|  |
| --- |
| Météo  Préparation au travail pratique individuel 2023 |

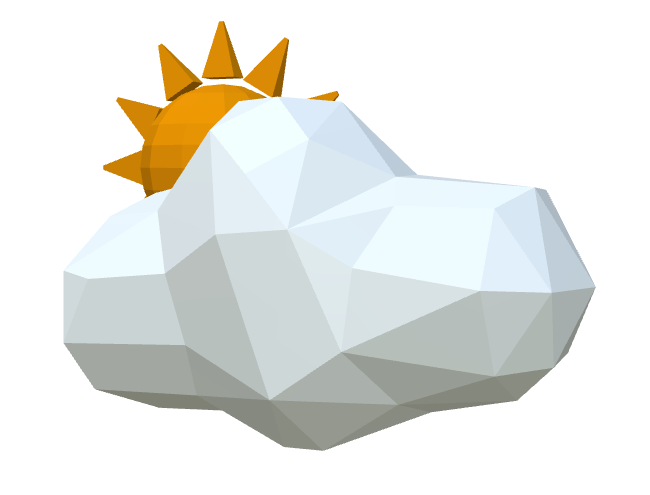


Figure 1 : Image d'un modèle 3D de nuage.

Candidat : Mikael Juillet

Chef de Projet : Benzonana Pascal

Table des matières

[1 Analyse préliminaire 3](#_Toc130892227)

[1.1 Introduction 3](#_Toc130892228)

[1.2 Cahier des charges 3](#_Toc130892229)

[1.1.1 Sujet 3](#_Toc130892230)

[1.1.2 Matériel et logiciel à disposition 3](#_Toc130892231)

[1.1.3 Prérequis 3](#_Toc130892232)

[1.1.4 Descriptif du projet 4](#_Toc130892233)

[1.1.5 Points technique évalués 5](#_Toc130892234)

[1.2 Objectifs 5](#_Toc130892235)

[1.3 Planification initiale 6](#_Toc130892236)

[2 Analyse / Conception 7](#_Toc130892237)

[2.1 Concept 7](#_Toc130892238)

[2.2 Données en fonction du temps 8](#_Toc130892239)

[2.4 Stratégie de test 9](#_Toc130892240)

[2.5 Risques techniques 9](#_Toc130892241)

[2.6 Planification 9](#_Toc130892242)

[2.7 Dossier de conception 10](#_Toc130892243)

[2.7.1 Logiciels / Framework utilisé : 10](#_Toc130892244)

[3 Réalisation 11](#_Toc130892245)

[3.1 Répertoires 11](#_Toc130892246)

[3.2 Wireframe et Mockups 12](#_Toc130892247)

[3.3 Création des modèles 3D 15](#_Toc130892248)

[3.4 Affichage des modèles 16](#_Toc130892249)

[3.5 Récupération des données météo 18](#_Toc130892250)

[3.5.1 Choix de l’api 18](#_Toc130892251)

[3.5.2 Création des requêtes 19](#_Toc130892252)

[3.5.3 Recherches 19](#_Toc130892253)

[3.6 Filtrage des données météo 20](#_Toc130892254)

[3.7 Animations des modèles 22](#_Toc130892255)

[3.8 Logo du site 23](#_Toc130892256)

[3.9 Publication 23](#_Toc130892257)

[3.10 Description des tests effectués 23](#_Toc130892258)

[3.11 Erreurs restantes 24](#_Toc130892259)

[3.12 Liste des documents fournis 24](#_Toc130892260)

[4 Conclusions 24](#_Toc130892261)

[4.1 Résumé 24](#_Toc130892262)

[5 Annexes 25](#_Toc130892263)

[5.1 Journal de travail 25](#_Toc130892264)

[5.2 Journal de bord 25](#_Toc130892265)

[5.3 Cahier des charges 25](#_Toc130892266)

[6 Table de illustrations 26](#_Toc130892267)

[7 Bibliographie 26](#_Toc130892268)

[8 Lexique 28](#_Toc130892269)

# Analyse préliminaire

## Introduction

Ce projet est réalisé dans le cadre de la préparation au travail pratique individuel (TPI) qui sera effectué à la suite de celui-ci.

Il a pour but de préparer les candidats au TPI a tous les niveaux, ainsi ce projet se déroule dans les mêmes conditions que celle prévue par le canton de Vaud. Hormis le temps de déroulement du projet qui s’effectuera sur un nombre d’heures de 67.5 à 69 heures au lieu de 90. Ainsi la préparation à la défense durera 4.5 heures au lieu d’une semaine.

Le projet consiste en la création d’une page web contenant les prévisions sur 7 jours d’une zone géographique choisie par l’utilisateur. Des modèles 3D représentant les conditions météorologiques sont affichés sur la page.

Ce thème a été choisi car il contient les technologies similaires à celles qui seront utilisées durant le TPI.

## Cahier des charges

### Sujet

Le projet consiste en la création d’une page web contenant les prévisions de 7 jours d’une zone géographique choisie par l’utilisateur. Des modèles 3D représentant les conditions météorologiques sont affichés sur la page.

### Matériel et logiciel à disposition

1 ordinateur type CPNV

* Windows 10
* PHPStorm ou Visual Studio
* Microsoft Office 2016
* Adobe Photoshop 2017
* Balsamiq Mokup
* Apache et MySQL

### Prérequis

Le candidat a une bonne connaissance dans la technologie Web et de développement d’applications.

### Descriptif du projet

Une page web affiche la météo des 7 jours à venir en partant du jour actuel, l’utilisateur peut choisir la ville de son choix via une barre de recherche.

La météo est ensuite récupérée via une API.

Sur la page, plusieurs informations sont affichées : premièrement la météo d’un jour est affichée en grand en haut de la page, affichant un model 3D représentant par défaut la météo du jour.

Cette partie contiendra des données tel que :

* Vent moyen
* Direction du vent
* Cumul de pluie sur la journée en mm
* Probabilité de pluie entre 0 et 100%
* Température minimale
* Température maximale
* Ensoleillement en heures
* Heure du lever du soleil
* Heure du coucher du soleil
* Durée du jour en heure et minutes

Une animation et une interaction est disponible au niveau du modèle 3D représentant la météo du jour, l’animation correspond à un vas et viens sur l’axe Z afin d’imiter une flottaison de l’élément sur la page. Une interaction se fera lorsque l’utilisateur cliquera sur n’importe quel modèle 3D, il effectuera une rotation à 360° avant de revenir à son emplacement d’origine.

En dessous de cette partie un récapitulatif affichera en petit les modèles 3D représentant la météo des 7 jours comprenant le jour même, les températures minimum et maximum du jour sont affichées en dessous du modèle 3D correspondant.

Lorsqu’un utilisateur clique sur un jour dans le récapitulatif des jours, le jour cliqué va remplacer les informations météorologiques du jour affichées en haut de la page et indiquer les informations correspondantes au jour cliqué.

Les modèles 3D sont la représentation des différents aspects de la météo :

* Un nuage = nuageux
* Un soleil = ensoleillé
* Un nuage et un soleil = nuageux et ensoleillé
* Un nuage et un éclair = orage ou possibilité d’éclairs
* Un nuage et de la pluie = pluvieux
* Un nuage et de la neige = enneigé

### Points technique évalués

La grille d’évaluation définit les critères généraux selon lesquels le travail du candidat sera évalué (documentation, journal de travail, respect des normes, qualité, …).

En plus de cela, le travail sera évalué sur les 7 points spécifiques suivants (Point A14 à A20) :

1. Les données météos sont récupérées via une api.
2. Il est possible de choisir la ville sur laquelle la météo va être affichée.
3. Les modèles 3D s’affichent.
4. Les interactions et l’animation fonctionnent.
5. Ergonomie de l’interface : elle respecte les standards vus en module I-CT 120.
6. Explication pour le déploiement de la web application sur un serveur
7. Description et qualité des tests effectués

## Objectifs

L’objectif de ce projet est de crée une page web contenant les informations météorologiques, mais aussi de prendre en main les technologies telles que Three.js et Ajax.

Tout au long de mon travail je vais me conformer aux critères d’évaluation établis par le canton de Vaud (Schwab, 2018).

## Planification initiale

Ci-dessous vous trouverez la planification initiale sous forme de méthodologie Waterfall (Wikipedia, 22), j’ai choisi ce format car c’est cette forme qui convient mieux pour des projets à durée déterminée contenant des parties définies au préalable. La création de ce diagramme de Gantt à été effectué sur la plateforme Wrike (wrike, -).

Cependant il nous est demandé d’utiliser une méthode agile, Scrum (claude aubry, 21) pour la réalisation du projet, ce qui sera effectué à l’aide du logiciel IceScrum (cpnv.ch, -).

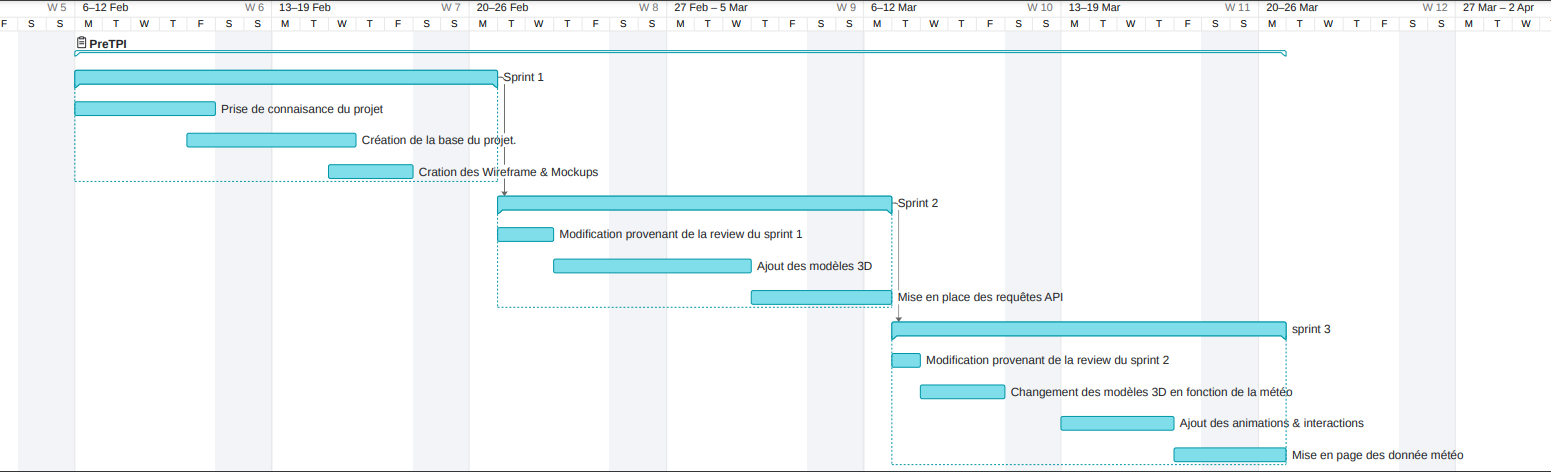


Figure 2 : Capture d'écran du diagramme de Gantt pour la planification initiale (méthode Waterfall).

*Figure SEQ Figure \\* ARABIC 1 : Capture d'écran du diagramme de Gantt pour la planification initiale (méthode Waterfall).*

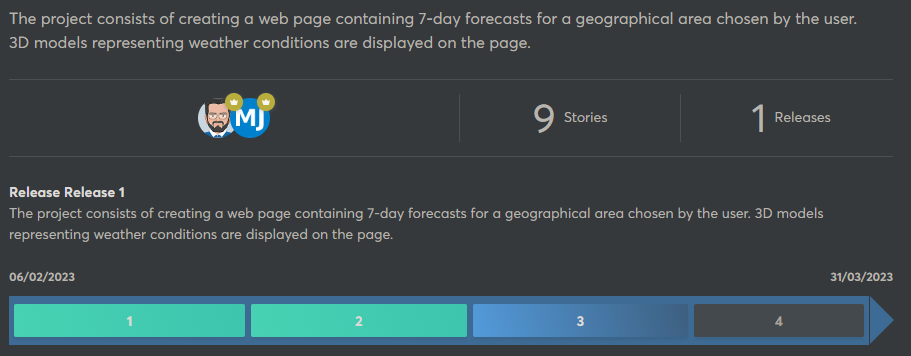
**

Figure 3 : Capture d'écran du planning du projet (méthode Scrum).

# Analyse / Conception

## Concept

Ce site est conçu sur une unique page, elle affiche la météo des 7 jours à venir en partant du jour actuel, l’utilisateur peut choisir la ville de son choix via une barre de recherche.

La météo est ensuite récupérée via une API.

Ci-dessous nous avons les éléments essentiels à récupérer de l’API :

* Vent moyen
* Direction du vent
* Cumul de pluie sur la journée en mm
* Probabilité de pluie entre 0 et 100%
* Température minimale
* Température maximale
* Ensoleillement en heures
* Heure du lever du soleil
* Heure du coucher du soleil
* Durée du jour en heure et minutes

Liste des animations à prévoir :

* Va et vient sur l’axe Z afin d’imiter une flottaison de l’élément sur la page.
* Rotation à 360° avant de revenir à son emplacement d’origine.

Les modèles 3D sont la représentation des différents aspects de la météo :

* Un nuage = nuageux
* Un soleil = ensoleillé
* Un nuage et un soleil = nuageux et ensoleillé
* Un nuage et un éclair = orage ou possibilité d’éclairs
* Un nuage et de la pluie = pluvieux
* Un nuage et de la neige = enneigé

## Données en fonction du temps

Voici les caractéristiques qui vont définir quel temps il va faire.

* Si total de quantité de neige est plus grand que 0cm, météo neigeuse.
* Si probabilité de pluie plus grande que 50%, météo pluvieuse.
* Si probabilité de tonnerre plus grande que 50%, météo orageuse.
* Si calculs heure complètes de la journée divisée durée du soleil plus grand ou égale à 70%, météo ensoleillée.
* Si calculs heure complètes de la journée divisée durée du soleil plus petit que 30%, météo nuageuse.
* Si calculs heure complètes de la journée divisée durée du soleil plus grand que 30% mais plus petit que 70%, météo nuageuse mais ensoleillée.

Voici les informations qui sont nécessaire de récupérer afin d’afficher les informations nécessaires et de calculer la météo.

|  |
| --- |
| Jours |
| Cumul de pluie sur la journée en mm |
| Direction du vent |
| Durée du jour en heure et minutes |
| Ensoleillement en heures |
| Heure du coucher du soleil |
| Heure du lever du soleil |
| Météo globale |
| Probabilité de pluie entre 0 et 100% |
| Probabilité de tonnerre |
| Température maximale |
| Température minimale |
| Total de neige en mm |
| Vent moyen |

## Stratégie de test

Etant donné la durée du projet, je n’aurai pas de temps pour faire des tests unitaires ou de fonctionnement. Cependant je ferais des tests manuels en impliquant le plus possible de possibilité.

## Risques techniques

Les risques techniques sont mon manque de connaissance à l’utilisation de Three.js (three.js, 2023) et Ajax (W3schools, -). Cependant je vais pallier ce manque de connaissance en prenant plus de temps sur ces parties ainsi qu’en créant un projet plus petit.

## Planification

J’ai effectué de multiples modifications afin de prendre en compte les Vacances scolaire de février, de plus la date finale a été corrigée par la suite.

J’ai aussi ajouté des taches plus précises car certaines ont été oubliées lors de la création de la première planification.

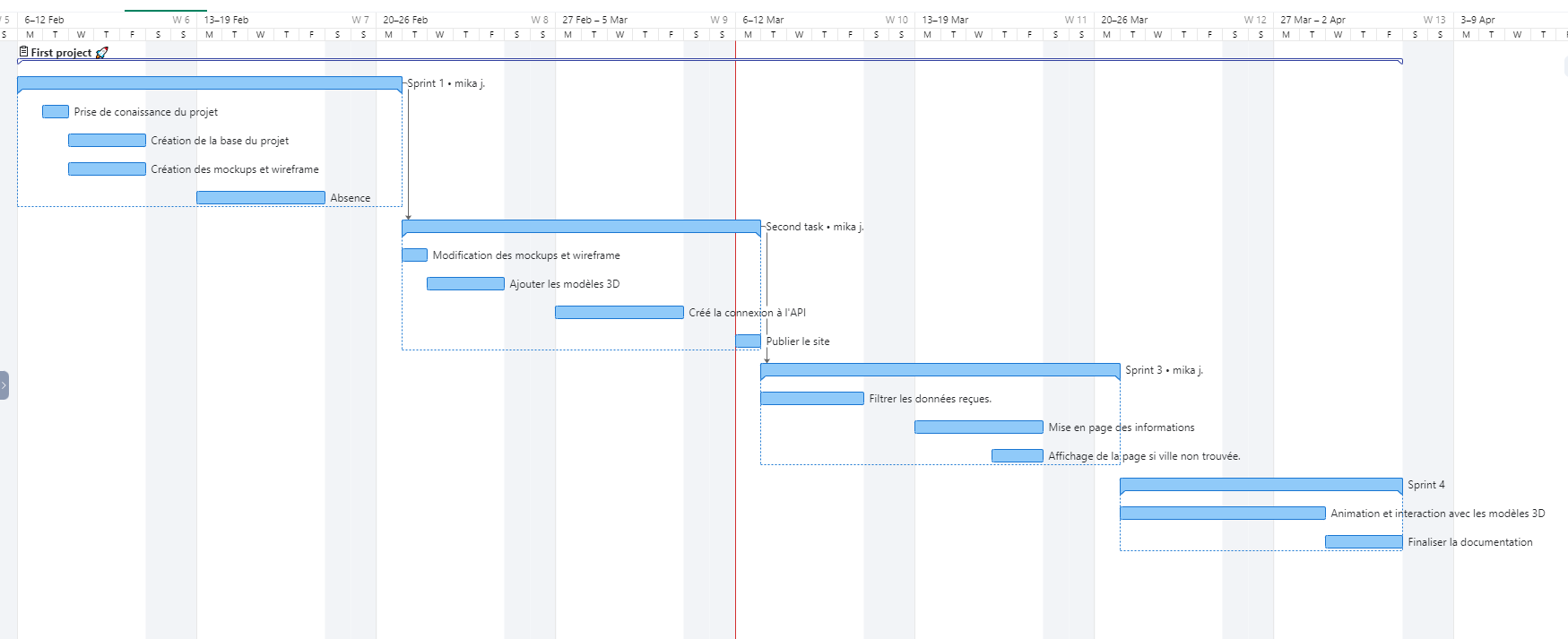


Figure 4 : Capture d'écran du diagramme de Gantt pour la planification finale (méthode Waterfall).

## Dossier de conception

### Logiciels / Framework utilisé :

|  |  |  |
| --- | --- | --- |
| **Nom** | **Version** | **Utilisation** |
| Visual Studio Code (visualstudio, 2023) | 1.74.3 | Editeur de code |
| Balsamiq Wireframe (balsamiq, 2023) | 4.6.5 | Wireframe |
| Figma (figma, 2023) | - | Mockup |
| HTML, CSS | Html 5, CSS 3 | Mise en page |
| Vite.js (vitejs, 2023) | 4.1.1 | Frontend Tooling |
| Vue.js (vuejs, 2023) | v3.3.0-alpha.4 | JavaScript Framework |
| Three.js (WebGL) (three.js, 2023) | 0.149.0 | Rendu 3D |
| Ajax (W3schools, -) |  | Requêtes |
| moment (momentjs, 2023) | 2.29.4 | Gestion des dates |
| dayjs (day.js, 2023) | 1.11.7 | Gestion des dates |
| Pinia (pinia, 2023) | 2.0.33 | Stockage |

# Réalisation

## Répertoires

Dépôts GitHub : <https://github.com/Juillet-Mikael/PreTPI23>

Documentation Google Drive :

<https://drive.google.com/drive/folders/1mftKBXD664VLmlqZODGvQzSIDgTccfL8?usp=share_link>

Documentation se situe aussi dans un dossier nommé doc au sein du projet Git.

**Architecture des documents :**

* PreTPI23
  + Journal de travail
  + Journal de bord
  + Dossier de Projet
  + *consignes*
    - Contient les documents en rapport au consignes tell que le cahier des charges et la liste des critères d’évaluation.
  + *projet*
    - maquettes
      * Contient les fichiers de conception des maquettes.
    - diagrammes
      * Contient les documents rapportant a des diagrammes.
    - modèles 3d
      * Contient l’ensemble des modèles 3d récupéré.

## Wireframe et Mockups

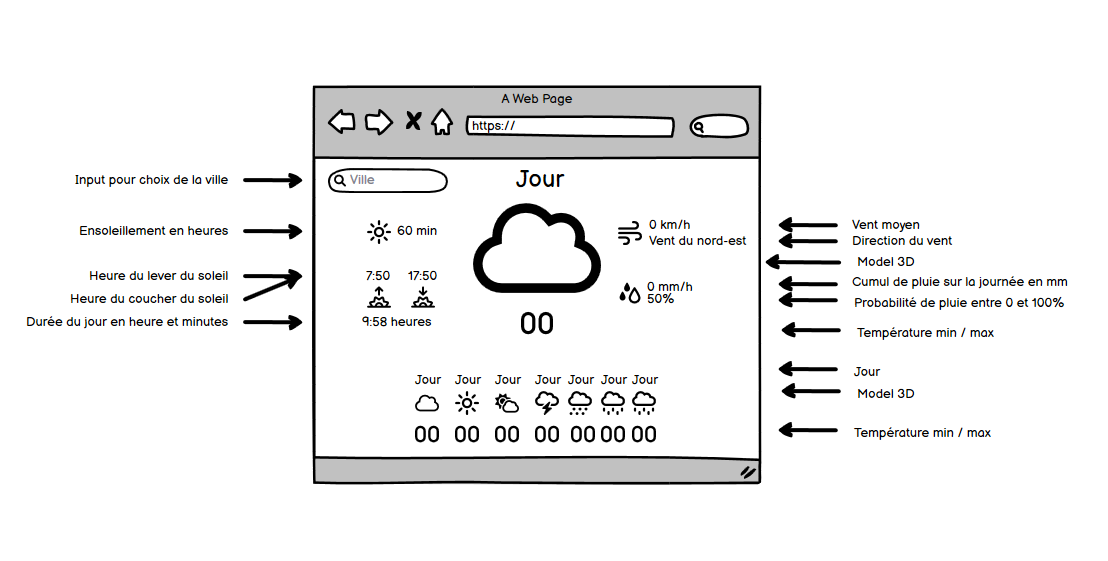


Figure 5 : : Wireframe desktop ville trouvée

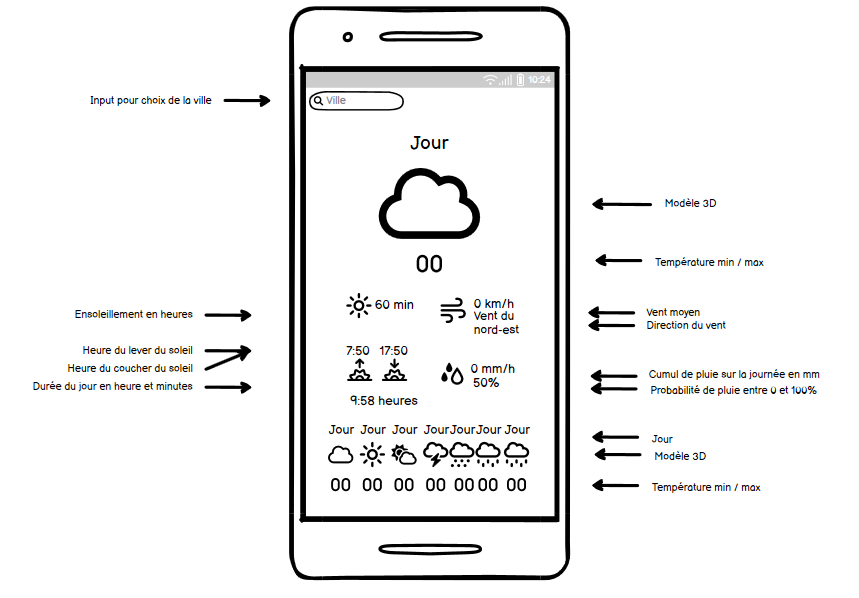


Figure 6 : Wireframe phone ville trouvée.

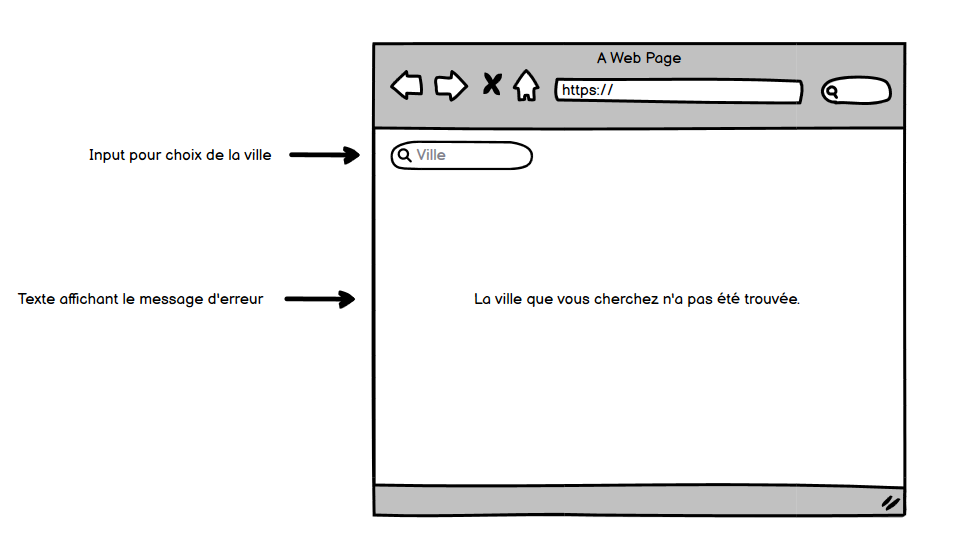


Figure 7 : Wireframe desktop ville non trouvée.

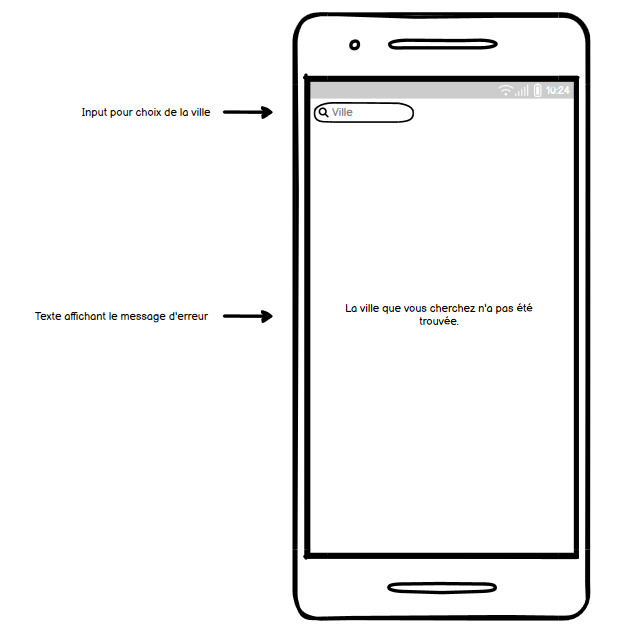
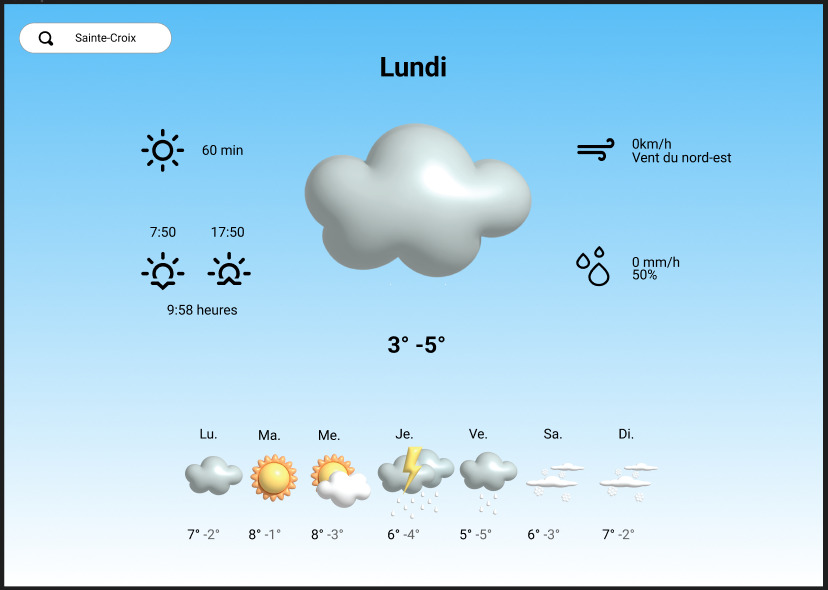


Figure 8 : Wireframe phone ville non trouvée.

Une image contenant graphique, diagramme

Description générée automatiquement

Figure 9 : Wireframe desktop et phone ville trouvée.



Figure 10 : Wireframe desktop et phone ville non trouvée

## Création des modèles 3D

J’ai récupéré sur le site de sketchfab.com (sketchfab, 2023) des modèles 3D près existant.

Vous trouverez ci-dessous la liste des modèles téléchargés, ainsi que les modèles que j’ai moi-même modifiés depuis les modèles téléchargés avec le logiciel Blender (blender, 2023).

|  |  |
| --- | --- |
| Basic Low Poly Cloud par Amedhh (Amedhh, 2021) | Figure 11 : Modèle représentant la pluie. |
| Thunder par amit (amit, 2022) | Figure 12 : Modèle représentant une éclaire |
| Modèle utilisant les modèles de Thunder par amit ainsi que Basic Low Poly Cloud par Amedhh. | Figure 13 : Modèle représentant un orage. |
| Sun from Poly by Google par IronEqual (IronEqual, 2021) | Figure 14 : Modèle représentant un soleil. |
| Cloud par Jiří Kuba (Kuba, 2017) | Figure 15 : : Modèle représentant un nuage |
| Modèle utilisant les modèles de Cloud par Jiří Kuba ainsi que Sun from Poly by Google par IronEqua. | Figure 16 :Modèle représentant un nuage et un soleil. |
| Modèle utilisant le modèle Cloud par Jiří Kuba.  J’ai ajouté des sphères blanches afin d’y faire de la neige. | Figure 17 : Modèle représentant la neige |

## Affichage des modèles

Je mis suis repris plusieurs fois sur la création de la scène Three.js, ma première idée était de mettre en place une page complète en three.js et de placer les modèles à des places précises. Lors de la mise en place des indications de la météo, j’ai remarqué que cette méthode me posera plus d’inconvénients que d’avantages, j’ai alors opté pour créer une scène par modèle ce qui me permet de les placer de façon à ce quelle face directement la taille voulue et qu’elle s’adapte en fonction du CSS crée.

Sur l’image ci-dessous le carré noir représente la scène, si j’étais rester sur l’option une.

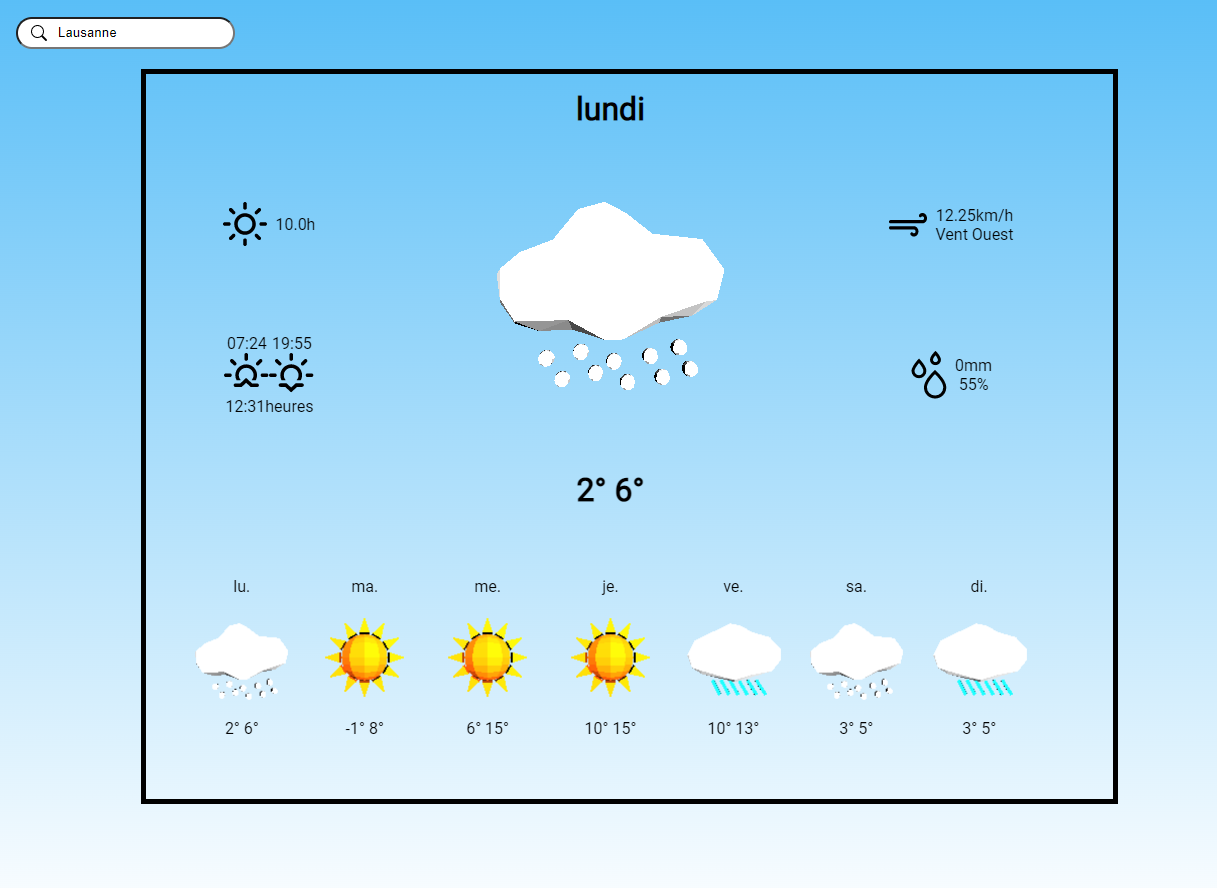


Figure 18 : Exemple d'une scène complète.

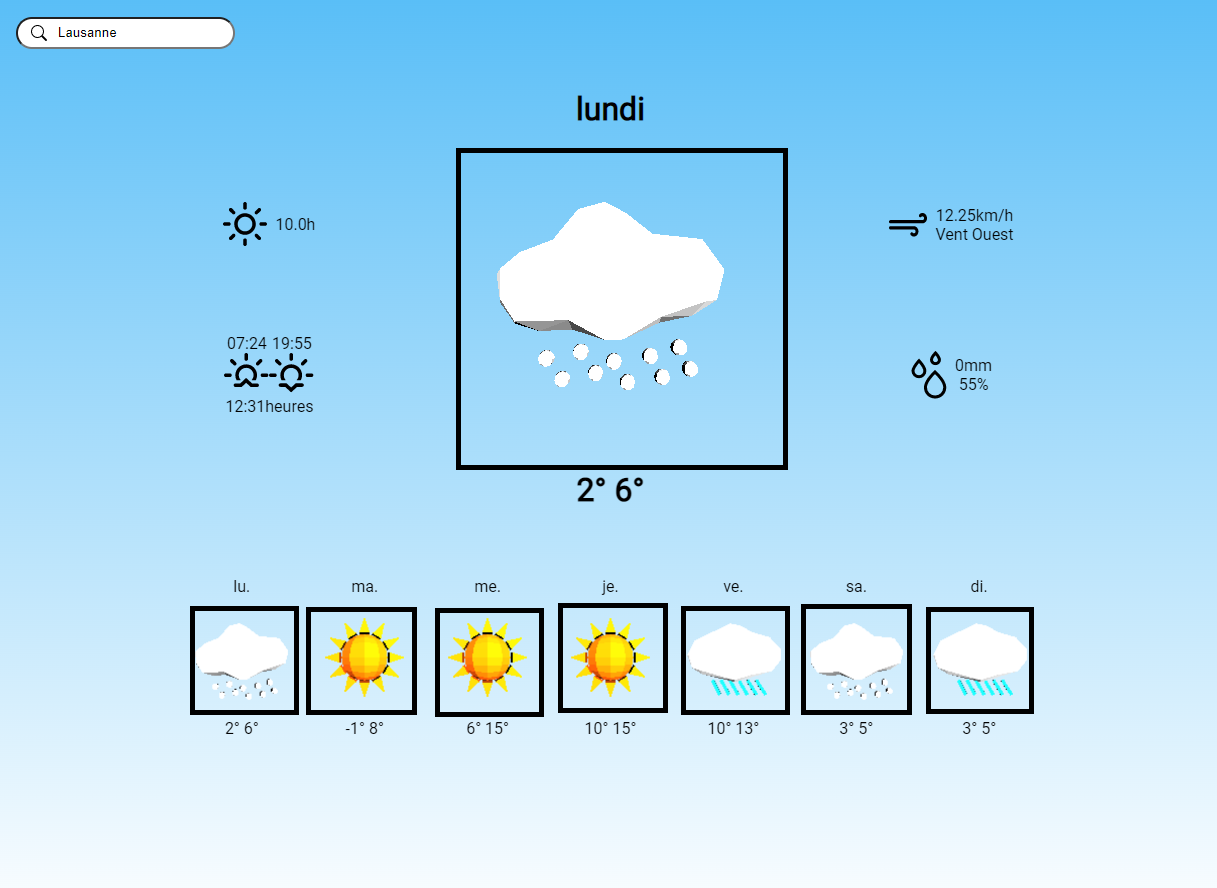


Figure 19 : Exemple d'une page avec de multiples scènes.

Afin que ma page puisse réagir de façon responsive, il a fallu que j’inclue une fonction qui s’occupe de changer la taille du canvas selon la taille définie par la dive qui elle-même change de façon responsive via le css. Je me suis inspiré du tutoriel suivant pour le responsive : <https://www.youtube.com/watch?v=A00WHN32qGE> (Coders, 2022)

J’ai créé un fichier qui vas générer automatiquement les canvas et modèle qui s’appelle MeteoRenderer.

Dans le documents ci-dessous nous voyons que j’appelle en passant le nom du modèle 3d ainsi qu’un index qui vas permettre d’identifier chaque canavas.



Figure 20 : Capture d'écrans de la partie template du fichier WeatherWeek.

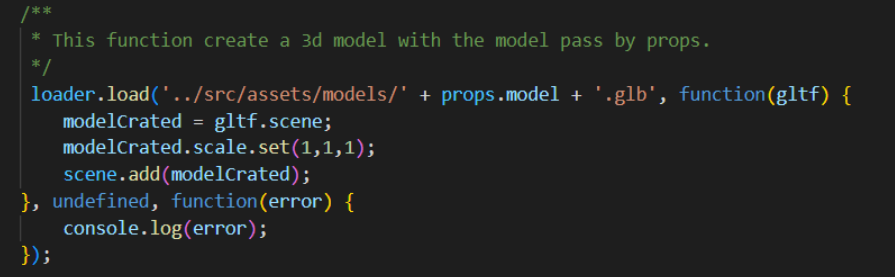


Figure 21 : Représente la fonction qui crée le modèle 3D.

## Récupération des données météo

### Choix de l’api

Il existe de nombreuses sources proposant des services d’api de météo, cependant j’ai besoin des informations suivantes de façon gratuite.

* Vent moyen
* Direction du vent
* Cumul de pluie sur la journée en mm
* Probabilité de pluie entre 0 et 100%
* Température minimale
* Température maximale
* Ensoleillement en heures
* Heure du lever du soleil
* Heure du coucher du soleil
* Durée du jour en heure et minutes

Le problème que j’ai eu est de trouver une API qui soit gratuite et qui me procure les données que je souhaite sur sept jours. J’ai trouvé l’API “[World Weather API](https://www.worldweatheronline.com/weather-api/api/)” (worldweatheronline, 2023) qui offre toutes les données voulues sur 14 jours. Malheureusement l'utilisation de cette api est payante mais elle propose un accès de d’essai de 30 jours avec un maximum de 500 requêtes par jours ce qui devrait convenir jusqu’à la fin du pré-TPI, cependant si je devais créer un tel logiciel pour une entreprise, je proposerais une autre api payante qui correspond mieux aux besoins.

### Création des requêtes

Une image contenant texte

Description générée automatiquement

Figure 22 :Capture d'écran d’une requête avec postman.

J’ai créé un modèle qui va faire une requête Get vers l’api spécifique pour le type Local Weather API qui va me permettre de récupérer toutes les données nécessaires.

J’utilise Ajax pour les requêtes n’ayant jamais utilisé Ajax je me suis basé sur la documentation de W3school (Wikipedia, 22). J’ai l’habitude d’utiliser Axios (axios, 2023) pour les requêtes mais celui-ci nécessite une installation or pour ce projet je n’ai besoin de faire qu’une seule requête et ne voulais par conséquent pas me prendre la tête et utiliser Ajax qui est pris en charge par tous les navigateurs modernes (Google Chrome, Mozilla Firefox, Apple Safari et Microsoft Edge).

Ma requête requiert de plusieurs choses :

* Méthode HTTP : GET
* Base URL : <https://api.worldweatheronline.com/premium/v1/weather.ashx>
* Paramètre q : correspondant à la location.

J’ai eu quelques problèmes sur l’itération d’un objet en JavaScript et ai trouver l’information sur :

<https://masteringjs.io/tutorials/fundamentals/foreach-object#:~:text=JavaScript's%20Array%23forEach()%20function,%2C%20or%20Object.entries()%20>.

### Recherches

Ici nous voyons que je crée un input et que je permets l’envoi via la partie @keyup.enter qui définis le faite de presser sur la touche enter.

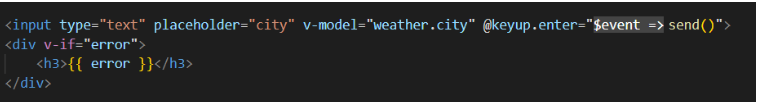


Figure 23 : Code représentant la partie recherche de la page.

## Filtrage des données météo

Pour le filtrage des données, il m’a fallu ajouter moment.js afin de traduire une durée en heure format 12 en format 24 heure.

J’ai aussi ajouter un storage avec pinia (pinia, 2023) afin que les données soient accessibles via toutes les pages.

*Une image contenant texte

Description générée automatiquement*

Figure 24 : Fonction qui permet l'ajout des donnée dans un stockage dans le fichier filterLocalWeather.js.

Les données que je retourne se forme comme sur l’image ci-dessous.

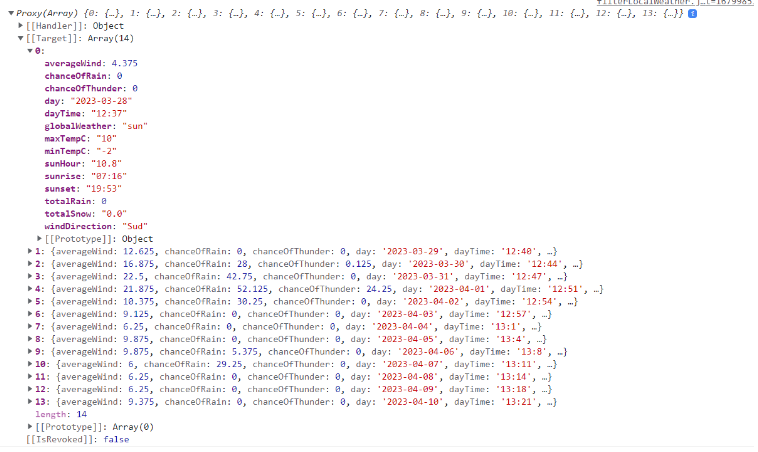


Figure 25 : Données filtrées.

Pour les villes non trouvées, je ne pouvais pas m’orienter vers un code erreur définit car l’api retourne un résultat sous forme de texte.

Voici le retour de la valeur non trouvée, retournée par l’api.

<data>

<error>

<msg>

Unable to find any matching weather location to the query submitted!

</msg>

</error>

</data>

Je vais filtrer cette réponse dans mon contrôleur afin de traduire le texte en français.

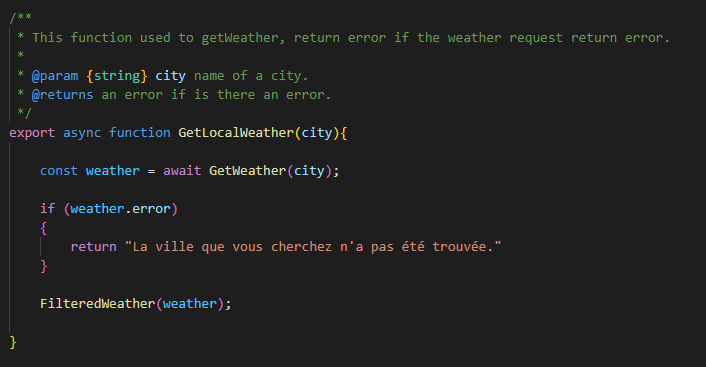


Figure 26 : Fonction GetLocalWeather du contrôleur requests.js

## Animations des modèles

L’animation de me modèles s’effectuent à l’intérieur de mon fichier MeteoRenderer.vue.

Pour l’animation de rotation à 360° j’attends un event de clic sur le model qui vas mettre une variable sur la valeur vrais.

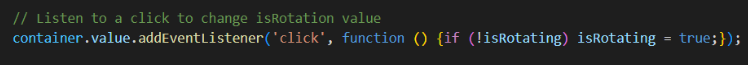


Figure 27 : Event listener, MeteoRenderer.vue.

Puis je crée une rotation qui vas s’arrêter lors ce que pi \* 2 qui est égale à la circonférence de 360° lors de la rotation.

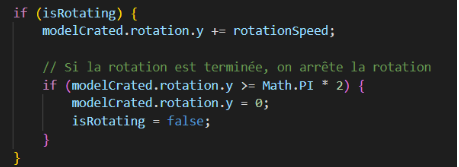


Figure 28 : Action de rotation du modèle 3D.

Pour l’animation de haut est bas je crée un calcul qui se base sur le sinus du temps.

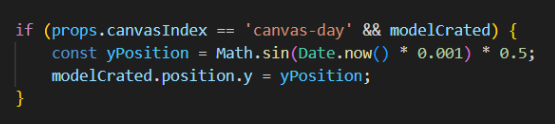


Figure 29 : Animation de haut et bas

## Logo du site

Pour le logo du site j’ai simplement fait une capture d’écran sur puis j’ai supprimer le fond.

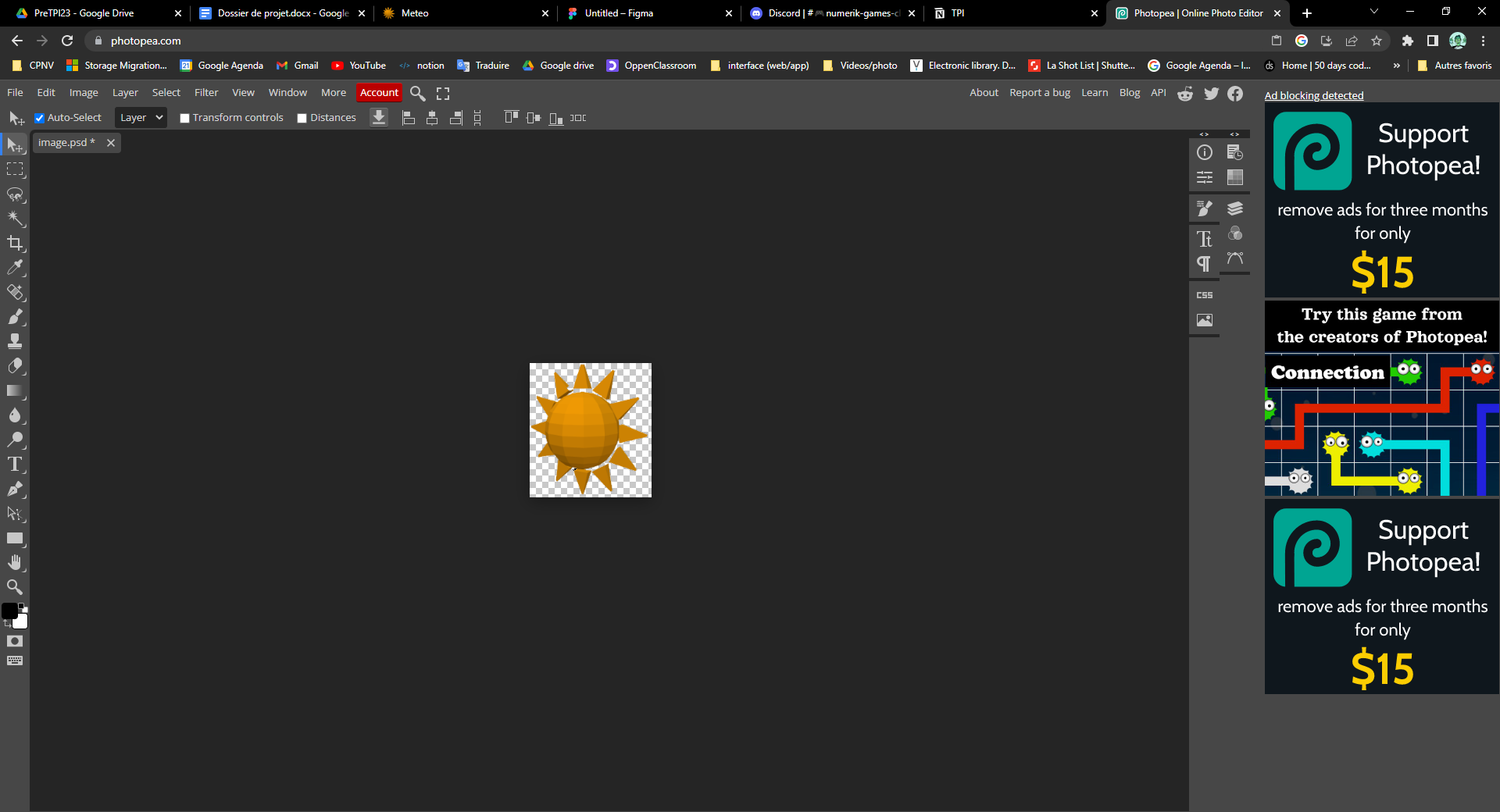
**

Figure 30 : Enlèvement du fond sur photopea.

## Publication

La publication s’effectue sur le site de swisscenter (swisscenter, 2023) via un accès FTP.

## Description des tests effectués

| **Nom** | **Résultats attendu** | **Résultats** | **Commentaires** |
| --- | --- | --- | --- |
| Un utilisateur fait une recherche de la ville Sainte-Croix. | Les modèles se chargent et changent en fonction de la météo reçue, les données sont affichées conformément aux mockups. | Réussi |  |
| Un utilisateur cherche la ville x. | Le site affiche le message suivant : La ville que vous cherchez n'a pas été trouvée. | Réussi |  |

## Erreurs restantes

Je n’ai pu apercevoir sur la dernière version aucune erreur.

## Liste des documents fournis

Tous les documents ont été mis à jour le 28 mars 2023.

* Accès au dépôt Github
* Documentation de projet
* Journal de travail
* Journal de bord

# Conclusions

Pour ce projet il me manque le changement de modèle lors du clic sur une autre date, pour réaliser cette tâche, j’estime qu’il me faudrait entre 5 à 8 heures de travail en plus afin d’implémenter cette fonctionnalité.

Outre ce détail j’ai atteint tous les points demandés.

J’ai eu de grosses difficultés sur la partie de réalisation des canevas et animations avec three.js.

Les suites possibles du projet sont une amélioration du design du site, l’ajout de changement de modèle 3d ainsi que l’affichage du pays et de la ville cherchée.

## Résumé

Ce projet est réalisé dans le cadre de la préparation au travail pratique individuel (TPI) qui sera effectué à la suite de celui-ci.

Il a pour but de préparer les candidats au TPI à tous les niveaux, ainsi ce projet se déroule dans les mêmes conditions que celle prévue par le canton de Vaud. Hormis le temps de déroulement du projet qui s’effectuera sur un nombre d’heures de 67.5 à 69 heures au lieu de 90. Ainsi la préparation à la défense durera 4.5 heures au lieu d’une semaine.

Le projet consiste en la création d’une page web contenant les prévisions sur 7 jours d’une zone géographique choisie par l’utilisateur. Des modèles 3D représentant les conditions météorologiques sont affichés sur la page.

Les technologies principales utilisées sont three.js, vue.js et vite.

# Annexes

## Journal de travail

Fournis dans un dossier annexe.

## Journal de bord

Fournis dans un dossier annexe.

## Cahier des charges

Fournis dans un dossier annexe.

# Table de illustrations

[Figure 1 : Image d'un modèle 3D de nuage. 1](file:///C:\Users\Mikael.JUILLET\Downloads\Dossier%20de%20projet%20(3).docx#_Toc130890102)

[Figure 2 : Capture d'écran du diagramme de Gantt pour la planification initiale (méthode Waterfall). 6](file:///C:\Users\Mikael.JUILLET\Downloads\Dossier%20de%20projet%20(3).docx#_Toc130890103)

[Figure 3 : Capture d'écran du planning du projet (méthode Scrum). 6](#_Toc130890104)

[Figure 4 : Capture d'écran du diagramme de Gantt pour la planification finale (méthode Waterfall). 9](#_Toc130890105)

[Figure 5 : : Wireframe desktop ville trouvée 12](#_Toc130890106)

[Figure 6 : Wireframe phone ville trouvée. 12](#_Toc130890107)

[Figure 7 : Wireframe desktop ville non trouvée. 13](#_Toc130890108)

[Figure 8 : Wireframe phone ville non trouvée. 13](#_Toc130890109)

[Figure 9 : Wireframe desktop et phone ville trouvée. 14](#_Toc130890110)

[Figure 10 : Wireframe desktop et phone ville non trouvée 14](#_Toc130890111)

[Figure 11 : Modèle représentant la pluie. 15](#_Toc130890112)

[Figure 12 : Modèle représentant une éclaire 15](#_Toc130890113)

[Figure 13 : Modèle représentant un orage. 15](#_Toc130890114)

[Figure 14 : Modèle représentant un soleil. 15](#_Toc130890115)

[Figure 15 : : Modèle représentant un nuage 15](#_Toc130890116)

[Figure 16 :Modèle représentant un nuage et un soleil. 15](#_Toc130890117)

[Figure 17 : Modèle représentant la neige 16](#_Toc130890118)

[Figure 18 : Exemple d'une scène complète. 16](#_Toc130890119)

[Figure 19 : Exemple d'une page avec de multiples scènes. 17](#_Toc130890120)

[Figure 20 : Capture d'écrans de la partie template du fichier WeatherWeek. 17](#_Toc130890121)

[Figure 21 : Représente la fonction qui crée le modèle 3D. 18](#_Toc130890122)

[Figure 22 :Capture d'écran d’une requête avec postman. 19](#_Toc130890123)

[Figure 23 : Code représentant la partie recherche de la page. 19](#_Toc130890124)

[Figure 24 : Fonction qui permet l'ajout des donnée dans un storage dans le fichier filterLocalWeather.js. 20](#_Toc130890125)

[Figure 25 : Données filtrées. 21](#_Toc130890126)

[Figure 26 : Fonction GetLocalWeather du contrôleur requests.js 22](#_Toc130890127)

[Figure 27 : Event listener, MeteoRenderer.vue. 22](#_Toc130890128)

[Figure 28 : Action de rotation du modèle 3D. 22](#_Toc130890129)

[Figure 29 : Animation de haut et bas 22](#_Toc130890130)

[Figure 30 : Enlèvement du fond sur photopea. 23](#_Toc130890131)

# Bibliographie

Amedhh. (2021, - -). *Basic Low Poly Cloud*. Récupéré sur sketchfab: https://sketchfab.com/3d-models/basic-low-poly-cloud-ae82c7379fc84444943b9b475c35d266

amit. (2022, - -). *Thunder*. Récupéré sur amit: https://sketchfab.com/3d-models/thunder-eb763328068b45769609eef6f3255c96

axios. (2023, - -). *axios*. Récupéré sur axios: https://axios-http.com/docs/intro

balsamiq. (2023, - -). *balsamiq*. Récupéré sur balsamiq: https://balsamiq.com/

blender. (2023, - -). *blender*. Récupéré sur blender: https://www.blender.org/

claude aubry. (21, mai 2018). https://claudeaubry.fr/post/2018/extraits-du-livre-scrum/. Paris, Paris, France.

Coders, L. N. (2022, aout 28). *Make your first Three JS Website and deploy on the web in less than 12 minutes*. Récupéré sur youtube: https://www.youtube.com/watch?app=desktop&v=A00WHN32qGE

cpnv.ch. (-, - -). *Icescrum*. Récupéré sur Icescrum.cpnv.ch: https://icescrum.cpnv.ch/#/

day.js. (2023, - -). *day.js*. Récupéré sur day.js: https://day.js.org/

figma. (2023, - -). *figma*. Récupéré sur figma: https://www.figma.com/

IronEqual. (2021, - -). *Sun from Poly by Google*. Récupéré sur sketchfab: https://sketchfab.com/3d-models/sun-from-poly-by-google-67b0952528b2476c9ccd4d2daead9a3e

Kuba, J. (2017, - -). *Cloud*. Récupéré sur sketchfab: https://sketchfab.com/3d-models/cloud-852ae741a60b44ec886fbf34bbd058f8

momentjs. (2023, - -). *momentjs*. Récupéré sur momentjs: https://momentjs.com/

pinia. (2023, - -). *pinia*. Récupéré sur pinia: https://pinia.vuejs.org/

sketchfab. (2023, - -). *sketchfab*. Récupéré sur sketchfab: https://sketchfab.com/feed

swisscenter. (2023, - -). *swisscenter*. Récupéré sur swisscenter: https://www.swisscenter.com/

three.js. (2023, - -). *three.js*. Récupéré sur three.js: https://threejs.org/

visualstudio. (2023, - -). *visualstudio*. Récupéré sur visualstudio: https://code.visualstudio.com/

vitejs. (2023, - -). *vitejs*. Récupéré sur vite: https://vitejs.dev/

vuejs. (2023, - -). *vuejs*. Récupéré sur vuejs: https://vuejs.org/guide/components/provide-inject.html#prop-drilling

W3schools. (-, - -). *AJAX Introduction*. Récupéré sur W3schools: https://www.w3schools.com/js/js\_ajax\_intro.asp

Wikipedia. (22, Février 2023). https://fr.wikipedia.org/wiki/Mod%C3%A8le\_en\_cascade. -, -, -.

worldweatheronline. (2023, - -). *Weather API*. Récupéré sur worldweatheronline: https://www.worldweatheronline.com/weather-api/api/

wrike. (-, - -). *wrike*. Récupéré sur wrike.com: https://www.wrike.com/main/

# Lexique

A

API: Application Programming Interface ou interface de programmation d’applications, 4

C

canvas: L'élément de canevas HTML à partir duquel charger la texture., 17

M

méthodologie Waterfall: Modèle en cascade qui consiste à la succession d'étape prédéfinies., 6

S

scène: Permettent de définir quoi et où doit être rendu par three.js., 16, 26

Scrum: Framework de gestion de projet flexible., 6