We Exist Together

An interactive installation

Table of contents

01

Problem

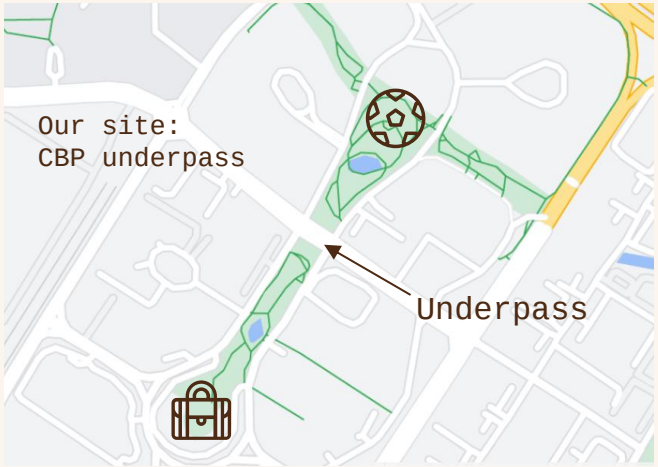
How might we engage users in a shared activity that provides perspective and integrates levity into their lives, enriching the unwinding process

Business Group

Business workers are often stressed after a long day of work. They feel drained and insignificant.

Leisure Group

The park is not engaging to families with children as there are not many opportunities for play.



Business and leisure groups keep to their own side of CBP. Users use the site at different times, limiting interaction. The underpass is an ideal location to connect to 2 groups.

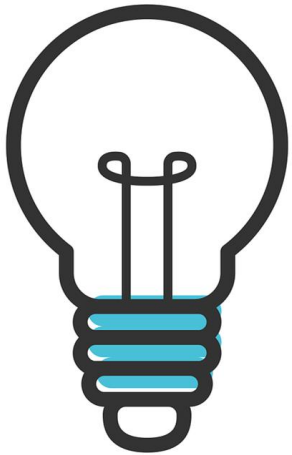
Our Site

The underpass connects the 2 halves of the park, serving as an ideal location to promote interaction between the business and user groups.

However, the underpass is currently dark and brutalist, deterring users from spending time inside and creating a negative experience.

Our Goal

How might we **engage users in a shared activity** that **provides perspective** and **integrates levity** into their lives, **enriching the unwinding process**



02

Solution

Inspirations – Cave Paintings

In the prehistoric ages, humans would mark cave walls with their handprints.

Hands signify human connection. We forge bonds through hand holding and handshakes.

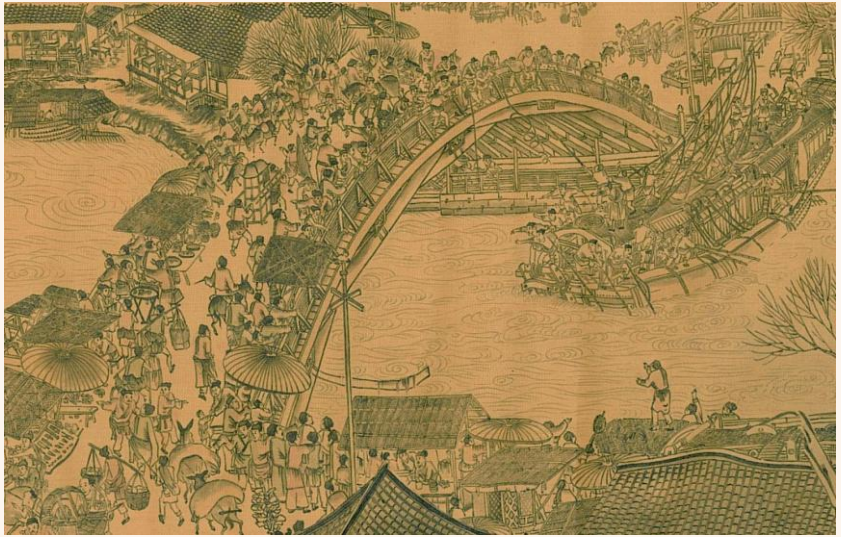
When users create their hand avatar, they are leaving their mark, like the handprints, leaving a sign of their existence.

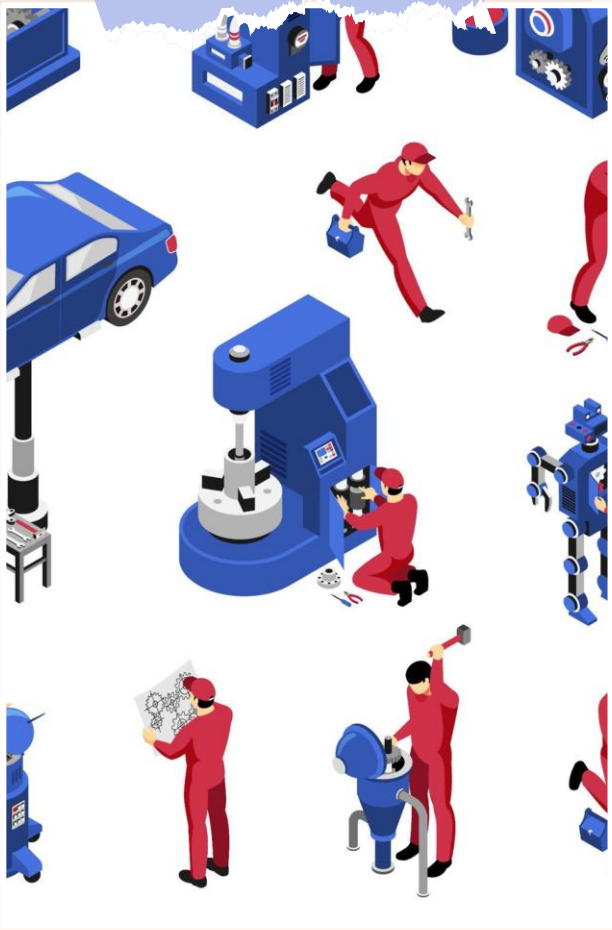
When users see their avatar interacting with the other hands, it will make them feel connected with other users.

Inspirations - 清明上河图

“Along the River During the Qingming Festival” is a famous Chinese painting. At a glance, the humans in the painting seem insignificant. But when we look closer, we see that each of them has a story to tell, and that their interactions with one another was what truly makes a civilisation vibrant.

In our design, each hand will move independently to assert each users individual significance. The hands will interact with each other, adding vibrancy to our village.





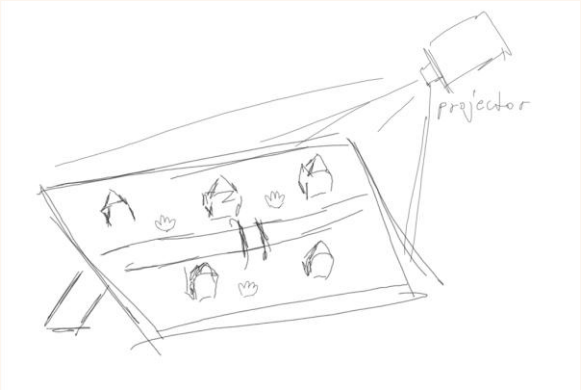
03

Prototyping

The Slanted Projection

We designed the hands to be projected on a slanted surface so we could blend 3D and 2D elements.

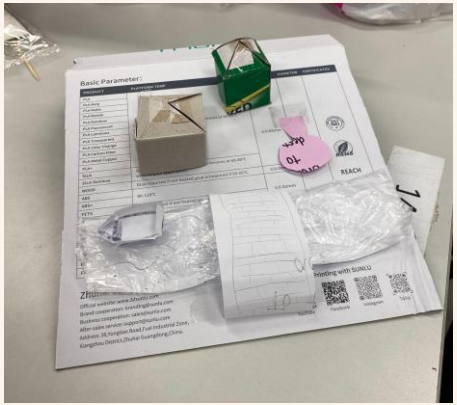
The 2D projected hands gave us more flexibility when adjusting the hand animations and added a whimsical, storybook feel.



Sketch of the projection setup

The 3D elements grounded the design, making looking at the project feel less like staring at a screen.

The slant prevents the hands from looking flat while allowing the 3D elements to protrude from the surface.



Our first prototype

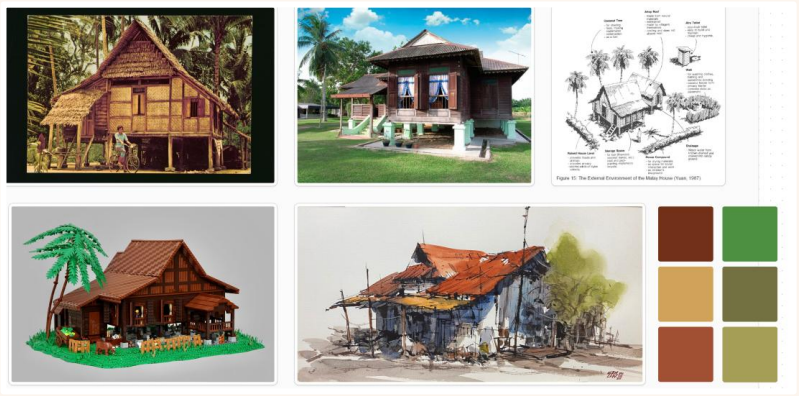
Designing the Houses

Our first set of houses were made of basic shapes to test the slant angle and determine the the ideal size for our houses.

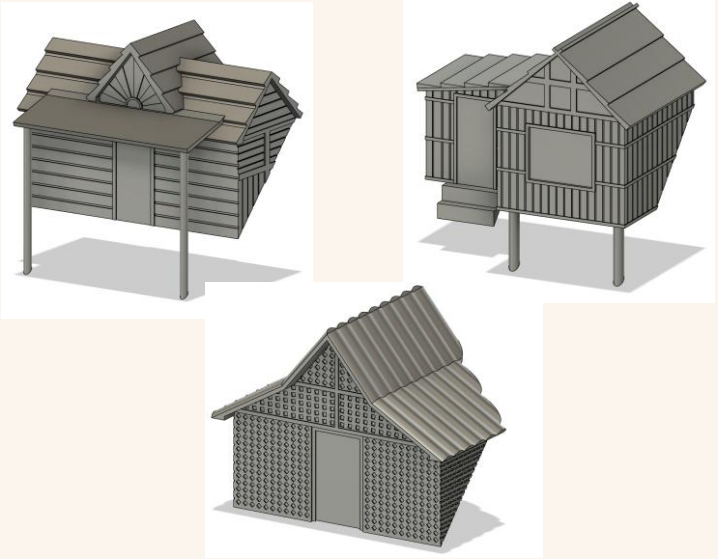
In the 2nd set of houses, we explored different styles and experimented with different textures.

Designing the Houses

We decided to take model our houses after kampong houses. The kampongs localise the “Along the River During the Qingming Festival” inspiration and draws and cultural solidarity.



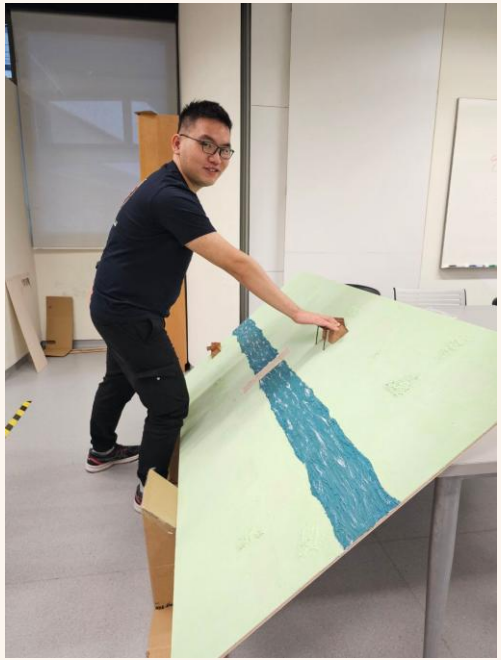
Moodboard for our houses



Final house designs modeled in Fusion

Our final set of houses capitalised on the intricate textures the 3D printer could produce.

Projection



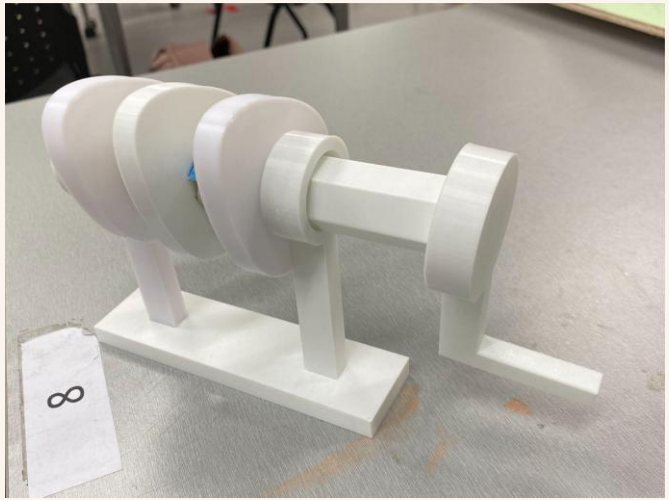
We tested the video of the walking hands on the board to determine and adjust the dimensions of the image

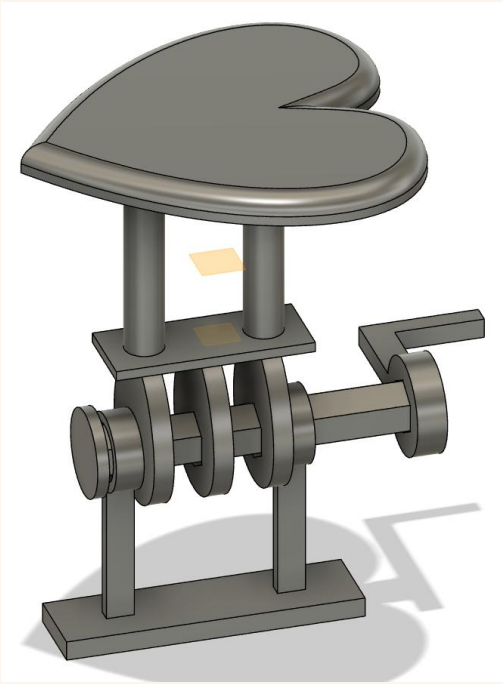
Pulsing Heart

In our prototype, the heart is supported by egg-shaped gears. As the gear rotate, the heart oscillates up and down.

Problems:

- The crank was not fast enough to mimic an average heartbeat
- The oscillation was too drastic
- The crank was not secure enough

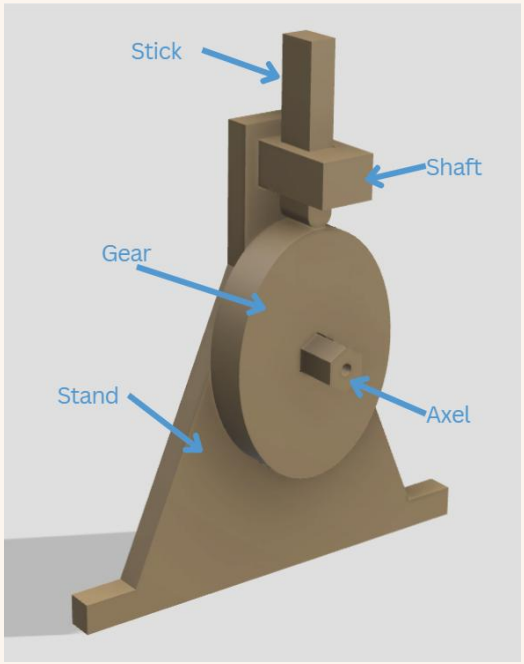




Pulsing Heart

Final design improvements:

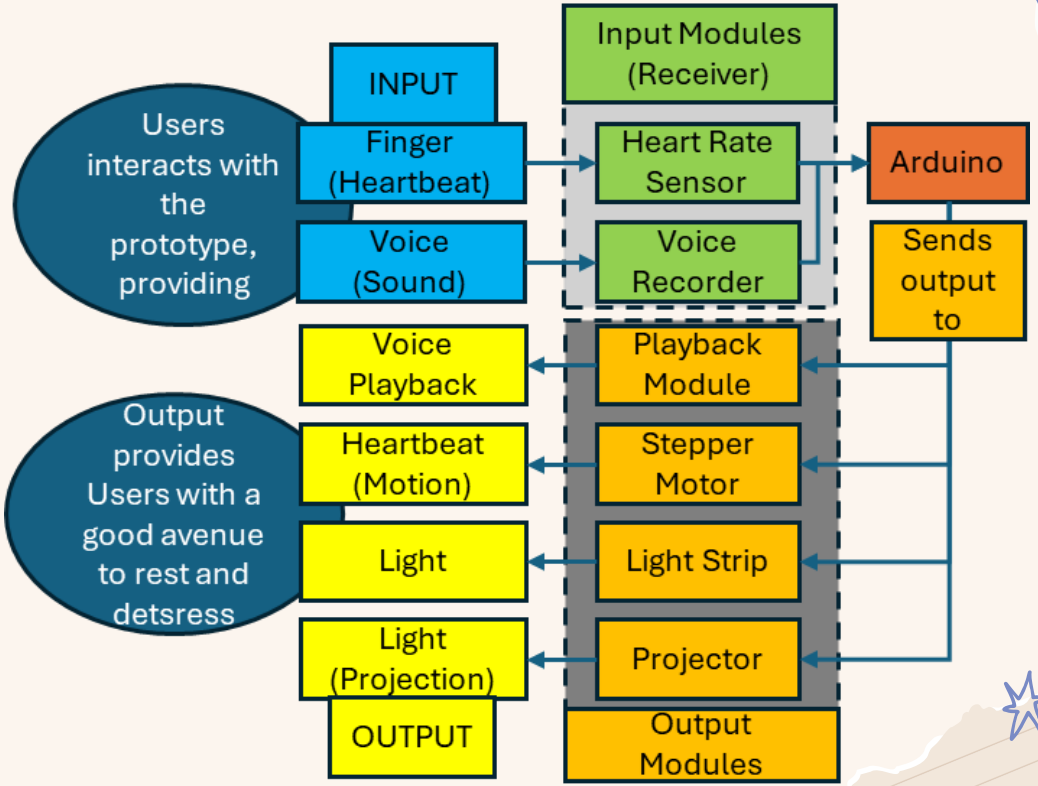
- Oval shaped gear to reduce the period of oscillation so we do not have to crank as fast
- Shaft to hold the stick in place and prevent slipping
- More compact design to save space
- Rounded end on the stick to reduce friction



Electronics/ Programming

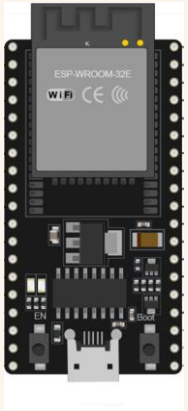
Functional Block Diagram

Before deciding on the electronics required, the team created the functional block diagram (Shown on the left) to understand the input/ output of the product involved, as well as the process behind them.



Arduino- ESP32

Arduino used in the Electronics is the ESP32, mainly due to the team's programmer being already familiar with programming the arduino

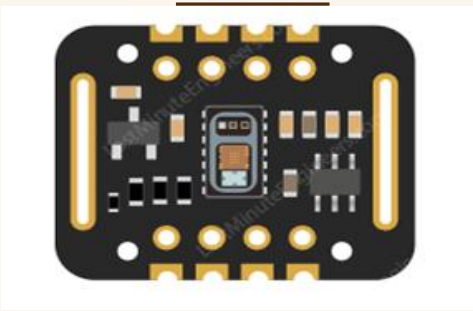


Electronics/ Programming

Criteria for Choice of Electronics Used:

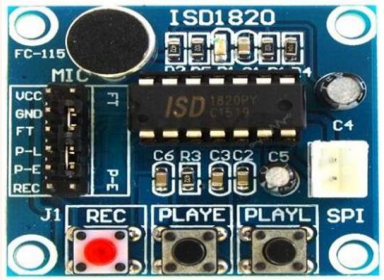
- Ease of wiring (preferably no need for external capacitor/ resistors etc.)
- Ease of Programming (a specialised Arduino library exist for the component)

Heart Rate Sensor- MAX30102

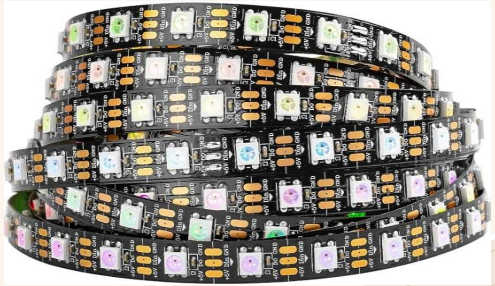


Projector

Voice Recorder/ Playback Module - ISD1820



Light Strip- WS2812B Addressable LEDs

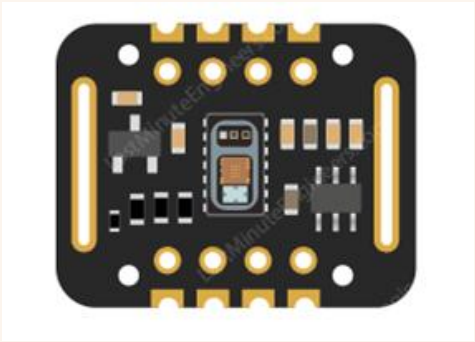


Electronics/ Programming

Heart Rate Sensor- MAX30102

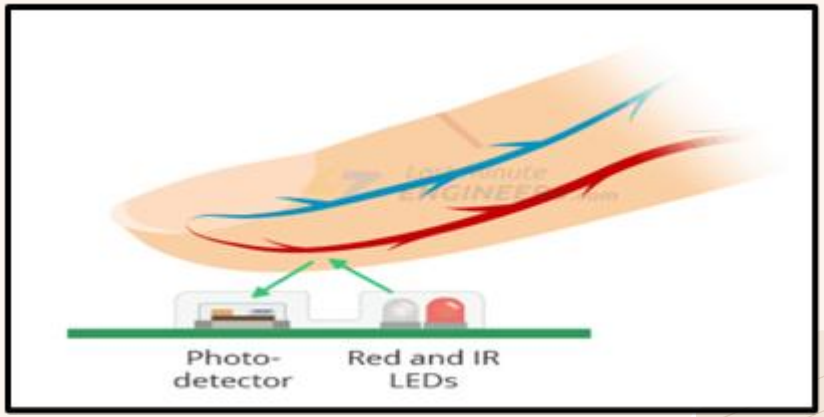
The Heart Rate Sensor used in the electronics is the MAX30102 due to its ability to read various data from the users, including:

- Heart rate (BPM),
- Temperature,
- Oxygen Saturation (SpO2)



How it works

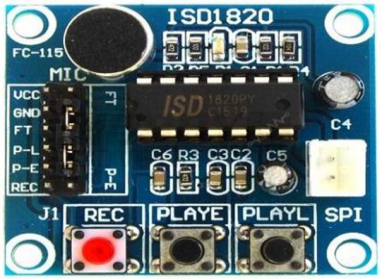
The Heart Rate Module utilizes both red and infrared (IR) light to illuminate the finger.
It then detects the reflected light using a photodetector integrated into the module.

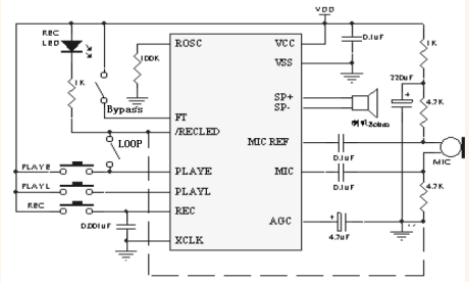


Electronics/ Programming

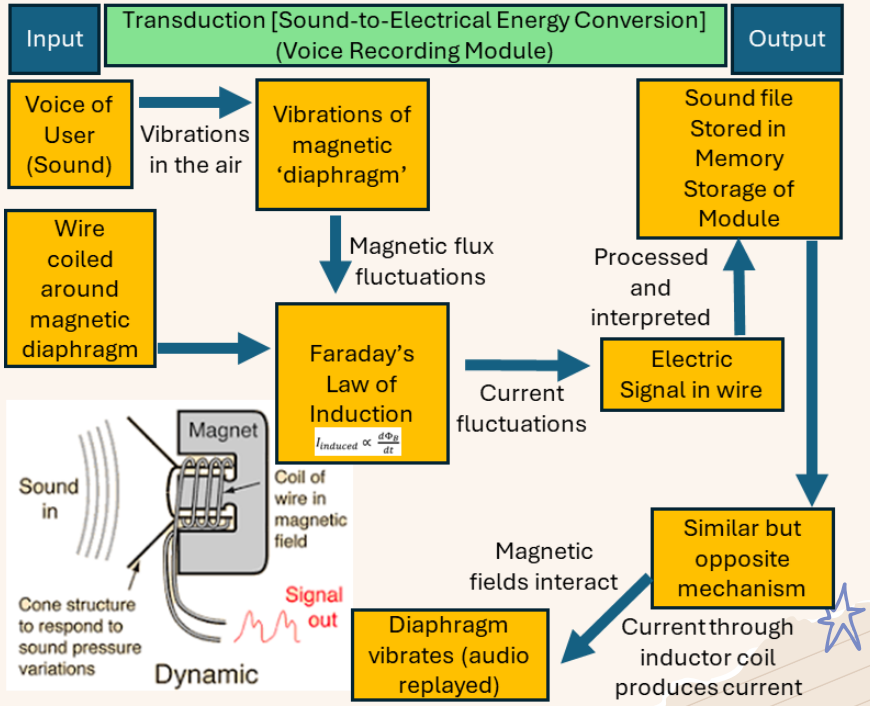
Voice Recorder/ Playback Module - ISD1820

The Voice Recorder/ Playback Module is used to record user's' voice and playback the voice.





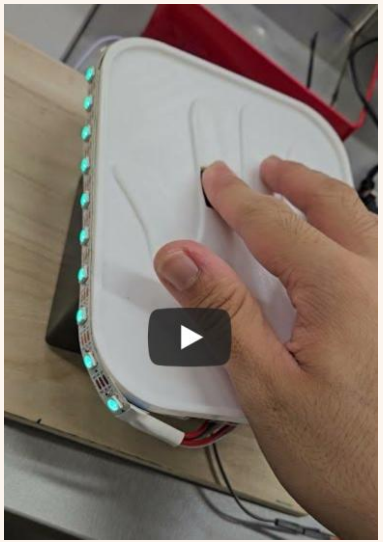
How it works



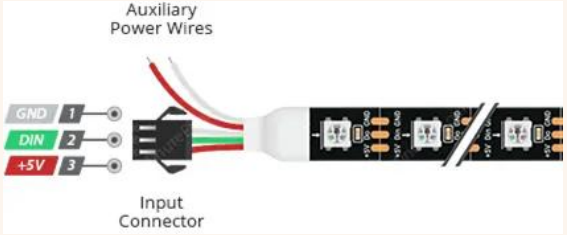
Electronics/ Programming

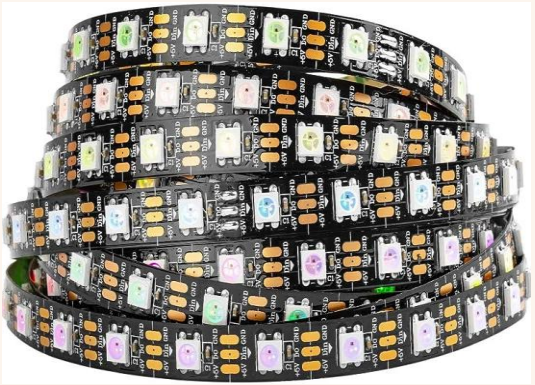
Light Strip- WS2812B Addressable LEDs

Light Strip Program Testing



Link
https://youtube.com/shorts/bSr_hOOo2fU?feature=share





Creating hand objects

```
hand_group = pygame.sprite.Group()

upper_bounds1,upper_bounds2= 220,290
lower_bounds1,lower_bounds2 = 608,710

hand_1 = Hand(animation_list, randint(70,1450),255,1,2,hand_group,None)
hand_2 = Hand(animation_list, randint(70,1450),259,1,1,hand_group,None)
hand_3 = Hand(animation_list, randint(70,1450),650,1,0,hand_group,None)
hand_4 = Hand(animation_list, randint(70,1450),690,1,3,hand_group,None)
hand_5 = Hand(animation_list, randint(70,1450),randint(upper_bounds1,upper_bounds2),1,4,hand_group,None)
hand_6 = Hand(animation_list, randint(70,1450),620,1,5,hand_group,None)

hand_7 = Hand(animation_list, 900,610,1,6,hand_group,None)
hand_8 = Hand(animation_list, 1000,610,1,7,hand_group,None)
hand_9 = Hand(animation_list, 350,250,1,8,hand_group,None)
hand_14 = Hand(animation_list, 450,250,1,7,hand_group,None)

hand_10 = Hand(animation_list, 550,610,1,9,hand_group,None)
hand_11 = Hand(animation_list, 500,610,1,10,hand_group,None)
hand_12 = Hand(animation_list, 1150,250,1,11,hand_group,None)
hand_13 = Hand(animation_list, 1200,250,1,10,hand_group,None)

hand_15 = Hand(animation_list, 550,255,1,3,hand_group,None)
#hand_15 = Hand(animation_list, randint(70,1450),randint(400,700),1,11)

hand_group.add(hand_1,hand_2,hand_3,hand_4,hand_5,hand_6,hand_7,hand_9,hand_10,hand_11,hand_12,hand_13,hand_14,hand_8,hand_15)
```

Creating hand objects require the animation list, coordinates, speed and starting action.

The library used is Pygame.

Coding - Hand object

Importing libraries

```
1   import pygame
2   from timer import Timer
3   from decimalrect import DecimalRect
4   from random import randint
```

Initialising variables

```
class Hand(pygame.sprite.Sprite):
    def __init__(self, animation_list, x, y, speed, action,group,excluded_list):
        pygame.sprite.Sprite.__init__(self)
        self.speed = speed
        self.yspeed = speed
        self.animation_list = animation_list
        self.frame_index = 0
        self.action = action
        self.update_time = pygame.time.get_ticks()
        self.talk = 0
        self.paint = 0
        self.walk = 0
        self.stand = 0
        self.walk_started = False
        self.group = group

        if self.action == 0 or self.action == 1 or self.action == 2 or self.action == 3 or self.action == 4 or self.action == 5:
            self.walk = 1
            self.walk_started = True
        if self.action == 6 or self.action == 7 or self.action == 8:
            self.paint = 1
        if self.action == 9 or self.action == 10 or self.action == 11:
            self.talk = 1

        self.creativity = randint(1,10)
        self.productivity = randint(1,10)
        self.socialbility = randint(1,10)

        self.walk_timer = Timer(self.productivity*5000)
        self.talk_timer = Timer(150000)
        self.paint_timer = Timer(self.creativity*5000)
        self.stand_timer = Timer(6000)

        self.image = self.animation_list[self.action][self.frame_index]
        self.rect = pygame.Rect(0, 0, 73, 73)
        self.rect.center = (x, y)
```

Updating Actions

```
class Hand(pygame.sprite.Sprite):

    def update(self, surface):

        if self.action == 0 or self.action == 1:  # Walking actions
            if self.rect.right > 1500:
                self.update_action(1)
            elif self.rect.right < 70:
                self.update_action(0)

            if self.action == 0:
                self.rect.x += self.speed
            if self.action == 1:
                self.rect.x -= self.speed

            if self.action == 0:
                if self.rect.top <260:
                    if self.rect.x == 250:
                        self.update_action(12)

        if self.action == 2 or self.action == 3:  # Walking actions
            if self.rect.right > 1500:
                self.update_action(3)
            elif self.rect.right < 70:
                self.update_action(2)

            if self.action == 2:
                self.rect.x += self.speed
            if self.action == 3:
                self.rect.x -= self.speed

            if self.action == 3:
                if self.rect.top <260:
                    if self.rect.x == 520:
                        self.update_action(15)
```

0,1,2,3,4,5, are walking actions
Self.rect.x += self.speed is how the hand is moved across the screen.

```
        if self.action == 4 or self.action == 5:  # Walking actions
            if self.rect.right > 1500:
                self.update_action(5)
            elif self.rect.right < 70:
                self.update_action(4)

            if self.action == 4:
                self.rect.x += self.speed
            if self.action == 5:
                self.rect.x -= self.speed

            if self.action == 4:
                if self.rect.top >450:
                    if self.rect.x == 810:
                        self.update_action(16)

        if self.action == 12 or self.action == 13 or self.action == 14 or self.action == 15 or self.action == 16 or self.action == 17:
            if self.stand == 1:
                self.stand_timer.activate()
                self.stand = 0
            else:
                self.stand_timer.update()
                if not self.stand_timer.active:
                    if self.action == 12:
                        self.update_action(0)
                    if self.action == 15:
                        self.update_action(3)
                    if self.action == 16:
                        self.update_action(4)

        self.update_animation()
```

Updating animation and action

```
def update_animation(self):
    #define animation cooldown
    ANIMATION_COOLDOWN = 25
    #update image depending on current action
    self.image = self.animation_list[self.action][self.frame_index]
    #check if enough time has passed since the last update
    if pygame.time.get_ticks() - self.update_time > ANIMATION_COOLDOWN:
        self.update_time = pygame.time.get_ticks()
        self.frame_index += 1
    #if the animation has run out then reset back to the start
    if self.frame_index >= len(self.animation_list[self.action]):
        self.frame_index = 0

def update_action(self, new_action):
    if new_action != self.action:
        self.action = new_action
        self.frame_index = 0
        self.update_time = pygame.time.get_ticks()
    if self.action == 9 or self.action == 10 or self.action == 11:
        self.talk = 1
    elif self.action == 6 or self.action == 7 or self.action == 8:
        self.paint = 1
    elif new_action == 0 or new_action == 1 or new_action == 2 or new_action == 3 or new_action == 4 or new_action == 5 :
        if self.walk_started == False:
            self.walk = 1
            self.walk_started = True
            self.walk_timer = Timer(randint(1,10)*1000)
    elif self.action == 12 or self.action == 13 or self.action == 14 or self.action == 15 or self.action == 16 or self.action == 17:
        self.stand = 1
```

Animation cooldown determines the frame rate.

Updating the action initialises the action sequence.

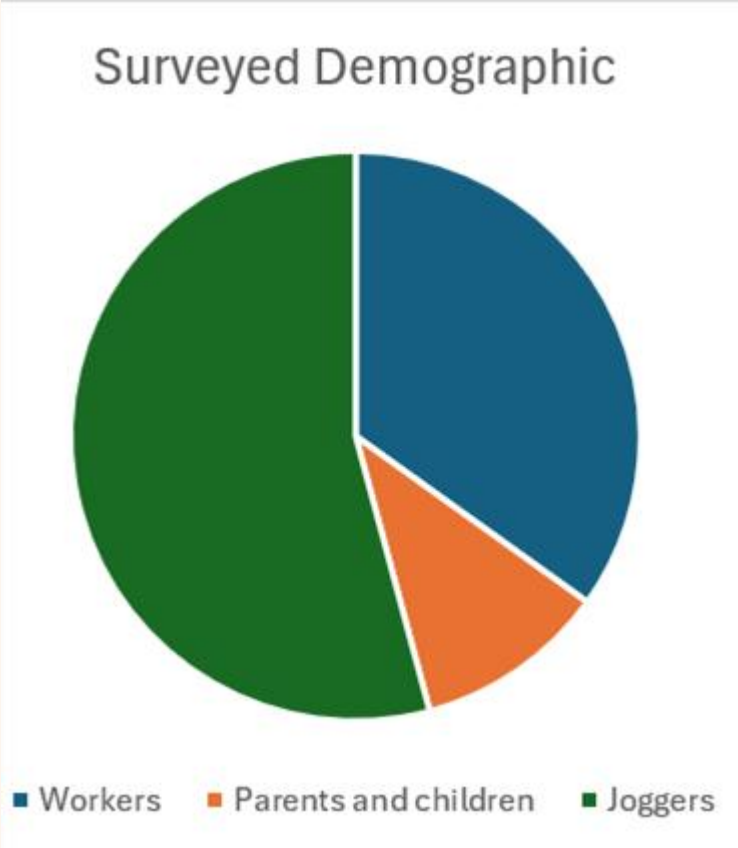
04

Testing

User flow

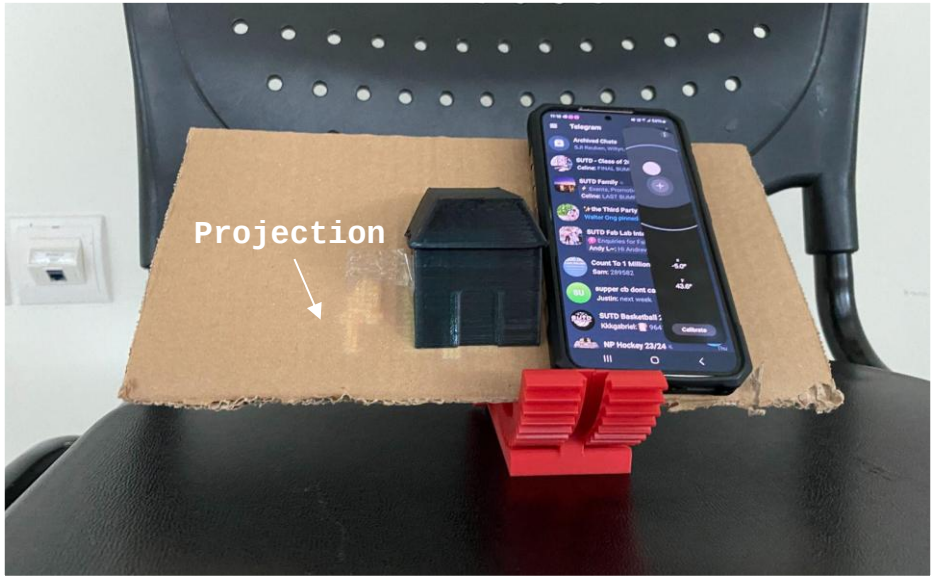
	Workers	Parents and children	Joggers
Monday	60	0	1
Tuesday	58	3	2
Wednesday	55	5	5
Thursday	49	0	1
Friday	45	8	9
Saturday	3	15	10
Sunday	1	20	13
Total	271	51	41
Grand total			363

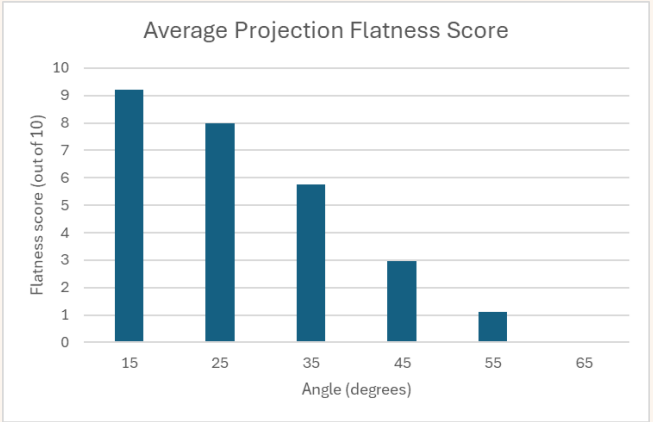
We carried out a site survey in the early phases to narrow down on site's demographic.



Angle Test

We tested for the optimal angle to slant the base, determined by the smallest angle where users scored the flatness 3 or bellow.





UI UX Testing

We tested our UI UX through user surveys. After exploring the interface, we asked users to rate their experience and suggest improvements. Based on the survey results, users most enjoyed the range of of functions, referring to the voice recording, customisation and viewing details of other users.

"I like the part where you can make your own hand, especially the faces. The faces are funny."

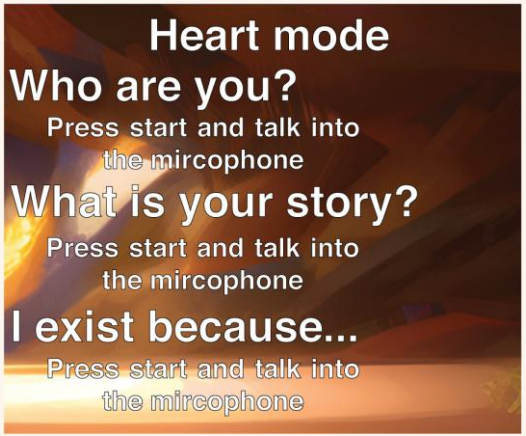
"It's quite a lot of text. If I saw it, I think I would just skip through, but I like reading the stuff about the other hands."

"It's an interesting idea. It's like a game, but you can learn about other people."

UI UX Testing

One critique was that the text was annoying or difficult to read, especially for the voice recording page. The staggered text and multiple titles made the page feel messy.

We improved our prototype by splitting the text into 3 pages, making the explanation clearer. We also only had 1 title per page.



Prototype

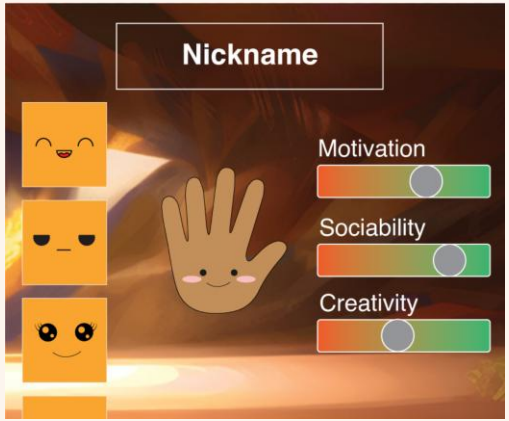
Improvement

Final

UI UX Testing

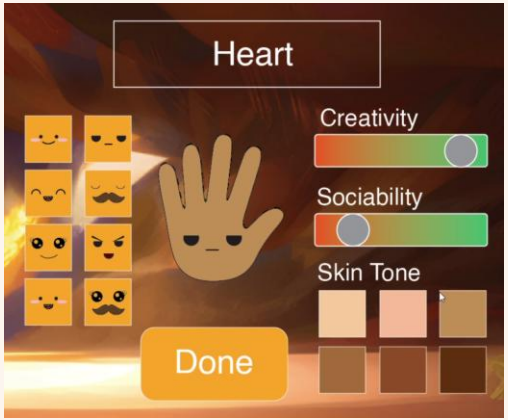
Users also complained about that in the customisation page, there was no option to adjust the skin tone of the hand. The hands are supposed to represent the individual user, but the default skin tone left some users feeling underrepresented.

To fix this we replace the motivation slider, a characteristic users were often confused by, with a skin tone selection menu with 6 different skin tones.



Prototype

Improvement

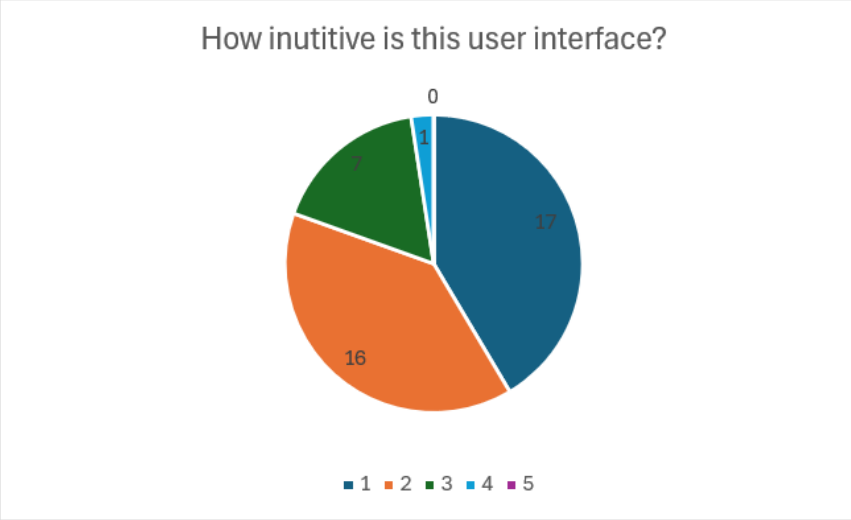


Final

UI UX Testing

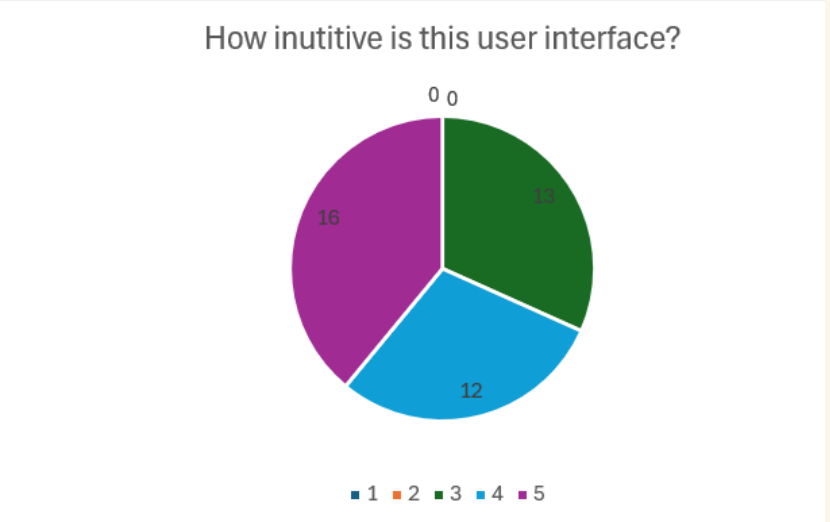
Other changes were also made to make the UI more intuitive, such as making clear “proceed” and “next” buttons. Based on a sample size of 40, a sizable improvement was made, increasing the average rating from 1.8 to 4.0

Before



					Average	1.804878

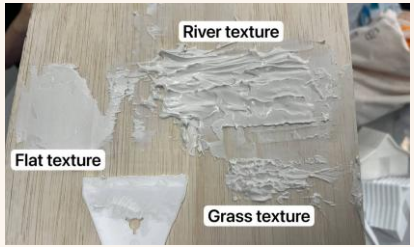
After



					Average	4.07317073

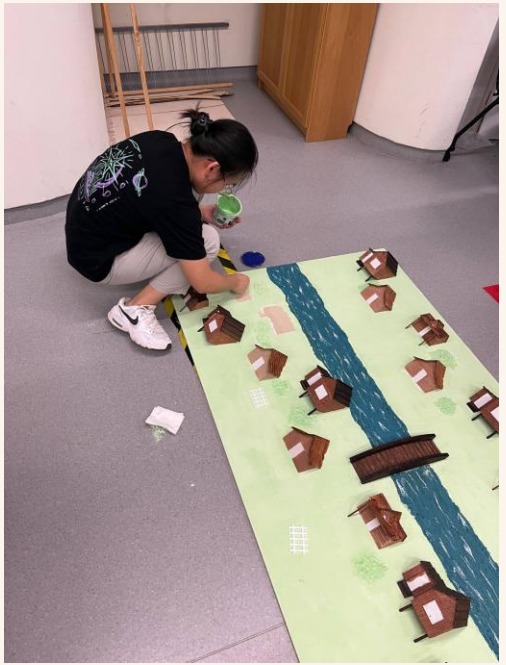
Colour and Texture Testing

Tested the paint on the materials



Experimenting with wall putty textures

We tested the colours under the projection light



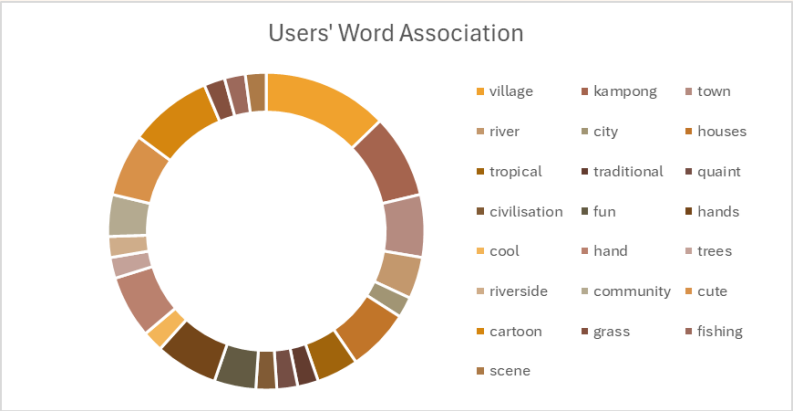
Painting the board and deciding on the layout of the houses

User Opinion Testing

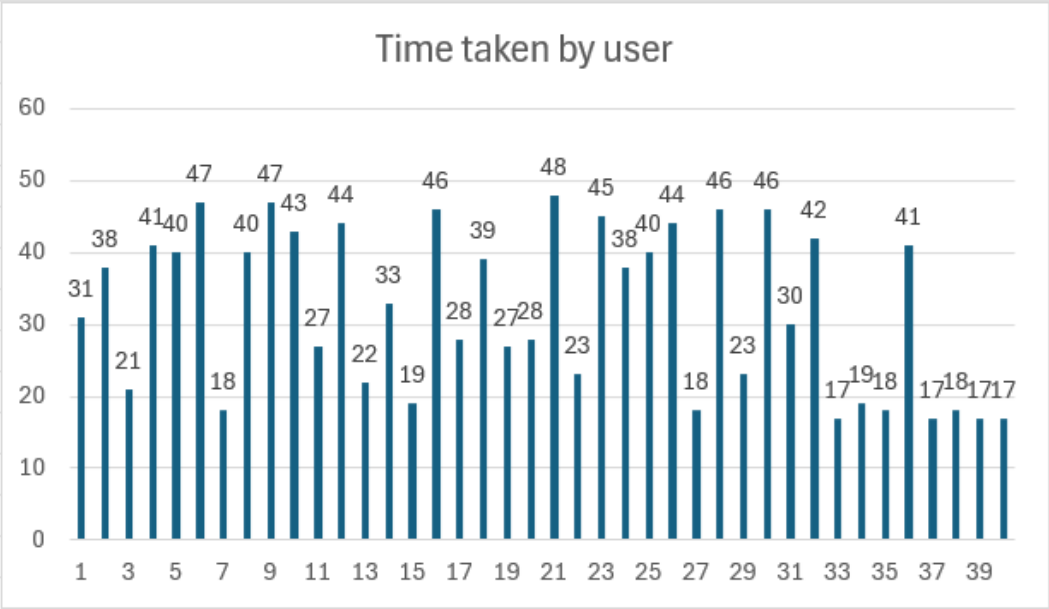
We asked users what they thought of our design to evaluate how well we communicated the themes of community and levity.

“Village” and “kampong” were 2 of the top five words users used to describe our design. Some even used the word “community” itself, showing a strong connection to the theme of community.

Users also described the design as “fun”, “cute” and “cool”, demonstrating an association to levity.



“Heartmode” recording length



Average time
32.15

In answering our questions, users took a minimum of 15s and a maximum of about 50s. The average time we calculated based on a sample size of 39 is 32.15.

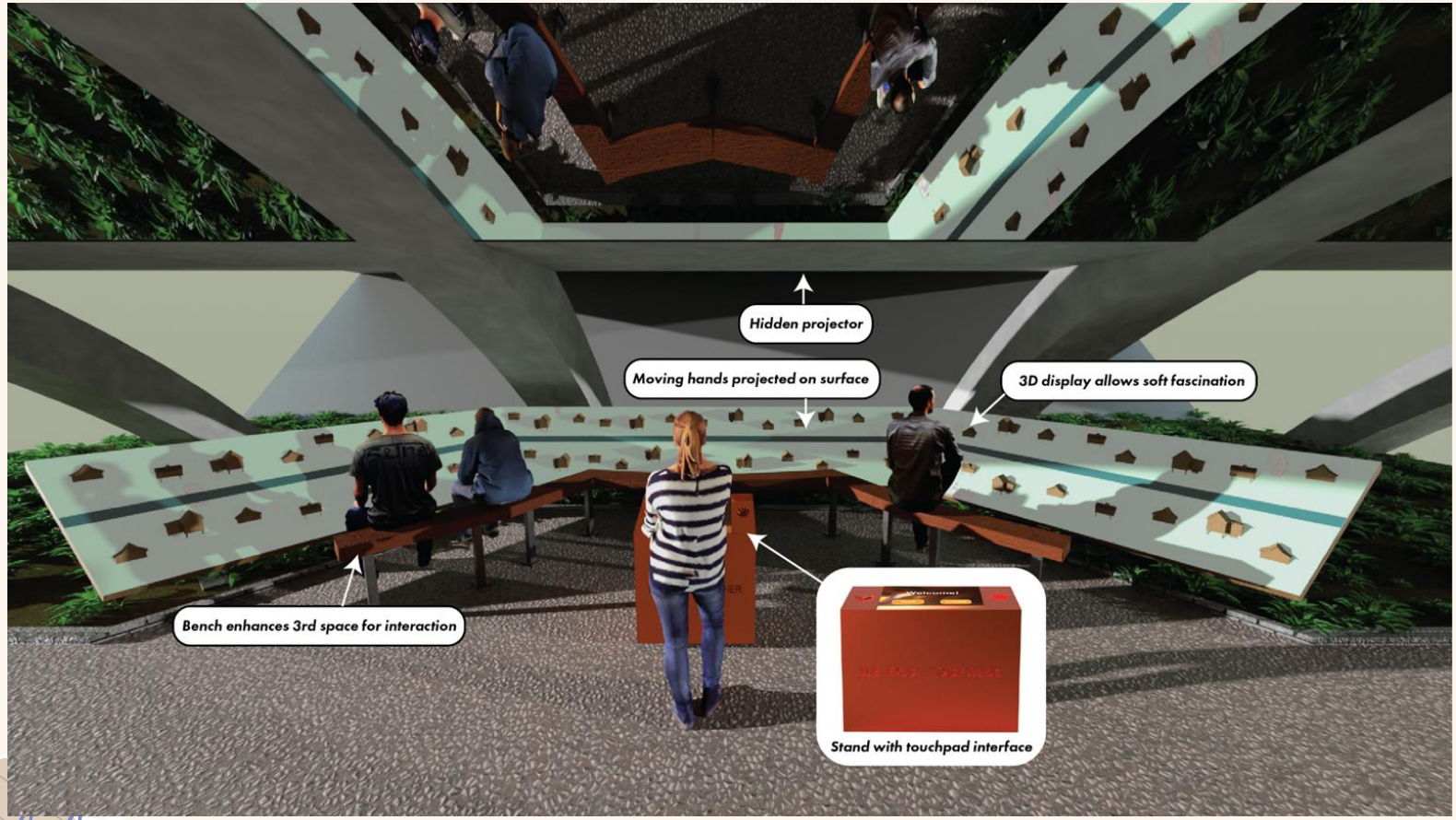
Despite the average being about 30s, we maintained that the time limit for “heart mode” be a minute so that users are able to fully express themselves.

A “done” button is added for users that do not need so much time.

05

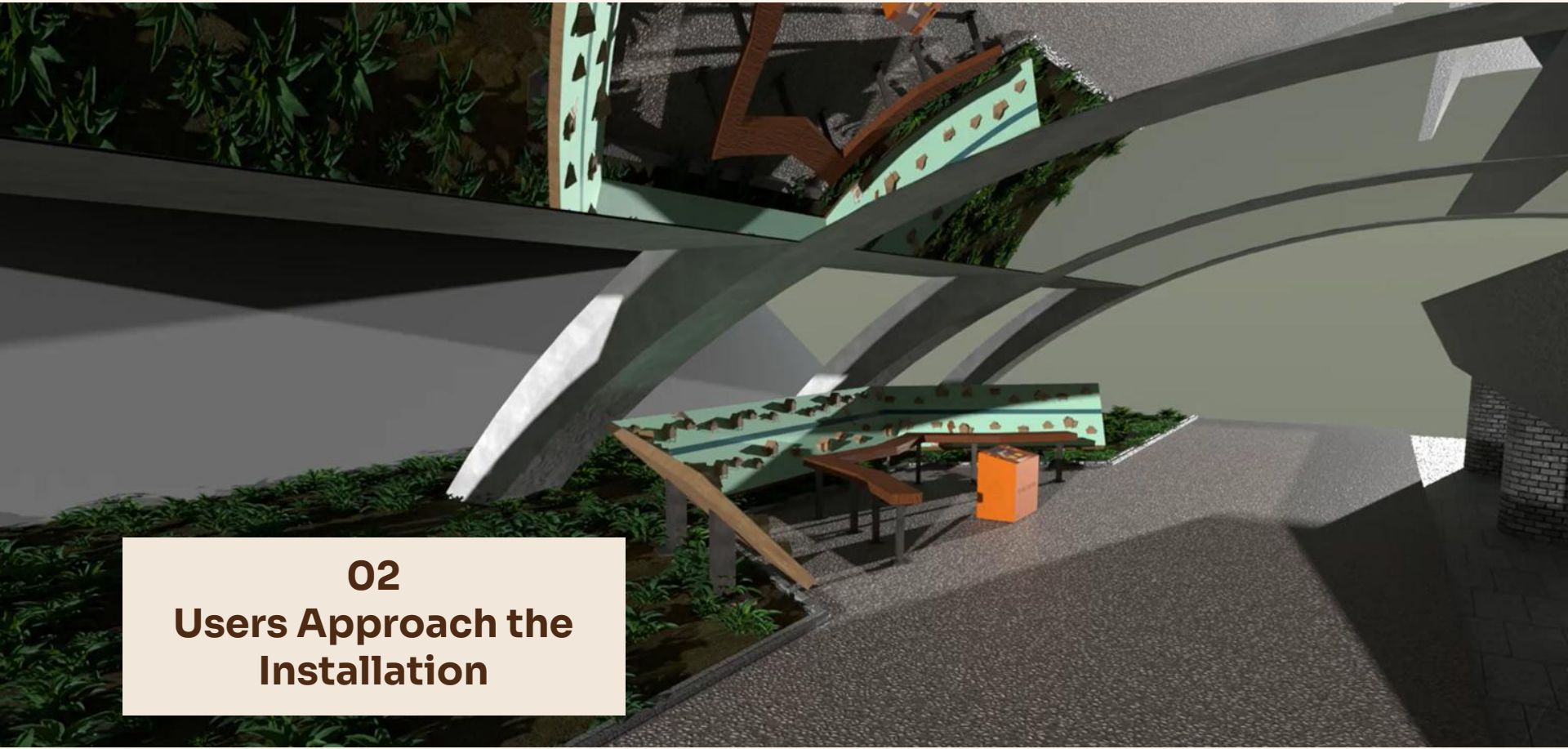
Final Product

One critical image

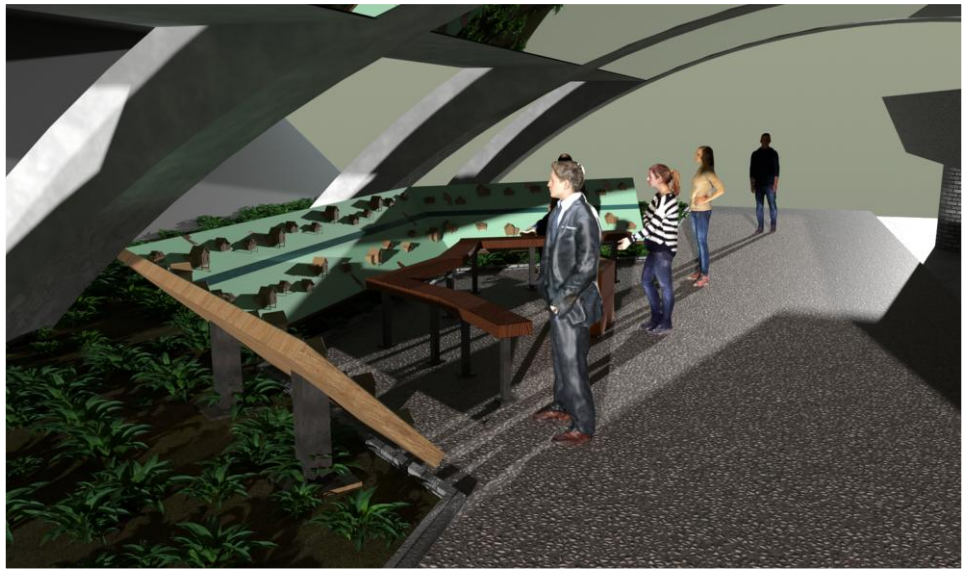


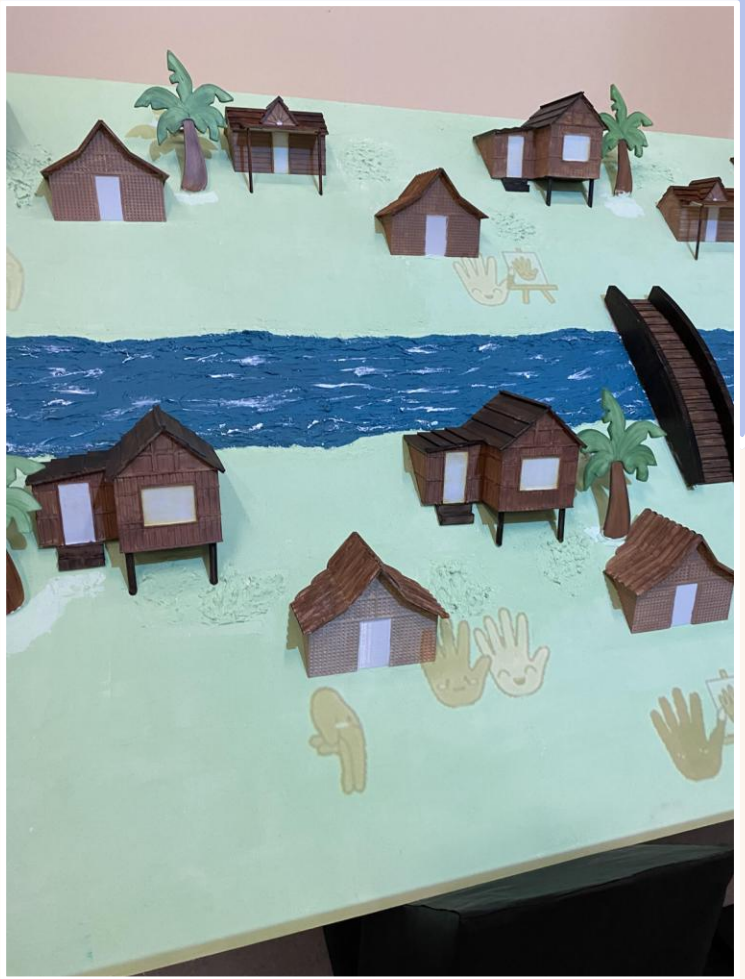
01

Users enter the underpass and see the mural



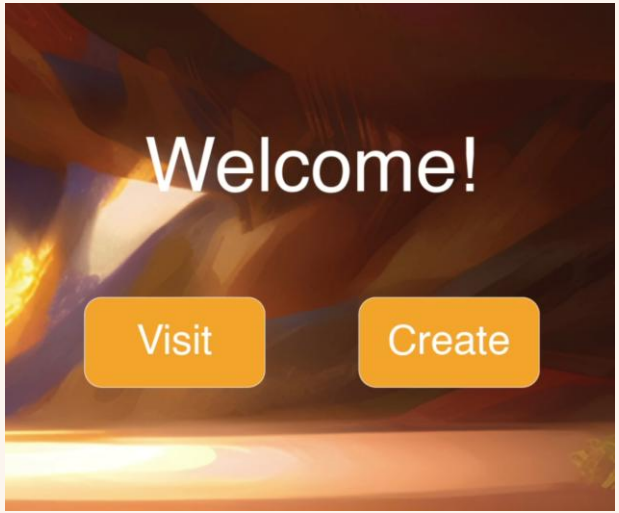
02
Users Approach the Installation





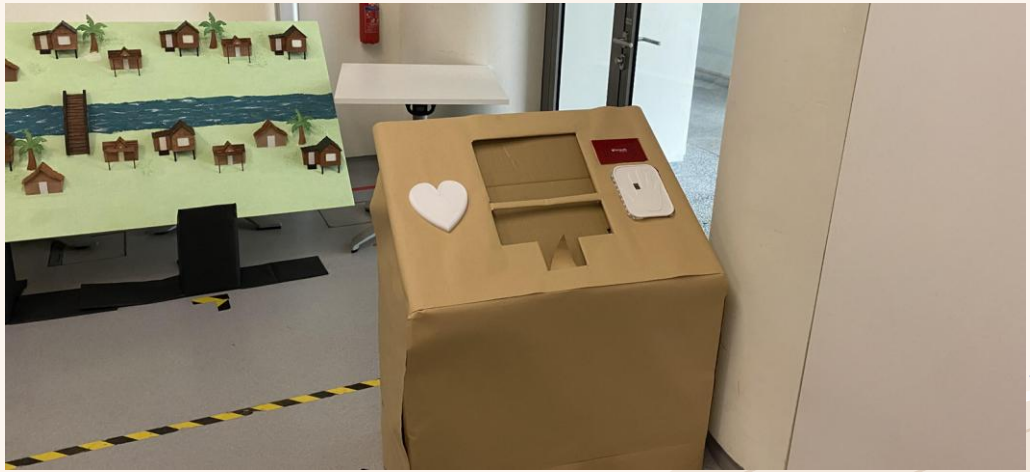
03

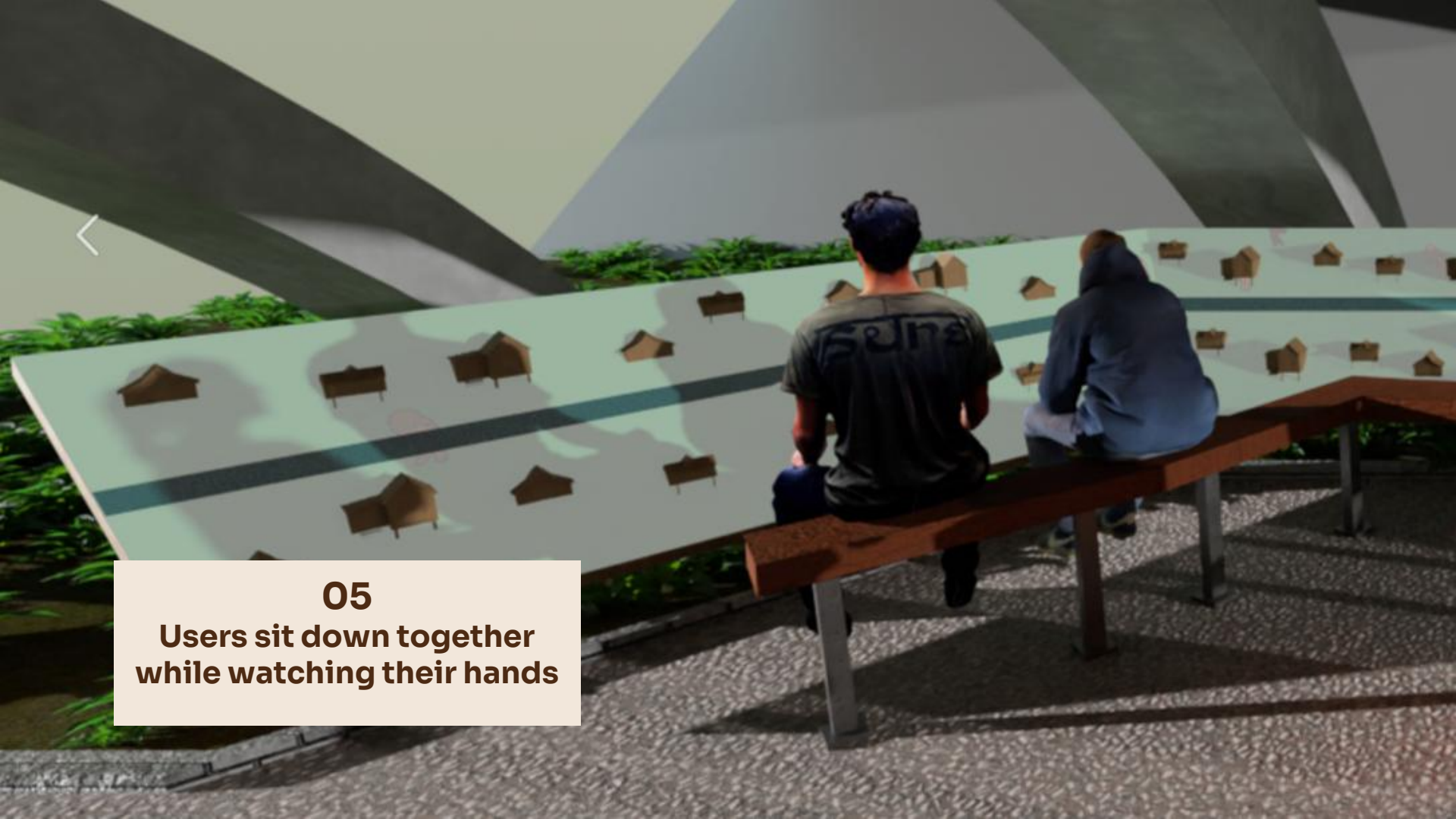
Users watch the hands walk around the village among the 3d printed decorations.



04

Users Use the stand to create their avatars get their heart rate recorded and record their voice.





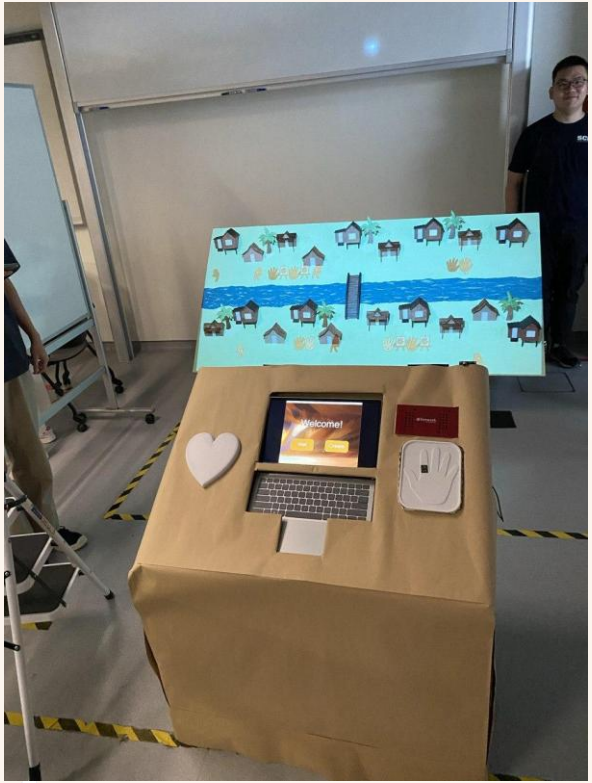
05

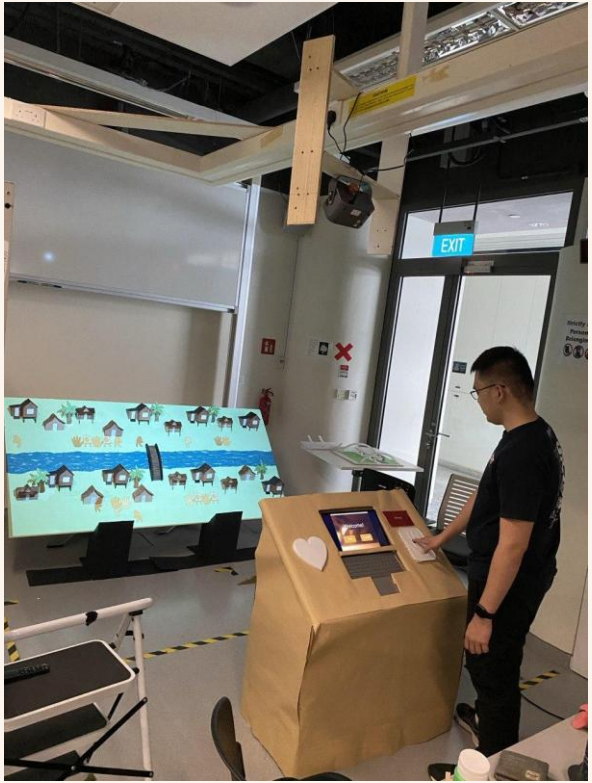
Users sit down together while watching their hands


Scale Model
1:50

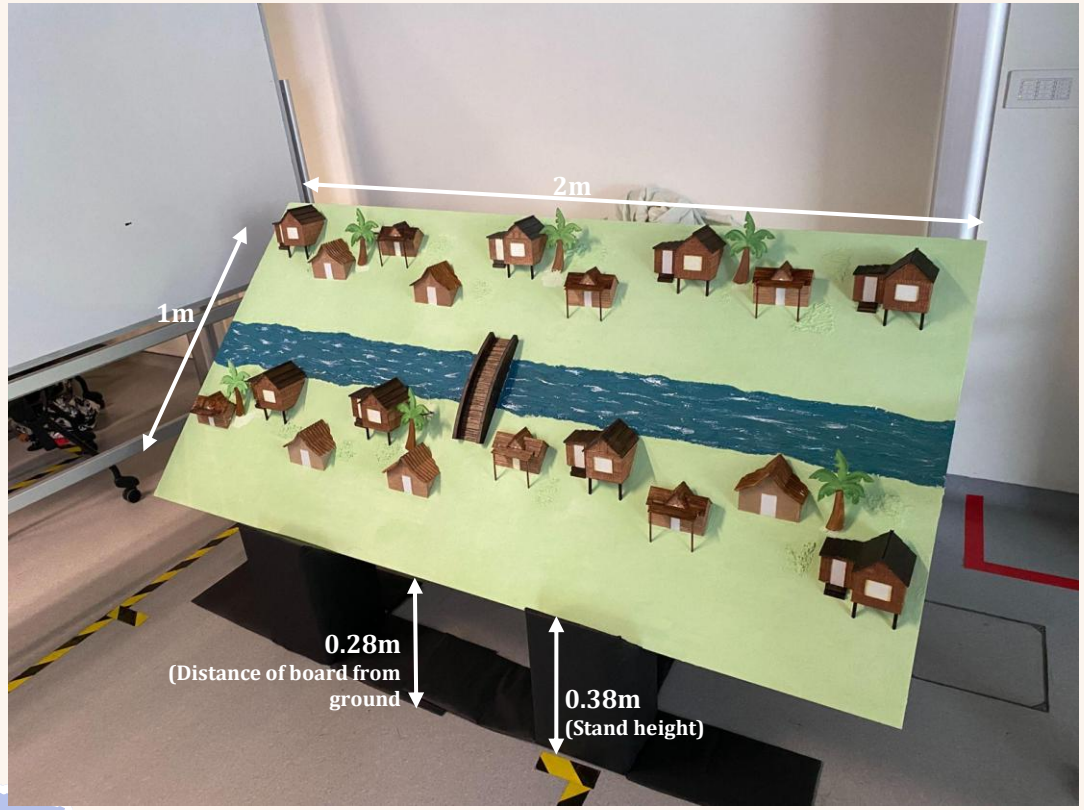
Demonstration Prototype

Demonstration Prototype





Technical Specifications



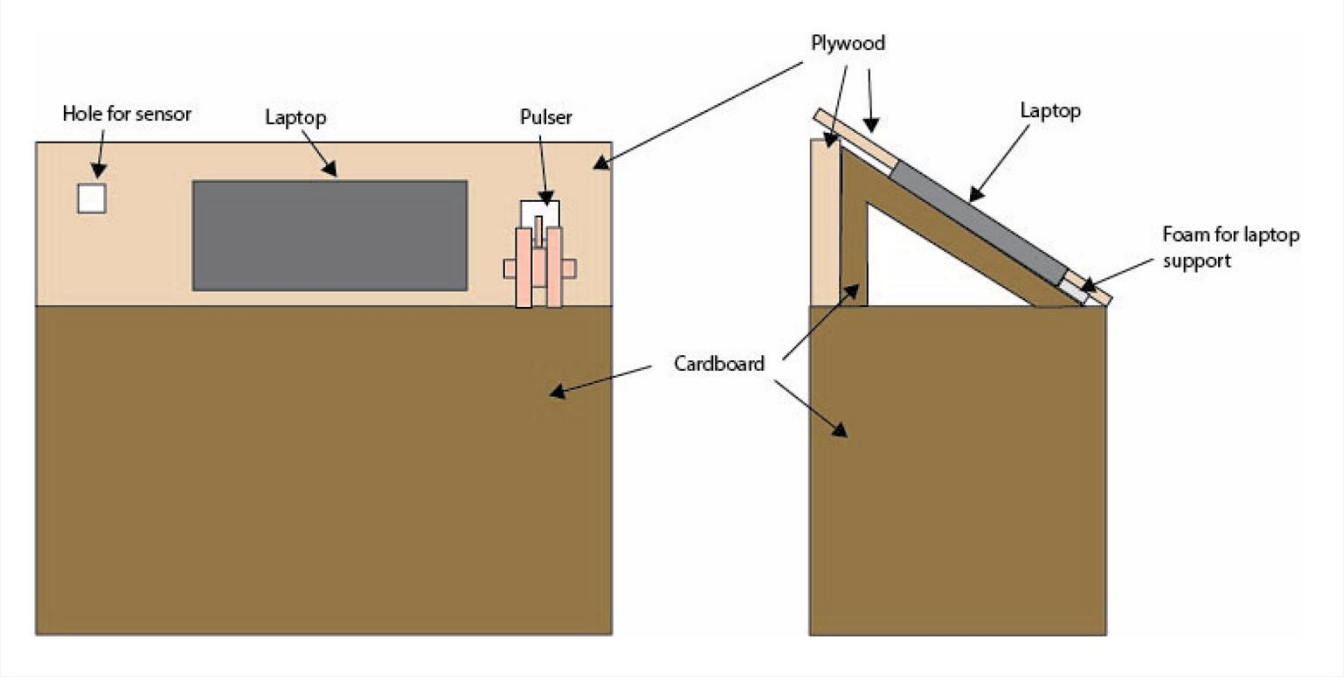
Board
- 2m x 1m plywood
- Covered in a layer of wall putty and acrylic paint

River
- Wall putty and acrylic paint

Houses, trees and bridge
- 3D printed with PLA
- Painted with acrylic paint

Technical Specifications

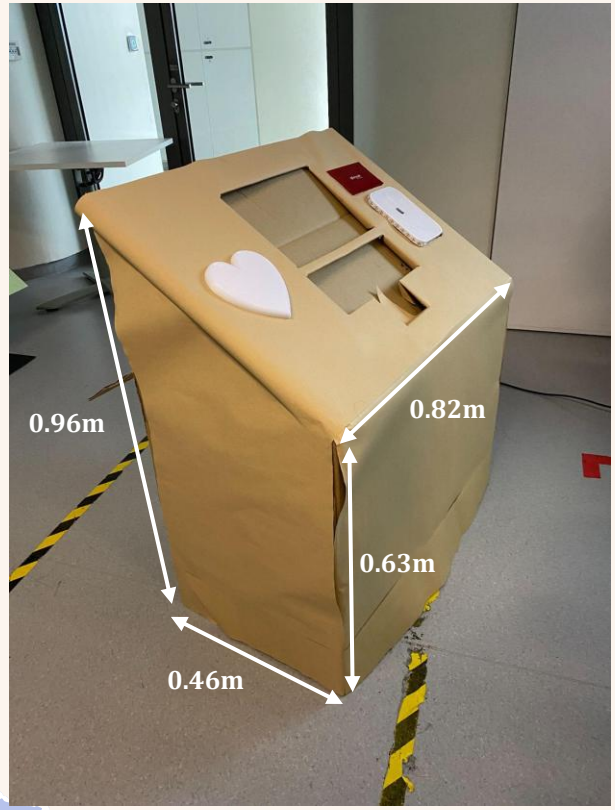
Stand Interior



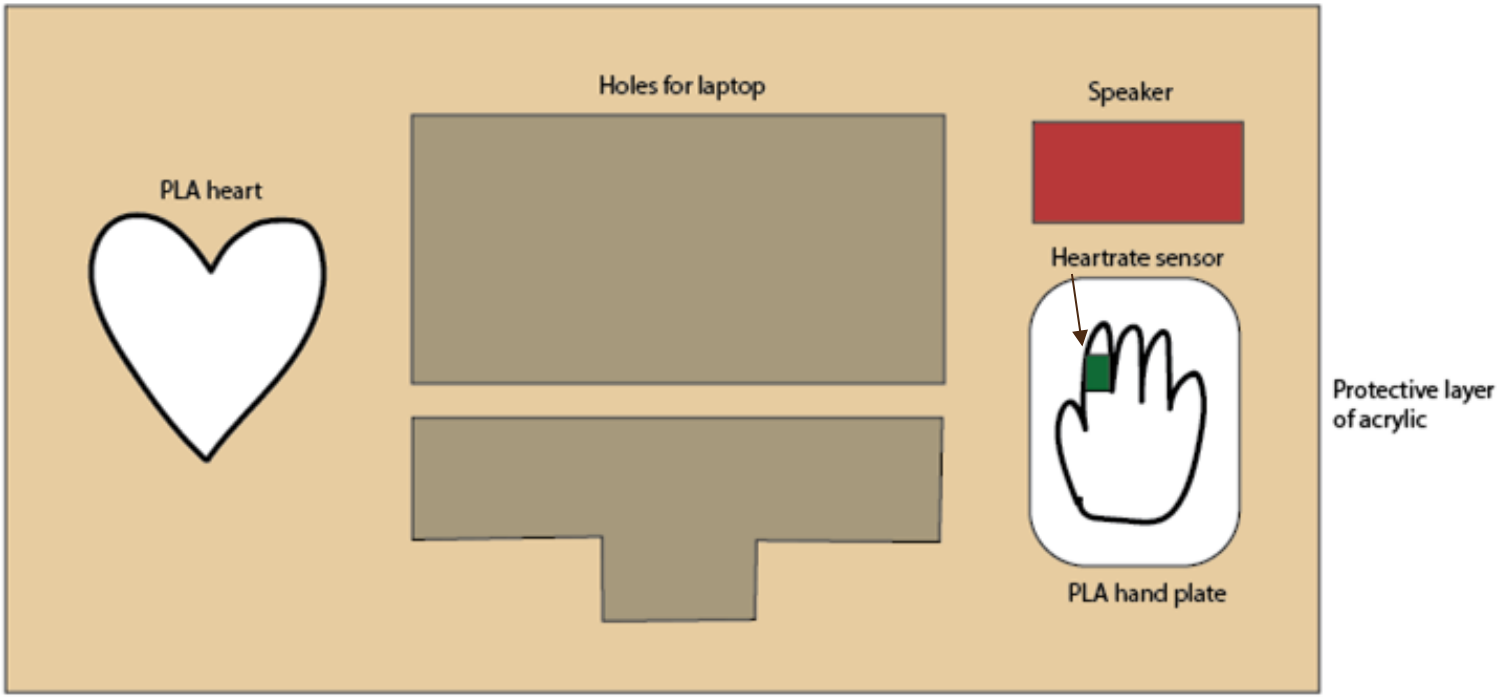
Back view

Side view

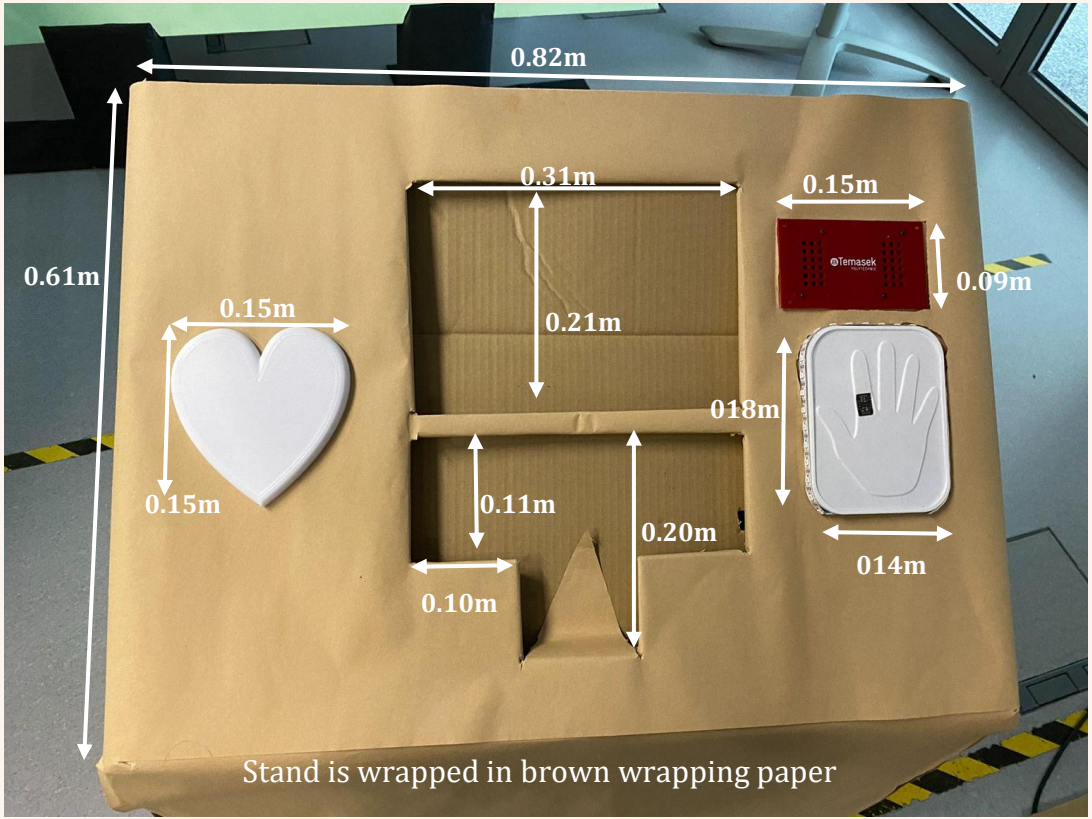
Technical Specifications



Technical Specifications



Technical Specifications



Future Improvements

- Install the electric motor into the stand so the heart pulser can function autonomously and accurately replicate the heartbeat
- Make the stand out of wood and make compartments to easily access internal components
- Design housing for the electronics
  - The LEDs around the hand are exposed
  - The electronics and wires are not organised
- Create a mount for the projector after determining the best location to mount it
- Replace laptop with a touchscreen
- Add selection by area on

06

Additional Information

Additional Information

Keywords:
Community, play, destress, soft fascination, third-space

Interaction/flow/operation:
When a user enter the underpass, they notice the panels. Curious, the user will approach and be drawn to the center of the panels, where the stand will be. At the stand, the user's heartbeat will be collected. Then, the user can create a hand-avatar to add to the village, choosing a face, their skin colour and personality. The user will also be able to leave a recording Their virtual avatar will be projected onto the panels and walk around the physical structures, occasionally interacting with the other hands in the village.

In heart mode, the user will be able to select the avatars of other users and view their details, such as their personality and who the avatars have interacted with. While in heart mode, the user can feel a recreation of the heartbeat from the owner of the avatar the user is currently viewing and hear the recording left by the user.

Additional Information

Location:
Changi Business Park (CBP) Park underpass

Abstract:
We researched factors that can enrich the destressing process and chose 3 of them: a non-commercial third space, soft fascination and play. Our aim was to attract people to convene in the underpass and provide an opportunity for users to interact despite the time and space boundaries. We expected to make the park a more interesting place to visit, provide a moment of rest through a fun activity and make users feel more connected to the community.

Creative Challenge:
Our group had many ambitious ideas, but did not have enough experience, time or budget to fully realise all our ideas. It was challenging to give up on parts of our vision, but it was necessary if we wanted to be realistic. We also had to reevaluate our goals so we could use our remaining time most efficiently.