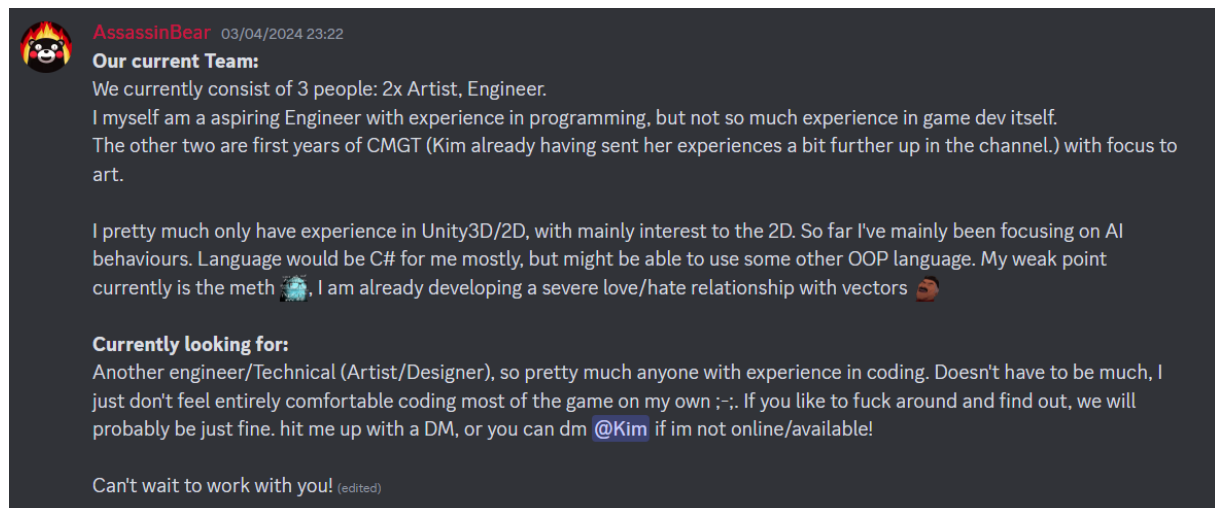


I decided to participate at the XP Game Jam by the association of my CMGT study as my first game jam. I had always found the game jam concept to be daunting, requiring the dedication of a block of free (weekend) time, crunch and stress for a few days straight, uncomfortable logistics and work responsibility to oneself or to a team of people. With this participation, I wanted to join the experience anyway, see how (un)founded my fears are, and possibly connect with new people.



The jam took place on location at the Saxion CMGT premises during the weekend of April 5-7, and as such featured the full experience where the motto for everyone was game development for 48 hours straight.

Team composition

After hesitation in choosing whether to work solo or in a team until the last day before the game jam, I tried my luck on one of the open offers in the game jam Discord server. Among my priorities and expectations when signing up for this jam were to approach the development with a laid-back mindset that tries to finish a game regardless of quality, scope, and previous work experience. The offer I went for seemed to complement my strengths and my expectations well enough, also being open enough to people not looking for professionalism:

A screenshot of a Discord message from a user named AssassinBear. The message is on a dark background with light text. It includes a profile picture of a character with a flame on its head. The text of the message is as follows:

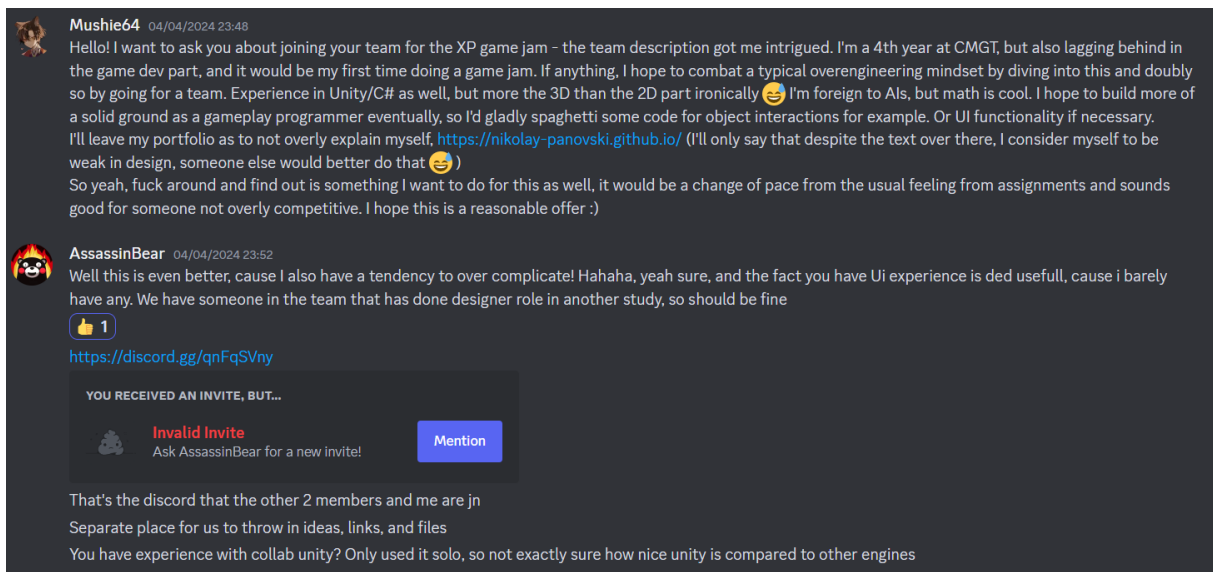
AssassinBear 03/04/2024 23:22
Our current Team:
We currently consist of 3 people: 2x Artist, Engineer.
I myself am a aspiring Engineer with experience in programming, but not so much experience in game dev itself.
The other two are first years of CMGT (Kim already having sent her experiences a bit further up in the channel.) with focus to art.

I pretty much only have experience in Unity3D/2D, with mainly interest to the 2D. So far I've mainly been focusing on AI behaviours. Language would be C# for me mostly, but might be able to use some other OOP language. My weak point currently is the meth , I am already developing a severe love/hate relationship with vectors .

Currently looking for:
Another engineer/Technical (Artist/Designer), so pretty much anyone with experience in coding. Doesn't have to be much, I just don't feel entirely comfortable coding most of the game on my own ;-). If you like to fuck around and find out, we will probably be just fine. hit me up with a DM, or you can dm [@Kim](#) if im not online/available!

Can't wait to work with you! (edited)

Despite that, I felt compelled to make a good first impression and in my first communication connect with the spirit of the offer and supposedly the team. In retrospect (even by looking at the immediate response), the most important part of my first message was the listing of my concrete areas of strength and the parts where I could definitely not deliver or would need help with. What I could improve is the language – my “pitch” lacked a confident tone for both strengths and weaknesses, which may leave unclear for later what types of ideas would be a risk to work on.

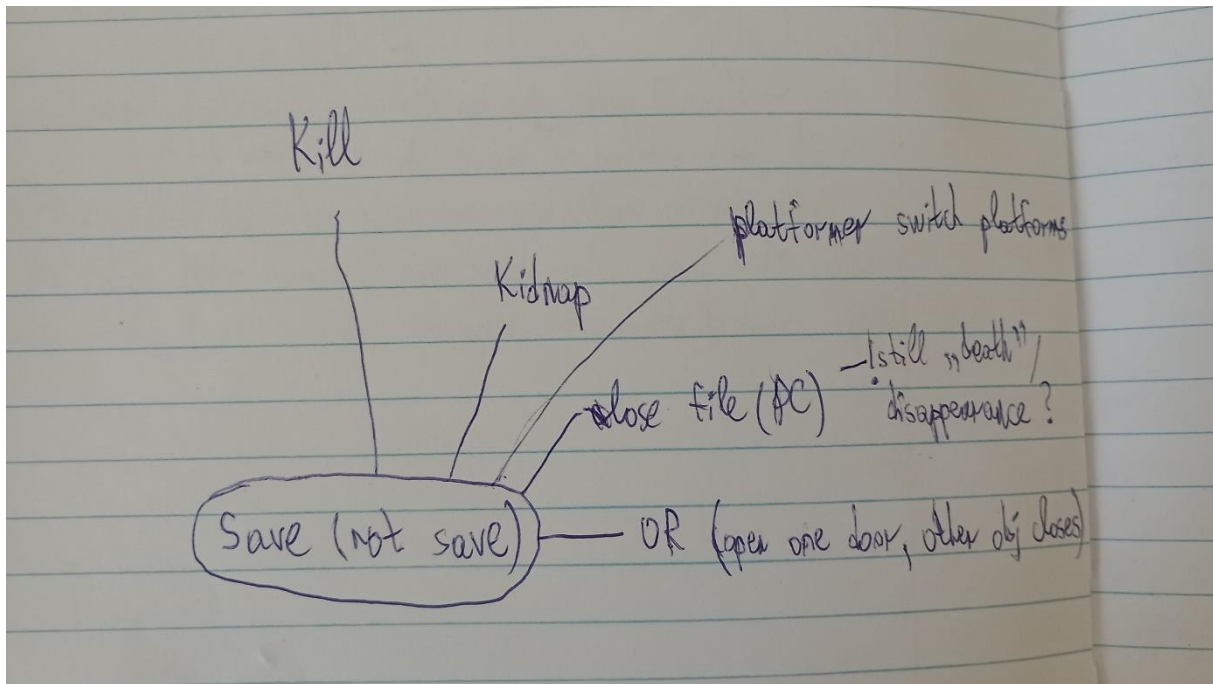


At the gathering time for the jam, despite the provided descriptions from the other team members, I did not know how I would approach them. By default I try to stay formal around new people, however in our first interactions my teammates helped me relax and find a friendlier, common language, mainly because I could now confirm in the flesh that the people taking the same challenge as me have similar expectations and limitations.

Day 1: Kick-off + Concepting

The team consisted of 2 programmers – myself (4th year student, Unity 3D, gameplay/UI), Beer (not yet in CMGT, Unity mainly 2D, interest in AI), and 2 artists – Cat (1st year, 2D art) and Kim (1st year, 3D art). The skill coverage felt sufficiently broad, and being with allegedly the most experience in the group came for me as a mixed blessing – I would feel lower pressure of a skill gap, but I would also feel responsible for the content I deliver and any potential advice I am asked for.

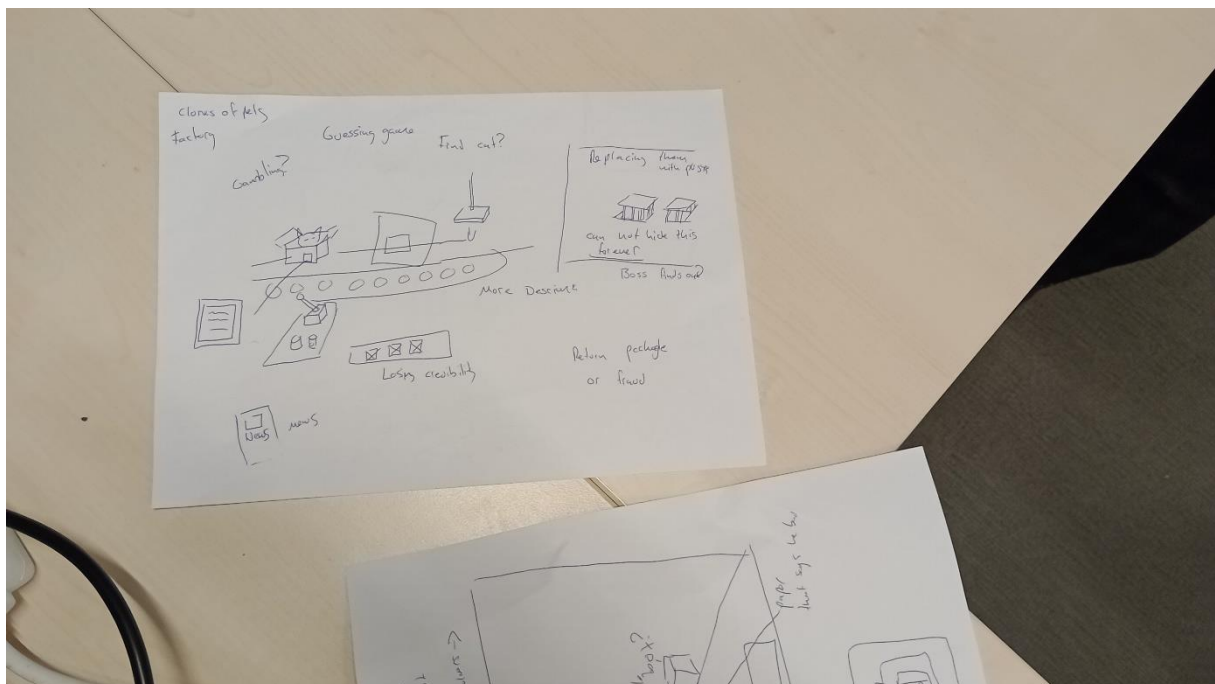
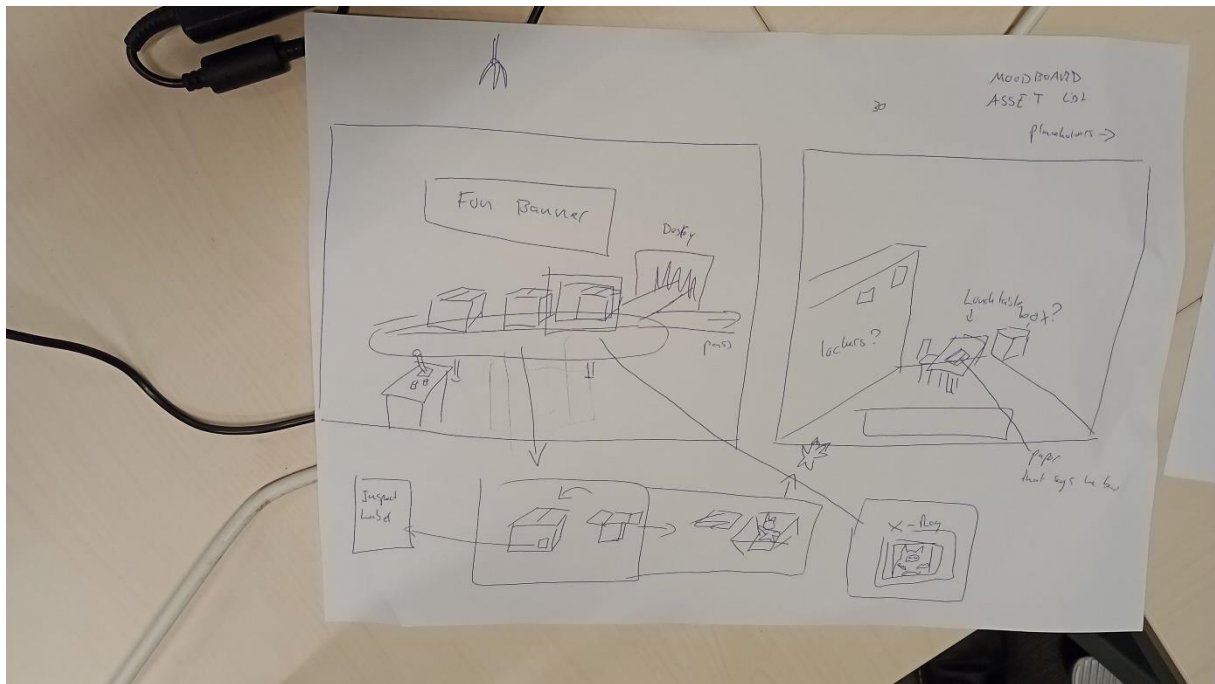
After an event kick-off presentation, the game jam theme was revealed to be **“You can’t save them all”**. The team’s first move was to sit down for a brainstorming session, determined to continue on our computers only after we have agreed on a solid concept. From the beginning, each of us tried to note down their first ideas before continuing to discussion. We all used pen and paper at this point, but our approaches towards ideation (and its expression) differed drastically – for example, I was the only one who tried to approach the theme with a mind map of interpretations of the word “(not) save”; the others were trying to come up with concepts and world settings outright.



When trying to bring ideas together, the discussion became chaotic. We were ready to consider each other's ideas too much at first – we started discussing further details around each idea after its initial presentation. Further difficulties included listening comprehension if more than one person is talking, everyone getting a different perception of someone's idea after its presentation, all compounded with the haste from the brainstorm/jam setting. My approach initially got consideration by my teammates, but fell off since it did not quickly lead to concrete concepts.

After not too long, the team started uniting around one idea: the player would be presented with a family scene and a moral choice, for example serving the unsuspecting people either cake or poisoned food. This would repeat multiple times across rooms of a tower, with each choice becoming subtly more difficult for the player via nuance. We were not certain how we would reveal the consequences and make the player feel responsible for their inevitable evil choices, especially without forcefully scripting those choices. The dominant idea was to end the game with a cinematic scrolling through the rooms in their choice effect state.

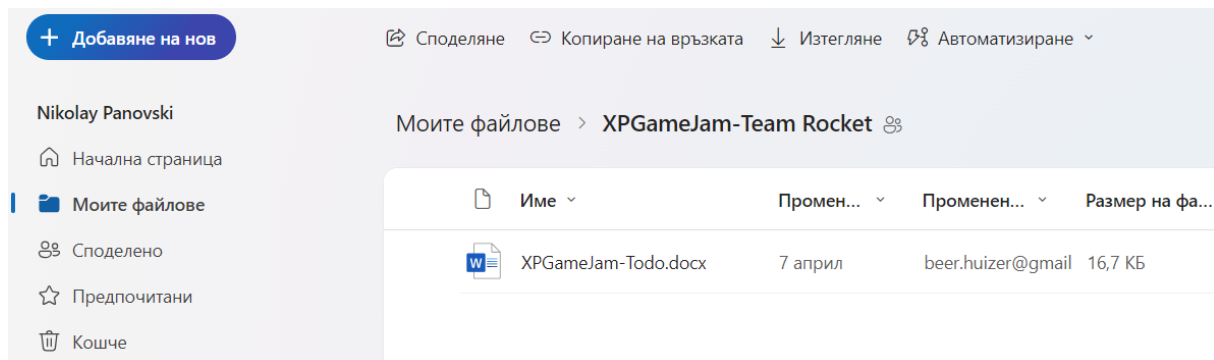
At this point, Beer pushed for a similar idea with a different flavor and more potential for a justified narrative – in a factory setting, boxes of returned pets would scroll through conveyors, with the players deciding after an x-ray inspection which get passed off safely and which get incinerated. The overall idea crystallized to us as similar to "Papers, Please", but the details were still up in the air. The artists sketched their perception of what they heard so far, which was useful to communicate all the details we didn't yet understand, but once again we didn't clearly and completely settle how each of these details will be implemented. However, we took this as a sufficient starting point and prepared to plan development.



One good thing was that we considered everyone's skills and limitations – an even split between 2D and 3D had us mindful on how to best bridge the skill gap for everyone's sake, and we chose to go with 2D asset implementation. Similar considerations drove the style choices like “factory setting with incinerator, but with a fun sounding banner and cute animals”.

Development: initial setup

The programmers' first action was to set up common workflow tools we knew we'd need soon, and in the meantime we could discuss implementation details. We opened up a new Unity project, and set it up on GitHub under my guidance (since I had prior experience with dos and don'ts of Unity in Git, while for Beer this would be a good exercise in the fundamental Git commands). For planning we went primitive – OneDrive for todo lists, without formal tools like Trello.



There were a lot of specifics to be discussed. I focused on the area of UI due to our upfront agreement, and since from prior experience I knew that neglecting UI until just before the deadline leads to a dangerously unpolished product presentation. Beer considered specifications of mechanics and environment. We immediately had different approaches to taking notes, and I had to dissuade him from his preferred table approach because here it would have only mapped one-dimensional information to two dimensions. It took him seeing this in practice to be persuaded, however. And for me, looking back, maybe his categorization would have been useful, for prioritizing or for brevity.

to do (functionality):

UI

- UI - screens: minimal start screen (artstyle not finalized yet; see FNAF start screen for reference)

+ different button / surrounding elements styles (for example text highlight; "> option" style)

- UI - screens: introduction image (newspaper) into "safety" tutorial video - work with fadeouts/fadeins in any case; be ready for other UI animations (tweens)

- UI - HUD: % chance of getting caught by the boss

- UI - HUD: progress counter for number of passed boxes -> ex. day counter

- UI ~ gameplay: dialog boxes?

	UI - INTER	MECHANICS
DESCRIPTION	Interactive	Informative
SCREEN: MINIMAL APP START	V	
SCREEN: INTRO IMAGE (NEWSPAPER)		

Soon after, Beer introduced a mechanic that would finalize a crucial part of the gameplay for all development roles – the returned animals are due to defects in their limbs, and the player has to compare an x-ray with a box label for the “correct” (as dictated) choice. When discussing this with the artists, they considered the defects on live animals to be too gruesome, so we settled on the pets

being clones instead. The player would be given the illusory urge that even the defect ones could and should be saved. We also needed a representative name for the factory at this point, and after I queried ChatGPT for a bit, from the picks the team found “BioBuddies” to be sufficient.

- [prefab] animals with modular joints/limbs (to make missing/extra limbs defects)

Defects:

- broken bones

- missing bones

- oversized head

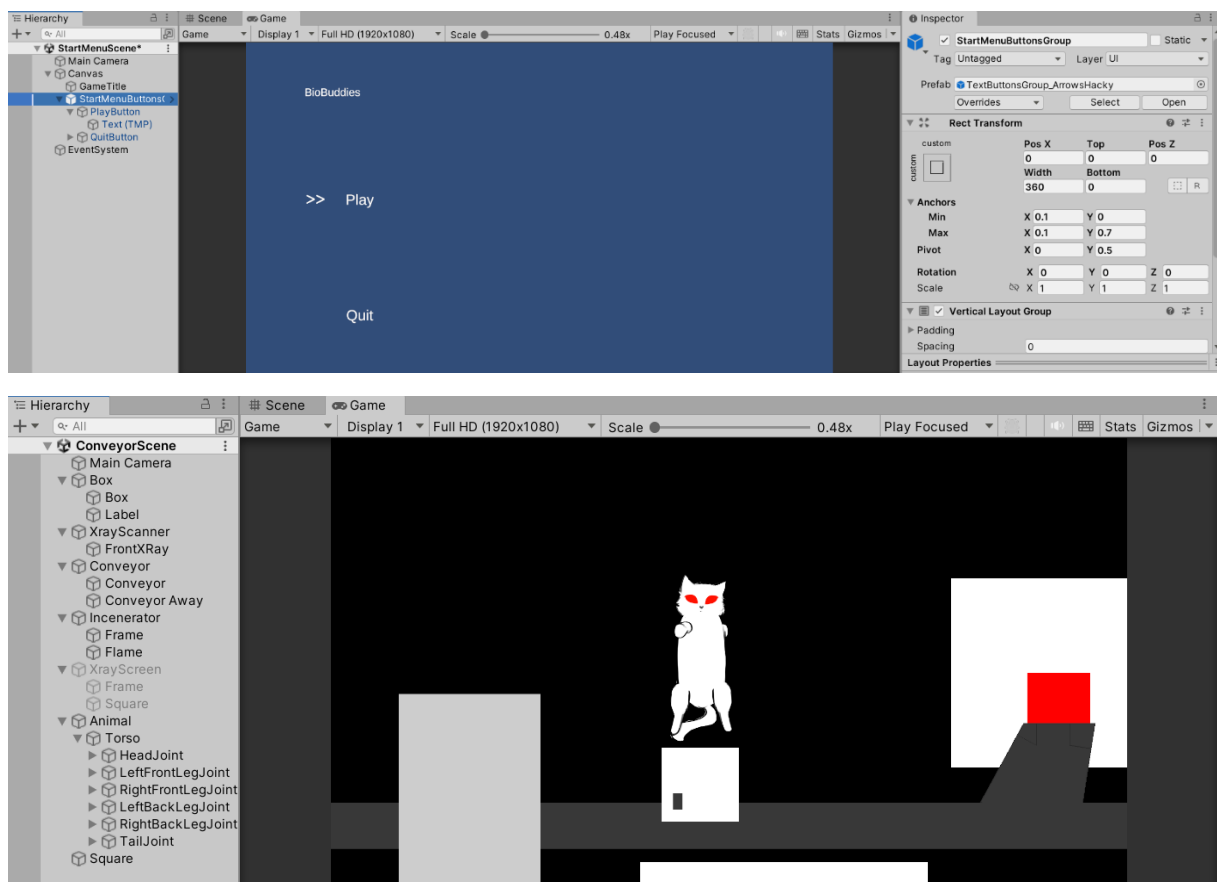
Label mismatches:

- size mismatch

- species mismatch

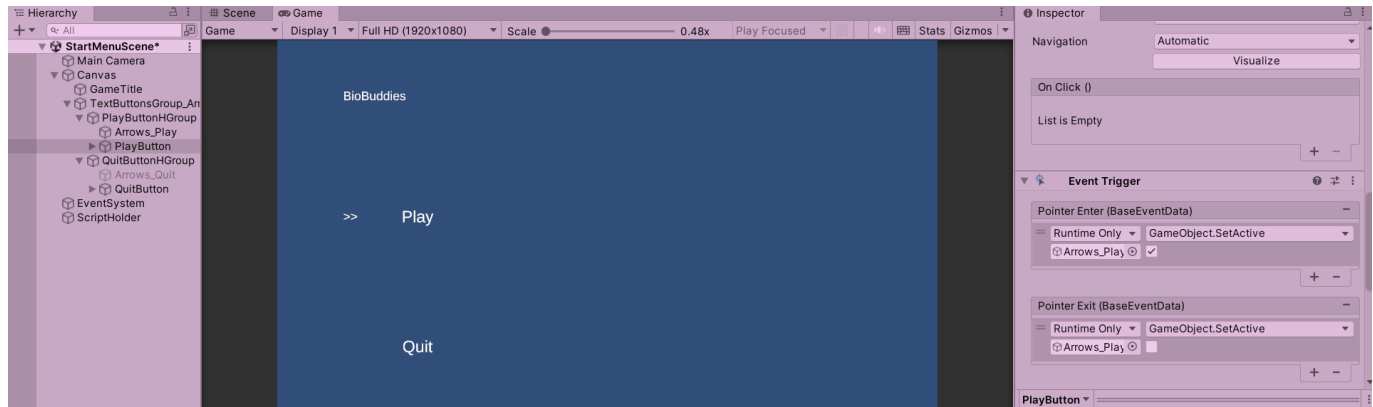
- missing/extra limbs vs label

I sketched a minimal start menu UI with a hacky way to highlight the text “buttons” in the style of Five Nights at Freddy’s. After I finished for the day, Beer first sketched the factory scene in Unity (with cat limbs already done by Cat!).



Day 2: UI and interactivity

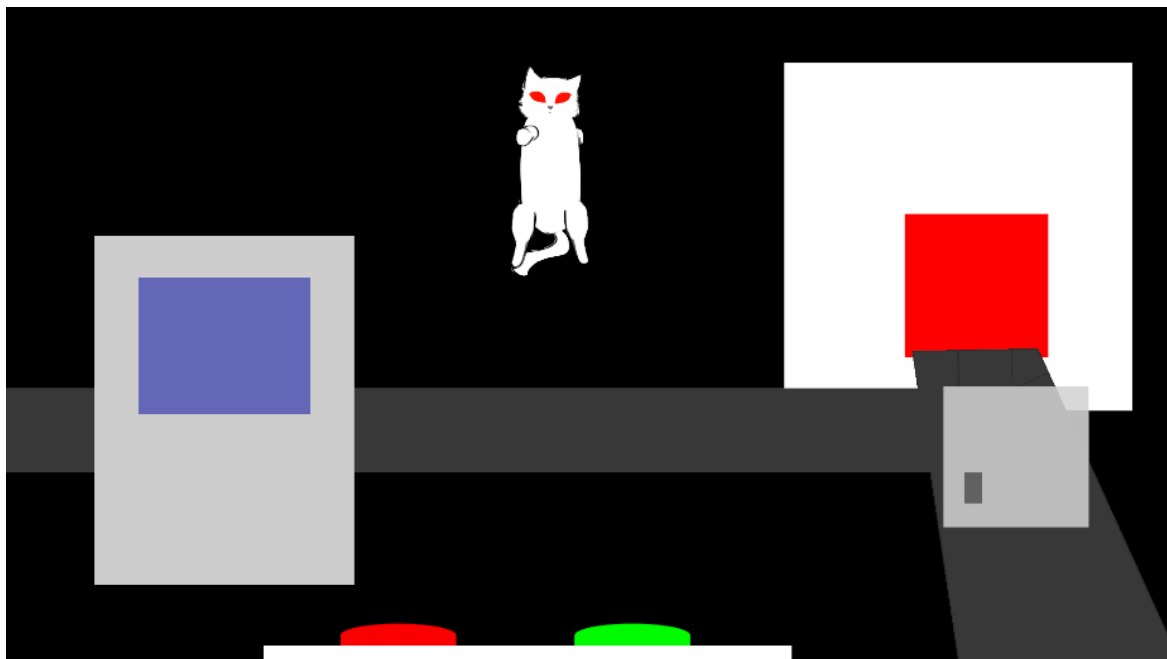
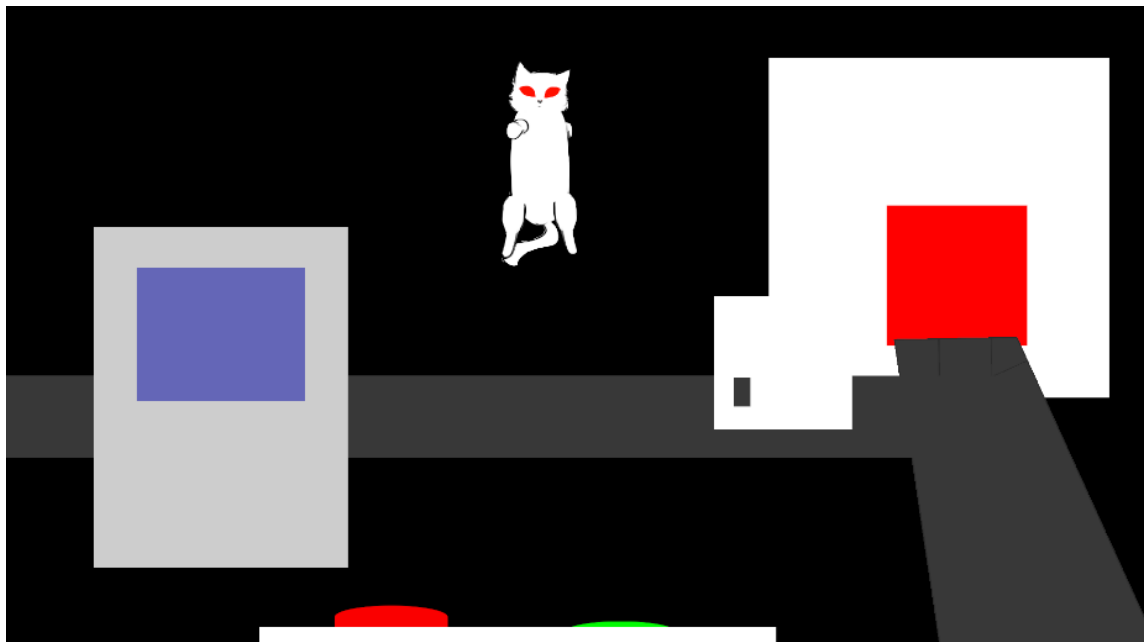
I started the next day with more variations of the start menu UI. I tried highlighting via bold text style and finding a scalable implementation of the earlier arrows. I couldn't control Unity UI's Layout Group components sufficiently (as usual for me), so "scalable" became "unusably buggy and a time waste" and I reverted to the original hack of toggling a static image of the two arrows on hover.



I also added a generic newspaper image over the full screen to prepare for scene management with additive loading (since we were already discussing starting the game with a "worker training video", in our case a comic strip the artists would be able to do on time).

It was already noon as I finally moved on, this time to the main scene. Even with many details still not clearly unified across the team, I knew that boxes would have to move through the scene and disappear, depending on the player interaction moment with choice buttons (keep or burn). I imported my favorite tweening engine, DOTween, and made a whole Inspector-based and callback-based animation setup for every transform property and color to cover all potential bases. At this point Kim was working on a 3D render of the scene with higher-fidelity elements in Blender (which we had agreed upon to produce a 2D background from in order to accommodate her 3D-based skillset), and we noticed the conveyors should have been placed differently, which I then corrected in my scene with placeholders.





Throughout the day, Beer was the adventurous one in scripts, dealing with representing animals in data and instantiating them based on the concept with defects. He wasn't shy about asking me code questions, and despite this being his first experience with Lists and Dictionaries and handling randomness, he bent them to do his bidding for animal parts (at the cost of unbelievably brittle code, as I would find shortly when I would need to pay full attention to his code).

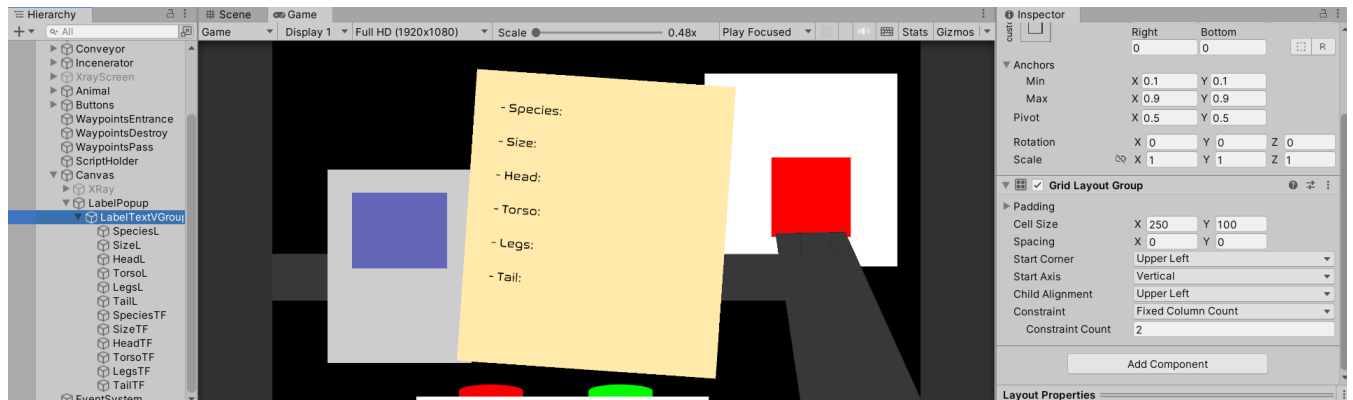
I had an evening break after my animation struggles, and after returning, I was feeling pressured and behind on the many crucial gameplay parts still missing. I pulled through the box label popup as the next major task and placing down as many existing or static UI elements as possible. The font that Kim found boosted both the game feel and my morale for this.

nasalization-rg.ott

BioBuddies

Note of the author

Nasalization is an ultramodern sans serif typeface inspired by the NASA logo from 1975. It's the go-to typeface for depicting science, futurism,



I was nearing the point where I could not do much more without access to Beer's animal mechanics – the stats for the box label would depend on the actual spawned animal. It was time for a code merge and a code review. With no merge conflicts to talk about, I still found some uninitialized values that I needed to debug before the scene would run properly with any animals.

```
[Tooltip("The head GameObject to use when the defect is a duplication")]
[SerializeField]
private GameObject dupeHead;
private Dictionary<GameObject, GameObject> damagedPartsDict;
private Dictionary<GameObject, GameObject> damagedPartsDict = new Dictionary<GameObject, GameObject>();
private AnimalDataController data;
private List<GameObject> partList = new();

// Start is called before the first frame update
void Awake()
void Start()
{
    SettingSetup();
}

void Start()
{
    SpawnCat();
    data.InitializeValues();
}
```

There were still mysterious bugs with missing body part spawns afterwards. I gave a fair shot at following the dictionary dependencies and range values in Beer's scripts, but could not find the causes of bugs myself. Here we are thankful to the fellow jam participants who stumbled upon our project and offered debug help and tutoring to Beer when I wasn't free to dive deeper myself. We discussed a recap of the currently progressed features and the expected ones for the deadline, and I had to firmly express my concerns that the list is feasible until we cut down whatever we could and prioritized the rest (during which the whiteboards on location received much more abuse than the initially prepared OneDrive).

In my eyes, we were still very behind, so I coded a grabber for the animal information text upfront. The caveat: There was no way to test whether my script helps at all before the other code was fixed. Afterwards I went to sleep, while he pushed on to get more advice and rework the code through the night.

Day 3: Fixing code and game loop

Going into the last hours of the jam, I was both very tired (common across the jam participants) and conscious of how behind our progress is, therefore in a state of trying to finish as much as possible as quickly as possible (which I label "crisis mode"). Beer got the effects of his code on the screen (finding an awesome SerializedDictionary plugin on the way) and I patched the label grabber to the new dictionary syntax. Even then, the features we had wished to ourselves had piled up and now was time for another firm but quick discussion to establish that we are not doing big and extra tasks like easter eggs, blur effects, dialog boxes, remaining with a list simply categorized as "Sunday".

to do:

UI

- HUD: % chance of getting caught by the boss
- HUD: progress counter for number of passed boxes -> ex. day counter
- UI ~ gameplay: dialog boxes (ala Undertale, Celeste...)?
- popup toggles (label, xray)
- blur effect during open popups (label, xray)
- + blur color matches the color of the box

Mechanics

- [Animal] Joints = Empty objects on predetermined positions.
- box spawner
- if decided to save: zoom box + head room -> cat comes out -> cat infront of box
- 2 Easter Eggs: Jeff the cloning liquid, Ginger Cat Jesus (crazy trigger like "after saving 10 ginger cats").

Art??

- As the level progresses, the animal starts taking different postures in xray machine (done modular aswell!). - NAH F--- NO

label mismatches – cancelled

Sunday.

=label grab finalize

3 strike system for “game over”

=actually finishing the defect

- the 3rd “save animal” option

+ on ??? trigger – “save” button (UI)

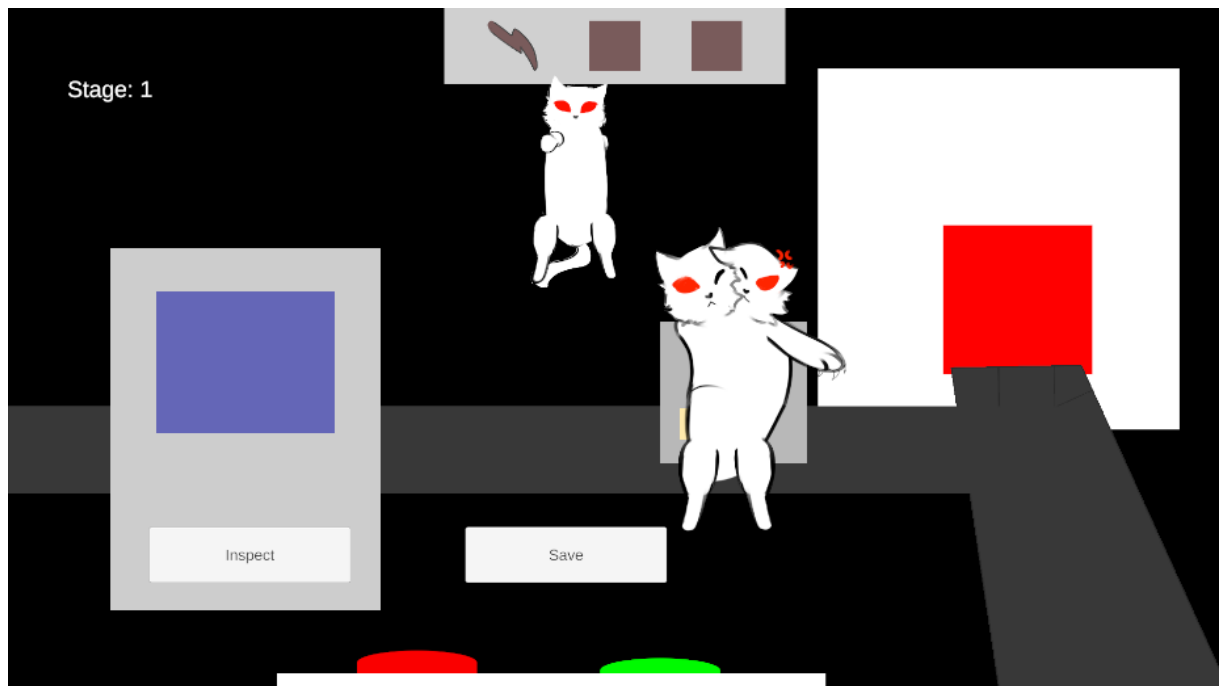
- second “break” room (transition: button UI)

+ 2 overlaid PNGs of the room/background, sliced in 6, with “holes” in boxes highlighted by sprites to hide cats in by click -> remove overlaid PNG to show final one

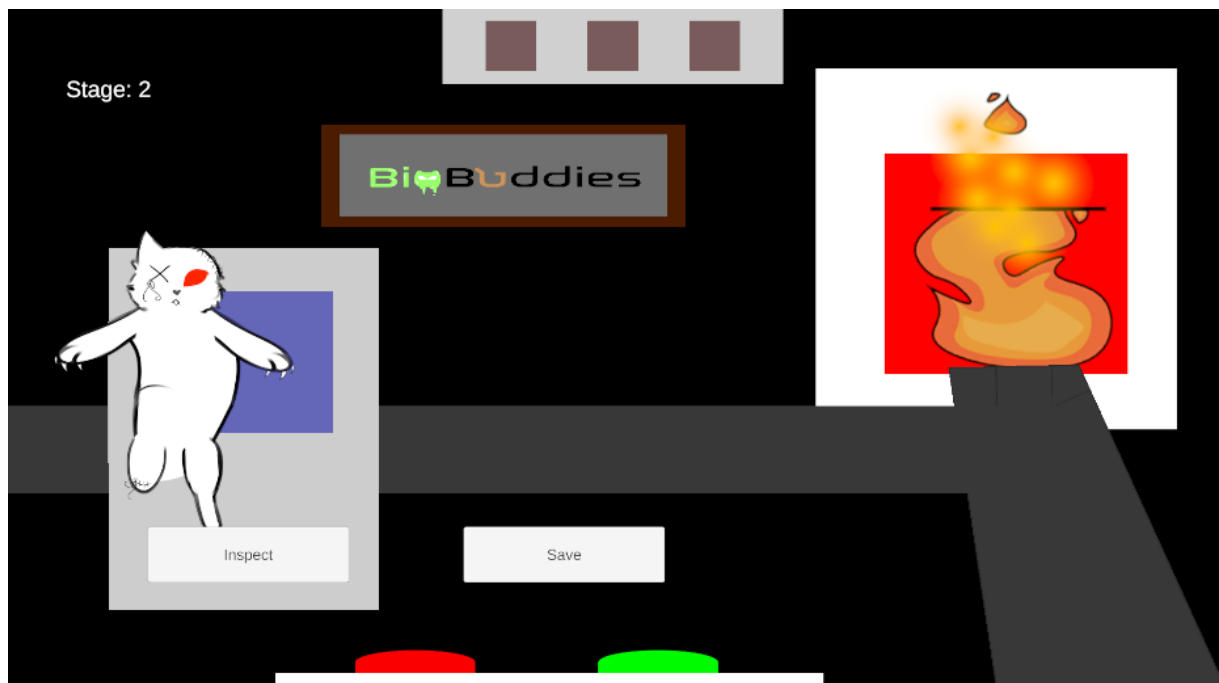
- x-ray [Kim]: physical in scene -> on passing through enable blurry default image on the small monitor -> enable x-ray popup on clicking on the monitor

- game over/ending screen???

Some friction still remained – for me the list was still definitively too big and in the remaining couple of afternoon hours we would be able to do only very small features and likely not even bug test, let alone playtest, while Beer fixated on his earlier idea for a second employee/storage room in which the narrative to the animal “saving” would resolve with a forced loss, since at least that was established earlier and should have made it in. We were unified in one thing – the most productive thing we could do while this deep into overscoping was to keep working, and that’s what we both did. I focused on the missing fundamentals that would barely provide a valid gameplay loop. I added minimal progression counters, notably one for immediate strikes if the wrong choice was made between “destroy” and “pass” against the provided labels for defects. “Inspect” and “Save” buttons also found their place on the UI, however I had little hope their functionality would actually make it in. I also adjusted the labelling and defects system to the last data format change.



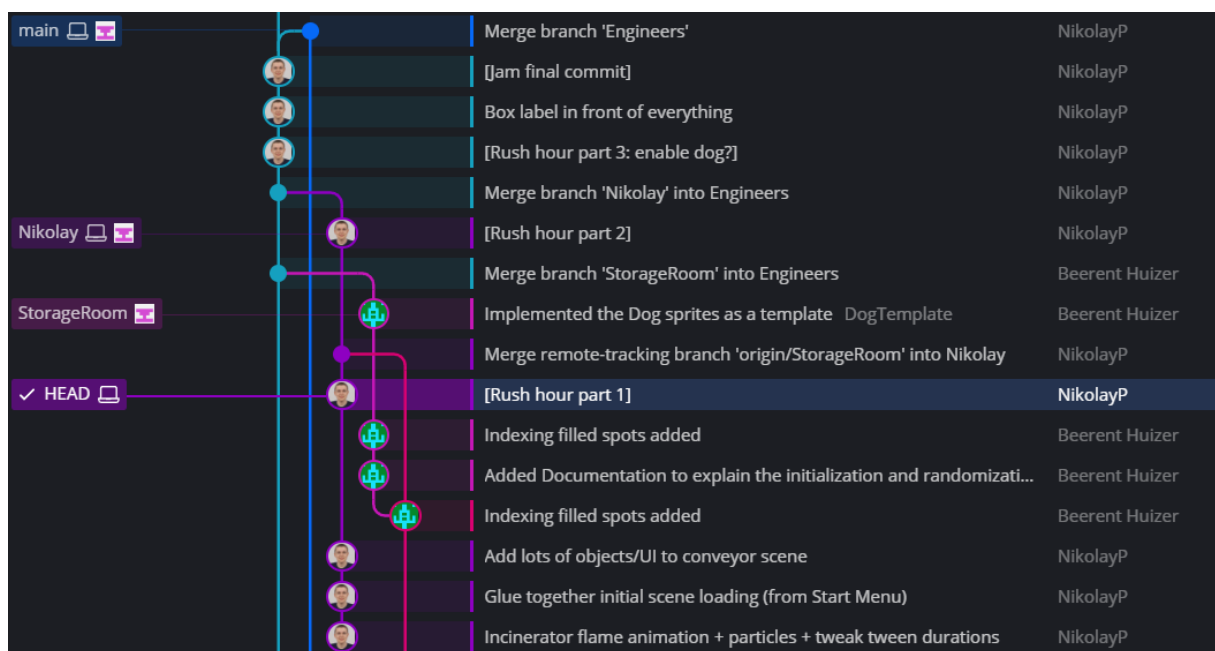
At this point, the artists were getting tense about their own assets getting implemented (since they were mostly idling from the end of the previous day), but also tried to encourage me to take the remaining time more chill than a deadline (I had already been expressing my concerns the previous day that I'm falling behind, we have overscoped, and features *will* have to be dropped at the end). Oddly enough, this did have a positive effect on me and I put many artist assets to good shape in the main scene:





The final hour

We were now 1 hour away from the submission deadline. For me, all changes blended together at this point, and the time to keep thoughtful Git commit messages had passed.



Within these, I updated the box and label images, the choice button images (where Kim helped by providing images of them cropped as I requested on time), the “inventory” counter of strikes, and added Cat’s death animation, which unfortunately bugged and stayed on the screen after the last frame, and that was time.



Even though it should have been over now and we could only stand up to what got in until this moment (we had a dedicated space in the open area to present our games), this game jam was loose enough to where we could patch things in place during the allocated final playthrough time. Beer tried his best at the storage room (presented below) and was unhappy already in the early afternoon at the notion that it would not make it in because I couldn't link the events of the main scene with that other scene in the remaining time.



And the artists saw numerous bugged or missing expected elements when I set up my laptop for the open playthrough, which made them ask that I add things in quickly if possible, in order to not scrap their work and drop the quality of the game. I shared the sentiment since we even hit the deadline crime of not building the game to an .exe, but I wanted to treat the development time as finished, however, I obliged. The game now had a start screen, a “death” screen and the background audio was enabled. I also made a build so the game could run fullscreen and without the Unity Editor around it in the eyes of players.





I committed these final changes 1 hour after the de facto jam deadline and 1 hour before the playthroughs and voting for the end of it were over. I saw people try out our game, and in either forms, be confused about the gameplay loop (because progression in it actually didn't work – a player could easily “save” infinite animals and just as easily lose after 3 strikes without any proper indication of what they did or why), yet still press the buttons for much longer than I would have expected. It was an underwhelming experience and by default an unfinished game with some unused ideas and assets, but thanks to these final efforts, not a complete player catastrophe as I imagined it.

Result

At the end, a team of 4 people with wildly different skillsets and no prior jam experience put together (most of) a “Papers, Please”-like game ensuring tortured animals and inevitable loss of some of them to the system (per the jam theme). The ideas for the game were ambitious, while the technical execution was quite basic and quite buggy/unfinished so that it affected comprehensibility for the players. I wish I could present this among the other few games in my portfolio, but I'll refrain from it since I do not believe it shows my full potential even by game jam standards. However, working in a sporadically formed team was an enriching experience and I don't regret committing to it. I don't know if our paths will cross again, but I don't mind doing another jam with this team.

As for the “missing” parts of the game versus the conceptualized ideas:

- Introductory/tutorial cinematic (comic strip?) screen in the style of a safety regulations video
- 3 *functional* choices per animal (box) – incinerate, pass, or save
- Reveal of a hidden procedure where “pass”-ed animals are actually still caged or otherwise disposed
- Implementing the “save” option and the storage room after several processed animals and the realization of the above
- Overall narrative of the above, with a boss figure who eventually catches the player after all strikes or filling all free spots in the storage room / dialog boxes
- Gameplay loop: implementing actual inspection (x-ray?) of the boxed animals to match against the labels + animations (on “Save” the box opens and the animal comes out ala chest loot)
- Easter eggs (see the early todo list)

Maybe here we have to admit to overscoping from the beginning and piling more on it later. Ambitious people are welcome to continue on these ideas (Beer expressed willingness to expand beyond the game jam).

Communication

Similarly to the development process of the project from my perspective, the communication can be summarized as “barely not enough or barely not effective most of the time, but effective in the most critical moments”. I acknowledge that our approaches to team communication were not extraordinarily wrong during the first day, since we had to get used to each other on the spot both as people and as skillsets. The initial brainstorming started fine, so did the initial setting up for the project. The first moments of lacking communication appeared later in brainstorming and setup, and can be mainly expressed as (lack of) team cohesion. Listening to each other’s first ideas was a good start, but shouldn’t have been done so in-depth; on the flipside, we should have spent more effort making sure *explicitly* that ideas and tasks were understood by everyone before committing to them and especially before physically separating. Beer and I fell victim to the classic engineer trap with technical specifications – we didn’t know what we could do in detail, so we couldn’t convey it to the others in the team either, and we couldn’t be held accountable nor get pulled out of overscoping. In a way we mostly covered up that we didn’t know what we were doing, so the artists put blind trust on us and completed the trap by separating their own work (since artists and engineers don’t depend on each other... right? – they do). This was furthered by the artists’ physical move to another room due to their preference for quiet versus Beer’s preference for a loud, chaotic environment (I prefer a team without splits regardless of whether or how often we need to interact, but there was no satisfactory solution in this case, and my preference was the least reasonable one to insist on.)

As for my own communication during the development period with Beer specifically, I believe to have done a decent job in explaining the fundamental workflow concepts that I needed to as the more knowledgeable one, and in nurturing a mindset of doing things for a reason. (For example, why spend more time on Git commit messages and branches? To be able to return to what we did later!) I only hope this “early headstart” is of value without mixing up with the starter content of the CMGT study. What I could improve in is firmly setting my foot down both myself and others when I notice signs of vagueness and overscoping. For example, the first programmer discussion while setting up was about the details of making animals modular and giving them defects, in which I already felt lost since I couldn’t imagine the whole idea in my head. I should have spoken up at least for more explanations of the gameplay flow during brainstorming. Instead, I made UI my first focus because I silently fell back to the area I have *any* concrete vision about.

Even during the jam I felt bad that I could not hold proper (work) communication often with the artists, and I believe that to be on me. As work is going messily and my feelings progressively go deeper into “we won’t make it like this”, it seems hard to present these feelings in a constructive way and easier to pretend that things are struggling but ultimately fine and receive peace and quiet for the moment. The truth surfaced in the last few hours and communication picked up on both sides in both pace and constructiveness, with features more clearly demanded, respectively either implemented or rejected. This should have stretched to the entire jam period. Still, I have to be thankful that the others kept a friendly tone and did not hold the decline of implementations at the very end against me despite their urges. This persuaded me to even try and not act like it was all over in defeat (even though we had technically crossed the deadline).

Looking back

What went well

- The whole mood and atmosphere of the game jam. I did not feel any *external* stress that I had anticipated before starting (team pressure, competitive environment, expectations for product and working hours by the deadline); I mostly had uncomfortable sleep, otherwise it was me riling myself up about things typically *punished* in school projects.
- The initial adaptations within the team to make sure ideas are heard and understood, and everyone does tasks they are comfortable at.
- The encouragement and friendliness across the team in and outside of work (in break times). I perceived only support, not negative feelings towards myself even in the crunch moments, and it helped me be productive for a bit longer under unreasonable physical conditions.

What I would/should do differently

- I need to speak up in the first moment when I sense overscoping in an idea or am otherwise not on board with imagining it, like in this case. Many following problems are symptomatic of an idea that should have been cut down at the start, and solving them is harder and helps less.
- I need to be clearer about things that I notice in advance will end up unimplemented or cut off. This includes overcoming the fear that by doing so I would sound incompetent or lazy, which instead makes me sound unconfident about the actual dangers and get ignored.
- Task definitions – people don't write technical specifications during game jams, but since we had no dedicated designer or manager, someone should have taken up the prioritizing of tasks and features. I feel responsible for this mainly as the most project-experienced one in the team at the time, but if I hadn't done this, we could have at least set up a Trello workflow.
- I could probably benefit from taking *more* breaks and spending less time, especially in the night after a long day, solving problems with varying effectivity. It is hard for me to switch my mind between work and break states quickly, but after a certain point grinding work, the output is uselessly low.

In general, I should try this kind of experience again another time! However, I am unsure of what setting I want to be in, since working from home allows for more comfort, but working on location like this time gives me a higher drive thanks to the environment.