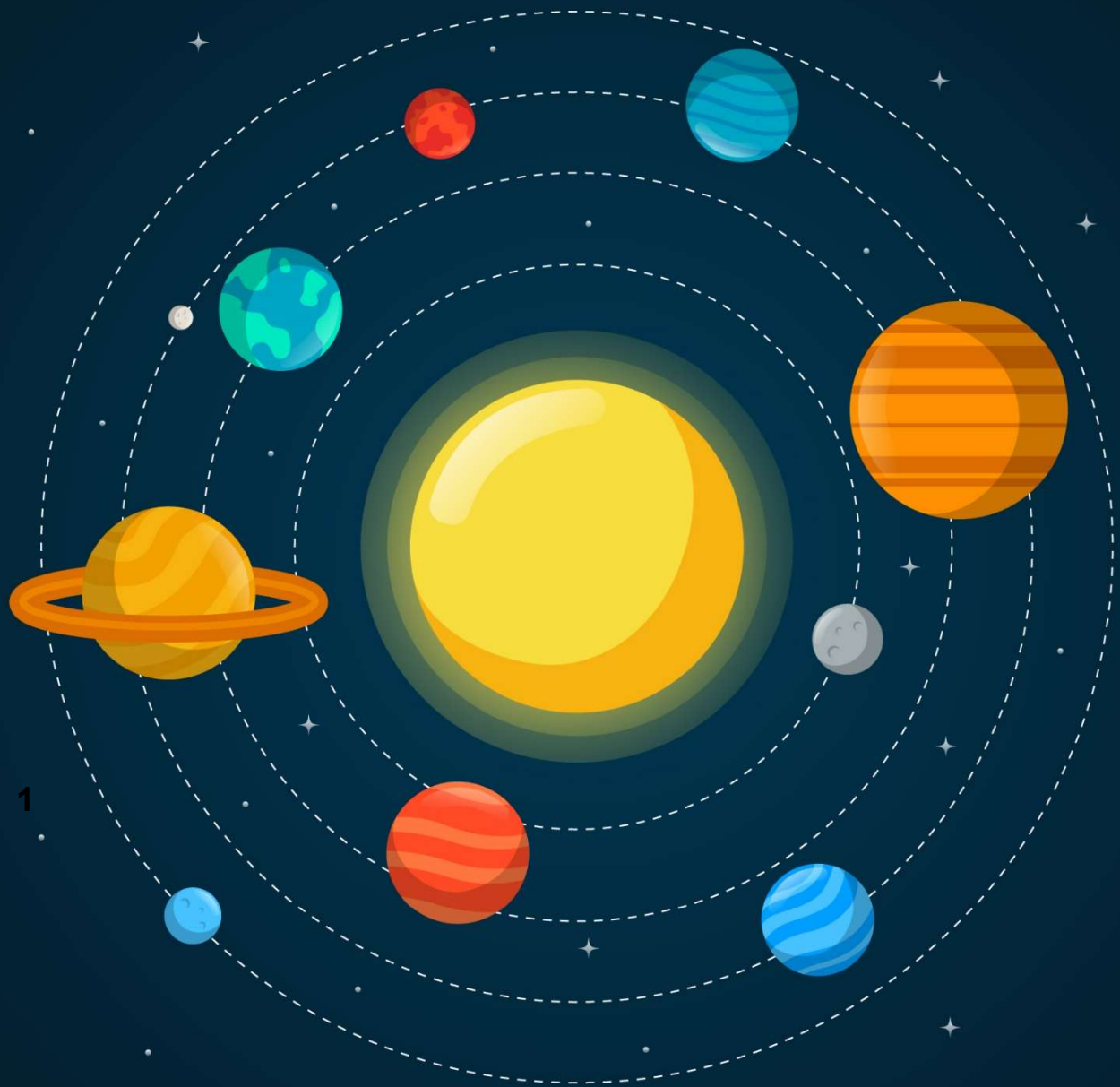


StellarMap



designed by  freepik

Table des matières

1	Table des matières	1
2	Analyse préliminaire	3
2.1	Introduction	3
2.2	Objectifs.....	3
2.3	Planification initiale	5
3	Analyse / Conception.....	6
3.1	Concept	6
3.2	Stratégie de test.....	9
3.3	Risques techniques	9
3.4	Planification	10
3.5	Dossier de conception	10
3.5.1	Logiciels / Framework utilisé :.....	10
4	Réalisation.....	11
4.1	Dossier de réalisation	11
4.1.1	Requête API	11
4.2	Favicon	17
4.3	Renderer.....	Erreur ! Signet non défini.
4.4	Planet	19
4.4.1	Création	19
4.4.2	Animation.....	19
4.5	Déploiement	20
4.6	Répertoires	21
4.7	Description des tests effectués.....	22
4.8	Erreurs restantes	22
4.9	Liste des documents fournis	22
5	Conclusions	23
6	Bibliographie.....	24
7	Table des illustrations.....	25
8	Lexique	26
9	Annexes.....	27
9.1	Planification initiale	27
9.2	Résumé du rapport du TPI / version succincte de la documentation.....	31
9.3	Journal de travail	31
9.4	Archives du projet.....	31

2 Analyse préliminaire

2.1 Introduction

Ce projet est réalisé dans le cadre du travail pratique individuel (TPI) qui s'effectue lors de la dernière année de CFC en informatique.

Ce travail s'effectue sur une période de réalisation de 90 heures, entre le 2 mai de 8h50 au 2 juin à 15h20.

Le sujet est une carte 3D interactive du système solaire, il a été choisi à la suite de la proposition de ce sujet par le candidat.

2.2 Objectifs

L'objectif de ce projet est de créer une carte interactive du système solaire sur lequel il sera possible de voir les 8 planètes et leurs lunes ainsi que le soleil. Il sera possible de tourner autour du soleil et d'observer les planètes sous un autre angle. Une description des planètes devra s'afficher lorsqu'un utilisateur clique sur celui-ci, de plus il sera possible d'accélérer la vitesse de déplacement des planètes.

Sept objectifs spécifiques sont à atteindre :

1. La carte s'affiche avec toutes les huit planètes.
2. L'utilisateur peut naviguer dans le système solaire.
3. Ergonomie et facilité d'utilisation du produit (Bastien et Scapin).
4. Les informations des différentes planètes s'affichent quand on clique dessus.
5. Le site est « responsive » et peut être utilisé depuis un smartphone ou une tablette.
6. L'utilisateur peut modifier la vitesse de déplacement des planètes.
7. Les angles de vue du système peuvent être déterminés par l'utilisateur.

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

Cahier des charges

2.3 Planification initiale

La planification initiale se découpe en cinq sprints découpés sur cinq semaines.
Le détail de la planification est disponible à [la section 9.1](#).

Ci-dessous vous trouverez la planification initiale sous forme de méthodologie Waterfall (Wikipedia, 22), j'ai choisi ce format car c'est cette forme convient le mieux pour la planification de projet demandée. Cependant le travail se déroulera en format agile comme le montre la deuxième image, l'analyse sera effectuée au fur et à mesure du projet.

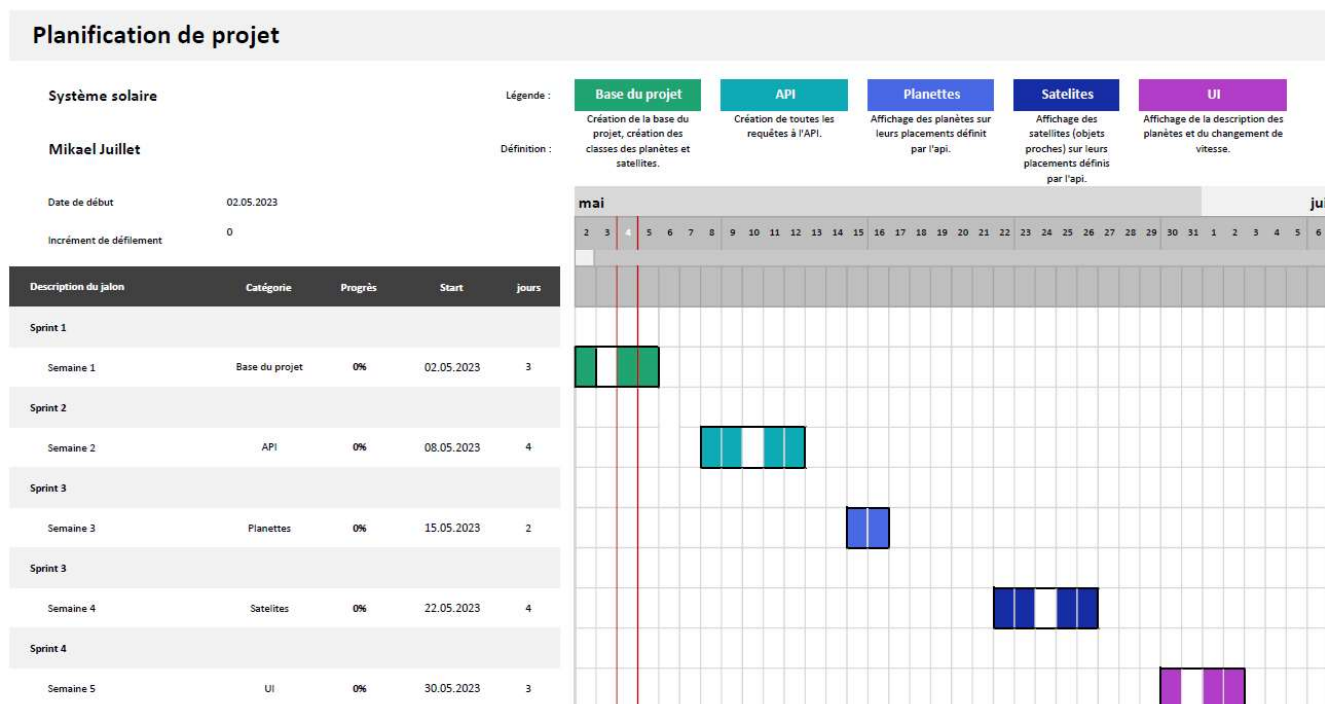


Figure 2 : Planification initiale du projet

Exemple d'un sprint d'une semaine



3 Analyse / Conception

3.1 Concept

Ce site est conçu pour afficher un système solaire en 3D, le visiteur pourra tourner autour de la carte et ainsi voir les planètes sous d'autres angles, il aura aussi la possibilité d'accélérer le temps afin de voir le déplacement des planètes à des vitesses différentes.

Ce système se verra affiché les 8 planètes du système solaire, leurs lunes ainsi que les astéroïdes à proximité de la planète terre.

Le projet sera hébergé sur swisscenter, les liens des accès au projet sont les suivantes :

Code source : <https://github.com/Juillet-Mikael/TPI>

Planification du projet : <https://icescrum.cpnv.ch/p/TPIJUILLET/#/project>

Documentation :

<https://github.com/Juillet-Mikael/TPI/blob/main/documents/documentation.docx>

Journal de travail :

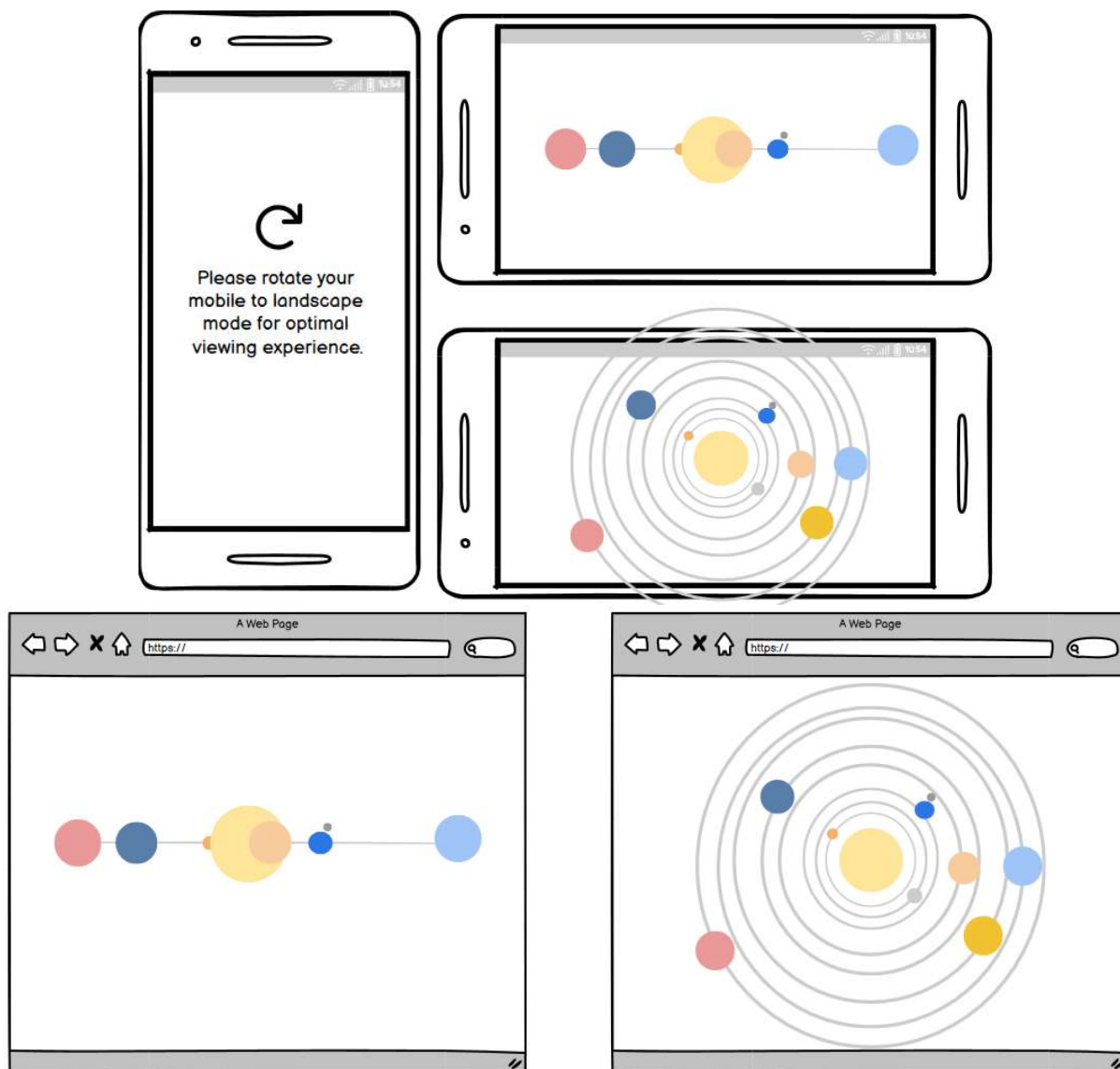
<https://github.com/Juillet-Mikael/TPI/blob/main/documents/journaux.xlsm>

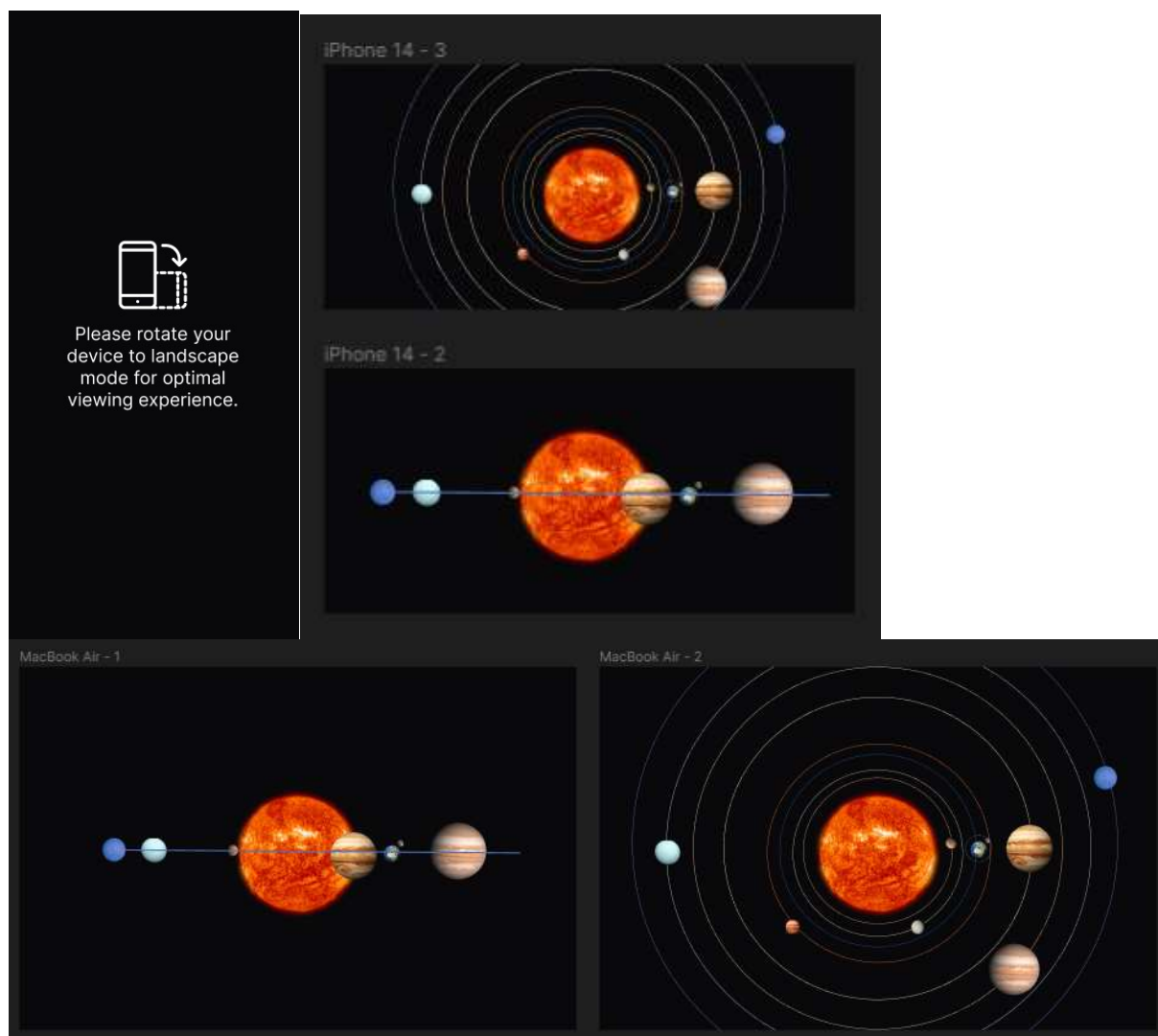
3.1.1.1 Requêtes

Il est prévu d'utiliser deux API de la Nasa, l'api « [horizon view](#) » permet de récupérer des informations précises sur les objets spatiaux dans notre système solaire. Horizon sera utilisé pour récupérer toutes les informations nécessaires au placement, et à la définition des planètes comme le volume, la densité, la position précise actuel, la température etc.

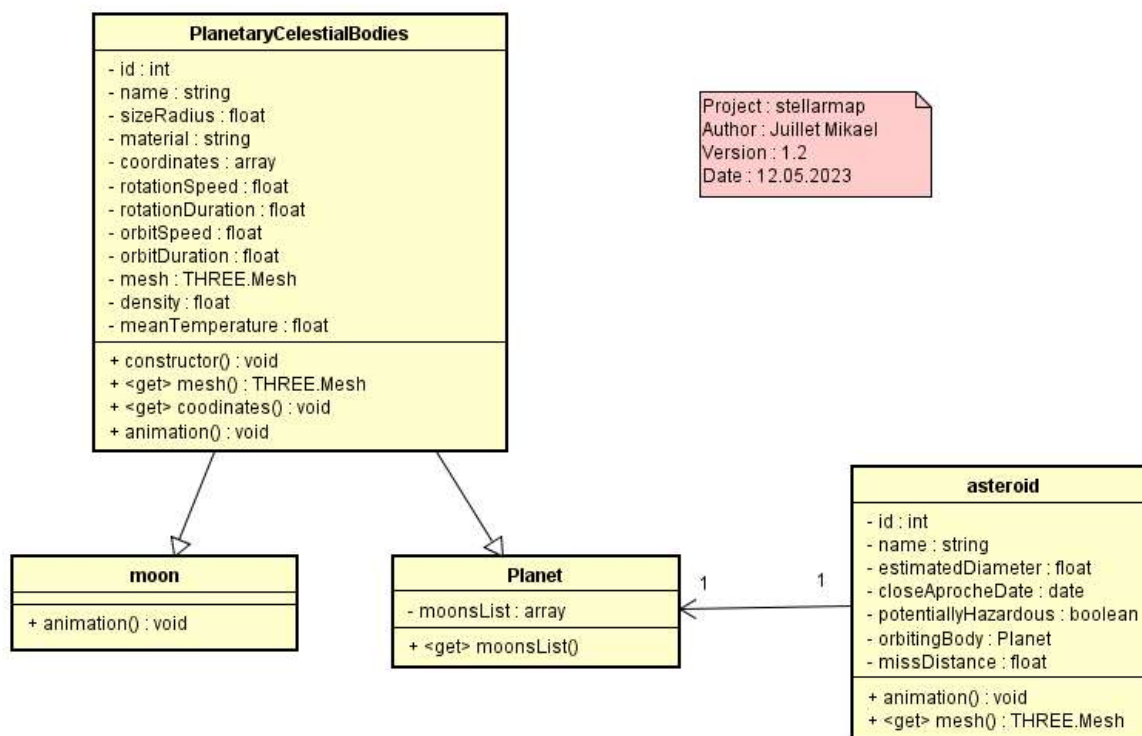
La deuxième API est « Near Earth Object » qui permet de récupérer la liste des objets proche de la terre à un temps donnée, elle sera utilisée pour placer approximativement les astéroïdes sur la carte car aucune donnée de placement précise ne peut être récupérer via cette api.

3.1.1.2 Maquettes





3.1.1.3 Diagramme de classe



3.1.1.4 Diagramme de séquences

3.2 Stratégie de test

Je vais essentiellement être effectuer manuellement pour la partie visuelle, j'ai choisi de faire cela car je ne dispose que de peu de temps pour crée mon projet de plus j'ai une très faible connaissance de la création de tests en Javascript.

Une exception sera faite pour ce qui est de la partie API en semaine 2, j'effectuerais des tests unitaires sur le filtrage des donnée et la réception des données.

J'effectuerai à chaque fin de sprint une série de tests dès l'implémentation de l'api en semaine numéro 2.

Les tests en rapport à la vue comme pour le déplacement des planètes seront des tests fonctionnels. Ces tests ont pour objectif de s'affurer du bon déroulement de l'affichage du système solaire.

3.3 Risques techniques

Les risques techniques sont mon manque de connaissance à l'utilisation de Three.js (three.js, 2023) et Ajax (W3schools, -), malgré de solides base acquises grâce à la préparation au TPI, j'ai tout encore besoin de beaucoup me référer aux documentations.

3.4 Planification

3.5 Dossier de conception

3.5.1 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
Three.js (WebGL) (three.js, 2023)	0.152.2	Rendu 3D
Ajax (W3schools, -)	-	Requêtes
Moment (momentjs, 2023)	2.29.4	Gestion des dates
Vitest (vitest, 2021)	0.31.0	Tests

4 Réalisation

4.1 Dossier de réalisation

4.1.1 Requête API

Les deux requêtes sont placées dans le dossier model dans un fichier appelé requests.js.

```
function formatURL(url, parameters) {  
    var fullURL = url + "?";  
  
    Object.keys(parameters).forEach( (key, index) => {  
        fullURL = fullURL + key + "=" + parameters[key]  
  
        if (index !== Object.keys(parameters).length - 1) {  
            fullURL = fullURL + "&";  
        }  
    });  
  
    return fullURL;  
}
```

4.1.1.1 Variables d'environnement

Un fichier .env a été créé pour stocker les données sensibles tel que la clé api.

```
VITE_API_URL_HORIZON=https://ssd.jpl.nasa.gov/api/horizons.api  
VITE_API_URL_NEO=https://api.nasa.gov/neo/rest/v1/feed  
VITE_API_KEY=APIKEY
```

4.1.1.2 Horizon system

Url de requête : <https://ssd.jpl.nasa.gov/api/horizons.api>

Certains paramètres sont absolument obligatoires en voici la liste (Nasa, 2022) :

- **Format**, sera retourné en Json.
- **Command**, représente l'id de l'objet ciblé (ex. terre = 399).
- **Objet_data**, représente les données de l'objet tel que la période de rotation.
- **Make_Ephem**, représente les données de placement des planètes.
- **Start_time**, la date de départ des données en format années-mois-jour.
- **End_time**, la date de fin des données en format années-mois-jour.
- **Step_size**, la durée séparant les informations de placement de l'objet ciblé.

Cette requête engendre une erreur corse c'est pourquoi j'ai ajouté dans mon header un Origin mais l'erreur ne se résout pas. (Braiam, 2017) (Simplified, 2021) Je ne parviens pas à récupérer les données pourtant le lien est valide et retourne des données.

Erreur Corse :

No 'Access-Control-Allow-Origin' header is present on the requested resource. If an opaque response serves your needs, set the request's mode to 'no-cors' to fetch the resource with CORS disabled.

J'ai pu récupérer les données depuis Postman.

The screenshot shows a Postman interface with a GET request to `https://ssd.jpl.nasa.gov/api/horizons.api?format=json&OBJ_DATA='YES'&MAKE_EPHEM='YES'&EPHEM_TYPE='VECTORS'&START_TIME='2023-05-04'&STOP_TIME='2023-05-05'&STEP_SIZE='1d'&COMMAND='399'`. The response status is 200 OK. The JSON response includes a signature and a detailed result for Earth (399) with various geophysical properties.

```

1 {
2   "signature": {
3     "source": "NASA/JPL Horizons API",
4     "version": "1.2"
5   },
6   "result": "*****\n Revised: April 12, 2021\n
Earth 399\n \n GEOPHYSICAL PROPERTIES (revised May 9, 2022):\n Vol. Mean Radius (km) = 6371.01+-0.02
Mass x10^24 (kg)= 5.97219+-0.0006\n Equ. radius, km = 6378.137 Mass layers:\n Polar axis, km = 6356.
752 Atmos = 5.1 x 10^18 kg\n Flattening = 1/298.257223563 oceans = 1.4 x 10^21 kg\n Density,
g/cm^3 = 5.51 crust = 2.6 x 10^22 kg\n J2 (IERS 2010) = 0.00108262545 mantle = 4.043
x 10^24 kg\n g_p, m/s^2 (polar) = 9.8321863685 outer core = 1.835 x 10^24 kg\n g_e, m/s^2 (equatorial) = 9.
7803267715 inner core = 9.675 x 10^22 kg\n g_o, m/s^2 = 9.82022 Fluid core rad = 3480 km\n GM, km^3/
s^2 = 398600.435436 Inner core rad = 1215 km\n GM 1-sigma, km^3/s^2 = 0.0014 Escape velocity = 11.186 km/s\n

```

Figure 3: Données retournées via Postman.

Mais malgré un retour de requête avec un code d'état de 200 je n'avais aucune donnée.

▼ Général

URL de requête: `https://ssd.jpl.nasa.gov/api/horizons.api?format=json&COMMAND=undefined&OBJ_DATA=YES&MAKE_EPHEM=YES&EPHEM_TYPE=IZE=1d`
Mode de requête: GET
Code d'état: 200 OK
Règlement sur les URL de provenance: strict-origin-when-cross-origin

▼ En-têtes de réponse

Afficher la source

Connection: keep-alive
Content-Type: application/json
Date: Fri, 12 May 2023 09:48:57 GMT
Server: nginx
Transfer-Encoding: chunked

▼ En-têtes de requête

Afficher la source

Accept: /*
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en;q=0.9,fr-CH;q=0.8,fr;q=0.7,de-DE;q=0.6,de;q=0.5,en-US;q=0.4
Connection: keep-alive
Host: ssd.jpl.nasa.gov
Origin: http://localhost:5173
Referer: http://localhost:5173/
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not:A-Brand";v="99"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: cross-site
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0 Safari/537.36

Figure 4 : Etat de la requête Horizon API sans données.

Échec du chargement des données de la réponse: No data found for resource with given identifier

Figure 5 : Erreur d'affichage des données reçues.

Après en avoir discuté avec mon chef de projet Monsieur Benzonana, nous avons décidé de l'envoi d'un e-mail au support de la Nasa, puis j'exporterai les données dans un fichier json que je traiterai ainsi afin de ne pas perdre du temps. Si le temps le permet je reviendrais éventuellement sur cette partie en fin de projet, en plus de ceci j'ai créé une [issue](#) sur GitHub afin de maintenir le projet.

La réponse de la Nasa a été expéditive :

« Les API de la NASA ne peuvent être intégrées à aucun site Web non-NASA. Donc, je crains que vous n'ayez besoin de trouver une autre méthode. C'est la politique de la NASA que nous sommes tenus de suivre. Vous pourrez peut-être appeler l'API à partir d'un script en dehors de votre serveur Web, puis déterminer comment connecter votre application Web à ce script. »

Pour donner suite à cette réponse nous avons décidé mon chef de projet et moi de garder l'exportation des données dans un fichier json.

4.1.1.2.1 Récupération des données

Les données reçues sont sous forme de String, je dois récupérer des valeurs précise dans ces tableaux c'est pourquoi j'utilise des expression rationnelles (regex). Celle-ci vont me permettre de récupérer des valeurs en fonction d'une expression régulière dans mon fichier. (mozilla, 2023)

```
*****Revised:April12,2021Mercury199 dataFilter.js?t=1683886315807:19
PHYSICALDATA(updated2021-Apr-
12):Vol.MeanRadius(km)=2440+-1Density(gcm^-3)=5.427Massx10^23(kg)=3.302Volume(x10^10km^3)=6.085Sideralrot.period=58.6463dSid.rot.rate(rad/s)=0
.0000124001Meansolarday=175.9421dCoreradius(km)=~1600GeometricAlbedo=0.106Surfaceemissivity=0.77+-0.06GM(km^3/s^2)=22031.86855Equatorialradius
,Re=2440kmGM1-sigma(km^3/s^2)=Massratio(Sun/plnt)=6023682Mom.ofInertia=0.33Equ.gravity/m/s^2=3.701Atmos.pressure(bar)=
<5x10^-15Max.angulardiam.=11.0'MeanTemperature(K)=440Visualmag.V(1,0)=-0.420bliquitytoorbit[1]=2.11'+/-0.1'Hill'ssphererad.Rp=94.4Sideralorb.p
er.=0.2408467yMeanOrbitvel.km/s=47.362Sideralorb.per.=87.969257dEscapevel.km/s=4.435PerihelionAphelionMeanSolarConstant(W/m^2)1446262789126Max
imumPlanetaryIR(W/m^2)1270055008000MinimumPlanetaryIR(W/m^2)666*****Ephemeris/API_USERMonMay807:4
*****Targetbodyname:Mercury(199)
@:322023Pasadena,USA/Horizons*****Starttime:A.D.2023-May-
{source:DE441}Centerbodyname:Earth(399){source:DE441}Center-
siteName:BODYCENTER*****
0400:00:00.0000TDBStoptime:A.D.2023-May-0500:00:00.0000TDBStep-
size:1440minutes*****Centergeodetic:0.0,0.0,0.0{E-
lon(deg),Lat(deg),Alt(km)}Centercylindric:0.0,0.0,0.0{E-
lon(deg),Dxy(km),Dz(km)}Centerradii:6378.137,6378.137,6356.752km{Equator_a,b,pole_c}Outputunits:KM-
SCalendarMode:MixedJulian/GregorianOutputtype:GEOMETRICcartesianstatesOutputformat:3(position,velocity,LT,range,range-
rate)EOPFile:eop.230504.p230727EOPCoverage:DATA-BASED1962-JAN-20T02023-MAY-04.PREDICTS->2023-JUL-
26ReferenceFrame:EclipticofJ2000.0*****JDTDBXYZVXVYVZLTRGR*****
*****$SOE2460068.500000000=A.D.2023-May-
0400:00:00.0000TDBX=6.439857249172552E+07Y=5.346674700502940E+07Z=1.766200542283878E+05VX=5.932433425858534E+00VY=-9.068016481428440E+00VZ=-4.8
94541217796761E+00LT=2.791973394962744E+02RG=8.370125667464858E+07RR=-1.238470929543310E+002460069.500000000=A.D.2023-May-
0500:00:00.0000TDBX=6.496800946250510E+07Y=5.274906761607163E+07Z=-2.463364871165343E+05VX=7.230568569470446E+00VY=-7.535644675529745E+00VZ=-4.
893499593699156E+00LT=2.791468567349237E+02RG=8.368612232353663E+07RR=8.778380713596032E-
01$SOE*****TIMEBarycentricDynamicalTime("TDB"orT_eph)outputwasrequest
ed.Thiscontinuouscoordinateis equivalenttotherelativisticproptimeofaclockatrestinareferenceframeco-
movingwiththesolarsystembarycenterbutoutsidethesystem'sgravitywell.Itistheindependentvariableinthesolarsystemrelativistic equationsofmotion.TDBr
unsatauniformrateofoneSIsecondpersecondandis independentofirregularitiesinEarth'srotation.CALENDARSYSTEMMixedcalendarmodewasactivesuchthatcalend
ardatesafterAD1582-Oct-15(ifany)areinthemodernGregoriansystem.Datespriorto1582-Oct-
5(ifany)areintheJuliancalendarsystem,whichisautomaticallyextendedfordatespriortoitsadoptionon45-Jan-
18C.TheJuliancalendarisusefulformatchinghistoricaldates.TheGregoriancalendar moreaccuratelycorrespondstotheEarth'sorbitalmotionandseasons.A"Greg
orian-
only"calendar modeisavailableifsuchphysicaleventsaretheprietaryinterest.REFERENCEFRAMEANDCOORDINATESEclipticathetstandardreferenceepochReferencee
poch:J2000.0X-Yplane:adoptedEarthorbitalplaneatthereferenceepochNote:IAU76bliquityof84381.448arcsecondswrtICRF-X-YplaneX-axis:ICRFZ-
axis:perpendiculartotheX-
Yplaneinthedirectional(+or-)senseofEarth'snorthpoleatthereferenceepoch.Symbolmeaning:JDTDBJulianDayNumber,BarycentricDynamicalTimeXX-
componentofpositionvector(km)YY-componentofpositionvector(km)ZZ-componentofpositionvector(km)VXX-componentofvelocityvector(km/sec)VYY-
componentofvelocityvector(km/sec)VZZ-componentofvelocityvector(km/sec)LTOne-waydown-legNewtonianlight-
time(sec)RGRRange;distancefromcoordinatecenter(km)RRRRange-
rate;radialvelocitywrtcoord.center(km/sec)ABERRATIONSANDCORRECTIONSGeometricstatevectorshaveNOcorrectionsoraberrationsapplied.Computationsby...
SolarSystemDynamicsGroup,HorizonsOn-LineEphemerisSystem4800OakGroveDrive,JetPropulsionLaboratoryPasadena,CA91109USAGeneralsite:https://ssd.jpl.
nasa.gov/Mailinglist:https://ssd.jpl.nasa.gov/email_list.ht*****
```

Figure 6 : Données récupérée après suppression des espaces.

Pour formater le texte, il faut en premier définir l'expression régulière de l'élément que l'on souhaite exporter via des regex. Puis faire une reconnaissance dans le texte de l'élément spécifier par le regex et finalement récupérer la bonne valeur.

Pour crée les regex j'ai utilisé chatGPT pour générer les regex en fonction des données, étant donné que chaque planètes as des données différentes il était plus rapide de demander à chatGPT de générer les regex en fonction des données voulus. (OpenAI, 2023)

Comme exemple le code ci-dessous qui cherche trois valeurs x=, y= et z= ils sont suivis de chiffres et d'une combinaisons E+- accompagné de chiffres.

```
const regexPlacment = /(-?\d+(?:\.\d+)?(?:E[+-]?\d+)?)(-?\d+(?:\.\d+)?(?:E[+-]?\d+)?)(-?\d+(?:\.\d+)?(?:E[+-]?\d+)?)/
const matchPlacment = texteSansEspaces.match(regexPlacment);

// Return data in object form
return {
  id : matchNameID[2],
  name : matchNameID[1],
  sizeRadius : matchMeanRadius[1],
  material : matchNameID[1] + ".png",
  coordinate : {
    x : matchPlacment[1],
    y : matchPlacment[2],
    z : matchPlacment[3]
  },
  rotationSpeed : matchRotationRate[1],
  rotationDuration : matchRotationDays[2],
  orbitSpeed : matchOrbitalSpeed[2],
  orbitDuration : matchOrbitPeriod[2],
  obliquity : matchObliquity[2],
  density : matchDensity[1],
  meanTemperature : matchMeanTemp ? matchMeanTemp[2] : "Unknown"
};
```

Figure 7 : Exemple de l'utilisation de regex

4.1.1.3 Near Earth Objects

Url de requête : <https://api.nasa.gov/neo/rest/v1/feed>

La requête near earth object est simple, j'ai fetch les données en passant une date de début et une date de fin ainsi que la clé api définie dans le fichier env.

```
export function GetNearEarthObjects(apiKey, startDate, endDate) {
  return new Promise(resolve => {
    const parameters = {};
    parameters.start_date = startDate;
    parameters.end_date = endDate;
    parameters.api_key = apiKey;

    fetch(formatURL(urlNeo, parameters))
      .then(response => {
        resolve(response.json())
      })
      .then(data => {
        resolve(data.data);
      })
      .catch(function (err) {
        resolve("Something went wrong!", err);
      });
  });
}
```

Figure 8 : Fonction de création fetch des objets proches.

```
▼ {links: {...}, element_count: 25, near_earth_objects: {...}} ⓘ
  element_count: 25
  ▶ links: {next: 'http://api.nasa.gov/neo/rest/v1/feed?start_date=20...&api_key=LTD8dyOvAiwdRywXsbe4dMf1eJrok44Ip5aZVe0F', previous: 'ht
  ▼ near_earth_objects:
    ▼ 2023-05-11: Array(15)
      ▼ 0:
        absolute_magnitude_h: 22.64
        ▶ close_approach_data: [{...}]
        ▶ estimated_diameter: {kilometers: {...}, meters: {...}, miles: {...}, feet: {...}}
        id: "2293726"
        is_potentially_hazardous_asteroid: false
        is_sentry_object: false
        ▶ links: {self: 'http://api.nasa.gov/neo/rest/v1/neo/2293726?api_key=LTD8dyOvAiwdRywXsbe4dMf1eJrok44Ip5aZVe0F'}
        name: "293726 (2007 RQ17)"
        nasa_jpl_url: "http://ssd.jpl.nasa.gov/sbdb.cgi?sstr=2293726"
        neo_reference_id: "2293726"
        ▶ [[Prototype]]: Object
      ▶ 1: {links: {...}, id: '2467460', neo_reference_id: '2467460', name: '467460 (2006 JF42)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/
      ▶ 2: {links: {...}, id: '3092313', neo_reference_id: '3092313', name: '(2001 QN142)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.c
      ▶ 3: {links: {...}, id: '3564040', neo_reference_id: '3564040', name: '(2011 H05)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.cgi
      ▶ 4: {links: {...}, id: '3670297', neo_reference_id: '3670297', name: '(2014 JR2)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.cgi
      ▶ 5: {links: {...}, id: '3696301', neo_reference_id: '3696301', name: '(2014 MW4)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.cgi
      ▶ 6: {links: {...}, id: '3742055', neo_reference_id: '3742055', name: '(2016 CB31)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.cg
      ▶ 7: {links: {...}, id: '3841669', neo_reference_id: '3841669', name: '(2019 HF4)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.cgi
      ▶ 8: {links: {...}, id: '3878610', neo_reference_id: '3878610', name: '(2019 TQ4)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.cgi
      ▶ 9: {links: {...}, id: '3989253', neo_reference_id: '3989253', name: '(2020 BD11)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.cg
      ▶ 10: {links: {...}, id: '54051138', neo_reference_id: '54051138', name: '(2020 QM)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.c
      ▶ 11: {links: {...}, id: '54087420', neo_reference_id: '54087420', name: '(2020 VL)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.c
      ▶ 12: {links: {...}, id: '54135432', neo_reference_id: '54135432', name: '(2021 GU3)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.
      ▶ 13: {links: {...}, id: '54184284', neo_reference_id: '54184284', name: '(2021 PF7)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.
      ▶ 14: {links: {...}, id: '54350904', neo_reference_id: '54350904', name: '(2023 FK2)', nasa_jpl_url: 'http://ssd.jpl.nasa.gov/sbdb.
        length: 15
      ▶ [[Prototype]]: Array(0)
    ▼ 2023-05-12: (10) [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}]
    ▶ [[Prototype]]: Object
  ▶ [[Prototype]]: Object
```

Figure 9 : Données récupérées via a l'api NEO.

4.2 Favicon

J'ai choisi comme favicon une image de soleil du site png art. (pngarts, -).

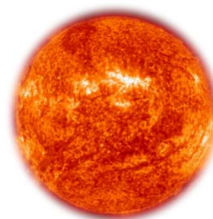


Figure 10 : Favicon

4.3 3D Affichages et animations

J'utilise pour ce projet la librairie Three.js qui permet d'afficher sur des pages web du contenu en 3D.

Threejs a besoin de 3 choses minimum :

- Une scène
- Une caméra
- Un moteur de rendu

La scène est une sorte de conteneur, elle permet de définir les éléments qui doivent être affichés ainsi que leurs placements. La caméra c'est elle qui va « filmer » ce que qu'il y a dans la scène et le moteur de rendu a pour objectif d'afficher la scène dans le navigateur via WebGL. (Bradley, 2022)

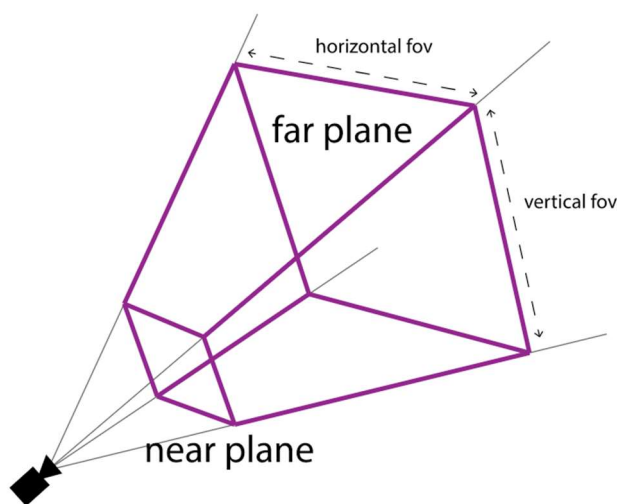


Figure 11 : Exemple de ce qu'est une scène. (Punkasem, 2017)

Il faut aussi ajouter une fonction d'animation, cette fonction crée une boucle qui fera afficher la scène via le moteur de recherche à chaque fois que l'écran est actualisé.

```
animate() {
  this.#planetList.forEach(planet => {
    planet.animation();
  });

  this.#sun.rotation.y += 0.00007292115;

  this.#renderer.render( this.#scene, this.#camera );
}
```

Figure 12 : Fonction d'animation du renderer

4.3.1 Responsive

```
// Resize render when listen to resize
window.addEventListener('resize', () => {
  this.#camera.aspect = window.innerWidth / window.innerHeight;
  this.#camera.updateProjectionMatrix();
  this.#renderer.setSize(window.innerWidth, window.innerHeight);
  this.animate();
});
```

Figure 13 : Code permettant au redimensionnement du renderer

4.3.2 Mouvement utilisateur

```
// Add users controls
new OrbitControls( this.#camera, this.#renderer.domElement );
```

Figure 14 : Code permettant le mouvement de l'utilisateur

4.4 Corps céleste planétaire

Les corps célestes planétaires définissent tout corps en orbite autour du soleil.

4.4.1 Création

Une mesh est objets basés sur un maillage polygonal triangulaire.

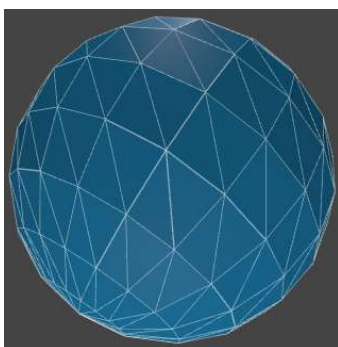


Figure 15 : Exemple de mesh sur une sphère

```
#createMesh() {  
  const geometry = new THREE.SphereGeometry(this.sizeRadius / 1000, 64, 16 );  
  const texture = new THREE.TextureLoader().load("./src/assets/images/" + this.textureFile);  
  const material = new THREE.MeshBasicMaterial({ map: texture });  
  const mesh = new THREE.Mesh(geometry, material);  
  mesh.position.set(parseFloat(this.coordinates.x) / 5000000, parseFloat(this.coordinates.y) / 5000000, 0);  
  mesh.rotation.set(3, 3, 0)  
  return mesh;  
}
```

Figure 16 : Création physique des planètes

4.4.2 Animation

```
animation() {  
  const deltaTime = this.#clock.getDelta();  
  
  this.mesh.rotation.x = 190  
  this.mesh.rotation.y += this.rotationSpeed * 10;  
  
  const rayon = (Math.sqrt((Math.pow(this.coordinates.x, 2) / 5000000 ) + (Math.pow(this.coordinates.y, 2) / 5000000)))  
  const distanceFactor = 2 * Math.PI * rayon; //km périmètre  
  const timeFactor = this.orbitDuration * 365 * 24 * 60 * 60; //years => seconds  
  
  const vitesse = distanceFactor / timeFactor;  
  this.#pointPivot.rotation.z += vitesse;  
}
```

Figure 17 : animation des planètes

4.5 Planètes

Les planètes héritent de la classe corps céleste planétaire et n'a aucune modification.

4.6 Moon

4.7 Astéroïde

Les astéroïdes n'héritent d'aucunes classes car elles sont particulières et n'ont pas le même comportement que les lunes ou planètes.

```
constructor(id, name, estimatedDiameter, closeAprocheDate, potentiallyHazardous, orbitingBody, missDistance) {  
  this.id = id;  
  this.name = name;  
  this.estimatedDiameter = estimatedDiameter;  
  this.closeAprocheDate = closeAprocheDate;  
  this.potentiallyHazardous = potentiallyHazardous;  
  this.orbitingBody = orbitingBody;  
  this.missDistance = missDistance;  
}
```

Figure 18 : Constructeur de la classe Ateroid

4.8 Déploiement

Le déploiement s'effectue sur swisscenter, afin de parvenir à la connexion au serveur j'ai créé une clé ssh depuis la plateforme de swisscenter.

Site web : <https://stellarmap.mycpnv.ch/>

4.9 Répertoires

Code source : <https://github.com/Juillet-Mikael/TPI>

Planification du projet : <https://icescrum.cpnv.ch/p/TPIJUILLET/#/project>

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

Architecture des documents :

- TPI
 - documents
 - journals
 - documentation
 - planification initiale
 - diagrams
 - diagramme de classe
 - diagramme de scéquence
 - instruction
 - src
 - model
 - view
 - controller
 -

4.10 Description des tests effectués

```
test('horizonAPIFilter() should return Jupiter data', () => {
  //Given
  const jupiter = Allplanets.planets[4].result;
  const jupiterExpected = {
    id : "599",
    name : "Jupiter",
    sizeRadius : "69911",
    material : "Jupiter.png",
    coordinate : {
      x : "7.900239544305509E+08",
      y : "3.977974646088730E+08",
      z : "-1.643804799922679E+07"
    },
    rotationSpeed : "0.00017585",
    rotationDuration : "9h55m29.71s",
    orbitSpeed : "13.0697",
    orbitDuration : "11.861982204",
    obliquity : "3.13",
    density : "1.3262",
    meanTemperature : "Unknown"
  };

  //When
  const result = horizonAPIFilter(jupiter);

  //Then
  expect(result).toEqual(jupiterExpected);
})
```

4.11 Erreurs restantes

4.12 Liste des documents fournis

5 Conclusions

6 Bibliographie

- balsamiq. (2023, - -). *balsamiq*. Récupéré sur balsamiq: <https://balsamiq.com/>
- Braiam. (2017, Mai 09). *No 'Access-Control-Allow-Origin' header is present on the requested resource—when trying to get data from a REST API*. Récupéré sur stackoverflow: <https://stackoverflow.com/questions/43871637/no-access-control-allow-origin-header-is-present-on-the-requested-resource-whe>
- claudio aubry. (21, mai 2018). <https://claudioaubry.fr/post/2018/extraits-du-livre-scrum/>. Paris, Paris, France.
- 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/>
- freepik. (-, - -). Vecteur gratuit système de système solaire classique avec deisgn plat. *Vecteur gratuit système de système solaire classique avec deisgn plat*. -, -, -. Récupéré sur <https://fr.freepik.com/>
- kjpargeter. (s.d.). Free photo starry night sky. *Free photo starry night sky*. feepick, -. Récupéré sur https://www.freepik.com/free-photo/starry-night-sky_7061153.htm#query=stars&position=2&from_view=search&track=sph
- momentjs. (2023, - -). *momentjs*. Récupéré sur momentjs: <https://momentjs.com/>
- mozilla. (2023, Mai 05). *Regular expressions*. Récupéré sur developer.mozilla.org: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_expressions
- Nasa. (2022, Septembre 1). *Horizons API*. Récupéré sur ssd-api: <https://ssd-api.jpl.nasa.gov/doc/horizons.html>
- OpenAI. (2023, - -). *Introducing GPT-4, OpenAI's most advanced system*. Récupéré sur Introducing GPT-4, OpenAI's most advanced system: <https://openai.com/>
- pinia. (2023, - -). *pinia*. Récupéré sur pinia: <https://pinia.vuejs.org/>
- pngarts. (-, - -). *Images Transparentes de soleil*. Récupéré sur pngarts: <https://www.pngarts.com/fr/explore/123391>
- Simplified, W. D. (2021, Mai 22). *Apprenez CORS en 6 minutes*. Récupéré sur Youtube: <https://www.youtube.com/watch?v=PNtFSVU-YTI>
- 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/>
- vitest. (2021, - -). *vitest*. Récupéré sur vitest: <https://vitest.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. -, -, -.
- wrike. (-, - -). *wrike*. Récupéré sur wrike.com: <https://www.wrike.com/main/>

7 Table des illustrations

FIGURE 1 : SYSTEME SOLAIRE (FREEPIK, -)	1
FIGURE 2 : PLANIFICATION INITIALE DU PROJET	5
FIGURE 3: DONNEES RETOURNEES VIA POSTMAN.	12
FIGURE 4 : ETAT DE LA REQUETE HORIZON API SANS DONNEES.	13
FIGURE 5 : ERREUR D'AFFICHAGE DES DONNEES REÇUES.	13
FIGURE 6 : DONNEES RECUPEREE APRES SUPPRESSION DES ESPACES.	14
FIGURE 7 : EXEMPLE DE L'UTILISATION DE REGEX	15
FIGURE 8 : FONCTION DE CREATION FETCH DES OBJETS PROCHES.	16
FIGURE 9 : DONNEES RECUPEREES VIA A L'API NEO.	16
FIGURE 10 : FAVICON	17
FIGURE 11 : EXEMPLE DE CE QU'EST UNE SCENE. (PUNKASEM, 2017)	17
FIGURE 12 : FONCTION D'ANIMATION DU RENDERER	18
FIGURE 13 : CODE PERMETTANT AU REDIMENSIONNEMENT DU RENDERER	18
FIGURE 14 : CODE PERMETTANT LE MOUVEMENT DE L'UTILISATEUR	18
FIGURE 15 : EXEMPLE DE MESH SUR UNE SPHERE	19
FIGURE 16 : CREATION PHYSIQUE DES PLANETES	19
FIGURE 17 : ANIMATION DES PLANETES	19
FIGURE 18 : CONSTRUCTEUR DE LA CLASSE ATEROID	20

8 Lexique

M

mesh

Objets basés sur un maillage polygonal triangulaire. · 19

méthodologie Waterfall

Modèle en cascade qui consiste à la succession d'étape prédéfinies. · 5

R

regex

Regex ou expressions rationnelles sont des motifs de combinaisons de caractères au sein de chaînes d'un texte. · 14

S

scène

Définit les éléments qui doivent être affichés ainsi que leurs placements. · 17

U

Un moteur de rendu

Responsable de l'affichage graphique des objets 3D dans une scène. · 17

9 Annexes

9.1 Planification initiale

Description	Catégorie	Progrès	Début	Heures prévu
Sprint 1				
Diagramme de classes	Base du projet	0%	03.05.2023	0.75
Diagramme de séquence	Base du projet	0%	03.05.2023	0.75
Création de la classe planète	Base du projet	0%	03.05.2023	1.00
Création de la classe satellite	Base du projet	0%	03.05.2023	1.00
Ajout des opérations dans les classes	Base du projet	0%	05.05.2023	2.25
Ajout de vitejs	Base du projet	0%	05.05.2023	0.25
Création du fichier détenant les codes de planètes	Base du projet	0%	05.05.2023	0.50

Sprint 2				
Création d'un contrôleur	API	0%	08.05.2023	0.25
Création d'un modèle	API	0%	08.05.2023	0.25
Ajout d'un fichier .env	API	0%	08.05.2023	0.25
Récupération de la clef API	API	0%	08.05.2023	0.25
Requêtes de récupération des planètes	API	0%	08.05.2023	3.00
Requêtes de récupération des objets proches	API	0%	09.05.2023	3.00
Requêtes de récupération des images	API	0%	11.05.2023	2.25
Récupération des erreurs dans le contrôleur	API	0%	12.05.2023	0.75
Lien entre la récupération des données et les classes	API	0%	12.05.2023	0.75

Sprint 3

Création de maquettes	Planettes	0%	14.05.2023	0.75
Ajout de three.js	Planettes	0%	14.05.2023	0.25
Création des planètes	Planettes	0%	14.05.2023	0.75
Placement des planètes	Planettes	0%	15.05.2023	0.50
Orbite sidérale	Planettes	0%	15.05.2023	0.50
Orbite autour du soleil	Planettes	0%	15.05.2023	0.75

Sprint 4

Ajout du déplacement utilisateur	Satelites	0%	22.05.2023	2.25
Création des Satélites	Satelites	0%	23.05.2023	1.50
Ajout des lunes	Satelites	0%	25.05.2023	1.50
Placement sur la carte	Satelites	0%	25.05.2023	1.50
Ajout de l'orbite	Satelites	0%	26.05.2023	1.50
Orbite sidérale	Planettes	0%	26.05.2023	1.50

Sprint 5

Création de maquettes	UI	0%	30.05.2023	0.75
Placement du canvas en arrière-plan	UI	0%	30.05.2023	0.75
Ajout de la description des planètes	UI	0%	01.06.2023	2.25
Ecoute d'un clique sur planètes	UI	0%	02.06.2023	0.75
Ajout de changement de vitesse	UI	0%	02.06.2023	1.50

Planification de projet

Système solaire

Mikael Juillet

Date de début 02.05.2023

Incrément de défilement 0

Légende :

Définition :

Base du projet

Création de la base du projet, création des classes des planètes et satellites.

API

Création de toutes les requêtes à l'API.

Planettes

Affichage des planètes sur leurs placements définis par l'api.

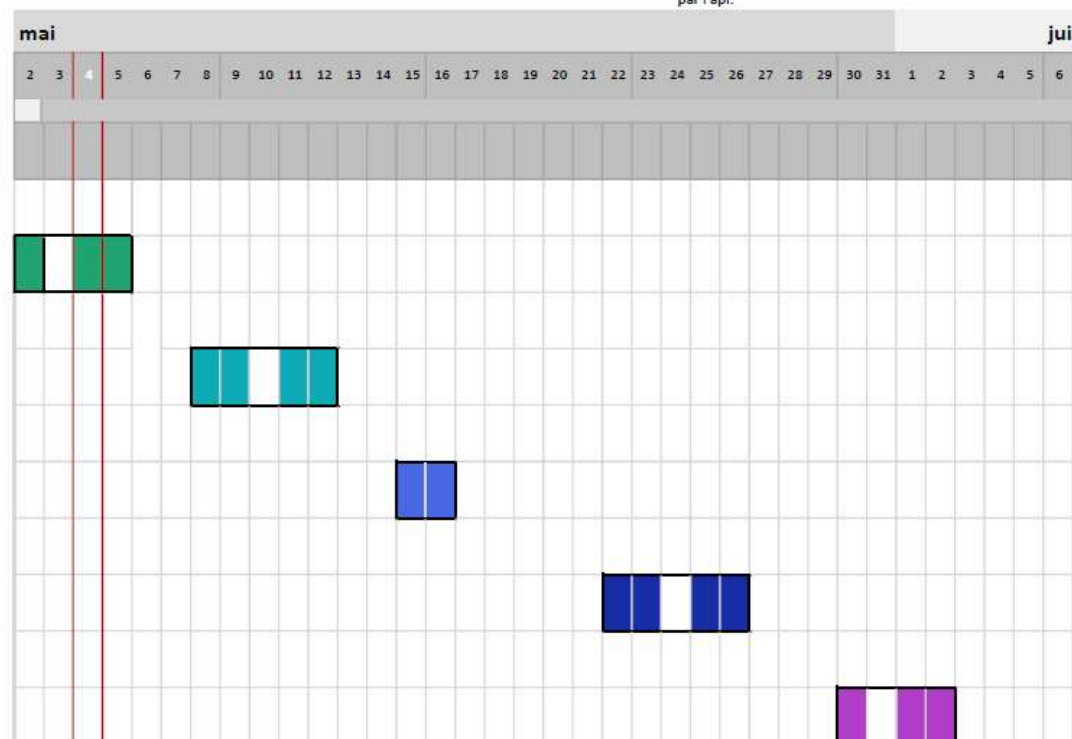
Satellites

Affichage des satellites (objets proches) sur leurs placements définis par l'api.

UI

Affichage de la description des planètes et du changement de vitesse.

Description du jalon	Catégorie	Progrès	Start	Jours
Sprint 1				
Semaine 1	Base du projet	0%	02.05.2023	3
Sprint 2				
Semaine 2	API	0%	08.05.2023	4
Sprint 3				
Semaine 3	Planettes	0%	15.05.2023	2
Sprint 3				
Semaine 4	Satellites	0%	22.05.2023	4
Sprint 4				
Semaine 5	UI	0%	30.05.2023	3



Exemple d'un sprint d'une semaine

Système solaire

Mikael Juillet

Date de début 02.05.2023

Légende:

Documentation

Implémentation

Analyse

Tests

Sprint reviews

* P = Période de 45 minutes

Horaire

Jours
Lundi
Mardi
Mercredi
Jeudi
Vendredi
Samedi
Dimanche

P1	P2	P3	P4	P5	P6	P7	P8	P9
Documentation	Documentation	Documentation	Documentation	Documentation	Documentation	Documentation	Documentation	Documentation
	Documentation	Documentation	Documentation	Documentation	Documentation	Documentation	Documentation	
Documentation	Documentation	Documentation			Documentation	Documentation	Tests	Documentation
	Documentation	Documentation	Tests	Tests	Documentation	Documentation		

Note :

Il est important de prendre en compte que la disposition changera car il y a par exemple des semaines avec seulement deux jours mais dans ces deux jours il y aura de l'analyse et des tests même s'ils ne sont pas prévus. C'est un schéma approximatif.

9.2 Résumé du rapport du TPI / version succincte de la documentation

9.3 Journal de travail

9.4 Archives du projet

Media, ... dans une fourre en plastique