HEAD4CHE PRODUCTION COMPANY

S2 PROJECT

3rd defense report

SUBWAY OUT

A1

Promo EPITA 2029

Jeanne Thalie DURAND

Tidiane BATHILY
Adam GRAZIANI

Arthur FIOLET
Valentin KIEP

Contents

I. Introduction

After eight months of work, the **Head4che Production Company** is excited to present their brand-new game ***SubWay Out***. ***SubWay Out*** is an exciting virtual escape game, set in the Parisian subway network. Available in both English and French, this first-person experience can be played alone or with a friend and offers a thrilling hour of puzzles and problem-solving immersive gameplay.

This final report offers an in-depth behind-the-scenes look into the hard work that led to the release of ***SubWay Out***. It presents the studio and its objective with ***SubWay Out***. It shares insight on all the features of the game, from the reasons behind their additions to the technical challenges that marked their implementation. It records the challenges, regrets and joys each member experienced throughout the eight months of active development.
This report is also written with the objective of helping future game creators, by giving them tips and examples for their future projects.

II. Head4che Production Company

1. History

The **Head4che Production Company** is a French video game studio founded in 2024. Composed of five active members, this studio was born from a shared passion for computer science and puzzles. Specialized in the creation of escape games, the **Head4che Production Company** offers unique and innovative experiences in the puzzle game genre.
It is a passion for solving puzzles and the drive to do so as fast as possible that motivates the team to create a video game open to the public, blending the challenge of complex puzzle-solving with the freedom offered by a three-dimensional environment.
The studio's composition stems from the way each member handles their academic work. Even prior to the studio's formation, the group often gathered to work together. Specifically, the group used to meet to work together on OCaml, the very first language studied by each team member. This efficiency for academic classes was thus important because working together created strong relationships and more importantly lasting friendships inside the group. Therefore, the **Head4che Production Company** is a studio where puzzles and complex reasonings are a fundamental part of the team's passion but above all, it is a studio where friendships have been made.

2. Members

a. Jeanne Thalie DURAND (Project leader)

I have always been interested in the field of computer science. Being accepted into EPITA, a computer science school will allow me to turn my passion into a future job. Eager to discover and learn more about computer science, this project has been an incredible opportunity to develop new skills and gain experience.
Methodical, organized, curious and loving complexity, puzzles stimulate my creativity and my reflection, which is something that I particularly enjoy. ***SubWay Out*** is a game that combines my interests for computer science and my appeal for puzzles. The realization of this project has enriched my technical knowledge concerning the use of ***Unity*** and also my coding skills in C#. In addition, ***SubWay Out*** helped me develop my social skills, as this was collaborative work done within a team and especially as I took the responsibility to be the project leader, a role that I am particularly proud of.

b. Arthur FIOLET

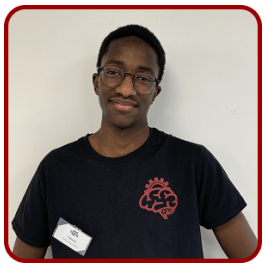
After discovering computer science, I chose to dedicate myself to the passion it became. I did so by joining the **EPITA** class of 2029, by taking part in several related contests, and also through several projects. For instance, I created a platform to manage events, an inventory management software, and some other personal initiatives. Developing the video game ***SubWay Out*** is part of this dynamic.
This passion for computer science is caused by a deeper interest which pulls me towards abstract and complex thinking. This is the reason why puzzles and escape games are so interesting to me, and because of that, ***SubWay Out*** is a perfect opportunity to develop my technical skills.

c. Adam GRAZIANI

My passion for technology has always been a part of my life, evolving from a child's amazement at electrical switches to computer science studies. My dream is creating programs that will be useful to people.
I have been contributing to a gaming wiki for more than five years, over the span of which I have created numerous MediaWiki templates and LUA modules. This year, I have been elected to the board of directors of **Block & Quill Ltd.**, the company that manages this wiki of over 1,400 active contributors across 12 languages.
SubWay Out is the first video game I have had the opportunity to create, and I hope I have been able to create memorable moments for the players. The project has also taught me how to use Unity and strengthened my knowledge of the C# language, but most importantly has immersed me in the dynamics and management of a group project in the field of game development.

d. Valentin KIEP

Self-taught developer, mainly in JavaScript, I have been learning how to code in order to develop ***Discord*** bots and automatizations with friends since middle school. Nowadays, computer science is my main field of study allowing me to create more advanced programs and in other coding language like C#.
As a video games amateur, I have wanted to create one since my childhood. This field takes a particular place in my heart as it has helped me to evade the real world when I was facing events that affected my life a lot.
SubWay Out has allowed me to make my first step in the world of game design and game development. This experience gave me an opportunity to put my passion for video games to use.



e. Tidiane BATHILY

I have always loved new technologies and problem-solving. Studying at **EPITA** is an opportunity to learn about these areas that I am passionate about. I have had the opportunity to use ***Unity*** to develop video games. My knowledge of ***Unity*** was an asset for the group, and this project taught me more about it. What is more, this project has put my logical mind to the test by moving on to the puzzle design side of things, and allowed me to discover the field of video game design.

III. SubWay Out

1. Origin

The subway is the most popular mode of transportation in Paris, with more than 1.4 billion passengers in 2023. For the majority of Parisians, it is part of everyday life. However, it is an often neglected space as people do not pay attention to their surroundings. All of the members of the **Head4che Production Company** use the subway every day. The team had the idea to set the IT project in this context while waiting for a train on a subway platform. The **Head4che Production Company** opted for this original theme in order to offer a unique challenge to the users.
Once the theme of the project was decided, the game needed a proper name. The team debated lengthily about it and decided to opt for "***SubWay Out***". Both the meaning of the words and the way it is written matter a lot. ***SubWay Out*** is perfectly appropriate for this game because it evokes the environment in which the game takes place as well as the objective of the game itself. The English word "Subway" sets the tone for the public, who instinctively knows which environment the game is about. In addition, the part "Way Out" describes the objective of the game: to escape from the subway. Therefore, the way ***SubWay Out*** is written matters a lot, as the upper-case letters separate the different meanings.

2. State of the Art

Behind Closed Doors (1988)
The very first escape game was created in 1988 in the form of the video game ***Behind Closed Doors***. In this adventure game released by the **Zenobi Software** studio, the player interacts solely through text. They embody Balrog, a man locked in his toilet, a room he must escape. The offbeat humor of this game brought it success, despite its very short playtime.

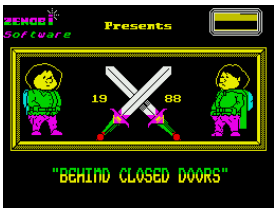


Figure 2.1: Loading screen of ***Behind Closed Doors***

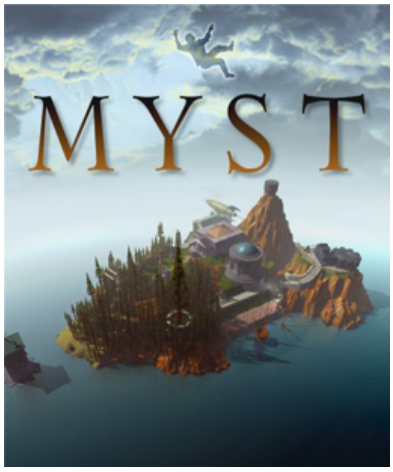


Figure 2.2: Cover of the 1993 edition of ***Myst***

Myst (1993)
Despite the relative success of ***Behind Closed Doors***, the escape game genre was still little known before 1993. The video game that changed this trend was the Macintosh title ***Myst***, developed by **Broderbund**. In this adventure game, the player is transported to a mysterious island by a very special book. On the island, they must solve a series of puzzles.
The game's success earned it the title of "worldwide masterpiece" and contributed to the popularization and development of the escape game genre.
This success was so significant that, unlike the other games mentioned here, ***Myst*** is still maintained to this day, with new versions continuing to be produced and published, in order to make the game available on new platforms, including virtual reality. Its innovative design made it the best-selling computer game for more than a decade. As such, ***Myst*** has been a major source of inspiration for the evolution of puzzle games, especially in the particular case of escape games.

Crimson Room (2004)
Crimson Room is a game created by Toshimitsu TAKAGI in 2004. It is considered as the ancestor of escape games, as it was the first "point & click" escape game. This game established the typical visual look and gameplay style of most escape games following it. Just like ***Myst***, it achieved great success and had an important role in the development of the escape games known today.
Its realistic look led to a new way to enjoy escape games: escape rooms. Indeed, counterintuitively, escape games (video games in which the goal is to escape from a restricted area) are older than escape rooms (physical games in which participants try to escape from a room in a limited time, usually one hour).

Figure 2.3: In-Game visual of ***Crimson Room***

3. Objective

a. Identity

SubWay Out aims to distinguish itself from other escape games thanks to an emphasis on immersion. One specificity of the game is the fact that it is played entirely in first-person perspective in a three-dimensional environment rather than being a classical "point & click" adventure.
Another specificity of the game is the way it handles puzzles. In most three-dimensional escape games, puzzles are elements in the games that the player can click on to open a pop-up frame on which the puzzle can be solved. This completely cuts out the surrounding environment and breaks immersion. In contrast, ***SubWay Out***'s puzzles are all solved directly within the game space, without interruptions. Another advantage of this system is the effect on multiplayer gameplay. Instead of having multiple players solving individual puzzles in parallel, the players can find solutions together in their common game space.

These particularites create a gaming experience unique to ***SubWay Out***, which helps it stand out amongst games of the genre.

b. Target Audience

The game is mainly designed for puzzle enthusiasts, escape game regulars and users of the Parisian *Métro*. Nevertheless, users with no real experience in solving puzzles will be reassured to learn that a hint system can be activated at any moment in the game to guide them.
The game is based on the Parisian subway. Even though players might never have traveled aboard it, ***SubWay Out*** offers an opportunity to discover it through the in-game environment. For Parisian *Métro* regulars, the goal is to get them to notice the little details hidden inside the car.

4. Scenario

The player is a passenger travelling on the Parisian subway. From their noise-canceling head-phones, they hear soft and gentle music that helps them to relax after a hard and exhausting day of work. Tired of this soporific and ordinary ride home, lost in the flow of passengers in the car and lulled by this pleasant sound, the player inadvertently falls asleep.
When they wake up, there is a sense of disbelief: the player is alone in the subway car, which is still moving. The other passengers are gone, although their luggage is still there, lying on the floor, as if it had been left behind in precipitation. The lights are flickering, a sense of anxiety is palpable, the atmosphere becomes oppressive: something has happened while the protagonist was asleep. The player's main objective remains precise: to escape from the subway.
Therefore, the objective of ***SubWay Out*** is to escape from the Parisian subway after falling asleep in a subway car.
To do so, the player will be able to be joined by a second player. In a cooperative way, they will need to solve puzzles more or less complex. After succeeding to do so, they will be capable of getting off the subway.
The game has no time limit and is catered to offer a one-time-only gaming experience thanks to its puzzles. Indeed, after solving a puzzle, the player will know the solution and thus be able to solve it again very quickly. The game is not meant to be rushed but rather to be taken at each player's own pace, giving them time to think about the puzzles.

5. Communication

a. Instagram

In order to communicate with its community, the **Head4che Production Company** has decided to be present on the social media app ***Instagram***. The company shares the overall progress of ***SubWay Out***, interacts with its followers and answers potential questions that are asked. This presence on social media allows the **Head4che Production Company** to stay closer to its target audience. For instance, task distribution among the members of the company, teasers of the game in development or even some videos of humoristic mistakes made during the project are shared on this social platform.

Figure 5.4: ***Instagram*** account of the **Head4che Production Company**

b. Website

The website is an essential communication tool for ***SubWay Out***. Being a video game, its website is the main interface where future players get their first impressions of the game and where they can download it. It is the central hub of information, as it is a direct way to communicate with the community. The ***SubWay Out*** website has been designed to match the standard quality of other game websites.

IV. Project

1. Game Design

SubWay Out is an escape game. This means its entire gameplay revolves around solving various puzzles presented to the player. As such, the puzzles had to be fully designed before development could start. Above that, the complete flow of puzzles had to be clear to each developer so that the system they develop fulfills its objective perfectly and aligns with the rest of the game.
To tackle this, the level design team created a full graph of the game systems' interactions. The graph was generated via the Graphviz DOT language in order to be both visually clear and easily modifiable as the project evolved. The resulting graph was available on the studio's internal GitHub repository for all developers to see, and ensured that all members had the same clear understanding of what was expected of them.
The full graph is attached to this document as appendix A.

2. Game Engine

a. Grab & Interact

The player can interact with their environment in two main ways: grabbing and interacting with various elements within the game. Each action is mapped to a separate button: right mouse-click for grabbing and left mouse-click for interacting.
Not all objects are responsive to the player's actions. Some can be grabbed but have no interaction, some have a specific interaction but can not be grabbed, while others can react to both at the same time.
The major challenge in implementing this system was finding a way to detect which object the player is trying to perform an action on. It was decided to cast a ray from the player's camera and store the objects it encounters. This data is sorted by distance, then interpreted in order to find the relevant object. The ray cast obeys different rules depending on whether it is meant for grabbing or interacting, as illustrated below:

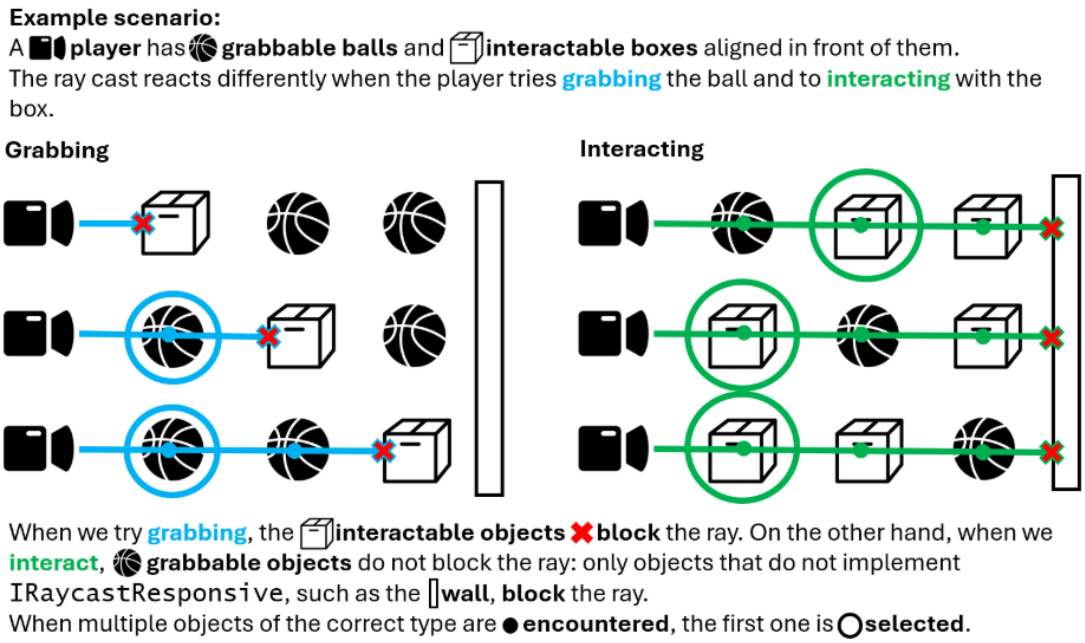


Figure 2.1: Different behaviors of the rays cast by players

It is important that interactable objects block rays cast when the player grabs in order to avoid the player grabbing an object inside an interactable box (i.e. if the box can be opened). On the other hand, grabbed objects should not block rays cast when the player interacts with an object, as when a player is holding an object, it tends to position itself at the center of the screen. Having the grabbed object block rays would mean that, while it is at the center of the screen, the player would not be able to interact with anything. This is especially an issue as, multiple times throughout the game, the player has to interact with an object while holding another one in order to combine them.
This was made possible by a specific class hierarchy, detailed below, such that:

- When the player grabs, the ray passes through all `IObjectGrabbable`, implemented by grabbable objects.
- When the player interacts, the ray passes through all `IRaycastResponsive`, implemented both by grabable and interactable objects.

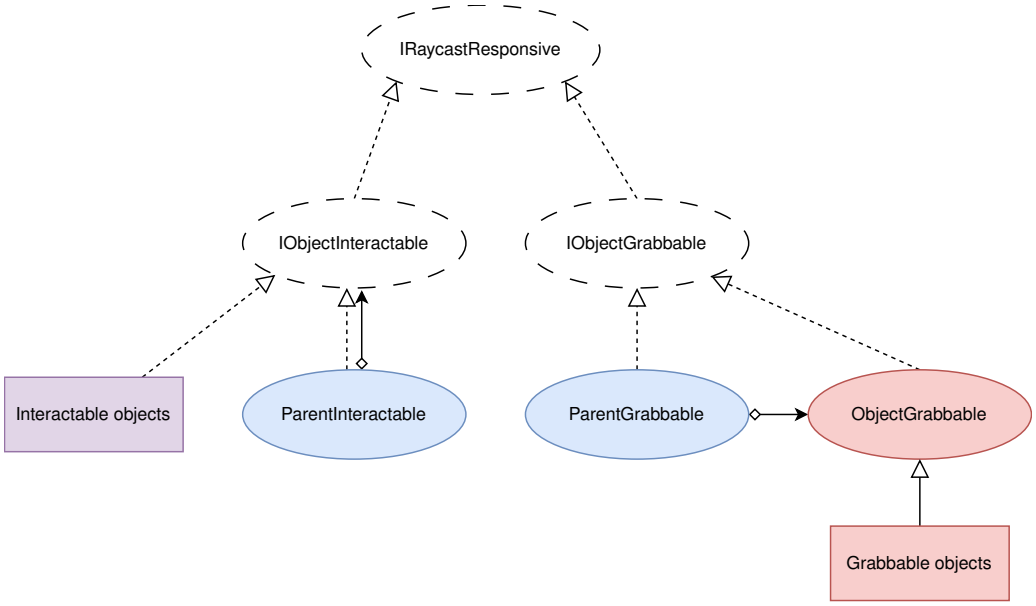
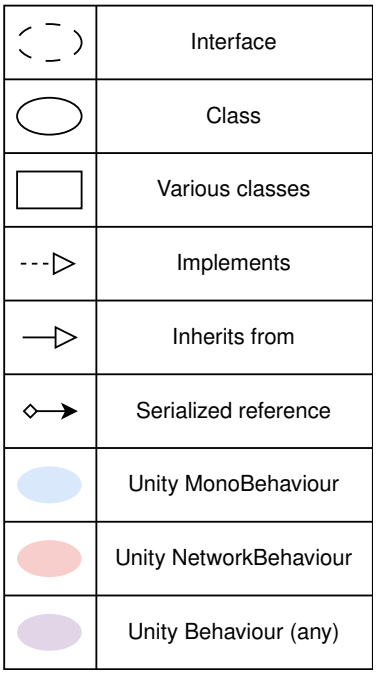


Figure 2.2: Class hierarchy for different object types



Grabbable and interactable objects are then implemented differently.
Interactable objects use a simple system: they implement `IObjectInteractable`, which contains a single method describing the behavior of the object when the player interacts with it. If an interactable object has multiple parts, each part inherits from the `ParentInteractable` class. The class contains a reference to another `IObjectInteractable` object: the root object, also referred to as parent object. When the player interacts with a part, the interaction is in reality happening with the referenced parent object.
Grabbable objects are considerably more complex. When the player grabs an object, its position must be updated dynamically. Furthermore, the transformation must be synchronized over the Network so that the object stays at the same place for all clients. There are also restrictions on when the object can be grabbed: two players can not grab the same object at the same time.
This behavior can not be handled solely by an interface: all grabbable objects must inherit from a `NetworkBehaviour` class that handles all the logic and physics: `ObjectGrabbable`. If a grabbable object has multiple parts, each part inherits from the `ParentGrabbable` class. The class contains a reference to an `ObjectGrabbable` object, representing the root object, also referred to as parent object. When the player grabs a part, the grab instruction is transferred to the parent object since only `ObjectGrabbable` can handle the logic correctly. This is especially important since, under ***Netcode for GameObjects*** limitations, only a single `NetworkBehavior` should be present per Prefab, and in ***SubWay***

Out almost all objects are Prefabs.

b. Highlight

To keep the game flow natural and avoid any frustration for players, a highlight system was implemented. Because players do not have an inventory to keep items they found in the game, it was decided to allow players to see an outline of the item
This system had two versions.

First version
This early version highlighted every object players interacted with. However, it resulted in visual clutter that made navigation difficult.

Second version
This final version allows players to highlight objects by pressing the `alt` key. Doing these changes prevents players from being distracted by constant outlines. Otherwise, keeping the objects constantly outlined may obfuscate the players' view.

c. Network

It was decided that ***SubWay Out*** would rely on a Client/Host configuration for its multiplayer architecture. ***SubWay Out*** is made for two players, which means that one of these two, called the host, has their computer used as both a server for the game and as an interface to play. The other player is a client, which means that their computer connects to the host's server, thus allowing the players to enjoy the game together.
The studio developed two iterations of its networking system, as described below.

First version
The first attempt at establishing a multiplayer connection through the network was done, using only the ***Netcode for GameObjects*** Unity package, and connecting players manually using the host's IP address encoded as a join code to link the client to the host. However, this system had many downfalls. On the one hand, it required to share the public IP address and a port of the host to have a multiplayer experience. On the other hand, the direct connection through IP required opening a port on the computer of the host to connect two players. This requirement made playing in multiplayer technically difficult, as opening a port had to be done manually by the user. This also raised security concerns. Thus, it has been decided to rework the entire system.

Second version
In order to fix the issues encountered in the first version, it was decided to use two services proposed by **Unity**. The first one are ***Unity Relay Servers***, which serve as intermediaries between players, allowing two clients to communicate without knowing the IP address of each other and without requiring opening any port. The second one is ***Unity Lobby***, which allows any player to create a session that another client can join thanks to a "join code", which is a six-character long string.
As mentioned above, the multiplayer system works with "sessions". Each one of those corresponds to an instance of the game, and in each of these sessions, players are able to experience what ***SubWay Out*** has to offer. A session is obviously temporary, as when all players leave, the session is destroyed. When players are connected, all interactions happen through the network and inside the session. Communication between players is handled by ***Netcode for GameObject***, ***Unity Transport*** and ***Unity Relay***. To communicate between clients, Remote Procedure Calls (RPCs) are used. They allow one

client to execute code on precise targets. Most of the time, this target is either all players in the session or only the host.

3. Assets

a. 3D Models

For the realization of the 3D models, the **Hea4che Production Company** decided to use ***Blender***, an open-source 3D creation software. Creating the 3D models from scratch allowed the project to have a more personal and distinctive look. Therefore, a lot of time was taken to design the necessary three-dimensional objects. The modelization of all the game elements was a challenging part as it was necessary to think about every element that the game needed. In order to be easily recognizable, ***SubWay Out*** opted for a dualistic visual identity, using a contrasting realism for the different elements of the game. The subway is very similar to real life, so that anyone who has ever used it would be able to recognise the scenery, but the design of the different characters is humoristic and comical.

Subway
The main parts of the game environment are the subway cars. Their models were based on the Parisian style, and more specifically on the *Métro 7*. The current car in use for this line is the MF77. Since all the members of the **Head4che Production Company** use this mode of transport, it felt natural to use this line as a reference for ***SubWay Out***. Seeing in life size what elements were needed in this project facilitated their 3D modeling in ***Blender***.
In the game, the subway is made with two different models, which were designed separately to get a better and complete representation.

These two different parts are:

- The head of the train
- The other cars

The head of the train, in which the driver's cabin is located, is the only playable car. The cars attached behind it are non-playable and only used for the scenery.

Non-Playable Cars
The first version of the subway's model became the design for the non-playable subway cars. The model has been duplicated in order to obtain a train of two non-playable cars. To join them, a coupler between the subway cars was added. A coupler, also called a coupling, is a mechanism used for railway vehicles or subways in order to connect cars together. It is located at the end of each car. Including them makes the connection between cars more realistic.
The 3D model of the door at the back of the car has been reworked during the project in order to prevent the player from glitching through it and escaping the subway without solving any of the puzzles. Textures were changed to accommodate the new design. Finally, the major element that is located in the non-playable cars is the instruction to unlock the final code to enter in the driver's cabin. It forces the player to search in its environment to find the code instruction.
The non-playable cars were implemented into ***Unity*** without any major issues. A simple file conversion to FBX, with embedded textures was enough to put the cars in ***Unity***.

Playable Cars

This model is the most crucial one of ***SubWay Out***, as it is the place where the player spends the most time in. The model of the playable car is separated into two subparts: the passenger part where the player begins their adventure and the driver's cabin where the player solves the final puzzle. Below is a schematic representation of this car.

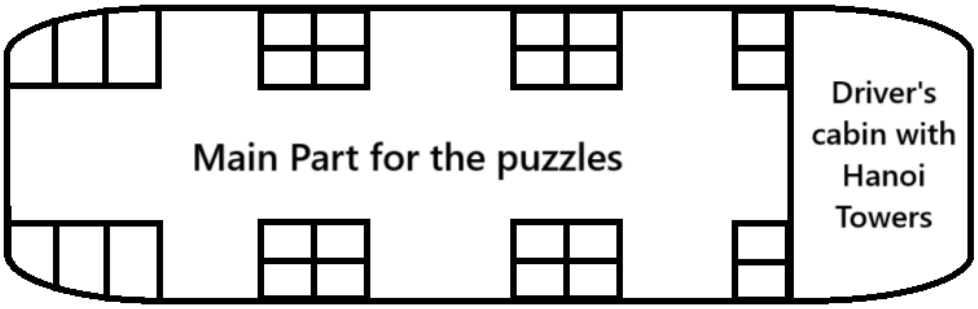


Figure 3.3: Layout of the playable car

The driver's cabin contains a dashboard, control buttons and some screens to add details. In addition, in this cabin, there is the model of the last puzzle: Hanoi Towers. This part of the subway is exclusively used for the resolution of this puzzle.

In the passenger section, several elements were modeled. First of all, the seats were created with a particular texture, mimicking the look of the seats on line 7 in order to ensure a better recognition of the scenery. In that way, players will thus be able to recognise a familiar environment, the Parisian subway. The seats were individually cut so it was easier to handle the animations. Modelizing elements with appropriate textures was the main concern. For example, the windows were a challenge as they were too transparent and thus not discernible by the player, who then could not tell if the window was opened or closed. This was inconvenient as the windows are useful for one specific puzzle. Even if they are cut in several interactable sections and even if textures were reworked to decrease their transparency, it was still difficult to see when a window was opened or closed. Adding an extra texture on the edges of the cut parts solved this issue.

Figure 3.4: Doors of the subway

Another problem for the windows was their collision. At one stage of the project, they did not have collisions, allowing the player to throw objects through them. This was a major issue as objects are needed to solve the puzzles. Mesh colliders were then added to avoid that.

The playable car was also modeled with specific doors. In fact, each door is built with several elements. As shown on this image, a door is constituted of four components: a major solid part, a window, a button holder and a button. It is fundamental to separate these components as the button needs to have its own behavior to open the doors.

To add realism to the subway, three vertical handrails were added to the car, anchored between the floor and the ceiling.

Characters

For the characters, a humorous style similar to that of **Nintendo**'s Miis was used. The game currently has two different player models. At first, the studio wanted to have five playable characters representing every member of the group. However, two designs are sufficient for ***SubWay Out***, as the game is only playable with two users. Therefore, it would have been a considerable waste of time to model three extra three-dimensional characters. Opting for character designs that represent at least two members of the **Head4che Production Company** reinforces the closeness that the studio wants to establish with the community of ***SubWay Out***. Creating a model matching the style envisioned by the studio was a challenge. Different versions of the models were created before settling on a final one. For instance, earlier player models were more square whereas final models are more round. The characters were also difficult to make because of their hair. It was important to keep a low-poly style but, at the same time, the members of the **Head4che Production Company** represented by the model had to be recognizable.

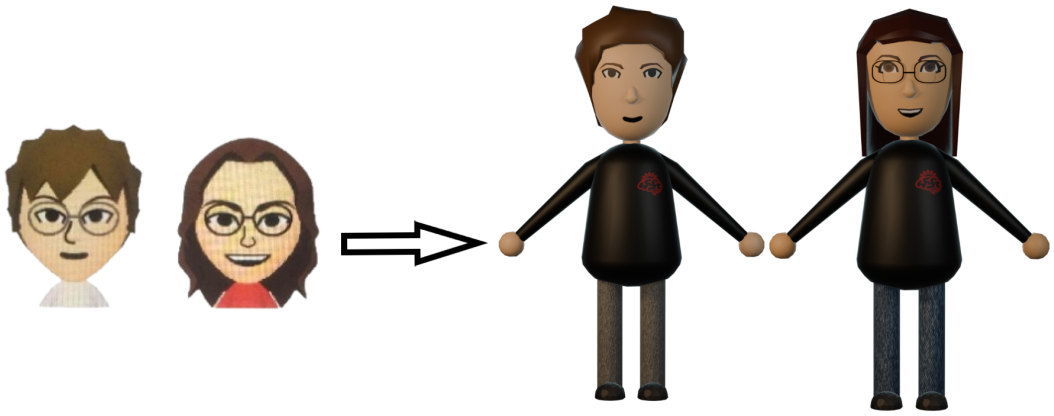


Figure 3.5: Inspiration and final design for the characters

An interesting fact to notice is how the models are dressed. Efforts were made to recreate the company's branded tee-shirt. With the logo of the company on it, this tee-shirt is also a way to represent the studio in the game directly.

Game Elements

Platform

Even if the subway cars are the main three-dimensional components of ***SubWay Out***, the platform is essential for both the beginning and the end of the game. Indeed, at the very beginning of ***SubWay Out***, the subway stops with doors stuck in order to indicate clearly to the players that they are stuck in it. This platform appears a second time at the end of the game. After solving the Hanoi Towers puzzle, the doors open and the players need to exit the subway car to finish the game. Therefore, a subway platform was needed. To stay as realistic as possible, the modelization of this game element was done after taking a closer look at the real Parisian stations. The colors and textures are aligned with the real ones, whether for the walls, which were designed with tiles materials, or for the presence of advertisements that reflects the consumer society in which players live in. In addition, a strip for the visually impaired people was created. The modelization of this platform is thus really realistic and

helps the players to recognise their daily-life environment.

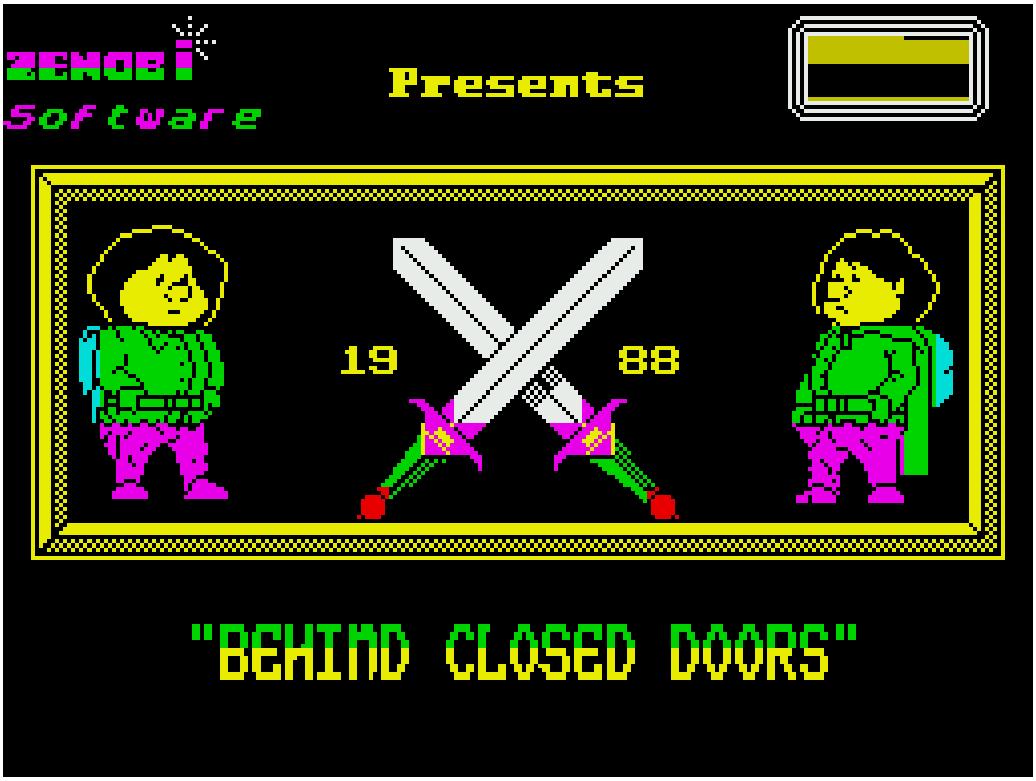


Figure 3.6: In-game subway platform

Metro Map

In the subway cars, a fictional subway line was created. References to this line in-game helped the cars feel more realistic. The color of the line and its number are the same as the Parisian *Métro* line 7, as it is the one used by each member of the studio to go to **EPITA**. However, the **Head4che Production Company** added its own comical touch.

As explained earlier, the dualistic visual identity in ***SubWay Out*** flows through the project, from the start to the end of the modelization. It was essential for the group to add a comical element inside the game alongside the character models. For the inspiration of the names of the stations, the Parisian subway lines were extremely helpful.

The group decided to combine famous names of Parisian stations with the term “camel” in English, or with its French translation “chameau”. In fact, the group gathered around the same joke: that of transforming all French words by adding the word “camel” in it. This idea was originally born thanks to the first programming language that the members of the **Head4che Production Company** studied: OCaml. In reference to its name, this language is represented by a camel. Therefore, the names of the stations inside the game are a combination of both Parisian station names and the creativity of the studio to invent original names based on a common premise.

Figure 3.7: Fictional subway line

Emergency Box

Another model designed and added into Unity was the box for the hint system. Custom textures were designed to clearly indicate where and how to find help during the game.
To go with this hint system box, a trigger can be inserted in the box. The choices made for the colors were pretty simple. Indeed, the red and white are predominant colors for this model because they are the colors associated with emergency situations. Red is particularly efficient to grab the attention of the players, and thus useful to indicate that this model is associated with external help such as hints.

Figure 3.8: Texture of the emergency box

Backpack

To represent the luggage forgotten by other passengers, a model of a backpack was produced. Designed in a similar way as an **Eastpak** backpack, this model has a pocket at the front and more precisely the logo of the **Head4che Production Company** on its main part. The backpack is interactable meaning that it can be in a close or open position. Inside the backpack, there are several elements that can be used for the rest of the puzzles. Thus it is relevant to hide objects in it, as if the objects were part of the passengers' belongings.

Figure 3.9: Backpack open and closed

Cheese

One element that is placed inside the backpack is the cheese. The cheese model is very simple. It is a simple wedge, as if it was a part of a wheel of cheese. With a yellow texture, its look is very discernible and with just a glance, players will understand that this model symbolizes a cheese. The difficult part was to create the several holes in it but after several attempts, the cheese was successfully created.

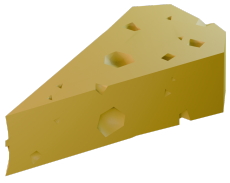


Figure 3.10: In-game cheese

Figure 3.11: In-game cheese ad

Cheese Advertisement
Another game element in the continuity of the backpack and the cheese is a cheese advertisement and its casing's model. As explained earlier, the realism is essential to allow the players to recognize their environment: that of the Parisian subway. Nowadays, advertisements are displayed in the subway so that a lot of people can see them everyday. The **Head4che Production Company** has decided to incorporate an advertisement to reproduce this environment. Nevertheless, it is not a simple advertisement: it is also useful for the resolution of a puzzle. To create this 3D model, several cubes were required to form both the poster, the glass pane and the frame. A keyhole was also added to point out the fact that the advertisements are the only non-electronic elements of the subway car of the game.

Key
A key is needed to open the cheese advertisement casing presented above. The modelization in Blender of this element was quick and easy. A cylinder forms the major part of it and then cubes represent the biting of a key. Obtaining this key is a part of one puzzle.

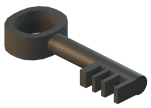


Figure 3.12: In-game cheese

Cat Cage
Another object that was forgotten by a passenger is the cat cage. This model was challenging because both the exterior and the interior of it had to be visible. In Blender, the rendering of the faces is very specific. Indeed, if one face is visible, then the opposite one is not, which makes it difficult to render both the outside and the inside of a face. To avoid seeing only one of the two faces, a duplication of the faces was made. It was pasted and rotated behind the visible face, which was then composed of two visible faces as shown on this graph below.
This model is interactable because the door of the cage can be opened or closed.
Consequently, it was necessary to cut the model in two parts:

- the main part of the cage
- the door part

In that way, only the door part can be moved and not the whole cage.

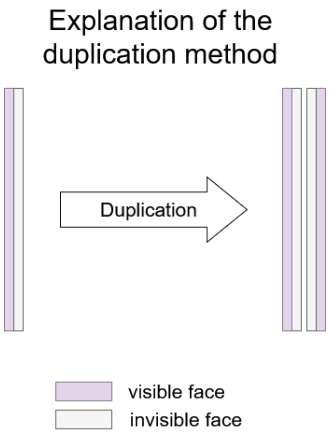


Figure 3.13: Duplication method illustration

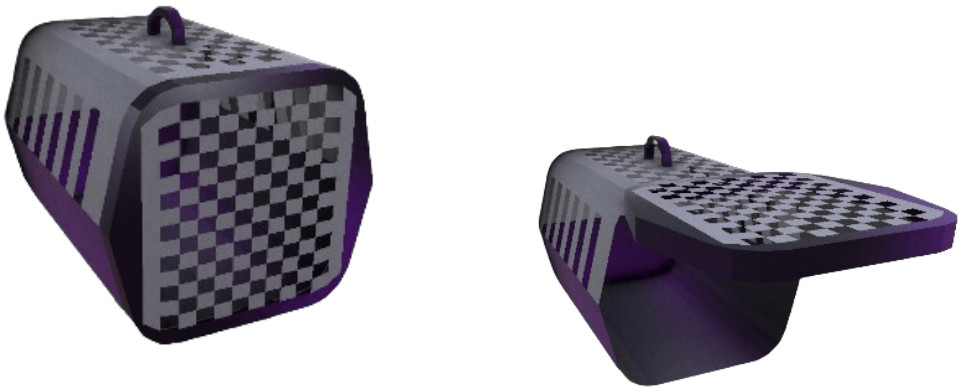


Figure 3.14: Backpack open and closed

Stickers
The studio also modeled stickers that represent the car the players are trapped in. They are very useful for one particular puzzle. The model is made of a cube with a custom texture.

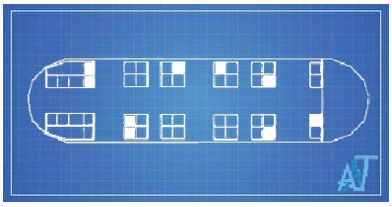


Figure 3.15: Duplication method illustration

Technical Supplies Box This model is made of two parts, in order to be able to open it. The design is copied from a real box in MF77 trains. Nevertheless, in reality, this box does not contain technical supplies but a static converter. Below is a comparison between a real box observed in the subway and the three-dimensional model that was created based on this examination. Details and resemblance were the main concerns for this model.

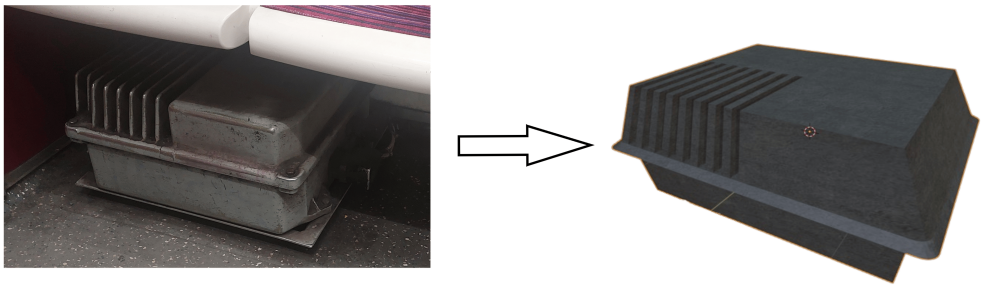


Figure 3.16: Inspiration and model of the technical supplies box

Hanoi

The model of this puzzle was the most complex to do. It is composed of two distinct parts: a box and the three pieces. It was challenging to model an accurate design for the puzzle to work correctly.

A piece in the Hanoi Towers puzzle consists of a puck, attached to a cylinder, that is itself attached to the main section of the piece. These pieces underwent several iterations to fix various issues in their implementation inside the puzzle. The choices made for their color matter a lot as they help the users understand that a piece with a darker color can not be on top of a piece with a lighter color.

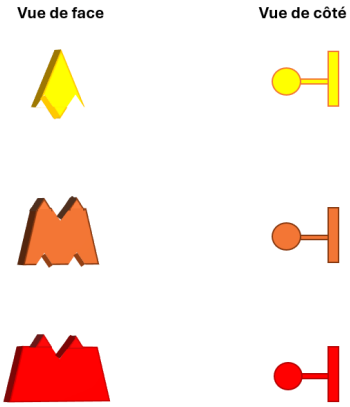


Figure 3.17: Hanoi pieces representation

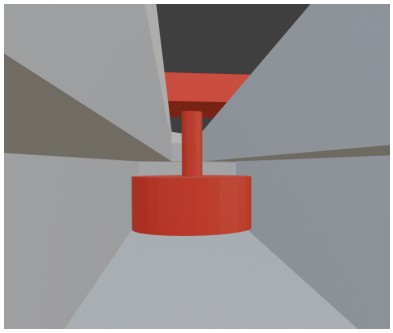


Figure 3.18: Hanoi mechanism to move pieces

The box for this puzzle was also difficult to make. The first version was made with perfectly rectangular shapes, whereas the last one is a mix with curved shapes. The curves are used for the inner path to facilitate the movement of the pieces. This path is similar to a rail where pieces could slide but not be removed. It was a priority to make sure that the pieces could not be taken out of the box, in order to make sure that the players would not break the puzzle.

Figure 3.19: Hanoi dashboard

Tunnel & Rat
Even if the subway is the most crucial component of ***SubWay Out***, the backdrop, in this case a subway tunnel, is also an important part in the creation of the atmosphere of ***SubWay Out***. The tunnel is a model imported from an external source with a compatible license. To strengthen the ambient effect, dim lights and ominous fog were added to it.
In addition, the model of the rat was imported from an external source, also in the public domain. In fact, because this rodent needed to be detailed, doing it manually would have taken too much time.

b. Animations

Character Animations
In order to bring the characters to life, the studio created two animations for them.
Firstly, the main animation is a walking phase, which is activated when the player is moving around the subway car. The major animated elements are the arms and the legs. To keep the low-poly style of the character, as well as its humorous style, the animations are very simple. Rotations along the x-axis in ***Unity*** are done to do so. The movements are coordinated between one leg and the opposite arm. In order to effectively represent the motion of the parts of the body while walking, the team of the **Head4che Production Company** observed repeatedly how the human body would normally work. Transposing the natural walk of a human to the digital version in ***SubWay Out*** was then much easier.
Secondly, an idle animation was conceptualized for the moments where the player is not moving. It is a simple standing position played in a loop.

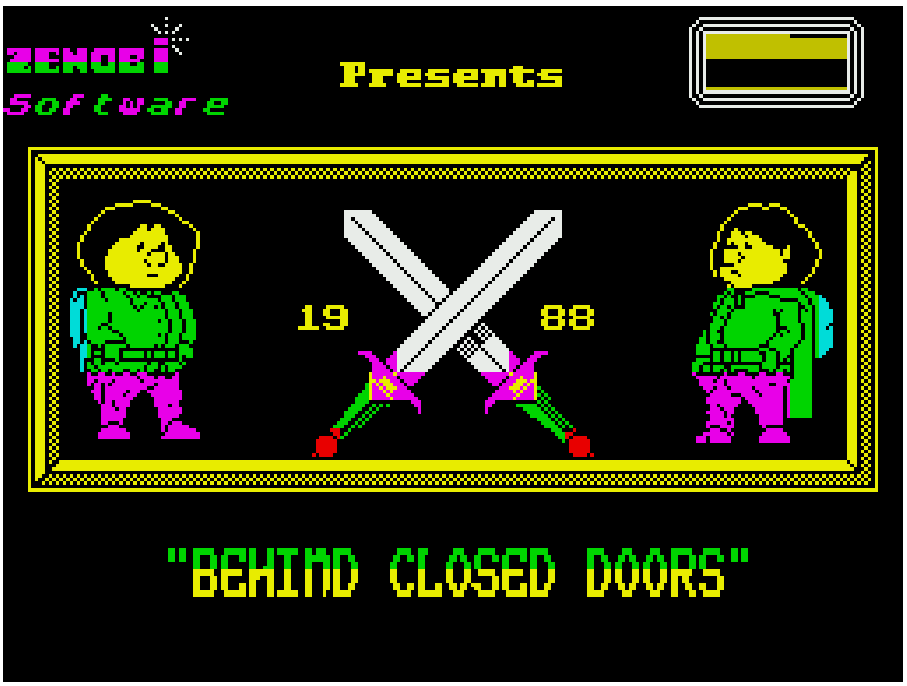


Figure 3.20: Animation

Tunnel Animations
In ***SubWay Out***, the player sees the train moving through the stations and tunnels. In fact, it is an illusion: only the outside of the train is moving, including the tunnel itself and the stations. This trick helps to avoid physics problems related to a moving train.

There are four animations related to the train and tunnel:

- Starting animation: The train leaves the current station and accelerates to its cruising speed.
- Moving animation: The train moves forward on the track at a constant speed.
- Stopping animation: The train slows down to stop.
- Stopped animation: The train stops at a station.

To simulate the train moving forward, the tunnel moves in the opposite direction and resets its position at the end of the cycle. This animation cycle makes the tunnel last as long as it takes for the player to solve all the puzzles and reach their destination.

Game Elements Animations
In ***SubWay Out***, several elements are animated.
These animations mainly concern boxes, windows, or doors that can be both opened and closed.
In addition, the seats have important animation because they form a puzzle themselves. In the same way as a real subway, they are folding seats that can be in an up or down position.
The rat is also animated. It has three different animations:

- Idle animation: the rat is moving its tail and head.
- Running animation: the rat is scurrying quickly.
- Walking animation: the rat is walking slowly.

c. Sounds

Sound plays a crucial role in enhancing player immersion. However, excessive audio can be counterproductive and hinder the immersive experience.
Therefore, only a limited amount of sound effects have been implemented: a clicking sound when entering a code to enhance the responsiveness of the action, and ambient subway noises alongside creaking chairs to further reinforce immersion.
The implementation of sound within the project is managed by a centralized `SoundManager` script, which handles the process of audio playback across the environment. When a sound needs to be played, the script is called by the relevant object. In response, the `SoundManager` creates a temporary `GameObject` at the position of the calling object. This new `GameObject` is assigned the appropriate `AudioSource` component along with the specified audio clip. Once the sound begins playing, the `GameObject` exists only for the duration of the playback and is automatically destroyed afterward. This approach ensures that sounds are played with spatial accuracy.
When handling sound, the `SoundManager` script sends an RPC to ensure that the sound is played for both players at the same time, regardless of whether it was triggered by the host or the client. This maintains a consistent audio experience across all instances of the game.

d. Visual effects

Lighting
Lighting has been developed to make ***SubWay Out*** realistic and to enhance the player's environment in the game. By adding lights inside the subway cars and on the walls of the tunnels, direct lights are created that affect the aesthetics of the cars.

SubWay Out uses *Global Illumination* algorithms to improve the quality and realism of lighting, providing ambient light and shadows that contribute to the game environment. These algorithms are used to manage indirect light reflected from surfaces in the scene by calculating the trajectory of light rays. The directional light maps can be found in the appendix B and C.
To make the game both fluid and beautiful, the brightness of static surfaces is pre-calculated using baked lightmaps. These are used in addition to the realtime calculated lighting effects to reduce computational load and maintain a consistent minimum quality regardless of the hardware used. The indirect lightmaps can be found in appendix D.
When the train is moving, the light rays from the tunnel lights can be seen passing through the car windows. The edges of the windows are projected inside the car to enhance the sense of a moving train.
A black ambient fog was added to prevent the player from seeing far and to create the feeling of an endless tunnel. It also makes the environment slightly darker, creating a heavy atmosphere.

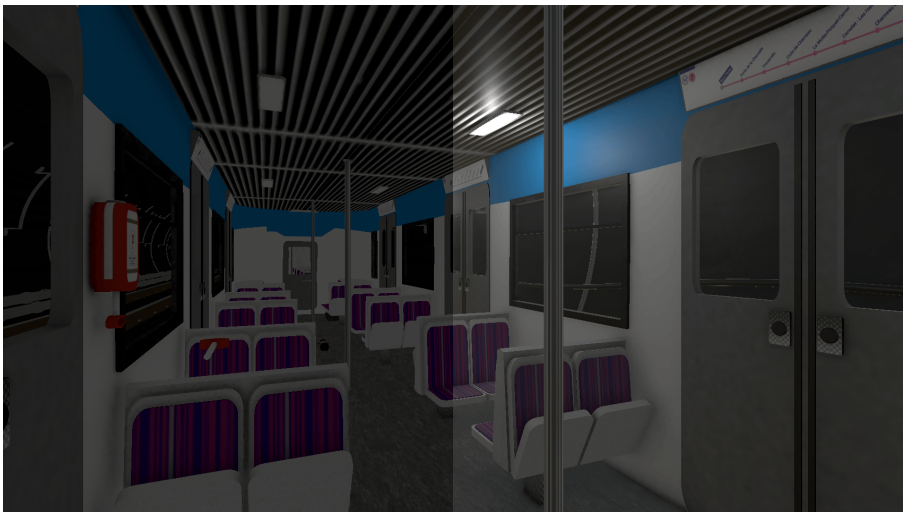


Figure 3.21: Comparaison: Before and After Lighting

Post-processing
In addition to lighting, post-processing was used to improve the overall graphical quality of the game and to enhance its emotional tone and atmosphere. These effects are a series of visual filters and adjustments applied to the rendered image just before it is displayed on the player's screen.
One of the key effects used was to replace neutral white colors with bright cold colors. This choice was made to emphasize a sense of isolation. The lighting and shadows' toning have also been adjusted to make the game more immersive and lifelike. By changing the color temperature and contrast of light sources and shadowed areas, the game atmosphere is more believable and emotionally resonant.
In addition, a film grain effect was applied to create a sense of scrambled thinking. This adds a texture to the image to reflect this slight instability. These post-processing choices are also intended to add a slight psychological depth to the game.

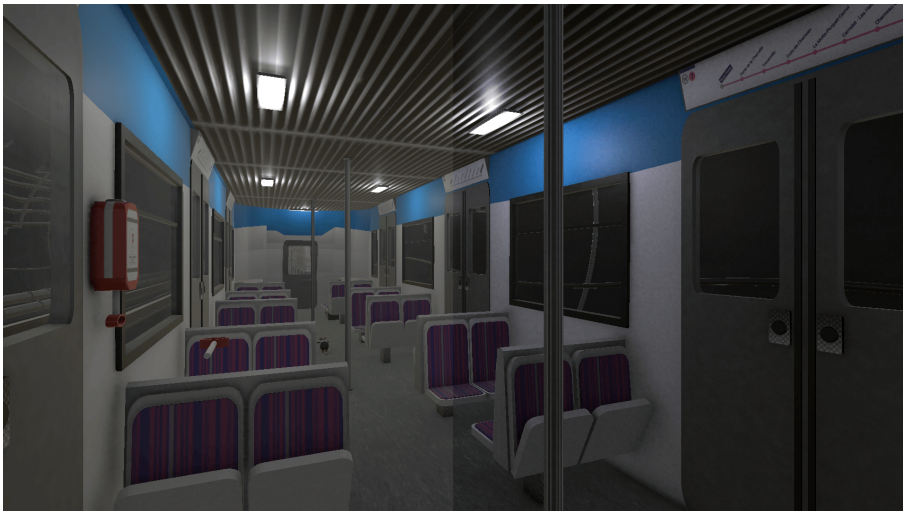


Figure 3.22: Comparaison: Before and After Post-process with lighting

4. Gameplay

a. Game Flow

Tutorial

In ***SubWay Out***, players are totally free in their actions. This is an important point, as in an escape game, figuring out what to do next is part of the game. This can cause issues, as new players might have no idea how to play – which is to say how to move, look around and interact with objects in their environment. However, a traditional step-by-step tutorial explicitly guiding the player does not fit an escape game, which by definition has the player stranded alone and unguided.

Because of this, it has been decided to implement a tutorial the player can choose to follow – or not – at any time throughout the entirety of ***SubWay Out***. This tutorial consists of three main parts.

At the beginning of the game, players have a little time to discover their surroundings: they are in a subway car, riding through a tunnel. After a few seconds, the train starts to slow down. This introduction allows players to discover the very intuitive controls of the game, in order to move and to look around.

After this, the train finally stops at a subway station. Upon arriving at a station, the player's first instinct is to exit the subway. The player will discover how to interact with an object, while trying to get out of the train by interacting with the button to open the door. However, the doors are stuck, meaning it is impossible to escape the subway for the moment. Instead, the subway restarts, the player still in it, and the trigger of the emergency call booth gets highlighted for the player to see. Interestingly, this trigger is not in the system, but is lying on the ground. The player can now learn how to grab an object and combine it with another, helped by on-screen hints.

After this, the player knows all useful commands for the game, and is totally autonomous to escape the train.

As mentioned before, nothing forces players to use the door buttons to restart the subway. Because of this, two fallback mechanisms have been established. Firstly, if the player assembles the emergency call booth system, they must have both grabbed an object and interacted with it, so they have the necessary knowledge to play autonomously. Secondly, if they manage to open the cabin door, they are very advanced in the game and will not need any more help. In both cases, the train will leave the station.

This tutorial has been designed to help the player learn how to interact with the game, while not being

too invasive. This is important for the game experience, as something more common would most likely break the immersion. Having this immersion and freedom to decide what to do as a player is a major concern in the experience offered by ***SubWay Out***.
This way of teaching how to play is therefore the most suitable one for ***SubWay Out***.

Non-linearity
The game aims to be as faithful as possible to the authentic escape room experience. An essential part of this experience is the freedom for the player to solve the puzzles at their own rhythm. Each player has a unique way to think and interpret their surroundings, and therefore they may need a different amount of time to solve the same puzzles. This needs to be taken into account because being stuck on a puzzle creates a frustrating experience for the player. The way escape rooms solve this issue is by having multiple puzzles the player can solve at a time. The player will first solve puzzles that seem easier to them, giving them time to think about the harder puzzles in the back of their mind. It also ensures they regularly have a feeling of accomplishment when they solve other puzzles than the one they are stuck on.
This is mirrored in ***SubWay Out***, where the player is offered three different puzzle lines to solve in whichever order they wish. This ensures the player always has something to do if they are stuck on a puzzle, and allows the player to have a less frustrating and more organic gameplay experience. All the paths then converge towards an ultimate single-step puzzle.

Hints
Another incredibly important feature to ensure that the player does not stay stuck on a puzzle are hints. These hints are optional snippets of the solution meant to guide the player's reasoning in the correct direction. In ***SubWay Out***, these hints take the form of audio messages that the player can request as soon as they finish the tutorial. In the game's story, the hints are given through the emergency call booth by an assistant of the company that operates the subway network. This helps reinforce immersion for the player by providing an interface coherent with the game's setting.
For a hint to be relevant, it must help with the puzzles the player is currently solving. As ***SubWay Out*** is fundamentally non-linear, this means the game needs to keep track of where the player is at in the game, regardless of the order in which they decide to solve the game. This is done by updating a list of available hints and synchronizing it over the Network. When the player requests a hint, a random one from the list is played for both players to hear. The players can hear the same hint multiple times if they need to, in order to ensure they do not permanently lose information if they miss a part of the hint when they first hear it.

Ending
After solving all the puzzles in the passenger part of the subway, players gain access to the driver's cabin. Opening the door of this cabin unlocks the final part of the game. This is one of the last steps players make to gain their freedom, to leave the subway car, and finally to escape this enclosed and oppressive space.
Once this door is opened, players have access to the final puzzle, the Hanoi Towers, that is described further in the report. This puzzle has been chosen to be the final one because it is a classical problem in computer science, mathematics and algorithmics. This symbolic problem was chosen as the final step towards freedom as it expresses one of the common interests of all members of the **Head4che Production Company**: computer science. This common interest is the one that caused the formation of the studio in the first place. This puzzle represents the core foundation of the company. This is

why it has been chosen as the final impression left to the player at the end of the game.
Once this last puzzle is solved, the train slows down, stops at a subway station and the doors finally open. The player is then free to leave the train, triggering the end of the game. Players are then redirected to the end game menu, where they are congratulated for their performance, and can see a timer showing how fast they have escaped the subway.

Figure 4.23: Player behind open doors

b. Puzzles

Rat Trap
The Rat Trap puzzle has one main objective: finding a green code. Essentially, the players have to trap a rat in a cage thanks to a cheese found in the subway car. The rat is holding a key in its mouth. By trapping the rat, it will release the precious key, essential for obtaining the code.
To be more precise, this puzzle can be decomposed into three different stages.
The first one consists in understanding the environment of the puzzle and the distinctive behaviors of the rat. The rondent is none other than the AI of ***SubWay Out***. When the players are too close to the rat, it flees them. When the distance between them is important enough, the rat is at rest. It wanders in the subway car towards random positions. When the cheese is in the cage, and the door of the cage is opened, the rat is attracted to it. Thus, understanding these different behaviors is the first necessary step to complete this puzzle.

Moreover, understanding that the rat must be trapped in the cat cage present in the subway car is mandatory. Interacting with the environment to discover where the cheese is located is also a part of this puzzle. As a matter of fact, the cheese is located in a backpack forgotten by a passenger. After being found, the cheese can be positioned in the cat cage in two different ways. The first one is by simply interacting with the cage while grabbing the cheese and the second one is to slide the grabbed cheese in it. Once the cheese is located in the cage, it is not grabbable anymore.

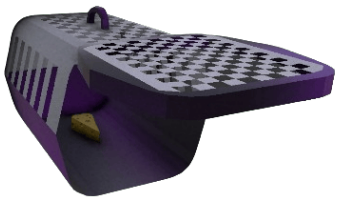


Figure 4.24: Cage and cheese

As indicated by the name of this puzzle, the second stage of the Rat Trap puzzle is to trap the rat. However, as mentioned above, the rat is lured in the cage only if the cheese is placed inside. Thus, the first step is required for the rest of this puzzle. Once the cheese is in the cage, the players need

to keep the door of the cage open, otherwise the rat will not enter it. If both of these conditions are met, then the rat is trapped in the cat cage. The cage is after this not interactable anymore, meaning that the door will stay closed. When entering the cage, the rat releases the key that it was holding in its mouth.

Figure 4.25: Green code

Finally, the last part of the puzzle is finding the green code. With the key in their possession, the players have to find a lock in which they can insert it. The only non-electronic element of the subway car is the cheese advertisement. The cheese design makes a connection between the cheese found at the beginning and the location where the key can be placed. By clicking on the advertisement, the glass pane will open and reveal the green code 58 on the border of the frame. This code is essential to open the driver's cabin door.

Folding Seats

To make the players pay more attention to their environment, the Folding Seats puzzle is based on observations and associations of ideas made by the players. At first, in the subway car, the players have to spot a technical supplies box that is located below the rear seats. After pulling the box from under the seats, a sticker glued at the top of the technical supplies box is made visible. However, it is torn into two, and one part is obviously missing, as glue marks are visible on the box. Their mission is thus to find the lost fragment in the car. After succeeding to do so, the players need to combine the two parts by interacting with the sticker in hand on the box. This will make the complete design of the sticker visible.

Figure 4.26: Sticker torn and restored

The sticker represents the subway car in which players are trapped in. Understanding that the shapes represented on this drawing are the seats of the subway car is required to proceed to the following part of the puzzle. Some cubes are colored in white while others are colored in blue. Thus, a distinction between the seats in an up position and the seats in a down position must be made.

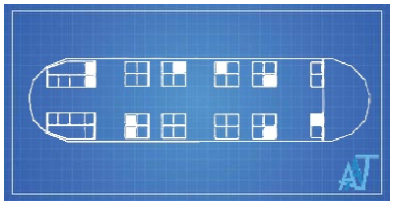


Figure 4.27: Sticker

As shown on this design of the sticker, players need to identify that all the seats which can not be folded are in blue, and that all the blue folding seats must be opened. In contradiction with that, the white squares represent seats that must be placed in an upward position.

After correctly placing all the seats in their designated positions, the technical supplies box will open automatically, revealing both the blue code 70 and an airflow instruction. The blue code discovered is required to find the correct code to enter the driver's cabin.

Figure 4.28: Blue code

Air Vent

On top of the door to the driver's cabin is an air vent linked to the car's ventilation system. This vent is closed, so the player can not see behind the vent's flaps.
After solving the Folding Seats puzzle, the player can find a piece of paper detailing instructions on how to activate the air vent. It represents one of the car's windows being closed with an arrow pointing to a vent with air coming out of it.

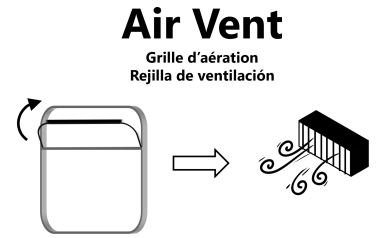


Figure 4.29: Ventilation system instructions

Figure 4.30: Red code

The player must follow these instructions and close all windows. If they do so, the air vent opens. This allows the player to see a red 110 behind the flaps, which will be useful for the cabin door's code.

Door Code

At the front of the train is a door, secured by a code lock. At first, players have none of the four elements needed to find this code. However, all along the game, when puzzles are solved, colored numbers are revealed.

When all these numbers are found, the player doesn't know what to do with them. In order to combine them, a fourth and last hint must be found by carefully inspecting the player's environment. This hint is located in the subway car behind the one the player is in. By looking through the rear window, the player can see on one of the walls of the car an instruction: "B × R - G". The instruction is localized in accordance with the local player's language.

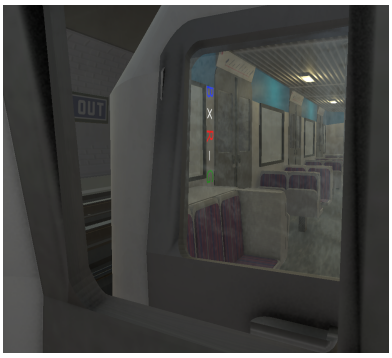


Figure 4.31: Instruction on the wall of the other car

Figure 4.32: Instruction on the wall of the other car

This instruction shows how to combine the previous clues. Indeed, by multiplying the numbers in the blue and red code, and then by subtracting the one in the green code, the player can obtain a 4-digit code, 7642, which will successfully unlock the door's mechanism.

Hanoi Towers

Once the player enters the driver's cabin located behind the door, they have access to the game's final puzzle: the train's controls. This puzzle is actually a disguised version of the famous Hanoi Towers puzzle.

The Hanoi Towers puzzle is the system that underwent the longest development cycle. It has seen four complete overhauls until reaching a fully satisfactory state. The three first versions of the system have been covered in past reports. The fourth and final version was completed shortly after the second defense, and considerably improved the game's logic.

The puzzle is inspired from the eponymous mathematical game. Its premise is the following: the player has in front of them three axes. On the left axis are three objects of different sizes. The player must move these objects to the right axis respecting three rules:

- Only one piece can be moved at a time, from an axis to another.
- Pieces can only be removed and inserted from the top of the axes.
- A piece can not fit on top of a smaller piece.

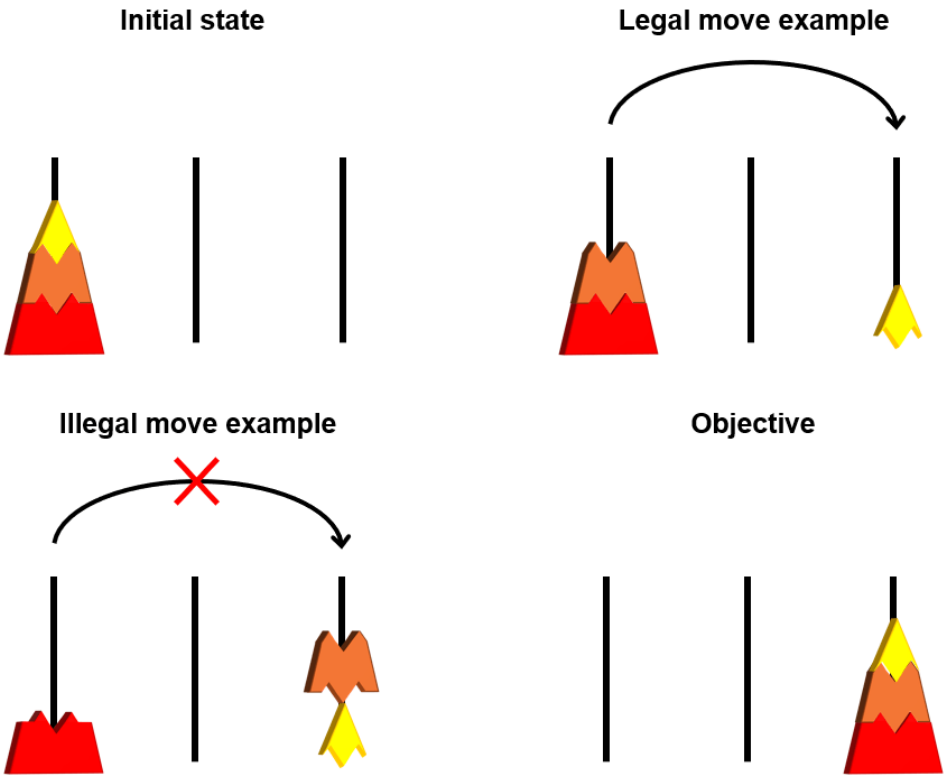


Figure 4.33: Sticker thorn and restored

The main challenge when implementing the system was ensuring the pieces would move smoothly along the tracks when grabbed. This meant overriding the regular grabbing behavior in order to lock the movement to a plane and precisely control the piece's position.
The logic for the fourth iteration is separated in three parts. It is repeated until the player lets go of the piece.

Firstly, a ray is cast from the player's camera. It records the position X at which it hit the game plane $\mathcal{P}$. It generates the vector $\vec{u}$ representing the transformation from the piece's current position in global space to the position the player is aiming at. The transformation is aligned with the plane. $\vec{u}$ is then transformed in local coordinates relative to $\mathcal{P}$: its vertical component is zero, and its vertical and horizontal components correspond to the vertical and horizontal directions on the plane.

Secondly, these coordinates are analyzed by the system in order to guess the player's intent. If $\vec{u}$ is of very small magnitude, any movement is negligible: no movement is applied. If the vertical component is more than double that of the horizontal component, the player is likely trying to move the piece up or down a vertical axis: the horizontal movement is suppressed for smoother movement. Oppositely, if the horizontal component is considerably larger than the vertical component, it is the vertical movement that is suppressed. Otherwise, $\vec{u}$ is not modified.
Finally, $\vec{u}$ is transferred back to global coordinates. It is normalized in order to get constant movement speed, and is applied as a force to the grabbed piece.

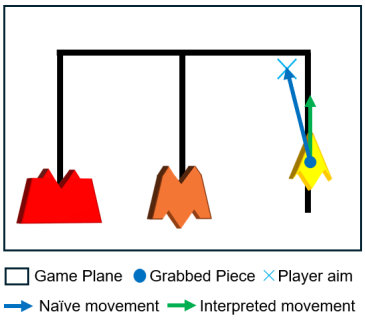


Figure 4.34: Instruction on the wall of the other car

This system ensures the piece goes smoothly towards where the player wants while accounting for the inaccuracy inherent to mouse controls.

c. AI

The Artificial Intelligence (AI) is a major element of SubWay Out, as it constitutes in itself the Rat Trap puzzle. It is an escaping and pathfinding AI, as the rat can both flee when a player gets too close to it and find a way to reach the cheese. A navmesh defines an area on which the AI can navigate. As mentioned in the Puzzles section, the AI has three different behaviors depending on three different situations. It is represented by an enum with three constants:

- Idle
- Fleeing
- Baited

Having comprehensible detailed states was helpful to clearly analyse in which phase the AI was. The three different behaviors are explained below.

First, for the idle state, the AI needs to wander at random positions then wait a few seconds before repeatedly doing that. The condition in order to consider the AI in the idle state is the distance between the rat and the nearest player. This means that the AI must detect the nearest player. By obtaining the list of all players, then computing the distance between each player and the rat, getting the nearest player is straightforward. If the players are far enough, then the rat stays in an idle state. A random position is computed. If this position is located on the navmesh then it is considered valid and the rat can go towards it. Otherwise, the AI continues to search for a valid position. Thanks to a coroutine, it is possible to make the rat wait a few seconds before searching for a new random position to go to.

The second state of the AI is the fleeing one. This state is activated when the nearest player is too close to the rat. The rat must flee in the direction opposite to the player. An angle is also added to this new random position, to avoid the rat fleeing in a straight line only.

Finally, for the baited state, the AI is attracted by the cheese which was put in a cage by the players. Three conditions must be met for the AI to enter this state: the nearest player must be far enough away from the rat, the cheese must have been placed in the cat cage and the door of the cat cage must be open. If all of these conditions are respected, then the AI will move towards the cheese. Once the AI is close enough to the cheese, it releases the key. The AI is deactivated by spawning the model of the rat only but not its AI components. This is done to avoid unintended situations. However, if the cage is not at the same level as the rat or the door of the cage is not opened, the rondent can not be close enough to the cheese to enter the cage. In this case, it will wait until the cheese is at the same height or until the door is opened by one player.

d. Multiplayer

A ***SubWay Out*** session cannot be joined after it has started. The game's state is stored across multiple `GameObjects` that each store their local state. This makes it impossible to synchronize all the states when a new player joins. Nevertheless, this also means there is no need to centralize all the data about the game on the network in order to implement a multiplayer system. Each object manages by itself the little data that needs to be synchronized. Most of the time, using RPCs whenever a change occurs was the solution chosen, but for some more complex cases, `NetworkVariables` are also used to ease the implementation and to ensure the compatibility of all systems in a two-player game.
In terms of user interface, the multiplayer system encountered several iterations. However, it always kept the same key elements: a field to enter the join code, a text displaying this code and buttons to join or start a game.

The picture aside shows what the first version looked like. After this, in order to offer better visuals, a simpler interface and overall a better user experience, it has been decided to split this page into two simpler ones: a page to start a game as the host, and another one to join a game as a client this time.

Figure 4.35: First multiplayer menu

Figure 4.36: Final version of the multiplayer menus: start and join

e. Menus

Home Menu
The main page redirects players towards different menus depending on what they want to do. Their options are:

- Starting a game
- Joining a game
- Selecting the character they play with
- Going to the game settings
- Viewing the credits
- Quitting the game

A visually-appealing interface has been implemented in order to give players a positive experience using ***SubWay Out***. In order to do so, research has been conducted about what was commonly done in the video game industry, about the common points between video games known for their good interface. These common points have then been implemented in the menus. For instance, the menu is always interactive, with elaborate dynamic indicators showing which part of the page you are hovering.

Skin Selector
A page is dedicated to skin selection. As previously mentioned, two models can be used in the game, representing members of the team. This menu allows the players to choose who they want to be for their escape game adventure. By clicking on the left or right arrows, the users can navigate between the different models in an appealing way. The skins are synchronised over the network. This means that if one player selects a particular skin, then the other player will see the skin selected. Once the players have finished selecting their skin, they can go back to the main menu to begin ***SubWay Out***.

Settings Page

The settings page is used to modify the behavior, the appearance and the settings of the game has been implemented. This page is important, because it allows every player to have the best experience possible, thanks to a game adapted to their personal preferences. The different settings implemented are:

- The language of the game
- The camera sensitivity
- The sound intensity
- The brightness of the game
- The display of the controls

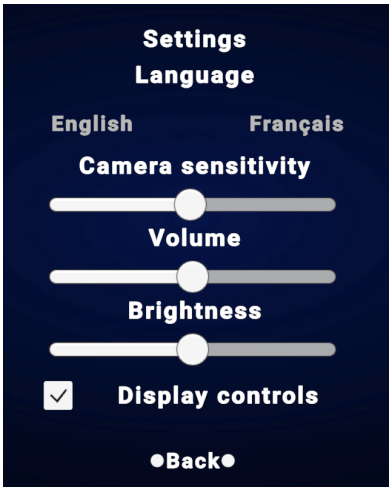


Figure 4.37: The settings page

Translation System

It is important for us that ***SubWay Out*** is accessible to as wide a public as possible. That is why English was originally chosen as the game language. However, after some reflection, the team realized that translating the game would make it even more accessible worldwide. This idea was implemented with an easy way to add supported languages to the game. Moreover, the software automatically detects where the player is from in order to select the more appropriate language. A translation system was created in order to be able to change the language between English and French. All the texts and buttons need to change depending on the choice of language. To do that, a table with the different translations was made. The ***Unity Localization*** package assigns and displays the corresponding text resource depending on the option selected at the beginning of the game. As the text contained in a button changes, it is necessary to adjust the button width. Furthermore, this translation system does not only handle visual texts and buttons but also voice lines used in the hint system. The audios for the hint system were recorded in both languages.

Camera Sensitivity

Each player has their own preference when it comes to mouse usage. In order to offer an experience that fits all users, it is important to allow one to change the sensitivity of the camera. Without this setting, the player can experience difficulties moving the camera efficiently in order to look around. Creating this option is a way to ensure players enjoy a comfortable experience while playing ***SubWay Out***.

Sound Intensity

The game offers an audio experience. However, depending on each individual, sounds can be either uncomfortable because too loud or inaudible because too quiet. In order to adapt the loudness of the sounds to each person, this setting was created.

Brightness Setting

Some players may have difficulty seeing some valuable items needed to complete the game. These difficulties are due to the low brightness of the game environment. To address this issue, a brightness control was created to allow players to adjust the brightness of the game using a slider on a settings page.

Show Visual Controls

Visual hints about how to interact with the game, for instance how to grab an object or how to zoom, are enabled by default. In order to allow more experienced players to have a more immersive experience, anyone can disable these visual hints. The game is indeed more realistic without the commands on the corner of the screen. The development of this option takes place in the efforts made to offer an experience as near to real life as possible.

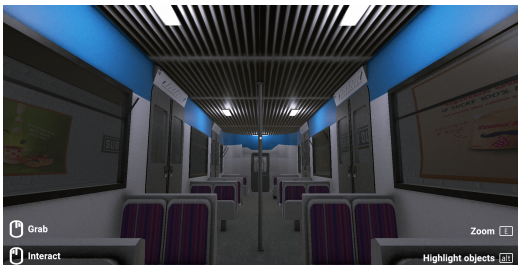


Figure 4.38: Game window with visual controls indicators

Pause Menu

A player might want to take a break during the game, therefore a pause menu has been implemented. It allows, at any time during a session, to pause the game and restart it afterwards, or to leave the session without entirely finishing the game.

Figure 4.39: The pause menu

5. Website

Having a website is mandatory for any video game, and ***SubWay Out*** is no exception. In order to communicate about the game, to allow players to download it and to give potential new players the drive to try the game, having a good website is a necessity. In order to make the best website possible, the **Head4che Production Company** went through two main steps: first, the conception and design of this website, and then its implementation.

a. Design

As a game, the ***SubWay Out*** website is the first real contact visitors have with the game they will be playing. This showcase has been designed to directly communicate the theme of the game with a color palette similar to that of **RATP**.
The design of the ***SubWay Out*** website is composed of three sections:

- **Hero** section: The first section visitors will see, it contains the download button for the game and an animated background with in-game captures.
- **News** section: To keep visitors informed of the latest news about the development of the game, with articles and an embed of the **Head4che Production Company**'s ***Instagram*** page.
- **Documents** section: Where visitors can find all technical documents related to the active development of the game and its specifications.

The layout of this site is mainly inspired by the websites of other games such as ***Genshin Impact***, ***Honkai: Star Rail*** (**Hoyoverse**), and ***VALORANT*** (**Riot Games**). The concept art of the ***SubWay Out*** website can be found as the appendix E.

b. Implementation

To implement the website, different tools were used. First, the JavaScript library React was chosen to develop the website source code that needed to be compiled into a static version. This static version is built by ***GitHub Actions***, and once ready, is automatically deployed online. This website is hosted by ***GitHub Pages***, and is available at the address head4che.co.
Initially, the website was developed to look like its concept art, but as new needs emerged, the original plan has been adapted. For instance, a page containing more details about the project has been added and a part with the trailer has been incorporated in ***SubWay Out***'s page. An image of the additional page can be found in the appendix F.

V. Recap

1. Task distribution

At the start of the project, a provisional task distribution was shown in the Technical Book of Specifications (TBoS). The distribution was always meant only as a general guideline and never as a strict task assignment, especially since the categories were imposed and did not fit the project's specific flow. Since the beginning, the team's objective was to create an organic workflow where each member would choose their tasks as soon as they finished the previous one. This was made possible by the MNMs detailed in section B.1.
This means the actual task distribution varies greatly from what was shown in the initial distribution. The more flexible approach taken by the studio enabled each member to test various types of tasks throughout the project, and ultimately focus on those they preferred rather than what was arbitrarily decided by the team at the time when it had no knowledge in game development whatsoever.
An example of this is the "Multiplayer" task imposed for the TBoS. Since the beginning of the project, the studio has kept a clear stance on the implementation of multiplayer gameplay. The company estimated that having a dedicated group of people going through each of the game's systems to implement multiplayer compatibility would be both impractical and inefficient. This would mean effectively writing all of the game's mechanics twice: once for the single player experience, and a second time for multiplayer. The studio assumed it would be much much more efficient, at least in the context of this game, to have every developer write their respective systems directly with multiplayer compatibility in mind. This assumption turned out to be correct. The time gained by shipping the features already functional on the network largely exceeds the time that had to be spent by each developer learning to use ***Netcode for GameObjects***.
The flexible approach the studio took in regards to task distribution let each member explore all aspects of the game development during the first months of development. Once every member had identified their strengths and preferences for the project, they naturally drifted towards tasks that fit them better. Some spent time reviewing other people's code and optimizing existing systems. Others preferred developing the graphical aspect of the game and the player experience.
In the end, each part of the project has been developed by someone naturally interested in that specific part, resulting in a game where each element has been crafted out of interest and passion.

2. Effective distribution

TASKS	Jeanne	Tidiane	Arthur	Adam	Valentin
GAME ENGINE					
Player & Camera		●	❍	●	❍
Grab & Interact				●	●
Networking			●		
AI	●			❍	
Hints system	❍			●	
Tutorial & Endgame			●		❍
PUZZLES					
Leve design	●			●	
Rat Trap	●		❍		
Folding Seats	●			❍	
Air Vent				●	
Door Code			●		
Hanoi				●	
GRAPHICS & SOUND					
3D Modeling	●				
Animations	●				●
Sounds		●	●		
Menus	❍		●		
Visuals effects					●
COMMUNICATION					
Branding					●
Website			●		●
Instagram	●		❍		
Trailer	●				

Table V..1: Summary of the effective task distribution

● : Main developer / ❍ : Helper

3. Means

a. Organization

A major realization during this project was the importance of organisation in order to work efficiently, especially when working in a group. Indeed, as the **Head4che Production Company** became more and more organized throughout the year, a significant increase in productivity has been noticed. This organization went through several steps. Right from the start of the project, a ***Discord*** server was created in order to communicate between members of the project.
After some brainstorming, it was decided that each member would assess their progress weekly, as well as decide and communicate about their next task. This was very important, because this step occurred at the start of the development of ***SubWay Out***, at a time where the team was developing the core of the game, and therefore the tasks were very interdependent: it often happened that a member of the group had to wait for the completion of the task of another member to start their own task. Communicating clearly about what was being worked on, what should be done in priority and the delays in which features were going to be ready was a major improvement. Knowing what is done was also much more motivating than having the feeling to work alone because of a lack of communication between project members. Fortunately, this decision was taken relatively quickly.
This weekly habit has been anchored by the creation of what has afterwards been colloquially called the “MNMs”, for “Monday Noon Meetings”. During these meetings, team members would assign themselves tasks for the upcoming weeks. ***GitHub Project*** has been a helpful tool used in these weekly meetings. Indeed, **GitHub** has a tool to manage tasks. Each one of the tasks were referenced on this tool. During the MNMs, tasks were assigned to members of the team, and a state was given to each task: “To Do”, “Paused”, “In Progress”, “To be enhanced” or “Done”.
The joint efforts of the weekly meetings and the use of ***GitHub Projects*** allowed the **Head4che Production Company** to keep track of each member’s progress. A positive side effect of this is the fact that when a member was struggling on a task, others would notice and usually somebody would help to finish developing the feature in time.
This organization continued to be used until the end of the project. Even when the Monday meetings could not be held in person, they were conducted online.
Working together online was so efficient that a virtual open space was created by the **Head4che Production Company**. This space was used daily for work. This online workplace was such an effective tool for working, focusing on tasks, and overall for being productive that it was not only used by the team members. Members of other groups regularly joined this space to talk, seek help or simply work on their own video games.
This open space has been a place to work peacefully, in a good mood at all times of the day and of the night. Working with others has been a relief and an important help to find motivation and discipline to work on the projects, even when being overwhelmed by tasks to achieve – or even worse, bugs.

b. Tools Used

The tools chosen to work are an important factor of the work efficiency and outcome. Hence, keeping track of which ones are used is important to know how work is done, and therefore to have a chance to improve. Thus, here is the list of the tools used during this project:

- ***Unity*** and its libraries
  Unity has been the game engine used to create ***SubWay Out***. Among the main libraries used with it are ***Unity Services*** (containing ***Unity Lobby*** and ***Relay Servers***), ***Netcode for GameObjects*** (for the multiplayer system) and the ***Localization*** package.
- ***Rider***
  Rider is the IDE (Integrated Development Environment) used to code in C#, this language being the one used to develop ***SubWay Out***
- ***Blender***
  Blender is the open-source 3D modeling software used to create 3D models for the game.
- ***GitHub*** and its services
  GitHub is the DVCS (Distributed Version Control System) of the **Head4che Production Company**, as well as a way to collaborate and work on differents task at the same time on different devices.
  - ***GitHub Project***
    GitHub Project is the tool used to manage the achievement of the tasks.
  - ***GitHub Actions***
    GitHub Actions is the tool used to automatically build the website and to run automated unit tests.
  - ***GitHub Pages***
    GitHub Pages is the service used to host the static version of the website
- ***Discord***
  The social media used to communicate between the members of the **Head4che Production Company**.
- ***Paint***, ***Inkskape***, ***draw.io***, ***GIMP***, ***Canva*** and ***PhotoFiltre 7***
  Various tools used to create visuals used during the project
- ***PowerPoint***
  PowerPoint is the software used to create presentations for defenses related to the **Head4che Production Company** as well as internal documents
- ***Figma***
  Figma is the online service used to design the website head4che.co
- ***Clipchamp***
  The software used to create the trailer of ***SubWay Out***.
- ***Instagram***
  Instagram is the main social media used to communicate about the **Head4che Production Company** and the game ***SubWay Out***.
- ***Visual Studio Code***
  Commonly called “VS Code”, this IDE has been used to code in other languages than C#.

- ***Google Docs***
  Google Docs is the online documents editor used to write a first version of the documents of the **Head4che Production Company**, before writing them with LaTeX.

- ***InnoSetup***
  The software used to create the installer of the game.

Several computer languages have been used throughout the project: programming languages, libraries, description and markup languages. The main ones are:

- C#
  The language used to create SubWay Out.

- JavaScript, more precisely the library ReactJS
  This library has been used to develop the website.

- Sass
  The SCSS syntax has been used to develop the style of the website, in a more efficient way than just with CSS.

- DOT
  The description language used to create graphs in order to plan work and gameflows.

- LaTeX
  LaTeX is the markup language used to create documents for the **Head4che Production Company**.

4. Challenges Encountered

This section concentrates on the challenges faced during the construction of ***SubWay Out***. From the very beginning of the project to its end, the **Head4che Production Company** needed to face difficulties, resolve issues and overcome obstacles. Thankfully, as the group is united, with the motivation and cohesion of everyone, finding alternative solutions was successful. Even if some problems could not be solved during this project, the team was determined to explore other possibilities. Below is the description of the obstacles encountered, how the studio succeeded to overcome them, or if not how it found alternatives.

a. 3D Models

During the realization of the three-dimensional elements that constitute ***SubWay Out***, several challenges were encountered.
First of all, it was the first time that the team had the opportunity to create 3D models with ***Blender***, so learning how to use this open-source creation software was difficult at first. Hopefully, thanks to a lot of videos and online tutorials, the studio was able to understand how the modeling toolset works and fully get started with ***Blender***.
In addition, the implementation of many models made in ***Blender*** into ***Unity*** was a major issue. Even if the skeletons of the structure were correctly implemented, the textures were not applied well, ruining all the work done. It was a total disaster in terms of visuals and colours. Nevertheless by regularly practising the manipulation of 3D models on ***Blender***, the members of the team found a button during the exportation process to embed the textures. This was the solution to correctly perform the implementation of the models in ***Unity***.

Furthermore, for some models, some faces were not correctly displayed and thus some meshes were invisible. This was a backface culling issue. Indeed, in ***Blender***, the open-source 3D graphics software that is used for modelization, the normals of the faces can be calculated from the outside or the inside, and are then either rendered or ignored. For example, for the trigger of the hint system, some normals were calculated from the inside and, as a consequence, not shown when imported into Unity. Therefore, recalculating the normals outside for the faces that were missing was the solution to the issue and then all the faces were visible.
Moreover, the models of the Hanoi Towers puzzle were, as previously mentioned, very challenging. For instance, for the three different pieces of the puzzle, the main problem was the mesh colliders. Indeed, they were not applying well due to the different convex meshes of the pieces. In the following development images, the green lines represent the shape of the mesh collider for the orange piece: the collider does not take the shape of the piece. As a consequence, when the pieces do not fit in the puzzle. This issue was fixed by cutting the pieces into subparts, allowing the collider to take the proper shape of the orange piece.

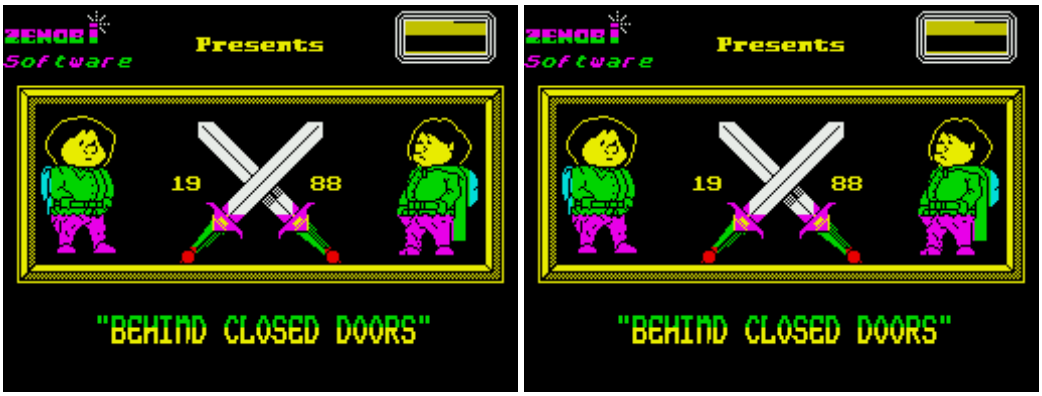
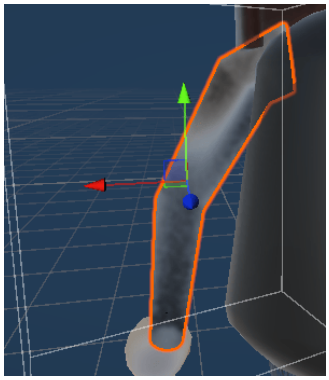


Figure 4.1: Corrected collider

Last but not least, another problem that concerned the vast majority of the game elements modeled for ***SubWay Out*** was the limit of authorized polygons in ***Unity***. A convex mesh must have at most 256 polygons. Although some elements were small such as the cheese, they were too high-poly, and thus generated warnings while imported into ***Unity***. To drastically reduce the number of polygons in ***Blender***, the team used a tool called the “Decimate Modifier”. This modifier allows the reduction of the face counts of a mesh without changing its minimal shape. Thus, the number of polygons was successfully reduced.

b. Animations

Making the animation of the players was challenging for the studio. In ***Blender***, a tool to create animations is present. A rig must be attached to the model of the character. A rig is a sort of skeleton, composed of bones. Attaching the model to it is complicated because each bone must be associated with one part of the model. This association is made thanks to weight painting. When a bone is selected in ***Blender***, colors appear on the model. The redder it is, the more attached the bone is for this part of the model, whereas if it tends to the blue, the bone is not very attached to this part of the model. Thus, attaching the rig to the model is difficult as the way weight painting is generated is a little bit randomized. While attempting to attach the rig to the model, parts of the model that the studio did not want to animate were moving, creating a strange walking animation.



Even after succeeding to properly attach the rig to the model, the model was not displayed correctly in Unity. At the same time, it was functioning as intended in Blender, with a proper rendering. Knowing where this error occurred was challenging, the team scoured tutorials on the internet to repair the broken model but unfortunately no solution was found.
Therefore, to solve this problem, the studio decided to do all the animations of the character directly in Unity. Even if the animations are not as detailed as the ones created in Blender, they work correctly, and the team is satisfied with the outcome.

Another issue that occurred was the synchronization of the character's animations over the network. When the player was walking, the animation did not play for the other player. To overcome this situation, the group used RPCs and managed to display the animations correctly across all clients.

c. Game Engine

A lot of time has been spent at the beginning of the year to create abstract mechanisms that were going to be used all along the development process. For instance, developing a functional player, creating abstract classes to grab and interact with objects and creating the assets of the game all had to be done in order to be able to create the core content of the game, which is to say puzzles.
Implementing the grabbing system was a real challenge because it is the only way for the player to move an object in its environment. The physics of the game and the multiplayer aspect had to be taken into account to make a good grabbing system.
The synchronization of grabbable objects positions over the network was the major issue encountered. The game uses a client-server topology, which means that the server has ultimate authority over most objects in the game. In order to avoid position desynchronization, all movements are computed on the server, and the new positions are read by the clients.
Time had been reserved for these tasks, but working together on such a restricted amount of interdependent tasks was a challenge, and more time was spent than what had been planned.

d. AI

Concerning the AI, the only real difficulty encountered was the delimitation of the navmesh area. At first, the navmesh was baked automatically by ***Unity***. However, the navmesh generated was not corresponding to the needs of the team. Some positions outside the subway car were taken into account in the navmesh. This could have resulted in a total disaster, as the rat could have been able to leave the subway car and never come back. The studio had to create the navmesh area manually.

e. Networking

The first version of the networking was really hard to implement, and the connection obtained was really unstable. Connecting a game to another was a really difficult thing to achieve as an user, and the journey to set this up was very complex.
The system managing the networking was badly organized. All functionalities were entangled, and calling a function not in the exact perfect context it has been conceived for would result in a plethora of errors.

Rewriting it has been a difficult decision to make, but also a very good choice, as it created a new strong and clean basis to work on. Having the experience of previously developing a network system granted the team a good and clear vision about what had to be done, and allowed it to be efficient to implement the multiplayer component of the game afterwards.

f. Hanoi Towers

The development of the Hanoi Towers puzzle spanned between December 2024 and March 2025. Countless hours of work were poured into this single system that only creates a few minutes of player experience.
In retrospect, starting development with such an elaborate system was an unwise decision, as the system could have been considerably faster to develop after having acquired experience from previous easier systems. Nevertheless, the team managed to put up to the challenge and create a robust system that remains the most advanced puzzle in the game.

g. 15 puzzle

Most of the puzzles planned during game design have made it into the final product. Nevertheless, there is one regrettable exception: the 15 puzzle. It was supposed to be another very localized puzzle where the player had to find a way to arrange pieces in a specific pattern, somewhat echoing the Hanoi Towers. It would have been the way to obtain the instructions on how to combine the three colored codes.
The system had to be scrapped due to its complexity. Like Hanoi, it required developing a custom grabbing behavior to have precise control over pieces sliding along a place. The studio estimated the time spent on that puzzle would be better put to use polishing off all other game mechanics.

h. Netcode for GameObjects

SubWay Out uses ***Netcode for GameObject*** (***NGO***) to handle the game's synchronization amongst clients. This system is simple as it is highly integrated into ***Unity*** through `NetworkObjects`, which are game elements synced on the Network. Nevertheless, as with any simple solutions, it comes with many limitations. The two biggest issues were the restrictions on Network Prefabs and nested `NetworkObjects`.
In ***NGO***, prefabs with elements synchronized on the Network can only have a single `NetworkObject` component that must be placed at the root of the prefab. This caused multiple issues, especially for grabbable objects. Most of them are prefabs made of multiple parts, each part with a separate collider. This means that different parts of the object could not be marked as grabbable, as grabbed objects have to be on the Network in order to sync their positions. To circumvent this, the grabbing system had to be reworked, and it is mainly for this reason the `ParentGrabbable` class detailed in the Game Engine section had to be developed.
Nested `NetworkObjects` designate `NetworkObjects` that have children objects, some of which are also NetworkObjects. These are not fully supported by ***NGO***, and using this structure can cause inconsistent crashes. Thankfully, the systems created to circumvent the first limitation also solve this issue.

5. Joys

a. Multiplayer Implementation

Before implementation actually started, making puzzles compatible with the multiplayer was perceived by the team as a task that was going to be really long and difficult. However, once the networking system was properly set up, it turned out not to be as difficult as anticipated. Moreover, quickly thinking about how to do it in multiplayer before creating the single player version was most of the time sufficient to reduce the changes to do, sometimes even to only a few simple lines.
This simplicity is due to RPCs. Once the concept was understood and the system around these Remote Procedure Calls was set up, everything became easier to implement.

b. Puzzle Implementation

As mentioned above, creating the game engine took a lot of time. However, this was a rather positive development. Even though the game engine was not simple to create, having this robust system allowed the **Head4che Production Company** to implement puzzles really fast. Creating this part of the game has been perceived by the team as a really fun and efficient task. Moreover, while creating puzzles, seeing the work done taking place in the game and becoming what had been designed through the whole year was very motivating.
All the team worked together and enjoyed creating puzzles for ***SubWay Out***.

c. Team Dynamics

The **Head4che Production Company**'s most valuable asset is its team dynamics. During the whole development process, the team members worked together flawlessly. No conflicts arose, and compromises were found whenever opinions diverged. Surpassing the working environment, the project helped develop strong relationships between the team members. The final product is the result of endless passionate hours of labor from the whole team, resulting in a game that would have been of significantly lesser quality, had a single member been missing or substituted.

VI. Conclusion

In conclusion, this project was an incredibly valuable experience for all the members of the **Head4che Production Company**. From not knowing each other at all to becoming a working group and more importantly a group of friends, ***SubWay Out*** has gathered **Head4che**'s five members around a common goal. Curiosity and motivation pushed the team to give their all for the project. At first, it was passion for puzzles that gathered the members of the studio together, but as the end of this project draws near, the team members agree to say that they are now bound by more than just a common interest. The group shared invaluable moments together, and is now full of joyful and pleasant memories. Working together is the common denominator in all these enjoyable memories. Under constraints, the team understood how helping each other out was crucial. This project is not only about coding together, it is about learning to collaborate, communicate, make decisions and develop technical and social skills together.

The **Head4che Production Company** is proud of the work that was done throughout the whole year. Each member of the group has progressed, both technically with better coding practices and knowledge, and personally, by being involved in such a rich team.

VII. Appendix

On the following pages are found the documents mentionned during the report. They are:

- Appendix A: Puzzle graph
- Appendix B: Directionally realtime global illumination debug draw
- Appendix C: Lighting receivers and Global illumination contributors
- Appendix D: Indirect realtime global illumination debug draw
- Appendix E: Concept art of the website
- Appendix F: Project page of the website

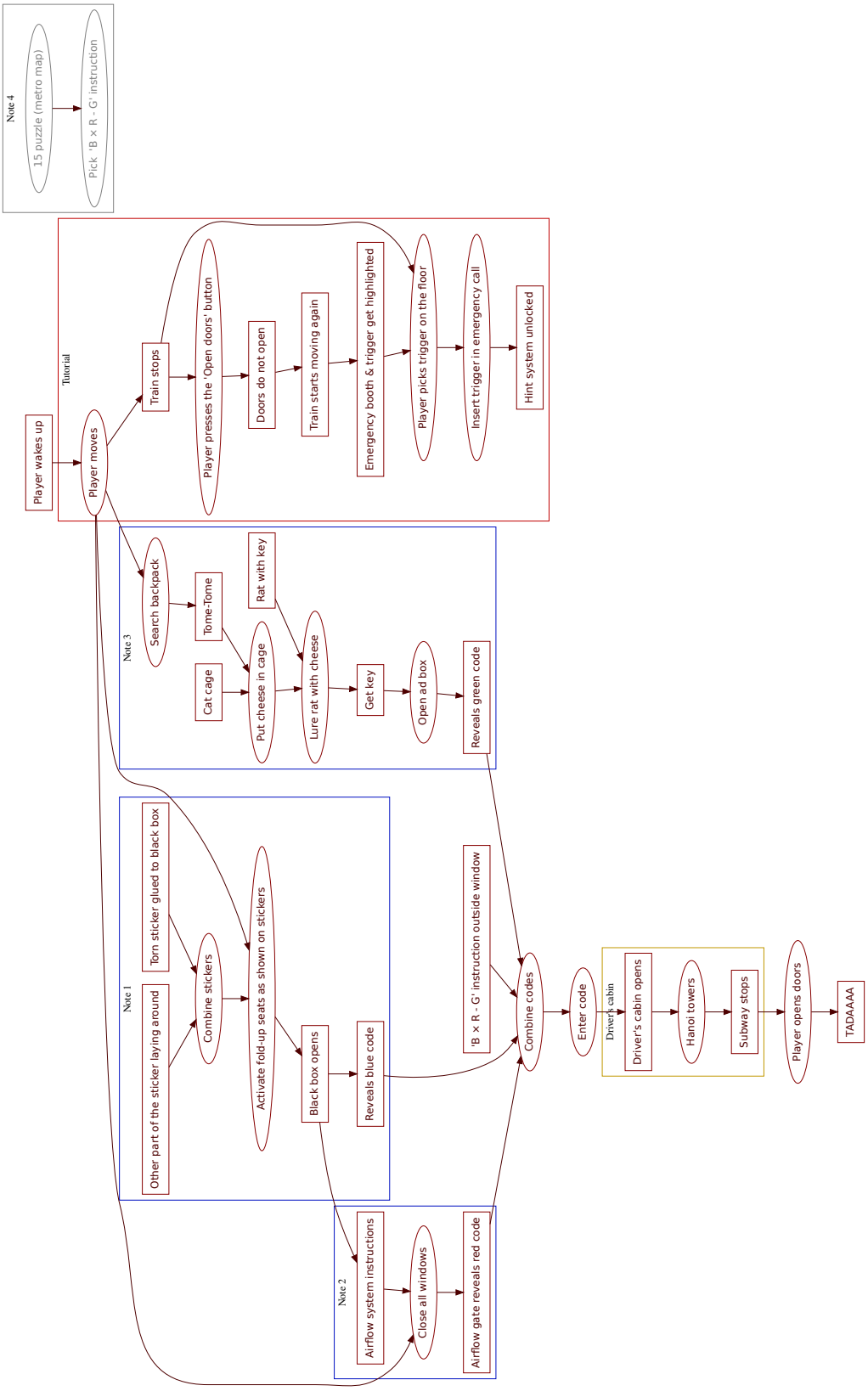


Figure 0.1: Appendix A: Puzzle graph

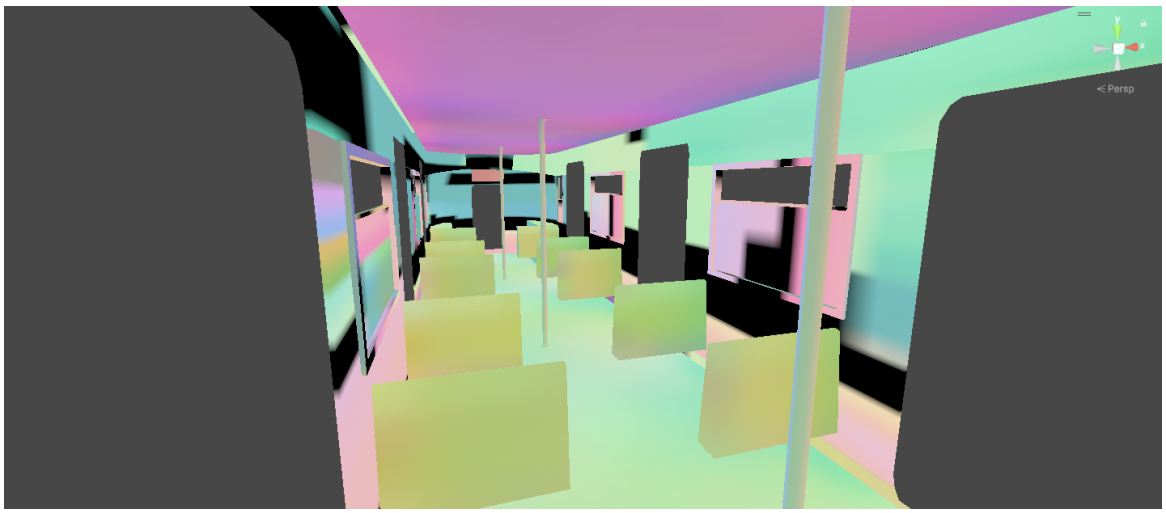


Figure 0.2: Appendix B: Directionally realtime global illumination debug draw

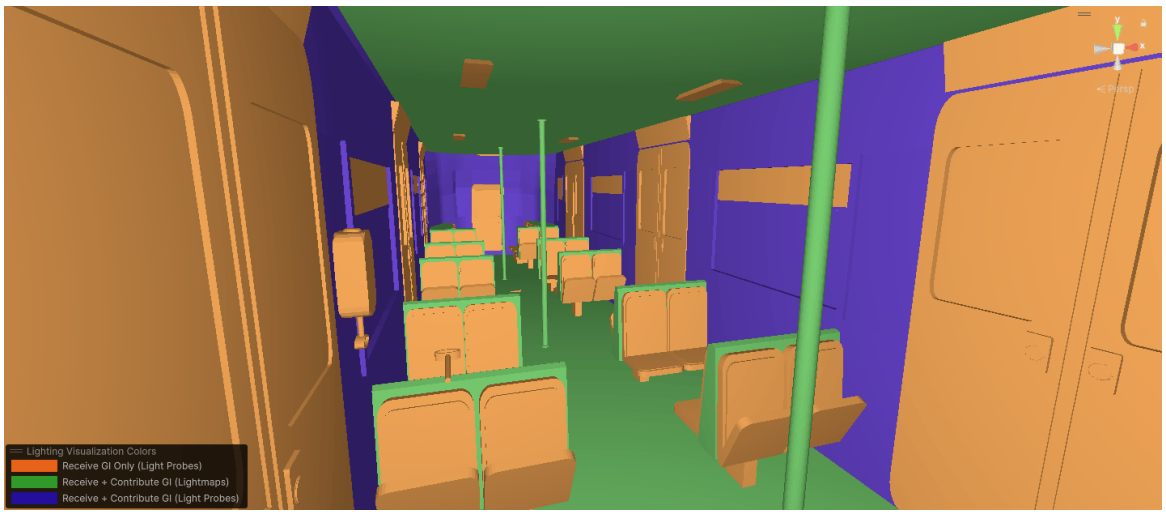


Figure 0.3: Appendix C: Lighting receivers and Global illumination contributors

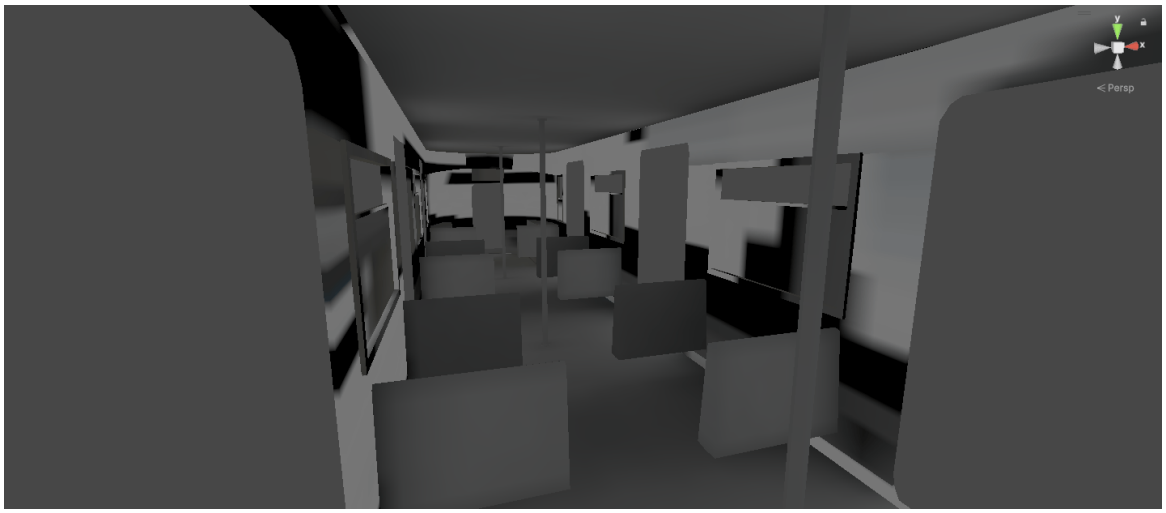


Figure 0.4: Appendix D: Indirect realtime global illumination debug draw

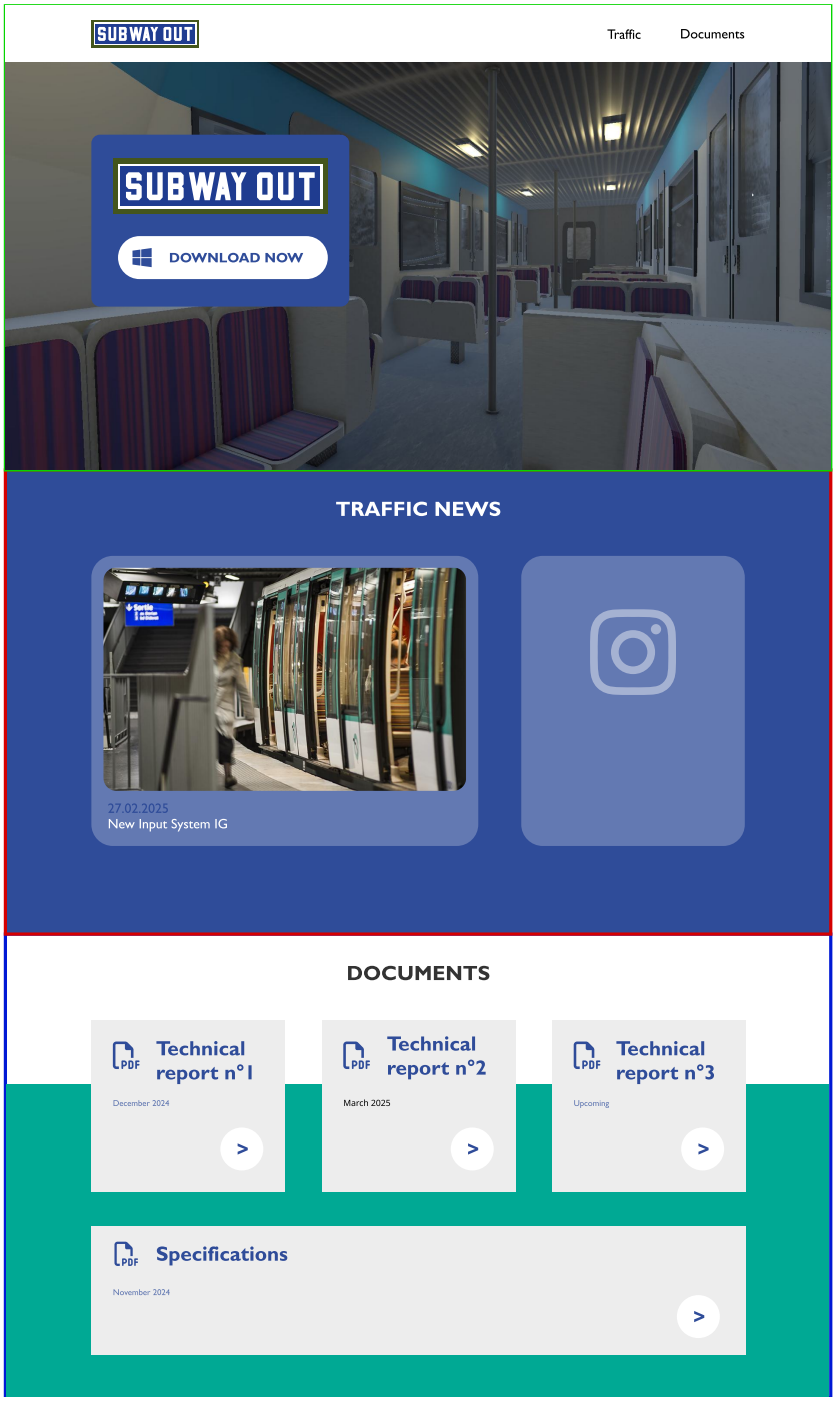


Figure 0.5: Appendix E: Concept art of the website

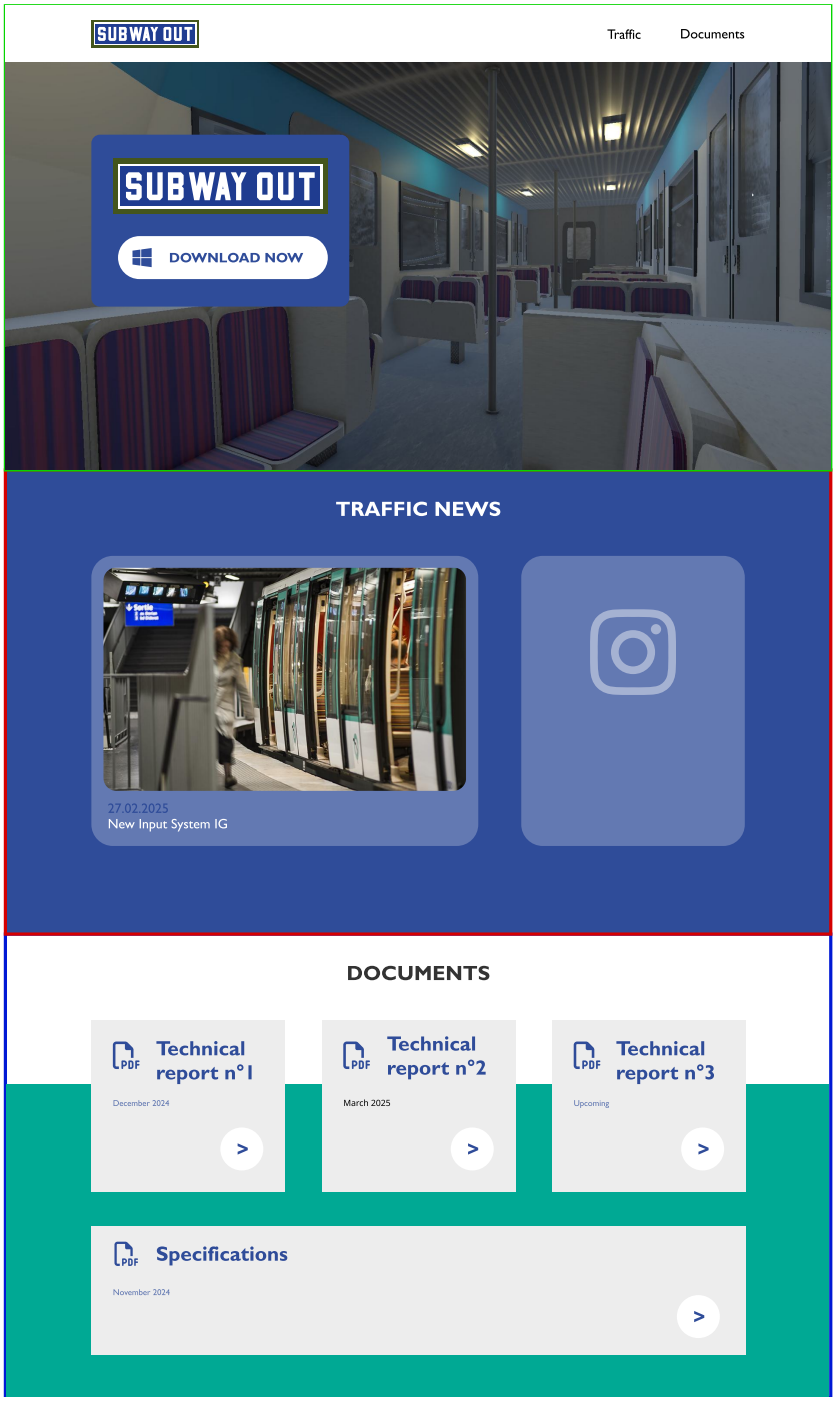


Figure 0.6: Appendix F: Project page of the website