Jens Waller Aasgaard, Ida Emilie Aspehaug,
Håvard Ellila Homme & Bahar Yasamine Nagafi

Teknisk rapport

IN1060 - Bruksorientert design

Institutt for informatikk, Universitet i Oslo

Vår 2020

Innholdsfortegnelse

Mål

Khloron samarbeider med miljøorganisasjonen Sabima for å lage en løsning som gir organisasjonen oppmerksomhet samtidig som de får videreformidlet deres kunnskaper om gode miljøtiltak i folks hager. Sabimas mål er å få tilskuere ved hagemesser til å velge positive tiltak som gjør tilskuernes hager til sunne hager.

Løsningen vår består av et bord fylt med elementer som gir inntrykk for en normal hage. Bordet vil samle hagemessens tilskuere til Sabimas stand og engasjere ved å la dem flytte på blomsterpotter med ulike hagemotiv. Blomsterpottene flyttes til senteret av bordet som er omringet av grønne og røde LED. Basert på hvilke blomsterpotter tilskueren flytte, lyser LED en farge som gir tilskueren en indikasjon på hvorvidt blomsterpotten er et positivt eller negativt tiltak for å oppnå en sunn hage.

Prototypens Maskinvare			
Del av prototypen	**komponent**	**Antall**	**Hvordan?**
Deteksjon av ulike brikker	Tilt-switch	1	Tilt-switchen registrerer om en brikke er plassert på feltet.
	Fototransistor	1	Fototransistoren registrerer mengden lys som absorberes. Den ene brikken slipper inn store mengder lys, mens den andre brikken slipper inn nær inget lys.
Lys-feedback	3mm rød led	4	LED kobles i hver sin krets rundt fototransistoren og lyser opp for å gi visible feedback.
	3mm grønn led	4	
Motstand	220 Ω resistor	10	220 Ohm resistor til hver av ledlysene
	10K Ω resistor	2	10K Ohm til tilt-switch og fototransistor

Prototypens Programvare		
Del av prototypen	**Program**	**Hvordan?**
Arduino-kode	Arduino Genuino 1.8.11	Programkoden til prototypen er laget i text-editoren og Atom
Digital arduino	Tinkercad	Visualisere hvordan arduinobrettet er koblet opp

Kort om koronavirussituasjonen og arduinokomponenter

Situasjonen rundt koronaviruset gjorde at vi hverken fikk tilgang til skolens arduino-komponenter eller 3D-printing. Vi hadde også et ønske om å ikke bruke penger på flere komponenter utover komponentene inkludert i arduinos starter kit. I denne rapporten vil vi diskutere komponentene vi har valgt som følge av dette, men også nevne de forskjellige alternativene vi vurderte underveis eller som vi kunne ha valgt dersom situasjonen rundt koronaviruset var annerledes.

Introduksjon av prototypen

Interaksjonen mellom prototypen og brukeren skjer idet brukeren flytter en brikke formet som en blomsterpotte fra brettet og plasserer den på et designert felt. Idet brikken plasseres skal prototypen automatisk finne ut hvilken type brikke som er plassert, og gi et tilpasset feedback i form av enten rødt eller grønt lys. Det er totalt tre forskjellige statuser: Ingen brikke plassert, “positiv brikke” og “negativ brikke” plassert. Prototypens tekniske deler som helhet kan beskrives i to deler: Deteksjon av ulike brikker (input) og lys-feedback (output).

Deteksjon av ulike brikker

Vi har i løpet av prototypingen eksperimentert med forskjellige metoder for å registrere brikker basert på hvilke muligheter og begrensninger vi har møtt på underveis. Sentralt for alle fremgangsmåtene er at prototypen vår var bygd rundt en fototransistor med et tilhørende fysisk felt der brikkene skulle plasseres.

Deteksjonsmetode 1: Tilt-switch og fototransistor

Når brukeren flytter en brikke fra hagen og inn på feltet, registreres brikken av en tilt-switch. Under feltet er det plassert en fototransistor som registrerer mengden lys som absorberes. I vår prototype vil de negative brikkene slippe inn mye lys, mens de gode brikkene ikke slipper inn noe lys. Dermed vil fototransistorens verdier for mengden absorbert lys kunne fortelle oss hvilken brikke som er plassert på feltet.

Løsningen vi tenkte for å skille mellom de tre forskjellige statusene var å benytte oss av et binært tilfelle for fototransistoren, enten absolutt mørke eller lys, i kombinasjon med å først sjekke for fysisk tilstedeværelse av en brikke ved hjelp av en tilt-switch. Ved å plassere en tilt-switch under feltet hvor brikken legges ned ville man kunne ta i bruk fototransistoren idet tilt-switchen har detektert en brikke.

Problemet vi oppdaget ved bruk av tiltsensoren var at den måtte tiltes veldig mye før den ble aktivert. Det var nødvendig med en større grad av tilting enn det plasseringen av en brikke ville ha gitt utslag for. Vi gikk derfor videre med en knappebryter som erstatning for tilt-switchen i prototypingsfasen.

Deteksjonsmetode 2: Manuell knapp og fototransistor

Denne metoden tok utgangspunkt i programkoden skrevet for tilt-switchen, men istedenfor en tilt-switch brukte vi en arduino-knappebryter som også er binær og styres av digitale porter. Slik kunne vi teste prototypen mer effektivt i praksis med manuell "wizard of oz"-basert initialisering av interaksjonen idet brikkene ble satt ned på sensorområdet.

Deteksjonsmetode 3: Helautomatisk fototransistor

Denne metoden var det vi anså som ideelt: Det ville kunne driftes helautomatisk, men var også vanskeligst å utforme og programmere. For å få til en helautomatisk løsning måtte vi ha presise inndata fra fototransistoren og klare skiller i programmet som bestemmer hvilken status lysene skal ha, og når. Vi begynte å utforme en kalibreringsmetode for å tilpasse skilleverdiene, men oppdaget at det var vanskelig å treffe riktig på skillet mellom ikke belyst ("positiv brikke plassert"), delvis belyst ("negativ brikke plassert") og fullt belyst (ingen brikke plassert) - selv med kalibrering. Det ble marginalt lettere å få til gode skiller med kalibrering, spesielt i omgivelser med sterkt dagslys. Men til tross for tilpasning av grenseverdiene viste skygge og mørkere omgivelser seg å være en stor utfordring. Selve inndataene fra fototransistoren var en utfordring for prosjektet generelt, og det følger mer om dette lenger ned i rapporten.

Hvorfor vi har valgt fototransistor

Brikkene skal kunne plasseres ut hvor som helst på brettet og lagres en annen plass når brettet ikke er i bruk.
Vi tenkte først å bruke en fargesensor slik at den kunne gjenkjenne de ulike brikkene basert på brikkens farge. Men da ville det også vært en risiko for at brukeren hadde sett brikkens farge som igjen hadde påvirket brukerens opplevelse av løsningen. Derfor trengte vi en sensor som var usynlig for brukeren.
Så det mest optimale ville vært å bruke en RF-ID sensor, og plassert en RF-ID-brikke inne i hver av brikkene slik at de hadde fått hver sin unike ID. Men som vi sa i innledningen, så

ønsker vi kun å bruke komponenter fra «Arduino Starter Kit» og har dermed hverken tilgang på fargesensor eller på RF-ID.
For at brikkene skal ha den fleksibiliteten vi ønsker, kan ikke brikkene inneholde arduino-komponenter som krever oppkobling med kabler fra hver brikke til brettets arduino. Vi trenger da en sensor som gjenkjenner de ulike brikkene basert på naturlige egenskaper som f.eks farge, lys, vekt, størrelse.
Vi kom frem til at fototransistoren ble det beste alternativet fordi den kan detektere mengden lys fra omgivelsene og gjøre mengden om til en verdi. Ved å la hver brikke slippe inn lys gjennom ulik mengde lys ned til fototransistoren, vil hver av de brikkene få hver sin tilhørende verdi.

Lys-feedback

Idet brukeren legger ned en brikke får brukeren en feedback i form av blinkende LED-lys basert på hvilken brikke som er lagt ned. Vi har bestemt oss for at LED skal ha mulighet til å blinke i ulikt mønster ved ulike stadier av interaksjonen. Da vi skulle koble opp lyssystemet, måtte vi bestemme oss for om vi skulle koble opp LED i én krets, eller om hver LED skulle få hver sin egen krets.
Hvis vi hadde koblet alle LED i én krets, ville vi styrt antall lysende LED utfra mengden strøm som hadde fått passere gjennom kretsen. Problemet er at vi ikke hadde hatt muligheten til å justere hver enkelt LED uavhengig av de andre LED, noe som var et av kravene vi satt for å få til ulike lysmønstre.
Ved å koble opp en krets til hver enkelt LED kan vi styre strømmen som sendes ut i hver av kretsene og kan dermed blinke med LED uavhengig av hverandre.
Det negative med å gi hver LED sin egen krets, er at det vil kreves en egen digital port til hver LED, og vi risikerer dermed å gå tom for digitale porter som hadde trengt til andre sensorer.
Dette problemet løses om vi velger å koble alle LED i én krets da vi kun trenger 2 digitale porter, én port for grønne led og én port for røde led.
Etter at vi fikk utarbeidet en oversikt over hvilke komponenter vi ville bruke i vårt prosjekt, fant vi ut at vi hadde nok digitale porter til at hver led kunne få sin egen krets og samtidig få plass til de andre komponentene. Dermed koblet vi opp 4 kretser med grønne LED og 4 kretser med røde LED.

Fremgangsmåte for bygging av prototypen

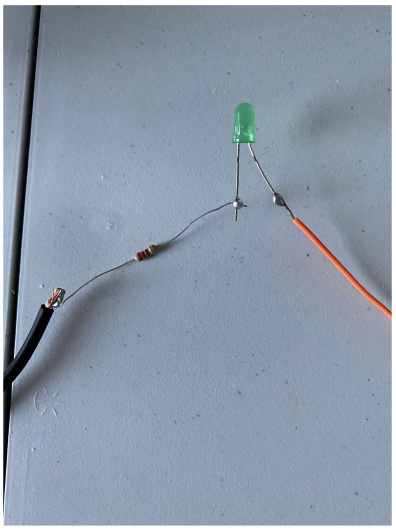
Vi loddet, isolerte og markerte alle ledninger fra komponenter til arduinobrettet. Ledningene som følger med i arduinoens starter kit er ganske korte. Derfor loddet vi på innkjøpt ledning hvor vi selv kunne tilpasse lengden slik at vi kunne plassere komponentene hvor vi ville på prototypen. Deretter koblet vi opp to input-sensorer, en fototransistor og en tilt-switch.
For å forhindre varierende forstyrrelser fra lyset i omgivelsene og for å få så forutsigbare verdier som mulig bygget vi opp vegger rundt fototransistoren slik at det blir tilnærmet

fullstendig mørkt når en brikke legges på.Vi bygget dette på en måte som gjorde at sensoren var beskyttet og ikke ville bli fysisk berørt, også når brikker ble lagt oppå.

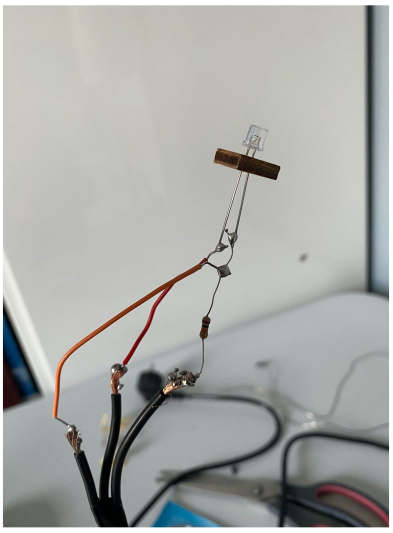
For å oppnå dette boret vi først hull i et bord der brikkene skulle plasseres. Vi festet fototransistoren på en overflate og limte en tom dopapirrull over fototransistoren for å utgjøre veggene. For å sørge for at brikker kunne plasseres over sensoren lagde vi et slags "spindelvev" av tynn hyssing. Vi vurderte også gjennomsiktige sirkelflater av enten glass eller plast, men vi bestemte oss for hyssingen da det fungerte godt og ikke hadde stor innvirkning på fototransistoren. Deretter boret vi 8 små hull i en sirkel rundt hullet til fototransistoren. Fra undersiden av bordet og opp gjennom de små hullene stakk vi LED annenhver farge rød og grønn.
Vi skjulte arduinoen i en pappeske festet på undersiden av bordet, og festet ledningene til hver LED til undersiden av bordplaten.

Figur 1

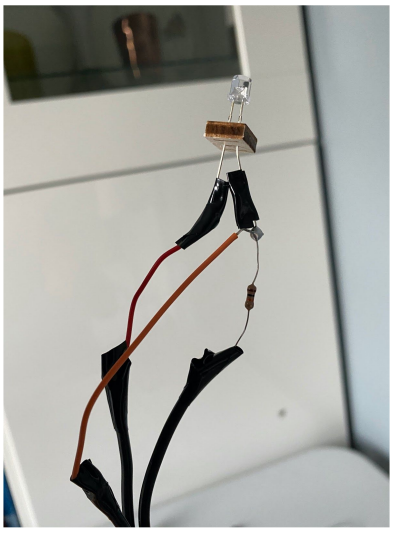
Figur 2



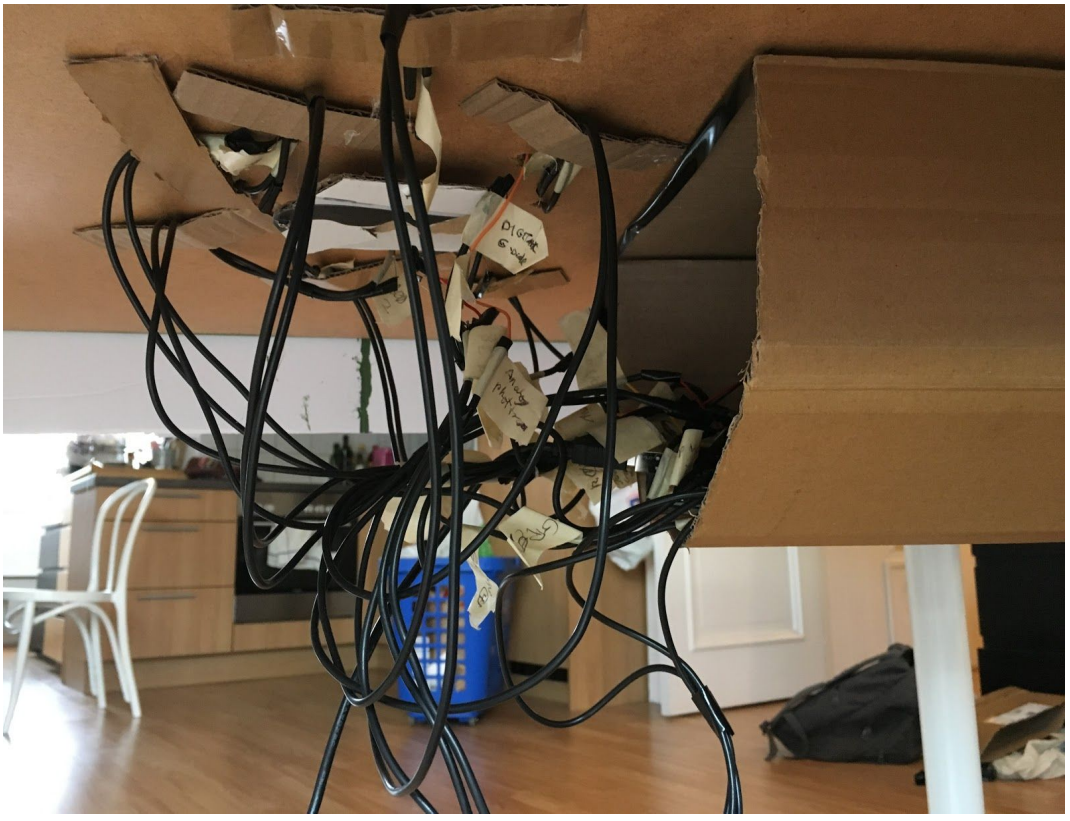
Figur 3



Figur 4



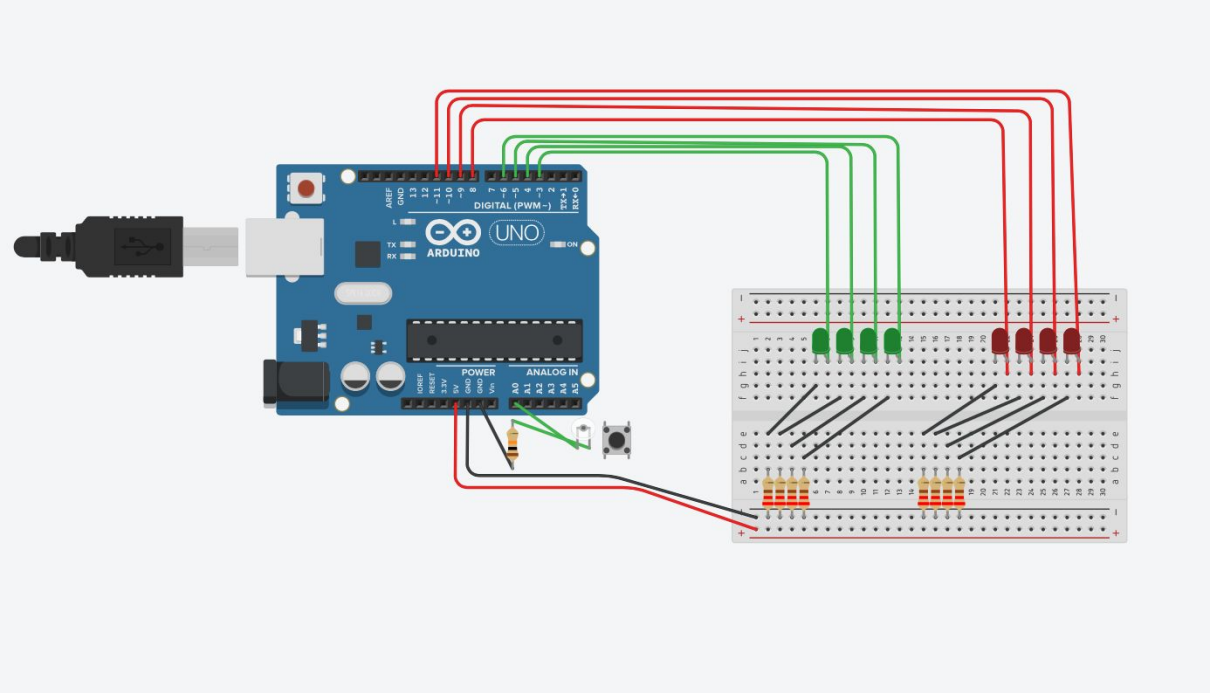
Figur 5



Figur 6

Figur 7

Figur 1: Hullet hvor tramnsistoren er plassert, “spindelsvevet” brikken skal stå på og LED som er plassert rundt i sirkel
Figur 2: Hullene sett i forhold til brettets størrelse før resten av hagen er utplassert
Figur 3: Loddet LED
Figur 4: Loddet fototransistor
Figur 5: Det er viktig å isolere kablene slik at ikke strømmen får ta snarveier
Figur 6: Vi koblet Arduino inne i en eske på undersiden av brettet. Ledningene ble festet med pappbiter til undersiden av bordet slik at ikke LED-pærene ikke skulle bli dratt ut av hullene sine.
Figur 7: Den endelige løsningen.

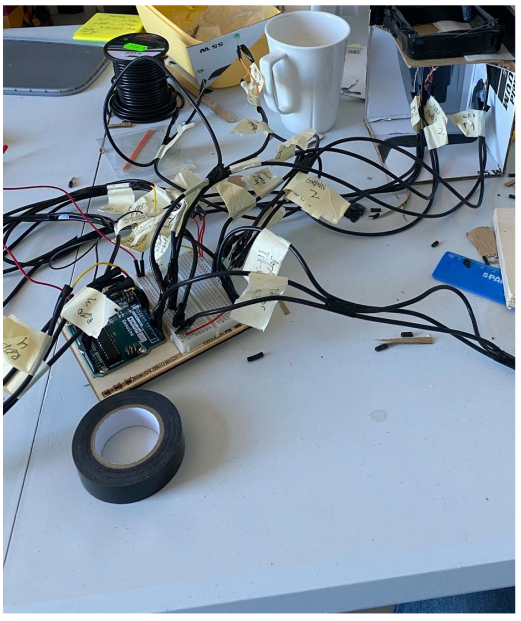


Ovenfor ser vi et bilde av arduino-oppkoblingen. Fototransistoren er utenfor bredboardet nedenfor de analoge portene. Her ser vi knappebryteren vi brukte i startfasen, men som vi ikke tok med i sluttløsningen. Vi har derfor ikke koblet den til kretsen i denne modellen.

Utfordringer og hvordan vi har forsøkt å løse dem

Vi hadde kun én type ledning som var tynn og lang nok for dette prosjektet noe som gjorde det vanskelig å skille de ulike ledningene i fra hverandre. Det løste vi ved å markere ledningen i begge ender med tape slik at vi raskt kunne se hvilket komponent og hvilken arduino port ledningen var koblet til.

Figur 6



Figur 7

Figur 6: Vi har markert alle kablene i begge ender
Figur 7: Kompleksiteten alle kablene utgjør viser hvorfor det er viktig å markere dem.

Et problem vi møtte under testing av fototransistoren var at fototransistoren er veldig sensitiv for variasjoner fra omgivelsene. Selv ved å kjøre programmet i kontrollerte omgivelser, fikk vi svært ulike verdier mellom hver kjøring. Vi forsøkte å detektere de ulike brikkene basert på hvor mye lys de slapp gjennom et halvt-gjennomsiktig farget papir, men verdiene til de ulike papirene ble så like at vi ikke fant noen store nok kontraster til å kunne skille brikkene fra hverandre. Vi endte dermed opp med kontrastene sort /hvitt, altså å slippe inn alt lys, kontra det å slippe inn inget lys.
Dette var en stor utfordring fordi det er tre forskjellige statuser brettet vårt kan være i: Ingen brikke, "negativ" brikke og "positiv" brikke. Disse tre nivåene avhenger av forskjellige grader av lys og forskjellige verdier på fototransistoren. Siden verdiene som kommer inn avhenger av lyset i omgivelsene var det vanskelig å finne riktige overgangsverdier for absolutt mørke, delvis mørke og naturlig lys.

Med den løsningen oppstod et nytt problem, nemlig at vi ikke kunne skille mellom en brikke som slipper inn alt lys og normalverdiene for når en brikke ikke er tilstede.
Det løste vi ved å legge til en binær switch som skulle registrere når en brikke var plassert for så å fortelle fototransistoren at den kunne begynne å beregne verdier fra omgivelsene.

Vi minket mengden lys fra omgivelsene ved å bygge opp rundt fototransistoren slik at det eneste lyset som absorberes var lyset som kom fra feltet der hvor brikken ble lagt på. Dermed fikk vi bedre kontroll over hvilke verdier som var normalverdier, verdier når en positiv brikke legges på, og verdier når en negativ brikke legges på. Men det var fortsatt stor variasjon i normalverdiene, så vi endte opp med å kode en funksjon for å kalibrere phototransistoren før den henter inn de nye verdiene. Dermed kunne vi bruke relative maks- og minverdier for å bestemme hvor overgangene burde plasseres. Dette viste seg å være veldig vanskelig å tilpasse, og vi ordnet først og fremst en minimumsløsning med et binært utfall som kunne skille mellom absolutt mørke eller lys.

En annen utfordring var at fototransistoren hadde en viss grad av variasjon og støy i målingene. For eksempel kunne det være en sekvens med omtrent like verdier avbrutt av et avvik som landet i en helt annen kategori. En slik mangel på presisjon ville måtte korrigeres og vi samlet opp verdiene i et array og brukte en gjennomsnittsvariabel av verdiene i dette arrayet til å ta programbeslutninger. Det var ikke en del av minimumsløsningen vår, men mot slutten av prosjektet fikk vi skrevet om på koden og lagt til støtte for flere gjennomsnittsavlesninger, og at programmet tok høyde for slik støy.

Presentasjon av video

Fototransistoren står og kalibrerer for å finne normalverdier fra omgivelsene. Kalibreringen er usynlig for brukeren. Brukeren legger brikken på feltet hvor fototransistoren er plassert. Fototransistoren leser lysverdiene som slipper forbi mellom brikken og hagemodellbordet. Den sammenligner de nye verdiene med de gamle normalverdiene for å finne ut om det er blitt lagt på en brikke og isåfall om den er en negativ eller positiv brikke. I videoen ser vi at brukerne legger på ulike brikker, hvor de store blomsterpottene dekker alt lys og registreres som positive brikker. Brukeren får da feedback i form av grønt blinkende lys, et positivt-tonet "pling" og fuglekvitring. Når brukeren legger på på de små blomsterpottene slippes det inn mer lys og brikken registreres som negativ. Brukeren får da tilbake en feedback for negativ brikke som er blinkende rødt lys og en negativ-tonet bilhorn/alarm-lyd.

Link til video

Video av prototype.

Kode

Koden inkluderer forklarende kommentarer ved de forskjellige kodeelementene.
Lenke til git-repository: https://github.com/Haavardeh/IN1060-prosjektKode.git