



utbm
université de technologie
Belfort-Montbéliard

Rapport MV50 : Jumeau numérique d'une éco-ferme

MV50 : Projet / Responsable UV : Majhoub DRIDI

*Eléanore Renaud, Quentin Trombini, Alexandre Viala, Benjamin Crouts De Paille,
Paul Cailler*

Table des matières

Introduction	4
Présentation de Badevel	4
Présentation de la ferme	4
Présentation du sujet.....	4
Présentation des robots.....	4
Analyse de l'existant	5
Le jumeau numérique de l'aéroport de Changi.....	5
Le jumeau numérique du port d'Oulu.....	8
Conception de l'interface	10
Démarche centrée utilisateur	10
Questionnaire	11
Réalisation de persona	12
Analyse fonctionnelle du besoin	13
Cahier des charges fonctionnel.....	14
Workflow	14
Github	14
Notion.....	14
Teams.....	15
Outils.....	15
Blender	15
Unity	15
Modèles 3D.....	15
Modèles récupérés sur Sketchfab.....	15
Textures Procédurales	16
Modifieurs personnalisés	16
Modèle 3D.....	17
Base de données	18
Mise en place de base de données sur MongoDB	18
Organisation des collections.....	19
Interaction avec les données	20
La classe DataFetcher	20
Interaction avec Unity	20
Importation des modèles dans Unity	20
Création du terrain.....	21
Mise en place du relief	21

Mouvements et contrôles	22
Interface utilisateur	22
Figma et design des différents composent	22
Pop-up et panel	24
Progress Bar.....	25
Graphiques	25
Système météo et cycle Jour/Nuit en temps réel	26
Conclusion	33

INTRODUCTION

PRESENTATION DE BADEVEL

Badevel est une commune française située dans le Doubs. Dans le cadre d'un plan de modernisation de la ville, de nombreux projets sont entrepris et certains sont confiés à des étudiants de diverses formations du bassin belfortain. Parmi ces sujets, nous retrouvons celui sur lequel nous travaillons : un jumeau numérique de leur ferme connectée.

PRESENTATION DE LA FERME

La ferme de Badevel est un espace géré par des bénévoles qui s'occupent de différentes cultures, aussi bien en intérieur qu'en extérieur. Cependant afin d'alléger la charge de travail des bénévoles, cette ferme est munie de multiples appareils connectés qui vont nous permettre de créer un jumeau numérique de celle-ci.

PRESENTATION DU SUJET

Notre projet de MV50 consiste en la réalisation d'un "jumeau numérique" de la ferme de Badevel. Un jumeau numérique est un logiciel visant à reproduire un objet ou une structure physique (ici la ferme) dans un environnement virtuel. L'utilisation d'un tel logiciel a pour vocation de rendre l'état de la ferme consultable à distance.

Pour fonctionner, le logiciel récupère des données depuis les différents capteurs présents sur les éléments de la ferme connectée pour reproduire l'état visuel de la ferme (par exemple, en utilisant des caméras pour afficher une image des plantes en tant que live texture sur les bacs de culture) et présenter les données récupérées par les capteurs dans une interface diégétique claire et simple d'utilisation.

Pour rendre l'utilisation du jumeau plus naturelle, celui-ci s'utilisera en contrôlant un personnage avec vue à la première personne. L'usage d'une telle caméra permettra à l'utilisateur de se déplacer dans la ferme, comme il le ferait en vrai.

La mise en place des différents capteurs dans la ferme de Badevel n'étant pas finalisée, nous allons relier ce jumeau numérique à une base de données fixe. Celle-ci pourra mettre à disposition de fausses données pouvant simuler le fonctionnement de la ferme.

PRESENTATION DES ROBOTS

Afin d'épauler les bénévoles, plusieurs robots connectés seront intégrés à la ferme :

- Serre Myfood

La serre Myfood permet de recueillir un grand nombre de données différentes pour les cultures intérieures telles que la température, le taux d'humidité et le pH des bacs d'aquaponie.

- Farmbot

Les Farmbots seront posés en extérieur sur différents bacs en bois. Ils permettent de renvoyer en temps réel un affichage de différentes plantations, ainsi que de gérer celles-ci.

- Compost connecté

Un compost connecté pourrait éventuellement arriver. Celui-ci pourrait réaliser différentes mesures et informer l'utilisateur sur la qualité du compost réalisé. Il présenterait notamment le taux d'humidité et la température.

- Ruches connectées

Des ruches connectées vont prochainement arriver, permettant, grâce à différents capteurs de vérifier l'état des abeilles et du miel.

ANALYSE DE L'EXISTANT

Afin de pouvoir concevoir un jumeau de la ferme de Badevel, nous avons été amenés à réfléchir à ce qu'est un jumeau numérique. Un jumeau numérique est un terme relativement récent dans l'histoire de l'informatique. Cette notion est apparue avec la démocratisation de la réalité virtuelle et du métavers. Un jumeau numérique est donc une reproduction numérique d'un lieu plus ou moins à l'identique dans le but de pouvoir interagir avec l'environnement virtuel. On peut ainsi augmenter les données visibles par rapport à la réalité depuis le jumeau.

Afin de mieux visualiser ce que nous devrions implémenter dans ce logiciel, nous avons consulté différents jumeaux déjà existants.

LE JUMEAU NUMERIQUE DE L'AEROPORT DE CHANGI



Figure 1 : Aéroport de Changi

Notre première source d'inspiration est le jumeau numérique de l'aéroport de Changi réalisé par l'entreprise Vouse. Cette entreprise, basée à Singapour, est spécialisée dans la conception de jumeaux numériques interactifs.

MV50 : Jumeau numérique de l'éco-ferme de Badevel

L'aéroport de Changi est visualisable selon deux modes principaux. Un premier mode permet à l'utilisateur de superviser l'aéroport grâce à une vue aérienne. Un deuxième mode permet de se déplacer dans le jumeau numérique en vue à la première personne au volant d'un véhicule terrestre motorisé.



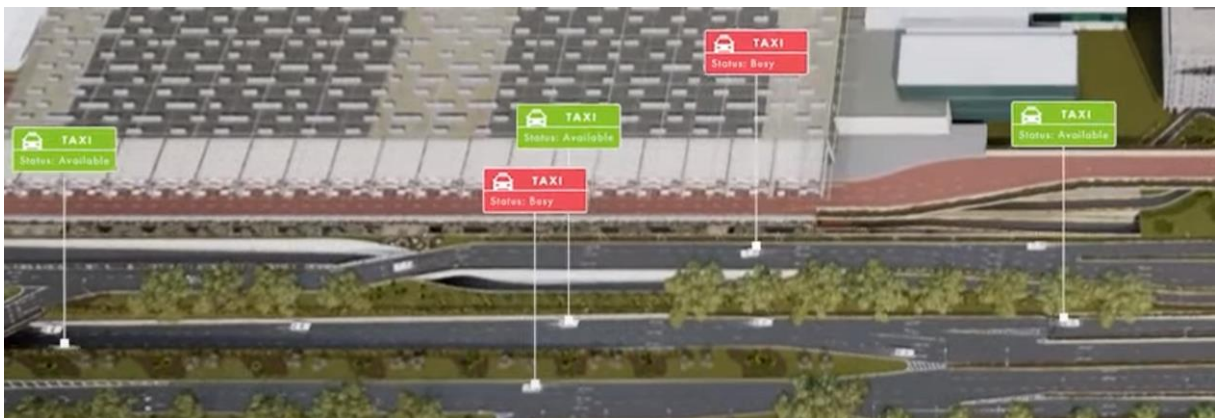
Figure 2 : Mode supervision



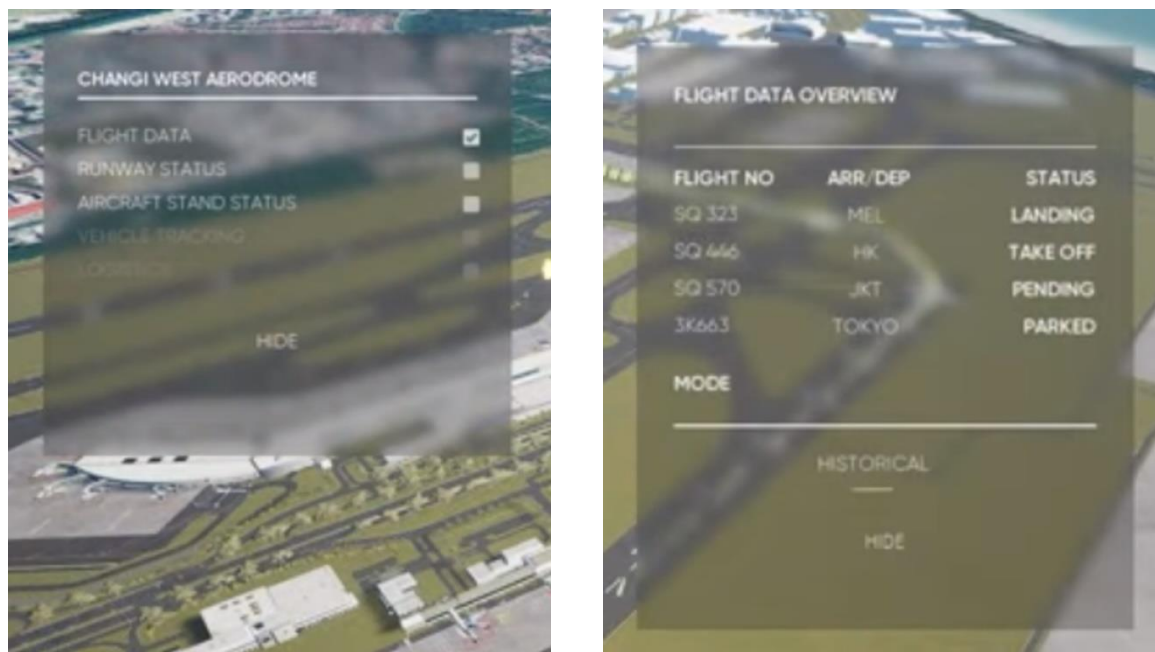
Figure 3 : Mode vue à la première personne

Dans cet exemple, on peut voir que les informations sont affichées de différentes manières :

- Tout d'abord certaines données sont affichées dans des sortes de phylactères. On remarque que ces données sont là pour indiquer l'état d'un élément de la scène. On voit, par exemple ici, l'itinéraire des taxis.



- On a ensuite certaines informations affichées dans des fenêtres de HUD en overlay par rapport au jumeau numérique. Ces fenêtres sont là pour indiquer des informations que l'on pourrait qualifier de "tabulaire". Il s'agit d'informations qu'on retrouverait typiquement dans un aéroport comme la liste des avions présents dans l'aéroport.



- Enfin, on retrouve un dernier type de représentation d'informations par le biais d'objets 3D fictifs. Ces objets ou effets ont été ajoutés à la scène, bien qu'ils n'existent pas dans la réalité, afin de mettre en évidence des objets bien présents comme les avions dans notre cas.



Figure 4 : Rayon rouge pour mettre en évidence un avion

Il est important de noter que ces informations ne sont uniquement visibles qu'en mode supervision. En mode de vue à la première personne, aucune information n'est affichée pour permettre une plus grande immersion.

Ce jumeau numérique mise avant tout sur l'immersion et adopte de ce fait un design très épuré. Ici, seules certaines informations bien choisies par Vouse sont affichées.

LE JUMEAU NUMERIQUE DU PORT D'OULU



Figure 5 : Exemple jumeau numérique port d'Oulu

Notre deuxième source d'inspiration est le jumeau numérique du port d'Oulu réalisé par Sitowise. Sitowise est une entreprise spécialisée dans la production de solutions liées aux villes intelligentes (les communications, l'internet des objets, etc.).

Tout comme le jumeau de l'aéroport de Changi, le port d'Oulu dispose de deux modes : un mode supervision et un mode immersion. Le mode immersion est disponible sur appareils mobiles. Il est à noter que le mode immersion utilise ici le gyroscope et les accéléromètres des appareils mobiles afin de pouvoir bouger la caméra en déplaçant le téléphone.

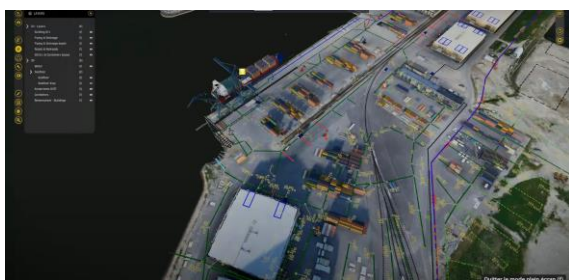


Figure 6 : Mode supervision

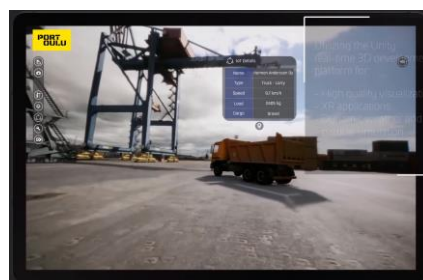


Figure 7 : Mode immersion sur tablette

Les informations relatives au jumeau numérique sont également affichées de trois manières différentes dans ce jumeau :

- Tout d'abord les données les plus représentées sont des données tabulaires qui sont cette fois-ci intégrées de manière contextuelle au jumeau. En effet, Sitowise a pris le parti

d'afficher un grand nombre de données liées à des objets dans des grands phylactères. On peut par exemple observer les données d'un bateau en direct en cliquant sur ce dernier.

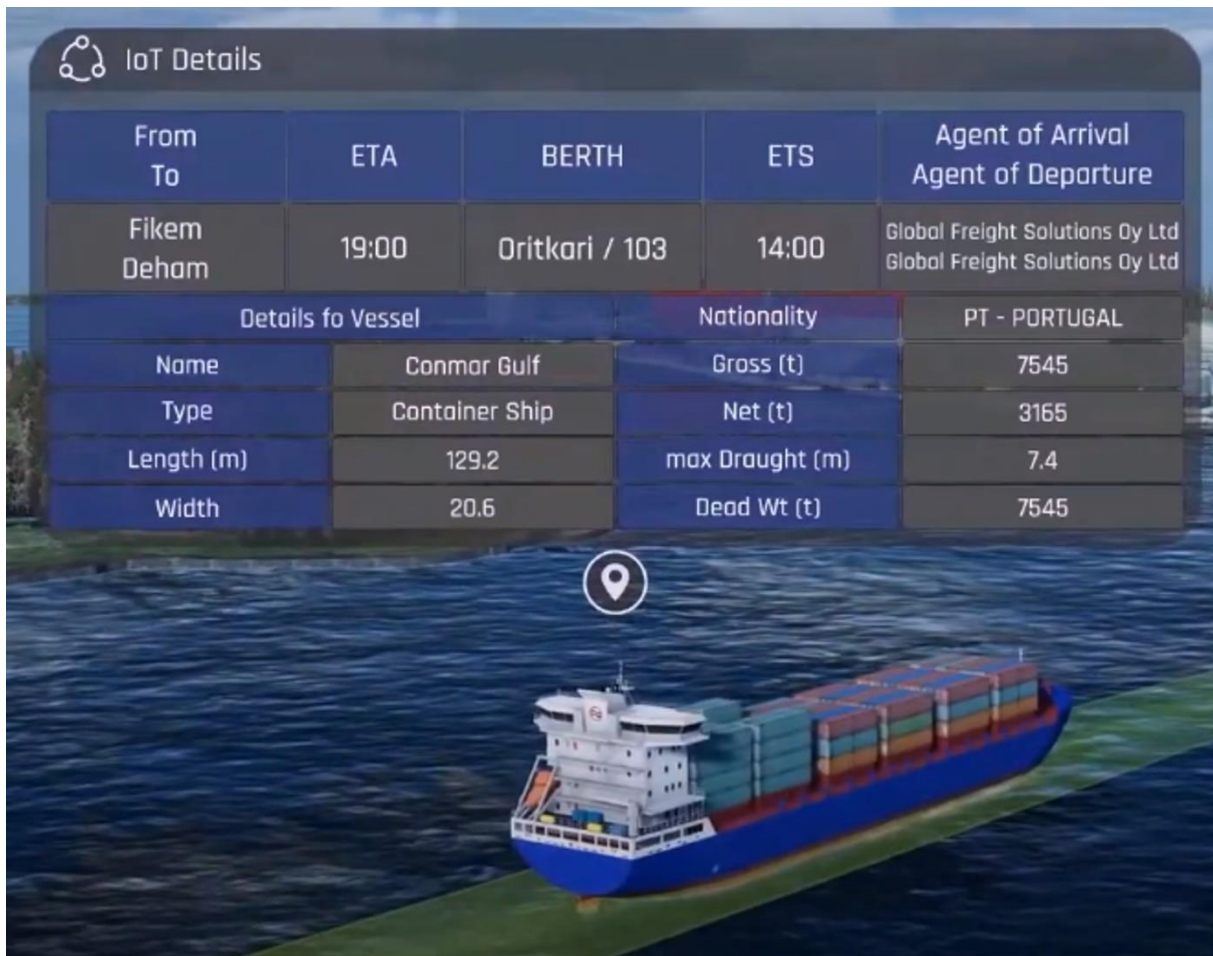
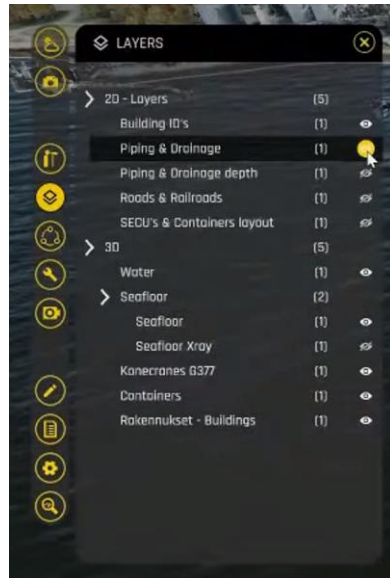


Figure 8 : Exemple jumeau numérique pour cargos

- On a ensuite quelques données de HUD en overlay qui vont surtout permettre de faire varier la richesse des informations affichées à l'écran. L'utilisateur peut ainsi choisir de mettre en évidence certaines informations tout en dissimulant celles dont il n'a pas besoin. Le HUD est beaucoup plus compact que celui du jumeau précédent.



- Ce jumeau ajoute un certain nombre d'objets et de lignes imaginaires sur le port existant. Ces données permettent notamment de visualiser les flux sur le port. Certaines utilisations permettent également de faire de la prédiction de mouvements.

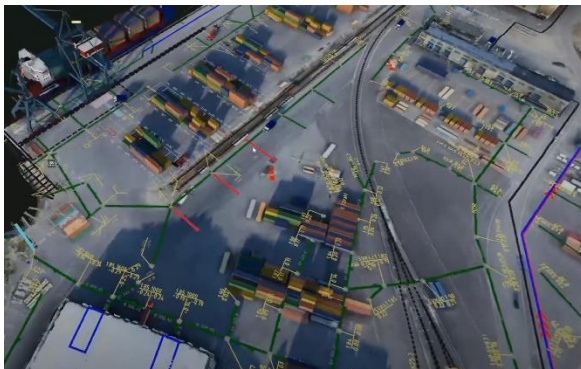


Figure 9 : Flux sur le port



Figure 10 : Prédiction des mouvements d'une grue

Le jumeau numérique du port d'Oulu utilise bien plus les objets fictifs que celui de l'aéroport. Sitowise a pris le parti de représenter un grand nombre de données et de permettre à l'utilisateur de choisir le degré de précision qu'il désire. Ce logiciel est pensé comme un réel outil de travail pour pouvoir optimiser et améliorer le port. Les données contextuelles sont d'ailleurs visibles également en mode immersif.

CONCEPTION DE L'INTERFACE

DEMARCHE CENTREE UTILISATEUR

Afin de concevoir un logiciel correspondant le plus possible aux besoins de nos utilisateurs. Nous avons suivi une démarche centrée utilisateur.

QUESTIONNAIRE

Lors de la présentation de notre sujet, nous avons posé un certain nombre de questions à notre encadrant pour savoir ce qu'il était attendu de nous. Nous lui avons ensuite posé des questions à plusieurs reprises pour désépaisser le problème et se rapprocher le plus possible de ce qu'attendaient nos futurs utilisateurs.

Enfin, pour pouvoir coller le plus possible aux attentes de nos utilisateurs, nous avons réalisé un court questionnaire de 4 questions pour se faire une idée globale des utilisateurs qui utiliseront l'application. Cette étape nous a permis de déterminer les différents profils types d'utilisateurs.

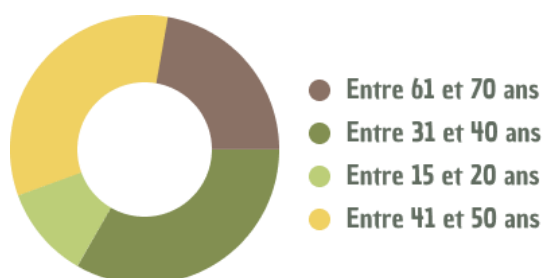


Figure 11 : Réponses à la question "Quel âge avez-vous ?"

Nous nous sommes ainsi rendus compte qu'une bonne partie de nos utilisateurs étaient d'âge mûr. Cela nous permet de savoir que certains automatismes que nous avons développés en tant que personnes plus jeunes ne seront pas forcément présents chez eux. Nous devons donc adapter certaines fonctionnalités en ajoutant, par exemple, des tutoriels.



Figure 12 : Réponses à la question "Quelle est votre catégorie socio-professionnelle ?"

Une grande partie de nos utilisateurs ont également une profession qui requiert d'utiliser un ordinateur. Ces personnes doivent donc au moins avoir des réflexes de base avec des logiciels de traitement de texte ou de tableur. Toutefois, un tiers d'entre eux est également à la retraite. Nous pourrions donc intégrer des commandes classiques de logiciels mais il faudra tout de même prévoir un moyen alternatif d'y accéder.

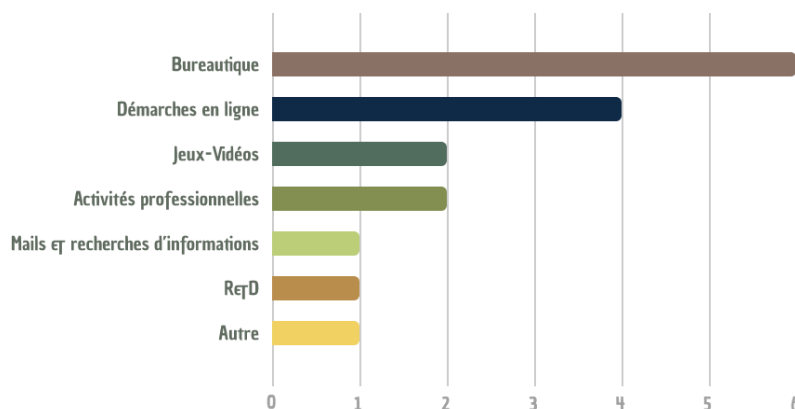



Figure 13 : Réponse à la question "Pour quelle raison utilisez-vous un ordinateur ?"

Enfin, au travers de notre dernière question, nous demandions aux utilisateurs quelles étaient leur utilisation d'un ordinateur. Comme nous pouvions déjà le voir au travers des questions précédentes, les utilisateurs sont pour la plupart habitués aux logiciels de bureautique. On voit également qu'ils utilisent leur ordinateur pour des démarche en ligne. Ces personnes sont donc habituées à remplir des formulaires et nous pourrions de ce fait, intégrer des formulaires (pour la prise de permanence par exemple) sans pour autant expliquer longuement le fonctionnement de ces derniers.

REALISATION DE PERSONA

Pour nous aider à concevoir une application proche de nos utilisateurs, nous avons conçu des personas correspondant à des profils types de nos utilisateurs. Ces personas nous permettront de vérifier que les fonctionnalités conçues correspondent bien aux utilisateurs que nous visons.

Christophe RÉMY



AGE : 35
FORMATION : Baccalauréat professionnel
STATUT : Marié
PROFESSION : Agriculteur
LOCATION : Badevel
CONNAISSANCES TECHNOLOGIQUES : Moyennement

Biographie

Il habite à Badevel depuis son enfance. Il est particulièrement investi dans la vie de sa commune en aidant à l'organisation d'activités ou d'événements (fêtes de villages).

Besoins primordiaux

- Christophe veut pouvoir accéder à la ferme à distance.
- Christophe veut pouvoir accéder au tableau des permanences.
- Christophe veut pouvoir montrer aux autres comment se servir de l'outil.
- Christophe veut pouvoir interagir avec la ferme à distance (démarrer le robot par exemple)


Frustrations

- Christophe est trop occupé par sa propre ferme.
- Il habite trop loin de la ferme connectée.

Personnalité

Passionné, Pédagogue, Ouvert d'esprit

Platform



Windows

“ Je sais me servir d'un ordinateur. Je m'en sers pour payer mes factures et faire des commandes en ligne. ”

Figure 14 : Persona d'une personne d'âge moyen

ANALYSE FONCTIONNELLE DU BESOIN

- FC 1 : Voir les éléments du jardin
 - Modéliser la serre
 - Modéliser le bassin d'aquaponie
 - Modéliser les bacs de plante
 - Modéliser le chauffage
 - Modéliser les fenêtres à battant
 - Modéliser le relief du terrain à l'aide d'une height map
 - Modéliser la ruche
 - Modéliser le compost ?
 - Modéliser les bacs de permaculture
 - Modéliser les bacs extérieurs avec maisonnette
- FC 2 : Rendre le logiciel interactif
 - Ajouter des menus
 - Ajouter des animations
 - Ajouter un système de gestion des permanences
- FC 3 : Lier l'application aux données
 - Héberger une BDD¹ intermédiaire avec les informations des robots
 - Créer une API ²pour interagir avec les données de la BDD
 - Intégrer le repository dans Unity
- FC 4 : Récupérer les données des robots
 - Créer un backend³

¹ BDD : Une base de données permet de stocker et de retrouver des données structurées, semi-structurées ou des données brutes ou de l'information, souvent en rapport avec un thème ou une activité ; celles-ci peuvent être de natures différentes et plus ou moins reliées entre elles.

² API : est un ensemble normalisé de classes, de méthodes, de fonctions et de constantes qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels.

³ Backend : un back-end (parfois aussi appelé un dorsal) est un terme désignant un étage de sortie d'un logiciel devant produire un résultat. On l'oppose au front-end (aussi appelé un frontal) qui lui est la partie visible de l'iceberg².

- Connecter l'appli au backend
- "Reverse engineer"
- FC 5 : Paramétrer à l'aide de fichier de config
 - Réfléchir à la structure des fichiers de config
 - Parser les fichiers dans Unity

CAHIER DES CHARGES FONCTIONNEL

L'application doit :

- Être une représentation la plus fidèle possible de la ferme existante à Badevel
- Afficher les différentes données de chaque objet en les récupérant depuis une base de données
- Permettre de se déplacer en vue à la première personne
- Permettre aux bénévoles de se connecter
- Permettre aux bénévoles d'écrire sur le tableau de permanences
- Afficher la météo et le ciel actuel de la scène via une skybox⁴

WORKFLOW

GITHUB

GitHub est une forge git disponible en ligne nous permettant de stocker les différentes versions de notre code et de nos assets au fil du temps et des sujets à développer. Il facilite la collaboration lors du développement. Pour le moment il nous a principalement servi à stocker les différentes versions des modèles qui ont été réalisées.

NOTION

Notion est un logiciel permettant de prendre toute sortes de notes et de les partager, il est aussi capable de créer des tableaux, des kanbans et peut donc être utile pour le suivi de projet. Nous l'avons lié à GitHub afin de pouvoir visualiser les issues de ce dernier directement dans Notions. Cette application est au cœur de notre projet puisque c'est dans celle-ci que nous rédigeons les documents, stockons les différentes ressources (liens, image, etc...) et que nous organisons le projet.

⁴ Skybox : est un procédé graphique permettant de donner, dans un espace tridimensionnel, l'illusion que cet espace est plus étendu qu'il ne l'est réellement.

TEAMS

Microsoft Teams nous permet de communiquer avec le porteur de projet et donc d'avoir des retours rapides lorsque nous avons des questions et des avancés à présenter, de plus c'est grâce à cet outil que nous pouvons organiser nos réunions à distance.

OUTILS

BLENDER

Blender est un logiciel de modélisation et d'animation 3D open-source. Le logiciel dispose de plusieurs modes de conception dont un mode d'édition de maillage et un mode sculpture. Contrairement à un logiciel de conception assistée par ordinateur, blender n'a pas pour vocation de permettre de concevoir des pièces mécaniques nécessitant des tailles précises. En effet, il n'existe aucun outil de cotation dynamique dans ce logiciel. Toutefois, Blender est particulièrement adapté à la conception de modèles "artistiques" ainsi que pour réaliser des objets dits "organiques" présentant des formes plus complexes (par exemple des plantes, des animaux, des micro-organismes, etc.). Blender est enfin utilisé pour réaliser des rendus et de l'animation de modèles 3D grâce à son large panel d'outils.

Dans le cadre de notre projet, nous avons utilisé Blender afin de concevoir les différents éléments de la ferme de Badevel et pour texturer ces éléments.

UNITY

Unity est un logiciel de création de jeux, nous l'avons choisi car c'est l'un des leaders dans le domaine, il est facile à prendre en main, possède une bonne documentation et facilite l'importation des modèles depuis Blender. Il nous permet de créer le terrain et de le modeler à nos souhaits puis d'importer tous les modèles que nous avons fait précédemment depuis Blender. De plus, il permet de créer facilement un système de déplacement à la manière de Google Maps avec une vue à la première personne ainsi qu'une vue de haut en 3D isométrique.

MODELES 3D

MODELES RECUPERES SUR SKETCHFAB

Les modèles 3D tels que ceux des robots de Farmbot ou encore la ruche qui n'existent pas pour l'instant, sont issues du site Sketchfab. Ce site met à disposition de manière gratuite (rarement payant) des modèles 3D réalisés par des organisations tel que des entreprises ou par des particuliers qui souhaitent partager leurs réalisations.

Farmbot met donc à disposition le modèle 3D de ses robots sous forme de fichier .collada (issue du logiciel 3D Sketchup) ou STL (fichiers de stéréolithographie). Blender ne prenant pas en charge les fichiers Sketchup, nous avons utilisé un plugin réalisé par un particulier pour importer le fichier. Ainsi, pour plus de facilité dans notre organisation et pour la programmation sous Unity, nous avons exporté tous nos modèles 3D en *.blend* et en *.fbx*.

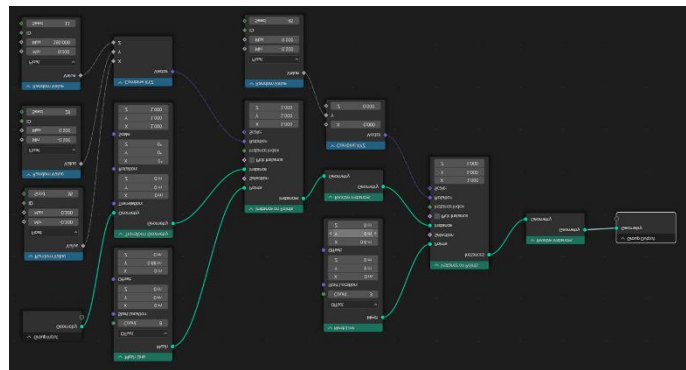
Concernant la ruche, le modèle 3D a été réalisé par un particulier qui l'a rendu disponible sur Sketchfab. La ruche possédait une vue globale ainsi qu'une vue de l'intérieur de la ruche avec

TEXTURES PROCEDURALES

MODIFIERS PERSONNALISÉS

Bien que de nombreux modifieurs existent, ils ne permettent pas toujours de réaliser la tâche que l'on désire, il est alors possible de personnaliser ces modifieurs afin de les faire coller à une tâche plus spécifique.

16 | Page



MODELE 3D

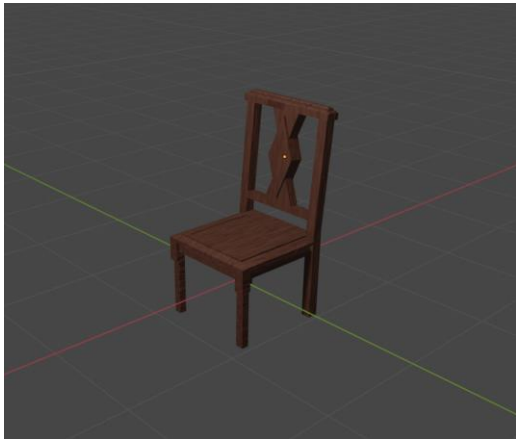


Figure 17 : Chaise de la serre



Figure 18 : Table de la serre



Figure 19 : Rideau métallique de la cabane

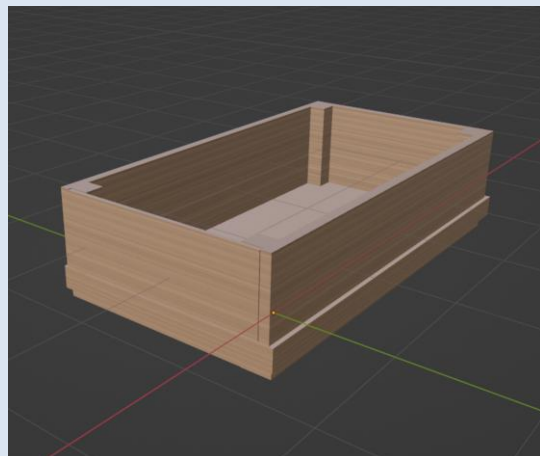


Figure 20 : Bac de permaculture

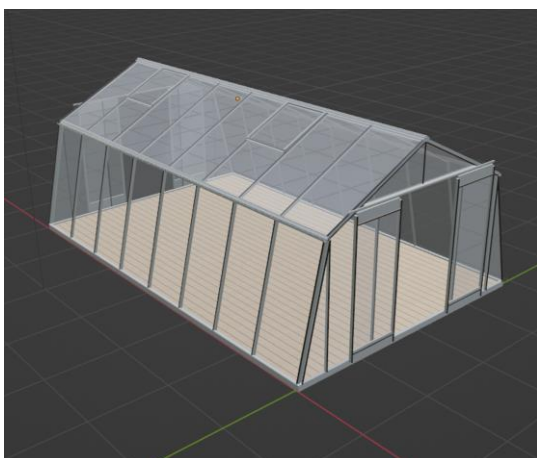


Figure 21 : Serre connectée



Figure 22 : Bac d'aquaponie



Figure 23 : Farmbot Genesis

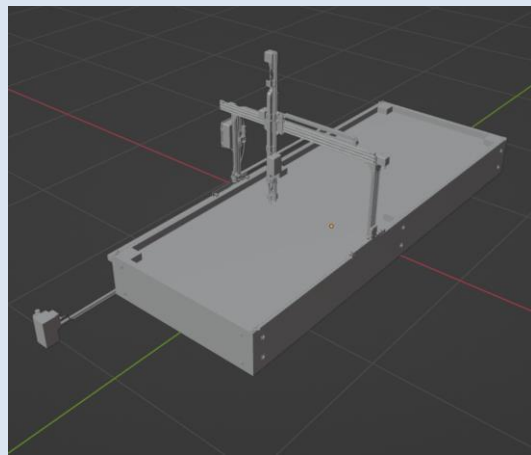


Figure 24 : Farmbot Express

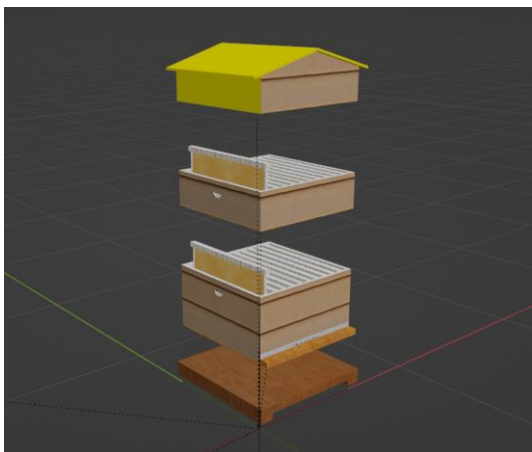


Figure 25 : Ruche connectée



Figure 26 : Poêle

BASE DE DONNEES

MISE EN PLACE DE BASE DE DONNEES SUR MONGODB

Pour ce projet nous souhaitons tout d'abord partir sur une récupération de données à partir de la REST API () de Pocket base. Cependant l'intégration de celle-ci et l'envoi de requête n'est adapté qu'aux applications ayant pour objectif un rendu WebGL. Il nous a donc fallu revoir notre backend. Nous avons décidé de passer plutôt par l'API de MongoDB et son extension Realms. Nous avons installé cette dernière dans le projet via NPM que nous avons réussi à lier à Unity. Une fois l'extension Realms installé, nous pouvions envoyer des requêtes à l'API afin qu'elle nous retourne les données. L'utilisation de Realms nous permet d'interpréter chaque collection comme un objet et de créer à chaque requête un objet contenant les données.

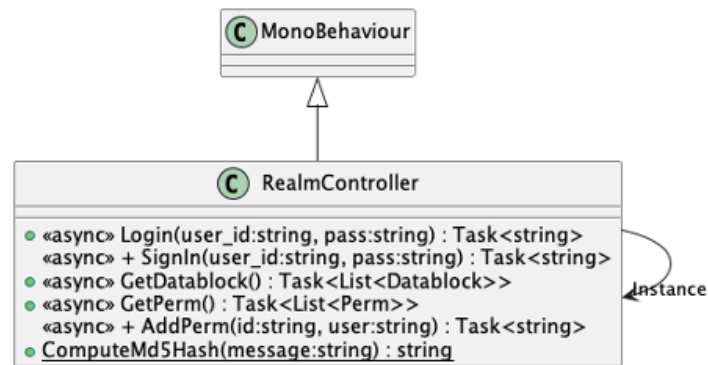


Figure 27 : realmController

ORGANISATION DES COLLECTIONS

Nous avons organisé notre base de données selon 3 collections :

- Tout d'abord la base de données des utilisateurs, assez basique chaque utilisateur est identifié par un nom qui lui sert d'identifiant unique (tant que le nombre d'utilisateur n'est pas trop élevé cela ne pose pas de problème) et un mot de passe. Afin d'éviter que le mot de passe ne soit stocké en clair dans la base de données les mots de passe ont été encryptés grâce à la méthode md5.
- Notre deuxième base de données correspond aux données que nous souhaitons afficher. Nous sommes partis du principe qu'il y aurait au maximum une prise de mesure par heure, ainsi pour trier nos données plus facilement il a été décidé de se servir de la date et de l'heure de mesure comme identifiant, sous le format AAAAMMJJHH (le 16 juin 2023 à 15H devient 2023061615). Cela permet aussi lors des requêtes de récupérer uniquement les données qui nous intéressent directement à partir de leur ID. En plus de cet ID chaque collection contient toutes les données que l'on souhaite récupérer, il est ainsi plus simple et rapide de tout afficher au lancement. Les données sont stockées sous la forme du nom de l'objet qui est associé puis du nom de la valeur (exemple : greenhouse_humidity).

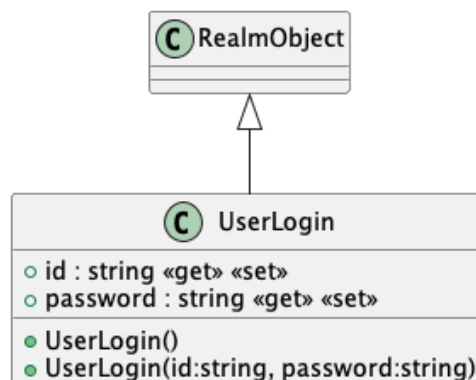


Figure 28 : realmObject

- Notre dernier objet contient les permanences auxquels les bénévoles peuvent s'inscrire, le système d'ID reprend le format de date utilisé par les collections de données et contient un deuxième champ contenant le nom de la personne inscrit à cette permanence.

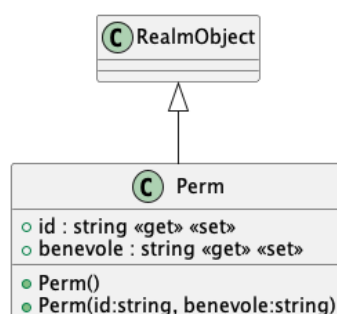


Figure 29 : realmObject

INTERACTION AVEC LES DONNEES

Pour se connecter à la base de données depuis Unity, nous utilisons le Realms SDK. Ce dernier contient les fonctions permettant de se connecter à l'API et d'en extraire des données. Une fois cette connexion effectuée, nous pouvons utiliser `Realms.find(test)`. Cette fonction va renvoyer tous les objets dont l'ID contient test. La fonction `realm.add()`, quant à elle, permet d'ajouter des entrées via différents objets.

LA CLASSE DATAFETCHER

La classe `DataFetcher` permet l'utilisation des données récupérées dans la base de données. Cette classe utilise un système d'observable, les événements Unity. En effet, la classe va exposer aux autres scripts un objet de type `UnityEvent`. Ceux-ci vont ainsi pouvoir « subscribe » à l'événement en spécifiant une fonction. La classe `DataFetcher` va ensuite récupérer les données de la base de données toutes les `n` unités de temps. Si celles-ci sont récupérées avec succès, l'événement est invoqué. Toutes les fonctions spécifiées lors du « subscribe » seront alors exécutées, et les données présentes dans l'interface seront alors mise à jour.

INTERACTION AVEC UNITY

IMPORTATION DES MODELES DANS UNITY

Afin d'importer les modèles et leur texture dans unity, il est nécessaire, dans le cas d'une texture réalisée avec des shaders sur blender, de **bake** la texture. **Bake** une texture dans Blender consiste à enregistrer une texture sur une image à plat pour l'appliquer ensuite à un objet 3D. Cela permet d'optimiser les performances de rendu, de créer des textures de haute qualité pour Unity, ou de prévisualiser rapidement des textures sur un objet. De plus dans le cas d'un objet avec des textures complexes comme pour la maisonnette extérieure, il a fallu réaliser plusieurs **bake** avec des caractéristiques différentes. Une fois la textures en jpg obtenue, elle peut être ajoutée au .fbx produit lors de l'importation. Ainsi, une fois dans Unity, l'exportation des textures permet à l'objet de retrouver ses motifs.

CREATION DU TERRAIN

Le terrain utilise le paquet "terrain tools" distribué par Unity. Celui-ci permet de réaliser le terrain à l'aide de différents pinceaux à appliquer sur une surface. Ceux-ci permettent de modifier le niveau du terrain, d'ajouter du bruit pour rendre le terrain plus naturel, et de peindre directement les texture sur le sol. Étant donné que nous n'avons pas de données de topologies précises, le terrain sera reproduit manuellement de manière à ressembler le plus possible à celui de la ferme de Badevel.

MISE EN PLACE DU RELIEF

Pour créer le relief nous n'avons pas accès aux données topologique de la zone. Nous avons donc décidé de créer une height map directement en fonction des visuelles auxquelles nous avons accès. Nous avons donc utilisé le logiciel Gimp pour réaliser cette height map.

Nous avons d'abord réalisé un bruit sur un fond gris moyen pour faire les micro-détails de la height maps puis nous avons utilisé l'outil assombrir/éclaircir de Gimp pour former le relief. En éclaircissant, on augmente la hauteur du relief et en assombrissant on diminue. Cela nous permet de créer un relief réaliste.

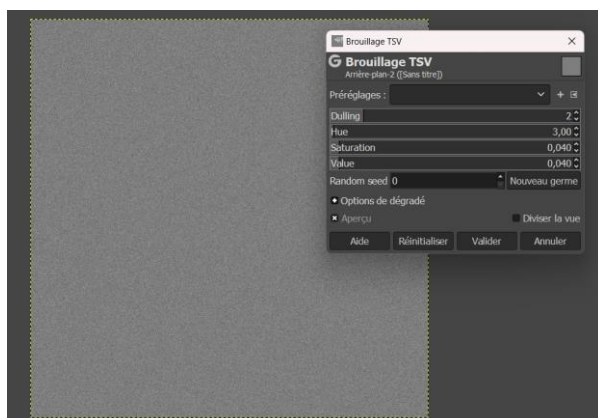


Figure 30 2 : Bruit de fond

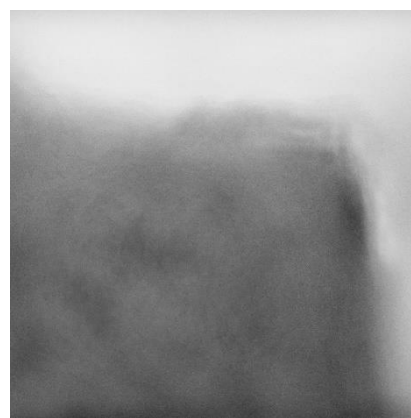


Figure 31 3 : Height map finie

Une fois la height map créée nous avons utilisé terrain tools (une extension unity) pour créer une texture de terrain et l'appliquer partout. Grâce à cet outil nous pouvons ajouter une texture de terre, d'herbe et des plantes au sol.

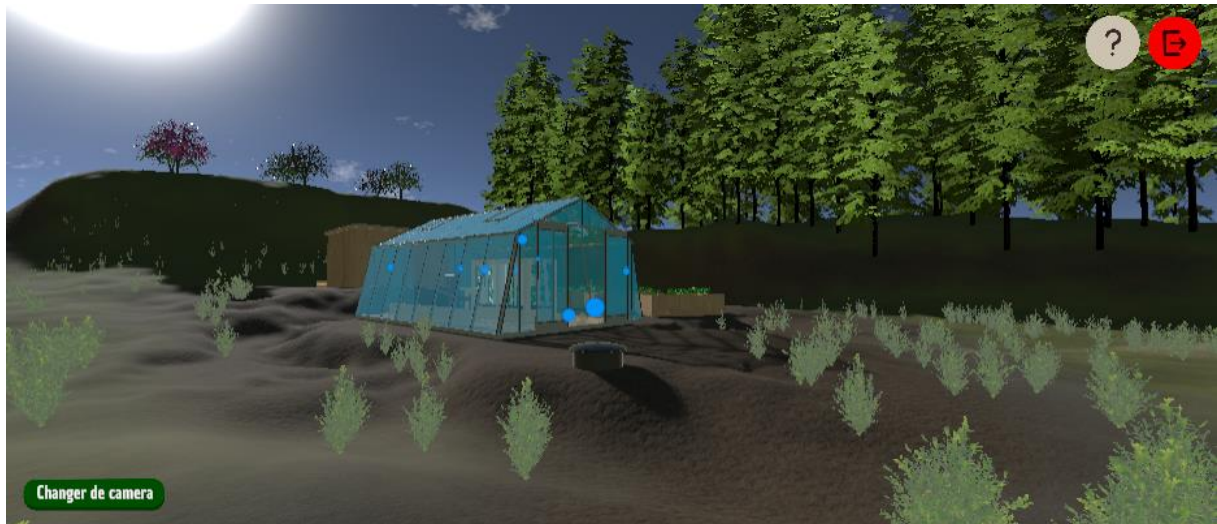


Figure 32 : Terrain final

MOUVEMENTS ET CONTROLES

Un premier système de déplacement a été développé. Celui-ci, inspiré du fonctionnement de Google Maps, a pour vocation d'être facile d'utilisation pour les personnes n'ayant pas l'habitude de se déplacer dans un univers virtuel 3D. Celui-ci utilise uniquement la souris. En maintenant le clic droit, il est possible de manipuler la caméra. Lorsque le curseur de la souris est placé sur une portion du terrain, un cercle bleu apparaît à cet endroit. Une clique gauche suffit alors pour déplacer le personnage à l'endroit indiqué.

En plus de ce système de déplacement, deux autres sont prévus :

- Un système de déplacement plus classique, reprenant les touches utilisées par les jeux "FPS". L'utilisateur utilise alors les touches ZQSD / les touches directionnelles pour déplacer son personnage et contrôle la rotation de la caméra directement avec la souris. Ce type de contrôle est principalement dédié aux personnes habituées aux jeux vidéo.
- Une vue totalement différente où l'utilisateur a l'aperçu entier de la ferme en vue de dessus. Cette vue s'inspire du "mode supervision" des différents exemples de jumeaux numériques que nous avons pu étudier.

INTERFACE UTILISATEUR

FIGMA ET DESIGN DES DIFFERENTS COMPOSANTS

Pour la réalisation de la partie UI sur Unity, nous avons déjà commencé par entamer une phase préliminaire de design à l'aide de Figma. Il s'agit d'un outil de conception et de prototypage d'interfaces utilisateur basé sur le cloud qui permet une collaboration interactive. Nous avons créé une charte graphique avec une palette de couleurs et une liste des différents éléments graphiques devant apparaître dans Unity.

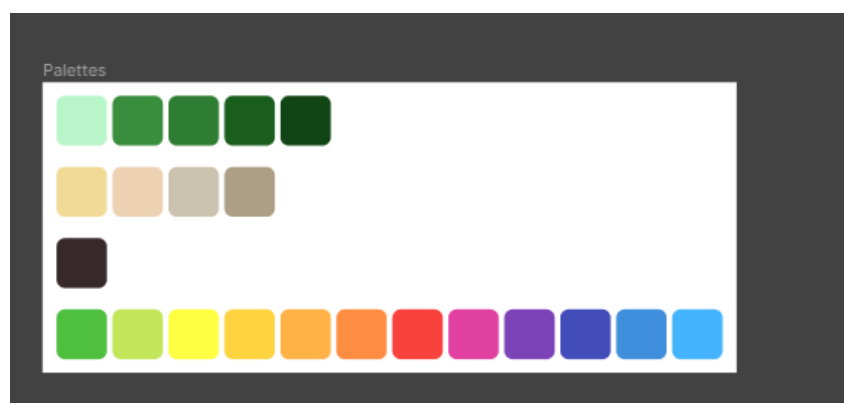


Figure 33 : Palette de couleurs

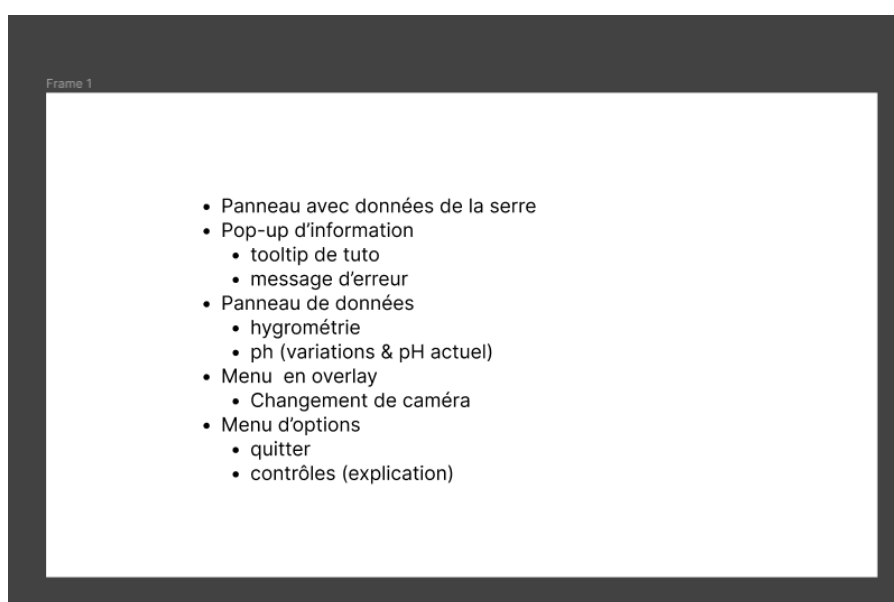
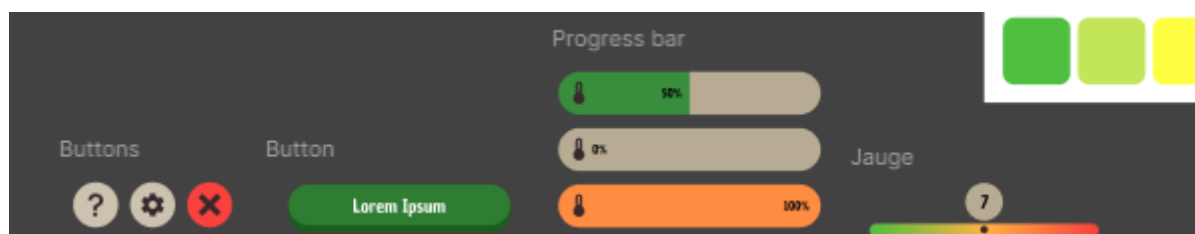
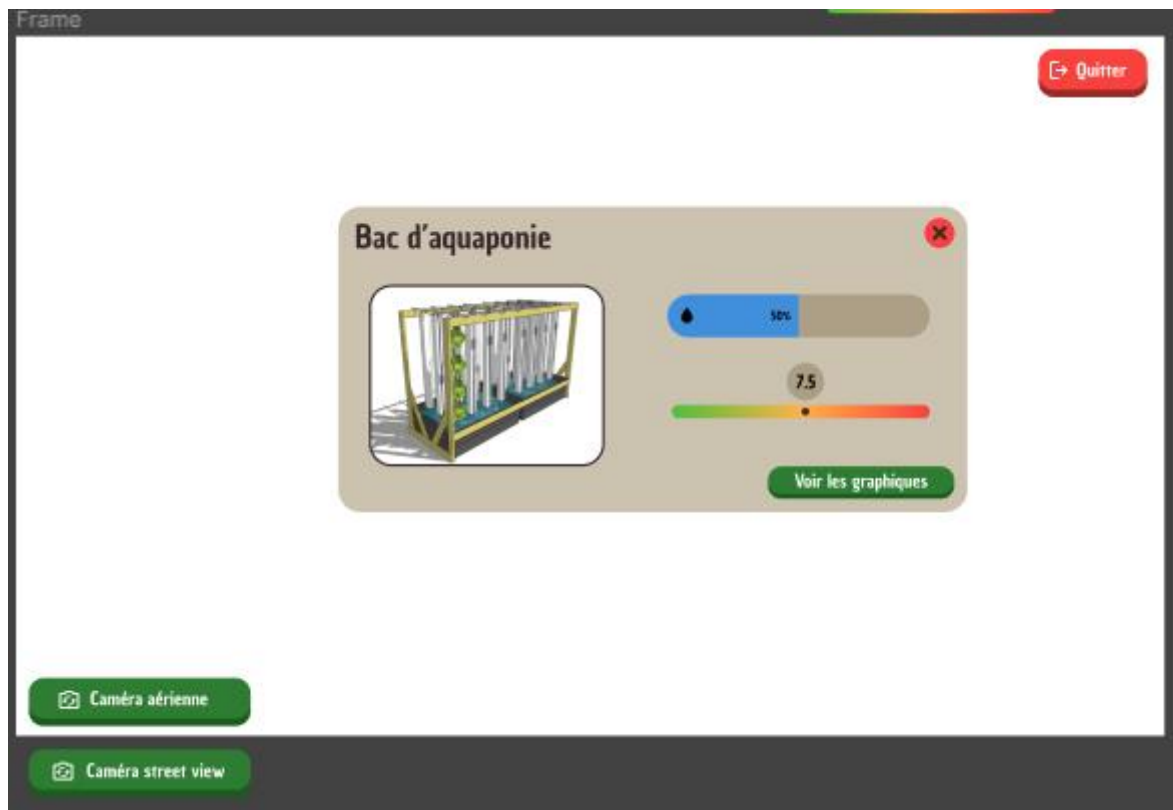


Figure 34 : Liste éléments UI

Nous avons par la suite créé et imaginé les différents composants en prenant en compte l'expérience utilisateur avec notamment la taille des boutons, l'affichage des données, la lisibilité et la fluidité de l'ensemble.





Une fois le prototypage terminé, nous avons utilisé une tablette graphique pour réaliser les boutons, les progress bar ou encore les icônes informatives par exemple. Le logiciel Procreate a été utilisé notamment pour son fonctionnement complet et son nombre d'outils. Le reste des éléments a directement été réalisé dans Unity avec les objets UI, TextMeshPro etc.



POP-UP ET PANEL

Pour afficher l'interface de notre application, nous avons imaginé deux systèmes de pop-up différents. Le premier est un popup extra-diégétique s'affichant simplement sur l'écran de l'utilisateur. Il est utilisé pour la connexion, l'aide, ainsi que pour afficher des données numériques. Le second, quant à lui, est diégétique : il s'affiche directement dans le monde

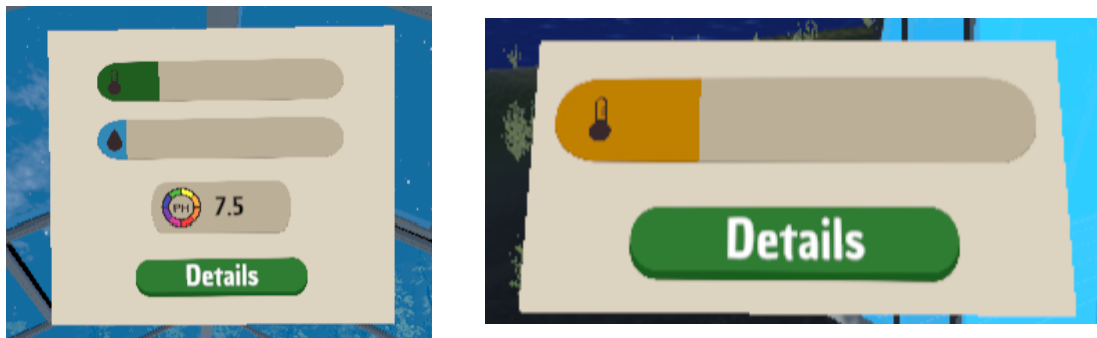
virtuel. Celui-ci s'affiche uniquement lorsque l'utilisateur s'approche de l'élément concerné, et présente essentiellement des visuelles tel que des barres de progression.

Ces deux types de pop-ups ont été réalisées de manière à s'adapter à tous les gameObjects de types « panels ». Ainsi, pour réaliser un nouveau pop-up, il suffit de créer un panel, de le remplir avec les informations nécessaires, et de le glisser dans les scripts respectifs chargés de les générer.

Cela nous permet de limiter le temps nécessaire à la réalisation de nouveaux pop-ups et de faciliter leur réutilisation.

PROGRESS BAR

Les barres de progression sont utilisées pour afficher graphiquement différentes données telles que la température ou l'humidité. Elles représentent ces informations sous forme de pourcentage. Pour réaliser cela, nous avons utilisé des objets d'image dans la section interface utilisateur (UI) d'Unity, en leur attribuant une couleur différente pour représenter la progression. Nous avons ajusté les paramètres d'affichage de l'image et développé un script pour lier les valeurs de la base de données à ces éléments visuels.



GRAPHIQUES

Dans le jumeau numérique de Badevel, il est nécessaire d'afficher les données des capteurs. Toutefois, simplement afficher la donnée actuelle d'un capteur n'est pas toujours suffisant pour rendre compte correctement de l'état d'un des dispositifs de la ferme. Parfois, il est nécessaire de visualiser les valeurs qui ont précédé pour pouvoir observer une tendance et détecter un potentiel dysfonctionnement.

C'est pourquoi, nous avons décidé d'implémenter des courbes retraçant l'historique des dernières valeurs des capteurs. Unity ne dispose pas d'API permettant de réaliser des graphiques ou autres charts, nous avons donc implémenter nous-mêmes un graphique, pour ce faire, nous avons suivi un [tutoriel](#) très détaillé du créateur de contenu Code Monkey. Après diverses modifications apportées au tutoriel pour adapter l'échelle du graphique automatiquement notamment, nous avons lié les graphiques à la base de données.



Figure 35 4: Evolution de la température de la serre

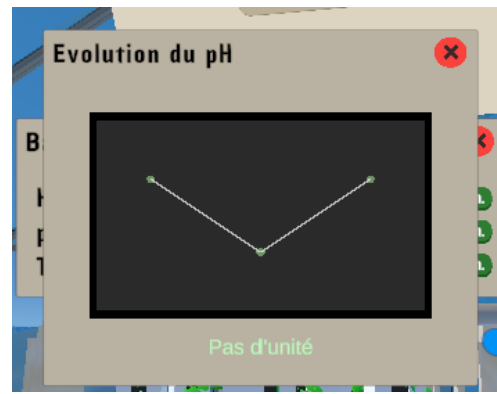


Figure 36 5: Evolution du pH de la serre

SYSTEME METEO ET CYCLE JOUR/NUIT EN TEMPS REEL

Par souci de réalisme, nous avons opté pour un rendu du ciel réaliste. Pour ceci, nous nous sommes munis d'une API météo pour pouvoir refléter au mieux le climat en temps réel sur le jumeau numérique.

Tout d'abord, nous avons créé un client en C# pour envoyer des requêtes HTTP à l'API ainsi que des classes représentant les différents JSON pouvant être réceptionnés.

Ci-dessous, le diagramme de classe pour faire le parsing du JSON en classe C# :

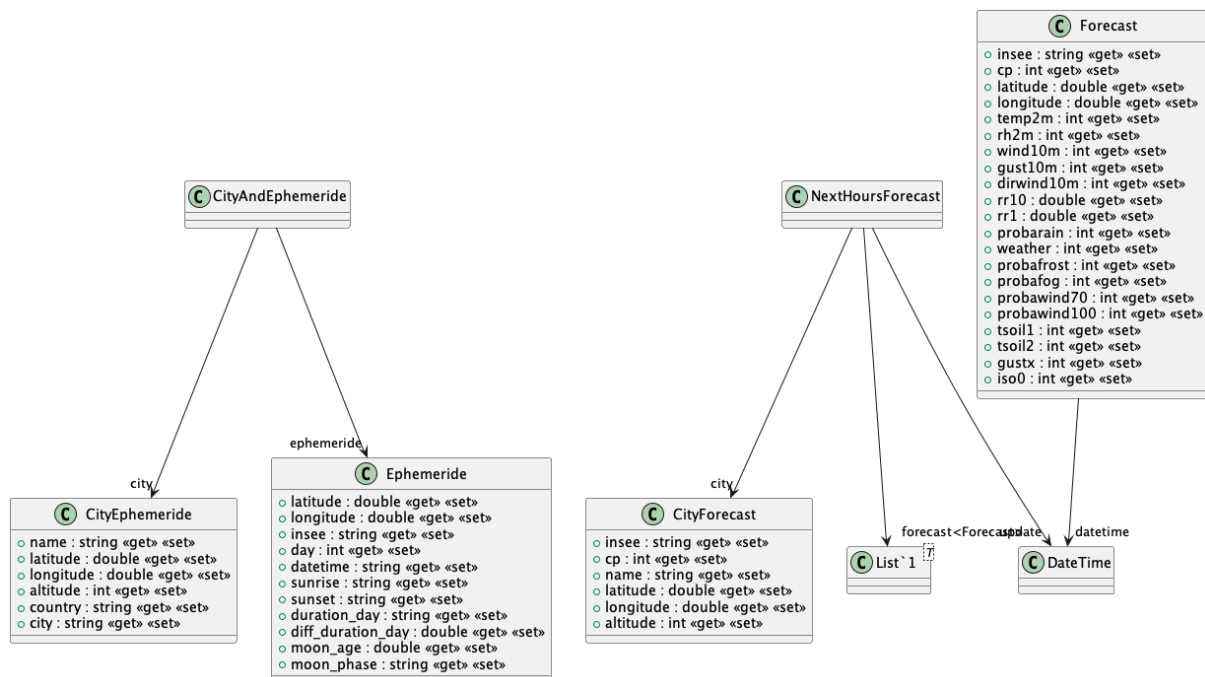


Figure 37 6 : Diagramme de Classe de MeteoAPIResponse

La classe CityAndEphemeride permet de récupérer l'éphéméride pour une ville donnée, quant à NextHoursForecast, cette classe nous permet de récupérer les prévisions météo pour les 12

prochaines heures dans une ville donnée. Pour ce qui est du client, la classe comprend une méthode qui permet de retourner une réponse en tant que chaîne de caractères à partir d'une requête donnée en paramètres :

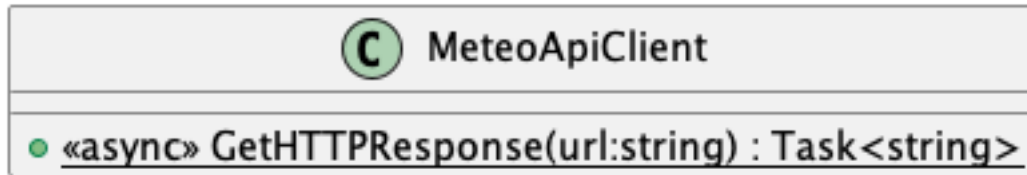


Figure 387 : Diagramme de classe de MeteoAPIClient

En effet, à chaque lancement de l'application, l'application envoie une requête à l'API Météo Concept pour recevoir l'éphéméride du jour à Badevel, cette information nous permet de connaître la durée du jour, le lever et coucher du soleil. À l'aide de cette information, on peut ainsi déplacer le soleil en temps réel pour pouvoir refléter l'ensoleillement réel, on obtient ainsi un cycle jour/nuit proche du réel. Le diagramme de séquence ci-dessous décrit le fonctionnement :

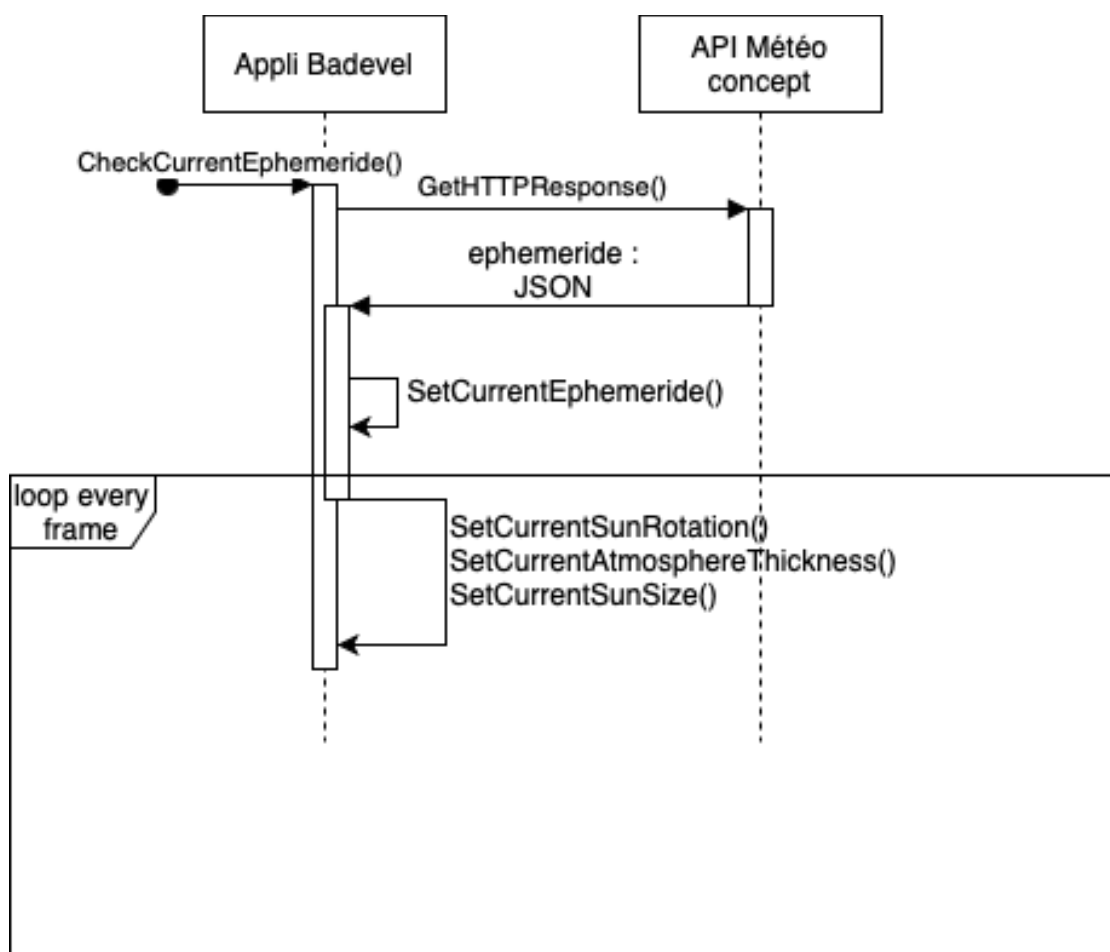


Figure 398 : Diagramme de séquence de DynamicSkybox

Le diagramme ci-dessous représente les différentes classes utilisées pour cette fonctionnalité :

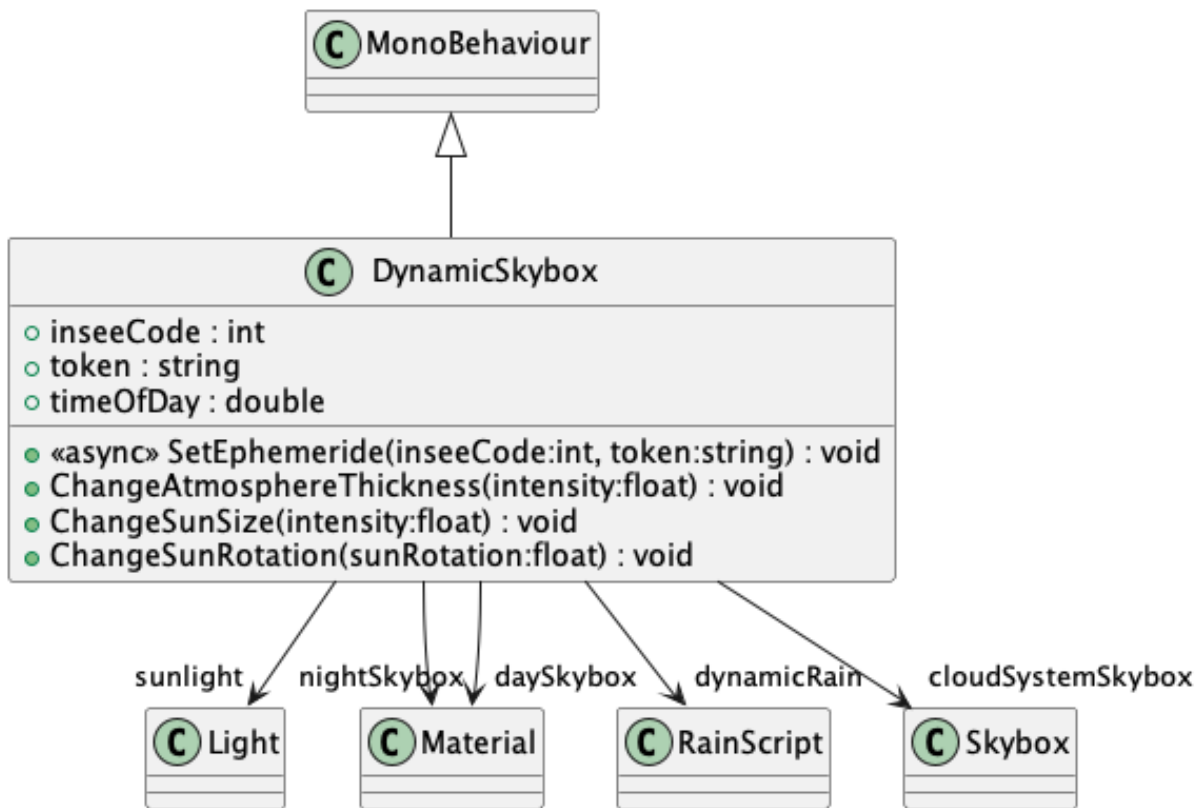


Figure 40 9 : Diagramme de Classe de DynamicSkybox

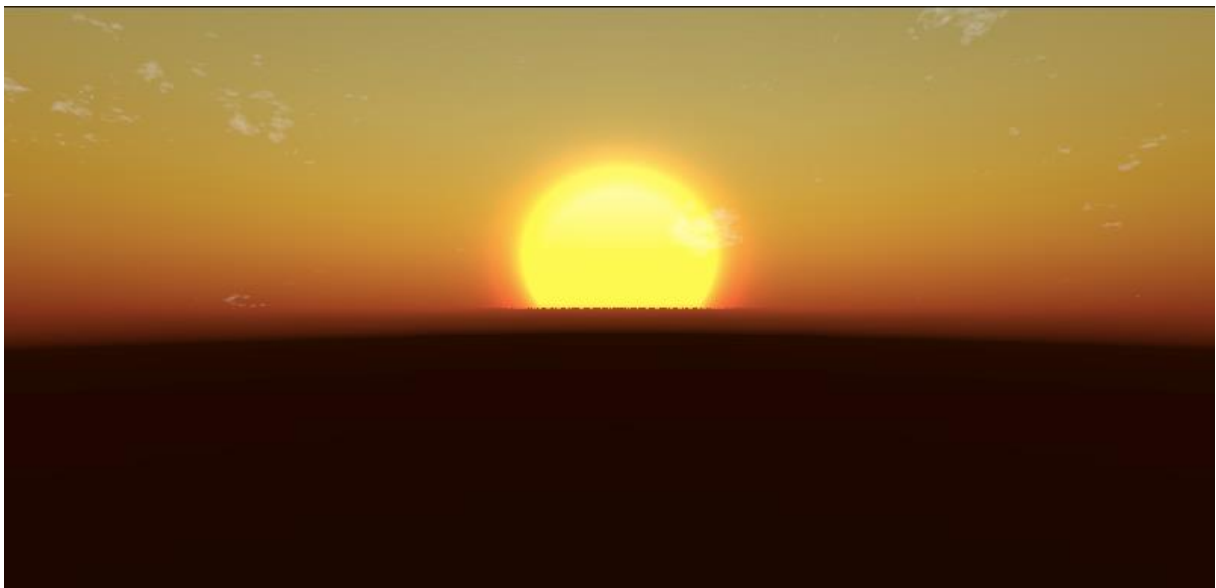


Figure 41 10 : Exemple de lever de soleil



Figure 42 11 : Exemple de coucher de soleil



Figure 43 12 : Exemple d'un milieu de journée

Pour pouvoir refléter la météo actuelle, il a été décidé d'utiliser l'API Météo Concept pour connaître les prévisions météo. À chaque heure, l'application envoie une requête à l'API pour recevoir les dernières prévisions météo, à l'aide de cette information, on peut ainsi savoir si le temps est nuageux, s'il pleut, ou alors s'il y a un orage. L'étape suivante est de relier ces informations au jumeau numérique pour les représenter au mieux. Pour cela, on utilise différentes skybox de ciel gris pour représenter les nuages et un asset gratuit RainMaker, qui permet de simuler de la pluie en temps réel, cet asset est hautement modulaire, en effet, on peut appliquer un effet de vent sur la pluie, de brouillard, on peut aussi régler l'intensité de la pluie. Ainsi, à chaque fois que l'API signale un temps pluvieux ou orageux, l'intensité de la pluie et des nuages va changer en conséquence sur le jumeau numérique, et ceci heure par heure.



Figure 44 13 : Exemple de pluie faible intensité



Figure 45 14 : Exemple de pluie moyenne intensité



Figure 46 15 : Exemple de pluie haute intensité

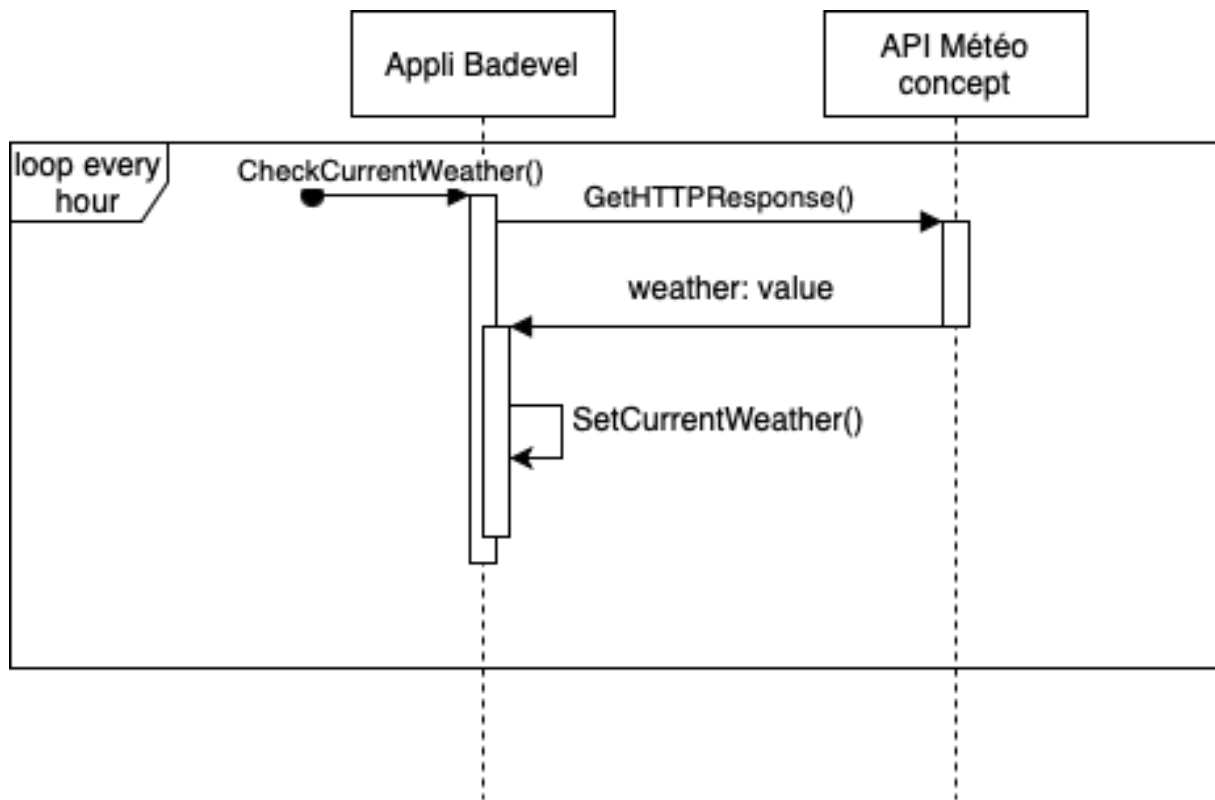


Figure 47 16 : Diagramme de séquence DynamicRainfall

Le diagramme ci-dessous représente les différentes classes utilisées pour cette fonctionnalité :

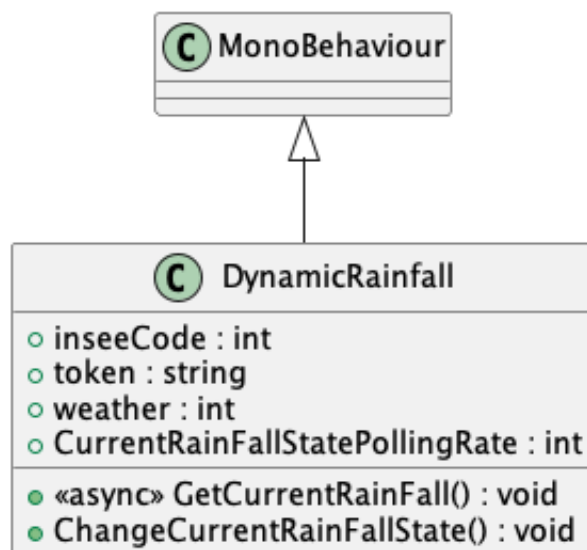


Figure 48 17 : Diagramme de Classe de DynamicRainfall

Pour ajouter une illusion de mouvement de nuage dans la Skybox, un asset gratuit SimpleCloudSystem a été ajouté. Il s'agit d'un shader placé sur un dome invisible et qui permet de simuler des nuages en mouvement tout autour du jumeau numérique.

CONCLUSION

Tout au long du semestre, nous avons avancé sur le projet dans l'objectif d'aider les bénévoles de la ferme de Badevel. Nous avons créé cette application de A à Z en nous inspirant des logiciels réalisés par Vouse et Sitowise. En utilisant les outils que nous avons vus en cours, nous sommes arrivés à un résultat satisfaisant. La répartition du projet n'a pas vraiment suivi le rythme des cours (nous avons dû commencer par ce que nous avons vu en toute fin de semestre.) nous avons donc dû apprendre en grande majorité en autodidacte. Le résultat est à la hauteur de nos espérances.

Le déroulement du projet ne s'est cependant pas passé sans encombre. L'ordre dans lequel nous avons réalisés les différentes fonctionnalités de la ferme était peu adapté à la réalisation d'un projet informatique compte tenu de nos connaissances. Nous aurions préféré commencer par la réalisation des scripts et de la base de données plutôt que la modélisation, domaine dans lequel nous avons peu d'expertise en début de semestre. Nous avons également rencontré des problèmes avec notre système de contrôle de versions. En effet, nous avons utilisé le VCS git avec la forge Github et le plugin git LFS permettant de gérer des fichiers de données plus conséquents. Github n'autorise malheureusement pas de larges bandes passantes sur la récupération et l'envoi de données à leurs serveurs. Il nous était impossible d'envoyer ou de recevoir des données sur la fin du projet, nous avons donc dû mettre en forme le projet final sur un seul ordinateur. Nous aurions dû utiliser un système tel que Plastic VCS ou un Gitlab hébergé localement.

Finalement, nous avons réussi à surpasser ces difficultés pour réaliser un logiciel correspondant à notre cahier des charges. Notre logiciel place déjà les fondations d'un logiciel plus ambitieux lié aux véritables données de la ferme de Badevel.

