

The background features a detailed, high-contrast illustration in shades of grey and white. On the right, a man's face is shown in profile, wearing a checkered hat. On the left, a hand is depicted holding a scalpel. The overall style is reminiscent of a forensic or medical illustration. Large, dark red blood splatters are scattered across the lower half of the page, adding a sense of mystery and horror.

The Ripper's Trail

Unit: Collaborative Unit (PU001744)

SKYLAR LAW - 22045627

The Ripper's Trail is a Jack the Ripper Interactive Murder Board Game in collaboration with **Katharina Trappe** and **Anukriti Gupta**.

PROJECT INFORMATION

Aim

- Integrate the disciplines of physical and creative computing and criminology to provide external connections to game design
- Design an intricate murder mystery game using an Arduino-based alternative controller where players will need to use their investigative skills as they scour through multiple locations to gather clues and solve the mystery
- Develop a well-structured narrative that seamlessly incorporates real-life historical events with an additional blend of fictional storytelling
- Provide player size of any kind to have enough tasks and interactions to engage in simultaneously, while ensuring the amount of information presented does not overwhelm or confuse the player(s)
- Incorporate multiple branching endings, resulting in a varied experience for players that ensures each playthrough sequence is unique
 - By integrating multiple branching narratives, the game will offer a range of possible outcomes or routes for players to discover, this will provide a sense of unpredictability

Expected Timeline

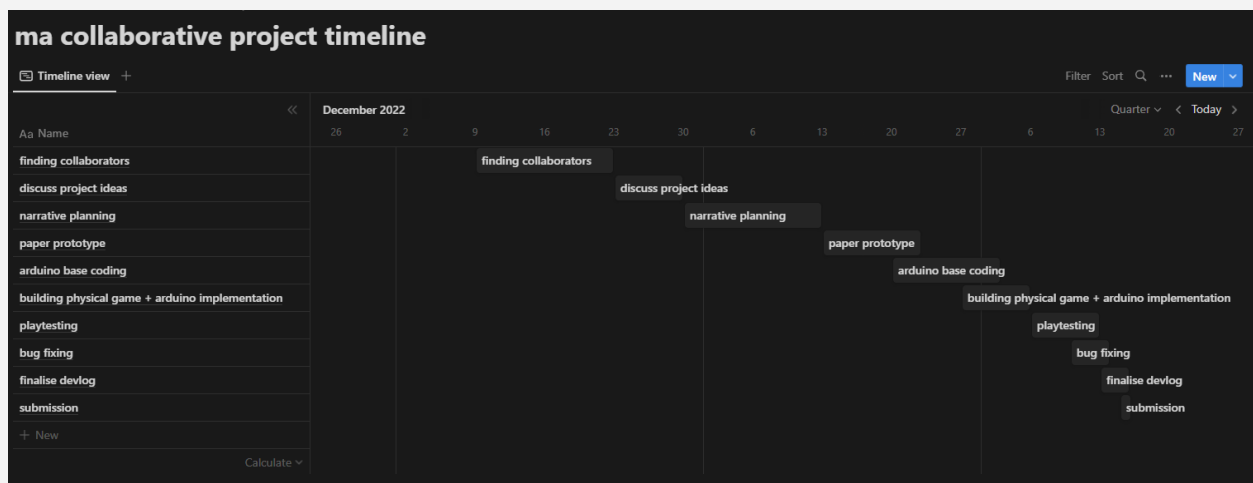


Figure 1: Expected Timeline of collaborative project

WEEK ONE - THREE

9 - 27 January 2023

Finding collaborators & inspiration

As we start off the first week of the second term, we were briefed for the unit assignment for this term. Some key takeaways from the briefing include:

- We are to work in a team
- It will be a self guided project
- Explore topics outside of game design
- Encouraged to collaborate with others outside of game design course

Given these guidelines, there are countless potential ideas to pursue for the project.

Our first task was to fill out our personal information on the *Meet and Greet Padlet Board*, in order to “advertise” ourselves for potential collaborators. My proposal was to design an alt-controller game that utilises a physical device/object that is capable of transmitting data to the game displayed on-screen.

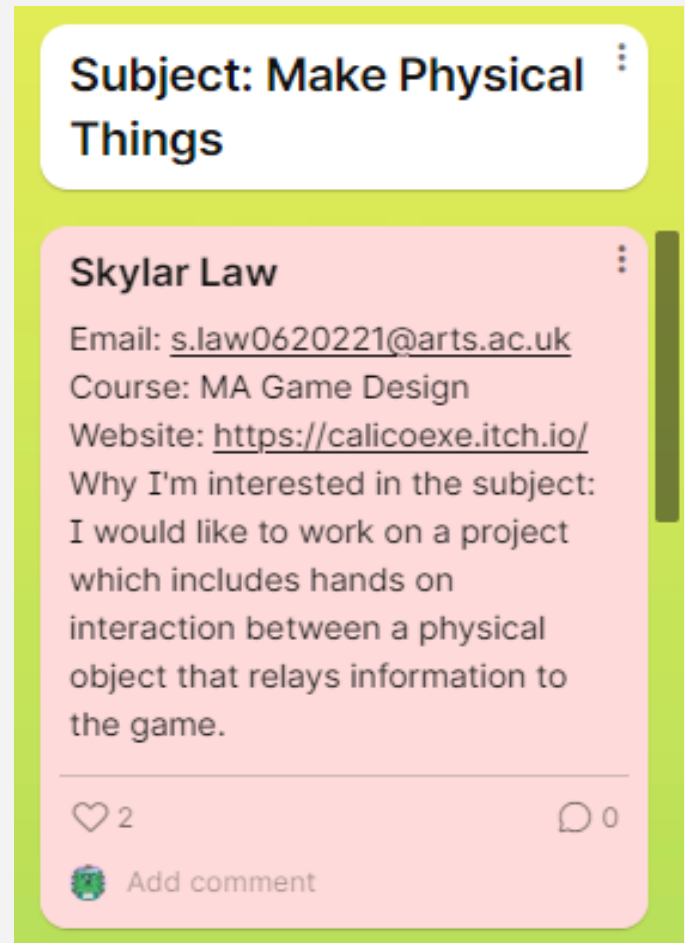


Figure 2: My entry on the Meet & Greet Padlet

Finding Collaborators

Finding someone to collaborate with is the toughest part of this project. In our Unity class, I discussed with Kathi about each of our visions of what we both wanted to do for the project individually. Kathi mentioned that she wanted to create a game with factual themes, stories that are based on a true event in history. After discussing with Kathi, I asked if she would like to team up for the project. I figured that in order to fulfil our shared creative goals for the

collaborative project, I suggested that she could incorporate factual events into the game's narrative section. With that, we have a 2-person team.

Inspiration & Initial Ideas

1 – MapFriend

The first thing that came into my mind was our lecturer, Richard Sheriff's alt-controller game which used a cycling exercise machine to control his game. With that in mind, I recalled a game that I recently came across on itch.io, [MapFriend](#) created by papercookies.

The entire purpose of this game was to explore a 3D Street POV of a Windows XP-age Google Maps lookalike, with the location you were given. Throughout the game, the locations given all correspond to real-life locations in the world. By the end of the game, you as the player discover the subtle horror of what you've been searching for.

Screenshots of game below:

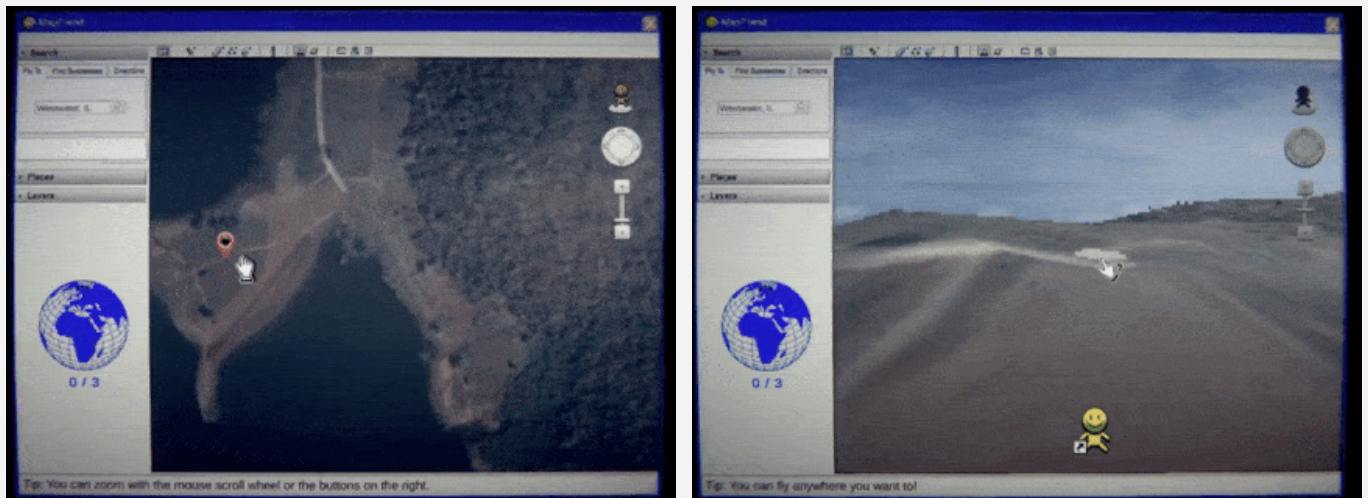


Figure 3, 4 & 5: Screenshots of MapFriend created by papercookies

With a rough idea, I sketched out a general layout of how I wanted the game to be like.

Narrative:

As a Google Street View Analyst, your role is to scrutinise footage captured by the field crew and ensure that it corresponds to the correct location. The field crew utilises various modes of transportation, such as cars, bikes, drones, and even pedestrian walking. However, as you review the footage, you begin to notice unfamiliar and disturbing visuals. As you delve deeper into your investigation, you stumble upon body cam footage that is not part of standard protocol. It becomes clear that this footage is linked to an active serial killer in your country.

Your mission is to piece together the clues you have discovered and conduct reverse image searches to identify the killer and bring them to justice.

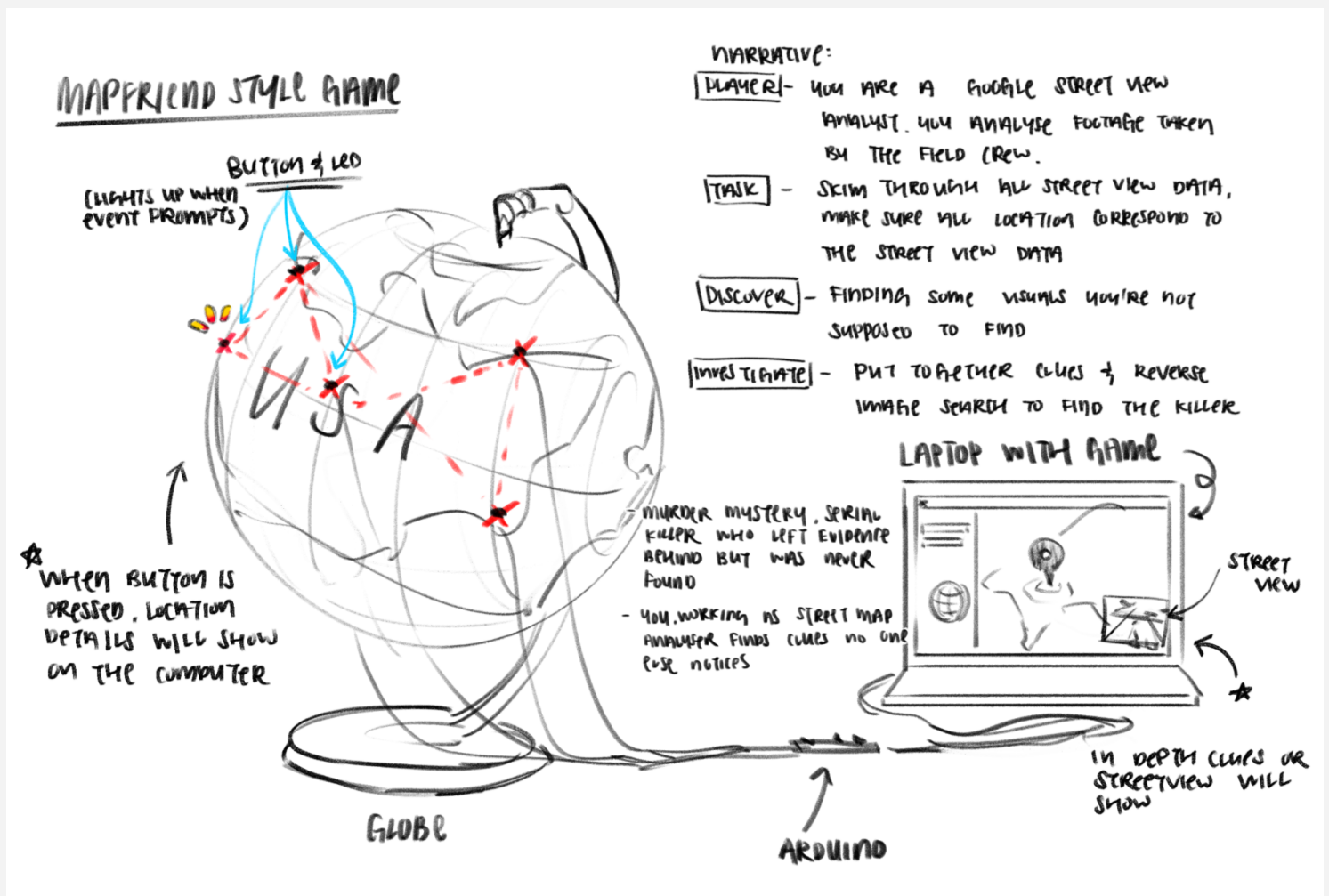


Figure 6: Ideation of the first game concept, influenced by MapFriend

Assets/software/skills needed:

- Arduino
 - Conductive wire/yarn
 - LED light bulbs
 - Buttons
- World Globe

2 – Jack the Ripper

The second idea/iteration we went through was the infamous Jack the Ripper, who roamed the streets back in 1888. I recommended exploring the Jack the Ripper case as I was reminded of an insightful video I had watched multiple times while working at my previous job. The video, produced by a popular YouTube content creator named LEMMINO, provided a thorough analysis of the Canonical Five and the possible suspects, all in a visually detailed recollection.



Figure 7: Screenshot of the YouTube video: **The Enduring Mystery of Jack the Ripper** by **LEMMINO**

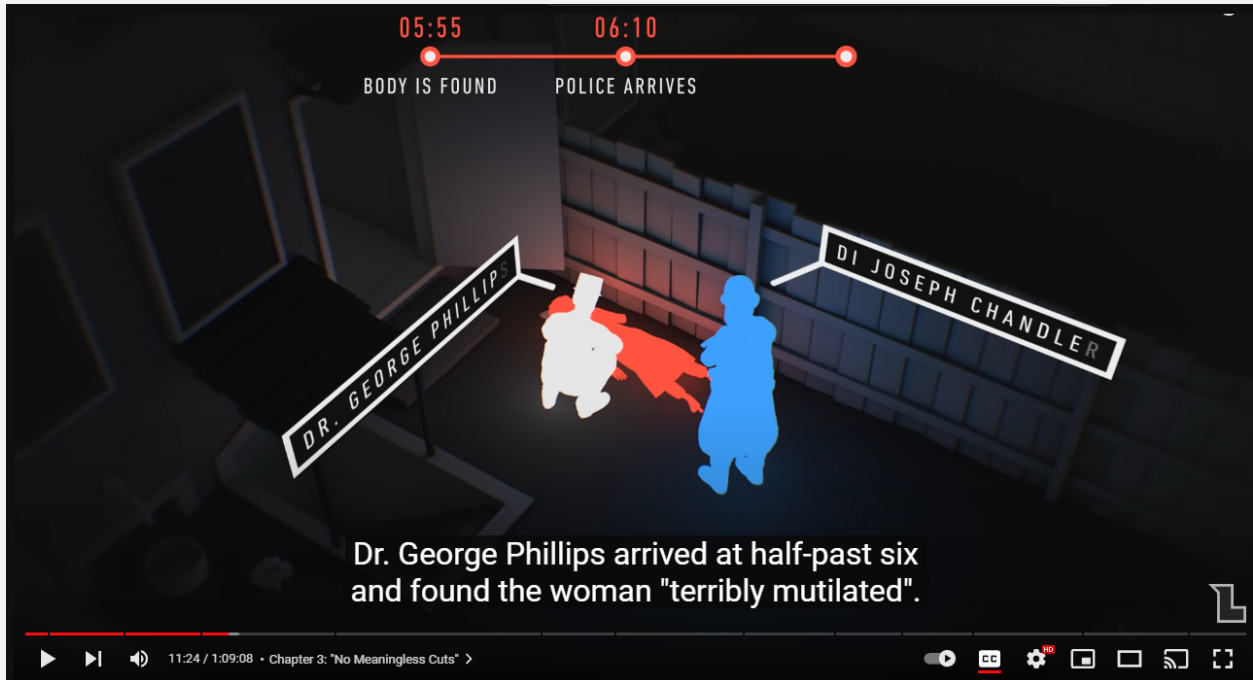


Figure 8: Screenshot of YouTube video: Anne Chapman's corpse, found by Dr. George Philip & DI Joseph Chandler

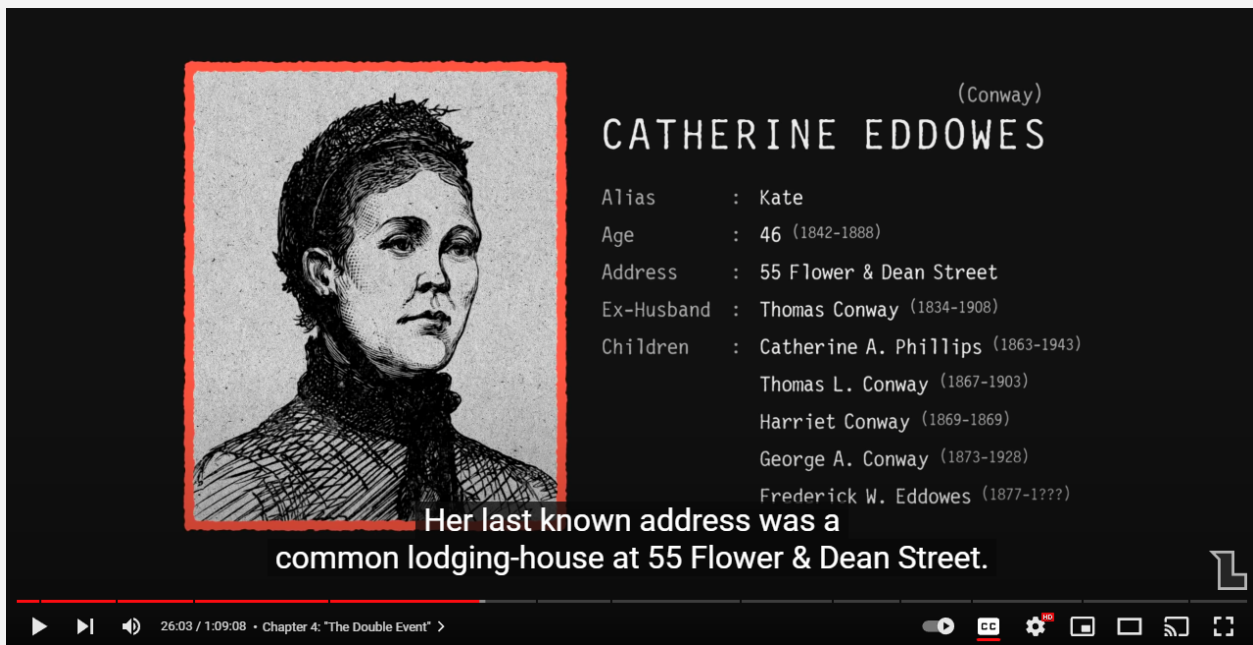


Figure 9: Screenshot of YouTube video: Catherine Eddowes' description & information

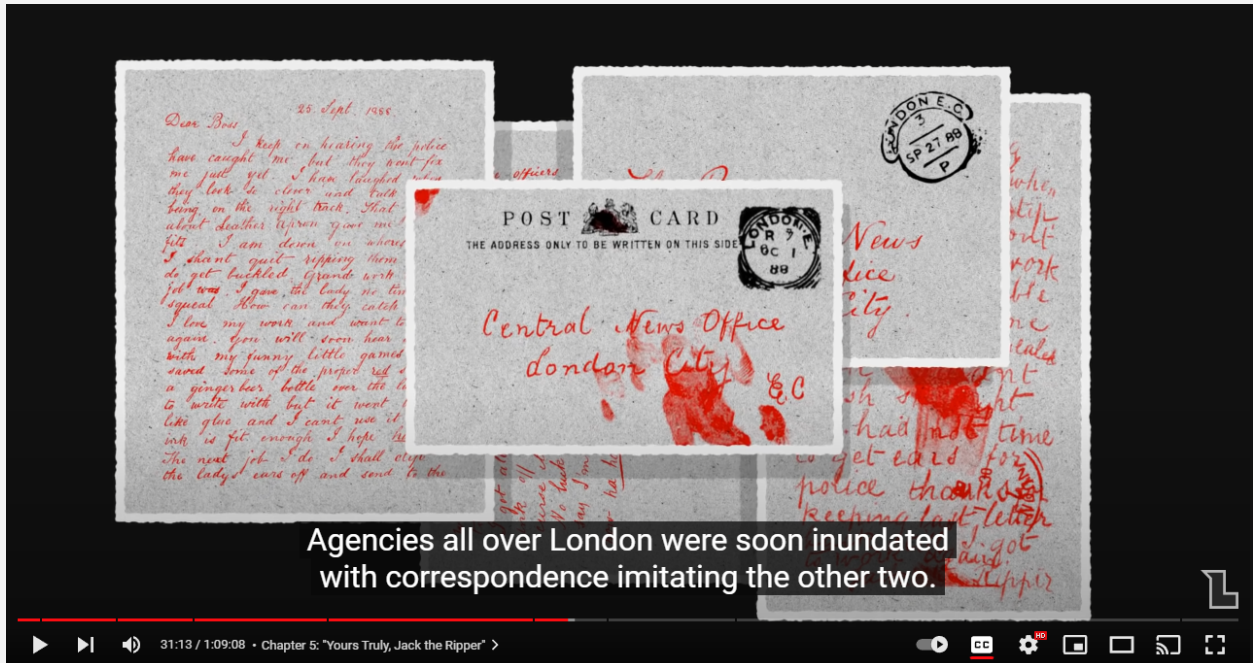


Figure 10: Screenshot of YouTube video: Blood smeared letters and postcards sent to the Central News Agency, signed by Jack the Ripper

With the amount of well-researched information, we would be able to reference the video and solidify our narrative much faster. Throughout the hundreds of years, multiple times people have tried to solve the mystery but had led nowhere. This would give us the opportunity to put our own twist of incorporating fiction into this factual horrific event.

A prime illustration of a game with a comparable style is the type of board games that involve being an investigator solving a murder mystery. We aim to create an immersive experience for players where they can fully immerse themselves in the world of mystery, grappling with clues and suspects, and feeling as if the answers are within reach.



Figure 11: Photo of a Hunt a Killer board game

Assets/software/skills needed:

- Arduino
 - Conductive wire/yarn
 - LED light bulbs
 - Buttons
- Wood cutting workshop
- AR/VR designers

3 – West/East Germany

One of the map-style concepts we discussed involved an intriguing tale about three daring brothers who successfully escaped from East Germany during the Cold War, despite the obstacles presented by the Berlin Wall. This idea was presented to us by Kathi, who had done some research on the subject.

The escape itself was nothing short of chaotic, as we learned from Kathi's findings. The first brother ingeniously used a pool mattress float to brave a freezing river, while the second brother employed a bow to anchor a rope to a building in West Germany and slung himself across. Meanwhile, the third brother waited patiently as his siblings returned with two planes they had constructed themselves to rescue him. Listening and reading up on this honestly sounded like an Uncharted game heist.

While we found this story to be thrilling and action-packed, we unfortunately felt that it lacked sufficient narrative compared to the infamous Jack the Ripper case. The Jack the Ripper case has been the subject of investigation for over a century, and the evidence and timeline are easily accessible online.

4 – Don't Be Late! (The Backup)

Don't Be Late is a quick and simple idea that I came up with that was based on my personal experiences of coming to uni every week. It is only to act as a backup plan, if all the other ideas above fail to come to conclusion or complete. As a foreigner who rarely uses the public transport system, it is hard for them to get in the habit of using one. London is one of the most efficient cities in the world with the most intricate tube system under our feets, or so they hoped to be.

One of the biggest issues I encounter with the tube system as a constant user after 6 months, is finding the right and fastest pathways to exit the tube station. As a long term resident in London, they probably figured that out by now, but not for this youngling.

Narrative:

You are a tired university student at UAL, who is late for class. Your objective is to arrive in class, on time. You will have just enough time from your starting station, Brixton, to class, but you're getting hindered by the poor underground footpath layout. You will need to master the skills of aligning the tube train doors, underground footpath corridors to ensure a timely arrival to class.

Assets/software/skills needed:

- Unity
 - Create the tube station level
 - Incorrect signs to confuse the player
- Blender
 - Creating all 3D assets in Blender
- Sound designers
 - Will require quirky sounds, crowd sounds (Similar to Persona 5 train station)

External Knowledge/Topics

Our project will incorporate external disciplines such as physical and creative computing and criminology. By incorporating criminology, we aim to add an additional layer of realism and depth to the game's mystery elements. We plan to achieve this by doing a comprehensive research on the historical murder case that was selected for the game. Our goal is to create a game that involves mystery and challenges players' thinking and observation skills.

By utilising physical computing, we aim to provide an interactive experience that fully immerses players in the game through a custom-built system that incorporates buttons, resistors, LEDs and other hardware components to create a tangible interface for players to interact with.

Meeting Potential Collaborators

In week 2, we discussed our idea with another friend, Anu. She was interested in creating something physical for her project and was looking to join a team. After some thorough explanation of the ideas that we brainstormed, she decided to join our team.

By week 3, we had to meet potential collaborators from the other MA courses. We knew we wanted a murder mystery game which involves using Arduino for the interactivity of the game. We also wanted to explore the potential in AR/VR interaction, which we would need students from the virtual reality course. 3D animation students will also be needed in order for us to create a model of the environment for the narrative.

Unfortunately, I was not able to participate in the Meet & Greet event with the other courses as I had to return home to Malaysia for the week. Kathi and Anu had presented our group and our ideas to the other students instead. Anu had created a brief description to show it to our potential collaborators.

The image is a screenshot of a mobile document titled "Collab Unit: Experimental board game". The document is displayed on a white background with a light gray border. At the top, there is a status bar showing the time "1:31 PM", the date "Tue 24 Jan", and the battery level "46%". Below the status bar, there is a navigation bar with a yellow arrow icon on the left and several icons on the right: a camera, a magnifying glass, a smiley face, and a checkmark. The main content of the document is a list of bullet points describing the game. The title "Collab Unit: Experimental board game" is in bold black text. The list of bullet points includes details about the game's theme (Murder Mystery), its design (Abstract inputs, formal output, Minimalist design), its components (2 game components, arduino based journal controller and digital game), and its gameplay mechanics. The list is organized into several sections, with some sub-sections like "Possible twists:" and "Gameplay:". The text is in a standard black font, and the overall layout is clean and professional.

1:31 PM Tue 24 Jan 46%

Collab Unit: Experimental board game

- Murder Mystery. Source material: TBD.
- Abstract inputs, formal output. Minimalist design.
- Single player experience.
- 2 game components, arduino based journal controller and digital game.
 - Game reacts to journal opening and closing.
 - Repetitive mechanic and the narrative unfolds around you. MECHANIC NEEDS TO BE SATISFYING.
- Game works in two halves, one the journal and second a fan site
 - Book contains actual case notes from the original time period.
 - Fan site/crime blog aka digital part of the game contains "new" evidence, forensics or hypothesis
 - Player end goal: complete police report given to them at the beginning?
 - Possible twists:
 - Operator of the fansite is the progeny of the killer (Jack the Ripper/zodiac killer/whatever.)
 - Operator of the fansite is the killer himself.
 - You are an agent of the killer and are trying to destroy evidence.
 - Government conspiracy gets revealed almost, fansite gets shut down right before the end of the game.

Gameplay:

- Clues can be rudimentary/abstract
 - Ex. A shape/colour that matches both on the fan site and journal points out a crucial detail in a 10 sec video clip. Bright red post it on the journal, bright red icon of the fansite blogger, bright red landmark seen in video which reveals the location of the killer.
- Player interaction limited to just opening and closing the journal?
 - **Possible Gameplay 1:** Opening and closing journal allows for game to work in stages. Stage 1 you simply read through the journal and extract whatever clues you can. Stage 2 you close the journal and explore the fansite and extract whatever you can. Stage 3 journal is open however fansite is "bugged" delivering only pieces of a video and some information. Stage 4 close the journal and map opens on the screen. Player collates all info they have at that point and figures out where or who the killer is.
 - **Possible Gameplay 2:** opening and closing the journal glitches the digital component and has it loop through 5-10 incomplete videos, audio files, articles etc. and delivers only pieces of information at a time. The journal is the

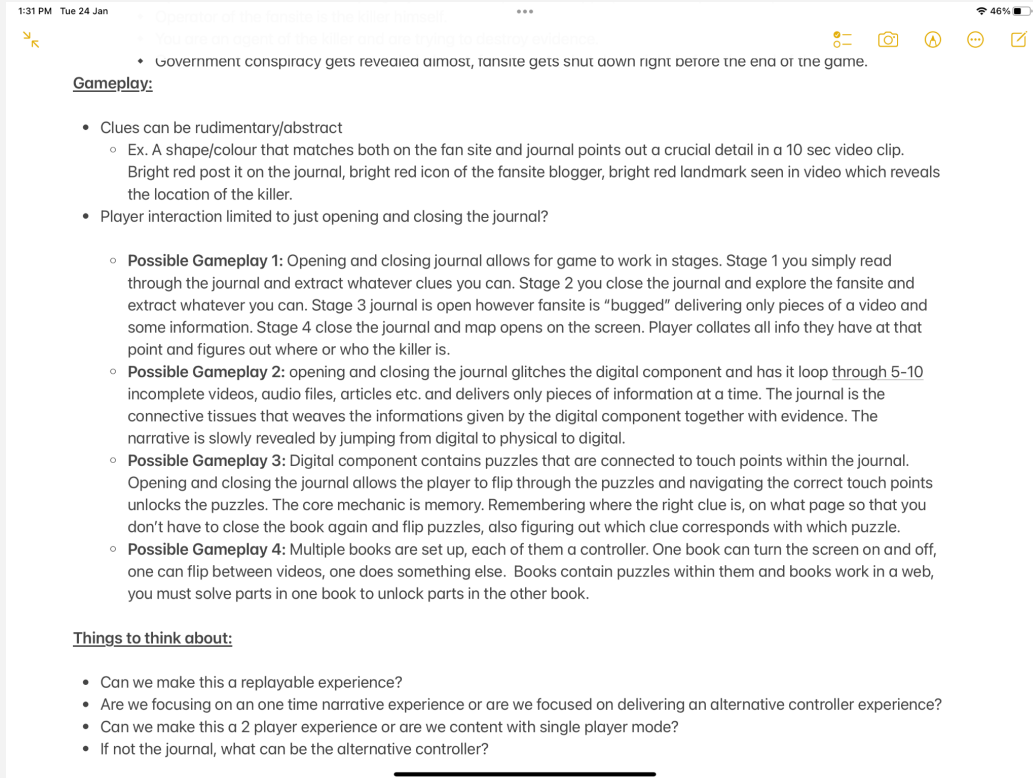


Figure 12-13: Screenshot of Anu's proposal of our murder mystery interactive game

Although I was not available in person, Kathi and Anu had shared the progress of finding additional collaborators with me on Discord. Unfortunately, the majority of the students were not interested in our idea (possibly due to the topics we wanted to work on), but we went into the event knowing that we should be as open minded as possible and be ready for changes in our ideas.

Another student approached us with a proposal to develop a game exploring urban legends about the Toilet Ghost from Korea. While the project was fascinating, we were not inclined to proceed with the collaboration with them. We were determined to create an interactive game that utilised an alt-controller, so we continued to search for individuals who shared our vision.

URBAN MYSTERY

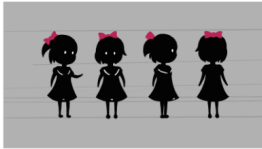

We are looking for sound designers who would like to work on a small music/ambient music on urban legends


Simple premise, just a girl and her companion dog (anti-anxiety relief) who venture through abandon/desolate city/school of (insert area here) and encounter 3 Urban legend ghosts, and they need to try and find a way to free the ghosts spirit using various clues left around the area, such as what happened to them, why are they malignant, and pieces of their backstory. The dog acts as a new mechanic to the game where it will alert you to threatening spirits, or spirits disguised as people trying to talk to you. It can also aid you to help lower your anxiety. You may also play as the dog in case of your partner being injured, as a second life to try and find a curing balm.

URBAN LEGENDS WE ARE EXPLORING

Toilet Ghost (Korea): <https://koreabridge.net/post/korean-urban-legends-7-toilet-ghost-intraman>

STYLE



WHAT WE ARE LOOKING FOR (examples)

- Creepy but very soft ambient
- Creaking footsteps/human and dog
- Heavy breathing (soft to anxiety breathing)
- Maybe very soft but not prominent looping music
- Dog panting

ANY QUESTIONS? CONTACT ME!

Cassia Edwards: cassiaedwards96@gmail.com / @cassia_edwards (insta)

Figure 14-15: Screenshot of urban legends concept

Reflection

After exploring various ideas for our collaborative project, we ultimately decided to focus on the enduring mystery of Jack the Ripper. As we delve deeper into the project, we have assigned specific roles and responsibilities among our team. Kathi will serve as the narrative designer, while Anu will assist with narrative design and create all of the game's assets. I will take on the role of programmer. Admittedly, programming is an area where I lack proficiency, but I see this project as an opportunity to improve my skills in this domain.

WEEK FOUR - SIX

30 January - 17 February 2023

Research & Narrative Planning

During the fourth week, we dedicated most of our time to searching for additional collaborators to join our team, but unfortunately, we were unable to find any suitable candidates. We were initially concerned about meeting the unit's requirements without

external help. However, after discussing our concerns with David, he reassured us that collaborating with outside assistance on an external topic would suffice.

The majority of our time over the fourth and fifth week was spent in the Creative Technology Lab (CTL), where we attended several beginner-level workshops to improve our skills in physical computing and creative coding. The workshops we took part in were Beginner Physical Computing (Arduino) by Joanne Leung, Introduction to Creative Computing in P5.js as well as Generative Design with Creative Coding with Michael-Jon Mizra. Although these workshops were only beginner level, we followed up with Joanne, who was an expert in all things physical computing. With Joanne's expertise in physical computing, we shared our game concept and plan to create an interactive murder mystery board utilising LED lights to track player progress.

During the Beginner Physical computing workshop, we gained knowledge on constructing an LED light system utilising NeoPixel LED strips. As we required an LED light to demonstrate a progression bar in our project, this newfound skill could prove to be valuable.

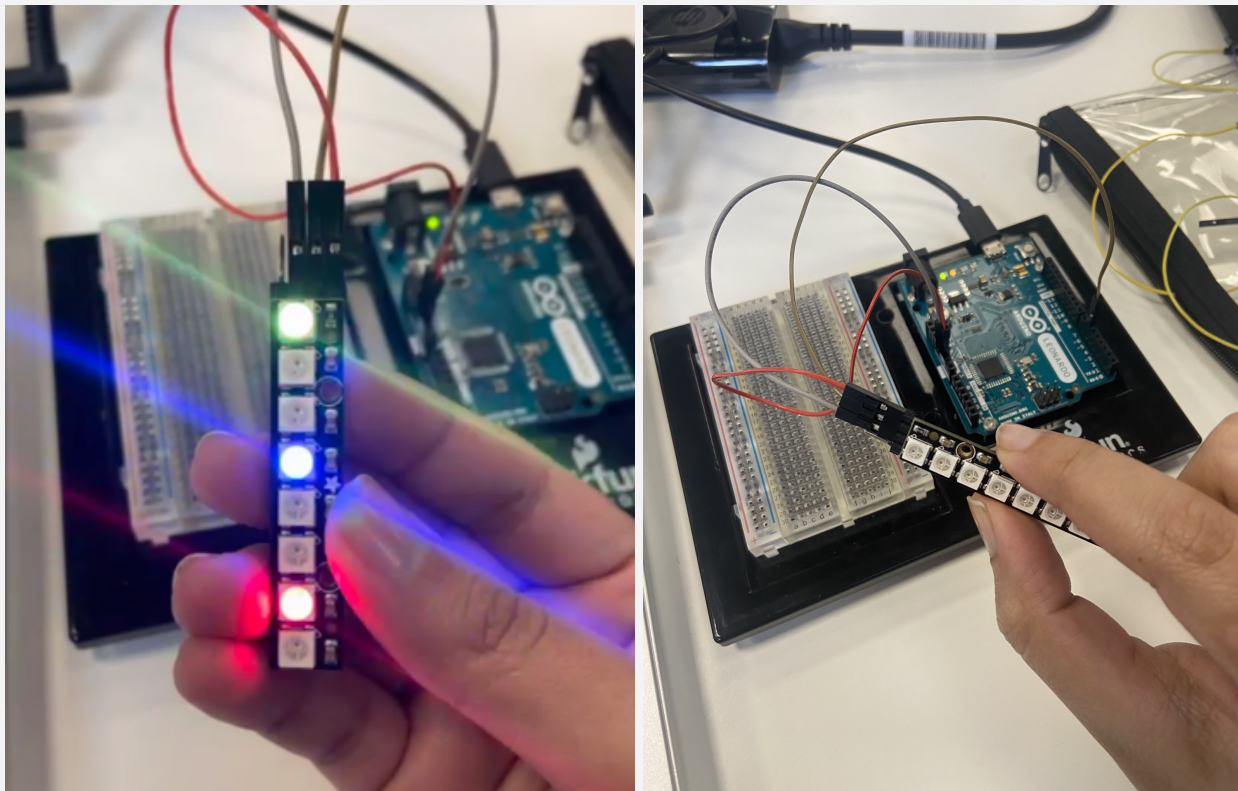


Figure 16-17: Beginner Physical Computing Workshop, working on NeoPixel LEDs

In addition to Arduino, the workshop also introduced us to other essential tools like breadboards, which can help extend the positive (volts/digital) and negative (ground) charges. However, it's important to note that unlike digital slots, analog slots cannot be extended to the breadboard. Therefore, while designing our interactive game, we must consider the number of analog slots required.

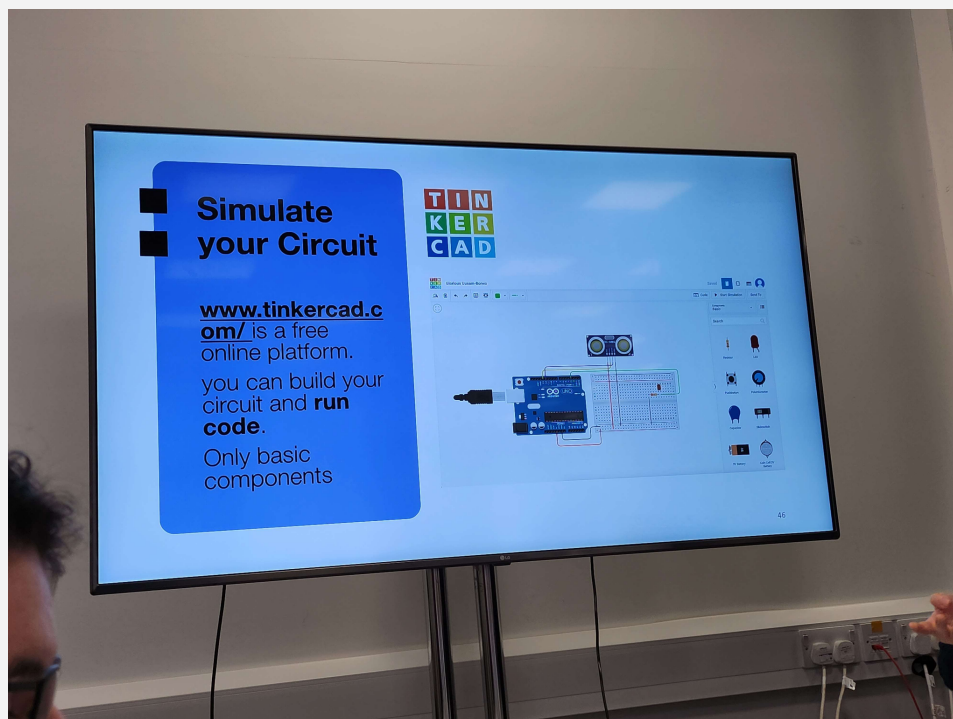
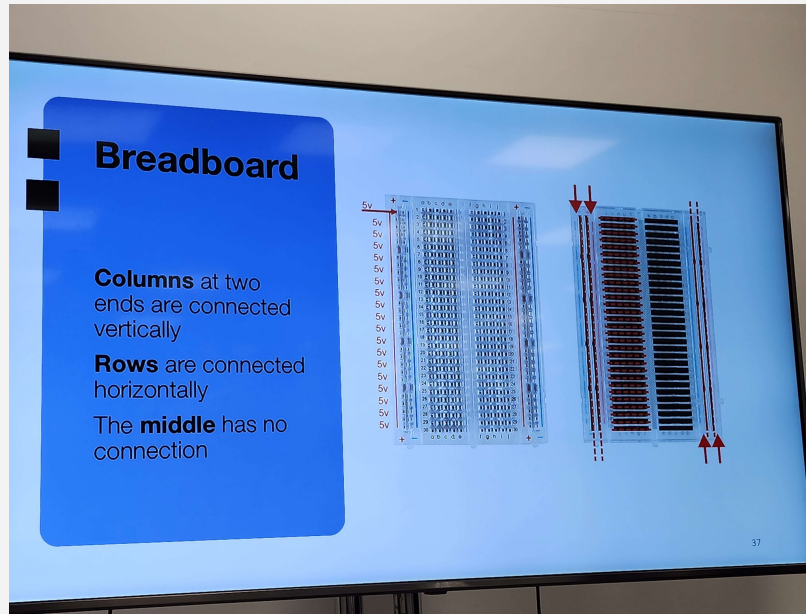


Figure 18-19: Photos from our workshop sessions with Joanne Leung

Following our discussion with Joanne, she provided a quick diagram illustrating how we could incorporate Arduino systems into our game board. She proposed the use of conductive yarn, which was readily available as a resource in the CTL. This would enable us to simulate the act of connecting clues and evidence. We could also integrate LEDs that would light up automatically once a set of information has been connected.

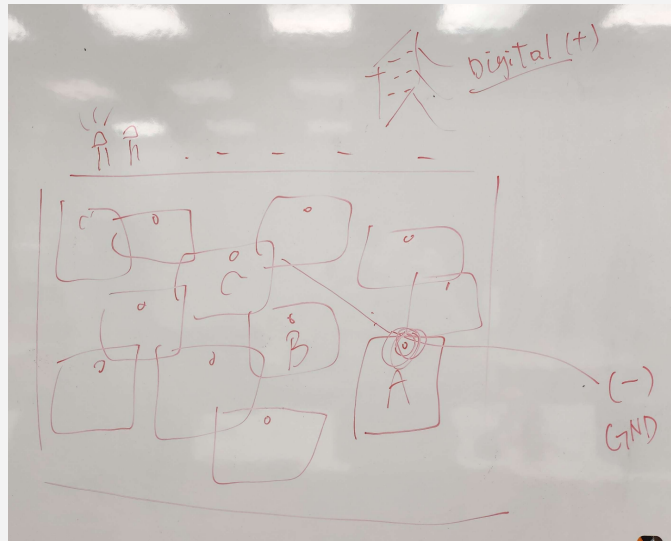


Figure 20: Photos of Joanne's draft diagram of the game board

Team Communication

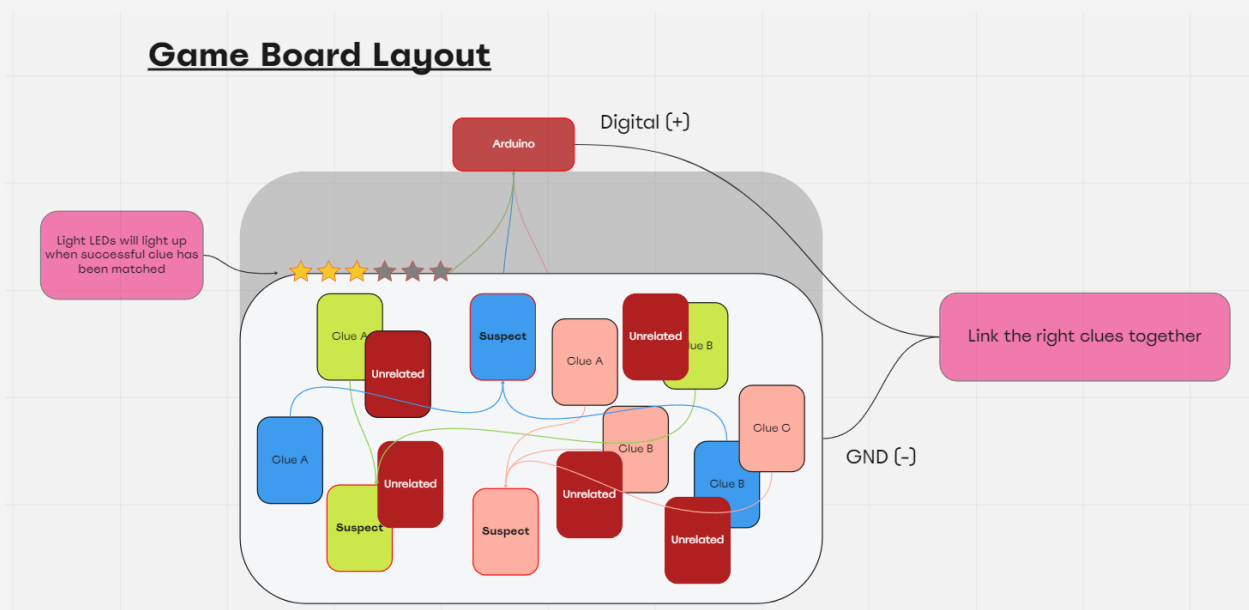


Figure 21: Screenshot of game board layout on Miro

Using the diagram as a reference, I replicated it on Miro to create a digital copy of the overall plan for our game board. At the same time, we established a Discord server to communicate outside of the university. To streamline our workflow, we created various Discord channels for specific messages or media. This ensured an organised workflow, with each channel serving its purpose and avoiding overlapping of different topics.

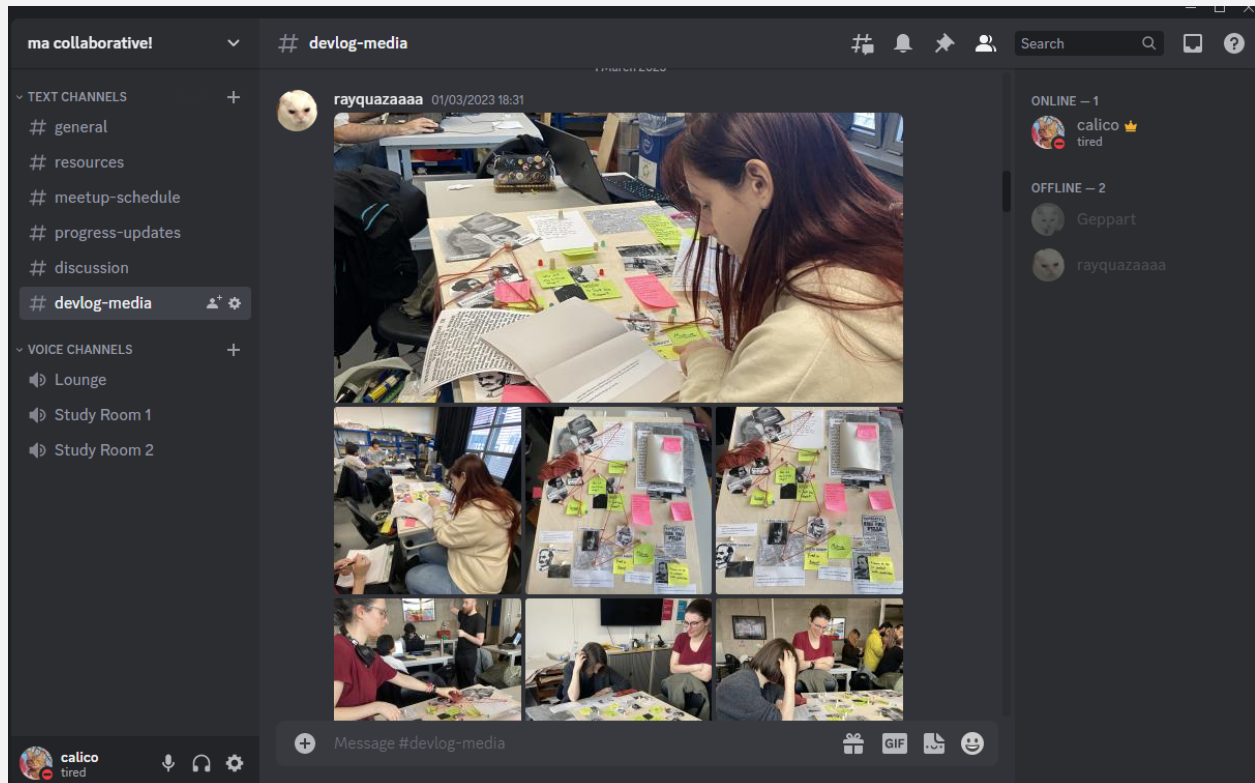


Figure 22: Screenshot of Discord server with separate channels

We do not intend to work on the project virtually since our game is rather experimental and requires a hands-on approach. We spent the majority of our time in the CTL, tinkering with Arduinos and brainstorming our narrative on whiteboards.



Figure 23-24: A little team workshop selfie (with frog filters)

Narrative Designing & Planning

With a rough concept for our game board in mind, we began to craft the narrative for our game. Given the notorious history surrounding the unsolved case of Jack the Ripper, we knew that there was limited evidence available to work with. Much of the evidence that had been found proved to be largely inconclusive. We relied heavily on references from LEMMINO's video essay, which was thoroughly cited and included links to resources on his website. This allowed us to easily backtrack and find primary documents from the 1800s.

To begin, we constructed a comprehensive timeline and gathered all available information on the victims, potential suspects, and methods of killing. However, we discovered that Jack the Ripper's signature method was not consistently used on all of the female victims. In some instances, the killer's near brushes with being discovered could have resulted in rushed and

sloppy murders. Having established the factual basis of the timeline, we were able to more easily incorporate fictional elements into the narrative, giving us greater creative freedom to craft our own story.

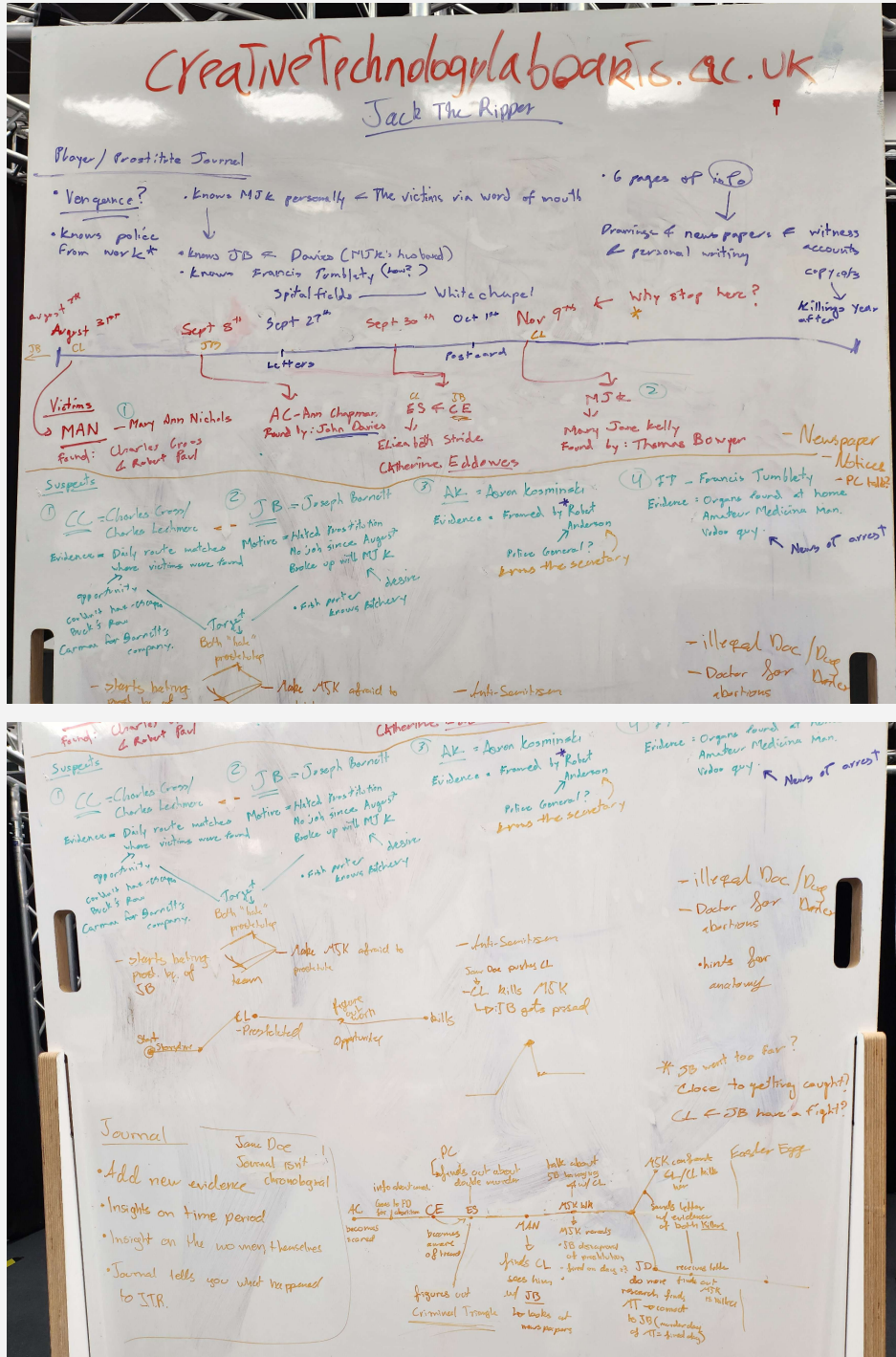


Figure 25-26: Image of our physical whiteboard plans of the narrative design of the game

Narrative Design

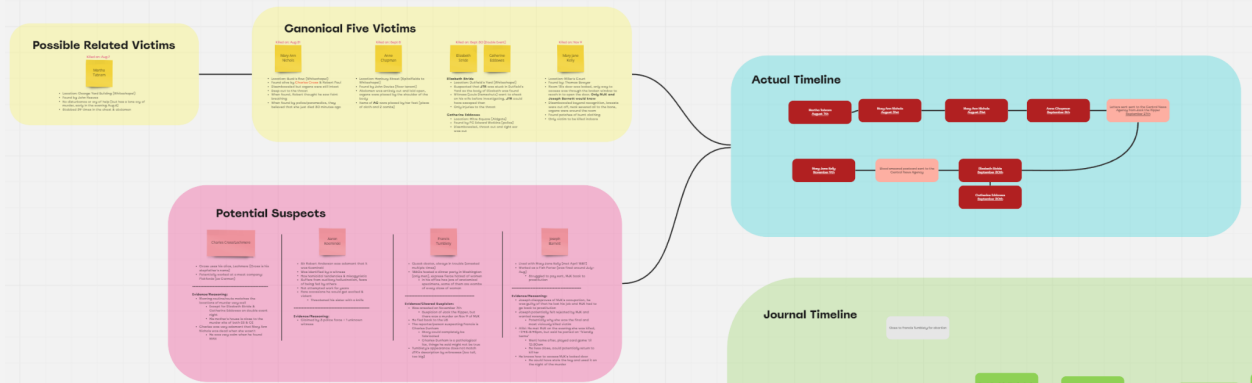


Figure 27: Screenshot of narrative design on Miro

To add more depth to the narrative, we came up with the idea of Andromeda Smith/Lady Scarlett, an external character who was a fellow prostitute and knew some of the victims during the time of the murders. As the killings occurred, Andromeda would keep a journal, chronicling the events and the possible suspects until she eventually narrowed it down to the actual culprits. Players would need to read her journal, find clues, and use them to connect the evidence and information on the board.

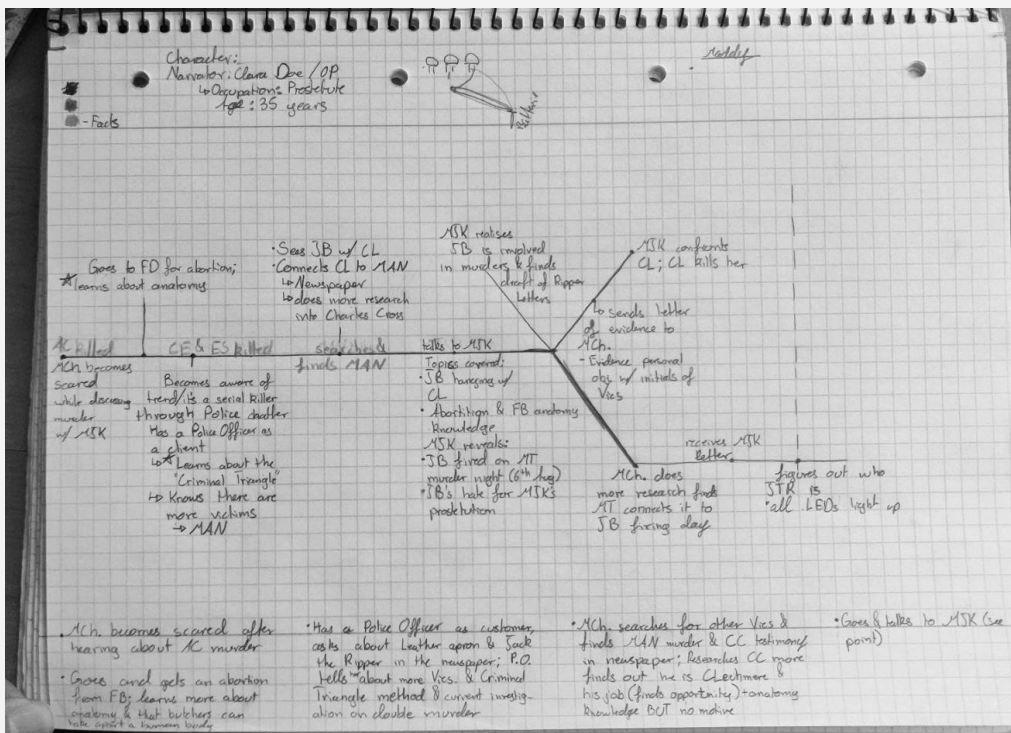


Figure 28: Physical copy of new timeline created by Kathi & Anu

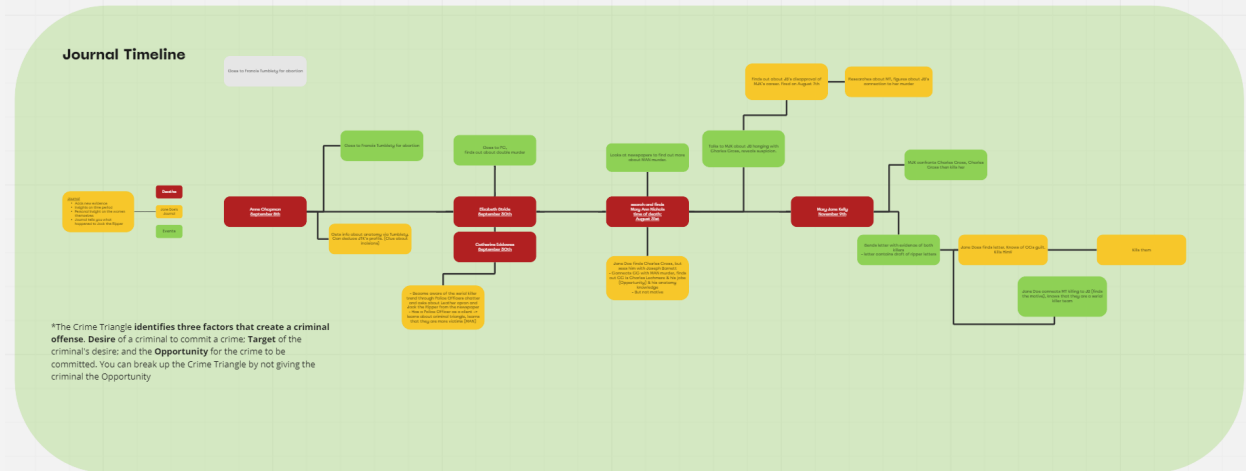


Figure 29: Screenshot of Journal Timeline on Miro

As Kathi and Anu were busy establishing the connections and timeline of the Jack the Ripper case, I took the opportunity to tinker with Tinkercad. With my limited knowledge of Arduino coding, I quickly constructed an LED light system using tutorials I found on Arduino documentations and YouTube.

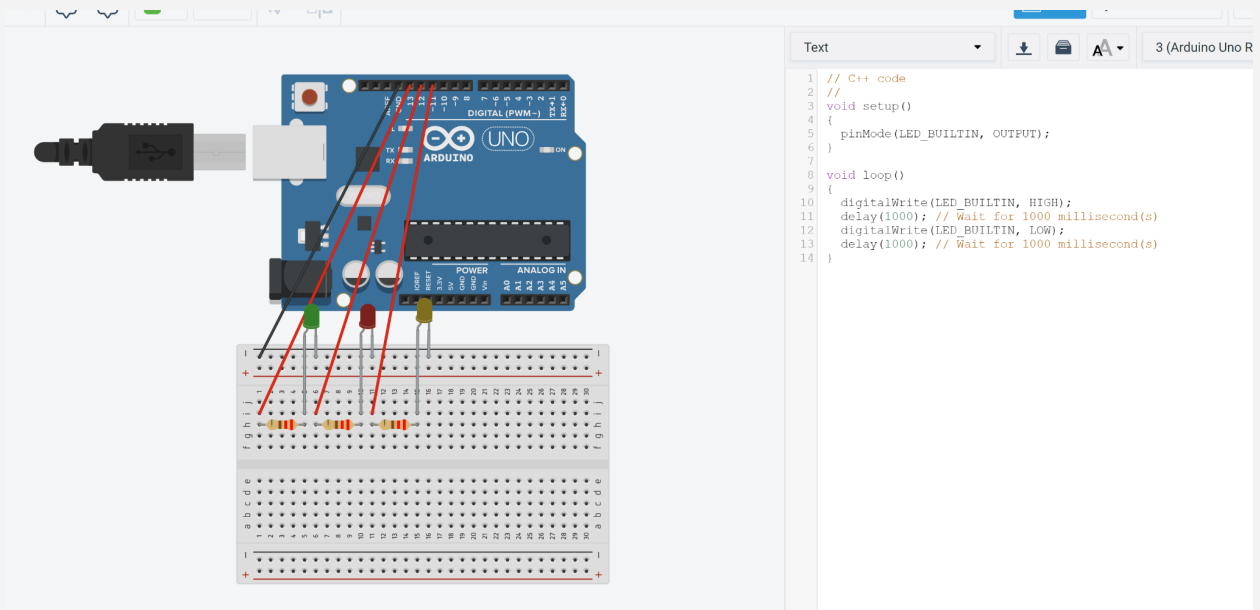
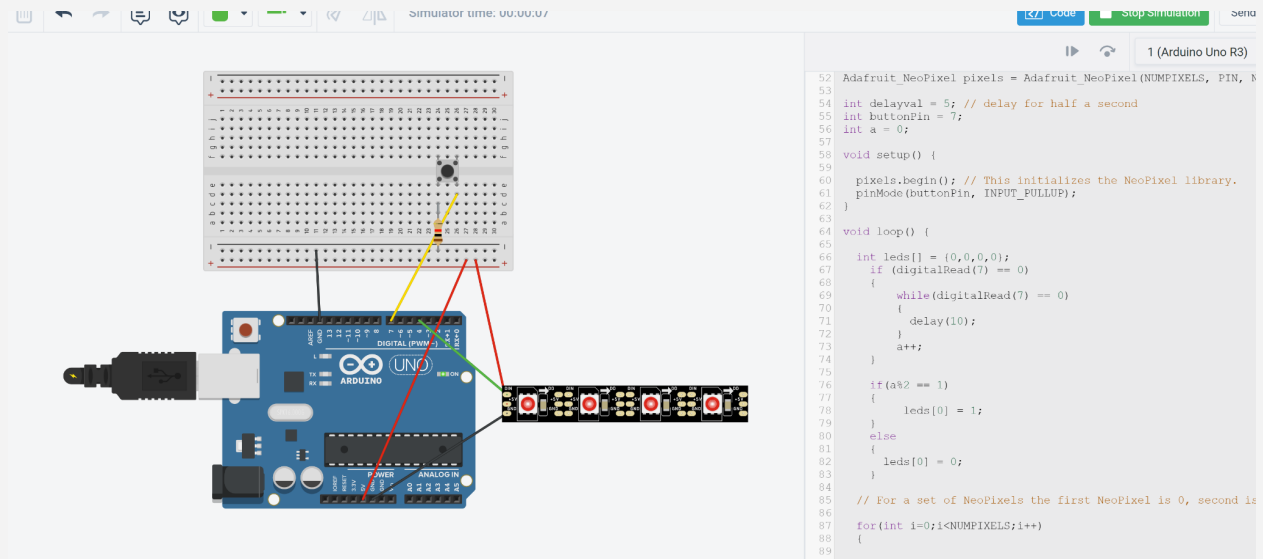


Figure 30: Tinkercad screenshot of simple LED circuit

At this time, I consulted with Richard about his experience with Arduino and requested his assistance with creating a system that would change the LED lights when a button is pressed. My idea was to use NeoPixels as a progress bar for the board.



Visiting Tate Modern

While in the midst of our narrative design planning, we took a school trip to Tate Modern where we stumbled upon an intriguing art piece titled 'a preponderance of aboriginal blood 2005' by Judy Watson.

The piece consisted of 16 papers stained with blood and was originally part of an artist's book that confronted the history of discrimination against the Aboriginal and Torres Strait Islander people. We learned about the racism that prevented Aboriginal people from voting in Queensland, and Judy's view that the original documents. Though the original documents were archived nicely, it did not accurately portray the terrible impact they had on the minorities on the island.

Despite the dissimilarity between our subject matter and the bloodied papers we saw at Tate Modern, we found inspiration in the methods used to create those works. We intend to apply similar techniques to craft postcards with blood stains and smears, just like the ones allegedly sent by Jack the Ripper to the Central News Agency. This will enhance the authenticity and immersion for the players as they unravel the mystery.

Judy Watson 1959
Born and works Australia. Waanyi people

a preponderance of aboriginal blood
2005

16 etchings on paper

These etchings are pages from an artist's book confronting the history of official discrimination against Aboriginal and Torres Strait Islander peoples. Its sheets include copies of electoral registration forms from the Queensland State Archives. These classified people according to race. Unless they had a white parent, Aboriginal people were not entitled to vote in Queensland until 1965. For Watson, the 'beautiful and old fashioned' aesthetic of the archival documents cannot be separated from their 'horrible content', designed to deny Aboriginal and Torres Strait Islander peoples the civil rights extended to the rest of the population.

Tate and the Museum of Contemporary Art Australia, with support from the Qantas Foundation 2015, purchased 2016 P82511

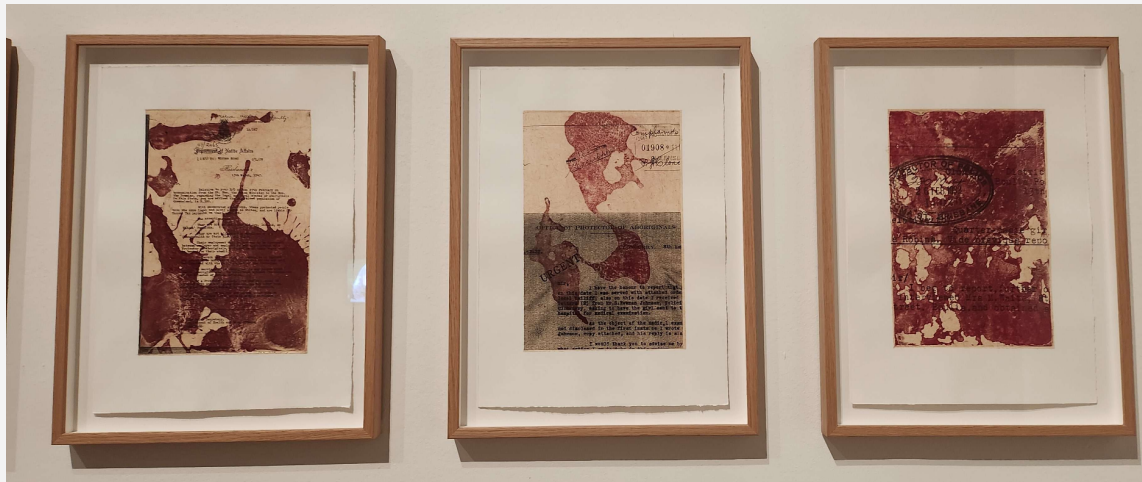




Figure 32-35: Photos of the 16 etched paintings by Judy Watson in Tate Modern

Additional Workshops

As previously mentioned, I also participated in Creative Coding workshops with Michael-Jon Mizra utilising P5.js. While not directly related to Arduino programming, it was beneficial for me to attend Joanne's subsequent workshop on Arduino: Interactive Generative Design. While Michael's workshop wasn't a prerequisite for my Arduino programming journey, it provided me with useful skills that could be applied to future projects. I can foresee using JavaScript to create a different type of interactive game using the skills I've learned.

Below are some tasks I've done in Michael's workshop:

```
sketch.js Saved: about 1 month ago Preview
```

```
1 let xPosition = 200;
2
3 function setup() {
4   createCanvas(400, 400);
5 }
6
7 function draw() {
8   background(220);
9   circle(xPosition,100,100);
10  circle(xPosition * 2,200,100);
11  circle(xPosition,300,100);
12 }
13
```

The preview window shows a gray rectangular background with three white circles. Two circles are positioned vertically on the left side, and a third circle is partially visible on the right edge.

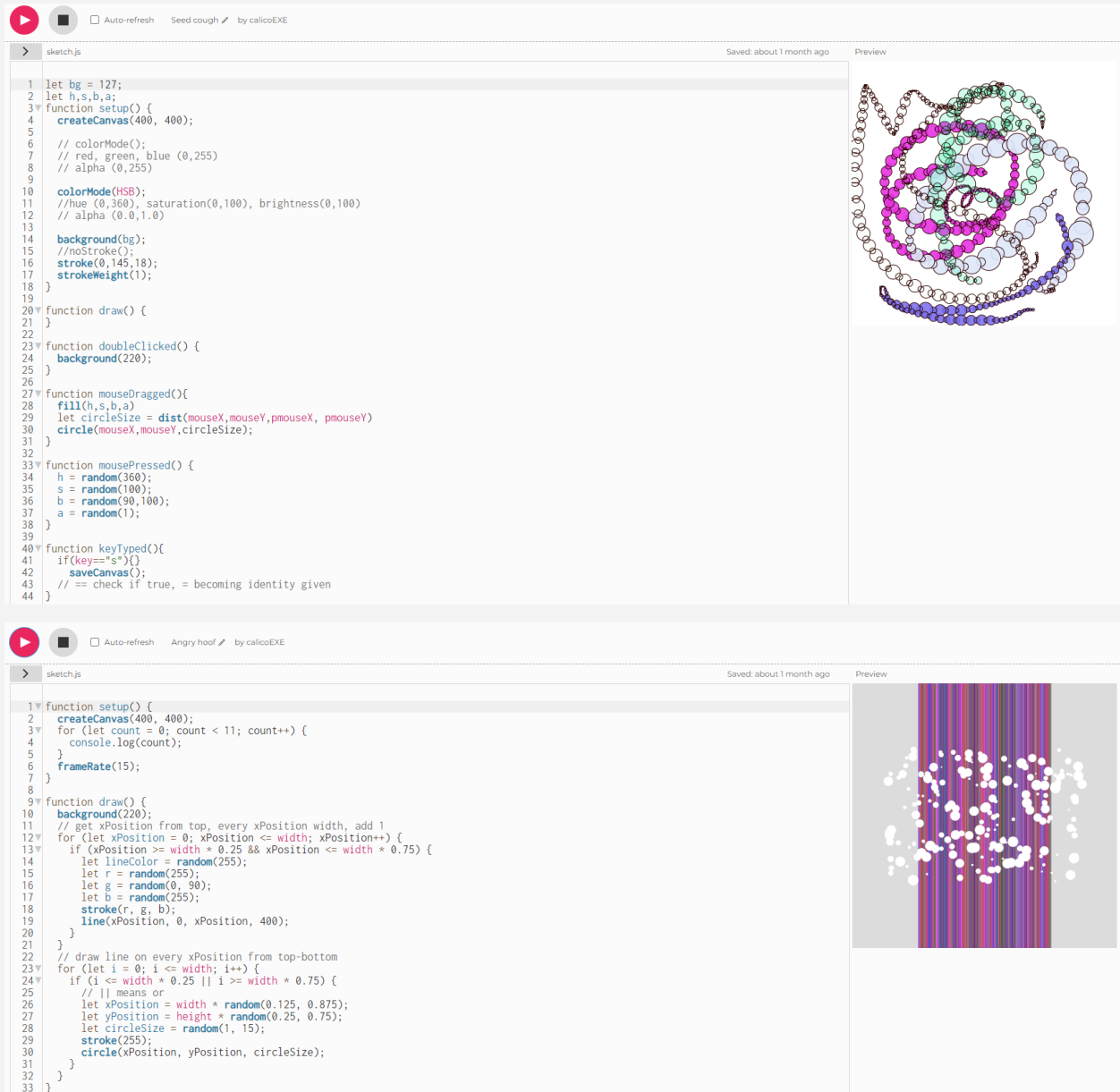


Figure 36-38: Screenshot of work done in Michael's workshop

Reflection

During the past few weeks, our team has spent significant time working on the game's narrative. While I wasn't as heavily involved in narrative planning as the others, I had the opportunity to learn Arduino programming extensively for our project. Although challenging, these experiences were also rewarding. It has the potential to enhance my personal portfolio and future projects. I look forward to continuing to refine these skills beyond this project.

WEEK SEVEN - TEN

20 February - 15 March 2023

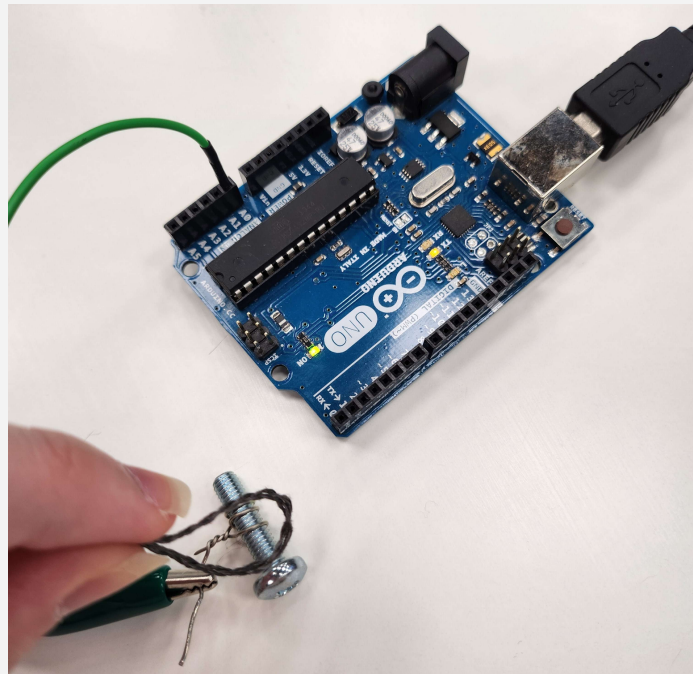
Paper Prototype, Physical Game Building, Playtesting & Finalising

Most of the physical game development took place over the course of these four weeks. While it was helpful to have a clear understanding of the narrative before making any concrete decisions about the physical game's design, it did create some time pressure. As a result, many changes to both the narrative and the game itself were made during this period.

Version 1 – Interaction using Conductive Yarn

The first version of our physical game was created with the idea of using a conductive yarn to connect the evidence together. This would give the interactivity of “solving a mystery on a corkboard full of evidence” to the player. My first task (given by Joanne) was to create a working Arduino system where the internal Arduino LED would light up when the yarn has touched the conductive screw that was connected to the Arduino via a crocodile clip.

With the internal LED working, I then added an external LED with the use of a breadboard and resistors to the Arduino, having a separate working LED instead.



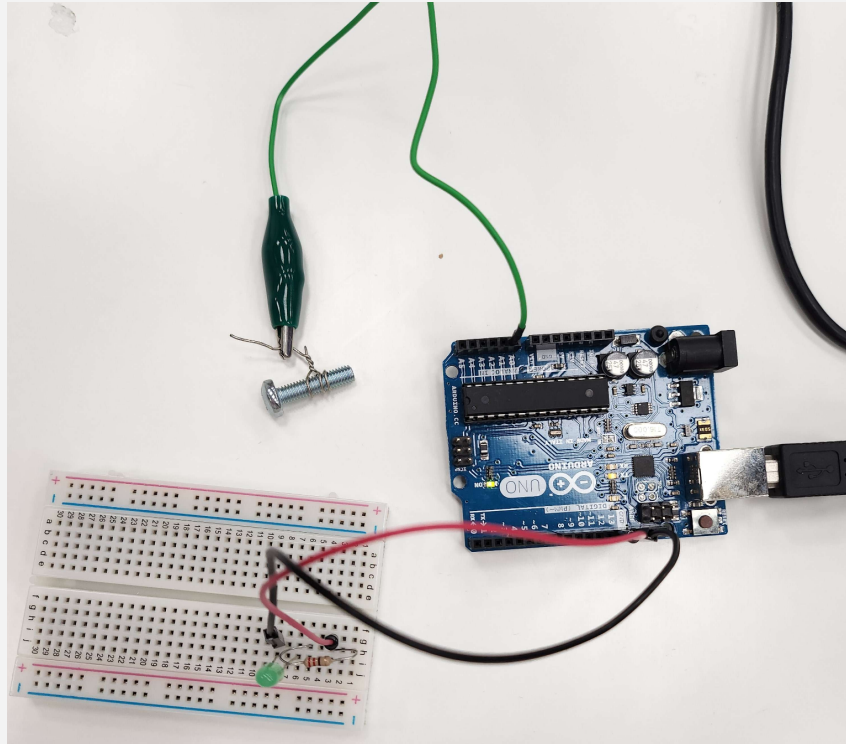


Figure 39-40: Interactive with conductive yarn Arduino board layout

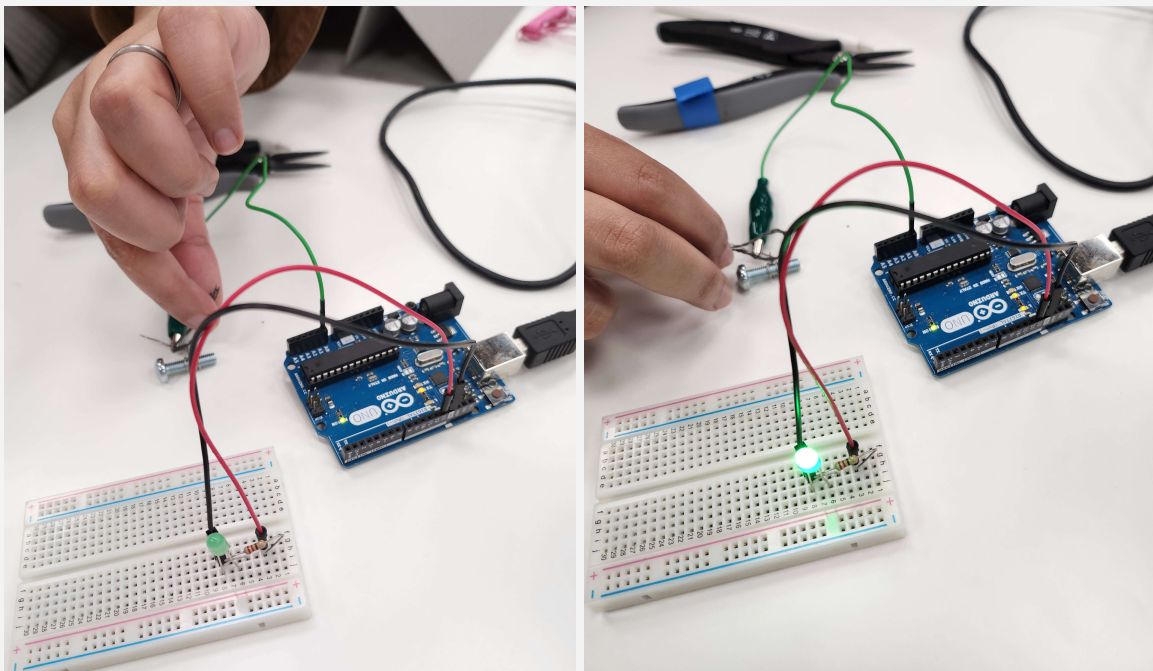


Figure 41-42: Working LED after conductive yarn (held by Anu) has touched the conductive screw connected to the Arduino

Knowing that our concept is working properly, I added two additional LEDs linked to individual screws for us to test it out.

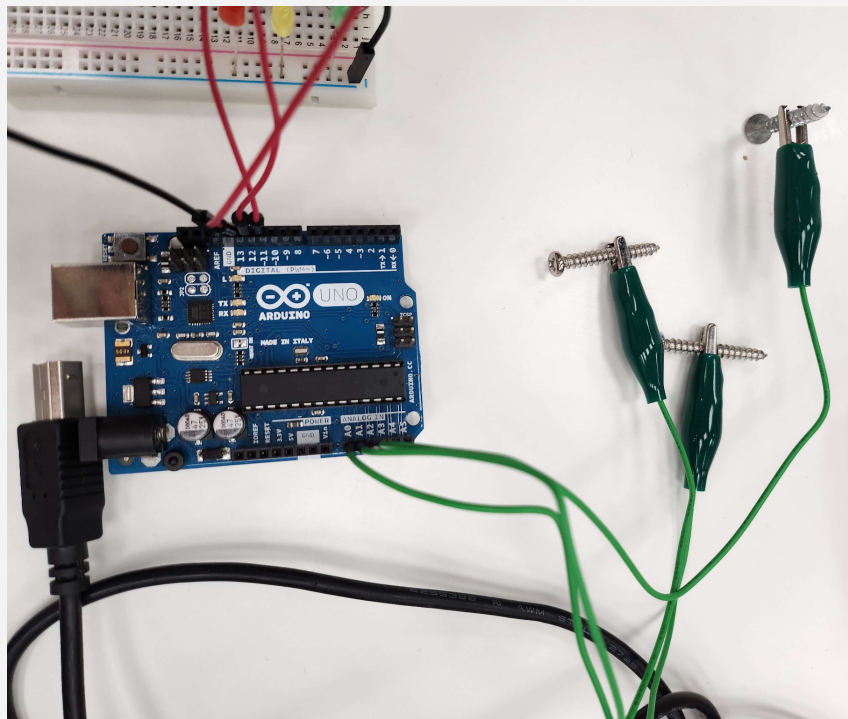
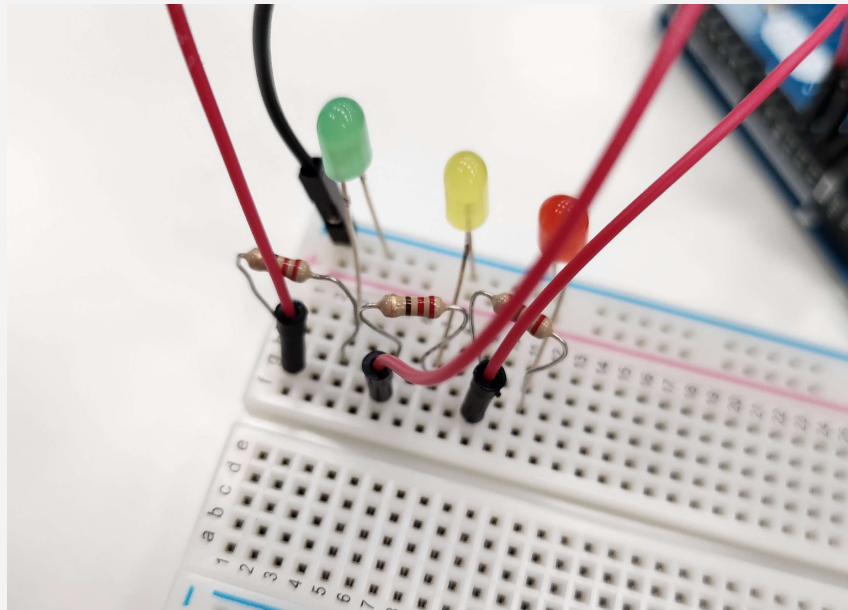


Figure 43-44: Additional 2 LED and 2 individual screws

While this was ongoing on the Arduino board, I've made sure to change the code on the Arduino IDE to cater to the additional LEDs.

```

JTRBoard Version 1
conductive yard, human touch as charge

#include <ADCTouch.h>

#define TOUCHPIN A0
#define TOUCHPIN2 A1
#define TOUCHPIN3 A2 // add for more led

#define RESOLUTION 100

#define SMOOTH 100 // determine how many readings are stored for smoothing
float multiplier = 1.2; // determine when the sensor is understood as "ON"

int previousReadings[SMOOTH]; // smooth data a little: the last readings
int previousReadings2[SMOOTH];
int previousReadings3[SMOOTH]; // add for more led

int currentIndex = 0; // used for cycling through the array
int currentIndex2 = 0;
int currentIndex3 = 0; // add for more led

int reading; // the latest reading
int reading2;
int reading3; // add for more led

// calculate the average of the previous readings
int average1(){
  unsigned long sum = 0;
  for(int i = 0; i < SMOOTH; i++){
    sum += previousReadings[i];
  }
  return sum / SMOOTH;
}

int average2(){
  unsigned long sum2 = 0;
  for(int i = 0; i < SMOOTH; i++){
    sum2 += previousReadings2[i];
  }
  return sum2 / SMOOTH;
}

int average3(){ // add for more led, remove when using arduino mega
  unsigned long sum3 = 0;
  for(int i = 0; i < SMOOTH; i++){
    sum3 += previousReadings3[i];
  }
  return sum3 / SMOOTH;
}

```

Figure 45: Preview of JTR Board Coding Version 1
 You may find full version 1 code here: [JTR Version 1](#)

One challenge we faced with this concept was that the player would have to continually touch the yarn in order to keep the LED light on, as we humans act as a low voltage charge rather than a battery. This was problematic as we intended for players to release the yarn they're holding after the correct connections had been made. An additional issue that I noticed was that having players hold/complete circuits could be a safety hazard due to potential electrical shocks after they had completed the game.

After considering the issue of the player needing to constantly touch the conductive yarn to keep the LED light on, we explored an alternative solution of using a small watch battery (CR3032 battery) to keep the light on even after the player released the yarn. However, upon discussing this idea with Joanne, we realised it posed significant safety risks, such as the potential for the battery to explode, the conductive yarn to burn and cause a fire, or even the player to be electrocuted. Therefore, we decided to abandon this idea in favour of finding a safer solution.



Figure 46: Joanne's installation in the CTL

While we were preparing to leave the CTL for the day, I noticed this little time warp installation created by Joanne. I noticed the way that she placed the LEDs which eventually gave me an idea on creating the wood base for the project.

Version 2 – Connecting via Crocodile Clips (Voltage Cap)

Another iteration that we came about which involved using crocodile clips and resistors to cap the voltage for each pin. However, since I was not familiar with working with resistors, identifying the appropriate one proved to be a challenge. The process required a great deal of calculation, so we consulted with Joanne for guidance. Regrettably, she alerted us to the potential safety hazards associated with this method. If the clips were connected incorrectly, it could cause a short circuit, and if too much power was applied, the resistor could overheat and potentially explode.

With this response from Joanne, we once again decided to abandon this idea.

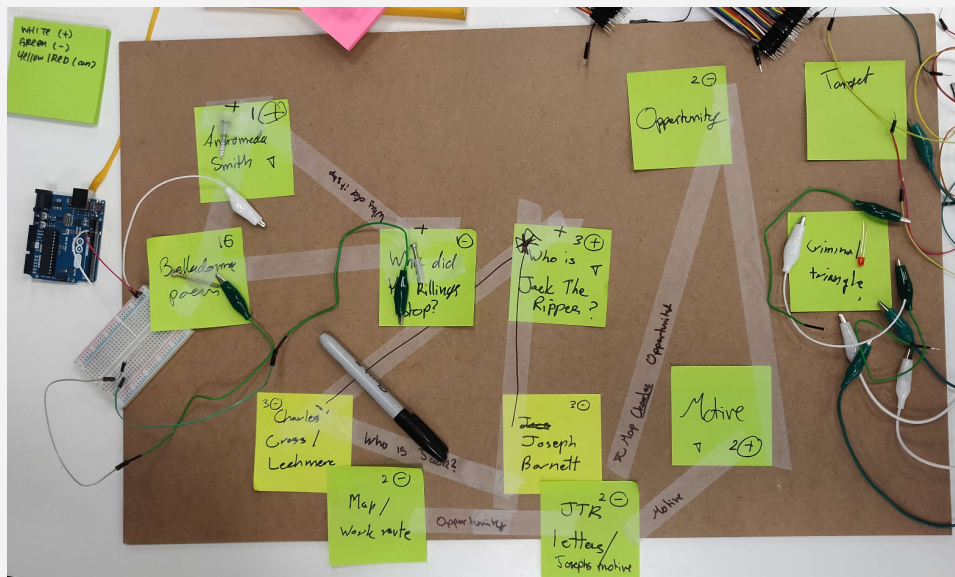
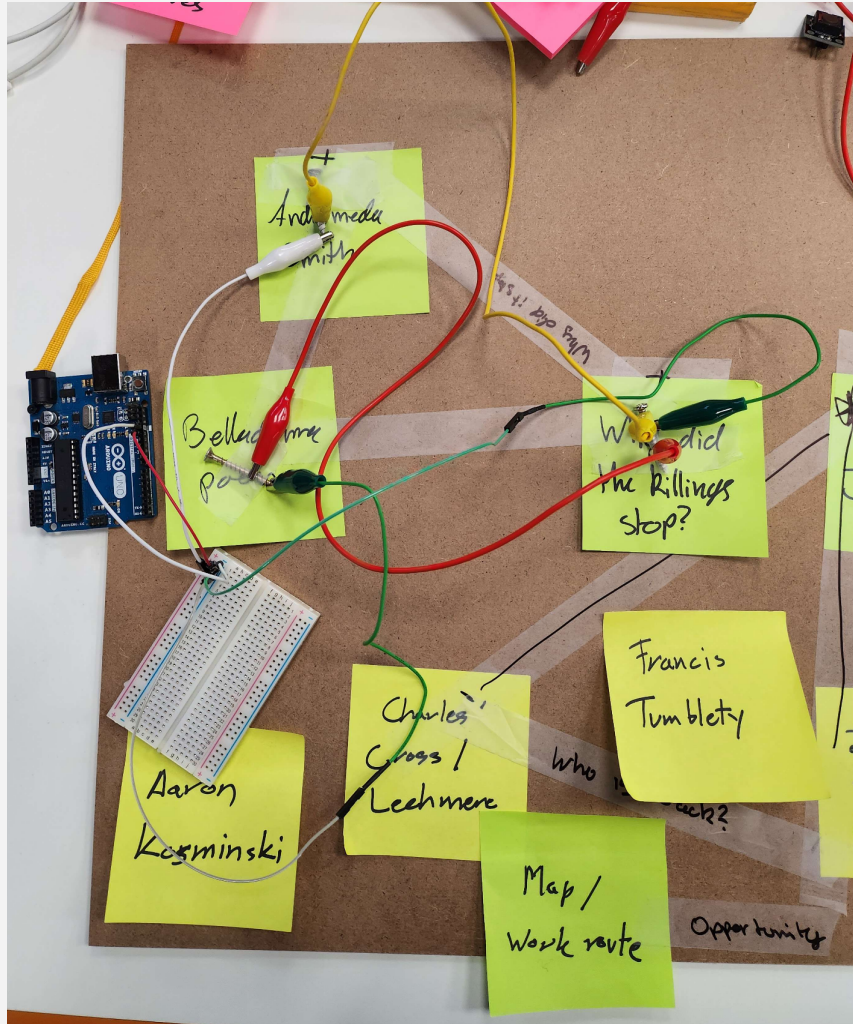


Figure 47-48: Preview of JTR Board Version 2

Even though we decided not to proceed with the idea, I took the initiative to draft a preliminary concept of how the code would have looked before approaching Joanne. As mentioned, we did not proceed with this idea due to safety hazards, therefore, I did not finish the code.

```

JTRBoard Version 2
voltage cap (does not work, safety issue)

#define TOUCHPIN A0
#define RESOLUTION 100
#define SMOOTH 100

float multiplier = 1.2; // determen when the sensor is understood as "ON"
int previousReadings[SMOOTH]; // smooth data a little: the last readings
int currentIndex = 0; // used for cycling through the array
int reading; // the latest reading

const float VOLTAGE_TO_CHECK = 5.0; // set the voltage you want to check

bool checkVoltage(){
  unsigned long sum = 0;
  for(int i = 0; i < SMOOTH; i++){
    sum += analogRead(A0);
    delay(10);
  }
  float average = (sum / (float)SMOOTH) * (5.0 / 1023.0); // convert to voltage
  return abs(average - VOLTAGE_TO_CHECK) < 0.1; // allow a tolerance of +/- 0.1 volts
}

void setup() {
}

void loop() {
}

```

Figure 49: Preview of JTR Board Coding Version 2
 You may find full version 2 code here: [JTR Version 2](#)

Version 3 - 3.1.2 – Button Toggle & Resets

There are a lot of subversions in Version 3 due to the addition of each button and to create multiple circuits.

Coding for V3-V3.1.2

I first created a simple circuit with one button, corresponding to an LED. The LED will light up once the button has been pressed. This is a quick and simple code where I would need to first tell the Arduino that both LED and previous button state are both LOW. If the button has been pressed, the current button state (btnState) will be HIGH. The idea of this version was to have players press the correct buttons at the same time, while the other false buttons act as a reset button.

The code below shown turned out to be problematic as I tried to make it as lazily as possible, which caused the if statement to loop constantly without being able to check the LED toggle at all. This could be caused if the button registers a click as a double click due to quick succession. This will then cause the LED to blink quickly and be unable to stay on. When I encounter this in the future, I will need to make sure that I separate the code up better and instead of using ! ledState, it would be easier and most likely work better by just calling LOW and HIGH individually.

```

C++ v
#define ledPin 12
#define buttonPin 4

boolean ledState = LOW;
boolean prevBtnState = LOW;

void setup() {
  pinMode( ledPin, OUTPUT );
  pinMode( buttonPin, INPUT );
}

void loop() {
  boolean btnState = digitalRead( buttonPin );

  if ( btnState == HIGH && prevBtnState == LOW ) {
    ledState = ! ledState;
    delay( 20 );
  }

  digitalWrite( ledPin, ledState );

  prevBtnState = btnState;
}

```

```

C++ v
#define ledPin 12
#define buttonPin 4
#define buttonPin2 5

boolean ledState = LOW;
boolean prevBtnState = LOW;
boolean prevBtnState2 = LOW;

void setup() {
  pinMode( ledPin, OUTPUT );
  pinMode( buttonPin, INPUT );
  pinMode( buttonPin2, INPUT );
}

void loop() {
  boolean btnState = digitalRead( buttonPin );
  boolean btnState2 = digitalRead( buttonPin2 );

  if ( btnState == HIGH && prevBtnState == LOW && btnState2 =
    ledState = ! ledState;
    delay( 20 );
  }

  digitalWrite( ledPin, ledState );

  prevBtnState = btnState;
  prevBtnState2 = btnState2;
}

```

Figure 50-51: Screenshot of Version 3 (left) and Version 3.1 (right)

You may find full version 3 code here: [JTR Version 3](#)

Poor and lazy coding could be found throughout the entire Version 3 iterations. Although there weren't any mistakes on the Arduino or breadboard itself, there were a lot of faults due to my poor coding. A mistake I kept making was continuously building the circuits off the coding which eventually caused all of the circuits to stop working as a whole. The code could potentially work if there is a small circuit with 1 button and 1 LED, but as the circuit size builds up, the code could not keep up anymore. This is something I could be more aware of in the future, further research could be done to prevent errors like these.

```

void loop() {
  boolean btnState = digitalRead(buttonPin);
  boolean btnState2 = digitalRead(buttonPin2);
  // for circuit2
  boolean btnState3 = digitalRead(buttonPin3);
  boolean btnState4 = digitalRead(buttonPin4);

  if (btnState == HIGH && prevBtnState == LOW && btnState2 == HIGH && prevBtnState2 == LOW) {
    ledState = !ledState;
    delay(20);
  }
  // for circuit2
  if (btnState3 == HIGH && prevBtnState3 == LOW && btnState4 == HIGH && prevBtnState4 == LOW) {
    ledState2 = !ledState2;
    delay(20);
  }
}

```

```

void loop() {
  boolean btnState = digitalRead(buttonPin);
  boolean btnState2 = digitalRead(buttonPin2);
  boolean btnState3 = digitalRead(buttonPin3);

  if (btnState == HIGH && prevBtnState == LOW && btnState2 == HIGH && prevBtnState2 == LOW) {
    ledState = !ledState;
    delay(20);
  }

  if (btnState3 == HIGH && prevBtnState3 == LOW) {
    ledState = LOW;
    delay(20);
  }

  digitalWrite(ledPin, ledState);
  digitalWrite(buttonPin2, ledState2);
}

```

Figure 52-53: Screenshot of Version 3.1.1 (top) and Version 3.1.2 (bottom)

In the end, I had to abandon the Version 3 code due to its complexity. I had stacked too many && statements, which confused the code, particularly when it had to check both current and previous button states in the same line. This caused a contradiction between the btnState and prevBtnState, preventing the code from proceeding.

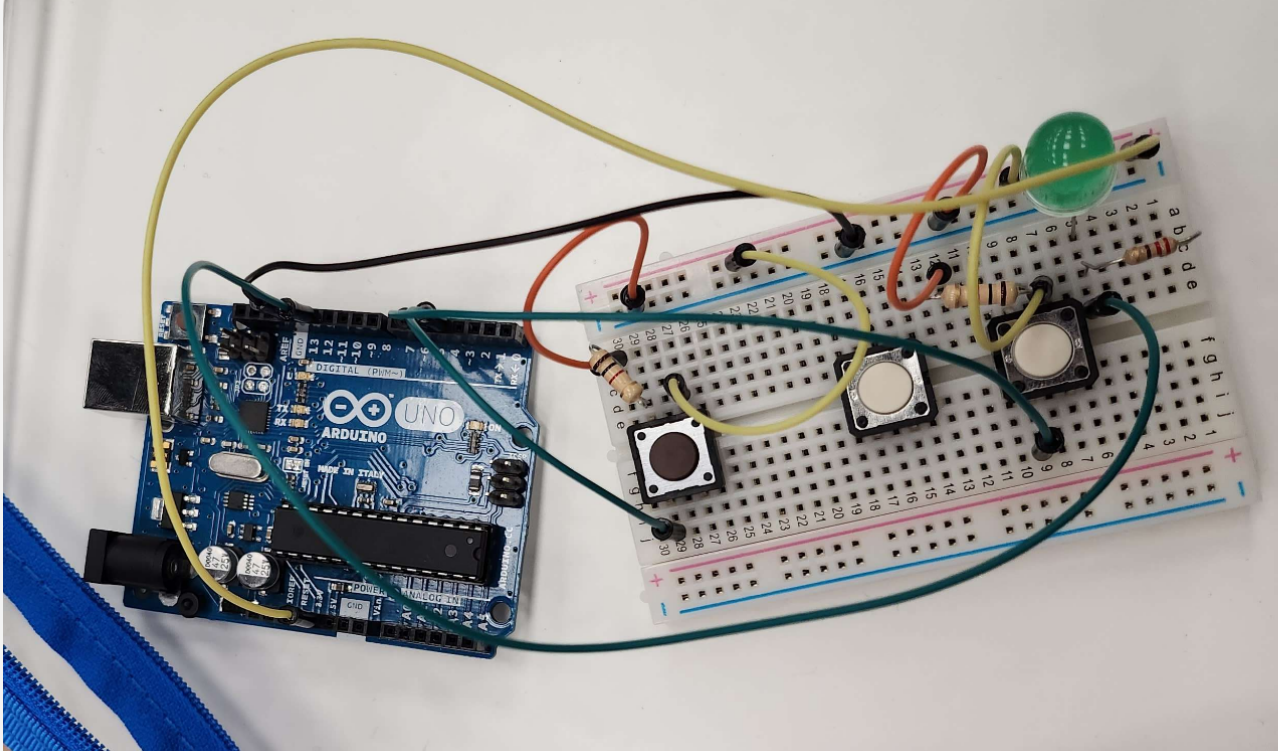
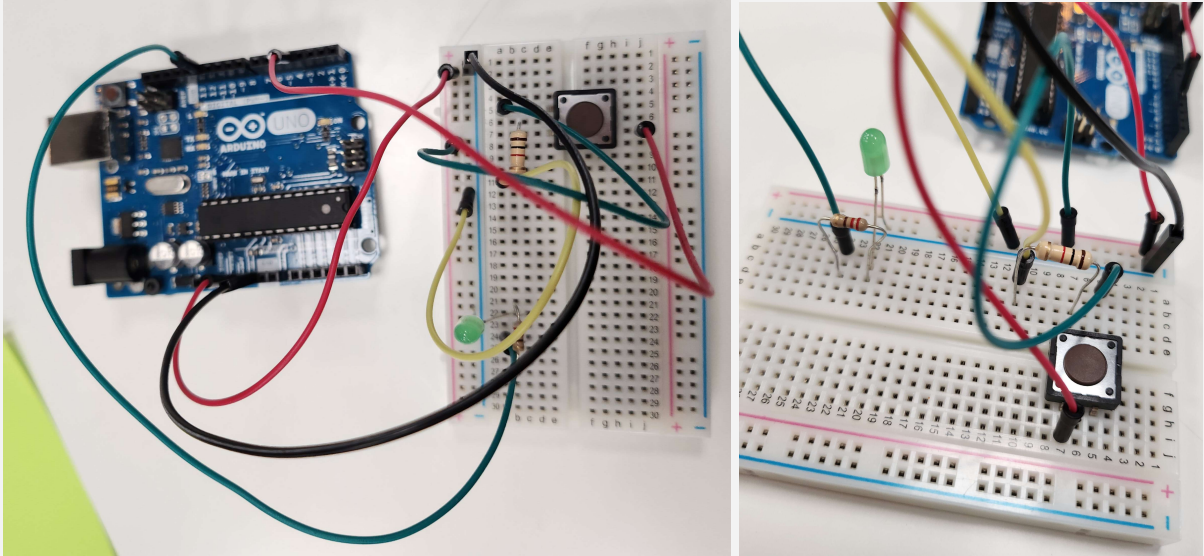


Figure 54-56: Single button circuit (top) & double button with reset button (bottom)

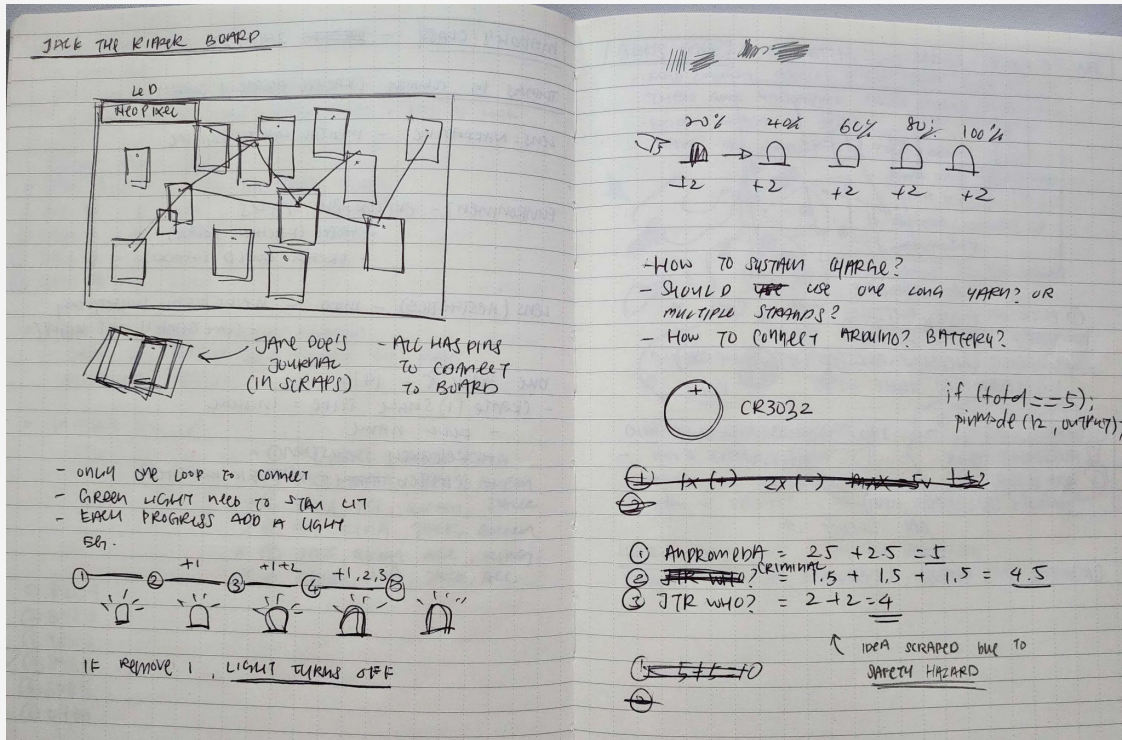


Figure 57: Notebook page of progression system plans

We decided to experiment on how we want the LEDs to light up. As we knew we wanted to have a progression system. I drafted up a system where each 2 pins has been connected and would add 20% to the progression of the game. Every 20% would then turn on one LED light.

Joanne then explained how a system like that is possible by using toggle buttons and creating a condition for the circuit to be checked.

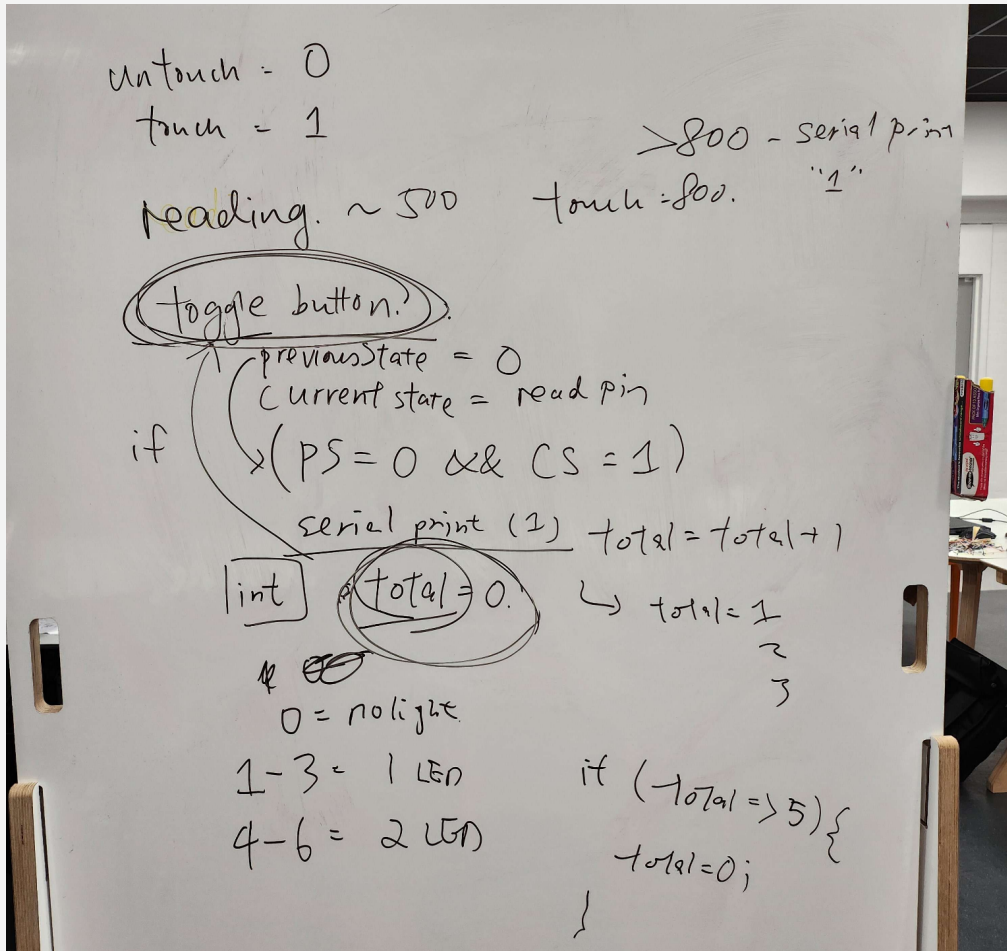


Figure 58: Joanne's code explanation on whiteboard

It is at this time I started to hit a wall, I was getting frustrated by the lack of progress with my plans. In response, I made the decision to temporarily shift my focus away from Arduino coding and circuit building, instead opting to begin on constructing the physical game board. I had drafted some concepts for the board's design, and I shared my ideas with both Kathi and Anu. My idea was to create a casing with multiple layers, allowing players the ability to close the top lid for immersion while dedicating the bottom layer to the Arduino and its wiring. Additionally, I planned to employ hinges and music box hinge locks to secure the layers, particularly the bottom layer to avoid having players fumbling with it.

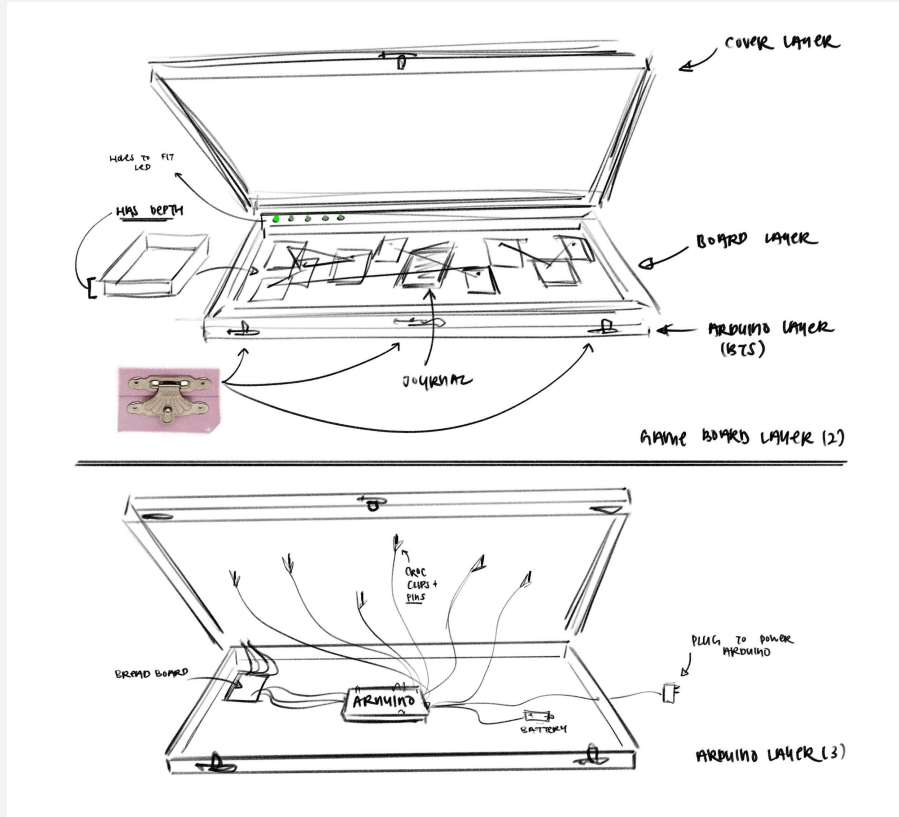


Figure 59: Physical board design 1

Another iteration of the board was to create a framed board with sides attached using a hinge that allows the player to close the board. Closing/hiding the board is a small immersive experience we wanted to give the players, letting them hide the evidence that they've been gathering and the progress they're making to solve the mystery.

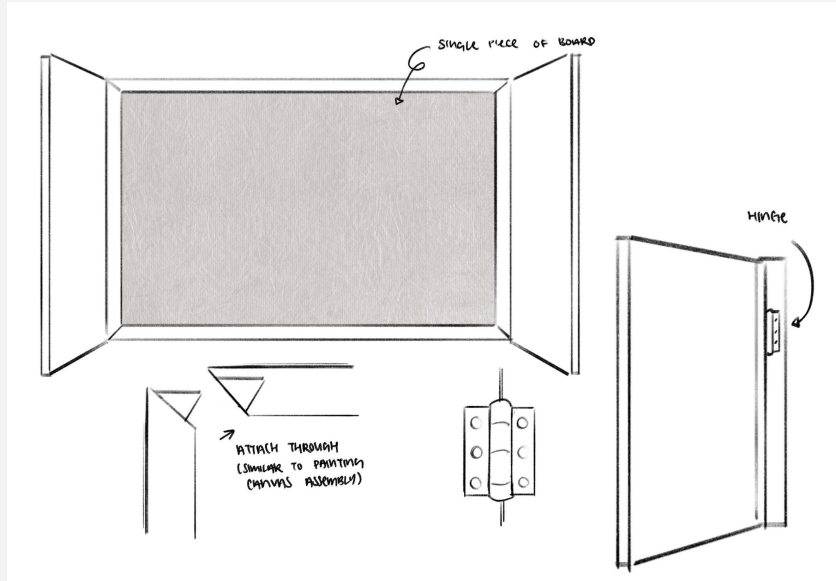


Figure 60: Physical board design 2

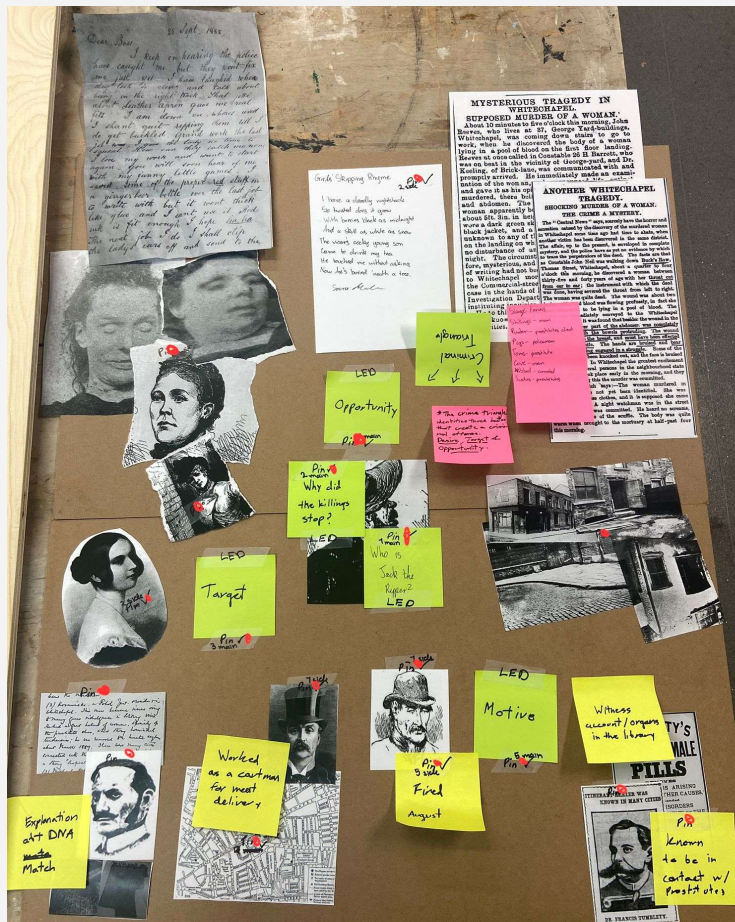


Figure 61: Physical board design prototype

After making the necessary arrangements, I secured a slot at the 3D workshop, the appointment was included with assistance. To work on the physical board, I purchased a lasergrade birchwood piece with the size of 1000x700mm and a thickness of 6mm. With the assistance of George, a skilled technician at the workshop, I had the material expertly cut down to the required dimensions of 800x650mm. To facilitate George in cutting the wood and pegs accurately, I drew a quick draft outlining all the pieces of wood that required cutting along with their respective dimensions.

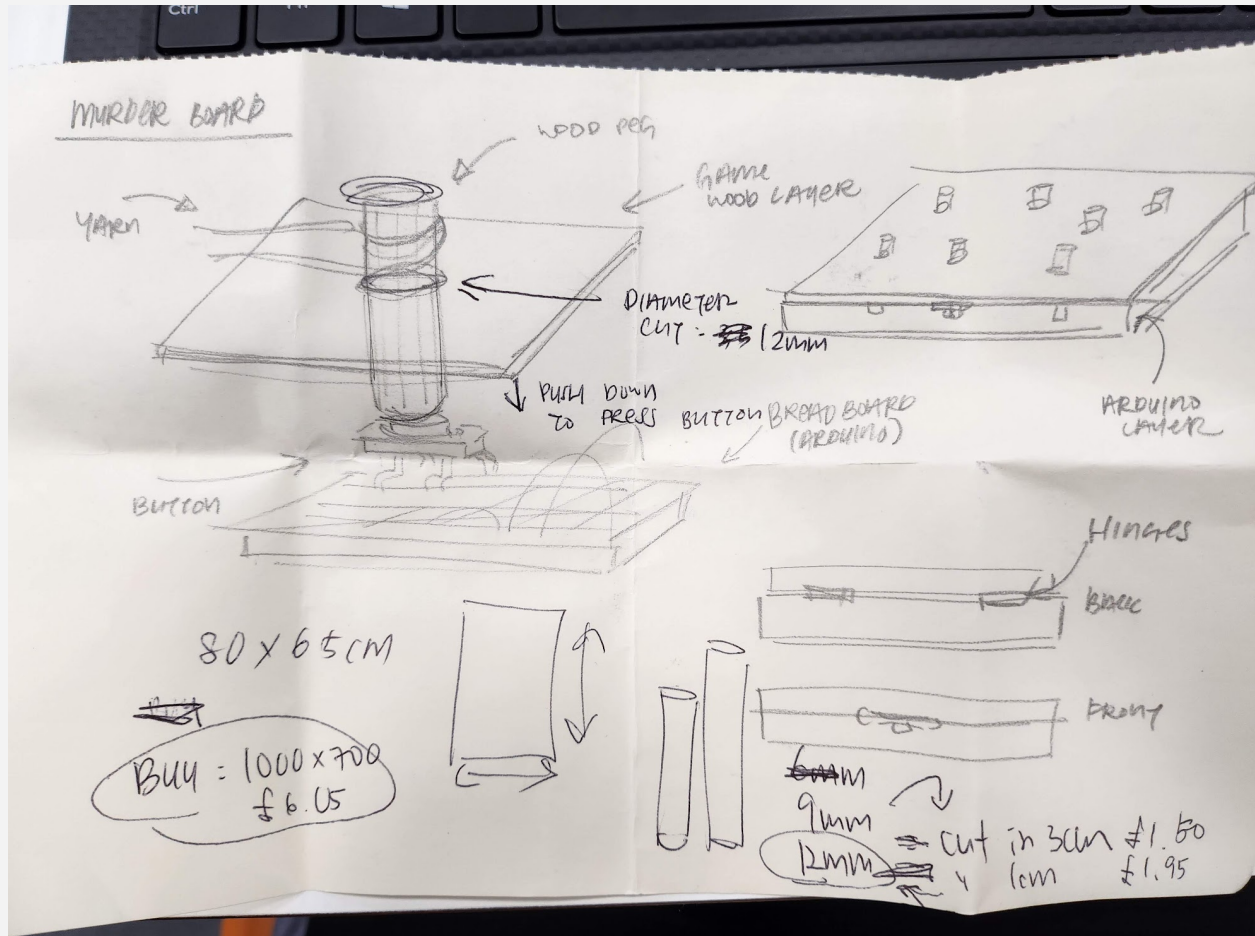


Figure 62: Final board design measurements for George



Figure 63-64: Board after cut and drills

After sanding down the board to a smooth finish, I proceeded to mark down the precise locations of all buttons and LED slots and started drilling onto the board. It was my first time drilling holes for wood, it was rather overwhelming but was a very fun experience. George had also helped cut down these cylinder pieces of wood which would then be made into pegs for the players to interact with the buttons under the board.



Figure 65: Board after cut and drills with wooden pegs

After ensuring that all the slots had been drilled through, I verified whether the wooden pegs could smoothly slide through the drilled holes. With this final step, the game layer physical component of the board is now complete.



Figure 66-67: Showcase of wooden pegs in action

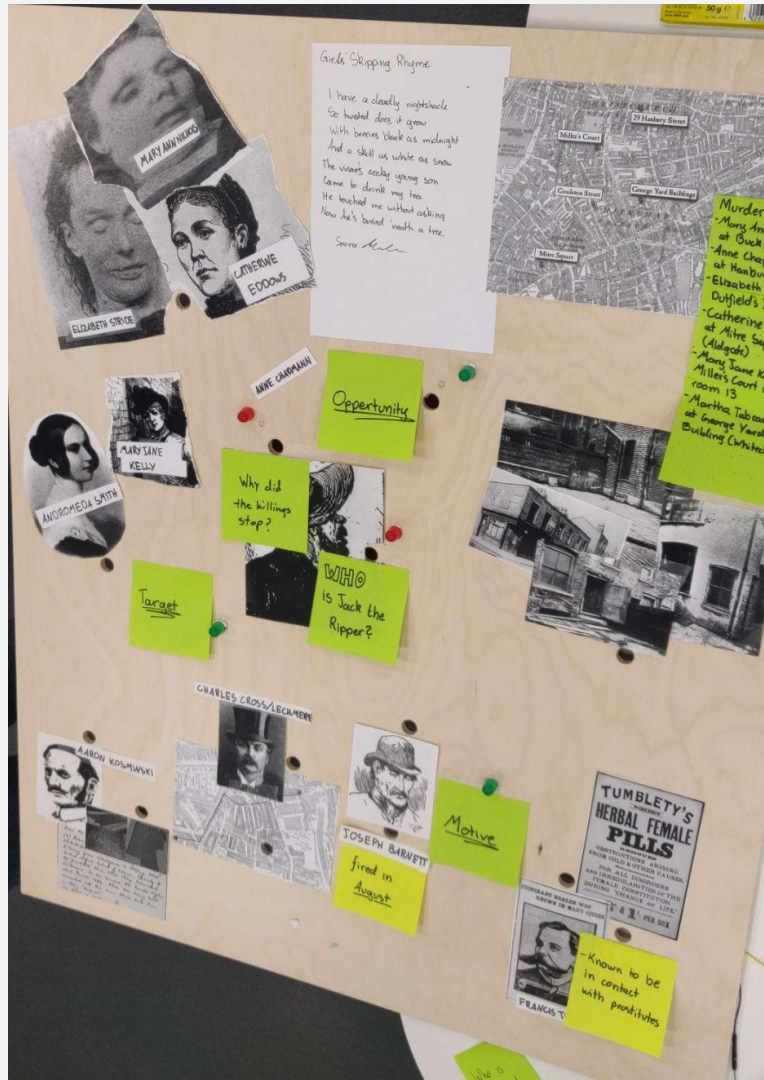
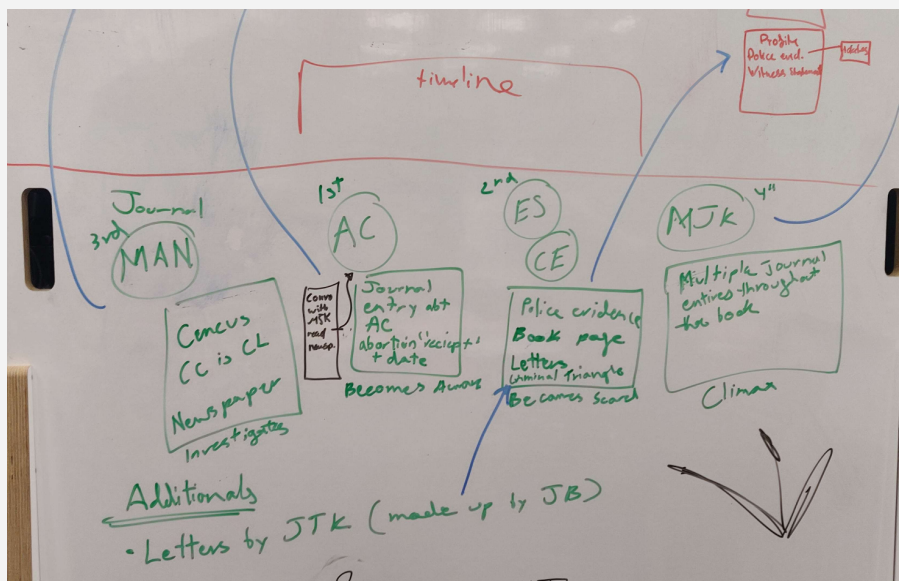
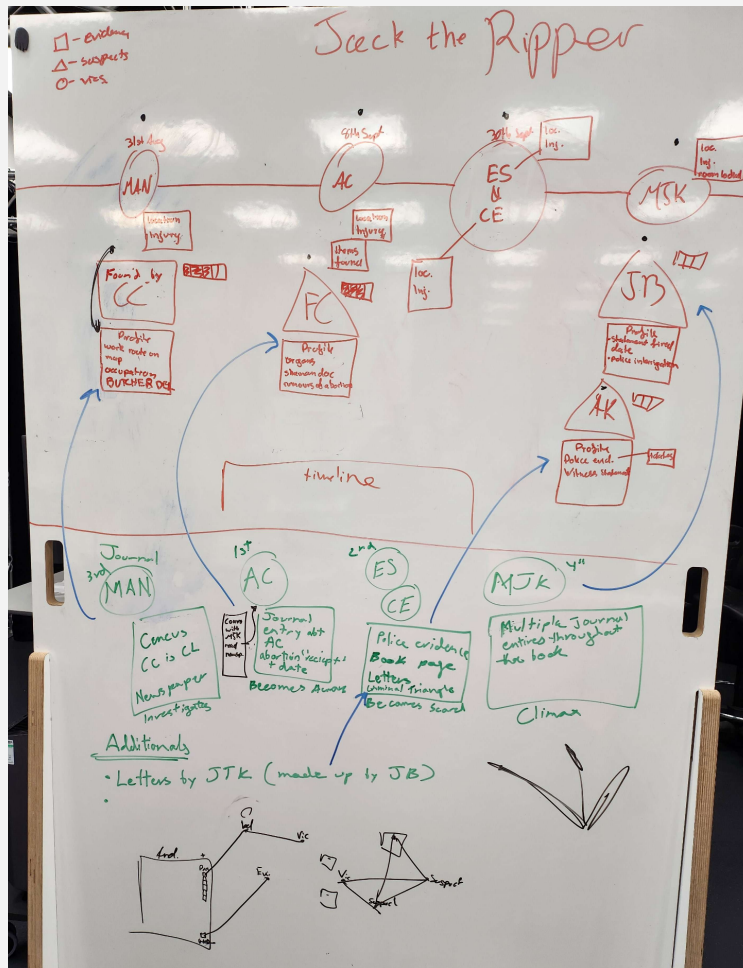


Figure 68: Board layout prototype on new and polished board

Version 4 – Double Button Clicks

During this phase of the project, significant revisions were made to the game's narrative. Additionally, I noticed the importance of implementing an indicator to confirm accurate button presses within the correct circuit is crucial, regardless of their order. I thought about potentially having the players pressing the correct buttons at the same time would be easier for us to handle the coding and Arduino circuits.

The changes made in the game's narrative were mainly for Andromeda's journal. Kathi and Anu worked on reorganising the timeline of the murders and seamlessly integrating Andromeda's role. As a key character in solving the mystery, Andromeda provides valuable insights into Jack the Ripper's identity and reason for ceasing their killing spree.



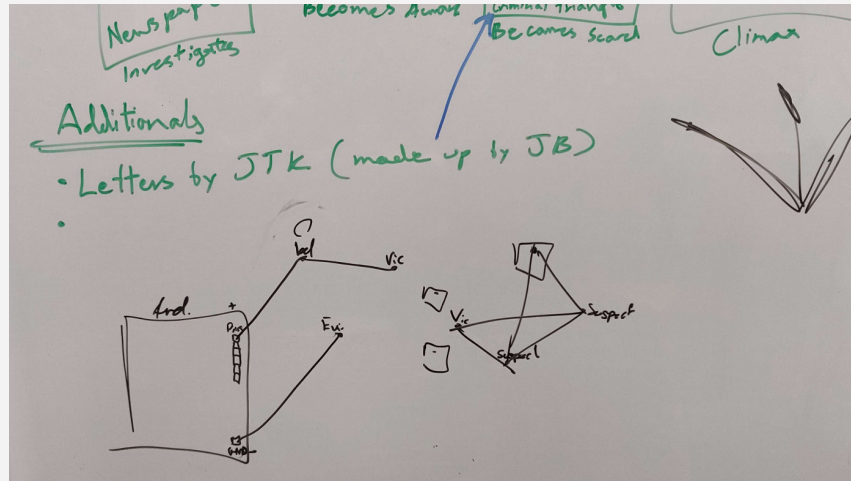
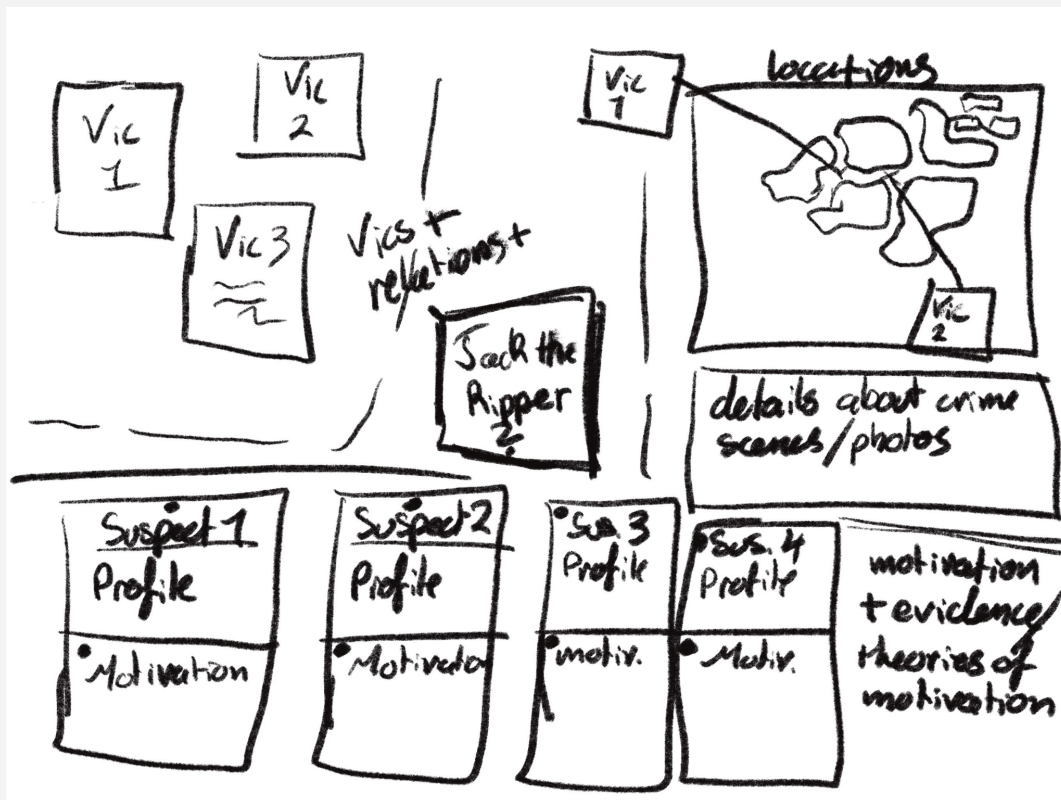


Figure 69-71: New board layout and narrative changes



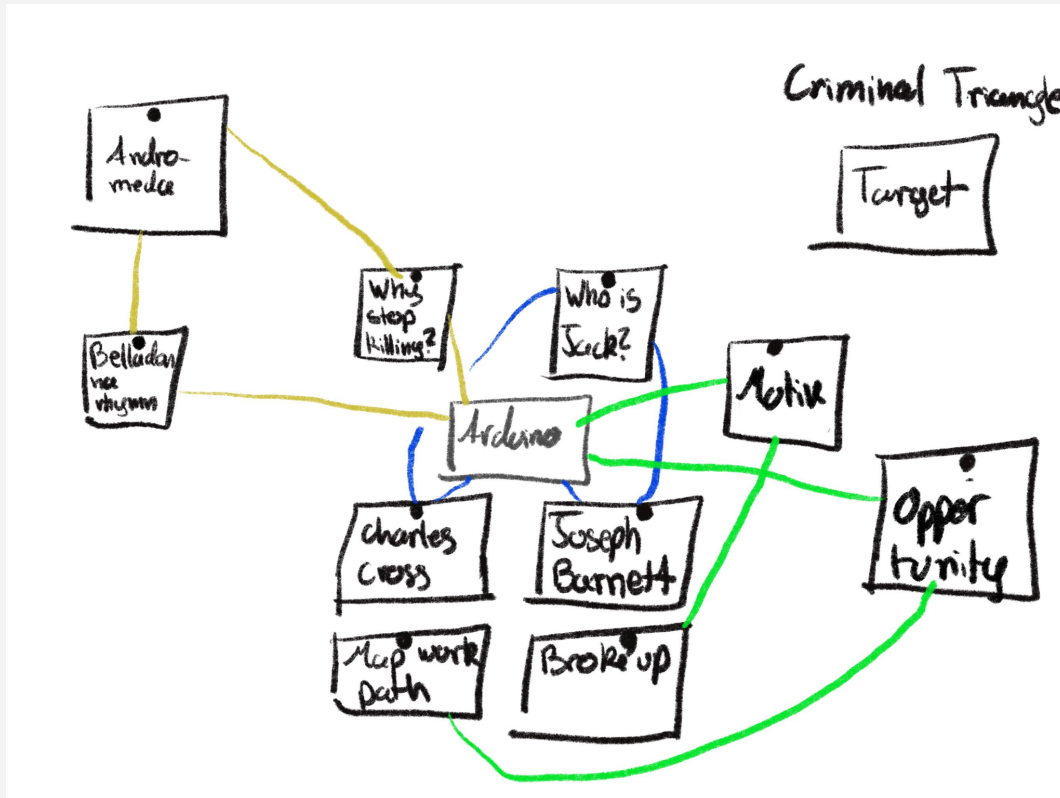


Figure 72-73: Anu's board layout design sketch

Since there are major changes to the narrative, the locations of each information and evidence would need to be shifted as well. Anu had drafted out a new board layout to cater towards the new information added.

```

void loopOld() {
  boolean btnState = digitalRead( buttonPin );
  boolean btnState2 = digitalRead( buttonPin2 );

  if ( btnState == HIGH && prevBtnState == LOW && btnState2 =
    ledState = ! ledState;
    delay( 20 );
  }

  digitalWrite( ledPin, ledState );

  prevBtnState = btnState;
  prevBtnState = btnState2;
}*/

```

```

/*
void loopOld2(){
  boolean btnState = digitalRead( buttonPin );
  boolean btnState2 = digitalRead( buttonPin2 );

  if ( btnState == HIGH && btnState2 == HIGH) {
    ledState = HIGH;

    digitalWrite( ledPin, ledState );
    delay( 20 );
  }
}

```

Figure 74-75: Screenshot of Version 4

You may find full version 4 code here: [JTR Version 4](#)

While Kathi and Anu are working on the changes to the narrative, I was working on incorporating the new changes into the Arduino system. In the two screenshots above, which showed the Arduino code, the first version (loopOld) was my initial attempt at creating a system that can detect multiple button presses simultaneously. Despite my efforts, some

issues with my coding were still evident in that version. This resulted in problems with the circuit and caused the LED lights to malfunction and not remain on as intended.

I then showed my code to David, who identified some errors and made minor adjustments that resolved the issues. However, the reset function that I had originally envisioned was not implemented in this version. After making the changes, David was confident that he had improved the code's efficiency by replacing the && operator with separate if statements.

```
#define ledPin 12
#define buttonPin 4
#define buttonPin2 5

boolean ledState = LOW;
boolean prevBtnState = LOW;
boolean prevBtnState2 = LOW;

void setup() {
  pinMode( ledPin, OUTPUT );
  pinMode( buttonPin, INPUT );
  pinMode( buttonPin2, INPUT );
  digitalWrite ( ledPin, LOW );
}

void loop(){

  boolean btnState2 = digitalRead( buttonPin2 );

  if (btnState2 == HIGH) {
    digitalWrite( ledPin, HIGH);
    delay( 20 );
  }
}
```

Figure 76: Screenshot of updated code with assistance of David

David examined the Arduino board and noticed that some of the wiring connected to the buttons was unnecessary and only created clutter. He explained that button pins work by carrying current from one side to the other. Therefore, it was only necessary to solder one side of the button to make it functional. This saved space on the circuits and reduced the need for conductive wires. With this in mind, I intend to apply this knowledge when designing future circuits for the project.

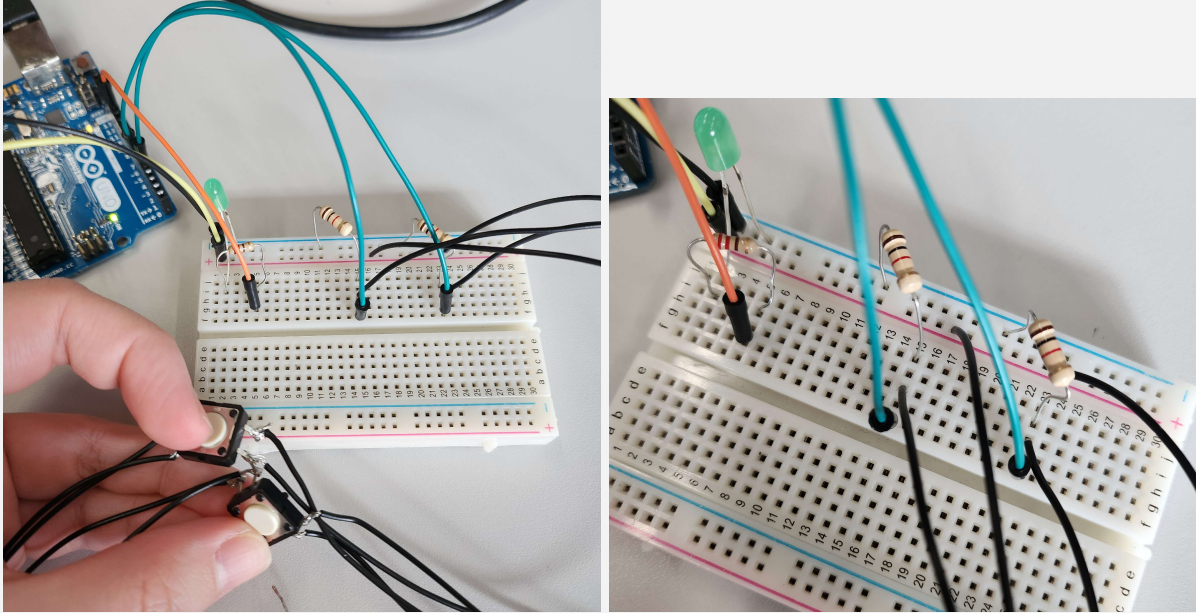


Figure 77-78: Photo of Arduino board and breadboard after David's suggestive changes

After Kathi and Anu had sorted out the new layout of the board, they conducted our first ever playtesting for the game, with playtesters such as Maddy and Kathi's housemate: Kalina. We had received some necessary insights from both Maddy and Kalina, both of their feedback were valuable for enhancing our game. The gameplay became more fluent with some minor adjustments.

Regrettably, the playtesters were unable to experience the Arduino interaction system in conjunction with the narrative aspect of the game. At that time, David and I were still discussing how we could improve the button interaction and LED progression system to provide a more seamless experience for players.

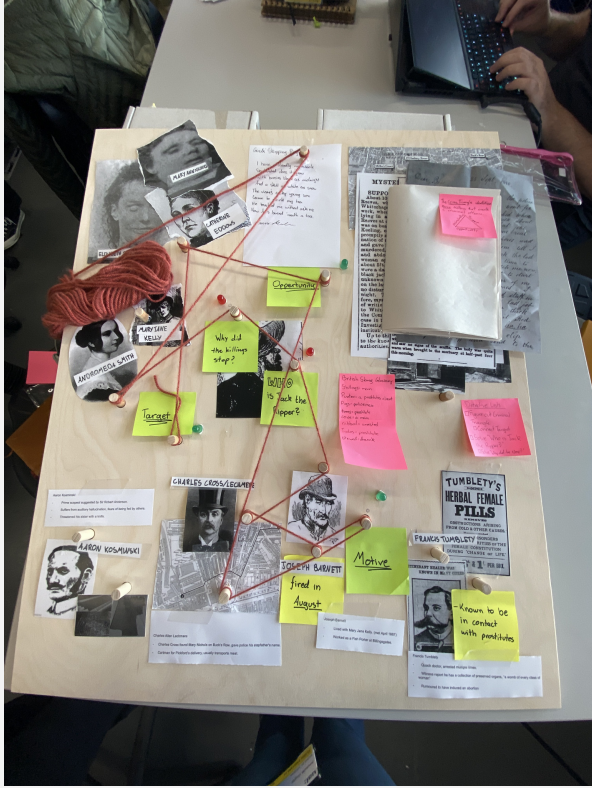


Figure 79-82: First playtesting with Kalina and Maddy

Version 5 – Timed Based Button Reset

While talking to David, I mentioned my initial plan of implementing a reset function for the buttons if incorrect buttons were pressed. David proposed using a timer reset function with `myTime` and `millis`. This would allow us to set a specific time for the button conditions to be verified. How it works is, firstly, the internal timer will only start when a button has been pressed from the circuit. If the second (and third) buttons are not pressed within X amount of seconds, the circuit will automatically reset. At this point, we also modified the button conditions. Instead of requiring players to press all buttons simultaneously within the circuit to activate the LED light, they now only needed to press them individually.

Reference > Language > Functions > Time > Millis

millis()

[Time]

Description

Returns the number of milliseconds passed since the Arduino board began running the current program. This number will overflow (go back to zero), after approximately 50 days.

Figure 83: Millis function tutorial from Arduino documentation

As shown in the code below, the line `if(millis() > myTime + 2000)` will check whether the interval between the last button press and if the most recent button press is under 2 seconds. At the end of the code, the if statement where `btnPressed` and `btnPressed2` conditions will then be checked whether they are both TRUE. If they are both TRUE, `ledState` will be set to HIGH (LED will turn on). With this code, we are much closer to our final version than ever before. Although at the moment, this code only works with one circuit.

```

void loop() {
  boolean btnState = digitalRead(buttonPin);
  boolean btnState2 = digitalRead(buttonPin2);

  if (btnState == HIGH){
    btnPressed = true;
    myTime = millis();
    Serial.println(myTime);
  }

  if (btnState2 == HIGH){
    btnPressed2 = true;
    myTime2 = millis();
  }

  if(millis() > myTime + 2000){
    btnPressed = false;
  }

  if(millis() > myTime2 + 2000){
    btnPressed2 = false;
  }

  if (btnPressed == true && btnPressed2 == true){
    ledState = HIGH;
    digitalWrite(ledPin, HIGH);
    delay(20);
  }
}

```

Figure 84: Extended updated code

You may find full version 5 code here: [JTR Version 5](#)

Version 6 – Timed Button Reset (Faulty Code)

With David's code changes, I was able to replicate it for a second circuit. With the second circuit being a three button circuit, I was able to test out how efficient the millis function was. Unfortunately, the code did not work as intended as the code seemed to only work for the first circuit that was interacted with. Any other circuits after the first would not be functioning.

I approached Joanne once again questioning the reason for the circuits malfunctioning. This was when Joanne mentions that due to the if statement getting constantly looped to check the condition of whether the btnPressed is true, it could never proceed to the next circuit. The mistake was not catastrophic, but it does cause the other circuits to not function at all. It was then Joanne had instructed on how I could fix the code, which allowed me to create the final working version of our game. You may find full version 6 code here: [JTR Version 6](#)

Version 7 – Working Final Versions with All Circuits

In Version 7, Joanne kindly pointed out errors I made in Version 6 and offered valuable suggestions for improving the code's structure. By implementing her recommendations, I was able to enhance the system's efficiency and make it easier to debug. A well-structured code allows for seamless troubleshooting of any issues that may arise in the future.

Joanne proposed a counter system to complement the timer reset system. Her idea was to assign an integer to all circuits starting from 0, which would track the number of button presses. As we progress, each button press would increment the counter by 1. During the setup phase, she also suggested setting the intervals before the loop function for button resets, making it easier to modify individual circuits.

```
int C1 = 0;
int C2 = 0;
int C3 = 0;
int C4 = 0;
int C5 = 0;
```

```
unsigned long previousMillis = 0;           // will store last time LED was updated
unsigned long previousMillis2 = 0;
unsigned long previousMillis3 = 0;
unsigned long previousMillis4 = 0;
unsigned long previousMillis5 = 0;
const long interval = 5000;
const long interval2 = 8000;
const long interval3 = 5000;
const long interval4 = 5000;
const long interval5 = 8000;
```

Figure 85-86: Joanne's suggestive changes in code (assigning previousMillis to 0)

You may find full version 7 code here: [JTR Version 7](#)

In order to consistently monitor the time intervals after any of the buttons in a circuit are pressed, the millis operator was integrated into the loop function, providing live and up-to-date time checking.

```
void loop() {
  unsigned long currentMillis = millis();
  unsigned long currentMillis2 = millis();
  unsigned long currentMillis3 = millis();
  unsigned long currentMillis4 = millis();
  unsigned long currentMillis5 = millis();
}
```

Figure 87: Joanne's suggestive changes in code (millis function indicator)

When it comes to the actual coding for individual circuits, take Circuit 1 (with 2 buttons) as an example:

```
// C1 (2 buttons)
if ( btnState == HIGH && btnPressed == LOW ) {
  C1 = C1 + 1;
  btnPressed = true;
}

if ( btnState2 == HIGH && btnPressed2 == LOW ) {
  C1 = C1 + 1;
  btnPressed2 = true;
}
if(C1 == 2){
  digitalWrite(ledPin, HIGH);
}else{
  digitalWrite(ledPin, LOW);
  // reset to base timer
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    C1 = 0;
    btnPressed = false;
    btnPressed2 = false;
  }
}
```

Figure 88: First proper working code with counter and timer system (with assistance of Joanne)

The circuit will first check whether the *btnState* is currently high when previously, the button was not pressed. If the condition of *btnState = HIGH* is true, C1 (Circuit 1) will + 1, which in turn, *btnPressed* would be true.

(Disclaimer: Now that I have re-read the code, the *btnPressed = LOW* should have been renamed to *prevBtnState* instead, this causes a reading issue, confusing anyone who eventually reads the actual code. The code is fully functioning now and I would rather not change that.)

The if statement will then need to check all buttons in the circuit. When it comes to checking the entire circuit, taking Circuit 1 as an example, we would need an if statement to check whether C1 is == to 2. If C1 is at the value of 2, the LED for the circuit will switch on. At the same time, when a button has been pressed in the circuit, the millis operator will start an internal timer of X seconds (5 seconds for C1). If C1 does not meet the threshold of the intended value (2) by the time 5 seconds has passed, C1 will reset to 0, therefore, both button pressed conditions will be false and the circuit will be reset.

It is also to note that the sole reason why *btnPressed = true* was added was due to the buttons we were working with were unreliable. Buttons such as those could often double click due to quick succession. Joanne's solution was to add that once the button has been pressed, all other presses would no longer be registered. This would then prevent the circuit to continue counting in increase due to a faulty button click.

Using this new code, I was able to replicate it across all circuits, gradually and satisfactorily completing our physical computing section of the game.

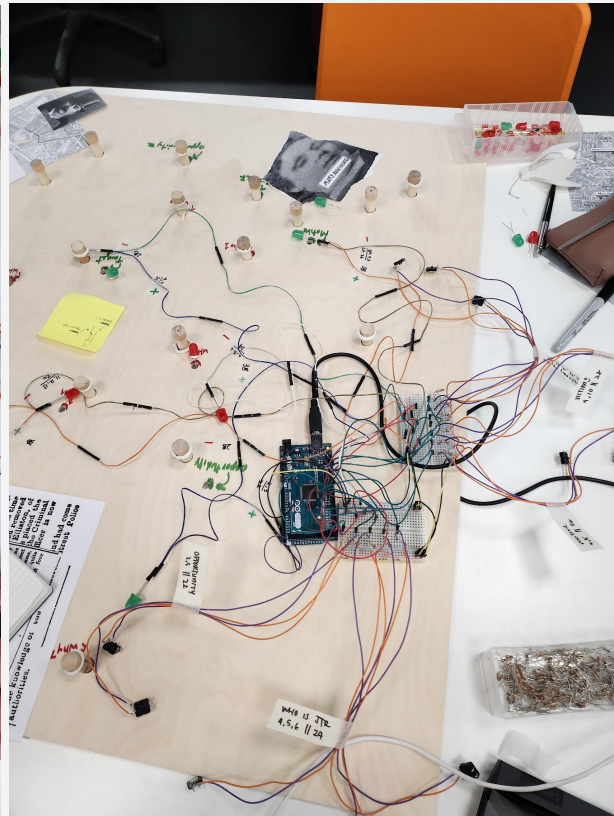
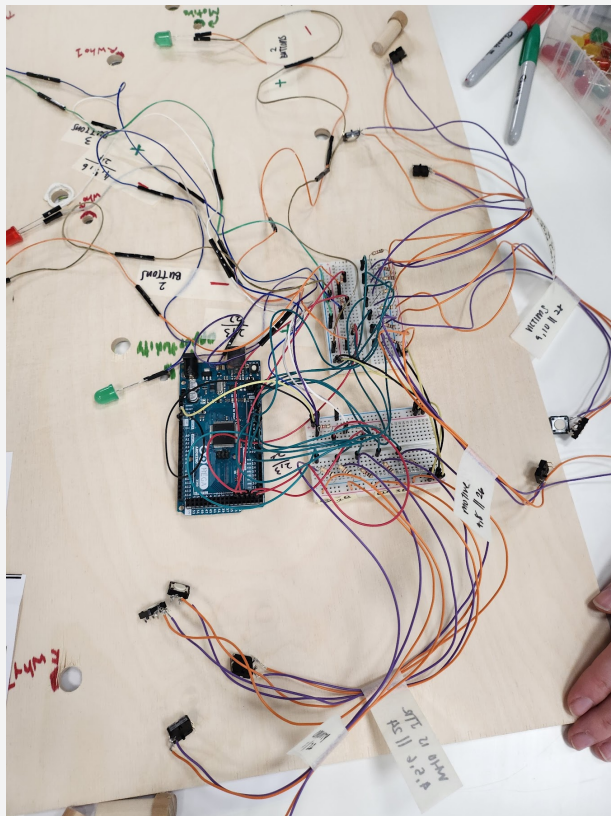
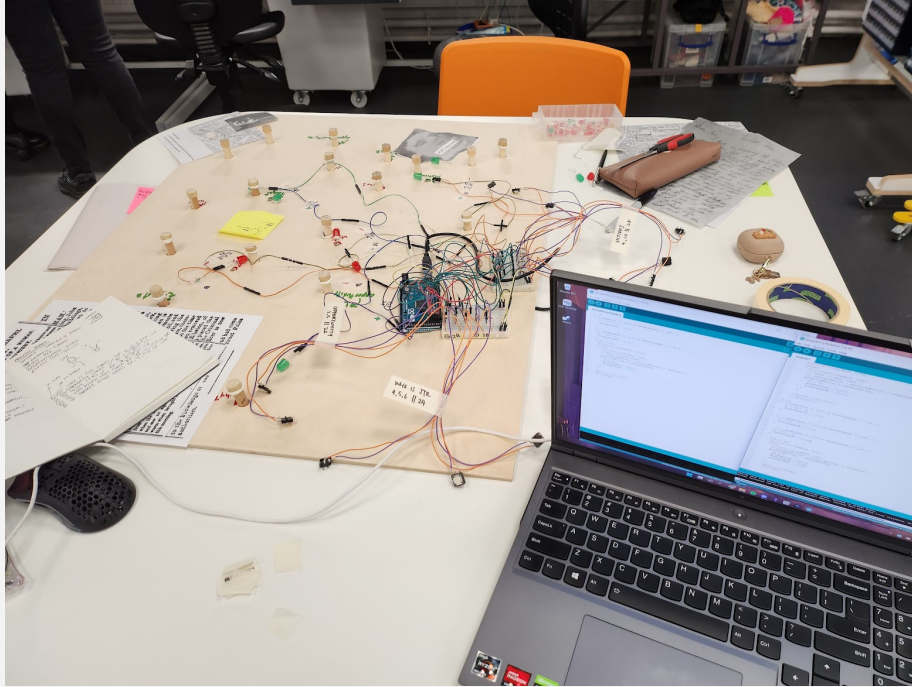


Figure 89-91: Final uncensored layout of a working Arduino system

During the following week, I dedicated most of my time to organising the physical components of the Arduino board. Although the wiring may have appeared messy at first glance, I ensured

that each wire was colour-coded, labelled, and ready to be assembled onto the board with the accompanying assets. Simultaneously, Kathi and Anu began constructing the board's foundation where the Arduino would be placed. This enabled us to assemble and finalise the board as a complete unit.



Figure 92: Kathi and Any working on the base of the board (Arduino base)

Final Outcome

While I was tidying up the Arduino board, Kathi and Anu were hard at work creating all the assets. They meticulously crafted every detail, from printing old letters to handwriting diaries and postcards with specialised nib pens, to immerse players in an old-timey experience.

The final game will include the following pieces:

- 1x Arduino-based alternative controller game board with attached evidences
- 1x Diary of Andromeda Smith
 - 2x Postcards from your best friend in Andromeda's diary
- 1x Victim Profile document

- 1x Suspect Profile document
- 1x Hints

Photos of final game board and documents attached below:

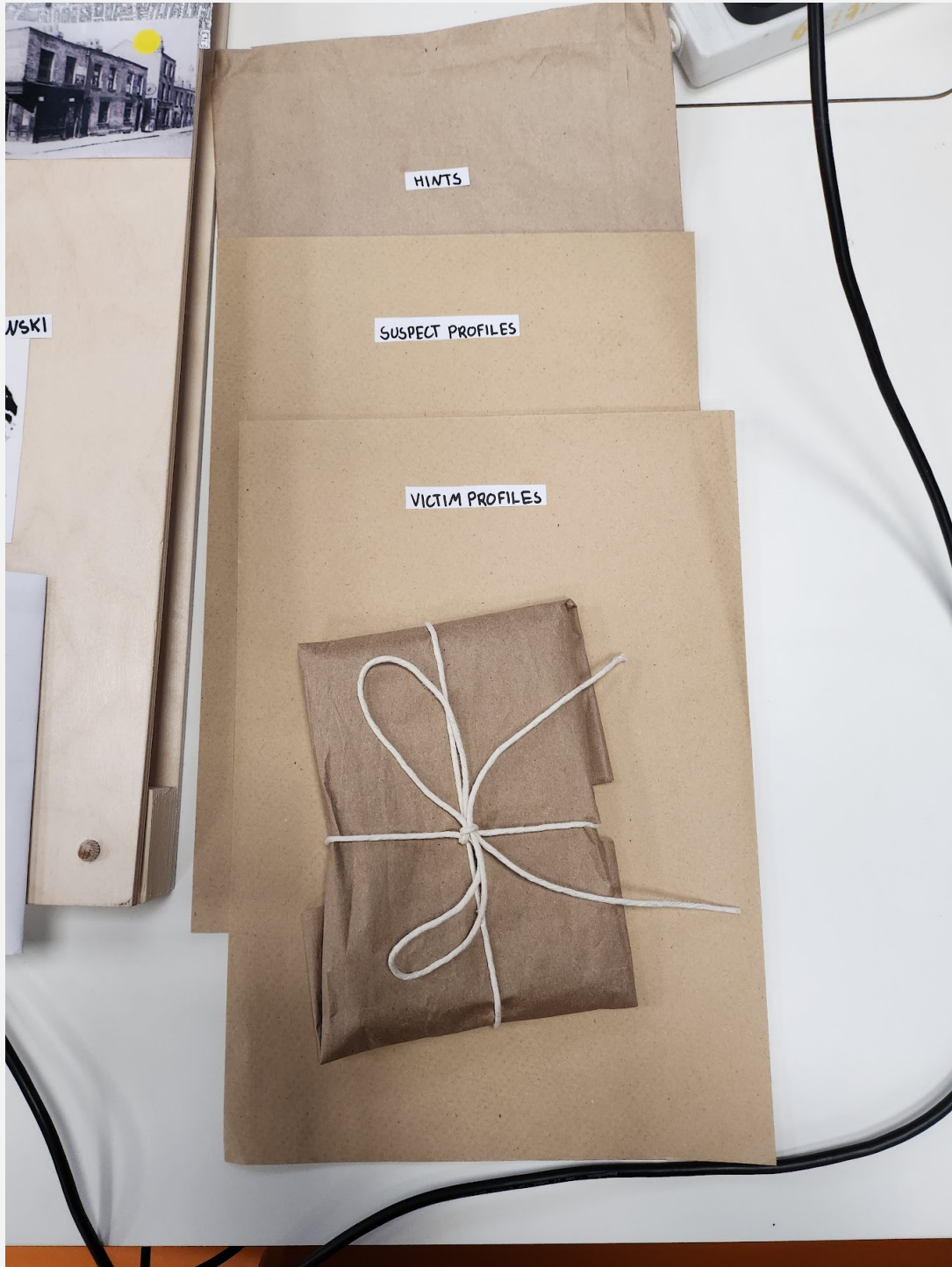


Figure 93: Welcoming letter and Andromeda's diary inside wrapped brown paper, victim profiles, suspect profiles and hints in brown letters

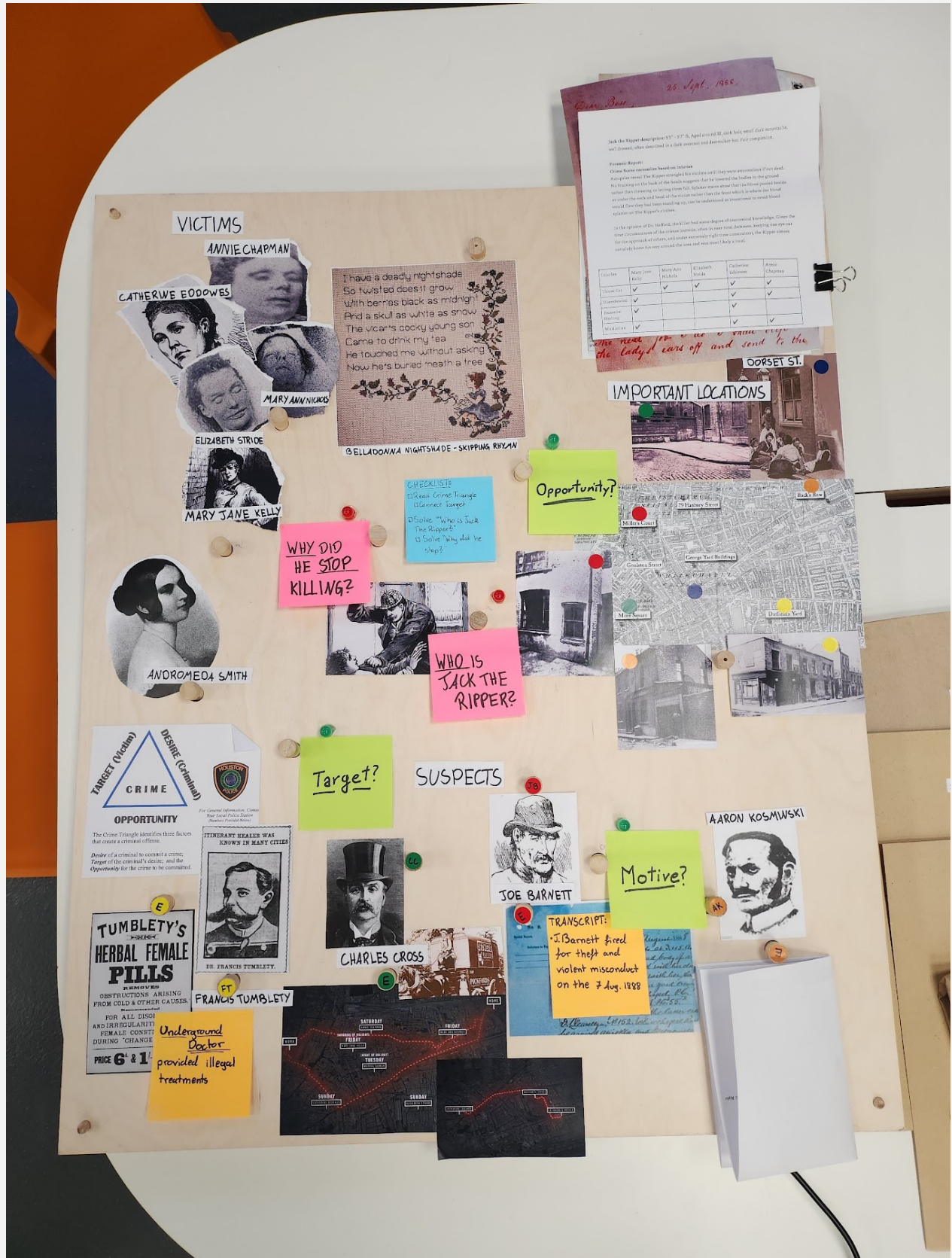


Figure 94: Top down view of the game board

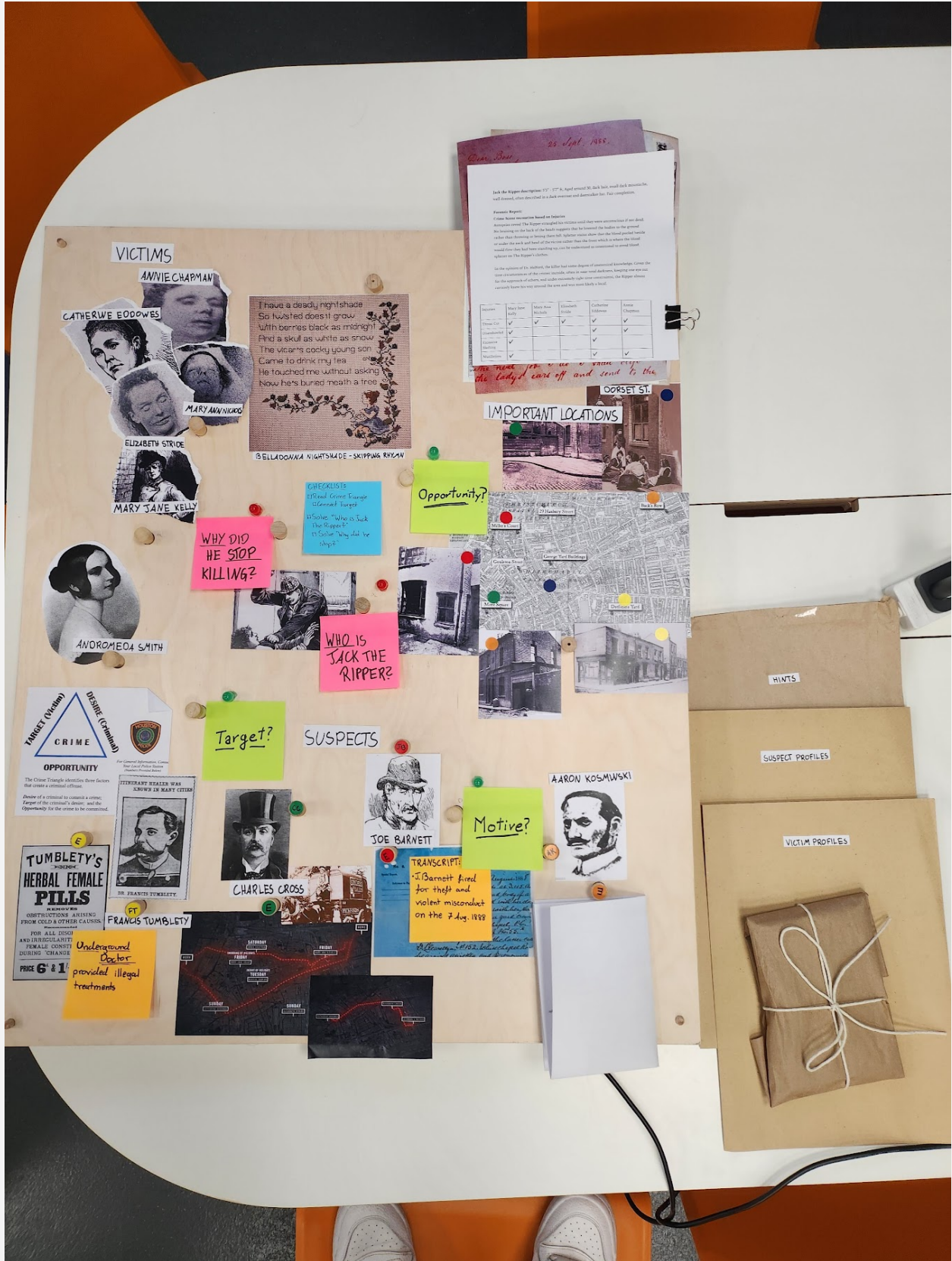


Figure 95: All game elements laid out

Reflection

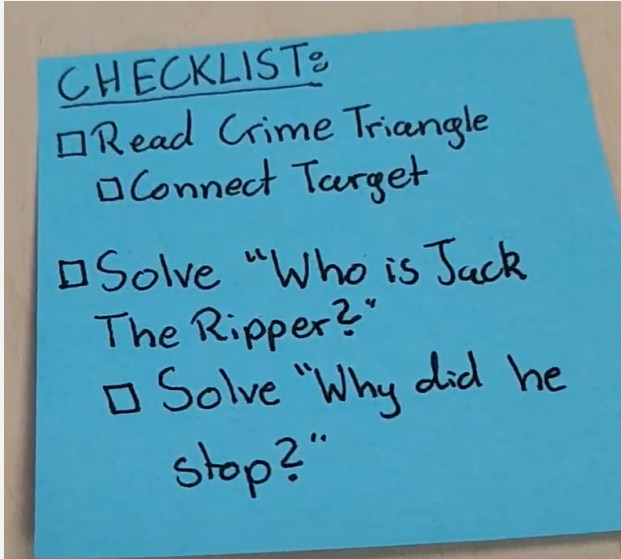

Throughout the past four weeks of running through multiple iterations, I have acquired a wealth of new skills and knowledge. One crucial lesson I obtained from my experience with Arduino was the importance of patience, and that there are numerous ways to approach its programming, even with the simplest systems. I also discovered an abundance of resources available online, such as Tinkercad, Arduino documentations and more. Additionally, UAL's Creative Computing Lab has its own resource page for Arduino tutorials, curated by Joanne.

Feedback

	Feedback/Observations	Changes	Reasons/Thoughts
1	External observers have expressed concerns regarding the safety of the conductive yarn iteration	We underwent several iterations to determine the safest method for interacting with the game. The final version now utilises buttons and a regular yarn to create an immersive investigative experience	While the idea of using conductive yarn to connect all the clues initially seemed interesting and enjoyable, but our top priority remains ensuring the safety of all participants
2	Players have reported difficulty in reading certain evidence due to the pixelated appearance	To enhance legibility, we reprinted all evidence in a larger and readable size	The prints utilised in the playtests were early prototypes
3	Observations revealed that the buttons were difficult to press down, resulting in some clicks not being registered	The buttons underneath the board were raised up, allowing better access for the wooden pegs to be pressed down	We raised the board height to accommodate the necessary space for the Arduino beneath the game board layer
4	Players feel overwhelmed and uncertain about which tasks to tackle first. Players also experience difficulty with certain questions, leading to a sense of	We included a sticky note containing a checklist of all the tasks that needed to be completed. Additionally, we incorporated hints to assist	Our intention was to emulate the messy and disorganised nature of an investigation board,

	being stuck	those who may require additional guidance to solve the mystery	but this approach may have led to feelings of overwhelm among the players.
5	Players feel uncertain about which buttons to press for the suspects due to the tightly spaced positioning of the buttons	To differentiate between the buttons used for suspects and those for evidence, we attached coloured circular sticky tags featuring the initials of each suspect and the letter "E" denoting evidence	I neglected to consider the amount of spacing required to accommodate the evidence buttons between the suspect buttons.
6	Through previous iterations, it was observed that pressing all the correct buttons within the circuit was necessary to demonstrate progression. This posed issues as buttons would not register properly, causing players to believe they have the wrong answers	New iterations have been made to enhance the player experience, with the final iteration utilising a counter and timer-based system for improved gameplay	Initially, implementing a reset circuit system posed a challenge. However, with the guidance of David and Joanne, they were able to suggest a more straightforward solution for me to work with.

Photos of Updated Changes

Feedback Number	Photos
4	 <p data-bbox="381 966 1055 997">Figure 96: Sticky note of checklist for players to check off</p>  <p data-bbox="381 1627 690 1659">Figure 97: Hints envelope</p>

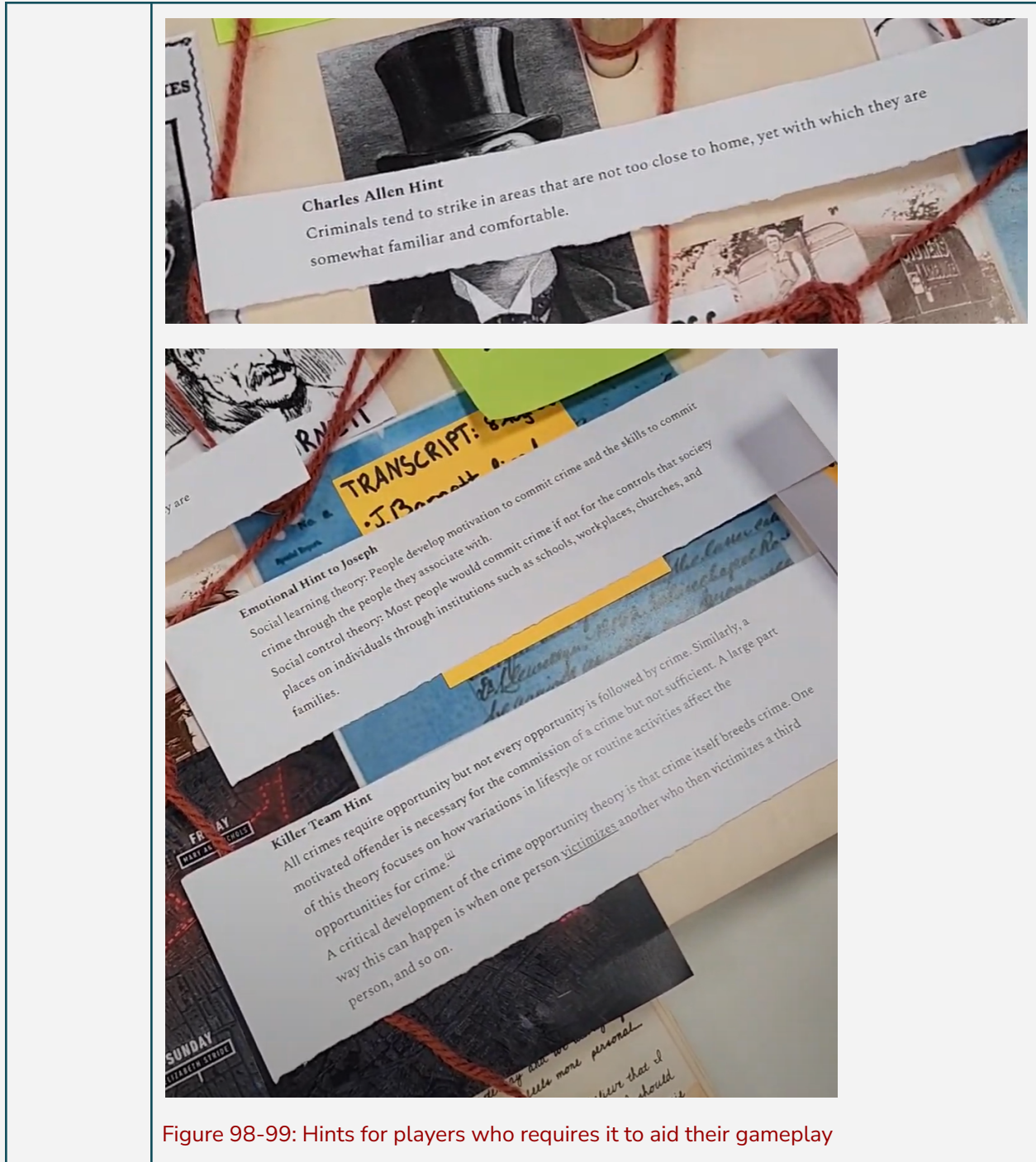
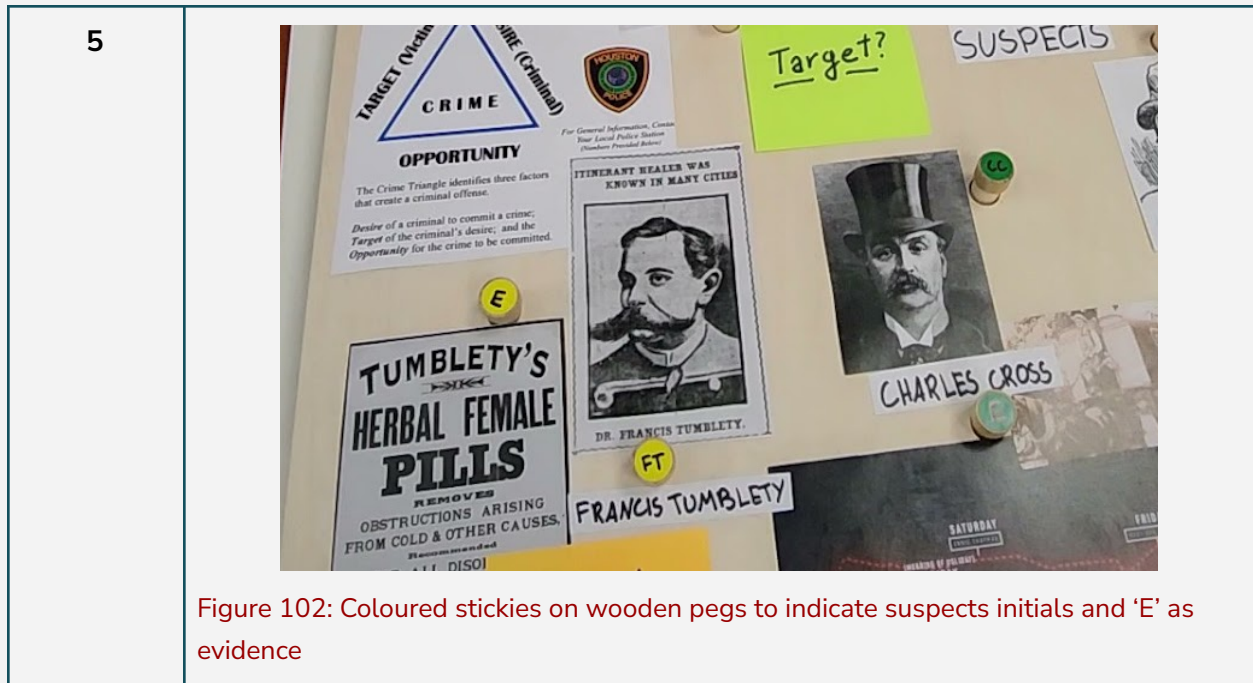


Figure 98-99: Hints for players who requires it to aid their gameplay

I found those
in [redacted] things,
keep them safe.
By the heavens,
use your head.

I was not kidding dear
Boss when I gave you the top
you'll hear about Ripper
Saway Jacky's work tomorrow
double events this time need to be
one on squelch but
couldn't couldn't

Figure 100-101: More hints in Andromeda's diary



Critical Reflection

Upon reflection, the primary objective I had in mind at the outset of our project was to design a murder mystery game using an Arduino-based alternative controller, thereby merging the areas of physical and creative computing with criminology to forge a more comprehensive link to game design. Since my previous term projects had centred around food and familial motifs, I sought to gradually explore and incorporate new themes for my final major project, such as mystery and psychological horror. This undertaking provided me with ample opportunity to gradually shift the themes of my games and hone my skills in devising different types of game interactions.

Prior to this project, I had no experience with Arduino. Having to squeeze months or possibly years of learning into 3-5 weeks was a very daring move. Although it was a challenging experience, it was ultimately a rewarding one. To improve my programming skills, particularly in C++ and JavaScript, I offered to take on all the programming work, as I was lacking in proficiency in this area. While my improvement may not have been significant, I am confident that my skills have been expanded and refined as a result of this project.

When it comes to crafting a well-crafted narrative, we utilised LEMMINO's video essay on Jack the Ripper entitled "The Enduring Mystery of Jack the Ripper" as a significant reference. In

addition to that, we conducted extensive research on our own. Kathi and Anu complemented the game's narrative with fictional storytelling that blended seamlessly with the factual storyline.

During the playtesting phase, we noticed that we were able to achieve our goal of having the game to cater to players of all sizes, offering them enough tasks and interactions to keep them engaged. However, some players found it overwhelming and were unsure of where to start. To address this issue, we made certain adjustments, such as incorporating hints and checklists, to guide players on their path of investigation.

Initially, we had considered including multiple branching endings, but we were pursuing a different narrative direction at that point. Upon further consideration, we opted against branching endings due to the heavier and more complex workload they would entail for the Arduino system design. Given that we were all relatively new to Arduino, our aim was to keep the system as straightforward as possible.

As a whole, this project proved to be an invaluable learning experience for each of us. It allowed us to effectively integrate various topics and expand our proficiencies in programming, narrative design, and game design. The obstacles we encountered during the development process taught us the importance of patience and perseverance. The end result was one of which we are proud, and we are eager to continue enhancing our abilities in game design for all of our upcoming projects.

Credits

Skylar Law (22045627) – Physical game building & Arduino programming

Anukriti Gupta (22018366) – Narrative Design, art asset design & gameplay video editing

Katharina Trappe (19016176) – Narrative Design & Lettering design