Bacheloroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

Aria Thuy Dung Bui
Jingyi Li
Julie Tverfjell

Utvikling av samarbeidsspill for spillplattformen Tiltspot

Bacheloroppgave i Dataingeniør
Veileder: Majid Rouhani

Mai 2020

NTNU
Kunnskap for en bedre verden

Aria Thuy Dung Bui
Jingyi Li
Julie Tverfjell

Utvikling av samarbeidsspill for spillplattformen Tiltspot

Bacheloroppgave i Dataingeniør
Veileder: Majid Rouhani
Mai 2020

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

NTNU
Kunnskap for en bedre verden

NTNU

Bacheloroppgave 2020

Studium: Dataingeniør

Institutt for datateknologi og informatikk

Tittel - norsk: *Utvikling av lokalt samarbeidsspill for spillplattformen Tiltspot* ***Tittel - engelsk:*** *Development of a couch co-op game for Tiltspot gaming platform*	**Oppgave nr.**: 34
Oppgaverstiller: Phonion Technologies	***Kontaktperson:*** Mathias Udahl
	Telefon: 94867686
	E-postadresse: mathias.udahl@phonion.com
	Postadresse: Munkegata 58, 7011
Studenter: Aria Thuy Dung Bui, Jingyi Li & Julie Tverfjell	
Veileder ved NTNU: Majid Rouhani	

Sammendrag:

Formålet med denne oppgaven er å dokumentere utviklingen av et teambasert 3D spill for Tiltspot. Produktet *Chef Poteto* omhandler utføringen av matlagingsoppgaver i idylliske omgivelser. Spillet er blitt utviklet i spillmotoren Unity, og tilhørende kontroller i HTML, CSS og Javascript. Arbeidet med produktet gir innblikk i hvordan spillmekanikk utnyttes til oppfordring av samarbeid. På grunnlag av en brukerorientert utviklingsprosess, har utviklerteamet utarbeidet et produkt som ivaretar brukerens opplevelse, og som tar hensyns til brukervennligheten.

Abstract in English:

The purpose of this project is to document the development of a team-based 3D game for Tiltspot. Chef Poteto revolves around the execution of various cooking assignments in idyllic surroundings. The game is developed with the game engine Unity, and the related controller is developed in HTML, CSS and JavaScript. The work with the product gives an insight into how game mechanics can be used to encourage collaboration. During the development, results provides insight into how game mechanics are utilized to encourage collaboration. Using a user-oriented development process, the development team has developed a product which has the user experience and ease of use at the center.

Forord

Denne bacheloroppgaven er utarbeidet i samarbeid med Phonion Technologies, og er den avsluttende delen av vår bachelorgrad i ingeniørfag, data ved NTNU. Vi valgte følgende oppgave på grunnlag av flere elementer. De viktigste grunnene for valget av dette prosjektet er som følger:

1. En grunnleggende interesse for spill og samarbeid med relevant bedrift
2. Prosjektets omfang. Produktet skal tilgjengeliggjøres på nettsiden tilhørende bedriften, og skal dekke et behov som ikke allerede er dekket.
3. Bedriftens verdier. Tiltspot er en nettbasert spillplattform som muliggjør bruken av smarttelefoner som kontrollere og datamaskin som konsoll. Dette er et gratis konsept, da de fleste er i besittelse av ovennevnt utstyr.

Det skal også nevnes at vi fikk et godt inntrykk av bedriften etter å ha avholdt et oppstartmøte med dem. Å arbeide med denne oppgaven har vært både stimulerende og utfordrende. Stimulerende ettersom vi fikk muligheten til å jobbe med noe vi interesserer oss for, og at vi fikk bidra til et konsept vi har sterk tro på. Utfordrende fordi dette er et nytt og ukjent terreng for samtlige, og det krevde en del arbeidsinnsats for å sette seg inn i de ulike aspektene ved denne typen spillutvikling. Totalt sett har dette vært en lærerik prosess, hvor vi har tilegnet oss betydelig mer kunnskap innen spillutvikling og utførelse av smidige arbeidsmetodikker.

Vi ønsker å takke følgende personer for støtte og hjelp til prosjektoppgaven:

- Majid Rouhani fra Institutt for datateknologi og informatikk, NTNU, for hans rolle som prosjektveileder.
- Mathias Udahl og Sindre Vingen fra Phonion Technologies, for teknisk og faglig veiledning under hele prosjektet.
- Famille og venner som tok seg tid til å komme med plausible innspill, råd og tilbakemeldinger i forbindelse med utførelse av bruktertester.

Dokumentet er utarbeidet etter mal fra Institutt for datateknologi og informatikk, NTNU

Trondheim, 19.05.2020

X _____ X _____ X _____

Aria Thuy Dung Bui | Jingyi Li | Julie Tverfjell

Oppgavetekst

Tittel: Spillutvikling med Tiltspot

Hensikt med oppgaven: Hensikten med oppgaven er at å tilby flere spill og tiltrekke flere brukere til plattformen. Tiltspot har behov for flere utviklere og spill på plattformen, og ønsker å benytte spillene for markedsføring av plattform.

Beskrivelse av oppgaven: Oppgaven går ut på å utvikle et spill (HTML/JS/CSS og/eller Unity) som vil bli publisert på Tiltspot. Studentene vil bli utviklere og utgivere bak et couch co-op (couch co-op, Vedlegg A, 2.13) spill med tilhørende kontroller som vil bli publisert på Tiltspot. De kan selv stå for konsept og ide, men med gitte rammer rundt lokal flerspiller. Spillet må benytte Tiltspot sitt API og studentene kan velge mellom å utvikle spillet i HTML/JS/CSS og/eller Unity (C#).

Et spill som lanseres på Tiltspot bør være repetitivt og passe flere enn fire spillere.

Forventninger til oppgaven

Studentene kan forvente en spennende og lærerik oppgave med stor grad av frihet. Oppgaven vil utfordre studentene til å lære nye konsepter rundt spillutvikling, design og brukertesting gjennom iterative prosesser. Spillet vil eies av studentene og eventuelle inntekter (om studentene ønsker å selge spillet eller innhold i spillet) vil følge en plattformsplitt på 70 % til studentene og 30 % til Tiltspot. Dette vil i så fall beskrives i egen kontrakt.

Sammendrag

Bedriften Phonion Technologies ble startet i 2018 på NTNU. De ønsket å undersøke kommersialiseringen av et konsept, utarbeidet av Sindre Vingen. Målet med konseptet var å utforme en lettvint måte å spille med familie og venner i sosiale omgivelser. Bedriften har tro på opplevelsen tilknyttet lokal flerspilling, og hvordan Tiltspot kan løse problemer som omhandler mangelen på utstyr og begrensninger i antall spillere.
Produktet er blitt utviklet i spillmotoren Unity, og tilhørende kontroller i HTML, CSS og JavaScript. Utviklingen av produktet bygger på bruken av prinsipper tilknyttet spilldesign, interaksjonsdesign og teknikker for utforming av arbeidsoppgaver.

Arbeidet med produktet har gitt økt kompetanse angående bruken av forskjellige teknikker og prinsipper for spilldesign. Det har blitt gitt innblikk i hvordan spillmekanikk kan utnyttes til å oppfordre brukerne til å samarbeide om å løse oppgaver mot et felles mål. På grunnlag av en brukerorientert utviklingsprosess, har utviklerteamet utarbeidet et produkt som ivaretar brukerens opplevelse og som tar hensyn til brukervennligheten.
Majoriteten av tilgjengelige spill på bedriftens spillplattform er sentrert rundt utformingen av et enkelt og brukervennlig brukergrensesnitt. Resultatet av det utarbeidede produktet Chef Poteto, utgjør et sterkt fundament for fremtidige spill tilhørende samme kategori på spillplattformen. Midlertidig burde det tas hensyn til begrensninger som følge av nettleservalg. Det er mange utfordringer knyttet til utformingen av et spill som skal kjøres på en nettside. Slike utfordringer omhandler kjøretid og belastning.

Denne rapporten tar for seg prosessen rundt utviklingen av spillet. Det gis innblikk i hvilke teknologier og teknikker som er blitt benyttet, og begrunnelse for disse valgene.
Resultatene av prosessen og produktet blir presentert, og diskusjon rundt utfordringer, resultater og problemer blir undersøkt og konfrontert.

Innholdsfortegnelse

1 Introduksjon

1.1 Akronymer og ordforklaringer

Se vedlegg A, Akronymer og ordforklaringer. Det anbefales å lese gjennom vedlegget før resten av rapporten.

1.2 Bakgrunn for prosjektet

Dataspill ble først populært på 70- og 80-tallet og er i dag en av verdens største underholdningsbransjer, sett fra et økonomisk perspektiv [1]. Sammen med bransjens vekst i omsetning har markedet også hatt en økende variasjon blant spillene som tilbys. Couch Co-Op er en type spill hvor flere personer som befinner seg i samme omgivelse kan spille samtidig, og er et tiltrekkende valg i disse dager. Følgende spill-interaksjon krever at spillerne befinner seg på samme lokale nettverk [2]. En av de grunnleggende elementene tilhørende denne typen spill, er fysisk sosial interaksjon mellom brukerne. Lagbaserte spill er en variant innenfor kategorien flerspiller, hvor spillerne deles inn i lag, forså å samarbeide innad teamet for å utkonkurrere de andre lagene. Denne typen spill inkorporerer elementer fra både samarbeidsspill og konkurransebaserte spill.

Oppdragsgiveren for dette prosjektet er den norske oppstartsbedriften Phonion Technologies. Phonion Technologies er et norsk firma som spesialiserer seg på kommunikasjonsteknologi mellom datamaskin og smarttelefon [3]. Tiltspot er en web-basert spillplattform, utviklet av Phonion Technologies, hvor spillerne spiller sammen på en felles skjerm. Det kreves ingen nedlastninger for å spille. Bakgrunnen og årsaken til prosjektet kan knyttes til mangelen av gratis spill. Tiltspot er et billigere og mer tilgjengelig alternativ til dagens spillkonsoller, ettersom de fleste i dag er besittelse av en datamaskin og smarttelefon.

Skaperne bak Tiltspot ønsker å skape en underholdende og sosial spillopplevelse for alle [3]. Per dags dato finnes det seks forskjellige spill på Tiltspot, hvor alle er konkurransebaserte spill. Tiltspot ønsker å utvide sin brukerskare ved å tilby et større utvalg og mangfold av spill på plattformen. I den anledning har Phonion Technologies utformet denne bacheloroppgaven som omfatter utviklingen av et spill, med gitte rammer rundt lokal flerspiller. Vi har valgt å opprette et teambasert spill, som de foreløpig ikke har i sortimentet. Og det er på følgende måte vi vil bidra til å øke mangfoldet på plattformen.

1.3 Problemstilling

Med tanke på frihetsgraden tilknyttet utdelt oppgave, er det utfordrende å utvikle et spill som lever opp til høyt standard-nivå. Etter å ha undersøkt tilgjengelige spill på plattformen Tiltspot, har vi opprettet et mål for å sikre konsept tilpasset plattform, og som samtidig sikrer en god brukeropplevelse. Dette medfører at spillerne og Tiltspot utgjør en grunnleggende rolle gjennom hele utviklingsprosessen, da spillet skal preges av brukernes tilbakemeldinger og ideer.

Det er en mangel av team-baserte spill i Tiltspot, dermed er det et ønske om at konseptet av spillet skal ta utgangspunkt i samarbeid. I lys av dette, er vår problemstilling følgende:

Er det mulig å benytte spilldesign til å skape et brukervennlig spill som oppfordrer til samarbeid mellom spillerne i Tiltspot?

For å verifisere resultatet av problemstillingen, vil det bli utført brukertester, og det vil tas hensyn til tilbakemeldinger og forslag om forbedring. Ettersom spillet er unntatt fra forskriften om universell utforming, tas det likevel stilling til grunnprinsippene om universell utforming. På denne måten integreres arbeidsmetoden vår med interaksjonsdesign for å oppnå brukervennlighet. For å vurdere om et spill er engasjerende og brukervennlig, har vi valgt å definere et utvalg forskningsspørsmål tilknyttet problemstillingen, med hensyn til brukervennlighet og samarbeid:

1. *Kan spillet lages responsivt, både kontrollere og hovedskjerm, slik at brukergrensesnittet tilpasses flere skjermstørrelser?*
2. *Hvordan kan brukerinvolvering bidra til å oppnå et mer brukervennlig spill?*
3. *Hvilke teknikker og tiltak kan bidra til å fremme samarbeid mellom spillerne?*

1.4 Struktur

Videre i denne rapporten starter vi med en beskrivelse av teorien som ligger til grunn for dette utviklingsprosjektet. Deretter følger et kapittel om hvilke metoder og teknologier som er tatt i bruk samt begrunnelse for valgene som er tatt. I de neste to kapitlene presenteres først de oppnådde resultatene og deretter drøfting av disse. Til slutt følger en konklusjon og forslag til videre arbeid med prosjektet. Referanser til vedlagte dokumenter oppgis underveis i oppgaven.

2 Teori

2.1Spilldesign

Spilldesign handler å oppfinne et spillkonsept, dets sentrale spillmekanismer og dets regler gjennom sentrale ideer, og viser til bruk av kreativitet og design for å utvikle et spill for underholdning eller pedagogiske formål. Det innebære å skape karakterer, regler, mål og utfordringer som fører til handlinger mellom andre brukere, tegn, eller objekter [1]. Innenfor spilldesign kan et godt kunstdesign visualisere konseptet og ideer, slik at spillerne oppnår en bedre spillopplevelse. Grafisk design av spillet skal justeres i henhold til forskjellige behov og formål, for eksempel fargevalg og størrelse på objekter.

I spilldesign er det vanskelig å definere regler og retningslinjer som gjelder for alle spill. Foreløpig vektlegges tre grunnleggende elementer for definering av et komplett spill, nemlig spillerne, kommunikasjon og appellen [2]. Et godt spill burde tydeliggjøre hvilken rolle hver spiller har, og hvordan de kan samhandle med spillet, slik at spillet foregår som forventet. Dernest er det et mål som spillerne kan arbeide mot. Følgelig blir spillet mer underholdende og utfordrende. Kommunikasjon er dermed hjelpemiddel som benyttes for å fortelle hva målet er, og hvordan de skal nå målet.

Det mest essensielle og krevende poenget er å få spillet til å appellere til spilleren. Derfor er det et grunnleggende behov for å innføre tester av spillet ved bruk av enkle prototyper fra starten av. Dette oppnås best mulig ved å ta hensyn til målgruppens opplevelser og interesser ved utførelse av testing og undersøkelser. Etter etablering av grunnlaget for spillet, utføres det et valg av design som sørger for at spillet følger prosjektets krav, på en best mulig måte. Alle beslutninger som tas ved spilldesign-fasen kan i ulik grad påvirke brukerens opplevelser, følelser og engasjementnivå på en betydelig måte.

2.2 Spillmotor

En spillmotor er en programvare som gir utviklere et utviklingsmiljø med et sett nødvendige verktøy, som gjør utvikling av spill raskt og effektivt. Motoren er altså et rammeverk for spillutvikling som støtter og bringer forskjellige kjernearenaer sammen, nærmere bestemt;

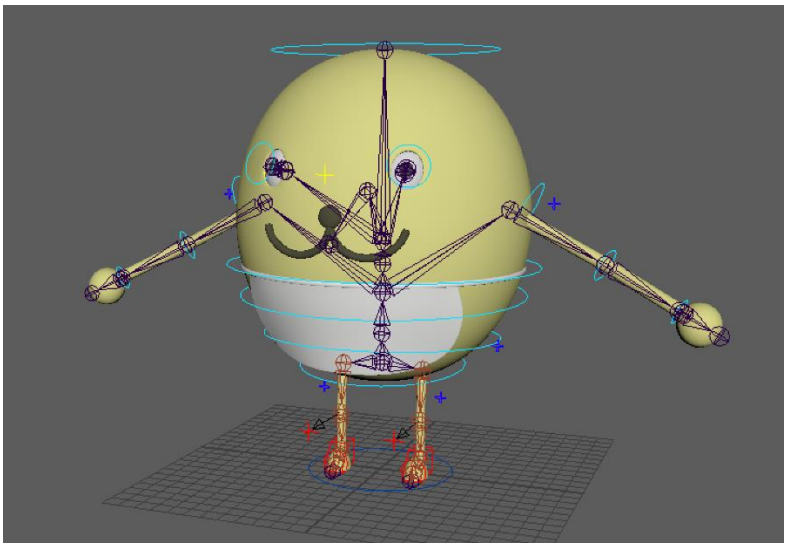
håndtering av kompliserte tema som parallellisering, fysikk, minnehåndtering og kompilering til forskjellige plattformer. Ved å benytte en spillmotor står man fritt til å importere kunst (3D eller 2D) og *assets* (Assets, Vedlegg A, 2.6) fra andre programvarer.

2.2.1 Particle System

Particle System er en komponent i spillmotoren som kan brukes til å simulere ulike fluider, som for eksempel væsker og gasser. Ved å generere mange små 2D bilder, også kalt partikler, i en scene kan vi skape en slik visuell effekt. Hver eneste partikkel i systemet representerer ett grafisk element i effekten. Disse elementene har mange ulike egenskaper som farge, form og størrelse som kan manipuleres. Systemet simulerer alle disse partiklene samtidig som sammen skaper den ønskede effekten [15].

2.2.2 Animasjonssystem

Animasjonssystemet inneholder informasjon om hvordan visse objekter skal endre sin posisjon, rotasjon eller andre egenskaper over tid. Dette gir utviklere mulighet til å styre handlinger og bevegelser tilhørende karakteren ved bruk av et "skjelett". Skjelettet opprettes i modellering-fasen ved hjelp av tredjepartsverktøyet Maya. Handlingen til karakteren som ses, er som følge av bevegelser av forskjellige bein i kroppen.



Figur 2.2- 1 Figur av rigging til karakteren Poteto

2.3 Brukervennlighet

Brukervennlighet omhandler brukergrensesnitt som er enkelt å bruke. Det innebærer at produktet er «vennlig» for brukeren, at det ikke er vanskelig å lære eller å forstå. Selv om brukervennlighet er et subjektivt begrep, benyttes det diverse felles-attributter som gjenspeiler brukervennlige grensesnitt.

Følgende attributter har blitt funnet i brukervennlige grensesnitt:

- **Enkel:** Et brukergrensesnitt er enkel og tilbyr rask tilgang til vanlige funksjoner eller kommandoer.
- **Ren:** Et godt organisert brukergrensesnitt gjør det enkelt å finne forskjellige verktøy og funksjoner.
- **Intuitiv:** Minimalt med forklaring er nødvendig for å forklare bruken av brukergrensesnittet (Ikke mer).
- **Pålitelig:** Et upålitelig produkt som forårsaker unødvendig frustrasjon for brukeren skal unngås. Et brukervennlig produkt skal være pålitelig.

Målet med det brukervennlige produktet er å bidra til en god brukeropplevelse. Resultatet avhenger av sluttbrukeren som produktet er designet for. For eksempel vil et brukervennlig barnespill ha et annerledes grensesnitt enn et profesjonelt redigeringsverktøy. Reglene ovenfor gjelder uansett for begge typer programvarer. Selv om et program inneholder avanserte funksjoner, er det fremdeles mulig å oppnå et brukervennlig produkt ved å designe det enkelt, rent og intuitivt [22].

2.4 Interaksjonsdesign

Interaksjonsdesign omhandler blant annet design av samhandling mellom mennesker og digitale systemer. Det handler om å designe systemer som er effektive, enkle, morsomme og behagelige å benytte. I sammenheng med vårt prosjekt skal vi se på hvordan brukerinteraksjon kan designes på en best mulig måte, ved å benytte konkrete metoder, rammeverk og teknikker som tilbys av designvitenskapen.

Prosessen interaksjonsdesign består av fire trinn:

1. Definerer krav til det som skal opprettes

2. Designe alternativer
3. Oppretting av prototyper
4. Evaluering

Følgende trinn gjentas flere ganger til produktet kan anses som ferdig. Interaksjonsdesignprosessen er altså en iterativ prosess i likhet med Scrum. Når man benytter en slik prosess, er det fem prinsipper en skal legge vekt på, nemlig, synlighet, feedback, begrensninger, tilbydelser og konsistens.

For at systemet skal kunne anses som et godt produkt, må funksjoner ved systemet være synlige, både ved å vise at de er tilstede og å vise hvordan de skal benyttes. Systemet burde også signalisere til brukeren hva som foregår ved å benytte lyd eller fargeforandringer (gjerne begge deler) for å forsikre brukeren om at han/hun er på rett vei. For å hindre brukeren fra å gjøre feil, kan man legge begrensninger på hvilke former for interaksjon som er tilgjengelig på et gitt tidspunkt, som for eksempel å hindre brukeren fra å trykke på en knapp ved å ignorer brukerens trykk.

Det legges også vekt på at systemet burde preges av konsistens. Således burde det legges vekt på at brukeren utfører ting på lignende måter, da det bidrar med å spare brukeren for tid og tankeaktivitet. I tillegg tas det hensyn til hvilke handlinger en bruker opplever han/hun kan utføre med et system. Følgende opplevelser henger sammen med hvordan systemet er laget og tidligere erfaringer en bruker har gjort med andre systemer, og som vi skal se senere, henger dette sammen med utformingen av den konseptuelle modellen (konseptuell modell, Vedlegg A, 2.5) tilknyttet fase 1[12].

2.5 Samarbeid i spillutvikling

Samarbeid handler om å jobbe sammen med andre mennesker om å nå et felles mål eller å løse en oppgave [16]. Når flere personer kombinerer sine kunnskaper og ferdigheter har de muligheten til å utføre en oppgave mer effektivt og løse mer komplekse problemer enn én enkelt person. I et samarbeid spill må spillerne ofte jobbe sammen om å utføre ulike oppgaver eller slå en felles motstander. I tillegg er oppgavene som skal løses som oftest mer kompliserte, og spillerne" tvinges" på den måten til å samarbeide med hverandre [17].

Samarbeid i spill kan implementeres på mange måter, og teambaserte spill er en mulig variant.

Det finnes mange anbefalinger fra personer med lang erfaring med dette feltet. Alexander Jonassen har analysert en rekke anbefalinger fra aktører innen spilldesign og har basert på disse formulert 11 retningslinjer for design som fremmer samarbeid mellom spillere [18]. Vi har valgt ut noen av disse punktene som vi vil utype i dette kapitlet.

Det første punktet som trekkes frem omhandler oppfordring til kommunikasjon og samarbeid, ved å la spillerne ta egne valg og komplementere hverandres arbeidsinnsats. Det er ifølge Jonassen essensielt å legge vekt på felles prestasjoner fremfor individuelle. Han presiserer derfor at det å ha et poengsystem som vurderer gruppen som en helhet fremfor hver enkelt person vil ha en positiv effekt på samarbeidet. Det kan også være en fordel at medlemmene i gruppen deler de samme ressursene slik at alle har det samme utgangspunktet.

Et annet punkt omhandler design av ulike ferdighetsnivåer. Med dette menes utforming av arbeidsoppgaver som krever forskjellige ferdighetsnivå, både med tanke på motorikk og strategi, slik at alle i gruppen har mulighet til å mestre en oppgave. Slik kan alle, tross ferdighetsnivå, ha glede av spillet, som igjen vil bidra til økt samarbeid i gruppen.

Det å nedtone konkurransen innad i gruppen er også et av retningslinjene på Jonassens liste. Dersom gruppen har et felles mål fremfor flere individuelle mål vil dette bidra til økt samarbeid. Det oppfordres derfor til å designe et spill som bedømmer medlemmene an en gruppe som helhet.

Til slutt trekkes det frem at spillerne burde få mulighet til å utrykke seg selv. Dette mener Jonassen kan oppfylles ved å la hver spiller ha sin egen stil i form av en karakter eller benytte andre personlige tilpassede trekk. Spillerne burde også ha mulighet til å velge ulike variasjoner av spillet, hvor reglene eller objektivet er annerledes.

2.6 Designmønster

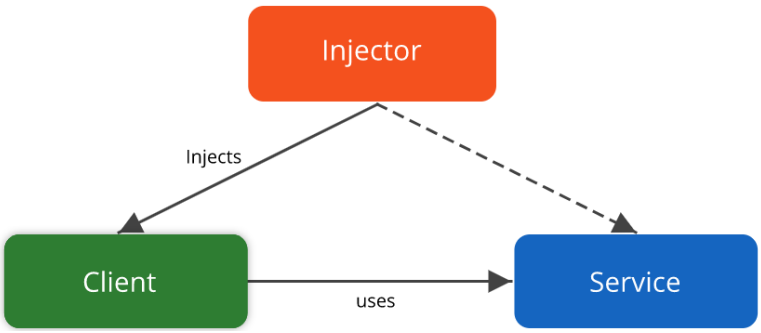
Et designmønster er en generell repeterbar løsning på et problem som forekommer ofte innen programvaredesign. Designmønsteret er en mal som implementeres for å skape god struktur og løst koblede komponenter.

2.6.1 Dependency Injection

Dependency injection er et programvare-designmønster, en teknikk som benyttes for å oppnå *Inversion of Control* (IoC). IoC er et designprinsipp som benyttes til å invertere forskjellig type kontroll i objektorientert design for å oppnå løs kobling. I dette avsnittet refereres kontroll som; eventuelle tilleggsoppgaver som en klasse har, som ikke er en del av dens hovedansvar. Dette inkludere kontroll over flyten av en applikasjon, og kontroll over flyten ved opprettelse av et objekt (uavhengig eller avhengig) og binding [7].
Dependency injection sikrer løs kobling ved å flytte opprettelsen av avhengighetsobjekter helt ut av klassen. For å lette forståelse på hva Dependency Injection er, tas det utgangspunkt i tre typer klasser.

1. *Klient klassen*: Klient-klassen er en klasse som er avhengig av Service-klassen.
2. *Service klassen*: Service-klassen (avhengigheten) er en klasse som bidrar med service til Klient-klassen.

Injector klassen: Injector-klassen injiserer Service-klasse objektet inn til Klient-klassen.



Figur 2.6-1 Dependency Injection [8]

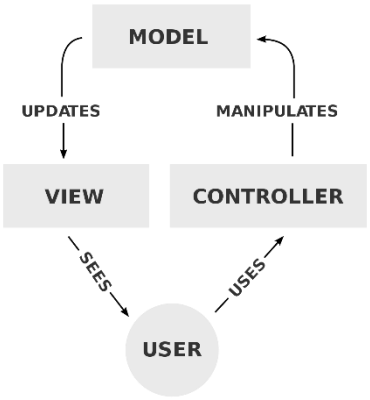
Injector-klassen oppretter altså et objekt av Service-klassen og injiserer følgende objekt til Klient-Objektet. Det er slik DI (dependency injection) har ansvaret for å opprette et Service-objekt utenfor klientklassen [9].

2.6.2 Model View Controller

Model View Controller er et designmønster som deler applikasjonen inn i tre hovedkategorier: *Model*, *View* og *Controller*. Følgende separering bidrar til å oppnå *seperations of concerns* (SoC, Vedlegg A, 1.2). En Model består av dataene som manipuleres, samt logikk og operasjoner som burde utføres av den. Et View er ansvarlig for å presentere innholdet gjennom brukergrensesnittet, og skal inneholde minimalt med logikk. Til slutt har vi Controlleren som har ansvar for håndtering av brukerens innspill og interaksjon ved å utnevne hvilken modelltype de skal jobbe med og hvilket View som skal gjengis. Med andre ord, kontrollerer en Controller hvordan applikasjonen skal respondere på en gitt forespørsel.

Ved å benytte dette designmønsteret blir brukerforespørsler dirigert til en kontroller som er ansvarlig for å samarbeide med modellen for å utføre brukerhandlinger eller innhenting av resultat knyttet til forespørsler. Kontrolleren avgjør visningen som skal vises til brukeren og den forsyner med modelldata dersom det trengs.

Følgende samarbeid kan illustreres ved bruk av diagrammet nedenfor.



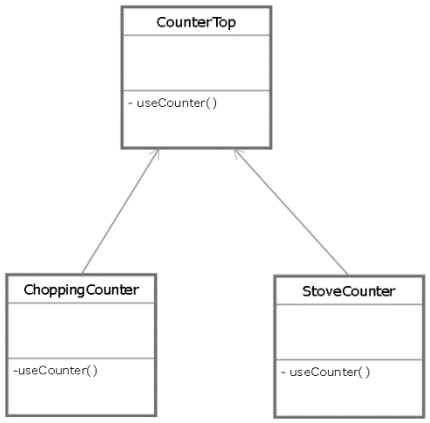
Figur 2.6-2 Interaksjonsdiagram for MVC [4]

En slik avgrensning av ansvarsområder bidrar til å skalere applikasjonens kompleksitet, ettersom det blir lettere å vedlikeholde, teste og kode noe som har et eget ansvarsområde. Formålet med følgende oppdeling er altså å skape løst koblede komponenter (lav kobling) og høy kohesjon, slik at senere bygging på systemet kan utføres uten problemer [5].

2.6.3 Polymorfisme

Polymorfisme er et designmønster som benyttes der objekter av forskjellige typer kan aksesseres av samme grensesnitt. Følgende oppstår når vi har mange klasser som er knyttet til hverandre via arv. Hver type kan oppgi sin egen uavhengige implementering av samme grensesnitt.

Arv lar oss arve datafelter og metoder fra en annen klasse. Polymorfisme utnytter disse metodene for å utføre forskjellige oppgaver, og tillater dermed utføringen av en enkel handling på forskjellige måter [6].



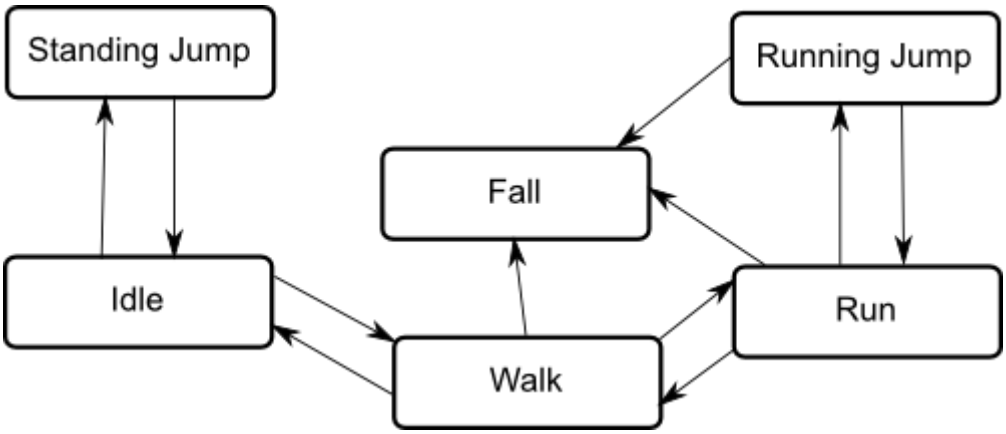
Figur 2.6-3Skjermutklipp av polymorfisme for CounterTop

Dersom vi har en baseklasse CounterTop som har en metode ved navn useCounter (), kan avledede klasser av CounterTop som ChoppingCounter og StoveCounter ha sin egen implementering av bruken av Counter-en(kjøkkenbenken). Ved å kalle på metoden UseCounter () hos en ChoppingCounter kan man for eksempel utføre kutting av grønnsaker, mens kall av samme metodegrensesnitt hos StoveCounter vil føre til steking av råvarer (følgende eksempel er hentet fra vår kildekode).

2.6.4 State Machine

State Machine er et designmønster, hvor den grunnleggende ideen er at en karakter er involvert i en bestemt type handling til enhver tid. Handlingene som er tilgjengelig vil være avhengig av spilltypen, men handlinger som inkluderer gåing, hopping og løping er involvert i det fleste spill. Slike handlinger refereres til tilstander, i den forstand at karakteren befinner seg i en tilstand der karakteren går, løper eller hopper.

Generelt vil det være begrensinger for hvilken tilstand en karakter kan befinne seg i, med andre ord kan ikke en karakter bytte umiddelbart fra en tilstand til en hvilken som helst tilstand. Følgende demonstreres i figuren nedenfor.



Figur 2.6-4 Figur av tilstandsoverganger tilknyttet en karakter [13]

Følgende figur demonstrerer at en karakter kan utføre ulike hopp ut ifra hvilken tilstand det er den befinner seg i. Dersom karakteren ønsker å utføre et hopp uten tilløp, må karakteren befinne seg i tilstanden *idle*, en tilstand der karakteren er i ro. Men dersom karakteren ønsker å utføre et hopp med tilløp, er det obligatorisk at karakteren befinner seg i tilstanden *run* for å kunne utføre hoppet. Dersom karakteren befinner seg i tilstanden *fall*, vil karakteren bli låst fast til denne tilstanden (helt til spillet starter på nytt) [13].

2.7 Evaluering

Evaluering benyttes for å avsløre feil, og hjelper utviklere med å oppdage ting de ikke har tenkt på. Det er en nyttig metode for å vise hva brukerne gjør, og evalueringen hjelper med å holde fokus på brukerne. Evaluering inkluderer tester, formelle modeller, ekspertgjennomganger og gjennomgang av sjekklister over typiske feil.

2.7.1 Brukertesting

Brukertesting omhandler testing av programvare på relevante målgrupper. Det er essensielt å la brukere prøve ut produktet underveis i utviklingen for å sikre et brukervennlig produkt for sluttbrukerne. Brukertester kan være kvalitative og/eller kvantitative, avhengig av teknikk og type spørsmål som blir stilt.

For best resultat anbefales gjennomføring av brukertest i kontrollerte omgivelser som etterligner en reel situasjon for brukerne. Utføringen av brukertestingen planlegges på

forhånd ved å opprette en testplan som tar stilling til målgruppe og funksjonalitet relatert til produktet.

2.7.2 Formativ evaluering

Formativ evaluering omhandler evaluering som påvirker utviklingen, for å sikre at designene man jobber med oppfyller kravene. Følgende evaluering kan utføres ved å kjøre enkle tester på den konseptuelle modellen. Ved å presentere konsepter og metaforer til utenforstående personer, kan man notere seg om eventuelle endringer for å tilfredsstille brukergruppen [14].

2.7.3 Opportunistisk evaluering

Opportunist evaluering er enkle tester som utføres tidlig i utviklingen, der man får tak i noen få brukere og ber dem gi rask tilbakemelding på en konkret ide. Slike tester er nyttige og billige, som kan benyttes til å avgjøre om et design bør tas videre til en prototype eller droppes [14].

3 Valg av teknologi og metode

Dette kapittelet handler om begrunnelser for beslutninger som omhandler valg av teknologi og metode benyttet i prosjektet. Basert på forkunnskaper, tilgjengelige ressurser og uttalelser fra veileder og bedrift, har vi valgt å benytte teknologi og metode som betraktes som hensiktsmessig i tilknytning til bacheloroppgaven.

3.1 Valg av design

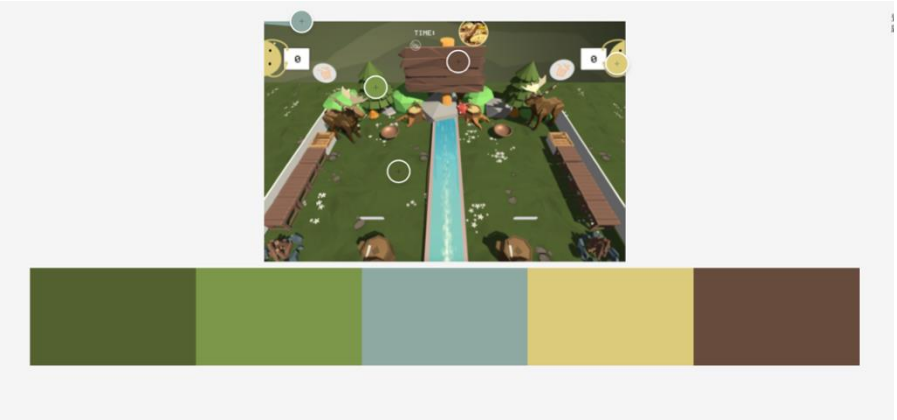
3.1.1 Spilldesign

Som nevnt i problemstilling og kapittel 2, dreier spilldesign seg om disse to essensielle begrepene: Brukervennlighet og Samarbeid.

3.1.1.1 Kunstdesign

For kunstdesign spiller ulike grafiske elementer en stor rolle for å oppnå en god brukeropplevelse. Vi ønsker at hele spillet er basert på en spesifikk kunststil. Designet på kontrollerne gjenspeiler hovedtemaet til spillet vårt. I tillegg til å benytte gunstige modeller, har vi også benyttet et tema med oversikt over ulike farger som støtter og sikrer spillets

integritet. Hovedfargen vi har valgt å vektlegge, er grønn som bidrar til en behagelig følelse [21]. Ved valg av farge for UI-elementer, ble det tatt stilling til et universelt/uniformt design. Oppsummert, vi designer og justerer rundt kundens følelser og tilbakemeldinger.



Figur 3.1-1 Fargepalett med oversikt over farger benyttet i spillet

3.1.1.2 Konseptdesign

I spillet har vi valgt å opprette flere tydelige instruksjoner og veiledninger som tydeliggjør spillerens rolle og oppgaver. Spilleren etablerer en oversikt over spillet gjennom bruk av ulike elementer i ulike situasjoner. Konseptdesignet er tilpasset til spillerens tiltenkte opplevelse.

Med hensyn på et engasjerende spill lå fokuset på målgruppens erfaring. Ved bruk av oppsettet i spillet, blir spillerne i det samme teamet tvunget til å jobbe tett sammen. Det er dermed ikke mulig å vinne konkurransen på egenhånd. Det blir et økt behov tillit til lagkameratene, og større krav til koordinering og kommunikasjon innenfor et team.

3.1.2 Interaksjonsdesign

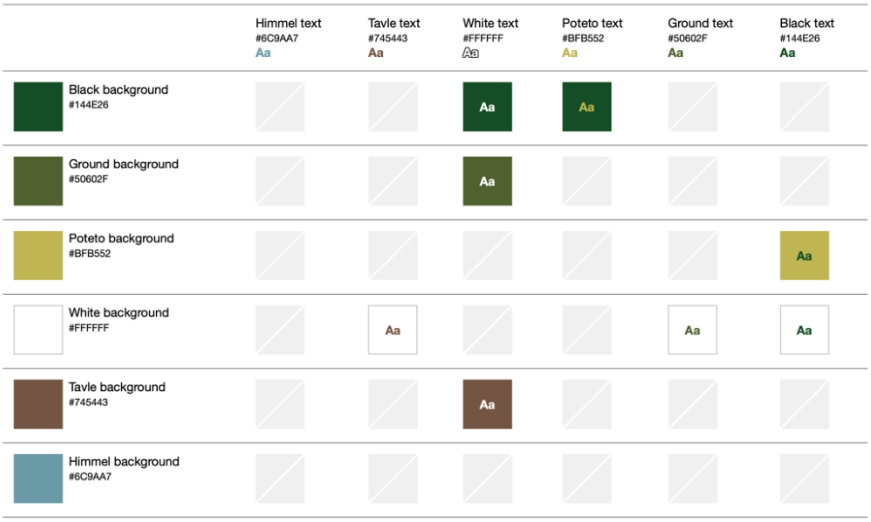
For at spillet skal være så brukervennlig som mulig, er det et naturlig valg å benytte prinsipper fra interaksjonsdesign. I henhold til vårt spill vil prinsipper i tilknytning til interaksjonsdesign vektlegge følgende handlinger:

- Rask identifisering av karakter og rolle i systemet
- En forståelse av systemet i løpet av kort tid
- En lett og forståelig kontroller
- Responsivt brukergrensesnitt.

Resultatene over skal verifiseres ved hjelp av brukertester.

3.1.2.1 Farge og kontrast

Accessible Color Matrix (se Vedlegg A, 2.10*)* ble utnevnt som fargeverktøyet for å sikre at alle tekst-elementer i både kontroll og hovedskjerm oppfyller kravene om kontrastnivå uttalt i *WCAG 2.0.* Dette sikrer at fargepalettene tar hensyn til personer med synsnedsettelser.



Figur 3.1-2 Skjermutklipp av Accesible Color Matrix

3.1.2.2 Spillerne

For å oppmuntre og glede spillerne, er det viktig å ta hensyn til etablering av kommunikasjon til og mellom brukerne. Prinsipper fra interaksjonsdesign tilbyr god støtte når det kommer til ivaretaking av brukerens interesser og oppfatninger i designprosessen. Selv om modellene til alle karakterene er likt utformet, har vi valgt å tilby hver bruker en unik farge som identifiserer karakteren deres. Tydelig informasjons som vises fram i ulike situasjoner og lettforståelig forklaring i *Loading-Scene* vil bidra til å sørge for en rask opplæring av spillet.

3.1.2.3 Kontroller

En godt designet kontroller kan gi støtte for avklaring av funksjonaliteter som kontrolleren tilbyr. Et godt design av kontrolleren er definert som enkel, men med innehold av essensielle funksjoner. Vi har valgt å forenkle designet til kontrolleren så mye som mulig. Det har blitt sørget for å benytte lettleselig og lettforståelig tekst på kontrolleren for å eliminere språkbarrierer og misforståelser. På spillkontrolleren ble det avgjort å beholde

den tradisjonelle "joystick"- designet (konsistent), slik at spillerne kan komme raskt i gang med spillet.

Figur 3.1-3 Skjermutklipp av kontrolleren team-valg side

3.2 Valg av spillmotor og redigeringsverktøy

3.2.1 Unity

For å kunne utvikle et spill som skal publiseres på Tiltspot, måtte vi utføre et valg mellom spillmotoren Unity eller et rammeverk for JavaScript. Valget falt på Unity ettersom det er en avansert spillmotor med millioner av brukere over hele verden. Selv om ingen av oss hadde erfaring med Unity fra før av, har vi funnet gode ressurser som er anvendelige og tilgjengelig på nett.

Offentlig dokumentasjon av Unity gir nøyaktige forklaringer på ulike komponenter, og inneholder detaljer om API kommunikasjon som Unity tilbyr. I forbindelse med det enkle brukergrensesnitt til Unity, kan vi bli kjent med spillutvikling på kort tid. I Unity har vi valgt å programmere i C#, et nytt programmeringsspråk som vi ikke hadde kjennskap til. Vi erkjenner at et slikt valg er faglig forsvarlig, spesielt i forbindelse med det å lære og beherske et nytt programmeringsspråk.

3.2.2 Maya og Procreate

Ved modellering av karakterer og spillobjekter for spillet, har vi valgt å benytte Maya (et kraftig 3D-modelleringsverktøy). Maya utnevnes som det avgjørende modelleringsverktøyet da det er gratis for studenter. I tillegg tilbyr den muligheten til å importere modeller direkte til Unity. For Sprite-redigering (Sprite, Vedlegg A, 2.8) har vi valgte å benytte Procreate, en app for digital tegning, Ved bruk av Procreate har vi større frihet til å konstruere designobjekter raskt og enkelt, da vi ikke er profesjonelle animasjonsdesignere. Designet kan eksporteres som PNG-fil og deretter importeres til Unity.

3.3 Valg av Designmønstre

3.3.1 Dependency Injection

Dependency injection har blitt benyttet som et ledd i forbindelse med å oppnå en løs kobling med høy kohesjon. Teknikken gjør det mulig å skille opprettelsen av et objekt fra bruken. Følgende er gunstig med tanke på å erstatte en avhengighet uten å endre en kode. Man oppnår friheten til å manipulere hvordan objektene reagerer på forskjellige metodekall, og tillater oss å teste på en fleksibel måte. Til slutt må det nevnes det at teknikken tilrettelegger for enkel injisering av stubs og mocks (Stubs og Mocks, Vedlegg, A 2.9) for verifisering av krav.

3.3.2 Model View Controller

Vi har valgt å benytte Model-View-Controller på kontrollersiden, ettersom en slik struktur sikrer løse koblinger mellom komponenter og bidrar til parallell utvikling. En slik utvikling gjør det mulig for utviklerteamet å fordele ansvar for forskjellige områder av kontrolleren, ved å tydeliggjøre ulike ansvarsområder gjennom bruk av MVC. Følgende metode er også gunstig med tanke på gjenbruk og modifisering av kode. Ved å sikre en løs kobling, kan andre adaptere og reformere koden til sitt eget behov. Dette er et godt utgangpunkt for å sikre et modifiserbart og utvidbart system, som legger til rette for videre utvikling av andre utviklere.

3.3.3 Polymorfisme

Vi har valgt å benytte polymorfisme da det legger til rette for at objekter kan innta ulike former. Bruken av polymorfisme bidrar til å sikre et likt grensesnitt for et rammeverk som

inntar en mengde av forskjellige objekter gjennom det samme grensesnittet. Ved opprettelse av en ny type objekt, er det ikke behov for å endre rammeverket for å imøtekomme den nye objekttypen. Således er polymorfisme betydelig i vårt produkt for å skille mellom de ulike kjøkkenbenkene og kjøkkenredskapene.

3.3.4 State Machine

State Machine er et kompakt designmønster for representasjon og visualisering av komplekse regler og betingelser. Følgende designmønster egner seg også for prosessering av ulike inndata, og dersom mønsteret er godt implementert, kan state machine regnes som selvdokumenterende ettersom hver logisk tilstand representerer en fysisk tilstand. En State machine kan bidra til effektiv kjøring og kompakt representasjon som kan opprettholdes lett, ved fornuftig lagring av regler som styrer tilstandsendringer. Fornuftig lagring kan for eksempel være lagring i en tabell.

3.4 Valg av evalueringsform

3.4.1 Brukertesting

Siden brukertesting er en av de viktigste virkemidlene for å evaluere resultatet av dette prosjektet valgte vi å benytte oss av to ulike typer brukertester. Vurderingstesting ble benyttet til å teste brukervennligheten til den grunnleggende funksjonen i spillet, og avdekke eventuelle feil og mangler underveis i prosjektet. Mot slutten av utviklingsprosessen brukte vi godkjenningstesting for å kunne evaluere spillets brukervennlighet som helhet. I denne brukertesten ønsket vi også å gjøre observasjoner rundt samarbeidet mellom spillerne. På grunn av situasjonen med Covid-19 har vi ikke hatt mulighet til å utføre brukertester med så mange deltakere samtidig som opprinnelig tenkt. Derfor har vi kun utført tester med maksimum fem deltakere på en gang. Deltakerne har hovedsakelig har vært begrenset til venner og familie.

I forkant av brukertestene ble det utformet en testplan som kartlegger testenes mål og prosesser for å få mest mulig utbytte av testingen. Det ble totalt gjennomført ni vurderingstester, hvor en person deltok av gangen. Dette var et valg som ble gjort på bakgrunn av smittesituasjonen knyttet til Covid-19. I disse testene ble det benyttet personer med en del spillerfaring fra før. Dette er fordi vi hovedsakelig ønsket tilbakemelding på brukervennligheten til de grunnleggende funksjonene, og det var derfor nyttig å bruke noen som hadde et godt sammenligningsgrunnlag. Feil og mangler som ble avdekket i disse

testene ble utbedret i etterkant av hver test, slik at vi også fikk testet endringene underveis i prosessen. I godkjenningstesten deltok totalt fem personer, hvor tre spilte på lag mot de to andre. Gitt situasjonen ønsket vi likevel at deltakerne skulle representere målgruppen på best mulig måte. Gruppen inneholdt derfor både personer med mye spillerfaring og personer med lite/ingen spillererfaring. To av deltakerne hadde erfaring med lignende spill.

Begge testene ble utført på samme måte, men med noe variasjon i innhold. Brukertestene ble utført i et kontrollert miljø som var tilrettelagt slik at den lignet mest mulig på en reell situasjon for spillerne. Testingen foregikk i en stue og det ble brukt en PC som konsoll for Tiltspot, og deltakerne stilte med egen mobil til kontroll. Før deltakerne fikk prøve spillet ble de stilt en rekke demografiske spørsmål om seg selv. Under testingen ble de oppfordret til å prøve seg frem og bli kjent med funksjonene. Deretter ble deltakerne gitt noen oppgaver designet til å teste viktig funksjonalitet beskrevet i kravdokumentasjonen. Avslutningsvis ble de stilt noen oppfølgingsspørsmål. Testpersonellet noterte observasjonene som ble gjort underveis og resultatet ble oppsummert i en endelig rapport.

3.4.2 Formativ evaluering

Underveis i utviklingsprosessen har gruppen hatt møter og samtaler med oppgavestiller hvor design og konsept har vært i fokus. Ved å vise frem prototyper og konseptuelle modeller for produkteier har vi kunnet sikre at design og ide tilfredsstiller kravene som er stilt. Tilbakemeldingene vi mottok fra disse møtene og samtalene ble brukt til å lede gruppen på rett spor videre i utviklingen.

3.4.3 Opportunistisk evaluering

Tidlig i utviklingsfasen benyttet gruppemedlemmene seg av ulike bekjentskaper for å lufte forskjellige ideer til konsept. Dette ble gjort ved at noen utvalgte personer fikk presentert en ide og fikk deretter muligheten til å uttrykke sine meninger om den. Disse tilbakemeldingene ble tatt imot av medlemmene i gruppen og ble tatt hensyn til i videre diskusjoner rundt konseptet.

3.5 Valg av Programmeringsteknikk

3.5.1 Structs

En struct er en strukturtype /verditype som kan innkapsle data og relatert funksjonalitet. Til vanlig benyttes strukturtyper for å konstruere små datasentraliserte typer ved innhold av

lite eller ingen oppførsel. Ved opprettelse av en struct, har variabelen den faktiske verdien og ikke en referanse til minnet. Dersom struct-en tilordnes en ny variabel, kopieres den. Endringer som blir utført på den ene kopien, vil ikke påvirke den andre kopien. Slik kan struct benyttes der det ikke er behov for en referansevariabel [10].

Vi anvender structs der det er behov for en liten datastruktur for å holde på data som ikke skal modifiseres senere, altså der struct egner seg bedre enn class. Det som skiller struct fra class er hvordan de overføres til metoder. Ved overføring av et class-objekt, vil kompilatoren kopiere adresseverdien inn til parameteren som er lagret på metode-stacken (Stack, Vedlegg A, 2.4). Men ved overføring av et struct-objekt, vil kompilatoren kopiere hele strukturinnholdet til et nytt struct-objekt. Av den grunn vil stacken bli større enn stacken i tilfellet med class-objektet. Det betyr også at det blir raskere å aksessere data, ettersom det ikke er noe hopp til minnet.

3.5.2 Attributter

C# tilbyr muligheten til å benytte attributter for å styre oppførselen til objekter. Attributter kan plasseres på de fleste deklarasjoner, men et spesifikt attributt kan begrense bruken av andre deklarasjonstyper. I C# angis et attributt ved å plassere navnet på attributtet i klammeparanteser over deklarasjonen av entiteten det gjelder.
Attributter anvendes under kjøring og benyttes ofte for å validere egenskaper til objekter som skapes. Følgende egenskap passer godt sammen med designmønsteret MVC, da den sikrer løs kobling ved å blant annet sette verdier til objekter som benytter attributtene [11].

Attributter anvendes under kjøring og benyttes ofte for å validere egenskaper til objekter som skapes. Følgende egenskap passer godt sammen med designmønsteret MVC, da den sikrer løs kobling ved å blant annet sette verdier til objekter som benytter attributtene.

Følgende eksempel er hentet fra vår kildekode, og demonstrerer bruken av attributtet *Serialiazable.*

```
[Serializable]
public struct foodRecipeItem
{
    public string foodContainerName;
    public string thumbnail;
    public List<ingredientStruct> ingredients;
}
```

Figur 3.5-1Skjermutklipp av kildekode RecipeGen.cs

Attributtet Serialiazable benyttes i dette tilfellet for å anvende spesifikke karakteristiske trekk til en struct, og vil transformere data i internminnet på en datamaskin til en sekvens av byter som kan overføres over nettverket.
Attributtene gjør at koden blir mer kompakt, mer leselig og tydelig.

3.6 Valg av utviklingsmetoder

3.6.1 Scrum

Scrum er et iterativt, smidig og inkrementelt rammeverk som benyttes for å optimalisere produktutviklingen. Det er tre hovedroller i Scrum-teamet: produkteier, utviklingsteam og Scrum master. Produkteieren er ansvarlig for prosjektets interessenter, og skal representere ønskene og behovene til komiteen. Utviklerteamet organiserer seg selv og styrer sitt eget arbeid. Det er de som utfører selve produktutbyggingen, og har dermed ansvar for å levere inkrementelle leveranser av et «ferdig» produkt i løpet av hver sprint. På denne måten sikres det at en potensielt verdifull versjon er alltid tilgjengelig. En Scrum master som har rolle å sørge for at Scrum er forstått og etterleves, ved å sørge for at Scrum-teamet adlyder teori, regler og praksis.

Før arbeidet kan settes i gang, er det nødvendig med en sprint planleggingsmøte. Planleggingsmøtet er et møte mellom produkteier, Scrum master og Scrum team. I dette møtet blir deltakerne presentert for et *product backlog*, en liste med funksjoner utarbeidet av produkteier. På bakgrunn av product backlogget, skal teamet finne omfanget som de har tro på at de vil klare å gjennomføre i løpet av neste sprint.
Utviklerne velger oppgaver på bakgrunn av et produkt backlog. Listen består av oppgaver med estimater og prioritet, og det er opp til utviklerteamet å estimere, ettersom det er de

som har best kjennskap til sin kapasitet og hastighet. Utviklerne deler opp arbeidet sitt i mindre oppgaver, som skal ferdigstilles innen et gitt tidsrom (sprint).

Fremgangen i sprinten fanges opp ved å gjennomføre daglige Scrum-møter der de informer hverandre om oppgaver de har utført, eventuelle problemer og hindringer. Disse møtene er essensielle for å holde oversikt over lagets hastighet og kapasitet. Det skal også sørges for å utføre en re-estimering av oppgaver slik at man kan fremstille en burn-down chart som gir en visuell representasjon av gjenværende oppgaver. Dette benyttes for å styre sprinten.

Sprinten avsluttes med en sprint review hvor utviklerteamet demonstrerer arbeidet som ble utført, til utvalgte interessenter, som da gir tilbakemelding til teamet. Til slutt rundes det av med en sprint retrospective mellom deltagerne i utviklerteamet, hvor de reflekterer over egen innsats og arbeidsform. Det opprettes også en plan for forbedringer som skal implementeres i neste sprint [3].

Vi valgte å benytte Scrum for å sikre en agil utvikling av prosjektet. Følgende utviklingsmetode tilrettelegger for fullføring av prosjekt på en rask og effektiv måte. Vi har tidligere benyttet Scrum i forbindelse med prosjekter relatert til studiet, og har altså god kjennskap til bruken av Scrum. Det som også påvirket valget av utviklingsmetoden Scrum, var behovet for nye ideer og forslag for å kunne utvikle et produkt i riktig retning.

For en brukerorientert spillutvikling er det fortrinnsvis urimelig å spesifisere alle detaljer på forhånd. Det er alltid behov for tilbakemeldinger fra kunder, får å kunne utføre relevante tiltak for å sikte et godt utviklet produktkonsept. Med Scrum kan dette oppnås ved bruk av ulike Sprint Reviews. Da Scrum også benytter verdier fra *Agile Manifesto* (Agile Manifesto, Vedlegg A, 2.7) direkte, som passer godt sammen med utviklingsmetoden interaksjonsdesign, konkluderte vi med at dette var det riktige valget.

3.6.2 Arbeids- og rollefordeling

I begynnelsen hadde vi ulike synspunkt når det kom til valget av spillinnhold. Ved å sette opp en konkret plan mot målet, har vi eliminert de fleste konfliktene. Den konkrete planen

og det tydelige målet, bidrar til en god arbeidsfordeling. Ved bruk av ulike verktøy for samarbeid og parallell utvikling, har samarbeidet utviklet seg optimalt (selv om vi har møtt på mostand i forbindelse med Coronaviruset).

En naturlig oppdeling av ansvarsområder for prosjektet ble følgende:

Kontroller-utvikler

Dette innebærer utformingen av kontrolleren ved bruk av HTML, CSS og Javascript.

Spill-utvikler

Dette innebærer utformingen av spillhandlingen ved bruk av C#.

UI- UX designer

Design tilknyttet til kontrolleren og spillet. Dette gjelder både design av brukergrensesnittet og animasjon for spill-instruksjoner.

API kommunikasjon

API kommunikasjon for å overføre meldinger til plattformen Tiltspot. Dette benyttes i forbindelse med flerspiller-spill-funksjonaliteten.

Selv om vi hadde delt opp oppgaven i tydelige ansvarsområder, arbeidet vi likevel på tvers av disse ansvarsområdene. Ansvarsområdene ble kun benyttet for å sikre en effektiv og opprettholdbar arbeidsprosess for å redusere tiden som måtte benyttes for å sette seg inn i teknologien tilhørende de ulike feltene. Fordelingen av oppgaver er delvis basert på trivsel, utfordringer for å styrke kompetansen tilknyttet spillutvikling, svakheter og styrker ved hver person i gruppen. På denne måten har teamet forsikret en rettferdig og anvendelig fordeling av oppgaver.

3.7 Valg av samhandlingsverktøy

3.7.1 Trello

Trello er en nettside som tilbyr en digital Kanban-tavle. Som følge av Corona-viruset, har vi måttet arbeide hjemme (individuelt). Derfor er Trello et kraftig verktøy for å administrere prosjektet vårt. Det gir en oversikt over alle oppgaver, hvilken status de befinner seg i, tydeliggjøring av større oppgaver ved bruk av sjekklister og ansvarlig medlem av oppgaven, for å unngå konflikter. Vi kan oppnå en kontinuerlig god oversikt over oppgaver selv om vi ikke sitter sammen. Trello tilbyr god støtte for organisering av prosjektet gjennom et enkelt brukergrensesnitt.

3.7.2 Discord

Diskusjon og kommunikasjon rundt prosjektoppgaven foregår ved bruk av kommunikasjonsverktøyet Discord. I starten hadde vi tenkt å holde oss unna dette kommunikasjonsverktøyet, siden ingen av oss hadde noe erfaring med å benytte Discord i prosjekter. Vår kunde Tiltspot anbefalte oss å benytte Discord for å holde kommunikasjon og kontinuerlige oppdateringer gående. Først av alt har de opprettet en egen kanal på Discord. I denne kanalen befinner en rekke mennesker seg, med mye erfaring tilknyttet spillutvikling ved bruk av Unity og interesse for Tiltspot.

Discord tilbyr oss en plattform for å kommunisere med partene tekstlig og verbalt. I Discord har vi også mulighet til å opprette mange sub-kanaler under, et eksempel kan være å opprette “design” sub-kanal for å eksplisitt diskutere om modellering og animasjon. Dette tilrettelegger for organisering av informasjon som vi har delt og formidlet verbalt. I tillegg har Discord mange nyttige funksjoner, et eksempel er skjermdelingsfunksjonen. Oppsummert er Discord et kraftig kommunikasjonsverktøy som sikrer effektivt og enkelt samarbeid.

4 Resultater

4.1 Vitenskapelige resultater

Problemstillingen fra kapittel 1.3 stiller spørsmål til hvordan man kan benytte spilldesign til å skape et brukervennlig teambasert spill for Tiltspot. Ved å teste og undersøke de tilgjengelige spillene på spillplattformen Tiltspot, og sammenligne spill med størst brukerskare som har fått positiv og betraktelig med respons, kan man ut ifra det formulere en felles oppfatning av hva slags spill brukerne på plattformen ønsker mer av. Problemstillingen besvares delvis ved å sammenligne elementer fra de ulike spillene, og sammenligne og konstruere et spill med innhold av de ulike utvalgte elementene. Ved å ha benyttet en empirisk tilnærmingsmåte som er bygget opp av observasjoner, spørreundersøkelser og dokumentanalyse har vi kommet et godt stykke på vei. Kontrolleren har et brukergrensesnitt som benytter responsive CSS-klasser. Gruppering av tomme spillobjekter og canvas håndtering bidrar til å sikre responsivt UI design tilknyttet

spillet. På den andre siden vil Tiltspot ta for seg fremvisningen av spillet tilpasset pc/tv skjermen.

Følgende elementer fra Unity er blitt fremmet for å sikre et responsivt UI design:

- **Anchors and pivots:** Ulike elementer tilhørende spillbrettet er blitt ankret fast i sentrum av rektangulærere transformer.
- **Canvas Scaler:** Skaleringen tar hensyn til skjermstørrelsen og UI-elementers posisjon i lerretet.
- **Aspect Ratio Fitter:** Aspekt modus er blitt angitt til å ivareta størrelsesforholdet i henhold til forelder-containeren.

Kontrolleren er utformet ved bruk av designmønster MVC. Ettersom spillet er utviklet i spillmotoren Unity, har vi benyttet designmønstre som dependency injection, polymorfisme og state machine (Kap. 3).

Vi har valgt å referer til produkt- og designresultater under ingeniørfaglige resultater. For videre diskusjon av de vitenskapelige resultatene, se kapittel 5.1

4.2 Ingeniørfaglige Resultater

Prosjektets mål omhandler opprettelsen av et flerspiller-spill som tilfredsstiller kravene satt i visjonsdokumentet (Vedlegg B, kapitel 5 og 6). I følgende kapittel beskrives produktets gjennomførbarhet og fullførbarhet i henhold til kravene. Se Vedlegg D, for installasjon og kjøring av produkt.

4.2.1 Evaluering

Kapitlet inneholder en kort oppsummering av resultatene fra samtlige brukertester utført i forbindelse med prosjektet.

4.2.1.1 Kontrollert brukertesting

Evalueringstesting

Tilbakemeldingene vi fikk i vurderingstestene handlet blant annet om karakterens bevegelser. Dette gikk ut på at den beveget seg for sakte og at bevegelsene var ujevne. Vi fikk også kommentarer på at karakterens farge var for mørk, slik at den var litt vanskelig å se mot bakgrunnen. Andre tilbakemeldinger gikk ut på at det var utydelig hvordan forskjellige ingredienser skulle tilberedes, og hvor man kunne kaste mat. Alle tilbakemeldinger ble tatt i

betraktning og endringer ble gjort med hensyn på disse. Tester utført i etterkant av utbedringene tydet på at problemene var løst. Ellers fikk vi generelt gode tilbakemeldinger på design og konsept.

En mer utfyllende beskrivelse av resultatene finnes i vedlegg G, *Brukertestplan*.

Godkjenningstesting

De to deltakerne med erfaring fra lignende spill forstod konseptet kjapt og uten problemer. De tre andre deltakerne brukte litt lengre tid på å lese instruksjonene. Laget med tre deltakere utførte oppgavene på relativt kort tid. Testansvarlig opplevde at laget kommuniserte bra, og hjalp hverandre med å utføre oppgavene. Laget med to deltakere brukte noe lengre tid på oppgavene, da de måtte stoppe for å lese instruksjonene en gang til. Ellers gikk gjennomføringen uten problemer. Dette laget hadde også god kommunikasjon og samarbeidet om oppgavene.

Helhetsinntrykket fra deltakerne var at dette var et enkelt og underholdende spill. Vi fikk også ros for fint design. En av deltakerne med mye spillerfaring kommenterte at oppgavene var i letteste laget og at de ikke var utfordrende nok. Sett bort ifra det fikk vi utelukkende positive tilbakemeldinger.

En mer utfyllende beskrivelse av resultatene finnes i vedlegg G, *Brukertestplan*.

4.2.1.2 Formativ evaluering

I forbindelse med møtene og samtalene med oppgavestiller har vi mottatt tilbakemeldinger på vårt produkt underveis i utviklingen. Tilbakemeldingene vi har fått har vært svært positive og vi har hele tiden blitt forsikret om at vi er på riktig vei. Ved fremvisning av vår prototype, hvor vi hadde en trestokk som representerte et samlebånd hvor matvarene var tilgengelig for spillerne, fikk vi forslag om å endre denne til å se ut som en elv. Begrunnelsen for forslaget var at det virker mer naturtro at matvarene flyter nedover på en elv enn at de beveger seg på en horisontal trestamme. Denne tilbakemeldingen tok vi til oss og implementerte forslaget.

4.2.1.3 Opportunistisk evaluering

Ved å pitche ideen til konseptet vårt tidlig i utviklingsfasen har vi mottatt en rekke tilbakemeldinger. Disse tilbakemeldingene har hovedsakelig vært positive og alle mente at dette virket som et gøyalt spillkonsept. Noen få mente derimot at spillet ville være litt for avansert til å kunne gjennomføres i en bacheloroppgave. På bakgrunn av disse tilbakemeldingene valgte vi å gå for den opprinnelige ideen, men med et noe forenklet konsept. Vi tok utgangspunkt i en enklere modell hvor vi fokuserte på de viktigste funksjonalitetene i konseptet, og hvor vi enkelt kunne bygge videre på dersom tiden strakk til.

4.2.2 Ikke Funksjonelle egenskaper

Spillet er bygget med løst koblede komponenter, plugins og API-er tilbydd av Tiltspot. Produktet benytter flere teknologier og designmønster for å oppfylle dette. Spillet er utviklet med fri programvare, utmerket for kryss-plattforms-utvikling og tilrettelegger for enkel teknisk støtte.

4.2.2.1 Utviklingsmiljø og vedlikehold

Selve spillet er utviklet i C# med Unity, mens kontrolleren er opprettet ved bruk av HTML, CSS og Javascript. Produktet benytter også ressurser hentet fra Unitys ressurs-butikk. Det har blitt benyttet 3D elementer for selve utføringen av spillhandlingen. 2D UI-design benyttes for visualiseringen av spill-handlings-instruksjoner, oversikt over gjenværende tid og poeng.

Skriving av kildekode ble utført i utviklingsmiljøet *Visual Studio 2017* og *Sublime Text*. Opprettelsen og videreutviklingen av 3D-elementer og 2D-sprites foregikk i Maya og Photoshop.

4.2.2.2 Plattform og operativsystem

Produktet skal i utgangspunktet kunne benyttes av alle operativsystemer som i besittelse av en nettleser. Kontrolleren og spillet er testet til å operere på følgende nettlesere:

- Google Chrome, versjon 81.0.4044.129
- Safari, versjon 13.0.5 (15608.5.11)
- Mozilla Firefox 67.0.1

4.2.2.3 Andre krav

Lav kostand og høy effektivitet er blitt sikret ved bruk av denne plattformen. Spillet som opprettes har som formål å kunne benyttes av personer i mange ulike land. Dette er et spill som skal være tilgjengelig på en nettbasert spillplattform som muliggjør bruken av smarttelefoner som kontrollere, og datamaskinen som konsoll. Ingen eksterne enheter er nødvendige, og ingen nedlastninger er nødvendige. Tiltspot.tv tilrettelegger for utvikling av spill til deres plattform ved å tilby funksjonelle plugins og API-er.

4.2.3 Funksjonelle egenskaper

4.2.3.1 Oppstart av spill

Spillerne presenteres for en team-valg-side som støtter opptil åtte spillere. For å starte spillet må brukerne angi et klar-status.

Figur 4.2-1 Team-Valg (Hovedskjerm)

4.2.3.2Utforming av oppskriftstavle og transportbånd

Produktet tilbyr en oppskriftstavle som gir oversikt over aktive oppskrifter i spillet. Dette er en oversikt over nødvendige ingredienser behøvd for å ferdigstille en matrett.

Figur 4.2-2 Oppskriftstavle som gir oversikt over aktive oppskrifter

De hvite sirkulære figurene angir hvilket element som må være på plass i tallerkenen før de resterende elementene kan tilsettes. I dette tilfellet understrekes det at lammekjøttet må være på plass i tallerkenen, før en kan tilsette kåla. Det er en bestemt rekkefølge for plassering av ingredienser i tallerkenen.

Figur 4.2-3 Transportbånd for utlevering av tallerkener og matvarer

Et transportbånd formes som elv benyttes for utlevering av råvarer og tallerken. Dette sikrer en rettferdig fordeling av råvarer og tallerken. Her gjelder det å være fokusert og rask, da det ikke er mulig å sette råvarer og tallerken på bakken igjen, etter å ha plukket de opp. Eneste mulighet er å plassere de på et bord eller kaste de i *søppelkassen, les mer om følgende i kap. 4.2.3.5*

4.2.3.3 Matbeholder

Stekefat

Stekefatet arver fra klasen matbeholder som tilbyr mulighet for innsetting av matvarer. Ettersom stekefatet benyttes til steking, er det begrensninger på hvilke matvarer det er som kan innsettes.

Figur 4.2-4 Stekefat tilhørende bålet

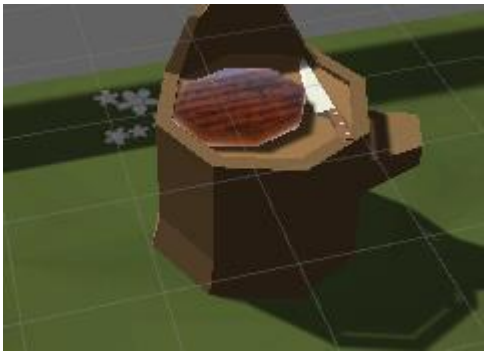
Serveringsfat

I likhet med stekeplaten arver også serveringsfatet fra klassen matbeholder. Serveringsfatet godkjenner midlertidig kun innsetting av objekter som tilhører klassen «Fårikål», for å sikre at spillerne konstruerer en matrett som eksisterer i oppskriftstavlen. Denne begrensningen forhindrer brukeren fra å utføre feil.

Figur 4.2-5 Serveringsfat

4.2.3.4 Fårikål og oppskriftsbeholder

Fårikål er en klasse tilknyttet et spillobjekt som arver fra klassen matbeholder. Spillobjektet fårikål har påtatt seg rollen som oppskriftsbeholder, og er det første elementet som plasseres på tallerkenen før resten av ingrediensene kan tilsettes.



Figur 4.2-6 Spillobjekt Fårikål

Oppskriftstavlen gir oversikt over spillobjekter med fårikål-funksjonalitet.

Figur 4.2-7 Fårikål-oppskrift

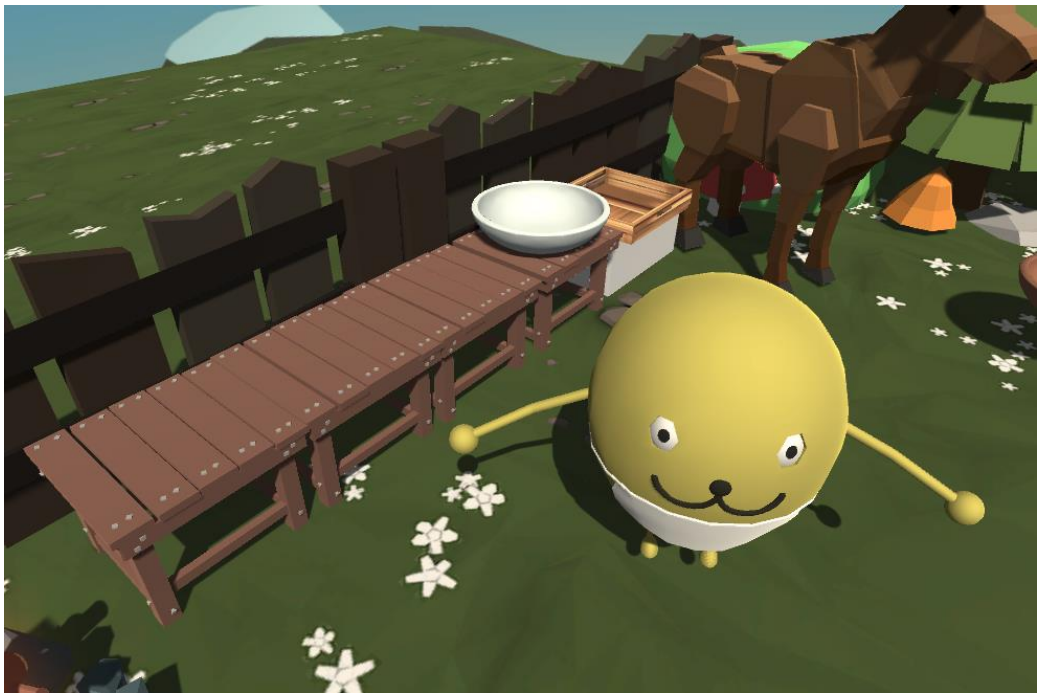
I tilfellet ovenfor (figur 4.2-7), har kjøtteobjektet matbeholder-funksjonaliteten. Kålen er kun en ingrediens som tilsettes etter at kjøttobjektet er blitt plassert på serveringsfatet.

4.2.3.5 Oppsett av kjøkkenbenker

Alle objekter som tillater plassering av verktøy og mat på deres overflate har blitt kategorisert som kjøkkenbenk.

Vanlige kjøkkenbenker

De vanlige kjøkkenbenkene benyttes kun til plassering av matvarer og redskaper som karakterene har i hendene (da matvarer og kjøkkenredskaper ikke kan plasseres på bakken). Utover det har benkene ingen annen funksjonalitet.



Figur 4.2-8 Vanlige kjøkkenbenker

Skjærebrett for kutting av råvarer

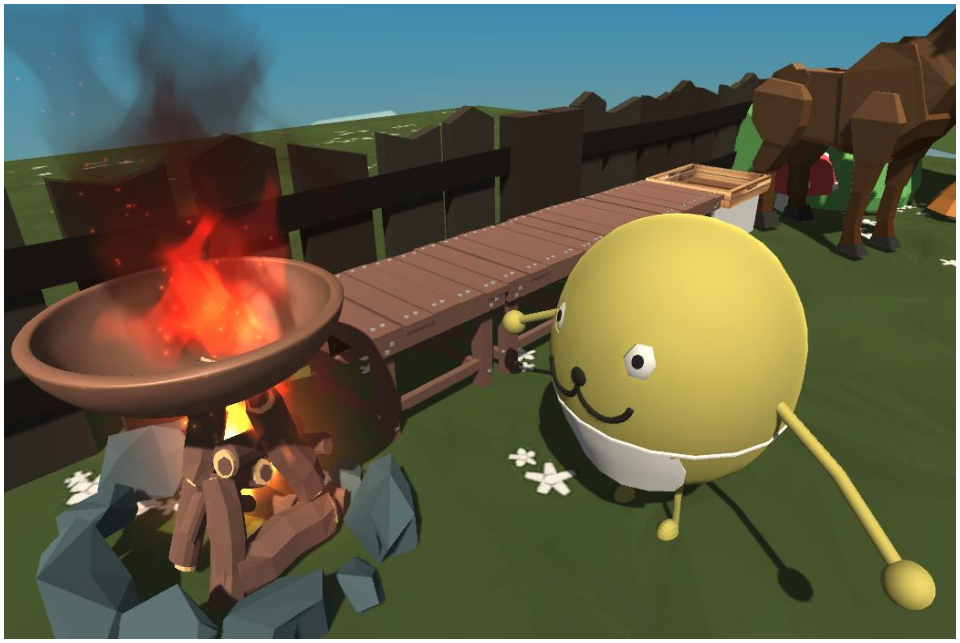
Skjærebrettet arver metoder og variabler definert i klassen kjøkkenbenk. Utenom funksjonaliteten som befinner seg i de vanlige kjøkkenbenkene, tilbys muligheten til kutting av grønnsaker.

Figur 4.2-9 Skjærebrett-kjøkkenbenk

Kutte-fremdriften visualiseres ved hjelp av en progress-bar. Slik gjøres en oppmerksom på å benytte tiden man har til rådighet, effektivt.

Bål for steking av råvarer

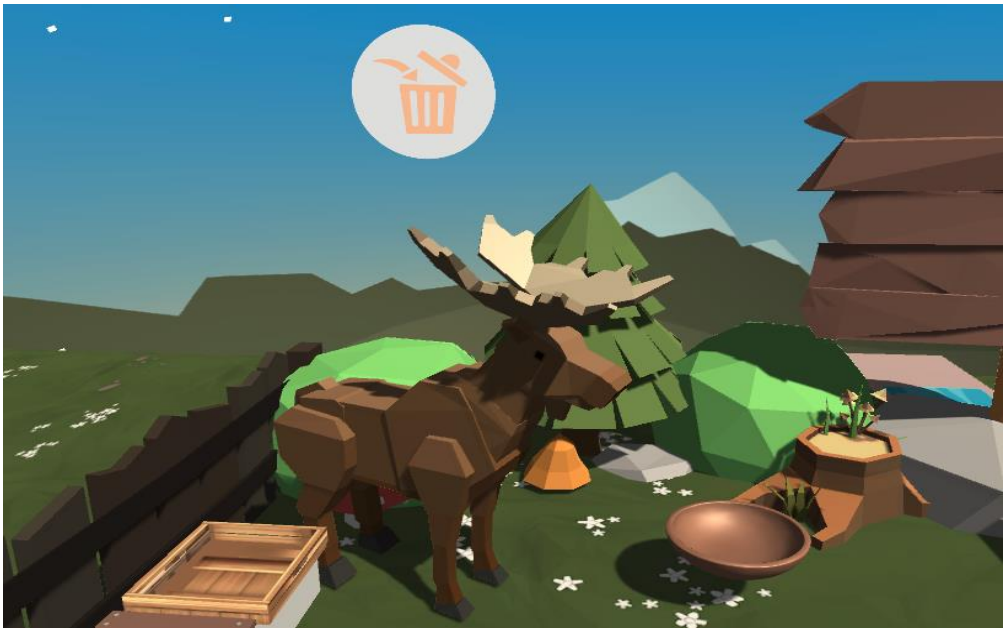
Bålet arver også fra klassen vanlig-kjøkkenbenk. Bålet tilbyr funksjonalitet tilknyttet tilberedning og steking av råvarer. Det er også blitt benyttet en progress-bar for å visualisere fremdriften. I motsetning til progress-baren som benyttes i forbindelse med kutting, er progress-baren som er tilknyttet bålet basert på en timer. Dette gir brukeren mulighet til parallell utføring av oppgaver.



Figur 4.2-10 Bål for tilberedning og steking av råvarer

Kompostbeholder for avfallshåndtering

En elg er blitt benyttet for visualiseringen av en kompostbeholder. Kompostbeholderen arver også fra klassen kjøkkenbenk, så det er mulig å plassere kjøkkenredskaper og råvarer på elgen. Elgens hoved-funksjonalitet omhandler levering av avfall. Ettersom elgen fordøyer råvarene i spillet, fungerer den også som et element som symboliserer redusert matsvinn.



Figur 4.2-11 Kompostbeholder for nedbryting av organisk avfall

Kjøkkenbenk for innlevering av ferdigstilte matretter

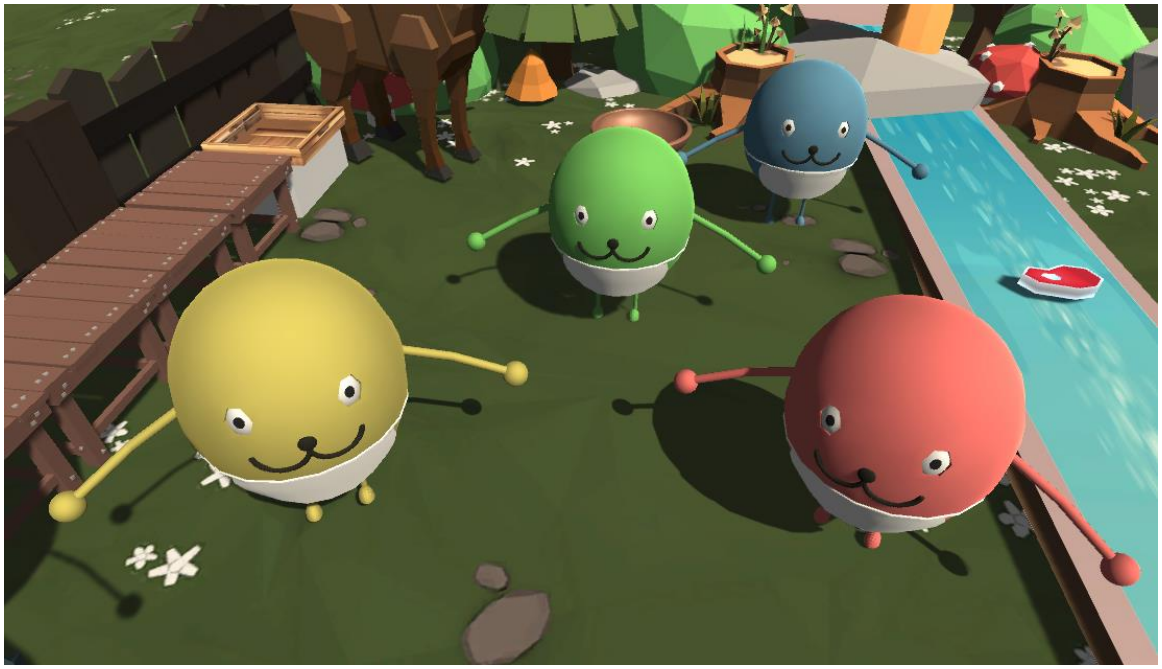
Brukeren leverer inn den ferdigstilte matretten til objektet som har ansvar for behandling av innlevert matrett. Kjøkkenbenken nedenfor arver nokså fra klassen vanlig-kjøkkenbenk. Benken håndterer innlevering og beregning av poeng, og vil dermed være i besittelse av en rekke viktige funksjoner som omhandler poengestimering. Ved innlevering av en matrett tar kjøkkenbenken kontakt med et spillobjekt som opererer i bakgrunnen, og som sørger for annonsering av vinner ved utløpet av tiden.

Figur 4.2-12 Kjøkkenbenk – Håndtering av ferdigstilte matretter

4.2.3.6 Karakter og bevegelse

Utformingen av karakteren er basert på et kulelegeme. Brukeren har mulighet til å forflytte karakteren vertikalt og horisontalt. I tillegg er det mulig å rotere karakteren i den retningen

forflytningen foregår langs.



Figur 4.2-13 Spillkarakter med ulike skinn

Et tilfeldig skinn legges til karakteren for å skille mellom de ulike spill-karakterene.

4.2.3.7 Kontrolleroppsett

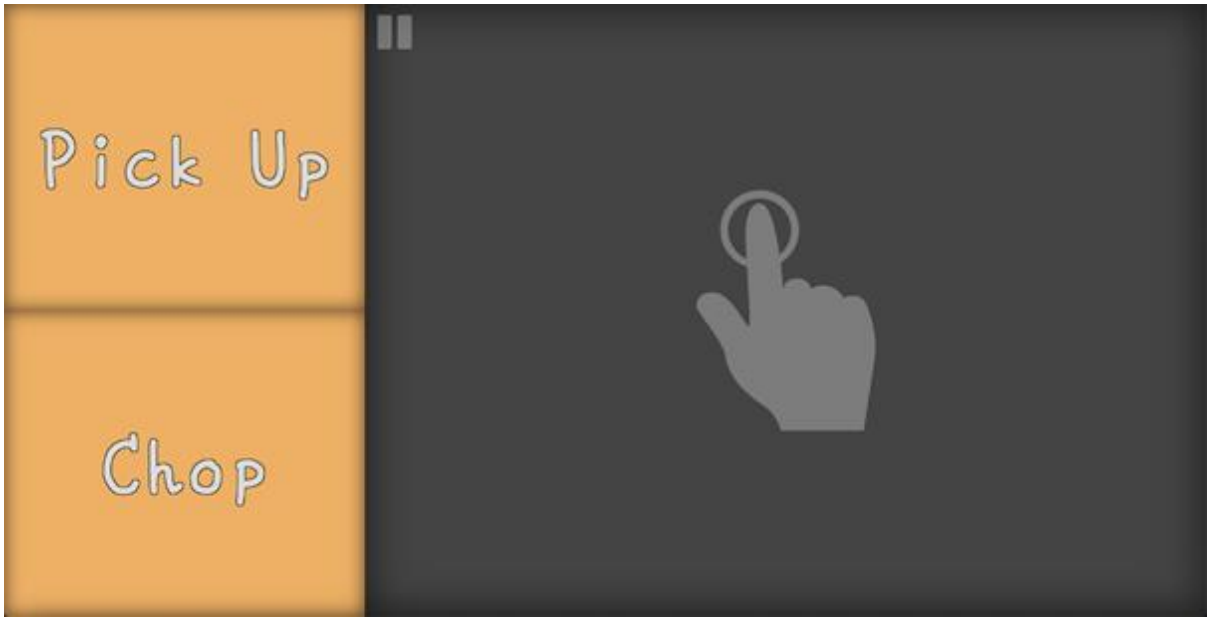
Team valg

Kontrollerer gir spilleren mulighet til å velge mellom to lag. Etter at det angis spillerens tildelte karakterfarge og foretrukket lag presenteres brukeren for selve spillhandlingen (etter at ready knappen er blitt aktivert).

Figur 4.2-14 Team-Valg (Kontroller-skjerm)

Bevegelse og handling

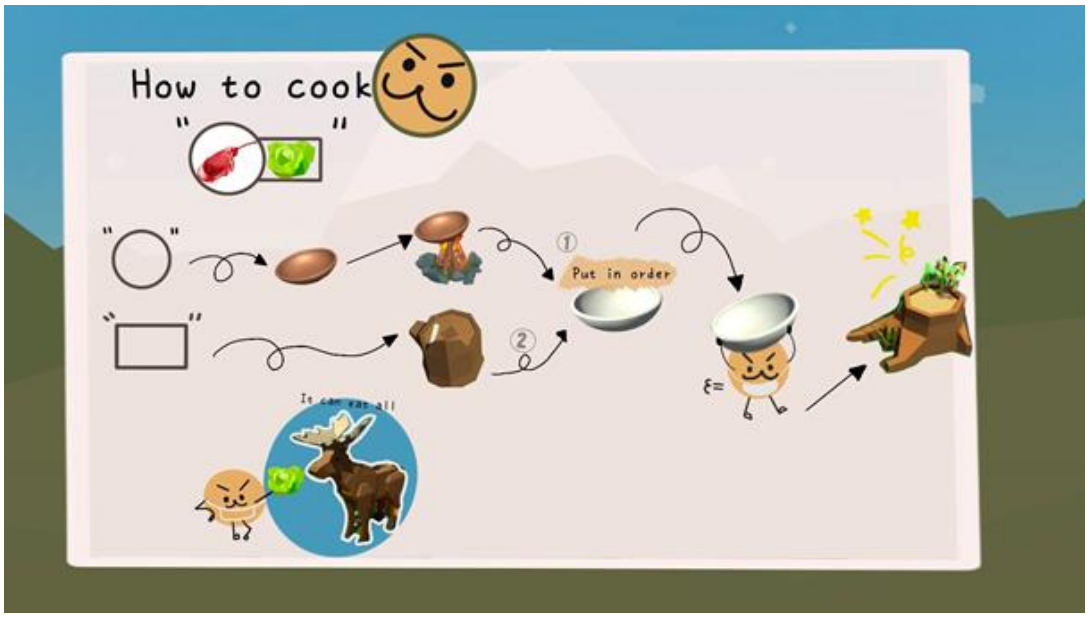
Kontrolleren tilbyr to knapper og en joystick- Joysticken benyttes for bevegelse av karakter (vertikal og horisontal forflytning). Knappene benyttes for kutting og plukking av råvarer og redskaper. Ved behov for pause, kan brukeren velge å benytte pauseknappen (øverst til venstre hjørne i joystick vinduet).



Figur 4.2-15 Kontroller med joystick og to knapper

Animasjon for opplæring

Ved initiering av et nytt spill, tilbys brukeren en rask opplæring av spillet. Animasjonen klargjør de ulike kjøkkenredskapenes brukerfunksjonalitet ved å understreke rekkefølgen av tilberedning og innsending.



Figur 4.2-16 Visuell forklaring på kjøkkenredskapenes bruksområde

4.3 Administrative resultater

4.3.1 Utviklingsprosess

Scrum-rammeverket ble benyttet i forbindelse med utviklingen av prosjektet. Følgende kapittel referer til dokumentasjon relatert til utviklingsprosessen.

4.3.1.1 User Stories

Bruken av user stories er angitt og utdypet i kravdokumentet (Vedlegg C, kapittel 2).

4.3.1.2 Sprints

Vedlagt finnes dokumentet med oversikt over sprintene, *Scrum – Sprint Review og Sprint Planning* (Vedlegg F).

4.3.2 Timeregnskap

Timeforbruket for de ulike aktiviteter er dokumentert ved bruk av ukentlige statusrapporter og timelister. Følgende er dokumentert i prosjekthåndboka (Vedlegg E, kapittel 4).

4.3.3 Måloppnåelse i forhold til framdriftsplan

Framdriftsplanen tilhørende prosjektet er presentert i form av et Gantt-diagram (Vedlegg E, 2.1). Gantt-diagrammet har gjennomgått to minimale endringer i løpet av prosjektarbeidet, som følge av omprioritering av oppgaver og tilegning av mer kunnskap om tidsforbruk. Endringene omhandler kategorinavnene «Oppsett av prosjekt» og «Øvrig Dokumentasjon». «Oppsett av prosjekt» ble endret til henholdsvis «Programmering», mens «Øvrig Dokumentasjon» ble endret til «Hovedrapport». Kategorien «Hovedrapport» skal dekke all dokumentasjon tilknyttet hovedrapporten, i tillegg til refererte vedlegg i hovedrapporten. For kategorien «Møter», ble det anslått å benytte 55 timer. I virkeligheten benyttet vi 78 timer. For «Programmering» ble det anslått 925 timer, men det faktiske antallet ble på 556 timer (369 timer mindre enn forventet). For «Research» ble det anslått 125 timer, men som følge av et kontinuerlig behov for research ved enhver feilmelding, ble antallet på 375 timer til slutt. For «Design» ble det anslått 155 timer, men det endelige tallet ble på 71 timer da vi valgte å fokusere mer på programmering og hovedrapport. For «Testing» ble det anslått 75 timer, men det ble benyttet fire timer mindre enn forventet. For «Hovedrapport» ble det anslått 170 timer, men ettersom vi har inkludert opprettelsen av syv vedlegg, endte vi opp på 411 timer.

Planlagt start- og sluttdato for alle kategorier (med unntak av hovedrapporten) stemmer fortrinnsvis med faktisk start- og sluttdato. Ellers er det tydelig at det er blitt utført en feilestimereing på cirka en uke, for omtrentlig start- og sluttdato.

Kategori	Anslått start og slitt	Faktisk start og slutt	Anslått antall timer	Faktisk antall timer
Møter	22.01 – 15.05	22.01 – 19.05	55	78
Programmering	24.02 – 14.05	02.03 -19.05	925	556
Research	24.01 – 20.02	20.01 -19.05	125	375
Design	26.02 – 14.05	04.03 -15.05	155	98
Testing	02.03 – 15.05	16.03 – 19.05	75	71
Hovedrapport	06.04 – 19.05	21.01 – 19.05	170	411

Tabell 4.3-1 Oversikt over faktisk tidsforbruk og anslått tidsforbruk

5 Diskusjon

5.1 Vitenskapelig resultater

Følgende delkapittel tar for seg en vurdering av de presenterte vitenskapelige resultatene fra kapittel 4.1. Vurderingen utføres i sammenheng med problemstilling og forskingsspørsmål uttalt i kapittel 1.3

5.1.1 Problemstilling

Problemstillingen stiller spørsmålet rundt det å utvikle et brukervennlig spill med fokus på samarbeid, ved bruk av forskjellige spilldesignteknikker og en brukerorientert utviklingsprosess. Utfordringene som tas opp her, omhandler utformingen av funksjonalitet som tilfredsstiller behovene for en spesifikk bruker, og som samtidig tilfredsstiller behovene til brukergruppen som helhet. Resultatene som foreligger i kapittel 4 har verifisert om teknikker og tiltak benyttet har ledet fram til et ønskelig produkt.
Resultatene som nevnes i kapittel 4.2.3 (funksjonelle egenskaper) er tilknyttet våre tidligere formuleringer av spillkonseptet og nødvendig funksjonalitet tilhørende spillet. Vi har gitt utrykk for at design tilhørende spillet har benyttet elementer fra interaksjonsdesign og universell utforming. Samtidig blir det vist hvordan objekter og funksjonalitet gjenspeiler det ønskede spillkonseptet.

Ved å modifisere design og funksjonalitet etter mottatte tilbakemeldinger og observasjoner fra første godkjenningstest, har vi opprettet et system som er ansett som mer brukervennlig for testpersonene. Ved å gjenta godkjenningstest-prosessen og modifisert etter behov, har vi oppnådd et produkt som legger grunnlag for videre evaluering.
Resultatene i evalueringen benyttes hovedsakelig til å forbedre spillet ytterligere.
Når det gjelder samarbeid, var fokus rettet på formulering av oppgaver som oppfordret til samarbeid innad spillet. Resultatet viser hvilke teknikker som er blitt benyttet til å fremme samarbeid mellom spillerne. I dette tilfellet har oppgaver utdelt i forbindelse med brukertesting bekreftet/avkreftet om de fremmer samarbeid mellom spillerne.

5.1.2 Forskningsspørsmål

I tillegg til å ha en problemstilling, har vi også ønsket å utforske tre forskningsspørsmål gjennom denne rapporten. Det første spørsmålet, “Kan spillet lages responsivt, både på kontrollere og på hovedskjerm?”. Spillets og kontrollerens responsivitet er blitt bevist ved bruk av skjermbilder med forskjellige skjermstørrelser i kapittel 4. Følgende har også blitt besvart gjennom bruken av responsive CSS-klaser.
Det andre forskingsspørsmålet var “Hvordan kan brukerinvolvering bidra til å oppnå et mer brukervennlig spill?”. Følgende forskingsspørsmål er blitt besvart i kapittel 4.2.1, hvor det tas hensyn til brukerens tilbakemeldinger og kommentarer til å forbedre systemet ytterligere. Det er ikke direkte besvart, men det er blitt tydeliggjort at brukeren har en direkte påvirkning på spillet, da endringer implementeres som følge av brukerens tilbakemeldinger.
Det siste forskningsspørsmålet var “Hvilke teknikker kan bidra til å fremme samarbeid mellom spillerne”. Her vises det til bruken av oppskriftstavle og transportbånd, som kan være nyttige elementer for å vekke konkurranseinstinktet. For å vinne konkurransen må laget sanke inn så mye poeng som mulig. Et lag som vinner sammen, styrker tilhørigheten innad gruppen og fremmer samarbeidsevner, da en oppnår en belønningsfølelse av å vinne. Det er også blitt vist til begrensninger som nærmest “tvinger” brukerne til å jobbe sammen. Utenom dette, er det ikke referert til noen tydelige bestemte eller fysiske teknikker.

5.2 Ingeniørfaglige resultater

5.2.1 Ikke Funksjonelle egenskaper

5.2.1.1 Utviklingsmiljø og vedlikehold

«Selve produktet skal utvikles med C#, samtidig som kontrolleren utvikles med HTML, CSS og Javascript» (Visjonsdokumentet, kap. 6)

Følgende står skrevet i visjonsdokumentet, og omhandler utviklingsmiljøet og vedlikehold av produktet. Ved å referere til kapittel 4.2.2.1, kan vi konkludere med at følgende ikke-funksjonelle krav har blitt oppfylt. Spillet og kontrolleren er blitt utviklet ved bruk av C#, HTML, CSS og JavaScript. Utover det er det ingen andre krav som spesifiserer bruken av andre utviklingsprogrammer. Produktet er blitt opprettet i en gratis, ulisensiert programvare som Visual Studio 2017. Enkelt vedlikehold er blitt sikret ved å benytte løst koblede komponenter, gjennom bruk av MVC og ulike designmønster nevnt i kapittel 2.6 og 3.3.

Det spesifiseres også at produktet ble anbefalt å utvikles ved bruk av en smidig metodikk, som for eksempel- Kanban eller Scrum. Resultatene i kapittel 4.3.1 viser til dokumentasjon på at dette er blitt utført. Bruken av Scrum har blitt integrert med elementer fra interaksjonsdesignprinsippet for å oppnå en fleksibel og brukersentrert utvikling. Følgende utviklings-metodikk sikrer enkelt vedlikehold og enkel omprioritering av arbeid underveis. På denne måten har vi oppnådd å sette fokus på verdier i det Agile Scrum Manifestet. Det Agile Scrum Manifestet har blant annet blitt gjenspeilet ved at vi har iverksatt forsvarlige endringer og omprioriteringer som følge av tilbakemeldinger fra ulike aktører. Disse endringene diskuteres grundigere i kapittel 5.2.3.

5.2.1.2 Platform og operativsystem

«Spillet skal tilpasses de vanligste nettleserne» (Visjonsdokumentet, kap. 6)

I visjonsdokumentet spesifiseres det at spillet skal kunne kjøres på de vanligste nettleserne. De vanligste nettleserne i per dags dato er: Firefox, Google Chrome og Safari [23].

Ved å henvise til resultatet i kapittel 4.2.2.2, konkluderes det med at følgende krav har blitt oppfylt.

5.2.1.3 Andre krav

Som øvrige krav spesifiserer visjonsdokumentet om tilgjengelighet innenlands og utenlands. Ettersom at spillet befinner seg på plattformen Tiltspot, og spillet er tilgjengelig på plattformen som benyttes av brukere både innenlands og utenlands, er følgende krav allerede opprettholdt.

Bruken av animasjon for opplæring på under ett minutt har også blitt sikret. Animasjonen vises automatisk før selve spillet starter (se kapittel 4.2.3.6).

5.2.2 Funksjonelle egenskaper

Spillets hoved funksjonalitet har ivaretatt de planlagte funksjonelle egenskapene og kravene i stor grad, oppført i visjonsdokumentet (Vedlegg B, kapittel 5) og kravdokumentasjonen (Vedlegg C). Ettersom spillet er under vårt eierskap, har vi anledning til å utsette lavprioritert arbeid. I dette tilfellet omfatter lavprioritert arbeid, arbeid for destruksjon av råvarer og ingredienser, finpuss av objekters posisjonering og arbeid tilknyttet musikk og lydeffekter.

Kravet som omhandler tilbakemelding i form av visuelle og auditive forandringer, ble delvis oppfylt. Funksjonaliteten "highlighter" er blitt implementert i alle objekter. Dersom spillkarakteren kolliderer med et objekt, vil objektet lyses opp for å visualisere at objektet er i fokus. De auditive forandringene er delvis implementert, da det gjenstår å finne passende lyd-innslag for kutting og steking. Da lydeffekter er blitt kategorisert som lavprioritert arbeid, har vi mulighet til å jobbe med dette i fremtiden

5.2.2.1 Oppstart av spill og team-seleksjon

Oppstart av spill utføres ved å benytte Tiltspot.tv. Ved å skrive inn koden som er oppgitt på Tiltspot.tv på mobilenheter, får man en eksklusiv tilgang til alle tilgjengelige spill på nettsiden. Produktet vårt befinner seg på denne nettsiden og kjøres ved trykkregistrering av knappen "start spill". Spillerne presenteres deretter for en team-seleksjons scene.

De funksjonelle egenskapene "start et nytt spill" og "velg et lag" (Vedlegg B, kapittel 5.1 og 5.2) er blitt oppfylt og opprettholdt. Kravet om fungerende flerspiller-funksjonalitet er blitt oppfylt ved bruk av Tiltspots API manager og vår egen Game Manger for håndtering av spillhandlinger tilhørende spillerens unike kontroller-ID.

5.2.2.2 Utforming av oppskriftstavle og transportbånd

Kravet om oppskriftstavle som viser aktive oppskrifter og random-genererte ingredienser som fraktes ut på banen, ble begge oppfylt. Disse kravene var viktige krav, da de er høyt prioriterte og nødvendige for å kunne fullføre gameplayet.

5.2.2.3 Fårikål og oppskrifsbeholder

Kravet om et objekt som registrerer om den ferdigstilte matretten befinner seg i lista over aktive oppskrift er blitt oppfylt i samarbeid mellom oppskriftbeholder, kjøkkenbenk (Kap.

5.2.3.5) og oppskriftstavle. Kjøkkenbenken sender oppskriftsbeholderen videre til oppskriftstavlen som bekrefter/avkrefter dens tilhørighet til aktiv oppskriftliste.

5.2.2.4 Matbeholder

Kravet om kjøkkenredskaper som stekepanne og skjærebrett for å kutte opp diverse grønnsaker og steke/koke resterende ingredienser er blitt oppfylt. Følgende krav er blitt oppfylt ved å tilby stekepanner som må plasseres på tilhørende stekeovn for å aktivere stekingen/kokingen. Skjærebrettet er midlertidig blitt implementert som en kjøkkenbenk, da den arver egenskaper fra klassen kjøkkenbenk av praktiske grunner.

5.2.2.5 Oppset av kjøkkenbenker

Som nevnt ovenfor er kravet om et objekt som registrerer om den ferdigstilte matretten befinner seg i lista over aktive oppskrift, oppfylt. Kjøkkenbenken som er ansvarlig for innlevering av ferdigstilte matretter tar hånd om beregning av poeng, og kommuniserer med et gameplay-objekt som sørger for å videresende resultatet til Tiltspots API manager, som annonserer poeng og vinner.

Kravet om en søppelkasse som kvitter seg med ingredienser, slik at de ikke hoper seg opp på spilleren bane er blitt oppfylt. Dette kravet er blitt ansett som viktig og nødvendig. Dersom det ikke er nok benkplass for plassering av matvarer, er det nødvendig å kvitte seg med matvarene for å frigjøre plass. Foreløpig er det ikke mulig å kvitte seg med kjøkkenredskaper, dette er gjort med tanke på å utfordre spillerne.

5.2.2.6 Karakter og bevegelse

Kravene om horisontal bevegelse, vertikal bevegelse og rotasjon ble alle oppfylt. Følgende krav var nødvendige for å sikre en akseptabel og virkelighetsnær forflytning av spillkarakterer. Kravet om å plukke opp redskaper og å sette fra seg redskaper, ble begge oppfylt. Som følge av innføringen av begrensninger for å forhindre feil, har spillerne ikke mulighet til å plassere gjenstander direkte på bakken etter å ha plukket de opp. Kjøkkenredskaper og matvarer kan kun plasseres på en kjøkkenbenk eller i en matbeholder. Vi har valgt å innføre begrensingen ovenfor for å forhindre at bakken overfylles av gjenstander som belaster den naturlige bevegelsesflyten.

5.2.2.7 Kontrolleroppsett

Kravet om responsiv design og støtte for flere nettlesere, ble begge oppfylt. Disse kravene er blitt ansett som nødvendige og kritiske, da et design som ikke er responsivt kan frata brukeren muligheten til å utføre kommandoer som de andre brukerne kan utføre. Dette er

blitt oppfylt ved å benytte ulike CSS-skjemaer. Responsiv design er midlertidig ikke blitt testet på eldre mobilenheter.

Vi har valgt å nedprioritere følgende krav: endring av karakterens antrekk og farge, meny som kan nås i kontrolleren og mulighet til å endre på kontrolloppsettet. Kontrolleroppsettet slik den er utformet, fungerer på de fleste enheter. Derfor ser vi ikke det på som nødvendig arbeid for å kunne spille spillet vårt. Slike funksjoner ser vi på som spesielle funksjoner som kan tas med i videre arbeid. Funksjonene kan innføres stegvis, uten at det skaper problemer for spillet.

5.2.3 Evaluering

I dette kapitlet ser vi på betydningen av resultatene knyttet til brukertesting i forrige kapittel i lys av følgende forskningsspørsmål definert i kapittel 1: *Hvordan kan brukerinvolvering bidra til å en bedre brukervennlighet?*

Kontrollert brukertesting

Som følge av situasjonen vi befant oss da prosjektet pågikk har ikke brukertestingen kunnet gjennomføres helt som planlagt. Vi har måttet begrense antall deltakere i brukertestene vi har gjennomført, og vi har også hatt et mindre utvalg i hvilke deltakere vi kunne ha med. Dette har ført til at resultatene fra brukertestingen er noe mangelfulle. Målgruppen i forbindelse med prosjektet er svært bred, og det er ønskelig at resultatene fra brukertestingen representerer denne på en god måte. Det at brukertestene har hatt færre deltakere gjør det vanskeligere å vurdere om resultatene gjør nettopp dette. Når det er sagt valgte vi personer med ulik spillerfaring og fra forskjellige aldersgrupper, for å representere målgruppen så godt som mulig gitt situasjonen.

Færre deltakere har også hatt konsekvenser for observasjonene vi fikk gjort rundt samarbeidet i spillet. Vi ønsket i utgangspunktet å kjøre flere brukertester med mange deltakere slik at vi fikk et større observasjonsgrunnlag på samarbeidet i spillet. Dette fikk vi ikke gjennomført, og de innhentede dataene angående samarbeid er dermed noe mangelfulle. Om resultatet fra denne brukertesten er tilstrekkelig grunnlag til å konkludere med at spillet oppfordrer til samarbeid eller ikke, er vanskelig å si noe om. Når det er sagt viste resultatene fra testen at deltakerne faktisk samarbeidet om oppgavene som ble gitt. Til tross for utfordringene knyttet til antall deltakere, har brukertestene gitt innblikk i en tilnærmet reell brukersituasjon av spillet. De øvrige planene knyttet til gjennomføringen av

testene har gått som planlagt og utføringen har ellers gått på skinner. Dette skyldes trolig et grundig forarbeid og planlegging fra gruppens side.
Feilene som ble avdekket i vurderingstestene som ble utført underveis i prosessen ble utbedret i etterkant av hver test. Det ble også lagt stor vekt på de øvrige tilbakemeldingene i utviklingen videre. Resultatene fra godkjenningstesten indikerte at feilene avdekket i tidligere tester var utbedret. Dette tyder på at brukertestene har bidratt til å gjøre spillet bedre. Totalt sett har innspillene vi har fått og observasjonene som er gjort i alle testene bidratt med et nytt perspektiv og muligheten til å gjøre forbedringer på spillet. På denne måten har resultatene fra testingen bidratt til å bedre brukervennligheten.

5.2.4 Problemer underveis

I utviklingsfasen oppsto det problemer forårsaket av eksterne faktorer, som forårsaket en negativ påvirkning på utviklingsprosess og resultat.
I startfasen av utviklingsprosessen ble vi tildelt en veileder som ikke svarte på mail, og som ikke så over tilsendt arbeid. Dermed kunne vi ikke justere og tilpasse arbeid i henhold til tilbakemeldinger, da vi ikke mottok tilbakemeldinger på dokumenter som visjonsdokument og møteinnkalling. Dette medførte til at vi startet utviklingen direkte uten å ha mottatt en validering av dokumentet vårt.
Problemene ble forholdvis løst etter å ha blitt tildelt en ny veileder (nåværende veileder), som bidro med rask respons, tilbakemeldinger og veiledningssamtaler.
Utviklingsprosessen og resultatet er blitt preget av COVID-19, spesielt i tilknytning til deltakelse på Norwegian Game Awards (NGA). NGA er Norges største spillutviklingskonkurranse, hvor vi hadde mulighet til å teste spillet til et mangfold av brukere, ved å observere og stille spørsmål underveis. På denne måten kunne vi ha oppnådd et stort datasett, som vi kunne ha benyttet til forbedringer av produktet. Gruppetesting har også blitt vanskelig å gjennomføre, og vi ble hindret fra å teste om systemet kunne støtte flere spillere om gangen.
Et annet problem oppstod i forbindelse med testing av pause-funksjonen i spillet. Grunnet en feil med Tiltspot-plugin, mottok ikke spillet meldinger fra kontrollen når spillet var i pausemodus. Dermed gikk det ikke å starte spillet igjen. Dette er en feil fra Tiltspot sin side, som de skal fikse i etterkant av prosjektet. Som følge av denne feilen fikk vi ikke testet denne funksjonen i våre brukertester.

5.2.5 Svakhetet ved produktet

Grafisk design som inkluderer modeller av objekter eller UI-element, er uten tvil det viktigste aspektet av et spill. Følgende elementer kan i noen grad påvirke brukeropplevelsen. Som dataingeniører hadde vi begrenset med ferdigheter når det kom til opprettelse av design og modeller. Det resulterte i at en del grafiske designelementer ikke nådde standarden vi ønsket. Derfor var hovedfokuset vårt å arbeide med implementasjon av funksjonalitet. For at å gjøre spillet mer unikt, benytte vi mye tid på å designe og modellere en unik hovedkarakter. Oppsummert kunne vi ha oppgradert brukeropplevelsen, dersom vi var i besittelse av bedre designferdigheter.

5.2.6 Styrket ved produktet

Spillet skal publiseres på Tiltspots plattform, som er tilgjengelig for alle brukere med tilgang til internett. For brukere som ikke spiller jevnlig, kan Tiltspot være en tiltrekkende alternativ, da spillerne kan enkelt delta i spillet ved å benytte en mobilenhet og pc/tv skjerm. De fleste er i besittelse av en mobilenhet og skjerm, dermed er det lite eller ingen økonomiske kostnader ved å spille vårt spill. At produktet kan benyttes av hvem som helst, hvor som helst er et tiltrekkende tilbud. Spillet skiller seg fra de andre spillene i Tiltspot-plattformen, da den er team-basert og konkurransebasert. Spillet har også tatt utgangspunkt i den norske kulturen, slike spill finnes det lite av på nettet, dermed vil spillet bidra til tiltrekking av norske spillere.

5.3 Administrative resultater

5.3.1 Utviklingsprosess

5.3.1.1 User Stories

User Stories tar utgangspunkt i brukerens perspektiv og har bidratt til å tydeliggjøre implementasjonen av spillet, med hensyn på brukernes krav.

5.3.1.2 Sprints

Utviklingsmetoden Scrum har ført til at vi har oppnådd et forventet resultat innen en begrenset tidsperiode. Det viktigste er at det har integrert godt med interaksjonsdesign, slik at vi har oppretthold fokuset på brukeren.

De forskjellige elementene i Scrum (se kapittel 3.2.1) har brakt følgende fordeler under denne utviklingsprosessen:

- En Sprint-varighet på to uker har hjulpet oss med å utvikle en demoklar og verdifull versjon av spillet, ved slutten av hver Sprint. Dette forsikret oss om at vi regelmessig hadde et stabilt system tilgjengelig.
- Daglige digitale Scrum-møter har hjulpet oss med å oppnå innsikt og forståelse for hverandres oppgaver, status og fremtidige hensikter. Slik ble misforståelser som følge av skriftlig kommunikasjon minimalisert.
- Sprint Review har bidratt til innsamling av tilbakemeldinger på det eksisterende systemet, og oppdagelse av uforventede og usynlige problemer. I tillegg har det gitt oss anledning til å avklare eventuelle uklarheter.
- Sprint Retrospektive har gitt oss mulighet til å diskutere eksisterende problemer, slik at vi kan forberede oss til neste sprint.

5.3.2 Timeregnskap

Et avvik mellom antall planlagte timer og gjennomførte timer var forventet. Ettersom det ble benyttet en iterativ metode, ble korrigering og vedlikeholdning utført med lite problematikk.

5.3.3 Måloppnåelse i forhold til framdriftsplan

Resultatene viser at tidsrommet mellom startdato og sluttdato for de ulike kategoriene ble lengre enn forventet. Det er tydelig at vi har gjort en del feilestimeringer. Faktisk antall timer ble mye mindre for mange kategorier. Kategoriene «Programmering», «Design» og «Hovedrapport» er blitt overvurdert, mens kategoriene «Research» og «Møter» er blitt undervurdert.

Mye av årsaken til de ovennevnte feilene ligger i bruken av Scrum. Den iterative utviklingsprosessen har gitt oss frihet til å justere og tilpasse etter tilbakemeldinger, og har på den måten tilrettelagt for gjentatte forbedringer og utrulling av nye versjoner.

5.3.4 Profesjonsetiske problemstillinger

Den opprinnelige intensjon med utviklingen av spillet Chef Poteto er å skape lykke hos brukerne. Det er unektelig at det finnes mange uetiske utviklingsatferder innen spillutviklingsverden. Noen spill legger til mange ulovlige elementer for å tiltrekke flere spillere, for eksempel bruken av ulovlige pornografiske bilder, noe som har villedet mange tenåringer. Noen spill legger til kjøpebare fordeler, slik at spillere får mulighet til å betale en sum av penger for å oppleve en lykkerus. Dette vil føre til uønsket konkurranse i spillet, men også være misledende for barn. Barn uten økonomisk forståelse oppfordres indirekte til å

misbruke en stor mengde penger. Spillet vårt tar avstand fra bruken av elementer nevnt ovenfor.

Det finnes mange spillutviklere som tar sikte på å gjøre spillere avhengige av spill, slik at selskapene deres kan skaffe seg mye penger og omdømme. Følgende er faktisk i strid med profesjonsetikken. Men paradokset er at når du designer et spill som gjør spillere avhengig, vil du bli definert som en vellykket spillutvikler. Spillselskaper bør ta ansvar for å minne spillerne om å ta seg en hvilepause. Vi kan ikke garantere at alle spillere unngår å bli avhengig av spillet vårt, men foreløpig kan spillsystemet vårt kun tilby spillerne en eller to runder med lykkerus.
Oppsummert har vi i stor grad fulgt de profesjonsetiske retningslinjene.

5.3.5 Gruppearbeidet

Gruppen har arbeidet jevnt og effektivt. Det er ikke blitt oppført fravær som følge av sykdom, arbeidsfrie dager i eksamensuker ble tatt igjen i helgene og fridager. Alle har vært til stede under møter med veileder og oppgavestiller. Arbeidsmengden per medlem ble jevnt fordelt, noe som gjenspeiles i timelistene (Vedlegg E, kapittel 4).
Ulike medlemmer har blitt tildelt sitt eget ansvars- område/områder og har jobbet nøye med dem. Oppdeling i ulike ansvarsområder hindret ikke arbeid på tvers av arbeidsområdene. Alle team-medlemmer bidro til å løse problemer som skulle oppstå i prosjektet, selv om det var utenfor deres eget ansvarsområde. Målet med oppdelingen var å sikre at et enkeltindivid hadde en robust oversikt over kvalitet og progresjon innenfor området.
Gjennom utviklingsprosessen har prosjektstyringsverktøyet Trello blitt benyttet for å holde en oversikt over de ulike gruppemedlemmenes oppgaver og progresjon. Dette har sikret en dynamisk og parallell utvikling.

5.3.6 Samarbeid

I kapittel 3 beskrives en rekke tiltak som er gjort med formål om å fremme samarbeid mellom spillerne. Teknikkene som er tatt i bruk er utformet på bakgrunn av teorien beskrevet i kapittel 2. I dette kapitlet ser vi på betydningen av disse valgene i lys av følgende

forskningsspørsmål definert i kapittel 1: *Hvilke teknikker og tiltak kan bidra til å fremme samarbeid mellom spillerne?*

Teknikkene som er tatt i bruk i dette prosjektet er basert på etablerte anbefalinger innen spillutviklermiljøet. Disse teknikkene er regnet å være en del av dagens gjeldende praksis når det kommer til utvikling av lokale samarbeidsspill. Selv om teknikkene er utviklet av erfarne aktører innen bransjen, er det gjort lite forskning på feltet. Anbefalingene er basert på erfaringer og personlige preferanser blant de ulike aktørene og effekten av disse er ikke blitt bevist gjennom vitenskapelige forskningsstudier. Samtidig kan man argumentere for at en person med lang erfaring innen feltet har en viss peiling på hva som fungerer og ikke.

I evalueringen av resultatet knyttet til bruken av slike teknikker og tiltak har brukertesting vært svært sentralt i vårt arbeid. Som nevnt i kapittel 5.2.4.1 om kontrollert brukertesting har situasjonen i forbindelse med Covid-19 påvirket dette resultatet. Mindre testing har ført til at vi har et mindre grunnlag å basere konklusjonen på. Når det er sagt tydet observasjonene som ble gjort under brukertesten på at spillet oppfordret til samarbeid. Begge lag som deltok i testen samarbeidet om oppgavene som ble gitt.

5.3.7 Feilkilder

I ulike deler av forskningsprosessen kan det noen ganger oppstå systematiske skjevheter. Disse skjevhetene kan føre til at resultatet av forskningen dras i en bestemt retning slik at resultatet blir misvisende. Forventningseffekter kan oppstå når forskerne på forhånd har en mening om hvordan resultatet vil se ut. I dette utviklingsprosjektet kan dette for eksempel ha skjedd under brukertesting. Dersom testansvarlig har spesifikke forventinger til utfallet av brukertesten, kan dette forme resultatene ved at vedkommende ubevisst tyder signaler og utsagn fra deltakerne på en slik måte at de samsvarer med denne forventningen. For å unngå slike feil kan man benytte en upartisk person til å lede brukertester. I dette prosjektet har vi verken hatt mulighet eller tid til å gjøre dette. Derfor kan vi ikke se bort ifra at slike feil har påvirket vårt resultat. Når det er sagt har gruppen vært klar over denne risikoen på forhånd og bevisst gjort en innsats for å være så upartisk som mulig under brukertestene.

Forventinger til resultatet av deltakerne kan også påvirke en brukertest. Dersom noen av deltakerne i testen har en formening om hvordan spillet er på forhånd kan dette påvirke vedkommende meninger. Eksempelvis dersom en deltaker har positive holdninger til spillet på forhånd, kan det hende at personen ubevisst bagatelliserer feil eller mangler ved spillet.

Dette gjelder spesielt dersom deltakerne av brukertesten har et forhold til testansvarlig. I dette prosjektet har venner og familie blitt benyttet i brukertestingen. Derfor kan vi ikke se bort ifra at noen av deltakerne har avgitt upartiske svar i testene. På en annen side kan denne effekten virke begge veier. Dersom en deltaker har negative holdninger til spillet på forhånd, kan det bidra til å forsterke inntrykket av feil og mangler.

6 Konklusjon og videre arbeid

6.1 Konklusjon

Denne rapporten har tatt utgangspunkt i følgende problemstilling:

Hvordan benytte spilldesign til å skape et brukervennlig spill som oppfordrer til samarbeid mellom spillerne?

I tillegg til problemstillingen, har rapporten diskutert og utforsket følgende forskningsspørsmål:

1. *Kan spillet lages responsivt, både på kontrollere og på hovedskjerm, slik at brukergrensesnittet tilpasses flere skjermstørrelser?*
2. *Hvordan kan brukerinvolvering bidra til å oppnå et mer brukervennlig spill?*
3. *Hvilke teknikker og tiltak kan bidra til å fremme samarbeid mellom spillerne?*

Svarene på både problemstilling og forskningsspørsmål begrunnes i diskusjonen fra kapittel 5. Fra diskusjonen om de vitenskapelige resultatene oppført i kapittel 5.1, avleder vi at vår brukerorienterte utviklingsprosess har frembragt en bedre brukeropplevelse. Et engasjerende spillkonsept og et oversiktlig brukergrensesnitt har oppfordret spillerne til å utføre gjøremål gjennom samarbeid. Det var ikke planlagt at en innføring av karantene skulle oppstå i løpet av arbeidet med bachelorprosjektet, og vi kan derfor ikke konkludere med at produktet vårt har nådd maksimal allsidighet og brukervennlighet. Til tross for dette, vil vi påstå at vi har etablert et spillkonsept og brukergrensesnitt som fremmer samarbeid.

De to første forskningsspørsmålene angår spesifikt brukervennligheten, og er derfor utfordrende å trekke konklusjoner for. Som nevnt i kapittel 5.2.4, har COVID-19 påvirket gjennomføringen av planlagte brukertester. Allikevel nøyer vi med å si at resultatene fra godkjenningstesten indikerer om at vi har oppnådd et mer brukervennlig spill. Eksisterende data indikerer at metodene og teknikkene våre har løst problemstillingen. Basert på

følgende, er vår vurdering at spillet overholder prinsippene om brukerinvolvering og universell utforming vellykket, men kun i en svært begrenset grad.

6.2 Videre Arbeid

I løpet av utviklingsprosessen har vi utviklet et spill med mange funksjoner. Siden dette er et tidsbegrenset prosjekt gjenstår noen funksjoner som vi opprinnelig ønsket å implementere, men dessverre ikke hadde tid til. Så langt har de grunnleggende funksjonene i spillet blitt implementert, med unntak av de som var kategorisert med lav prioritet i kravdokumentasjonen. For eksempel er det ikke blitt implementert en funksjon for å starte spillet på nytt, da denne kan ha innvirkning på rettferdigheten i spillet.

Med tanke på videre arbeid hadde det vært ønskelig at systemet kunne tilby spillerne større frihet til å uttrykke sin egen stil. Dette kan for eksempel gjøres ved at hver spiller får muligheten til å velge hvilke farger og/eller utseende deres karakter skal ha. En annen mulighet for videre utvikling er å implementere flere forskjellig baner med ulike tema, for å skape variasjon i spillet. Det er også stort potensiale for å lage flere og mer avanserte oppskrifter. Ved å benytte ulike baner og oppskrifter kan man også regulere vanskelighetsnivået i spillet og dermed tilby spillerne å velge mellom forskjellige vanskelighetsgrader. Dette vil gjøre at spillet blir mer allsidig og attraktivt for flere spillere. Spillet har også muligheten til å tilby støtte for et større antall spillere, men da er det nødvendig å sikre at systemet ikke blir tregt.

Utviklingsprosessen skal være brukerorientert, og påse at ingen funksjonalitet påvirkes brukervennligheten negativt. I tillegg ønsker vi å holde spillet oppdaterte mot nye versjoner av Unity- og Tiltspot APIet. Dette er essensielt for at utviklingsprosessen kan foregår jevnt i videre arbeid.

7 Referanser

[1] Store norske leksikon, “Dataspill.,” 2020. Tilgjengelig: https://snl.no/dataspill [hentet 16.04.2020]

[2] Wikipedia,“Multiplayer video games.,” 2020. Tilgjengelig: https://en.wikipedia.org/wiki/Multiplayer_video_game [hentet 23.04.2020]

[3] Tiltspot (2020). Tilgjengelig: https://dev.tiltspot.tv/ [hentet 18.04.2020]

[3] J. Sutherland og K. Schwaber, “The Scrum Guide.,” 2016. [Internett]. Tilgjengelig: https://scrumguides.org/docs/scrumguide/v2016/2016-Scrum-Guide-Norwegian.pdf [Funnet 7 april 2020].

[4] «Diagram of interaction within the MVC pattern., » 2020. [Internett]. Tilgjengelig: https://upload.wikimedia.org/wikipedia/commons/a/a0/MVC-Process.svg [Funnet 9 april 2020].

[5] S. Smith, «Overview of ASP.NET Core MVC., » 2020 [Internett]. Tilgjengelig: https://docs.microsoft.com/nb-no/aspnet/core/mvc/overview?view=aspnetcore-3.1 [Funnet 9 april 2020].

[6] w3schools, «C# Polymorphism.,» 2020. [Internett]. Tilgjengelig: https://www.w3schools.com/cs/cs_polymorphism.asp [Funnet 9 april 2020].

[7] tutorialsteacher, «Inversion of Control,.» 2020. [Internett]- Tilgjengelig: https://www.tutorialsteacher.com/ioc/inversion-of-control [Funnet 10 april 2020].

[8] «Dependency Injection.,» 2019. [Internett]. Tilgjengelig: https://blog.ona.io/assets/images/2019-04-23/koindi.png [Funnet 10 april 2020].

[9] tutorialsteacher, «Dependency Injection,.» 2020. [Internett]- Tilgjengelig: https://www.tutorialsteacher.com/ioc/dependency-injection [Funnet 10 april 2020].

[10] «Structure types (C# reference),.» 2020. [Internett]. Tilgjengelig: https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/builtin-types/struct [Funnet 10 april 2020].

[11] “Attributes (C#)” 2020. [Internett]. Tilgjengelig; https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/attributes/ [Funnet 10 april 2020]

[12] Y. Teo , «What is Interaction Design?.,» 2020[Internett] Tilgjengelig: https://www.interaction-design.org/literature/article/what-is-interaction-design [Funnet 11 april 2020]

[13] Unity Manual, «State Machine Basics., » 2020[Internett] Tilgjengelig: https://docs.unity3d.com/Manual/StateMachineBasics.html [Funnet 11 april 2020]

[14] A. Wrålsen, “Interaksjonsdesign som prosess 2.,” 2016 [Leksjon]

[15] Unity (2020) Particle Systems. Tilgengelig fra: https://docs.unity3d.com/Manual/ParticleSystems.html (Hentet: 11. April 2020).

[16] Crawford. Christ. «Chris Crawford on Game Design., » 2003. [Bok] [Funnet 12 april 2020].

[17] Pedersen R E.« Game design foundations» 2003. [Bok] [Funnet 12 april 2020].

[18] Wikipedia, “Teamwork.,” 2020. [internett] https://en.wikipedia.org/wiki/Teamwork [Hentet 06.05.2020]

[19] Wikipedia, “Cooperative gameplay”, 2020. [Internett] https://en.wikipedia.org/wiki/Cooperative_gameplay [hentet 06.05.2020]

[20] Alexander Jonassen, “Designing for couch co-op.," 2017. [internett] https://www.ntnu.edu/documents/139799/1279149990/17+Article+Final_alexajon_fors%C3%B8k_2017-12-08-00-02-45_TPD4505.Alexander.Jonassen.pdf/73ba010c-308e-4044-8059-7bc34f96de10 [hentet 07.05.2020]

[21] Wyszecki.Gunter and Walter Stanley Stiles. «Color science., » 1982. [Bok] [Funnet 12 april 2020].

[22] TechTerms, «User-Friendly.,» 2014 [Internett]. Tilgjengelig: https://techterms.com/definition/user-friendly [Funnet 9 mai 2020].

[23] J. Martin, “The best web browser for 2020.,” 2020[Internett]. Tilgjengelig: https://www.techadvisor.co.uk/test-centre/software/best-web-browsers-3635255/ [Funnet 10 mai 2020]

8 Vedlegg

Vedlegg A: Akronymer og ordforklaringer

Vedlegg B: Visjonsdokument

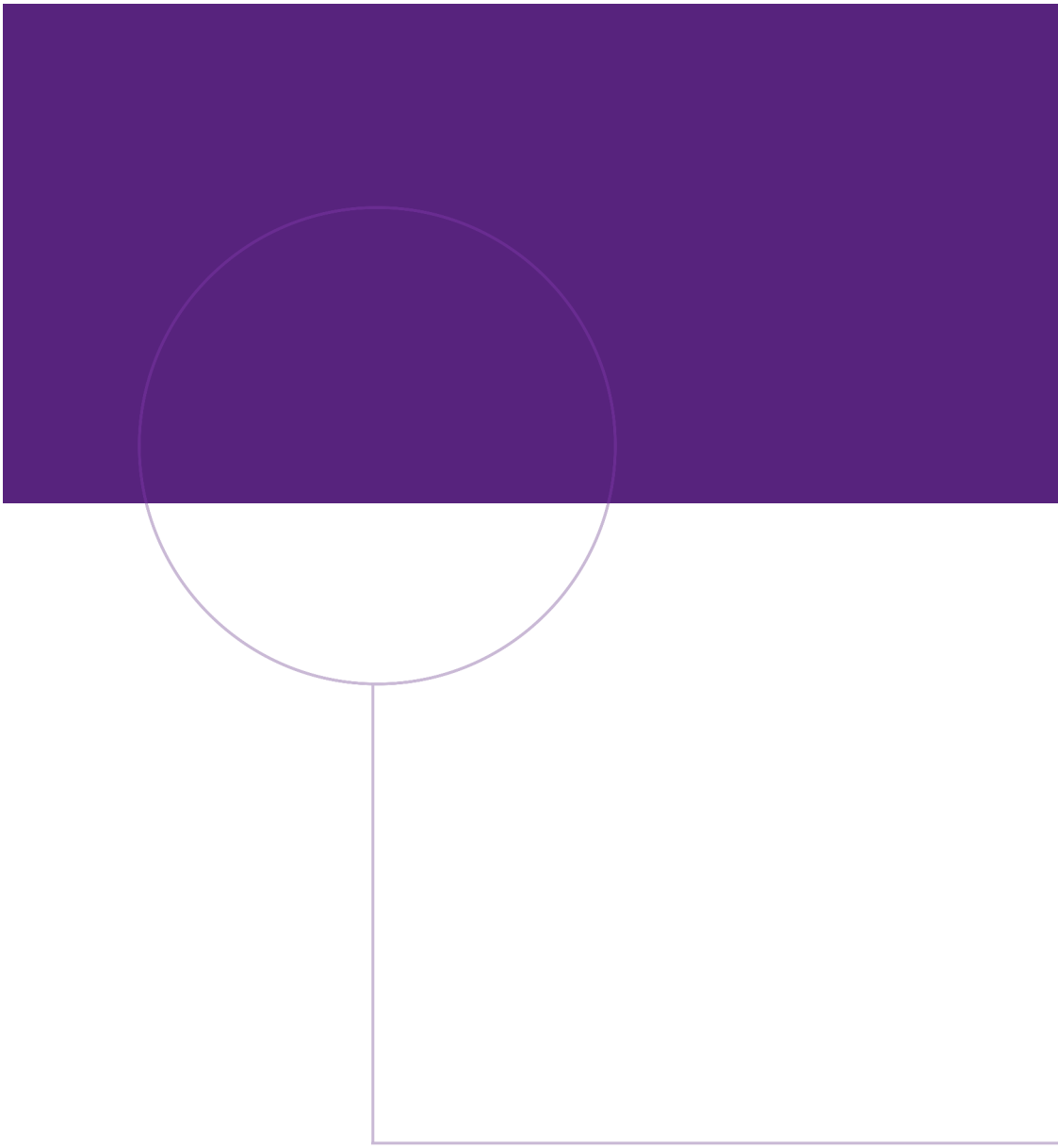
Vedlegg C: Kravdokumentasjon

Vedlegg D: Systemdokumentasjon

Vedlegg E: Prosjekthåndbok

Vedlegg F: Scrum – Sprint Review og Sprint Planning

Vedlegg G: Brukertestplan



NTNU
Kunnskap for en bedre verden