), for efterfølgende at skrive spillerens nummer med "setCursor()" og "print()" funktionerne.

```
lcd_Spiller_1.begin();
lcd_Spiller_1.clear();
lcd_Spiller_1.backlight();
lcd_Spiller_1.setCursor(0, 0);
lcd_Spiller_1.print("Player 1");
```

Figur 50 - Udsnit af display opsætning til spillere 1 taget fra "Displays.h" scriptet

Det samme sker for de tre andre spillere. Til sidst afrundes setup funktionen med at køre "UpdateDisplays" funktionen. Her gentages alle linjer, igen, for hver spiller (Figur 51). Funktionen bliver, for nemheds skyld, kørt hvert loop. Igen bruges funktionen til at flytte skrivepositionen, samt skriver den spillerens score, ud fra spillerens score variabel.

```
lcd_Spiller_1.setCursor(0, 1);
lcd_Spiller_1.print("Score: ");
lcd_Spiller_1.print(player_1_score);
```

Figur 51 - Udsnit af opdateringen af spiller 1's display taget fra funktionen "UpdateDisplays" fra "Displays.h" scriptet

GameSequence.h scriptet

I dette script findes "gameLoop" funktionen. Denne funktion startes i "Main.ino" scriptet, når en knap trykkes og gentages efterfølgende hvert loop. "gameLoop" funktionen er den funktion, der styrer spillet, når det er sat i gang. På bordet er der placeret en knap på hver side af bordet foran spillernes kortholdere. Disse knapper er til at starte spillet, samt for at afslutte ens tur. Originalt var det hensigten, at knappen skulle give visuelt feedback, når den blev trykket, men grundet tidsmangel, blev dette ikke implementeret. I starten af "gameLoop" funktionen kontrollerer et if-statement, om knapperne er blevet trykket på. Hvis alle knapperne er trykket ned og sluppet igen, øges turen, som set nedenfor (Figur 52).

```
if(Button_1.isReleased() || Button_2.isReleased()) {
    Turn++;
}
```

Figur 52 - Udsnit af tjek af nedtrykningen af knapperne fra "GameSequence.h" scriptet

Hvis turen går over 12, har alle spillere spillet deres kort, hvilket vil sige, at runden er ovre. Derfor sættes turen til 1, samt bliver runden øget med 1. Eftersom runden er ovre, skal brætte udregne points til hver spiller. Dette gøres med seks switch-statements: Et for hver arena felt. Nedenfor (Figur 54) er et statement udvalgt, da de andre er identiske bortset fra variabel navnet "currentPlayer_1", som er arenafeltet 1's værdi. currentPlayer_1 indeholder en enum, oprette under "variables.h" scriptet, med spillernes respektive farver. Her er 0 = None, 1 = Blue, 2 = Purple, 3 = Orange og 4 = Yellow. Da det er en enum-værdi, kan spilleren svarende til nummeret indeholdt af currentPlayer_1, bruges til at øge spillernes score. På denne måde udregnes scoren for hvert arenafelt. Til sidst opdateres displays'ne med de nye scores.

```
if(Turn > 12){  
    Turn = 1;  
    Round++;  
  
    switch(currentPlayer_1){  
        case 1:  
            player_1_score++;  
            break;  
        case 2:  
            player_2_score++;  
            break;  
        case 3:  
            player_3_score++;  
            break;  
        case 4:  
            player_4_score++;  
            break;  
    }  
  
    UpdateDisplays();  
}
```

Figur 53 - Udsnit af arenafelt 1's håndtering af playerscore i slutningen af runden fra "GameSequence.h" scriptet

Det sidste der tjekkes i game-loopet er, om der er nogle spillere, som har vundet (Figur 54). Dette gøres ved først at tjekke om spillet er i gang ved at spørge, om gameState er true. gameState er true, når spillet er i gang, og false når spillet ikke er i gang. Derefter tjekkes der, om enhver af spillernes score er lig med eller over 10. Hvis begge krav er sande, initieres "WinSequence" med tre iterationer som parametre, hvorefter spillet bliver genstartet ved at sætte gameState til false og sætte lysne tilbage til standard med funktionen "LightBoard()".

```
if (gameState && (player_1_score >= 10  
    || player_2_score >= 10  
    || player_3_score >= 10  
    || player_4_score >= 10)){  
    WinSequence(3);  
    gameState = false;  
    LightBoard();  
}
```

Figur 54 - Udsnit af tjekket om en spiller har vundet fra "GameSequence.h" scriptet fra "gameLoop" funktionen

Højtaleren

Først var det planlagt, at højtaleren skulle installeres i bordet og forbindes til samme Arduino som alle andre komponenter i brættet. Men som tidligere beskrevet, var det problematisk at få strøm nok til RFID-readers'ne, så de kunne læse spillernes brikker. Derfor var det ikke muligt at spille lyd højt nok til at høre samtidig med at spille virkede. Derfor tog gruppen en beslutning om at tilslutte højtaleren til en dedikeret Arduino, som kun styrede højtaleren. Eftersom højtalerenens eneste formål var at gentage det samme baggrundsmusik, var det ikke nødvendigt at kunne kommunikere med den anden Arduino. Derfor blev denne Arduino tilsluttet et 9 volts batteri, samt kode til at gentage samme lydfil for evigt blev uploaded.

For at styre højtaleren brugte vi et DFPlayerMini modul. En DFPlayerMini er et MP3-modul med et indbygget SD-kort slot til opbevaring og håndtering af MP3 lydfiler. Samtidig med at have indbygget SD-kort slot, har den også indbygget amplifier, samt forskellige equalizer muligheder til justering af lyden (DFRobot, 2023).

DFPlayerMini'en kræver to libraries: "SoftwareSerial.h" og "DFRobotDFPlayerMini.h". Da højtaleren har en separat Arduino, er alle pins ledige. Derfor bliver pins 10 og 11 brugt til at oprette en softwareSerial, til at kommunikere med playeren. Scriptet indeholder kun to elementer. Først bliver playeren oprettet og initieret i setup funktionen, hvorefter der kontrolleres om playeren er opsat korrekt, ved et if-statement der spørger "player.begin(softwareSerial)", der returnerer true, hvis den er påbegyndt korrekt. Hvis ikke logges det til serial-skærmen, at den ikke er opsat korrekt. Når den er opsat korrekt, bliver lydstyrken sat til et passende niveau fra 0-30, samt bliver musikken påbegyndt med ".play()" funktionen. ".play()" funktionen modtager et indeks til sangen på SD-kortet. Det er dog ikke muligt at afspille en lydfil ud fra navnet på filen, men dette er ikke et problem, da der kun er en enkelt lydfil, der gentages for evigt.

```
Serial.begin(9600);
softwareSerial.begin(9600);

if (player.begin(softwareSerial)) {
    delay(1000);
    player.volume(28);
    player.play(1);
}
else {
    Serial.println("Connecting to DFPlayer Mini failed!");
    delay(100000);
}

delay(1000);
```

Figur 55 - Udsnit af "setup" funktionen fra "Højtaler.ino" scriptet

Efterfølgende bliver der i loopet udført et andet element af scriptet (*Figur 56*). Her bliver lydklippen gentaget. Pin 7 er også tilsluttet DFPlayerMini'en, til en pin kaldet "BUSY". Denne pin returnerer 0 (LOW), hvis en lydfil afspilles og 1 (HIGH), hvis ikke. Derfor, kan dette if-statement tjekke, hvornår lydfilen afsluttes og påbegyndes igen.

```
if (digitalRead(7) == HIGH) {
    player.play(1);
}
```

Figur 56 - Udsnit af hvordan lyd gentages fra "loop" funktionen fra "Højtaler.ino" scriptet

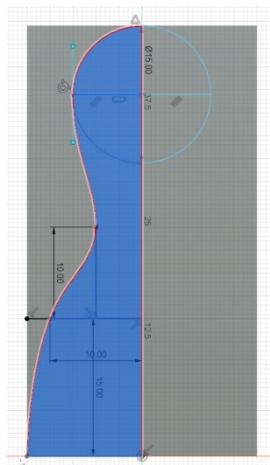
Produktion af spillerbrikker

Spillerbrikkens design er baseret ud fra et standard meeple-design, ligesom spillerbrikken fra spillet Ludo (*Figur 57*). Figurens design havde nogle specifikke krav. Bl.a. skulle den være håndterbar, så spilleren nemt kunne samle den op og flytte den rundt på brættet. Derudover skulle den have en udhuling i bunden, sådan at der var plads til en RFID-brik, til at scanne spillernes brikker, når de lander på en af de seks forskellige arenafelter på brættet.

Spilleren er designet ud fra følgende skitse i Fusion 360 (*Figur 58*), hvorefter skitse er roteret 360 grader omkring den horisontale akse. Dette har resulteret i en symmetrisk figur lignende en standard meeple (*Figur 59*).



Figur 57 - Billede af en ludobrik taget fra (Berger, 2012)



Figur 58 - Udklip af skitse i Fusion 360 til spillerens design



Figur 59 - 3D-modellen til spillerbrikken

Produktion af kort

Efter at have designet kortene og deres egenskaber var det tid til at producere nogle fysiske kort. Kortene blev printet i størrelsen 63.5 x 88.9mm, da det er én af standardstørrelserne for kort, og f.eks. Magic the Gathering-kort bruger denne størrelse (Draftsim, 2019). Kortene blev indsat i et Word-dokument og printet i rækker og kolonner på almindeligt A3-printerpapir. For at gøre kortene mere holdbare og give spilleren en bedre taktil oplevelse, blev de sat i Magic-kortlommer med et almindeligt spillekort bag hvert kort (*Figur 60*).



Figur 60 - Kort i lommer

Produktion af lyd

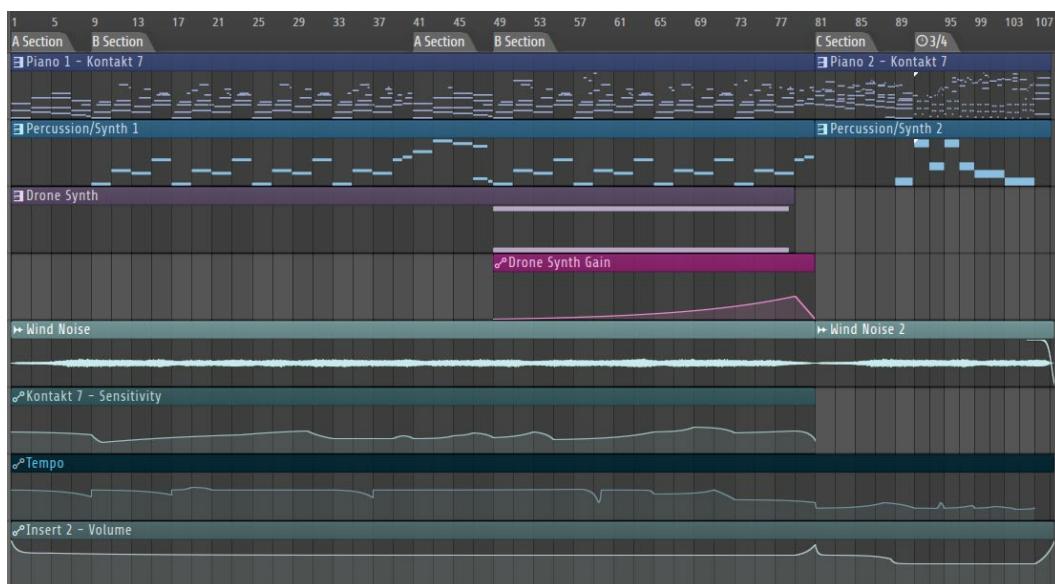
Efter at have besluttet at lave et minimalistisk lydspor centreret rundt om klaveret, skulle selve produktionen af lyden også sættes i gang. Lydsporet blev komponeret og produceret i musiksoftwaren FL Studio, da det er det program, som lyddesigneren i gruppen (*Mikkel Erikstrup*) var mest erfaren i. Samtidig er det også en komplet løsning til lyd og musik, der lader brugeren skrive musik i MIDI-format via forskellige digitale instrumenter, arrangere kompositionen af forskellige spor, og mixe og master lyden så den er klar til brug (Image Line, 2023).



Figur 61 - Overblik over klaverstykket

Den første og vigtigste del i produktionen af lydsporet var selvfølgelig klaverstykket. Oprindeligt blev det skrevet i to sektioner: et kort A-stykke, der er rytmisk simpelt og tjener som en form for pause når lydsporet looper, og et længere B-stykke, der har en relativt mere indviklet, men stadig minimalistisk sans for rytmekombinationer og melodier. For at gøre det mere varieret, gentages hver sektion to gange med små ændringer i melodien. Efter en intern testrunde i gruppen, var der bred enighed om at der stadig mangede noget variation for at holde loopet interessant i længere tid. Derfor blev der skrevet et C-stykke, der spiller efter hver anden gentagelse af B-stykket (Figur 61).

Endnu en ting der skaber variation, er tonaliteten af de forskellige stykker. A-stykket er ikke grundet i nogen bestemt toneart, og har derfor så at sige ikke noget sted i akkordprogressionen, der føles som "hjem", hvilket bidrager til den svævende stemning. B-stykket er til gengæld hovedsagelig baseret i tonearten B mol, som har en forholdsvis dyster tonalitet (Hamann, Mol-akkord på I-trin, 2020). I C-stykket skifter tonearten til G lydisk, som i forhold til resten af klaverstykket har en lys, let lyd (Hamann, Dur-akkord på I-trin, 2020). Sammen med den langsomme vals i $\frac{3}{4}$ -rytme, der udspiller sig i anden halvdel af C-stykket, giver dette et løft fra den til tider mørke stemning i de andre sektioner. Det betyder, at loopet ikke ligger i én stemning hele tiden, men bevæger sig rundt - en fordel, når musikken ikke er dynamisk, da det så betyder at loopet føles knap så ensformigt.



Figur 62 - Komposition af lydsporet

Efter at have komponeret klaverstykket, blev de andre lag i musikstykket tilrettelagt (*Figur 62*). For at forbinde det tematisk med spilbrættets naturbaserede udtryk, blev der lagt vindlyde ind over. Vindlydene er taget fra Splice, som er en abonnementstjeneste hvor brugere kan finde lyde til personligt og kommercielt brug (Splice, 2023). Derudover blev der tilføjet et synthesizerlag, der spiller hver akkords grundtone og indeholder en raslende percussionlyd, der passer til lyden af vinden. Det sidste lag af lyd, der blev tilføjet, er en synthesizer, der droner på de samme to toner og progressivt bliver højere gennem den anden gentagelse af B-stykket - igen for at skabe varians mellem gentagelserne. Desuden blev der tilføjet en ekkoeffekt på klaveret, der fylder lydbilledet lidt mere ud. De tre nederste spor på figur 62 er automatiseringer, der styrer parametre i bestemte lag af lyden. Tryksensitiviteten af det digitale klaver er én af dem - det medvirker til at gøre lyden mere menneskelig, da et normalt menneske ikke vil trykke på alle tangenterne med præcis samme mængde pres; desuden skaber det varierende intensitet gennem musikken. Det samme gælder tempoet af musikken, der varierer lidt over tid, og bliver en del langsmmere i C-stykket. Det gør musikken mere troværdig, da solomusikere typisk vil variere deres tempo efter fornemmelse i stedet for det faste tempo i rytmisk musik eller samspil. Den sidste automatisering styrer volumen af ekko-effekten på klaveret, der også skaber hvid støj. Støjen intensiveres omkring overgangen mellem B- og C-stykket og loopet tilbage fra C til B, for at lave en mere flydende transition. Efter at have produceret resten af musikken, blev lyden mixet, og var dermed klar til at blive brugt til spillet (Erikstrup, 2023).

For at opbevare og afspille lyden valgte vi at bruge en kombination af en DF-player Mini, der kan opbevare og håndtere lyden på et SD-kort, og en standard 4-ohms højtalere, da dette er en løsning vi har brugt flere andre gange i forbindelse med lignende projekter, der skulle bruge lyd. Højtaleren blev installeret i siden af bordet skjult bag én af træpladerne, så det ikke tog spilleren ud af oplevelsen.

Test af spil

Her vil der blive beskrevet og vist de forskellige test der er blevet udført i løbet af udviklingen af brætspillet. Dette inkluderer både test fra gruppens først valgte idé samt den endelige idé.

Første test af skattespillet

I starten af udviklingen af den første idé var det idømt relevant at teste hvordan brættet var at interagere med og hvordan det at stå op og bevæge sig rundt om brættet påvirkede indlevelsen. Dette blev gjort med en observationstest efterfulgt af et gruppeinterview. Under testen var spillet i en meget tidlig del af udviklingsprocessen så de fleste regler var endnu ikke lavet eller indført i spillet. Selve udformningen af spillet var også meget primitiv og midlertidigt (*Figur 63*). Eftersom testen fokuserede på hvordan spillerne oplevede rent fysisk at gå rundt om bordet, samt hvordan spillerbrikkerne bevægede sig på brættet, blev det vurderet at en test af spillet godt kunne fungere på trods af de manglende regler.

Observationstest som metode er valgt for forhåbentligt at kunne få de mest realistiske data. Metoden fungerer ved at fremskynde en så virkelighedsnær situation som muligt, hvor det er muligt at observere så præcist som muligt hvordan produktet vil blive brugt. Helt ideelt ville test scenariet efterligne den reelle brugssituation så meget som muligt for at få de mest præcise data (Hvass-Raun & Dalsaa, 2023).

Direkte efter observationstesten blev spiltesterne utsat for et gruppeinterview. Interviewet bliver holdt i en gruppe for at give mulighed for diskussion og forhåbentligt få noget data der reflekterer interaktionen mellem både spillerne og brættet, men også spillerne og spillerne med hensyn til at bevæge sig rundt om bordet. Gruppe interview formatet giver også mulighed for at spiltesterne kan reflektere over hinandens svar og bygge videre på dem. Formatet betyder dog også at der kan være data som ikke fremkommer i interviewet hvis en spiltester måske er ukomfortabel med at stå ved det i en større gruppe. Interviewet var semistruktureret for med de åbne spørgsmål at understøtte den evt. diskussion, samtidigt med at sørge for interviewet ikke går i stå da der er planlagte spørgsmål (Hvass-Raun & Dalsaa, 2023).



Figur 63 - Spilbrættet under første test

Testen blev udført med tre spiltestere. Af praktiske grunde var spiltesterne klassekammerater og venner af projektgruppen, hvilket godt kan have en effekt på deres opførsel og svar. Under selve testen blev der observeret at en af spiltesterne, som havde lange arme, ikke bevægede sig rundt om bordet lige så meget som de andre, da de nemmere kunne nå felterne uden behov for at bevæge sig. De to andre spiltestere bevægede sig dog rundt om bordet som de bevægede deres spilbrik for at kunne nå felterne. Der var venlig og sjov tale imellem spiltesterne under selve testen, og taleemnet forblev på selve spillet, og ikke om andre ting.

Som tidligere nævnt var interviewet et semistruktureret gruppeinterview. Det betød at når der blev stillet et spørgsmål til spiltesterne var der som regel et svar på spørgsmålet fra en enkelt spiltester og derefter andet feedback som ud fra samtalen. I tabel 2 kan alle spørgsmål og feedback findes. Hvis noget feedback har været et direkte svar på et spørgsmål, vil spørgsmålet stå udfør feedbacket, ellers vil den feedbacket stå alene.

Spørgsmål	Feedback
Hvad tænker i om det her med at man bliver nødt til at stå op fordi man skal gå rundt? Giver det nogle udfordringer ved at spille det?	"Jeg vil sige det gør det sjovere, det bliver mere interaktivt"
Hvad tænker i om det her med at man bliver nødt til at stå op fordi man skal gå rundt? Giver det nogle udfordringer ved at spille det?	"Jeg tror I skal passe på med at der ikke går for lang tid mellem hver tur, hvis der går for lang tid uden det er ens tur bliver det nok et problem"

Hvad synes i om størrelsen af felterne?	"Magneter på tiles og spillere så de passede sammen, eller en anden måde at få dem til at passe med hinanden"
Hvordan er mængden af veje og afveje på stierne?	"Der kan nok være flere splits, men det kommer også an på mængden af events"
Hvordan er størrelsen på brættet?	"Jeg kan godt lide størrelsen fordi det føles forfriskende når man er vant til brætspil som altid gør ting så småt som muligt"
Har i nogle andre tanker?	"Jeg kan godt lide man kan bestemme hvor man går hen"
	"Det er meget mere interessant fordi der faktisk er dybde i"

Tabel 2 - Spørgsmål og feedback fra brugertest omkring bevægelse omkring brættet

Anden test af skattespillet

Da skattespillet var blevet mere udformet med fastlagte regler og spilbræt blev endnu en test foretaget. Testen var igen en observationstest. Formålet med denne test var at få data på balanceringen af spillets regler og mekanikker. Under testen blev der observeret at spillerne undgik at interagere med hinanden med stjæl mekanikken da det var mere sikkert og nemmere selv at samle skatte op som lå rundt omkring på bordet. Der blev også observeret at snakken mellem spiltesterne ikke omhandlede spillet, men derimod andre urelatede emner i modsætning til den første spiltest. Problemet opstod også flere gange at spillerne mistede fokus på spillet og ikke vidste hvis tur det var.

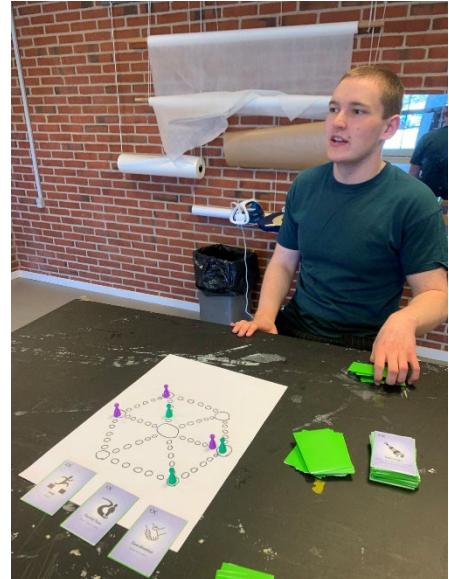
Quick and dirty test

Efter at have skiftet til produktet *Territory Takeover* og designet reglerne og kortene, udførte vi en kort brugertest ud fra metoden quick and dirty (Moe, et al., 2012). En quick and dirty test har til formål at give et hurtigt indtryk af, hvorvidt et produkt er på det rette spor - her blev metoden brugt, for at finde ud af om reglerne for spillet fungerede og for at se om der var nogle store designproblemer. Testen blev opstillet som en 1-mod-1-dyst mellem et af gruppens medlemmer og en frivillig klassekammerat (Figur 64). Testeren blev introduceret til spillets regler før spillet blev sat i gang. Et par fejlkilder i denne test var, at spillet og brættet primært er designet til 3-4 mennesker, og at en bekendt ikke nødvendigvis giver lige så objektiv feedback af hensyn til høflighed. Dog er dette acceptable fejlkilder, når målet blot er at få et hurtigt indtryk af hvorvidt spillet fungerer.

Resultaterne af testen var, at gruppen kunne observere at spillets regler generelt fungerede som planlagt. Derfor blev der kun lavet små ændringer som resultat af denne test; som beskrevet tidligere blev spillerens håndstørrelse ændret fra syv til seks kort.

Test af Territory Takeover

Det har desværre ikke være muligt grundet tidsbegrænsninger at teste det samlede produkt, som er spillet *Territory Takeover* med den detaljerede spilleplade. En sådan test ville kunne give den mest præcise data på hvor indlevet spillere kan blive ved brug af en interaktiv spilleplade som også er udsmykket i tre dimensioner. Data fra sådan en test sammenlignet med en test af spillet uden den



Figur 64 - Quick and dirty test

detaljerede spillerplade som den eneste forskel ville kunne give klare data på indflydelsen af at indføre en spillerplade der kan interageres med samt reagerer på spillernes handlinger. Evt. kunne der også testes med spillepladen, men hvor forskellige parametre såsom lyd, lys og udsmykning fjernes ellers hvor intensiteten af det ændres. Ved at justere på parametrene ville der evt. kunne fremstå data der indikerer hvor vigtigt sammenspillet mellem dem er og om der er nogle parametre der gør mere end andre med henblik på indlevelse.

Overordnet set er en test af det fulde produkt altid en god idé, da det er med til at sikre kvaliteten og kan hjælpe med at finde fejl og mangler som kunne være blevet overset. Optimalt ville en test af det fulde produkt være en observationstest som gruppen ville kunne overse, uden at testspillerne kan se gruppemedlemmerne. Da produktets formål er at øge indlevelse, ville det også være bedst for testen hvis der ikke var forstyrrende elementer til stede under testen som ikke ville finde sted under et normalt brugsscenario af produktet. Testspillernes indlevelse ville sandsynligvis blive påvirket af observatører som ikke selv er med i spillet. Derfor ville en test af det fulde produkt derfor forhåbentligt kunne være fundet sted i et lukket lokale kun med testspillerne samt skjulte, eller i hvert fald ikke iøjnefaldende, kameraer så projektgruppen stadig kan observere.

Evaluering

Ud fra gruppens egen erfaring med at spille spillet, har brættets evne til at øge indlevelsen ved kombinationen af de fysiske og digitale interaktionselementer været en succes. Dog er der i denne vurdering en vis bias, eftersom det er i gruppens egen interesse at spillet opfylder problemformuleringens krav. Der blev aldrig udført en reel brugertest af det færdige produkt med brugere, som ikke er bekendte med spillet. Det har der været flere årsager til, herunder manglen på struktur indenfor gruppens projektstyring. Derudover har idegenereringsprocessen været langtrukken, hvilket har resulteret i en relativ kort produktionstid. På baggrund af dette, samt en længere fejlfindingsproces af komponenter, har det ikke været muligt at udføre en dedikeret brugertest af det færdige produkt.

Dog kan der tages udgangspunkt i en tidligere "Quick and dirty" brugertest, som blev udført på en tidlig skitsering af det endelige spil, der antydede at spillets grundlæggende struktur og regelsæt var fungerende. Derudover indeholder produktet alle de elementer nævnt i problemformuleringen, herunder RFID som digitalt interaktionselement, kort som fysisk interaktionselement, stemningsmusik i baggrunden, et udformet landskab med håndskabet miniaturemodeller og LED-lys til feedback fra spilbrættet.

Diskussion

Brugen af SCRUM-metoden blev som tidligere nævnt tilsidesat imod slutningen af forløbet i sammenhæng med at strukturen af arbejdssagen også blev mere flydende. I fremtidige projekter hvis en lignende situation skulle opstå ville det give mening at skemalægge specifikke tidspunkter hvor SCRUM-møder vil finde sted. Dette ville sikre at selv når der evt. ville være arbejdssage uden et naturligt start- og sluttidspunkt ville SCRUM strukturen stadig kunne være gældende og bidrage til projektstyringen. Den bedste måde at gøre det på ville sikkert være at allerede fra starten af forløbet indsætte SCRUM-møder på faste tidspunkter af arbejdssagen, fremfor i de mere løse tidspunkt af start- og sluttidspunktet på arbejdssagen. Start- og sluttidspunktet af en arbejdssdag kan være meget varierende, især når den forløber i sammenhæng med en skoledag der evt. kan indeholde moduler med andre fag. I større projekter, har gruppen især erfaring med at der kan forekomme lange arbejdssage mod slutningen af projektet, dette skete også til dette projekt, hvilket også har en indflydelse på sluttidspunktet af arbejdssagen. Opsummeret kan det derfor være en fordel at holde SCRUM-møder på faste tidspunkter som ikke nødvendigvis falder ved start- og sluttidspunktet af arbejdssagen for at sørge for at den struktur SCRUM kan give forbliver gældende over hele projektforløbet.

Nogle af gruppens medlemmer havde nogle personlige ønsker til hvilke dele af projektet de gerne ville arbejde med. Ønskerne var bl.a. at arbejde med kode, arbejde med lyd eller arbejde på én stor del igennem projektet. Det var dog de færreste ønsker som gik i opfyldelse da der af praktiske årsager var brug for fleksibilitet i arbejdssprocessen. Af dette kan det læres at ønsker til arbejdsopgaver er fine at have, men i realiteten skal gruppen og dens medlemmer være omstillingsparat og klar til at påtage sig nogle opgaver og fralægge sig andre, selvom det ikke stemmer overens med ønskerne.

Konklusion

Der er blevet udviklet et brætspil der opfylder problemformuleringens krav mht. fysiske og digitale interaktionselementer. Dette blev gjort på trods af komplikationer med tid, der bl.a. var forårsaget af skift af idé og lang idégenerering. Dog har det ikke været muligt at teste hvorvidt selve produktet øger indlevelse hos spilleren grundet de førnævnte tidsproblemer. Hvis der var mere tid til at arbejde på projektet, burde der foretages en dybdegående brugertest af det endelige produkt for at afgøre om det lever op til problemformulerings krav om at øge indlevelse. Gruppen har lært fra dette projekt at vigtigheden bag at være konsekvent i sin projektstyring og evt. lave faste tidspunkter at holde SCRUM-møder, fremfor at gøre det på mere arbitrære tidspunkter. Alt i alt er gruppen tilfreds med produktet, især hvis der tages højde for den tidsbegrænsning gruppen desværre satte for sig selv ved at ændre idéen af spillet.

Bibliografi

- Arduino. (2023, April 21). *SoftwareSerial Library*. Retrieved from Arduino Docs: <https://docs.arduino.cc/learn/built-in-libraries/software-serial>
- Autodesk Fusion 360. (2023, 04 23). *Foren design, teknik, elektronik og produktion med Fusion 360*. Retrieved from autodesk.dk: <https://www.autodesk.dk/products/fusion-360/overview>
- Berger, T. (2012, 09 28). *Historien om Ludo*. Retrieved from papskubber.dk: <https://www.papskubber.dk/artikel/historien-om-ludo>
- Calleja, G. (2022). *Unboxed: Board Game Experience and Design*.
- Campbell, S. (2023, 04 25). *BASICS OF THE I2C COMMUNICATION PROTOCOL*. Retrieved from circuitbasics.com: <https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/>
- Canva. (n.d.). *Home - Canva*. Retrieved from Canva: <https://www.canva.com/>
- Components101. (2021, 06 15). *I2C Serial Interface Adapter Module for LCD*. Retrieved from components101.com: <https://components101.com/modules/i2c-serial-interface-adapter-module>
- Compton, C. (2018, juni 25). *Luck Vs Skill: The False Dichotomy*. Retrieved from Game Developer: <https://www.gamedeveloper.com/design/luck-vs-skill-the-false-dichotomy>
- Creality. (2023, 04 25). *Creality*. Retrieved from creality.com: <https://www.creality.com/>
- DFRobot. (2023, 04 25). *DFPlayerMini*. Retrieved from wiki.dfrobot.com: https://wiki.dfrobot.com/DFPlayer_Mini_SKU_DFR0299
- Draftsim. (2019, december 20). *MTG Card Size/Dimensions, Weight, and Much Much More*. Retrieved from Draftsim: <https://draftsim.com/mtg-card-size/#:~:text=For%20now%20we'll%20look,%2C%20or%200.305mm%2C%20thick.>
- Erikstrup, M. (2023, april 21). *DDU Eksamensprojekt Soundtrack*. Retrieved from SoundCloud: https://soundcloud.com/prodmokkel/ddu-eksamensprojekt-soundtrack/PSy7DHc4u9V?si=a3f72ce84f7d408b8d035ee854981895&utm_source=clipboard&utm_medium=text&utm_campaign=social_sharing
- Game-Icons. (n.d.). *Already 4131 free icons for your games / Game-icons.net*. Retrieved from Game-icons.net: <https://game-icons.net/>
- Hamann, M. (2020, januar). *Dur-akkord på I-trin*. Retrieved from Popmusik & modalharmonik: <https://popmusikogmodalharmonik.systime.dk/?id=154>
- Hamann, M. (2020, januar). *Mol-akkord på I-trin*. Retrieved from Popmusik & modalharmonik: <https://popmusikogmodalharmonik.systime.dk/?id=155>
- Holtug, N. (2023, april 21). *empirisme*. Retrieved from Den Store Danske: <https://denstoredanske.lex.dk/empirisme>
- Hvass-Raun, R., & Dalsaa, A. (2023). Undersøgelsesmetoder. In R. Hvass-Raun, & A. Dalsaa, *Digitalt Design og Udvikling* (pp. 91-98). Systime A/S.
- Image Line. (2023). *FL Studio [Official] / Overview*. Retrieved from Image Line: <https://www.image-line.com/fl-studio/>
- Jeppesen, M. M., Henriksen, L. B., & Routhe, H. W. (2019). *3.1.1 Brainstorm*. Retrieved from Projektarbejdet | Systime: <https://projektarbejdet.systime.dk/?id=170>

- Lean Enterprise Institute. (2023, april 21). *What is Lean?* Retrieved from Lean Enterprise Institute: <https://www.lean.org/explore-lean/what-is-lean/>
- LEDWATCHER. (2020, 08 4). *How to diffuse LED lighting?* Retrieved from ledwatcher.com: <https://www.ledwatcher.com/how-to-diffuse-led-lights/>
- Moe, S., Fenger-Grøn, A., Bern, B. F., Wiener, C. V., Lungholt, H., Nowak, S. B., & Pii, K. (2012). *Quick and dirty.* Retrieved from Kommunikation og IT A | Systime: <https://kommita.systime.dk/index.php?id=132>
- Mullich, D. (2017, april 10). *Implementing Risk/Reward Decisions In Games.* Retrieved from David Mullich - Electrifying Education and Entertainment Since 6502: <https://davidmullich.com/2017/04/10/implementing-riskreward-decisions-in-games/>
- Parallax. (2023, April 24). *RFID Card Reader – Serial.* Retrieved from Parallax: <https://www.parallax.com/product/rfid-card-reader-serial/>
- Plastindustrien. (2023, 04 25). *Polystyren (PS-plast).* Retrieved from plast.dk: <https://plast.dk/det-store-plastleksikon/polystyren-ps/>
- Schwaber, K., & Sutherland, J. (2023, april 24). *The 2020 Scrum Guide.* Retrieved from scrumguides.org: <https://scrumguides.org/scrum-guide.html>
- scrum.org. (2023, april 24). *What is Scrum?* Retrieved from scrum.org: <https://www.scrum.org/learning-series/what-is-scrum>
- SEN News. (2018, Oktober 16). *Proximity Access Readers: 125kHz or 13.56Mhz?* Retrieved from Sen.News: <https://sen.news/proximity-access-readers-125khz-or-13-56mhz/>
- Splice. (2023). *Splice.* Retrieved from Splice: <https://splice.com/>
- The Thoughtful Gamer. (2017, marts 28). *Catch-Up Mechanisms - How Games Combat The Runaway Leader Problem.* Retrieved from The Thoughtful Gamer: <https://thethoughtfulgamer.com/2017/03/28/catch-up-mechanisms/>
- Twilight Imperium: Third Edition.* (n.d.). Retrieved from Board Game Geek: <https://boardgamegeek.com/boardgame/12493/twilight-imperium-third-edition>
- Ultimaker. (2023, 04 25). *Ultimaker.* Retrieved from ultimaker.com: <https://ultimaker.com/>
- Wikimedia Foundation. (2023, april 5). *Hearthstone.* Retrieved from Wikipedia: <https://en.wikipedia.org/wiki/Hearthstone>
- Wikimedia Foundation. (2023, april 16). *Ludo.* Retrieved from Wikipedia: <https://en.wikipedia.org/wiki/Ludo>
- Wikimedia Foundation. (2023, april 9). *Magic: The Gathering.* Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Magic:_The_Gathering
- Wikimedia Foundation. (2023, april 10). *Mario Party.* Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Mario_Party

Bilag

Projektets kode er vedhæftet den digitale aflevering som et zip-arkiv, af hensyn til længden af rapporten.