Team 2 Group Report

Game Technology Center | ETH | ZHdK | 31.05.2019

Sophie Walker- Production, 3D Artist

Sven Knobloch - Lead Programmer

Julian Bauer - Art Direction

Leonardo Del Giudice - Algorithms Programmer

Nihat Isik - Visual Programmer

Julian Schönbächler - Technical Artist, Game Designer

Table of Contents

Chapter 1. Formal Project Proposal

1.1. Game Description

Hazmat is a game about tradeoffs, it will show players how every choice they make will potentially kill or save them. The concept can be categorized into the survival genre - everything out there is trying to kill the player, and they'll have to make choices about where to go, what to do and what to use to survive. The player's energy is constantly being drained by the actions they perform. Every action has different cost, so players have to think twice about their actions. The unforgivingness of the game, combined with a procedurally generated map, should allow for a lot of replayability. Players are able to try different strategies every time.

1.1.1. Overview

The game will be a 2.5 dimensional wasteland survival, with the possibility of cooperation (***see chapter 1.1.3.1 Core Mechanics***). Players are fighting to stay alive by killing and looting enemies. A very simple crafting system allows for the combining of looted components, but be careful, as more powerful gear will drain energy faster!

1.1.2. Background Story

In 2099, we are in the middle of a cold cyber war, which is about to escalate. In everyday life, people heavily rely on technology in their homes, vehicles and daily interactions. This dependency is targeted with a nationally coordinated cyber attack.

The player lives in a small city and works at IT security of a nuclear plant. They are woken up by a phone call (they were up late playing VR with their friends so they slept through the city evacuation) and are informed that the city is suffering a cyber attack. Buildings and public systems in the city are under enemy control. Public bots and drones that facilitate daily tasks have been overridden and turned against citizens. Worst of all? Someone hacked into the power plant to initiate a meltdown. All security has been shut off and radiation is spreading.
Now, the player is the only one left who can reach the power plant, stop the hacker and manually stabilize the core to contain damage.

Game
After the phone call, players put on their hazmat suit and try to reach the power plant as fast as possible. With the city under enemy control, the phone call is being intercepted by the enemy, who is now using the city's AI to sabotage the mission. Players are instructed to gear up with weapons to defend themselves against hostile AI. As soon as players step out of their house, they are attacked by drones and robots. They pick up their weapon of choice and hurry towards the nuclear plant. In their fight against enemy AI, they are on their own. Since their suit runs on energy, they have to destroy hostile technology for their parts, or activate public charging stations to remain protected from radiation.

When the players finally reach the power plant, the plant's defense system is under enemy control and tries to prevent them from entering the building. Once the defense system is defeated, players overload the reactor's mainframe to end the hack and stabilize the core.

Setting 2099

- Digitized daily life
- High reliance on technology
- Water shortage (dry environments, a lot of technology, not a lot of plants)

Player

- Clumsy
- Wobbly

1.1.3. Design Decisions

1.1.3.1 Playthrough

A. Story introduction: phone call

Someone hacked into the city's technology hub (all "smart houses", automatic cars etc) and initiated a nuclear meltdown.

B. Game screen: Player house

- Energy starts draining
- Timer starts
- Character wears suit. Suit has low energy
- Futuristic UI indication on where to go

C. Gearing up (Walmart)

- Grab a gun
    - First come, first served, guns within game world (pick up with X)

D. First Enemy: Micro tutorial

- Killed enemy drops a battery (refills energy)

E. Exploring

Possible encounters

- **Enemy (camps)**
    - Can be avoided
    - Getting closer to them will aggro
    - If defeated resources will be dropped
- **Checkpoints/Hotspot** work as a *one time* refiller for player energy
- **MiniBosses** (Extra)
    - Skilled players may find a challenge by finding and killing minibosses

World Rules

- **Suit** drains faster the closer the player is to the power plant
- **Enemies:**
  - When killed: drop items and/or energy
  - **Item:**
    - Can be used to improve one stat
      - Popup, time stops for a few seconds and forces a choice. Otherwise the item is lost.
  - **Energy:** recharges suit
- **Loot** disappears after a while

F. Endboss

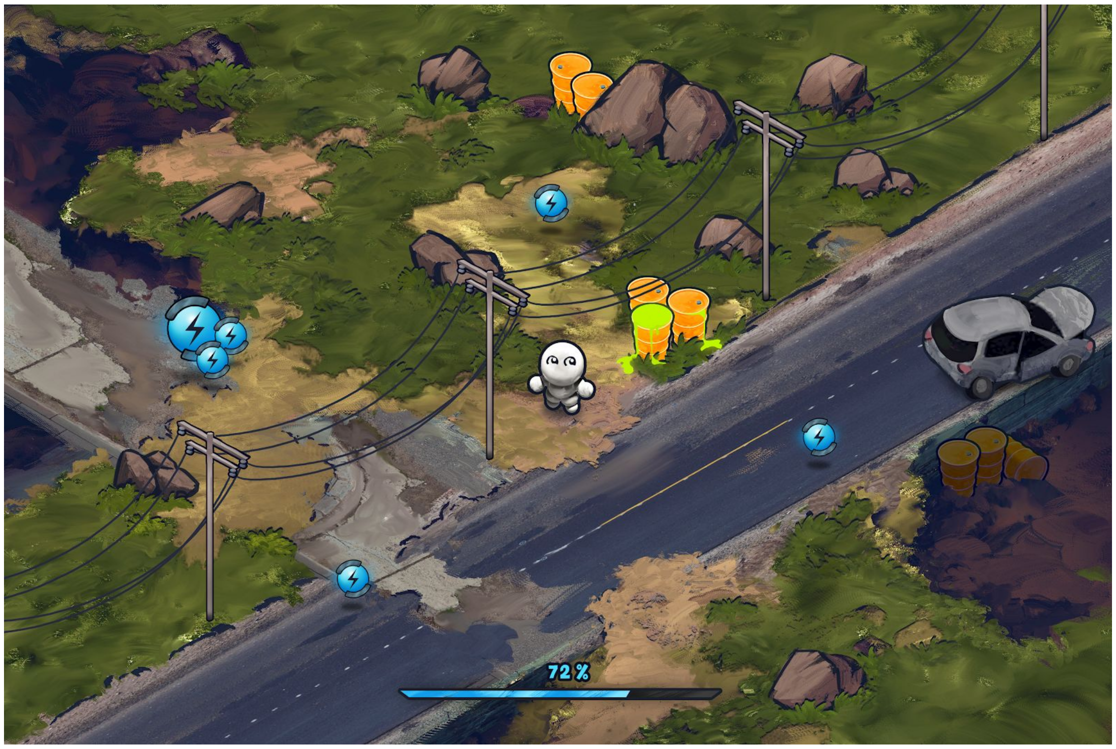
- The plant's security and defense system has to be defeated
- Overloading mainframe to kick out hacker and reboot the system

G. Stabilizing plant

- Interaction with the core to stabilize the plant
- Players use their own energy to stabilize the core

H. Game complete

- Victory screen
- Player stats



1.1.3.2 Core Mechanics

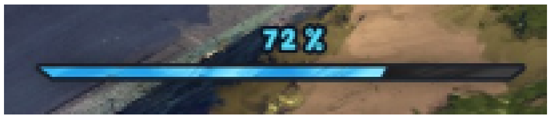
Reactor Stability

During the game the UI will show the Reactor's Stability constantly decreasing. The stability level of the reactor influences the speed at which the player's hazmat loses energy, the lower the stability the more energy per second will be drained. This has the effect of an ingame timer without the necessity to show it explicitly.

Combat

The core gameplay stems from the combat systems. This will resemble a twin-stick shooter with multiple weapons and upgrade paths. Enemies have different types of melee and ranged attacks and will try to damage the player until the player runs out of energy.

Energy System



The supporting mechanic is the player's energy system. The player has an energy total that acts like a health system. This energy will drain over time, slowly at first, but scaling with the players proximity to the power plant and its stability, requiring the player to take action. This results in a fast-paced gameplay. In addition, all actions will cost the player some amount of energy. This forces players to be resourceful and motivates fast decision making. Energy can be restored by collecting various pickups that can be found by looting. There will also be upgrades that alter the behavior of the energy system, such as increasing capacity or efficiency.

Looting

Items in the game will be acquired via a looting system. Enemies and lootable objects (i.e. chests) will drop randomly generated items that can be used in crafting. The value of the items dropped will scale with the difficulty of the enemies and locations to reward players for taking on more difficult enemies and exploring dangerous areas.

Crafting/Upgrades

The game will feature a small crafting/upgrade system to further the tradeoff theme of the game. Players will be able to use the items they acquire while playing to combine and upgrade their gear (primarily armor and weapons). This gives a sense of progression since the player can grow stronger with better equipment. For each acquired item, the player will have a limited time to choose what piece of equipment to upgrade, with the risk of losing the upgrade after waiting too long.

Scoring System

Certain actions in the game (like slaying enemies) will reward the player with points. A running tally of the score will be shown on screen to encourage the player to continue playing and increasing their score. Scores will be listed on a leaderboard, with different categories for number of players.

World Events/Side Objectives

In order to add to the immersion/story of the game, the world will feature some small, procedurally generated “events”, like clusters of enemies guarding some loot or a more difficult enemy that the player can fight for better loot.

Cooperation

All players share the same energy system. This creates a need for cooperation amongst the players since one player performing poorly can end the game for the entire team. Players will need to decide what the best approach to progressing is, whether to move forward individually to cover more ground and find more loot, or to band together and protect one another to succeed in more dangerous areas.

1.1.3.3 Gameplay

Fast-Paced/Survival

The feel of the game should be constant action. The constant pressure of losing energy forces the player to explore and search for loot in order to survive, forcing conflict and dangerous situations.

Short Playtime

The game itself will have a targeted playtime of under 30 minutes. This allows players to experience the scope of the game relatively quickly but continue to explore the game with multiple playthroughs.

No Persistence

All state that the player has at the end of the game (items, weapons, armor) will be lost after they lose. This creates a new experience every time a game is played and encourages the player to try out more ways to play.

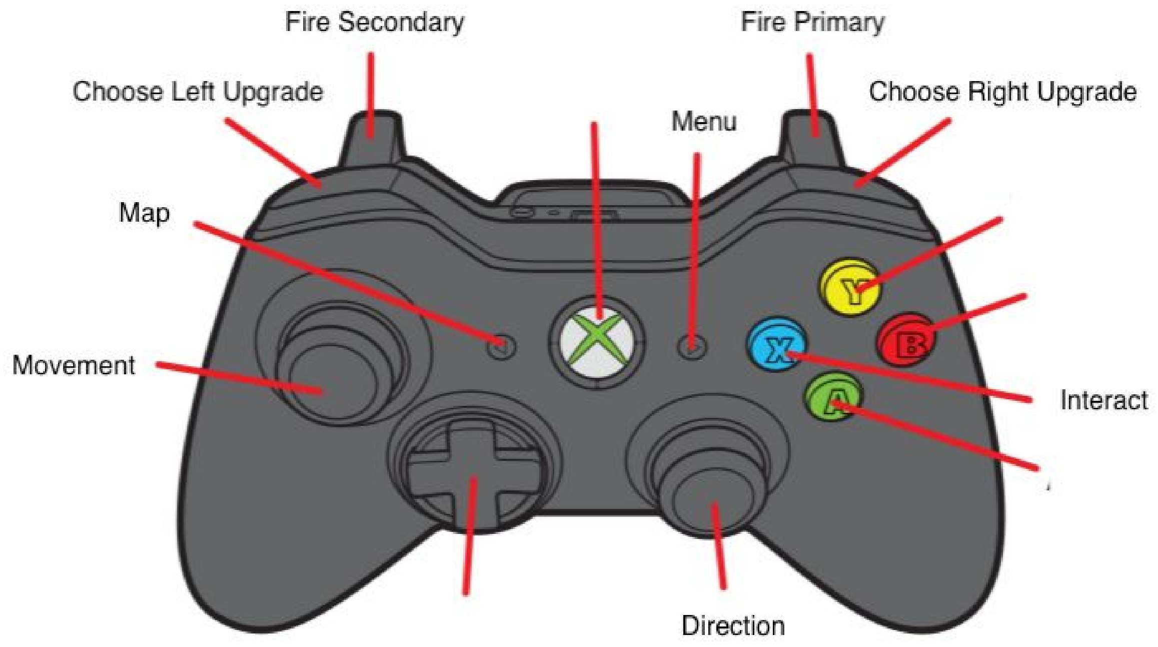
Replayable

No two games should be alike. Each instance of the game should have a different sense of progression because of the random elements from looting and procedural map generation.

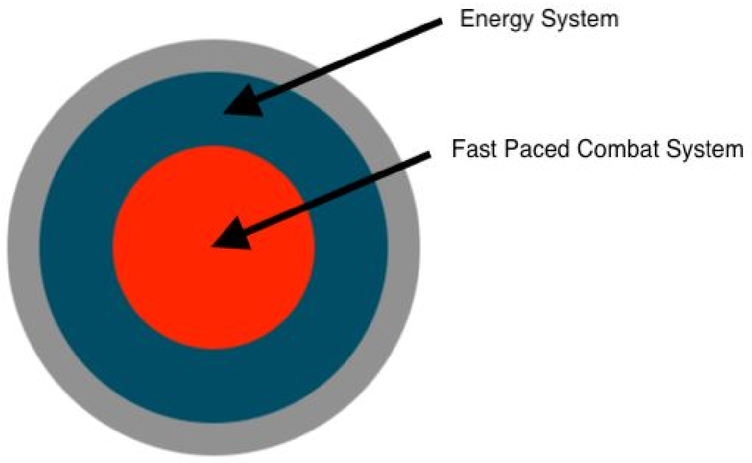
Progression

The player should become more powerful over time by obtaining and crafting better weapons and armor. This enables them to explore further on the map and face more difficult enemies, which repeats the progression cycle. The environment will also change as the player moves, indicating progress is being made.

1.1.3.4 Controls



1.2. ‚Big Idea‘ Bullseye



Primary Mechanic

The primary mechanic of the game revolves around the combat system. The player should be engaged in a fast-paced environment that gives the player a thrilling feeling and a strong need to survive. (**See Core Mechanics (1.1.3.2)**)

Supporting Mechanic

The supporting mechanic is the energy system. It helps make the player think more about what they are doing and if they can afford to if they want to stay alive. For multiplayer, the fact that the energy systems of all players are tied together adds a need for cooperation and forces the players to think about new and more advanced strategies. (**See Core Mechanics (1.1.3.2)**)

Technical Components

Procedural Generation

The map is procedurally generated (***See Technical Achievements(1.3)).***

Artificial Intelligence

An artificial intelligence will be implemented to control enemies (***See Technical Achievements(1.3)).***

1.3. Technical Achievement

Procedural Generation

Each game should have a brand new world to explore, which will be achieved with procedural generation. This will be used to generate not only the environment that the player can explore, but also the enemies and loot. The challenge will be to implement the algorithm so that the world looks natural and continuous.

Environments

- Residential Area
- Forest (procedural)
- Power Plant (not procedural)

Artificial Intelligence

Since the game has enemies, some artificial intelligence will be needed to control these enemies. This will require some implementation of decision/behavior trees or some other sort of state-based decision making, as well as some other challenges that come along with artificial intelligence, such as pathfinding.

1.4. Development Schedule

After the initial kick off, our team has 12 weeks (56 work days) to develop the game prototype, which will be publicly presented on May 28, 2019. Students from ETH earn 10ECTs for this project (= 250h = 20-21h/week), while ZHdK students are expected to contribute work reflecting 2ECTs (= 50h = 4-5h/week).

The team will work in project cycles where working prototypes are iterated on and improved. Targets for each cycle are described in sections 1.4.1 to 1.4.5. A detailed planning for technical and asset deliverables has been created for every development target (more in 1.4.2.)

1.4.1. Layered Task Breakdown: Development Targets

1.4.1.1. Functional Minimum

Gameplay: The bare minimum requirements are a functioning energy system with one player associated to it, that will spawn in a 2 dimensional plane and is able to walk around and shoot basic enemies (no AI implementation). Some assets and the basic UI is implemented.

Art: Simple 2D pixel art. Abstract pixel art allows the artist to create a lot of assets in a relative short amount of time.

Audio: Basic sound effects for supporting the most important feedback systems of the players actions. Mixed from various audio samples, no own recordings.

1.4.1.2. Low Target

Gameplay: In addition to the functional minimum, the low target includes some rudimentary AI that will try to engage the player in some combat actions, the basics of the crafting system and a basic map, including the power plant, via procedural generation. The player will have a UI with a timer and direction indication to guide them to the power plant.

Art: Abstract comic (cell-shading) look.

Audio: Providing audio feedback for all the players actions, as well as UI/UX interface sounds.

1.4.1.3. Desired Target

Gameplay: In addition to the low target, the desired target includes implementation of multiplayer, improvements to the AI system, including more challenging behavior and playstyles and environments with pathing for the procedural generation. The crafting/looting systems should also be expanded upon to include more items/crafting recipes. It also should have a start screen and a better UI.

Art: None-abstract, but a painterly look and feel for characters and environment. Every scene in the game looks like a painting, all assets are visually cohesive and blend seamlessly together.

Audio: Include environment and atmosphere mix. Accentuate the gameplay with a fitting soundtrack.

1.4.1.4. High Target

Gameplay: In addition to the desired target, the high target includes the questing system as well as some smaller fun game additions such as pets/companions to help the player. Overall polish to the gameplay mechanics, AI/procedural generation systems and UI/visuals would also be included.

Art: Finding a new innovative art style that has never been used before.

Audio: Record, mix and post-process home made sound effects.

1.4.1.5. Extras

The extras for the game include even more additions to the crafting/looting systems with all sorts of items, as well as some fancy animations/visual mechanics, such as a dynamic multiplayer camera. More biomes and flashy extras like emotes and pets would also be an option.

Art: Inspire other game developers and players by setting trends with this new artstyle.

Audio: Enhance the underlying narrative with audio from a voice-acting session.

1.4.2. Task Lists

Comprehensive task lists for each of the four targets have been created. There are three main lists:

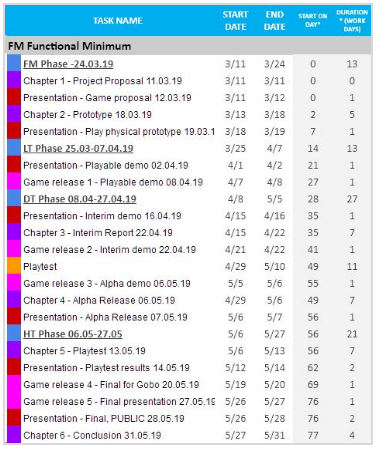
- Development Tasks
- Visual and Audio Tasks
- Miscellaneous Tasks

Throughout the project, the lists are subject to change. They are attached to the appendix as they were, at the beginning of the project.

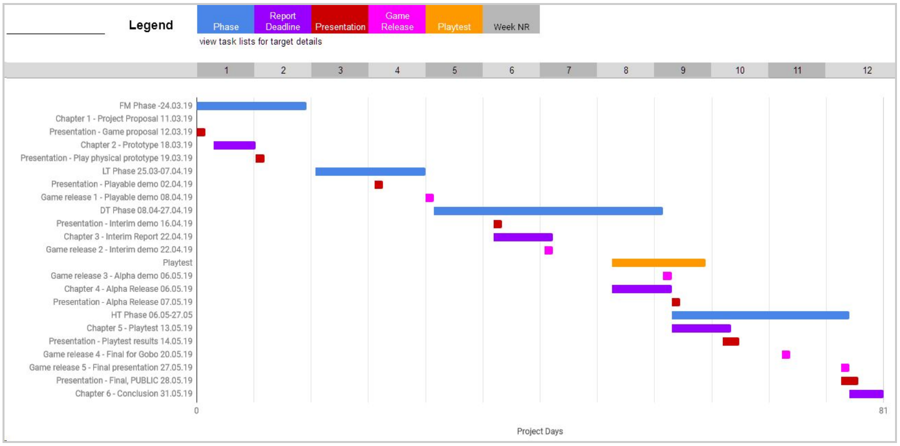
1.4.3. Timeline

A Gantt table was created to illustrate the global project planning. Details for the Targets are in the aforementioned task lists

Preview Project Overview



	TASK NAME	START DATE	END DATE	START ON DAY*	DURATION * (WORK DAYS)
	FM Functional Minimum				
🟦	**FM Phase -24.03.19**	3/11	3/24	0	13
🟪	Chapter 1 - Project Proposal 11.03.19	3/11	3/11	0	0
🟥	Presentation - Game proposal 12.03.19	3/11	3/12	0	1
🟪	Chapter 2 - Prototype 18.03.19	3/13	3/18	2	5
🟥	Presentation - Play physical prototype 19.03.1	3/18	3/19	7	1
🟦	**LT Phase 25.03-07.04.19**	3/25	4/7	14	13
🟥	Presentation - Playable demo 02.04.19	4/1	4/2	21	1
🟪	Game release 1 - Playable demo 08.04.19	4/7	4/8	27	1
🟦	**DT Phase 08.04-27.04.19**	4/8	5/5	28	27
🟥	Presentation - Interim demo 16.04.19	4/15	4/16	35	1
🟪	Chapter 3 - Interim Report 22.04.19	4/15	4/22	35	7
🟪	Game release 2 - Interim demo 22.04.19	4/21	4/22	41	1
🟧	Playtest	4/29	5/10	49	11
🟪	Game release 3 - Alpha demo 06.05.19	5/5	5/6	55	1
🟪	Chapter 4 - Alpha Release 06.05.19	4/29	5/6	49	7
🟥	Presentation - Alpha Release 07.05.19	5/6	5/7	56	1
🟦	**HT Phase 06.05-27.05**	5/6	5/27	56	21
🟪	Chapter 5 - Playtest 13.05.19	5/6	5/13	56	7
🟥	Presentation - Playtest results 14.05.19	5/12	5/14	62	2
🟪	Game release 4 - Final for Gobo 20.05.19	5/19	5/20	69	1
🟪	Game release 5 - Final presentation 27.05.1	5/26	5/27	76	1
🟥	Presentation - Final, PUBLIC 28.05.19	5/26	5/28	76	2
🟪	Chapter 6 - Conclusion 31.05.19	5/27	5/31	77	4



1.5. Assessment

At first glance the game seems to be a classic top-down shooter game. However, it is characterized by its fast paced combat and spiced with a tactical resource management. Since the player has to reach a destination where the chances of surviving become harder and harder, the game profits of a natural increase of the degree in difficulty. It's about finding the smartest way to maintain your life energy and at the same time secure your existence in the given environment.

The game loop is supposed to be very punishing by design which will lead to a short play-cycle. As the games environment is procedurally generated, the players experience is not exactly the same with each playthrough and they need to adapt fast to the currently presented world. Therefore, it is also not impossible that a player will be thrown directly into a tough situation right from the start, that may eliminate them very quickly. By restarting and basically retrying, the player is now again faced with a different surrounding.

The game is placed in the distant future (2099) on a depleted futuristic world that holds dangers around every corner. On that account, it is targeted at top-down shooter enthusiasts that like a pinch of tactics and exploration in their gameplay.

Chapter 2. Prototype

This chapter illustrates multiple prototypes which constitute the first stages within the Hazmat development. A main distinction can be made between our analogue paper prototype and two digital ones. Additionally, there will be an outline of design decisions made in the art department.

2.1. Prototype Setup

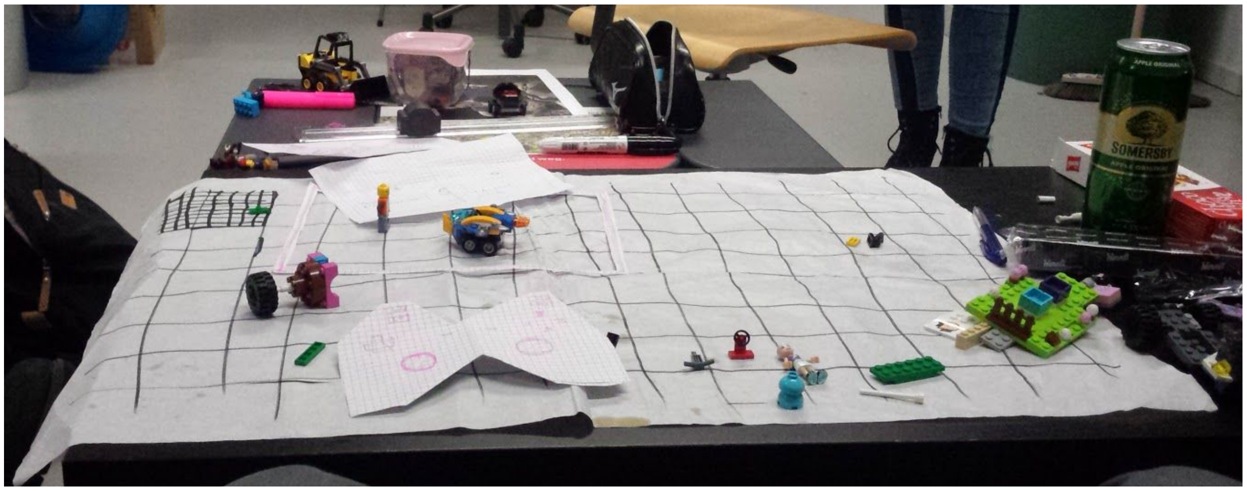
The paper prototype was used to experiment with core mechanics and to test which parts of our designs worked or did not work . After the paper prototyping, digital prototypes were created to get a better feel of previously chosen mechanics and (future) technical challenges. Design decisions made in both prototypes are completed by a short elaboration on current art guidelines.

2.1.1 Paper Prototype 1: Core mechanics & Playthrough

In order to test and further define the core mechanics and elements of our game, we needed to have a playable prototype as soon as possible. The created paper prototype let us play the game from start to end and helped us define all the elements for a minimum viable product.

Our game features a 2.5D isometric top-down view with a procedural world. This gamefield has been simulated using a paper with a drawn grid onto it. However: Although the world is based and generated in a grid-like view, dynamic objects, including the player, are not strictly bound to this system. This was one of our earliest decisions when playing the prototype, giving more freedom to the player and making it easier for the overall development. Gameplay elements like characters, entities, gameobjects and pick-ups have been recreated with Lego pieces. Important resources like the player's energy level were either drawn onto paper or displayed with an indicator on a ruler (simulating a bar).

Fig.1) Paper Prototype 1

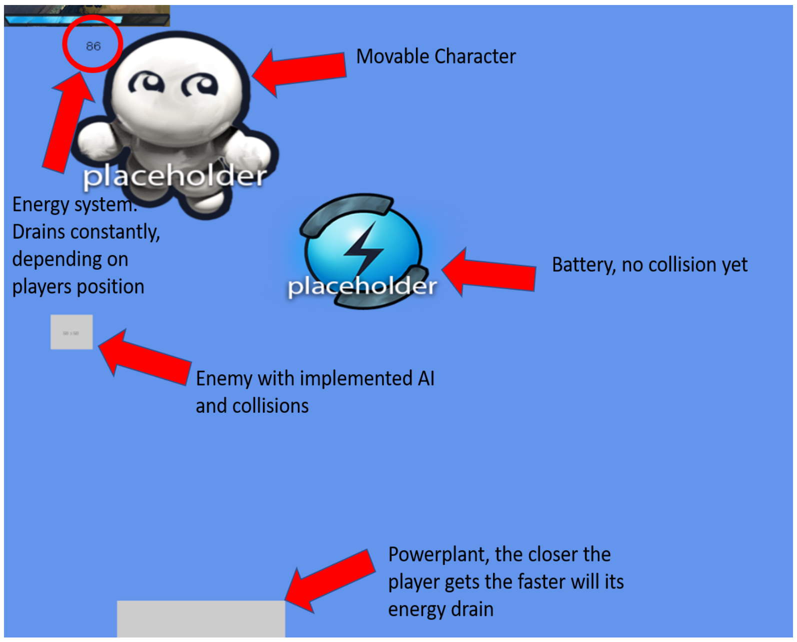


During the testing, one person was playing the "player", another acted as the game engine. The player was able to move around, shoot onto, interact or pick-up stuff that lies on the game board. The game engine on the other hand "spawned" gameobjects and actively manipulated the player's energy level. This play session made it possible to discuss the core of our game.

2.1.2 Digital Prototype 1: Movement, Energy, Spawning, Collisions, AI

This prototype shows player movement in the environment. It implements basic AI for enemies, energy system, and the power plant. The assets used are a combination of pieces from concept art as well as placeholder images.

Fig.2) Digital Prototype 2


86
Movable Character
placeholder
Energy system: Drains constantly, depending on players position
placeholder
Battery, no collision yet
Enemy with implemented AI and collisions
Powerplant, the closer the player gets the faster will its energy drain

Spawning

The first step is to spawn the player and the power plant on the screen. The player is spawned at the origin and the powerplant is spawned somewhere randomly on an arc at a given distance from the player. These positions are currently implemented with screen coordinates but will be mapped to world coordinates later.

Energy

The player spawns with full energy. This energy will keep decreasing as time goes on. The rate at which the player's energy decreases, scales with the distance from the spawned power plant.

Movement

The player can move around the screen using a WASD control scheme.

Collisions

When the player tries to walk into the power plant, the collision system prevents the two entities from overlapping.

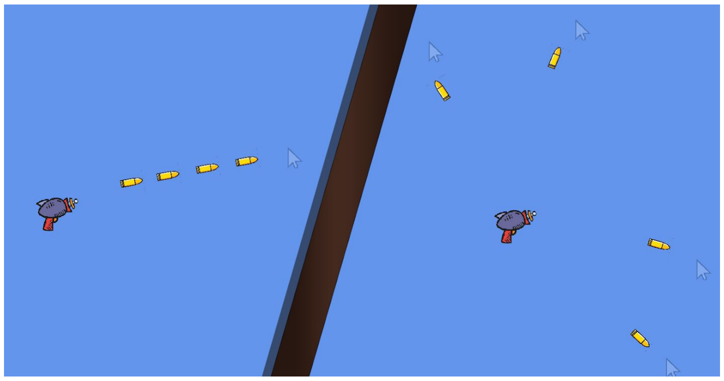
AI

Basic AI is spawned on the map and will start to follow the player as soon as they get into a certain range. In the future, it will stop as soon as it is within attack range of the player and will attack once implemented. If the player moves too far away from the AI, it will transition back to standby mode, waiting for the player to get close.

2.1.3 Digital Prototype 2: Input Handling and Shooting Mechanics

In this prototype an input handler and shooting mechanic have been implemented.

Fig.3) Digital Prototype 2



Input Handler

A class that is used to handle input based on commands. Different types of commands can be created and applied for each different button press. One of the first things that needs to be done during the game loop is to read the input. After that the execution of each input can be called whenever necessary (e.g. when updating the players state we want to execute the movement and shooting commands).

Shooting Mechanics

Texture

There is the possibility to easily change the texture of the gun and the bullets, by changing the parameters during the creation of the gun.

Shooting Direction

The bullets follow the position of the mouse (this needs to be changed to work with the XBOX controller).
The bullet asset gets rotated according to the direction it gets shot.

Reload Time

With this implementation it is very easy to setup the reload time by just changing one parameter. This parameter stands for the interval of time that needs to pass between two bullets.

Speed and Radius

The speed of the bullets can be determined and also the travel time (easy to set up long vs short range guns).

2.1.4 Art Design Decisions

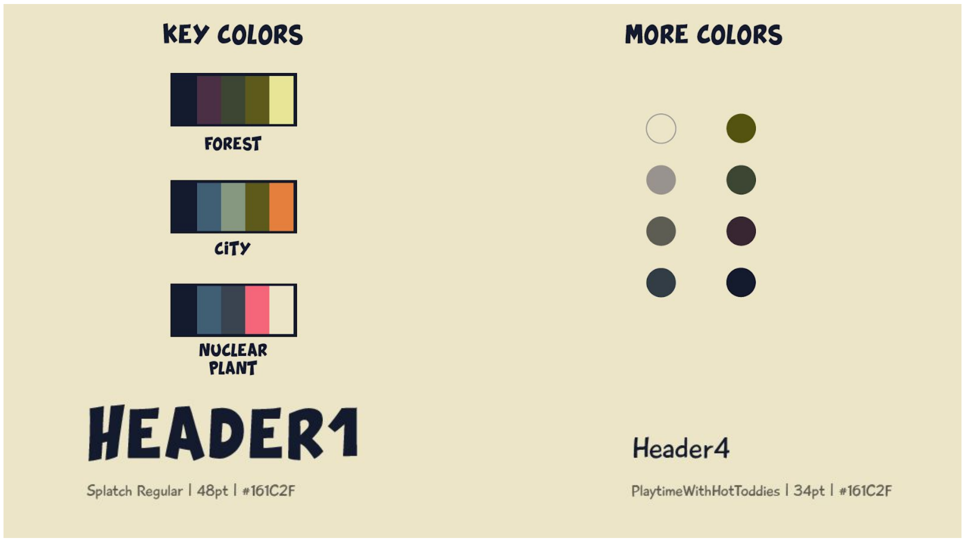
For the overall visuals and game aesthetics we chose a painterly cartoon style that is fun, readable, and realistic within in the timeframe of the game development and also compatible with the procedural generation of the maps. Below a short overview on visual guidelines:

- General artstyle & Visual Design Guidelines
    - **3D isometric camera view**
    - **Shape Language: Round vs Edged**
        - **Round:** friendly, safety, collectables, character (Mat)
        - **Edged:** hostile, dangerous, enemies. Explosives
        - **Round/Edged merged UI Elements:**
          Focus, Crosshair, Interaction, Objectives
- Color
    - Follow provided swatch set (see below)
- Typography

The Font Families *Splatch Regular* & *PlaytimeWithHotToddies* are used to support and complement the cartoon art style. *Splatch Regular* is used in all big head titles while *PlaytimeWithHotToddies* is responsible for all running text labels and small head titles.

- Environment
    - **Decorative elements**
        - General environment
        - Darker, lower contrast
    - **Gameplay relevant elements**
        - character, enemies, interactive elements and projectiles.
        - Highlighted to provide a contrast to the general environment for easy focus on important elements.
- Character
    - Round shapes
    - Bright colours
    - Needs to be the most adorable element in the environment.
- UI
    - Minimal
    - Only functional elements
        - End boss life stats
        - No lifestats for trash mobs
        - Direction to power plant

Fig. 3) Illustrations of visual guidelines


KEY COLORS
FOREST
CITY
NUCLEAR PLANT
HEADER1
Splatch Regular | 48pt | #161C2F
MORE COLORS
Header4
PlaytimeWithHotToddies | 34pt | #161C2F

2.2. Playing Experience

As development is in the early stages, the extent to which current prototype player experiences reflect the final gameplay are limited. At this point there have been multiple playthroughs on the paper prototype, whereas the digital prototypes (playable on PC) gave more insight into player experiences accompanying selected game mechanics such as the interaction with enemy AI and the shooting ability.

2.2.1 Paper Prototype 1: Core mechanics & Playthrough

The game was played by distributing rolls simulating the final product. There was one "player", while other team members took the role of the game engine or were analysing the gameplay. We had multiple playthroughs using our paper prototype, altering different gameplay aspects on the fly. In the beginning, playing as "the player" felt strange, as the gameplay (What do I have to do? How do I interact with the environment?) had not yet been fully defined. During the session, upcoming questions were discussed and directly tested. After several playthroughs we managed to induce the sense of game-feel we would like to convey.

2.2.2 Digital Prototype 1: Movement, Spawning, Collisions, AI

The digital prototype as it stands right now doesn’t include the core game mechanic, which is the combat. It consists only of walking around and colliding with objects, which is satisfying from a development perspective, but not so much as a player.

2.2.3 Digital Prototype 2: Input Handling and Shooting Mechanics

In the first prototype, the position of the mouse cursor defines the direction of the bullets. One the direction is chosen a bullet is fired by pressing F. Since the final version will be played using the XBOX controller, we will need to test this again, but the idea would be to use the joystick in order to choose the orientation of the player and shoot using R2.

The feeling of shooting top down using the mouse feels pretty comfortable and intuitive.

It is easy to get used to since by moving the mouse and pressing F the visual effect shows how the two processes are connected.

2.3. Findings and Conclusion

2.3.1 Paper Prototype 1: Core mechanics & Playthrough

Playing our paper prototype helped us realizing what the core mechanic of our game should be. Until the paper prototyping, we had the energy system and making tradeoffs with this energy as our core mechanics. This mechanic alone cannot sustain an entertaining game. It is an important supporting mechanic, that can spice up the gameplay. We realized that the fast-paced combat (shooter) is the actual core of our game. The game can work with this mechanic alone and gets really interesting when combined with the energy tradeoff mechanic.

Furthermore, we were able to pin down which gameplay elements are needed and which are not, how the games structure is laid out and also created a basic narrative thread. Once we started playing the game prototype, a lot of unanswered questions appeared as “players” progressed. We had a lot of healthy discussions about our project, that really brought us forward.

One thing that didn’t quite work was to reason about the feel of the core mechanic. As we could not a fast paced shooter on paper, things that we think are fun might not work. That is why we settled with the plan on creating a digital prototype of this very core of our game as soon as possible.

2.3.2 Digital Prototype 1: Movement, Spawning, Collisions, AI

The workflow with the ECS has been excellent. Once you get to thinking in terms of components and systems, adding new functionality to the game becomes extremely simple. There were some minor challenges we ran into with the library we are using, like UI elements and order of the gameplay systems, which may lead us to re-evaluate using this specific library. However, in terms of gameplay elements, the digital prototype is not established enough to draw any conclusions about the mechanics of the gameplay yet.

2.3.3 Digital Prototype 2: Input Handling and Shooting Mechanics

Handling the input in a way that makes the code less messy was the first challenge. Instead of having a bunch of code in different classes for different uses cases where input is involved, a specific class has been set up in order to take care of the input.

Creating a simple shooting mechanic is easy if the only interest is moving some object on the screen. In this case a more generic class has been created in order to take into consideration different possibilities of guns the player could use. It is easy to change different parameters to simulate a type of gun, for example changing the reload time or the speed of the bullets. Another thing to take into account was to make the sprite rotate in the right direction based on the orientation of the player. After different attempts to make this work it finally worked out. However we will need to make the interface of the position and orientation of the player more robust in order to facilitate this kind of computation (e.g. allowing some objects to be child objects of the player).

Chapter 3. Interim Report

3.1. Progress

3.1.1 Completed Layers

After four weeks of development, we have completed our Functional Minimum (FM) and Low Target (LT). The player can walk around in a procedurally generated world, interact (shooting) with AI enemies. The summary of our FM and LT targets offers a first overview of currently implemented functionalities. This list is complemented by following screenshots and further elaborations.

Functional Minimum	***Low Target***
Player Spawn	Basic Procedural Generation (Power Plant w/Pathing)
Input Handler	AI (go closer to aggravate, spawn in camps)
Player Movement	Basic HUD/UI (stability bar: animated)
Shooting Mechanic	Test 2D animations
Enemy Spawning	Loot boxes
Basic Enemy AI (spawning, move towards player)	Shaders (2D glow, toon, texture + diffuse shader)
Energy System	Interaction with objects (loot, pick up gun)
Story screen	
Implement placeholder assets	

Overview Functional Minimum and Low Target

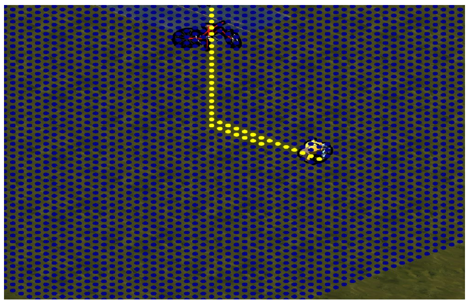
3.1.2 Development

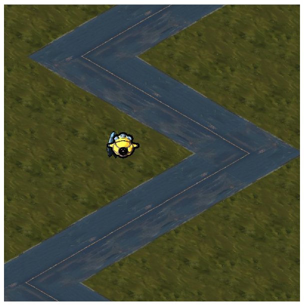
Engine

To help facilitate some of the other systems, some work went into adding additional engine features. The first addition was the collision handling. We implemented an abstraction that allows the collision handlers to have very fine-grained control over which entities they wish to handle, making it very simple to add new collision logic. We also have a similar system for interactable objects like the loot boxes that can change the behavior of items that the player interacts with. An event system was also created. This helps with the logic for one time events that are triggered by performing certain actions in the world. Finally, a custom input handler was also added. It watches the input state and outputs events based on the changes. For basic button inputs it can output press, hold and release events, and for the triggers and thumbsticks it notifies changes in input values.

Graphics/Assets

The graphics from the artists have been added to the game. It involved some testing to find the best way to load and reference the assets, especially with the static sprites all being in one sprite sheet. Animations are handled using the Spine runtime. The 3D models have also been imported and we are currently still working on the animations for those. A custom camera was implemented to handle the logic for the perspective and to allow for fine-tune control of the rendering.


AI Pathfinding


Procedural Environment

Enemies & AI

We currently have two different enemy types: a simple drone that will follow the player around trying to make contact and deal damage, and a basic shooter, that when the player is to close and in his line of sight attacks and follows the player. Both enemies have similar but different FSM that govern their behavior. To find and follow the player we implemented a path finding algorithm, that uses a grid (see above) to decide if a region is walkable or not. On that grid we run the shortest path algorithm A*.

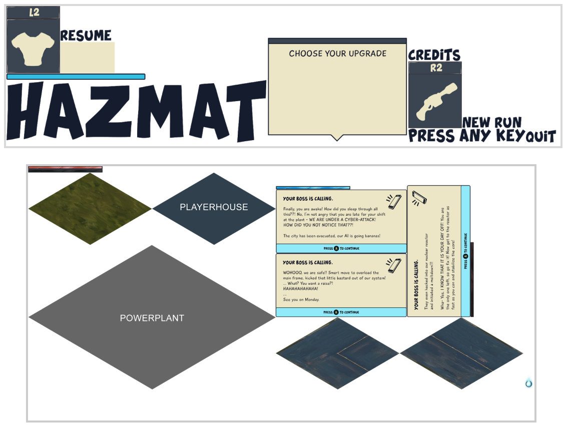
Procedural Generation

The terrain is randomly generated at the beginning of each game. The power plant (our objective) is spawned at a random angle from the origin (with a fixed distance). Afterwards a street connecting the origin and the power plant is generated with random changes of direction. Finally the enemy camps are put on the map they are currently also being generated at the beginning of a new game, later we will do it procedurally while the game is being played, to reduce the total amount of active entities during session.

3.1.3. Art

3.1.3. 2D Art

MAT 72 %
CORE 36 %
City Environment Concept



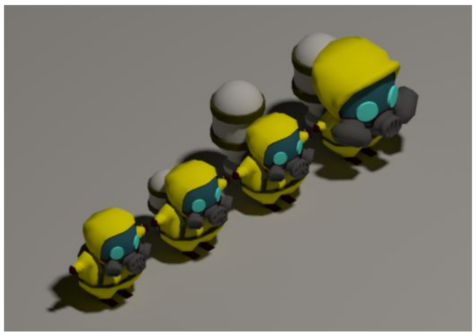
Spritesheets - Static & Animated Sprites

The environment concept has been further specified. Focus lies on a city environment.

Sprite sheets contain basic sprites to test out the pipeline for implementing the textures.

3.1.3. 3D Art

Player Character



Character with gear upgrades

At this stage there is a basic 3D character whose gear has attachable upgrade items.

Weapons

There are two types of guns which the player can choose at the beginning of the games. Each gun has its own attachable upgrade elements.

3.2. Challenges

3.2.1 Development

Libraries

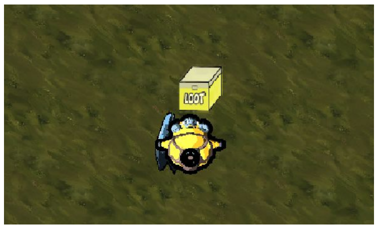
We had some issues with the library that we originally chose to use. We made the switch to DefaultECS to handle the ECS and are using Aether2D for the physics. These have worked relatively well so far.

Perspective View

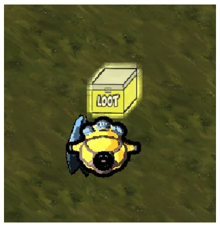
The perspective view led to some small challenges with mapping and such. Since the assets are drawn in perspective, the camera has to place them in the right position without actually rotating/scaling the asset to the proper proportions. The 3D assets in a 2D world are still not 100% correct and are being revised.

Shaders

Since some assets in our game will be in 3D we spent a lot of time trying to figure out how to properly draw our models in a fashion that suits also the 2D assets. We wanted to create a glowing effect on 2D sprites and in order to achieve that we programmed a pixel shader. This shader was underestimated and is still under development. The biggest issue was that almost no one in our team had a good background in shader programming and this resulted in some troubles.

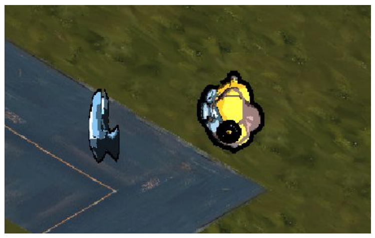
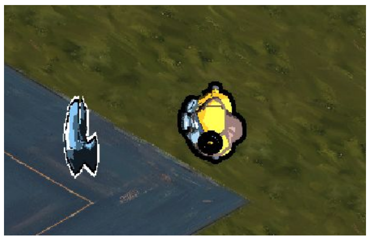


Cel Shader



Glowing Shader

As we see the character has an outline and a cartoonish look. This was achieved using the cel shader which takes the intensity on a defined pixel and based on that it clamps the color value to a given intensity. We can also see a glowing effect on the edges of the lootbox, that is the glowing shader which is animated over time on the alpha channel.

Outline Shader 3D

The glowing shader was only based on pixel level, but since some of our assets are in 3D we realised that we needed an equivalent effect also for these ones.
This is a similar shader, which uses the cel shader(next) but also animates an outline.

Overall the hard parts are to manage to program a working shader and set up a good pipeline in order to apply them.

Interaction

During the game the character has to interact with loot boxes, guns, and other objects so we set up an interaction system that makes handling of different interaction cases really simple. In this part also the shaders come into play in order to create a graphical hint for a possible action the player might be able to take.

3.2.2 Art

2D Art

Harder than expected: Finding the right angle for the isometric textures while staying flexible with an orthogonal top down view on the texture in Photoshop, the overall management of isometric textures.

Easier than expected: Preparing the export of assets after doing the concept art for the city. Most of the elements in the concept piece were export ready as soon as they were done.

3D Art

We were not sure how the light would function and how this affects the display of our materials for the 3D models. As we found out, transparency can be displayed and there is a global lighting system. Since the development team is currently also working on shaders, the final level of detail on 3D textures still has to be determined. Only using standard materials while not having an unwrapped 3D model caused issues when displayed in the engine. 3D objects need to be UV unwrapped and textures applied per individual object.

Creating the character model took longer than anticipated, the modular upgrades took less time. At the moment there are two weapons, the original idea was that there would be at least 3 types. With all weapons having individual updates, we might lower the amount of different weapons.

3.3. Future Work

In the next three weeks, the team is working on the Desired Target (DT). This sprint ends with the Alpha release on May 6, when the game will be evaluated by teachers and external experts. Our goal is to complete a self explanatory prototype which includes all our core mechanics, story elements and visuals as well as basic sound. Due to unforeseen technical challenges, some LT development goals have not been completed. As a result, we have reduced the complexity of the end game and defined the completion of said elements as priority.

Priority tasks:

- Basic Combination Mechanic (chip + armor or weapon = upgrade)
- Player Pick Up Window (in game space, timer, follows player)
- Different Collectables (add chip collectable)
- Basic HUD/UI (map)
- Test 3D animations

Individual contributions for the DT sprint are listed and elaborated on below. Sound design will be outsourced.

3.3.1. Development

Task	Explanation
Basic Combination Mechanic	Implement the upgrading system
Player Pick Up	Add the UI for the upgrading system in the world
Different Collectables	Add the upgrade item to the loot tables
Test 3D animations	Implement the 3D animations for walking, shooting, etc.
Sound effects	Add sound effects to the game
Co-op	Add support for multiple players.
Good AI	Further improve the AI, and more types of AI
Procedural Map Generation - Medium (Path, Basic Environments)	Continue improving the procedural generation and include the town/power plant environment
Comprehensive UI (Start Screen, HUD, Waypoints)	Improve the current UI state and add the new menus and overlays. Also work on adding a map for the player.
Event System (Story events)	Finalize the story events by adding the conclusion and game end.
enemies drop items or energy	Set loot chances on enemy kills
City generation	Add procedural generation of the town
Loot containers scattered on map	Add lootable containers to the procedural generation
Check points	Add checkpoints to the map where the player can restore their energy
Victory screen/Player stats	Implement win/loss screen and game results.
Visual representation of 3D upgrades	Add model swapping for upgrades
Prepare playtest data collection	Add some logging for analysis of gameplay tests

3.3.2. Art

Visual assets created for the FM and LT are polished and enhanced where needed.

3.3.2.1. 2D Art

Task		Time	Elaboration
Chip collectable	JB	2	Creating a collectable chip to upgrade the character's armor or weapon.
Complete town tiles	JB	8	Export of the city textures from the city concept art.
Polished game & story UI	JB	10	Bringing the UI Elements into a final state.

3.3.2.1. 3D Art

Task		Time	Elaboration
Character rig	SW	8	Creating bones and skin 3D model for animations and definition of attachment nodes for attachable gear
Char walking	SW	5	Animated walk cycle
Char shooting	SW	2	Animated shooting action
Gun 1 upgrades	SW	1	New mesh + create gun bone and skin modular elements to it
Gun 1 UV + textures	SW	3	Unwrap mesh and create textures
Gun 2 upgrades	JS	1	New mesh + create gun bone and skin modular elements to it
Gun 2 UV + textures	JS	2	Unwrap mesh and create textures
Char UV + textures	SW	5	Unwrap mesh and create textures
3D enemy static	JS	5	New mesh
3D enemy rogue	JS	5	New mesh

3.3.2. Playtest

The playtest of the DT Alpha release will be organised in the end of April and carried out between May 7 and 10. The report is completed for hand-in by May 13.

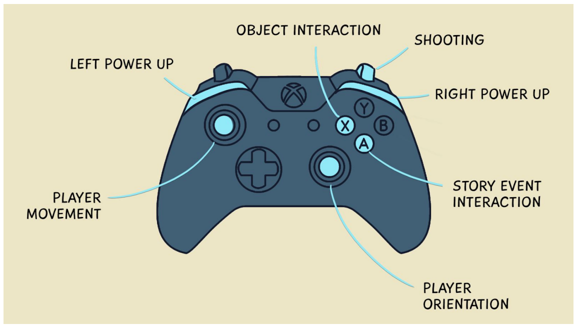
Task		Time	Elaboration
Organisation	Team	2	Finding playtesters, date and location
Defining playtest protocol	Team	1	Defining playtest protocol
Questionnaire Design	Team	4	What do we want to know? What do we ask players?
Conducting Playtest	Team	4	Collecting data while people play
Evaluation of results	Team	8	Analysing collected data

Chapter 4. Alpha Release

4.1. Progress

While not all goals of our Desired Target (DT) have been met, there has been substantial progress. Prototype assets have been polished and implemented. On the technical end, previous issues were resolved. We implemented two character pick-ups and character upgrade possibilities, 3D animations, basic city generation and began to balance gameplay.

Finally, a tutorial has been added in text and visuals.



Desired Target: New Implementations:

Development	**Art**
Basic Combination Mechanic	Chip collectable
Player Pick Up (window in game space, follows player)	Victory screen/ player stats
Different Collectables (battery, chip)	Character rig
Basic HUD/UI (map)	Char walking
Good AI	Gun 1 UV & textures
Procedural Map Generation - Medium (Path, Basic Environments)	Gun 2 upgrades
Event System (Story events)	Gun 2 UV & textures
enemies drop EITHER item OR energy	Char UV & textures
City generation	3D enemy static
Victory screen/Player stats	3D enemy rogue

4.1.1 Development

3D Graphics

At the last release, the game had a strange mixture of 2D and 3D assets as well as varying perspectives. This led to some hacky implementations to make things look correct and still left some open issues with relative positioning and such. One of the main points of feedback from the last demonstration was to try and move to full 3D to alleviate these issues and simplify the logic overall. Since then, the assets have all been moved to 3D space, whether the textures are rendered onto a plane or are actual 3D models. Additionally a new camera system was implemented to provide a correct perspective when rendering the 3D models.

Animations

The animations for the player and enemies have been partially added. Currently we are using the Aether library but there are some issues with some of the assets we have. The player animations seem to work for the most part but enemy animations and the gun don't.

Shooting

With all the new changes with our graphics framework we had to work on the shooting again to make it work as expected.

Enemy Health Bar

We also implemented a Health Bar for enemies.

Post Processing Effects

Being able to change the colors settings of the game, such as hue, contrast, saturation, brightness is really useful to set the right mood of our game.
A vignette was also added to give a more cinematic look.

Music and Sound Effects

A general class for managing sound effects and music has been set up. For now our game has shooting sound effects and a background music during gameplay.

Tutorial

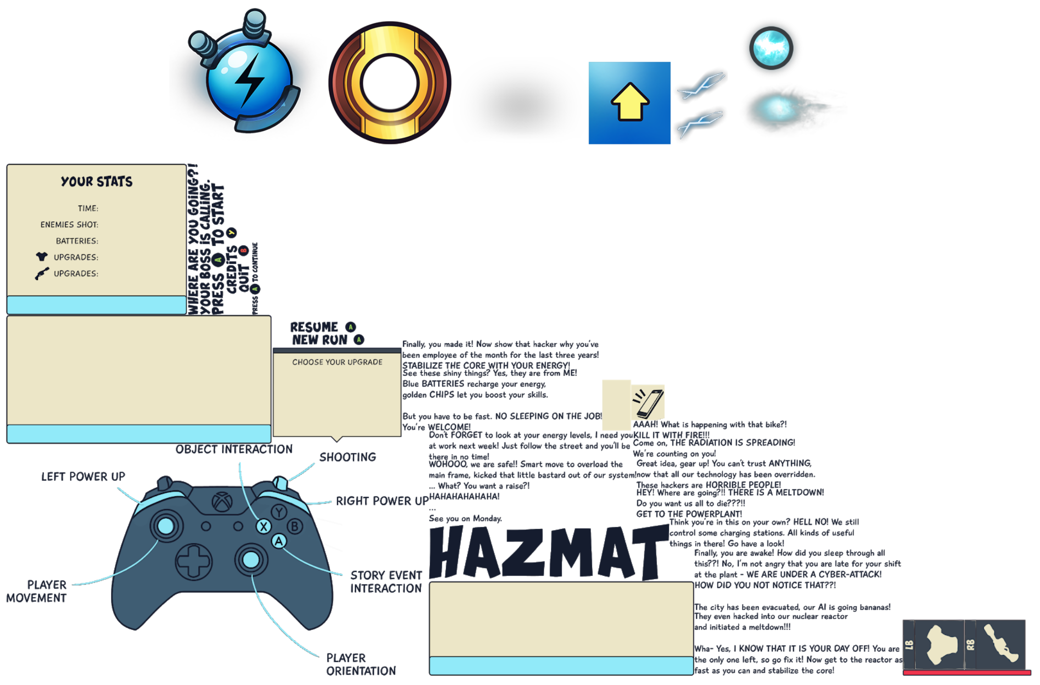
To help the player understand how to play the game and interact with the various systems, we implemented a basic tutorial system that gives tip popups. These popups occur whenever the player encounters something new or to give them some indication of where to go and what to do, like at the beginning to notify them that they should pick up a weapon or what the battery drops do, etc.

4.1.2 Art

With the assets now being in 3D, there has been a change in the art pipeline, where most environmental objects are now created in 3D.

2D Art

UI popups, projectile and collectables have been reworked and animated in spine.



Finally, you made it! Now show that hacker why you've been employee of the month for the last three years!
STABILIZE THE CORE WITH YOUR ENERGY!
See these shiny things? Yes, they are from ME!
Blue BATTERIES recharge your energy,
golden CHIPS let you boost your skills.

But you have to be fast. NO SLEEPING ON THE JOB!
You're WELCOME!

Don't FORGET to look at your energy levels, I need you at work next week! Just follow the street and you'll be there in no time!

WOHOOO, we are safe!! Smart move to overload the main frame, kicked that little bastard out of our system!
... What? You want a raise?!
HAHAHAHAHAHA!
...
See you on Monday.

AAAH! What is happening with that bike?!
KILL IT WITH FIRE!!!

Come on, THE RADIATION IS SPREADING!
We're counting on you!

Great idea, gear up! You can't trust ANYTHING, now that all our technology has been overridden.
These hackers are HORRIBLE PEOPLE!

HEY! Where are going?!! THERE IS A MELTDOWN!
Do you want us all to die???!!
GET TO THE POWERPLANT!

Think you're in this on your own? HELL NO! We still control some charging stations. All kinds of useful things in there! Go have a look!

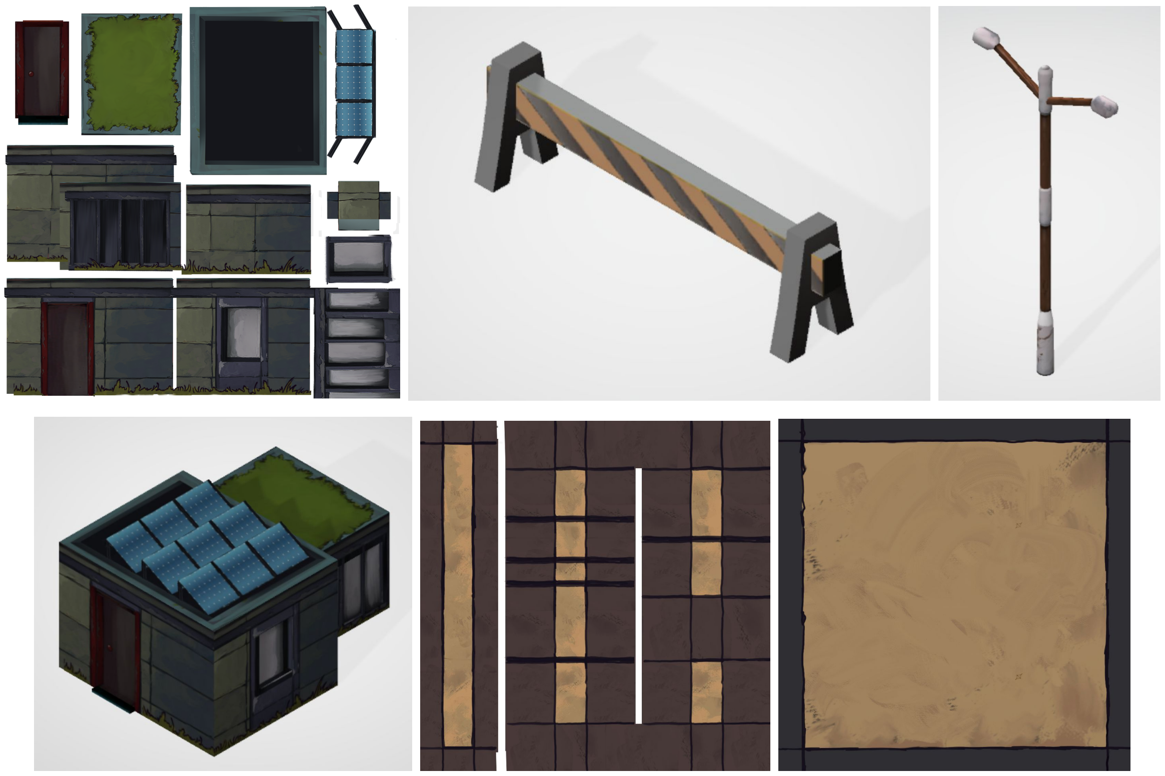
Finally, you are awake! How did you sleep through all this??! No, I'm not angry that you are late for your shift at the plant - WE ARE UNDER A CYBER-ATTACK!
HOW DID YOU NOT NOTICE THAT??!

The city has been evacuated, our AI is going bananas!
They even hacked into our nuclear reactor and initiated a meltdown!!!

Wha- Yes, I KNOW THAT IT IS YOUR DAY OFF! You are the only one left, so go fix it! Now get to the reactor as fast as you can and stabilize the core!

UI updates

The UV maps for the basic environment objects have been created and repainted.

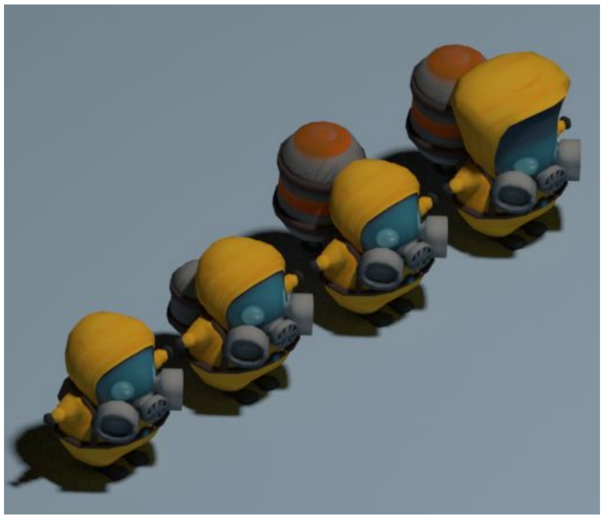


Overview: new 3D textures

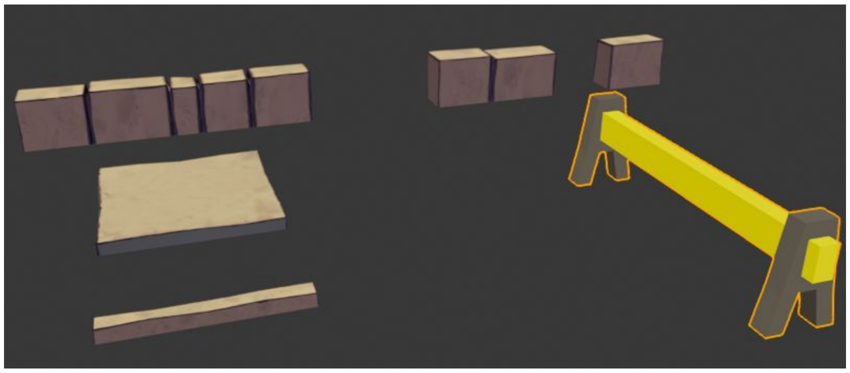
3D Art

The player character has been adjusted, is skinned to an animation rig and has a walking animation. Attachable character upgrades are skinned to the animation model.

Multiple environment objects have been created.


Character upgrades

3D house


Environment props

4.2. Challenges

4.2.1 Development

3D Graphics

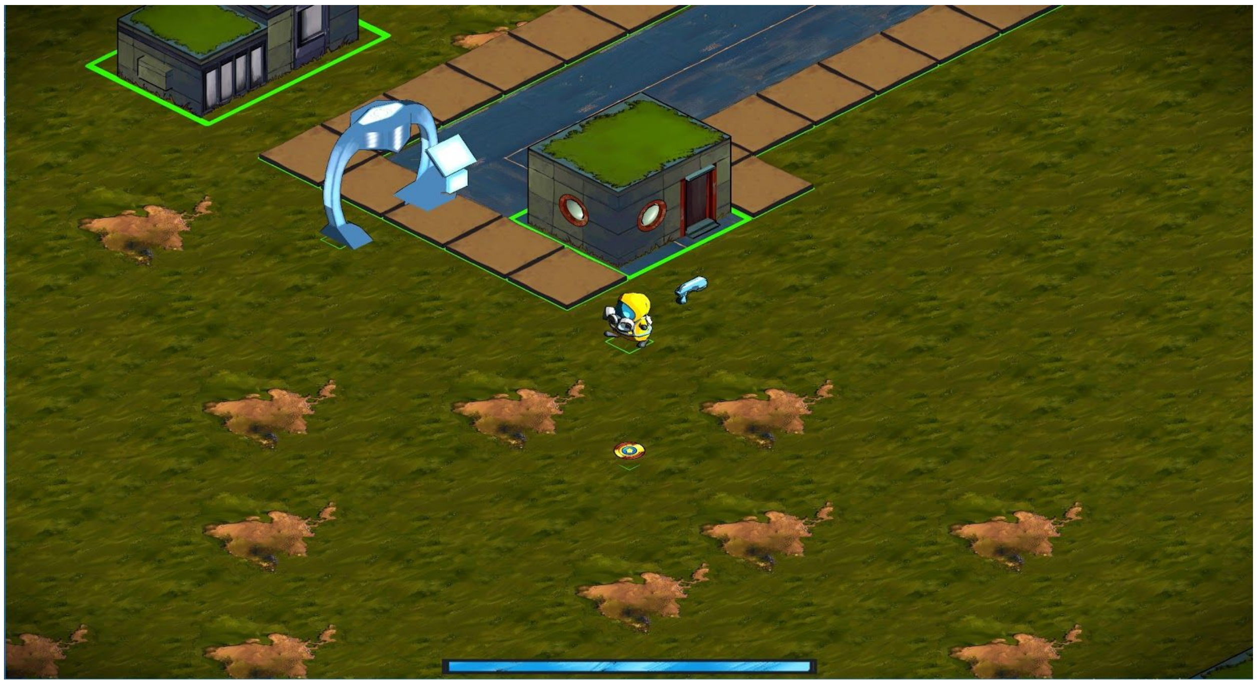
The integration of 3D rendering with the 2D animation library that is being used was a bit hacky. Setting the view and projection matrices let the animations be played in 3D space, but they maintained the x/y axes and therefore looked to be flat on the ground. This was fixable by adding a custom billboarded model matrix to have the rendering face the camera but the batching of the renderer moved all of the animations to the position of the last rendered object. It now uses one batch operation per animation, which is not optimal in terms of performance but seems to work well enough.

Animations

As stated before, the animations are leading to some issues. We haven't completely figured out why but are working on it and exploring other libraries to see if we can figure out why. On top of animations, the bone attachments are problematic as well. Since the animations aren't running they way they should, it is not possible to properly attach objects to specific parts of other models because the positions are incorrect.

Post Processing

Setting up the post processor class wasn't the easiest thing. But overall the hardest part was writing the shaders to manage all the colors in the game and also the shaders to show the vignette.



4.2.2 Art

2D Art

Due to the pipeline shift, assets previously drawn in perspective had to be remade.

3D Art

There have been multiple issues with importing animations and textures that are stored in FBX files into Monogame. Since some assets had to be exported multiple times to identify the issue, there was time lost that could have been used otherwise.

4.3. Future Work

For our High Target, we will prioritize the following points:

- User friendly tutorial
- Balancing
- Implementation of visual assets
- Sound & Music
- Multiplayer

Concerning usability and balancing, we will conduct a playtest and use the results to improve the player experience. Our next steps are listed below.

Hight Target Task Lists:

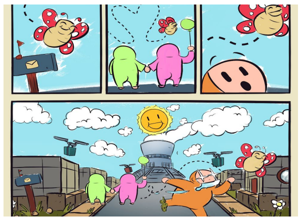
Development	**Art**	**Misc**
User friendly tutorial	Environment UV maps	Tutorial & Story text
Test 3D animations	Polished game & story UI	Music
Sound effects	Char shooting	Sound
Co-op	Gun 1 upgrades	
Comprehensive UI (Start Screen, HUD, Waypoints)	Detailed story screens	
Loot containers scattered on map	Polished 2D assets	
Check points	Polished 3D assets	
Visual representation of 3D upgrades	Final Collectables/ upgrades	
Visual feedback armor/weapon upgrades		
Procedural Map Generation - Advanced (Path, 3 Environments, Connection to Power Plant)		
Music		
Tweaking Stats		
Advanced AI		
Final collectables		

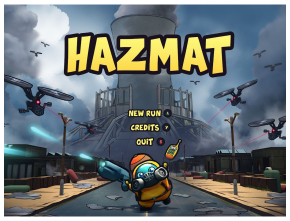
Chapter 6. Conclusion

6.1. Final Results

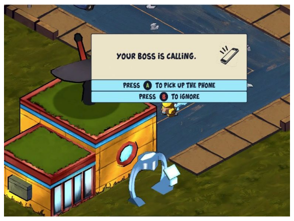
Player experience

Right at the start of the game, players are introduced to the story. After a short video, they can start their first run. Here A tutorial leads them through the experience. Time pressure is created to a large amount of enemies, the draining health and disappearing upgrade options. Since individual runs last between 1-4 minutes, players are motivated to try to reach their goal over and over again. Gameplay and visual feedback is complemented by sound effects, music and voice overs.

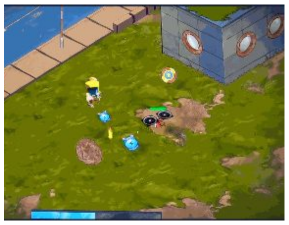

Story



Start screen



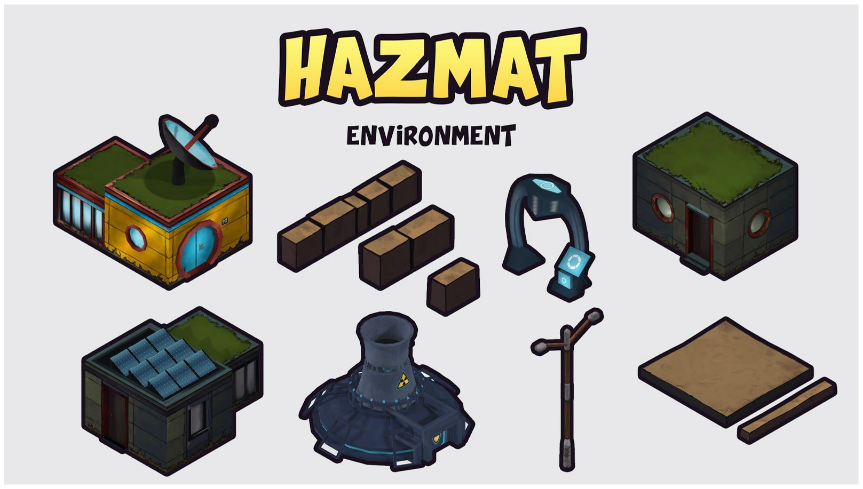
Tutorial

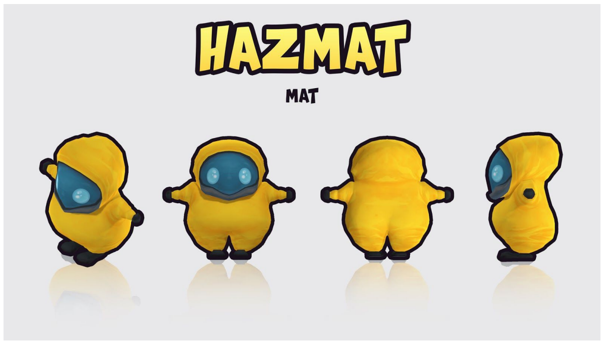

Gameplay

6.1.1 Development

Not too many changes from the development side since the alpha. Some refactoring and optimizations for performance as well as adding the new assets from the artists, but nothing major in terms of functionality.

6.1.2 Art



	HAZMAT UPGRADES

6.2. Experience: Development Reflection

General Dev process

In the first months of development, the team worked at a relatively steady pace. Workload and pressure increased towards the final weeks. Communication between the different members functioned well. After we had decided on a concept during the Functional Minimum phase, most challenges came up while working on the Low Target. Many components were not completed and were transferred during the Desired Target phase. We never fully made up for this delay, so we scratched a couple of features from our Desired- and High Targets (eg. maps, end boss, limited amount of upgrade possibilities). While in the beginning Trello was used as a means to establish process overview, it got updated less towards the end of the project. Similarly, Slack was used less, while on the other side our use of Whatsapp increased. The last few weeks of development were more intense, since not everything had been completed within previously defined sprints or internal deadlines. Every member did their best to complete our vision.

6.2.1 Development

The initial development goals for our team were fairly ambitious. We decided on using an Entity Component System to run the core logic of our game and set out to develop various technical achievements, including artificial intelligence, procedural generation, multiplayer and more. In the first few weeks of development there was a lot of back and forth as to who was changing what. This is because everyone wants to work on their parts, but without the foundation and core engine systems, it is hard to have everyone work on their individual tasks. After setting up the basics, the workflow became much more fluid. Each developer had their section to work on without having to worry about what the others were doing. Each week we made improvements to the subsystems in the game, as a result of player feedback or other goals in our to-do list. We did end up falling a bit behind the projected schedule due to bug fixing and having to refactor large parts of the system, like the 2D/3D rendering sections. This led to us not being able to implement certain features like multiplayer from our High Target, but we still managed to hit the Desired Target. Throughout prototyping and playtesting, we received a lot of external feedback to help improve the game. These suggestions were usually small quality of life, or visual changes which meant we didn’t have to overhaul too much gameplay or core functionality. This let us take more time to work on improving what we thought needed to be done and avoid fixing/changing too much of the existing code.

6.3 Personal Impressions

6.3.1 Sophie Walker- Production, 3D Artist

This course was a great experience with regards to teamwork and interdisciplinary communication. We did not achieve everything that was planned, but we managed to implement a lot. I appreciate having had the chance to work with such a skilled team and thank you for setting up this collaborative program.

It is evident that you are dedicated to create something valuable for your students, but there is a focus on ETH. As a ZHdK student, I did not benefit as much from the lectures, because we learned these things in our bachelor education. It might be an idea not to have them mandatory for us, since we spend the required amount of time on asset creation anyway. Which brings me to my next point.

The problem of the games industry and this course
The amount of credits awarded to ZHdK students is not fair.
2ECs is nothing for a cooperation-based project like this.

Competitive artists will always go the extra mile to create work that they are proud of. That is okay, and standard in many industries. The problem arises when even if the official requirement is to spend 5h/week - as soon as there is one artist who produces assets worth 20h/week, this level of quality is used as measurement for all other artists. Overtime is a choice, not a requirement.

This distribution of credits also reflects a common theme in the game's industry: Game developers are paid better than artists, while artists are expected to work over-hours to create something that looks fantastic ("fantastic" = something that measures up to the 20h/week person in our example, and is impossible to achieve during official working hours). And just like in the industry, we did not get "paid".

As you say, the final course results are improving every year. This means that in the future, there is more and more pressure on the artists as well. Finally, you are urging us to publish our results (which is great!), so what we contribute is public. Artists want this project to reflect their best work, too. And that is impossible with 2ECs.

It is clear that this was not the intention of the course creators, but it is not okay to already have this imbalanced, overtime-centered industry mentality introduced in an education. Please, change this. Either by rewarding an equal amounts of credits or by (re-)setting realistic expectations on visual results.

6.3.2 Sven Knobloch - Lead Programmer

Overall, the course was fun and challenging. I got a taste of what it's like to work in an interdisciplinary team and gain some insight into the capabilities and expectations of others in the game development world. As lead developer, I had to manage the development and act as the go-to representative when communicating with the producer/artists, which was a nice way to work on my management and communication skills. Even though it's not perfect, I think we can all be very proud of our game. The idea evolved a lot over the short time we spent working on it and I wasn't quite sure what to expect at the end, but the game came out well and is surprisingly fun!

The largest technical difficulty for me was getting all the different software we used to play nice with one another. Our artists worked in various programs for design and animations, some of which worked well with MonoGame and others which were a catastrophe, as seen in some of our animations. Some things were relatively easy to fix by adding some functionality to bridge the incompatibilities but others would have required writing and entirely new system which is a huge task and something we definitely didn't have time for. This was also one of the downsides of using MonoGame. The library support for MonoGame is fairly limited and involves stitching together bits and pieces to make something work. Ideally, it would be nice to have a more comprehensive environment so that we can focus on improving or custom implementing specific subsystem, like AI, procedural generation, rendering, etc., so that more focus can be placed on the gameplay and the technical achievements instead. Writing the engine while simultaneously writing technical systems on top of the engine is a tough task for such a short timespan.

I think the theme for this year was good. It gave a sense of direction for all the games while not being too limiting, definitely better than last year's in my eyes. I think total freedom would let teams come up with a broader spectrum of more creative and innovative projects but this would come at the cost of structure and would still prefer a theme.

6.3.3 Julian Bauer - Art Direction

The development of HAZMAT over the first semester helped me improve my communication and art management skills in an interdisciplinary environment. This project stood out from all the other projects I have done so far because of the well-structured project management of our producer. I consider myself lucky to have had the opportunity to work in an utterly friendly team. The development was a challenging and fun process and a valuable experience for me.

The biggest challenge as an Art Director for this project was to ensure a visual direction in the early stages of development and finding the right balance between applying the feedback from my team and my own vision.
Another big challenge was to figure out the pipeline of asset export and integration. The time we had to dedicate to figuring out the right process that would let us produce assets on a larger scale postponed most of the art production to the second half of the whole project.
Another task as an AD was to communicate the visual red line to other artists and to the programmers to guarantee a consistent visualization of the game.

As a team, we agreed to interpret the theme "2099" as a year, which was not quite in my favour as an AD since the setting of a near future already appears in the game industry quite often and felt uninspiring to me, however my efforts to convince my team with a more abstract game concept that distances itself from the idea of 2099 as a year was not successful and I had to adjust to my team's wishes.
One of the biggest problems of this course was the unjustified **2 ECTS** points for the game designers. This is an unreasonable amount relative to the workload we had to put on our shoulders to bring the game to an acceptable state.

Of course, the workload always depends on the kind of game being developed; however, a lot of participants from the ETH have never developed a game before. They were unaware of the bottom part of the floating iceberg of game development when it comes to game design decisions and would therefore often underestimate the available time frame for accomplishing their initial vision - even after the rest of the team communicated the impossible amount of work that would need to be done to meet these ambitious goals. As a member of a new team without any hierarchy yet in place, it is hard to change the mind of team members without being instantly categorized as a mood breaker and an utterly pessimistic person. Personally, I decided to prioritize healthy social connections with everyone instead of insisting over changes to the game concept that would, in my opinion, lead to more inspiring game mechanics
.
It feels bad to admit it, but I feel like it needs to be said: the end presentation also showed uninspiring game mechanics in the games of the other teams. There were a few exceptions, and sadly HAZMAT was not one of them. It’s understandable, given the tight deadline for brainstorming ideas and actual production, however I am disappointed to see myself and other students invest so much time and energy into their game, only to develop a poor man’s version of something that has been done before multiple times.
In conclusion, the biggest mistake the ETH Gamelab made was to give everyone the possibility to decide on a game concept. It would have been more practical to leave this task to those who have studied and already gained experience in the field of game design; therefore, eliminating the time that we needed to spend trying to consolidate everyone’s ideas and gently trying to put down the enthusiastic but unoriginal concepts from inexperienced team members.

The ETH Gamelab is looking for new innovative game concepts; therefore, I would suggest a requirement for the next Gamelab that encourages the participants to find game mechanics that haven’t been developed before instead of working around a theme. Perhaps it would be easier for the programmers to be creative if they had to think in terms of innovative mechanics instead of focusing on the visual and storytelling aspects of the game.

6.3.4 Leonardo Del Giudice - Algorithms Programmer

The course itself was well structured and helped me a lot by reducing this huge project in smaller tasks. I’m happy with what we achieved in the very limited time frame we had, it was very satisfactory to see the game evolve and grow. Working with theme had a lot of positives, as we always had to find clever ways to insert references to the ambient set by our futuristic setting, but never felt limiting, as it was very generic and open to interpretation.
I had a very hard time using monogame and finding workarounds to the many problems it has, also the lack of any documentation made it even harder to do what we wanted to do.
My biggest success was building my own (very basic) AI with builtin pathfinder, it was a huge project and felt really satisfying when my AI managed to take down any real world player. On the other hand I feel that there would be so much to change and optimize, as for any other aspect of the game, I’m very happy with what we did, but I do not consider it a finished project.

6.3.5 Nihat Isik - Visual Programmer

Yes although we didn't manage to implement one important part of the game which was coop. I'm proud of our game but the time wasn't enough to do everything.
Biggest technical difficulty from my side was trying to implement all the different shaders and understand how monogames pipeline works. I'm happy with a theme since it gives a hint on a

common direction from all the teams. I'm really happy with my team and with what we have accomplished.

6.3.6 Julian Schönbächler - Technical Artist, Game Designer

The most valuable thing I could draw out of this course, was the interdisciplinarity and the exchange with the students from the ETH. We had a great team and I could expand my network in knowing people from different fields interested in the same thing. The structure of the course was not bad, but for me personally not really beneficial. I think especially the design aspect could really need some improvements. I realise that the course is primarily tailored at the technical side of game development, but in order to support innovation in finding game mechanics, general basics in design are helpful. The inputs we had in the classes haven't touched this area deep enough in my opinion.
Overall I'm very happy with the project, our output and the work of everyone in the team. We reached quite some milestones and it was a super fun experience.

Appendix 1: Task Lists

Table 1: Development tasks

Task	**Responsible Member**	**Time est**
Functional Minimum		
Player Spawn	NI SK	4
Input Handler	NI	10
Player Movement	NI	8
Shooting Mechanic	NI	8
Enemy Spawning	LDG SK	8
Basic Enemy AI (spawning, move towards player)	LDG	5
Energy System	NI	3
Story screen	SK	6
Test 3D animations	SK	6
Implement placeholder assets	SK	4
Low Target		
Basic Procedural Generation (Power Plant w/Pathing)	LDG	20
AI	LDG	15
Very simple combining	SK LDG	20
Basic Combination Mechanic	NI	20
Player Pick Up Mechanics	NI	20
UI Direction Of Power Plant (Onscreen)	SK	10
Different Collectables - What enemies drop	JS	3
Basic HUD/UI (TIMER , MINIMAP)	LDG SK	15
Desired Target		
Final collectables		
Sound effects	NI	10
Shaders?	NI	30
Co-op	SK	5
Good AI	LDG SK	25
Procedural Map Generation - Medium (Path, Basic Environments)	LDG SK	15
Comprehensive UI (Start Screen, HUD,	SK	20

Waypoints)		
Event System (Story events)	SK	15
High Target		
Procedural Map Generation - Advanced (Path, 3 Environments, Connection to Power Plant)		
Boss	NI	30
Music	Outsource	
Tweaking Stats	LDG NI SK	15
Advanced AI	LDG SK	20
Extra		
Split Camera	extra	
More Biomes	extra	
More collectables	extra	
Emotes	extra	
Pets	extra	

Table 2: Art tasks

Task	**Responsible Member**	**Time est**
Functional Minimum		
Mood board: Environment	JB	1
Defining art style	JB SW	2
Mood board: Colour scheme	JB	0.5
Defining artwork resolution & dimensions	JB	0.5
Testing 3D Animations	SW	0.5
Character concepts	SW	3
Environment concepts	JB	3
Placeholder splash screen	JB	0.5
Placeholder story screen	JB	0.5
Placeholder assets 3D	SW	1
Placeholder Energy bar	JB	0.5
Low Target: ***PLACEHOLDERS***		

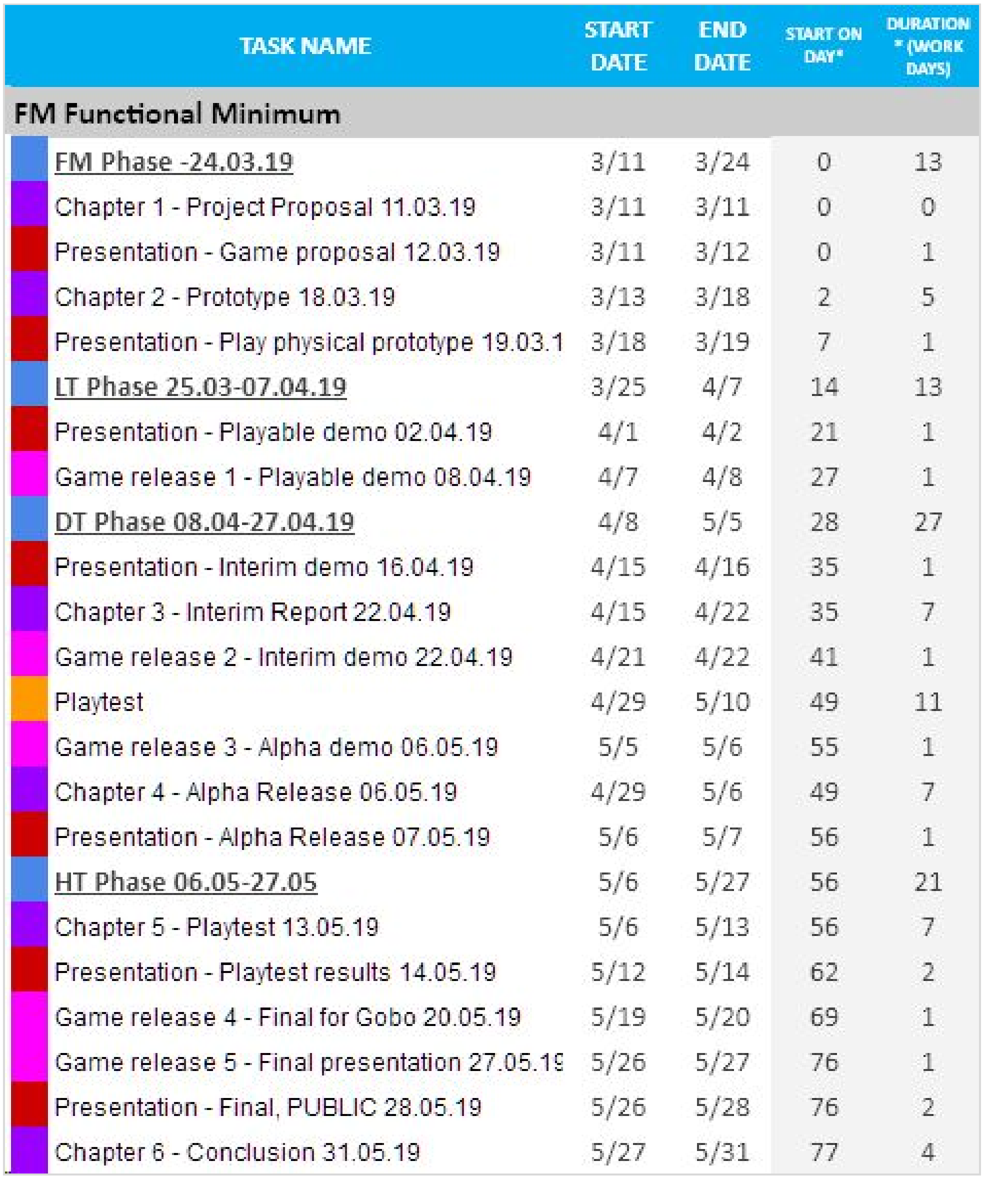
Procedural environment design, define needed tilesets	JB, SW?	4
3D Enemy	SW	3
3D Character	SW	3
3D Gun	SW	3
Collectable: Pop-up window design	JB	0.5
Collectable: Icon designs	JB	2
Futuristic UI overlay (Direction Of Power Plant) design & placeholders	JB	1
UI timer	JB	1
UI minimap	JB	1
UI buttons & frames from start to finish	JB	1
Desired Target		
Complete environment tiles	JB	8
3D Characters complete	SW	10
Basic animations	SW JS	5
3D items	JS	5
Shaders	JS	10
Polished game & story UI	JB	10
Sound effects	JS	5
High Target		
Detailed story screens	JB	15
Polished 2D assets	JB	10
Polished 3D assets	SW JS	10
Fluent animations	SW JS	10
Music	GM (Outsource)	

Task List 3: Miscellaneous

Task	**Responsible Member**	**Time est**
<u>Playtest</u>		
Questionnaire Design (what do we want to know? What do we ask players?)	JS	8
Defining playtest protocol	JS	1
Organisation of playtesters, date and location	JS	2
Conducting Playtest	JS	4
Evaluation of results	JS	8
<u>Report</u>		
Chapter 1 - Project Proposal 11.03.19	Team	8
Chapter 2 - Prototype 18.03.19	Team	8
Chapter 3 - Interim Report 22.04.19	Team	8
Chapter 4 - Alpha Release 06.05.19	Team	8
Chapter 5 - Playtest 13.05.19	Team	8
Chapter 6 - Conclusion 31.05.19	Team	8
<u>Presentation Preparation</u>		
Presentation - Game proposal 12.03.19	JS	3
Presentation - Play physical prototype 19.03.19	JS	3
Presentation - Playable demo 02.04.19	JS	3
Presentation - Interim demo 16.04.19	JS	3
Presentation - Alpha Release 07.05.19	JS	3
Presentation - Playtest results 14.05.19	JS	3
Presentation - Final, PUBLIC 28.05.19	JS	3
<u>Deadlines</u>		
Game release 1 - Playable demo 08.04.19		
Game release 2 - Interim demo 22.04.19		
Game release 3 - Alpha demo 06.05.19		
Game release 4 - Final for Gobo 20.05.19		
Game release 5 - Final presentation 27.05.19		

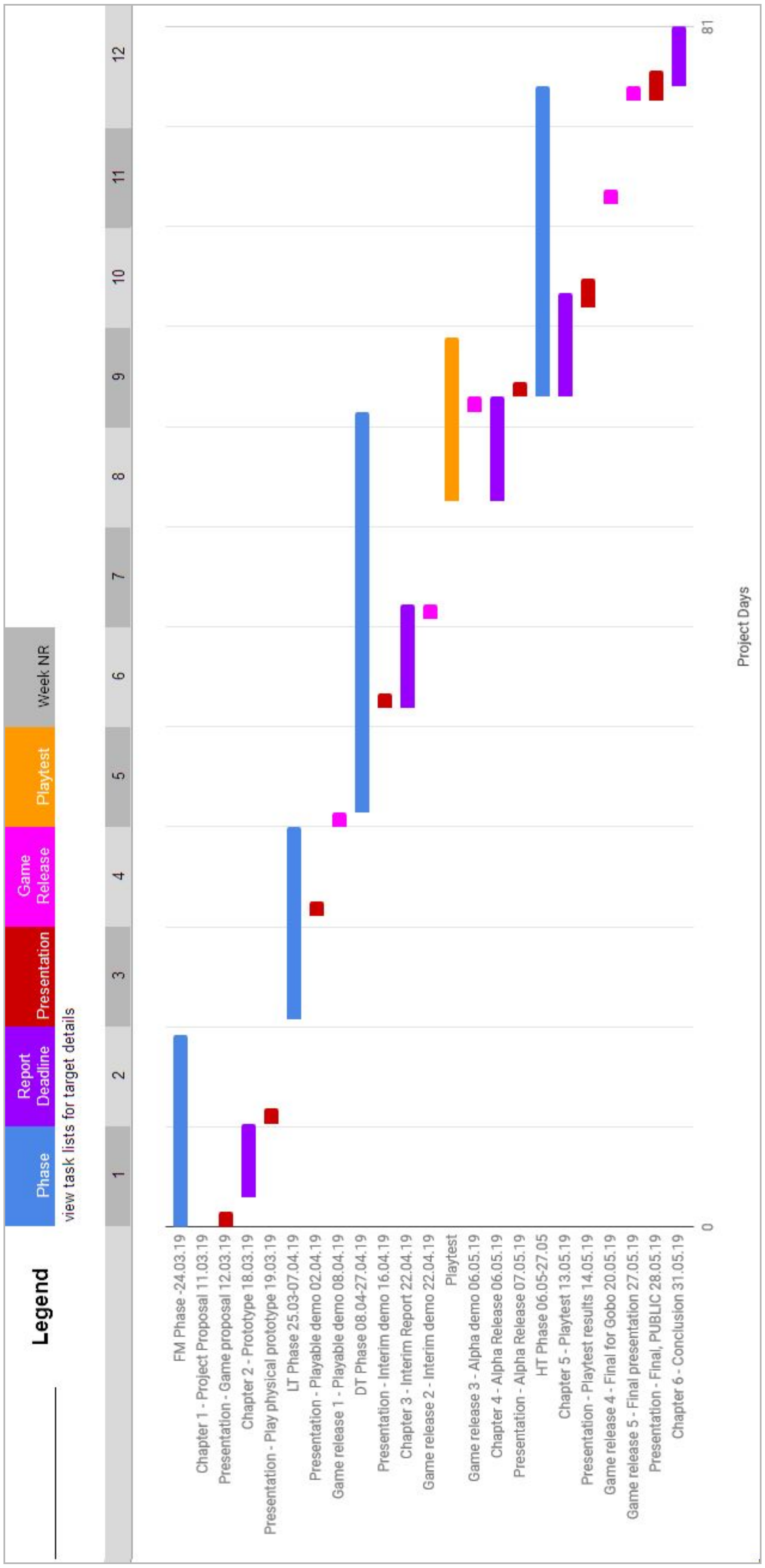
Appendix 2: Gantt Table

Gantt table pt.1



	TASK NAME	START DATE	END DATE	START ON DAY*	DURATION * (WORK DAYS)
	FM Functional Minimum				
🟦	FM Phase -24.03.19	3/11	3/24	0	13
🟪	Chapter 1 - Project Proposal 11.03.19	3/11	3/11	0	0
🟥	Presentation - Game proposal 12.03.19	3/11	3/12	0	1
🟪	Chapter 2 - Prototype 18.03.19	3/13	3/18	2	5
🟥	Presentation - Play physical prototype 19.03.1	3/18	3/19	7	1
🟦	LT Phase 25.03-07.04.19	3/25	4/7	14	13
🟥	Presentation - Playable demo 02.04.19	4/1	4/2	21	1
🟪	Game release 1 - Playable demo 08.04.19	4/7	4/8	27	1
🟦	DT Phase 08.04-27.04.19	4/8	5/5	28	27
🟥	Presentation - Interim demo 16.04.19	4/15	4/16	35	1
🟪	Chapter 3 - Interim Report 22.04.19	4/15	4/22	35	7
🟪	Game release 2 - Interim demo 22.04.19	4/21	4/22	41	1
🟧	Playtest	4/29	5/10	49	11
🟪	Game release 3 - Alpha demo 06.05.19	5/5	5/6	55	1
🟪	Chapter 4 - Alpha Release 06.05.19	4/29	5/6	49	7
🟥	Presentation - Alpha Release 07.05.19	5/6	5/7	56	1
🟦	HT Phase 06.05-27.05	5/6	5/27	56	21
🟪	Chapter 5 - Playtest 13.05.19	5/6	5/13	56	7
🟥	Presentation - Playtest results 14.05.19	5/12	5/14	62	2
🟪	Game release 4 - Final for Gobo 20.05.19	5/19	5/20	69	1
🟪	Game release 5 - Final presentation 27.05.1	5/26	5/27	76	1
🟥	Presentation - Final, PUBLIC 28.05.19	5/26	5/28	76	2
🟪	Chapter 6 - Conclusion 31.05.19	5/27	5/31	77	4

Gantt table pt. 2



Appendix 3: Libraries

Monogame

http://www.monogame.net
We will be using the **Monogame** framework to build the core of the game. It provides various hardware and graphics abstractions that will be used, like the core Game class.

Monogame.Extended

https://github.com/craftworkgames/MonoGame.Extended
We will also be using the **Monogame.Extended** library, which is an add-on to **Monogame** itself. It provides numerous extensions to the core behavior that we will be using, including an Entity Component System framework, a Physics/Collision system and various additional tools for graphics and animations.