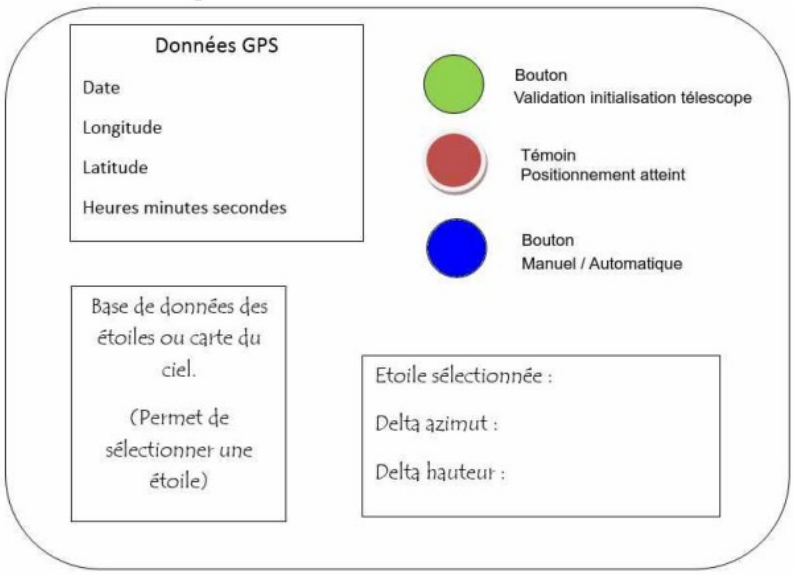
KOEHREN Teddy
1SN BTS
18/01/2023

E6.2 Téléscope : Fiche Technique IHM WEB

Table des matières

Contexte :

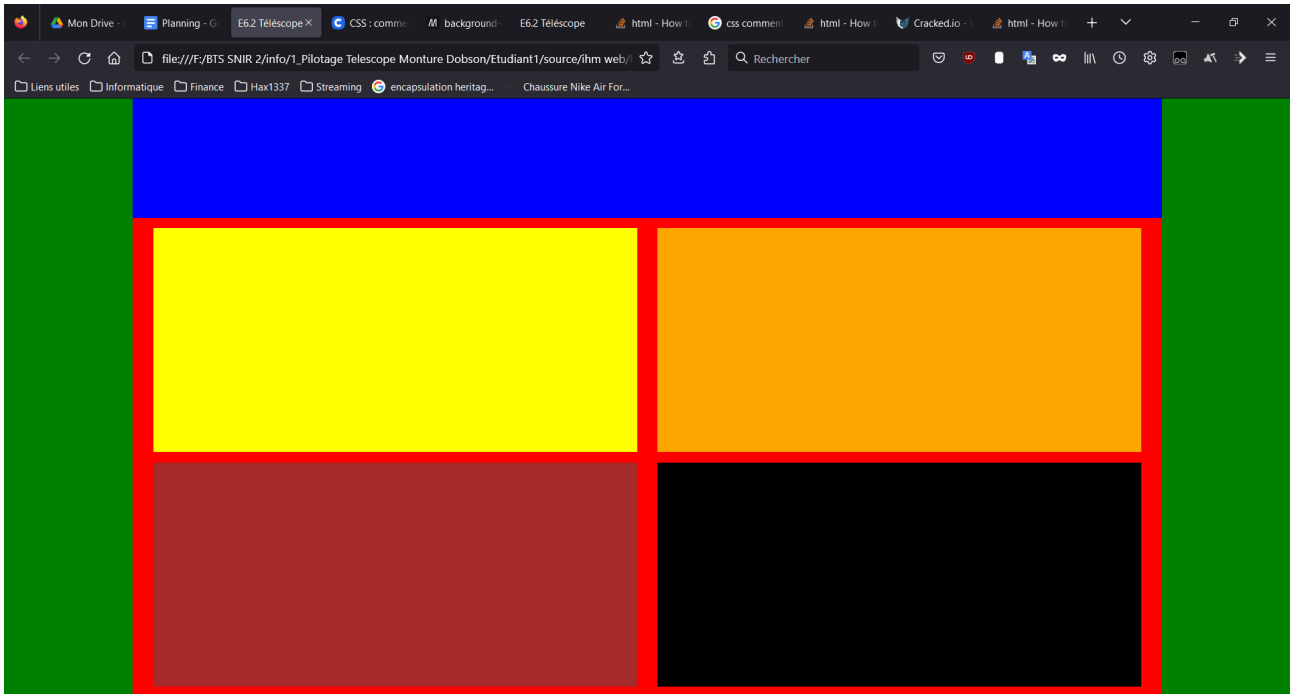


L'IHM WEB permettra à l'utilisateur d'interagir avec l'entièreté du système, le gps, la base de données, le calcul de coordonnées et la communication avec l'ESP devra se réaliser sur celui ci. Une maquette est fourni selon les demandes du CDC :

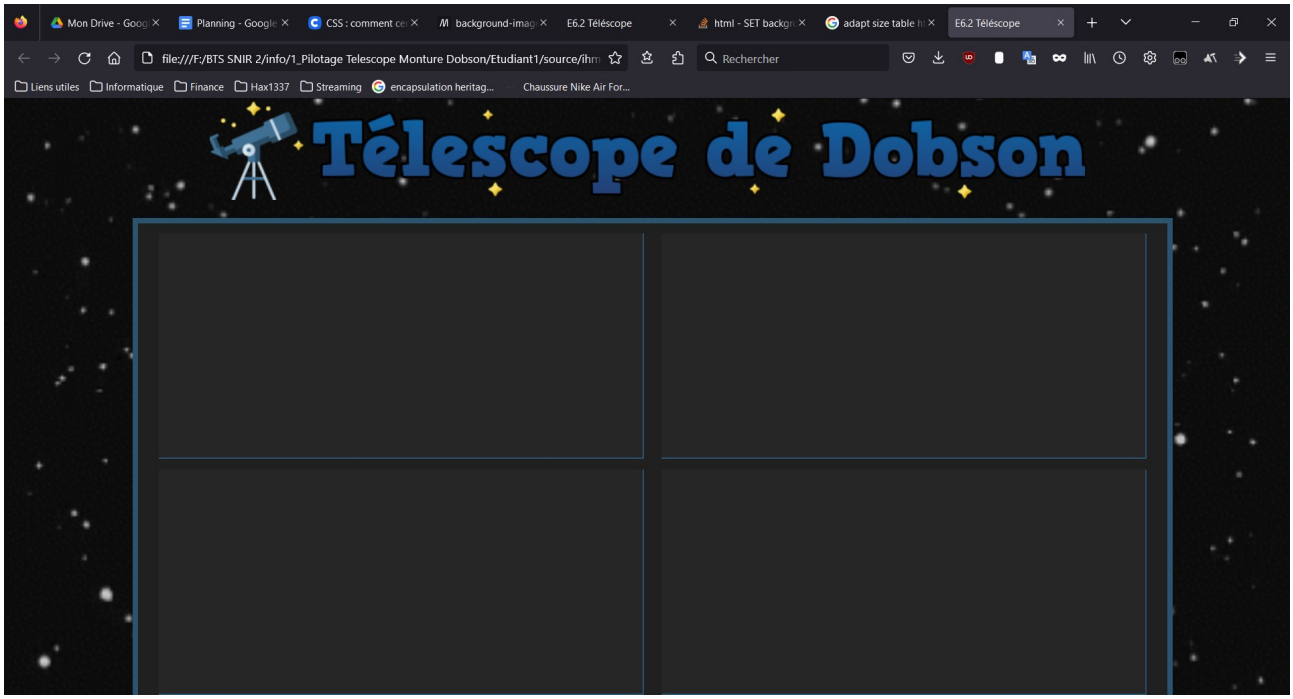
1. Création d’une page WEB statique au préalable

Code source disponible dans /source

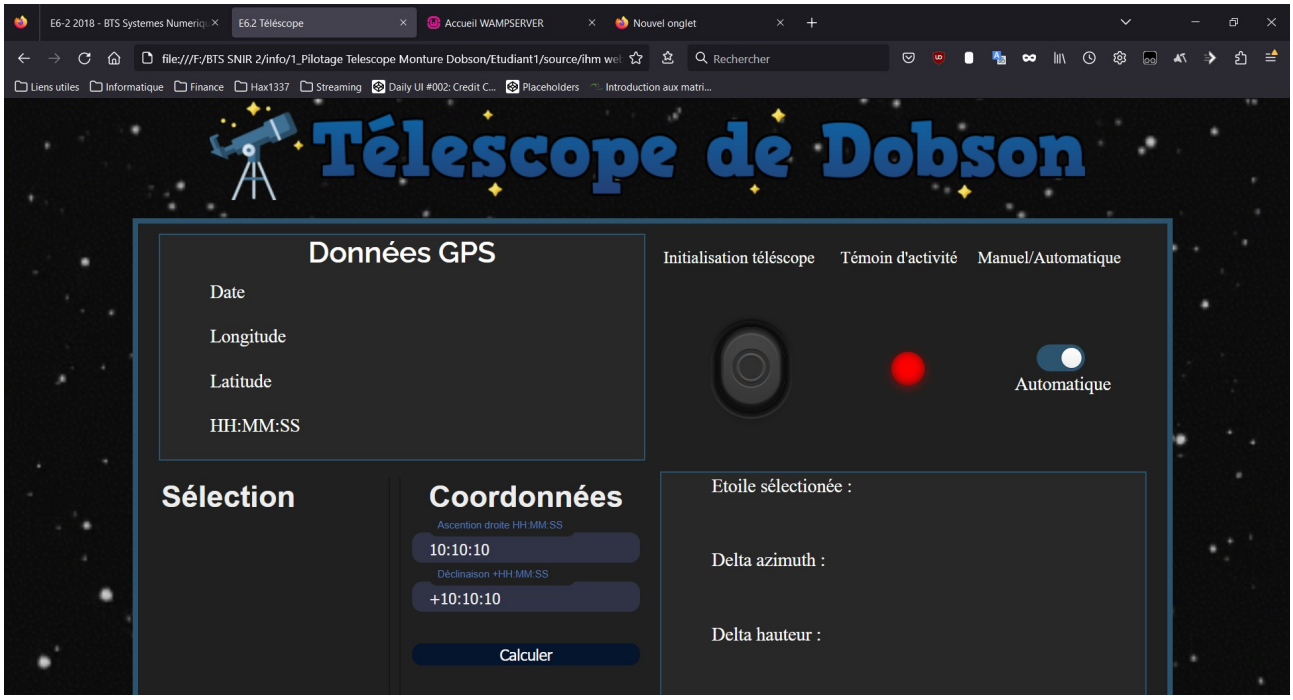
1) Création du modèle avec des blocs div :



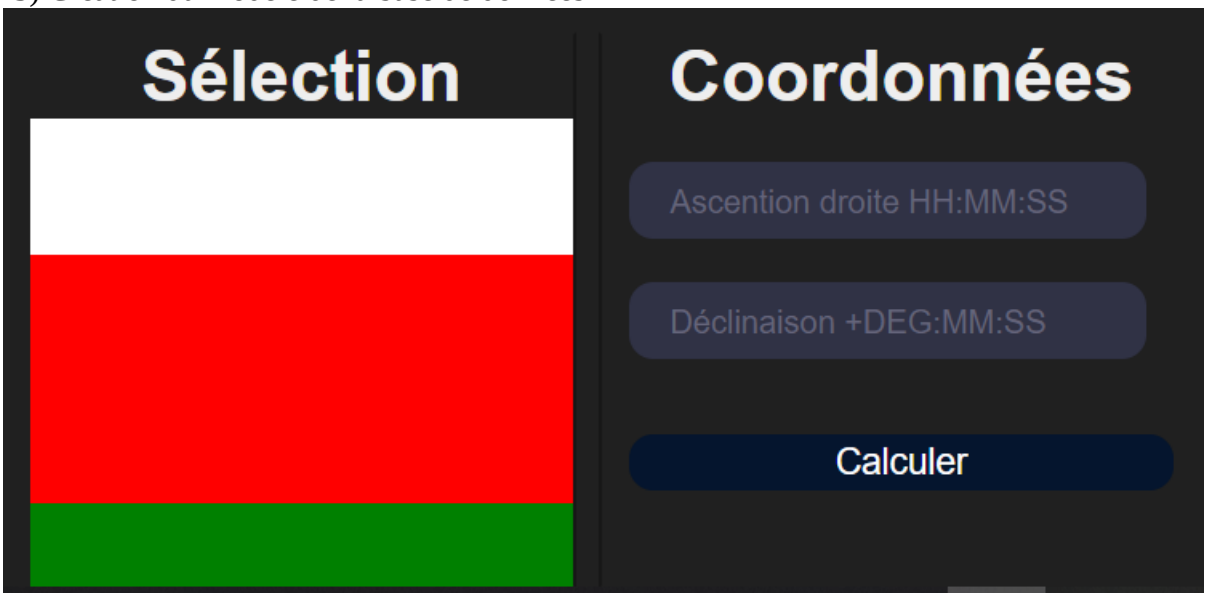
2) Mise en forme du modèle créé :



3) Ajout des éléments web sur la page :



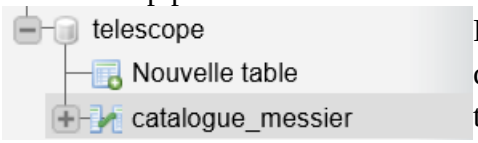
5) Création du modèle de la base de données



2. Création et communication avec la BDD telescope

Code source disponible dans /source

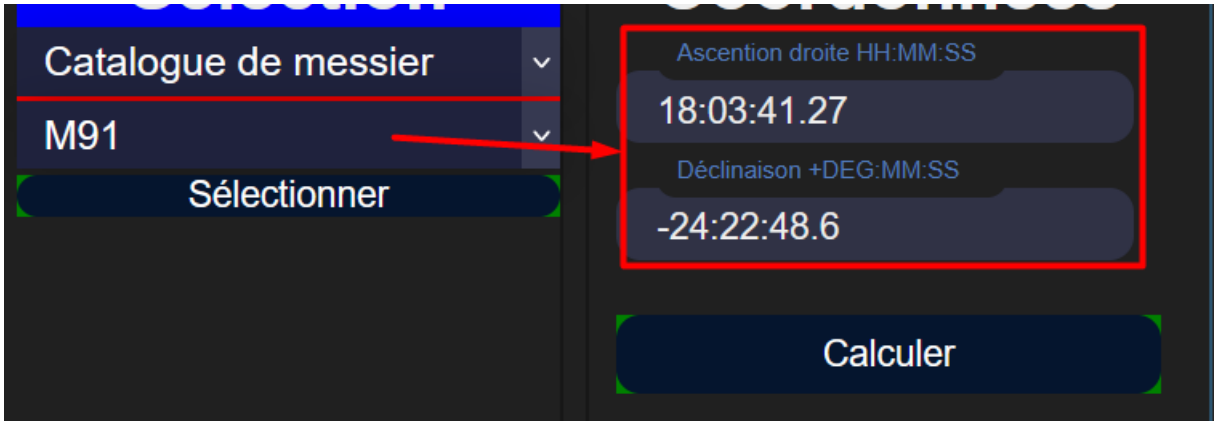
1) Installation Wamp pour créer le début de la base de données télescope :



Récupération du catalogue de messier et ajout des coordonnées sur certains objets astrales et ajout de la table dans la DB « telescope ».

2) Import des lignes de la colonne de Messier sur l'IHM web en PHP

3) Import des coordonnées de l'astre sélectionné



La page est en PHP et vérifie qu'une valeur du formulaire de sélection existe à chaque chargement de page, si c'est le cas, une requête SQL est envoyé pour récupérer l'ascension droite et la déclinaison.
Si une valeur est trouvé, elle est placée dans les champs d'entrées.

3. Migration du site web sur la raspberry

- Installation de la pile Lamp :
sudo apt-get install apache2 php libapache2-mod-php mariadb-server php-mysql

- Installations des modules PHP les plus courants :
sudo apt-get install php-curl php-gd php-intl php-json php-mbstring php-xml php-zip

- Deplacement du site et des images dans le répertoire PHP /var/www/html/ et de la base de données telescope de wamp dans /var/lib/mysql/

- Installation de phpmyadmin avec la configuration automatique sur apache2 :
sudo apt-get install phpmyadmin
Mot de passe : password pour l'utilisateur phpmyadmin

- Mis à jour et redémarrage :
sudo apt-get update && sudo apt-get upgrade && sudo reboot

- Creation utilisateur root mysql
sudo mysql -u root password

- Ajout du mot de passse password sur la page php pour se connecter à la DB
sudo nano /var/www/html/version_finale/home.php

- Accès sur le pannel de phpmyadmin possible via root:password →
http://192.168.4.1/phpmyadmin en wlan
http://172.20.81.221/phpmyadmin en eth0

- Redémarrage de Mysql :
sudo systemctl restart mysql

- Accès au site via :
http://192.168.4.1/version_finale/home.php en wlan
http://172.20.81.221/version_finale/home.php en eth0

4. Réception et formatage de la trame GSV en PHP

Un script python a été réalisé et permet de récupérer la trame GSV contenant toutes les informations nécessaires à l'IHM Web.

```
/* RECUPERATION DONNEES GPS AVEC SCRIPT PYTHON */
exec("python3 /home/pi/gps/gps.py",$gps_result); // envoie un array des print réalisés
$gps_trame = $gps_result[0];
echo "résultat: $gps_trame[0]";
```

résultat: $GPRMC,111825.513,A,4843.8952,N,00215.1852,E,0.39,333.94,080323,,,A*6C

Si une erreur est reçu, le site affichera le résultat GPS comme étant une erreur, dans le cas contraire, on découpe la trame à chaque séparateur avec explode() et on récupère les données qui nous sont nécessaires.

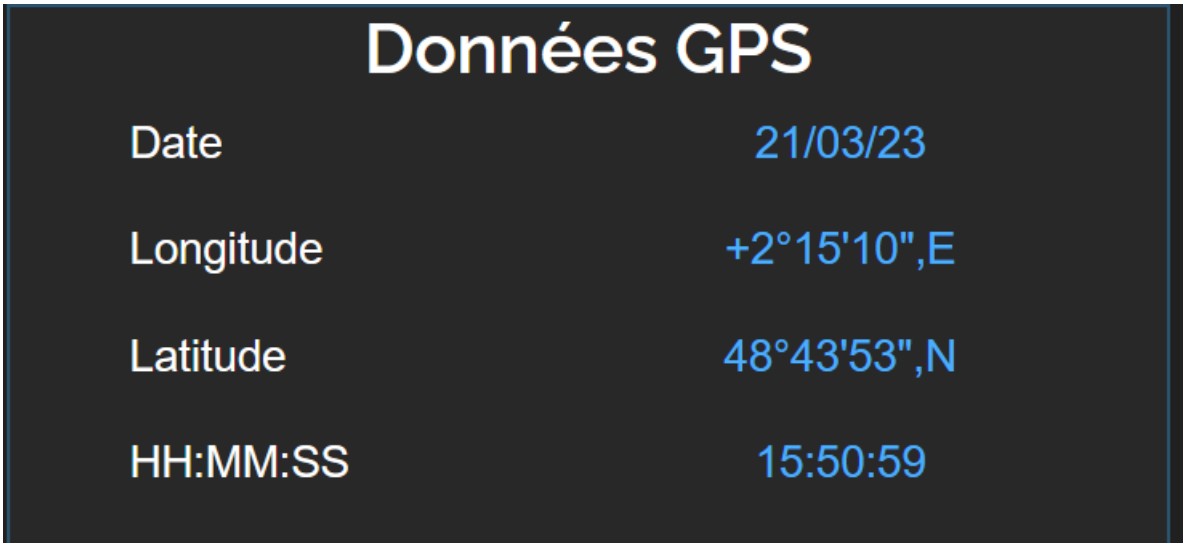
```
if ($gps_trame == "ERREUR GPS") // si un problème survient avec le gps
{
        $gps_var_date = $gps_var_longitude = $gps_var_latitude = $gps_var_time = "ERREUR GPS";
}
else
{
        $gps_trame_array = explode(',',$gps_trame);
        $gps_var_date = $gps_trame_array[9]; $gps_var_date = substr_replace($gps_var_date,'/',4,0);
$gps_var_date = substr_replace($gps_var_date,'/',2,0); // ajout slash sur la date
        $gps_var_longitude = $gps_trame_array[5].','.$gps_trame_array[6];
        $gps_var_latitude = $gps_trame_array[3].','.$gps_trame_array[4];
        $gps_var_time = substr($gps_trame_array[1],0,6); $gps_var_time =
substr_replace($gps_var_time,':',4,0); $gps_var_time = substr_replace($gps_var_time,':',2,0);
}
```

Un formattage est également effectué : on rajoute des slashs sur la date, deux points sur l'heure avec la fonction substr_replace(), et concatène pour les coordonnées géographiques.

On formate l'heure et les coordonnées :

```
$gps_trame_array = explode(',',$gps_trame);

$gps_var_date = $gps_trame_array[9];
$gps_var_date = substr_replace($gps_var_date,'/',4,0); // ajout slash sur la date
$gps_var_date = substr_replace($gps_var_date,'/',2,0); // ajout slash sur la date

$gps_var_longitude = strval(intval(substr($gps_trame_array[5],0,3))).'°'; // on recupere les 3 premiers
nb pour les degrés, on enlève les 0 en trop en passant la string en int
$gps_var_longitude = $gps_var_longitude . substr($gps_trame_array[5],3,2).'\''; // récupération des
minutes
$gps_var_longitude = $gps_var_longitude .
strval(intval(intval(substr($gps_trame_array[5],6,2))/100*60)).'",'; // conversion des degrés décimales en minutes
$gps_var_longitude = $gps_var_longitude . $gps_trame_array[6];
if (intval(substr($gps_trame_array[5],0,3)) > 0) // si le nombre est positif on rajoute un + pour le
format physique
        {$gps_var_longitude = '+'.$gps_var_longitude;}

$gps_var_latitude = strval(intval(substr($gps_trame_array[3],0,2))).'°';
$gps_var_latitude = $gps_var_latitude . substr($gps_trame_array[3],2,2).'\'';
$gps_var_latitude = $gps_var_latitude .
strval(intval(intval(substr($gps_trame_array[3],5,2))/100*60)).'",';
$gps_var_latitude = $gps_var_latitude . $gps_trame_array[4];

$gps_var_time = substr($gps_trame_array[1],0,6);
$gps_var_time = substr_replace($gps_var_time,':',4,0); // on met : de séparation
$gps_var_time = substr_replace($gps_var_time,':',2,0); // on met : de séparation
$gps_var_hour = intval(substr($gps_var_time,0,2))+1; // recuperation heure en integer
if ($gps_var_hour > 23) // on rajoute une heure pour UTC+1
        {$gps_var_hour = 0;}
if ($gps_var_hour < 10) // on rajoute un zero si un seul chiffre, on remet en string
        $gps_var_hour = '0'.strval($gps_var_hour);
$gps_var_time = $gps_var_hour . substr($gps_var_time,2,7); // on rajoute l'heure en concatenant
```

Résultat final :



5. Création du programme de calcul en JavaScript

On choisira d'effectuer le calcul du coté client à l'aide d'un script JavaScript pour permettre d'économiser la puissance du serveur.

Déclaration des variables des données nécessaires ainsi que leur description en commentaire :

```
/* Liste des données */

// Données équatoriales //
var timed;       // Temps reçu
var SS;          // Seconde
var MM;          // Minute
var HH;          // Heure
var HHf;         // Heure décimale
var date;        // Date reçu
var year;        // Année
var month;       // Mois
var day;         // Jour
var JJ;          // Jour julien
var J2000;       // Jours depuis 2000
var Lat_dms;     // Latitude degré minute seconde
var Lat_dd;      // Latitude degré décimal
var Long_dms;    // Longitude degré minute seconde
var Long_dd;     // Longitude dégré décimal
var LST_dd;      // Temps sidéral local degré décimal
var HA;          // Angle horaire
var HA_polaire; // Angle horaire étoile polaire

// Données astrales //
var RA_hms;      // Ascencion droite heure minute seconde
var RA_hd;       // Ascencion droite heure décimal
var RA_dd;       // Ascencion droite degré décimal
var RA_polaire; // Ascencion droite étoile polaire degré décimal
var Dec_dms;     // Déclinaison degré minute seconde
var Dec_dd;      // Déclinaison degré décimal
var Dec_polaire;// Déclinaison étoile polaire degré décimal
var Az;          // Azimut
var Alt;         // Hauteur
var Az_polaire; // Azimut étoile polaire
var Alt_polaire;// Hauteur étoile polaire
var DeltaAz;     // Delta azimut
var DeltaAlt;    // Delta hauteur

// Variables pour le calcul //
var a;
var b;
var y;
var m;
```

Création d'un bouton pour appeler une fonction qui se chargera de faire le calcul et d'afficher le résultat.

Un traitement des données récupérés sur le site avec Javascript est nécessaire pour isoler les valeurs :

```
    // Import des données //
    timed = document.getElementById("var_time").innerText;
    SS = parseInt(timed.substring(6,8)); // chaine vide les 10 premiers char
    MM = parseInt(timed.substring(3,5));
    HH = parseInt(timed.substring(0,2));
    date = document.getElementById("var_date").innerText;
    year = parseInt(date.substring(6,8))+2000;
    month = parseInt(date.substring(3,5));
    day = parseFloat(date.substring(0,2));
    Long_dms = document.getElementById("var_long").innerText;
    Lat_dms = document.getElementById("var_lat").innerText;
    RA_hms = document.getElementById("ascention_i").value;
    Dec_dms = document.getElementById("declinaison_i").value;
    RA_polaire = 37.954600;
    Dec_polaire = 89.264100;

    // Traitement des données //
    HHf = HH + MM/60 + SS/3600; // Heure décimale
    Lat_dd = parseFloat(Lat_dms.substring(0,2)) + parseFloat(Lat_dms.substring(3,5))/60 + parseFloat(Lat_dms.substring(6,8))/3600;
    if (Lat_dms.substring(10,11) == 'S') {Lat_dd = Lat_dd * -1;} // Si sud, alors latitude négatif
    Long_dd = parseFloat(Long_dms.substring(0,3)) + parseFloat(Long_dms.substring(4,6))/60 + parseFloat(Long_dms.substring(7,9))/3600;
    if (Long_dms.substring(11,12) == 'W') {Long_dd = Long_dd * -1;} // Si ouest, alors longitude négatif
    RA_hd = parseFloat(RA_hms.substring(0,2)) + parseFloat(RA_hms.substring(3,5)/60) + parseFloat(RA_hms.substring(6,11))/3600;
    RA_dd = (RA_hd*15).toFixed(6);
    Dec_dd = parseFloat(Dec_dms.substring(0,3)) + parseFloat(Dec_dms.substring(4,6)/60) + parseFloat(Dec_dms.substring(7,12))/3600;
```

Les types doivent être conservés en tant qu'entier ou nombre flottant afin de ne pas poursuivre un calcul avec une string.

Des découpages sur chaque valeurs string sont nécessaires et une conversion avec ParseFloat() ou ParseInt() permet l'isolation flottante de la valeur.

Note : l'utilisation de fonctions d'arrondis pour réduire le nombre de décimales peuvent retourner la valeur flottante en string ce qui provoquera un problème pour la suite. On choisira d'utiliser ToFixed() pour obtenir un nombre flottant en sortie.

Des conversions sont nécessaires sur les coordonnées en degré/heure minute secondes grace à des calculs simples ((h+m/60+s/3600)/15 etc.)

Après résolution de nombreux problèmes de types, on affichera la variable grace à un élément marqué par un ID sur la page web qui permettra son unicité on utilisera
Pour les formulaires :
document.getElementById("myIDDD").innerHTML;
Pour le reste :
document.getElementById("myIDDD").innerHTML;
Pour récupérer aussi la balise html :
document.getElementById("myIDDD").outerHTML;

```
function check360(number)
{

    while (number>360 || number<0)
    {

        if (number>360)
        {
            number = number - 360;
        }
        else if (number<0)
        {
            number = number + 360;
        }
    }

    return number;
}
```

Une fonction est créé pour ramener les valeurs dans la plage de 0 à 360°

Note : L'utilisation d'un modulo 360 sera plus optimisé pour la suite

création fonction plage 360
fonction degrée radians pour Math
paramètres radians
nouveau algo jj
feuille calcul

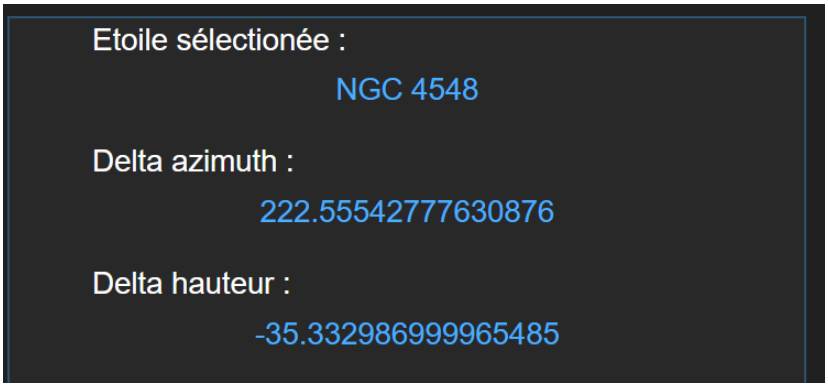
Afin de calculer Azimut et hauteur, on utilisera la bibliothèque Math permettant d'utiliser les fonctions trigonométriques en radians seulement, on aura alors :

```
23  function toDegrees(angle)
24  {
25      return angle*(180/Math.PI);
26  }
27
28
29
30
31  function toRadians(angle)
32  {
33      return angle*(Math.PI/180);
34  }
```

Pour le calcul de delta azimut et delta hauteur :

```
Alt = toDegrees( Math.asin( Math.sin(toRadians(Dec_dd))*Math.sin(toRadians(Lat_dd))+Math.cos(toRadians(Dec_dd))*Math.cos(toRadians(Lat_dd))*Math.cos(toRadians(HA)) ) );
Az = toDegrees( Math.acos ( (Math.sin(toRadians(Dec_dd)) - Math.sin(toRadians(Alt))*Math.sin(toRadians(Lat_dd)))/(Math.cos(toRadians(Alt))*Math.cos(toRadians(Lat_dd))) ) );
if (Math.sin(toRadians(HA)) > 0)
    Az = 360 - Az;
Alt_polaire = toDegrees( Math.asin( Math.sin(toRadians(Dec_dd))*Math.sin(toRadians(Lat_dd))+Math.cos(toRadians(Dec_dd))*Math.cos(toRadians(Lat_dd))*Math.cos(toRadians(HA_polaire)) ) );
Az_polaire = toDegrees( Math.acos ( (Math.sin(toRadians(Dec_dd)) - Math.sin(toRadians(Alt_polaire))*Math.sin(toRadians(Lat_dd)))/(Math.cos(toRadians(Alt_polaire))*Math.cos(toRadians(Lat_dd))) ) );
if (Math.sin(toRadians(HA_polaire)) > 0)
    Az_polaire = 360 - Az_polaire;

DeltaAlt = Alt - Alt_polaire;
DeltaAz = Az - Az_polaire;

document.getElementById("var_deltaalt").innerHTML = DeltaAlt;
document.getElementById("var_deltaaz").innerHTML = DeltaAz;
```

On affiche bien le résultat du calcul :



Afin de faciliter le mouvement du télescope on ramène dans une plage de 180° pour accélerer le processus (pour 222 on aura donc -138, cela signifie qu'on ira plus rapidement dans le sens anti-horaire)

Note : L'utilisation d'une étoile pendant la saison actuelle est conseillé puisqu'elle permettra d'être dans une delta hauteur observable (entre 0 et 90°)

6. Création d'une fiche de données sur l'IHM sur le calcul

Après de nombreux problèmes rencontrés, pour les résoudre et afin de faciliter l'affichage des données pour l'utilisateur ou le développeur on rajoute une div caché en arrière plan affichable grâce un bouton. Celle ci affichera un tableau de l'ensemble des valeurs utilisées pour le calcul :

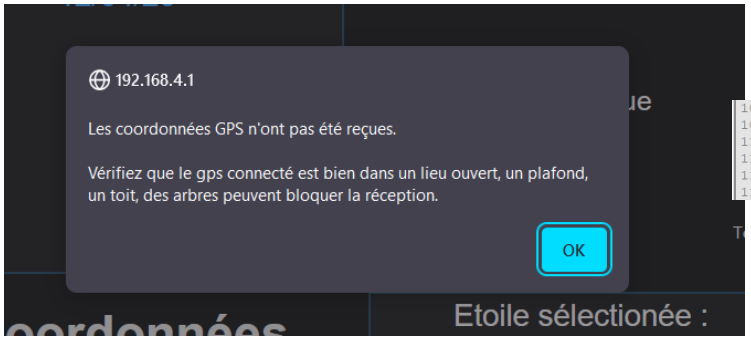
Chaque valeur affiché est composé d'un ID permettant l'ajout sur la page web depuis Javascript :

```
function showMe()
{
    document.getElementById("wrapper_fly").style.display = "block";

    document.getElementById("var_timed").innerHTML = timed;
    document.getElementById("var_date2").innerHTML = date;
    document.getElementById("var_HHf").innerHTML = HHf;
    document.getElementById("var_a").innerHTML = a;
    document.getElementById("var_b").innerHTML = b;
    document.getElementById("var_y").innerHTML = y;
    document.getElementById("var_m").innerHTML = m;
    document.getElementById("var_JJ").innerHTML = JJ;
    document.getElementById("var_J2000").innerHTML = J2000;
    document.getElementById("var_Lat_dms").innerHTML = Lat_dms;
    document.getElementById("var_Lat_dd").innerHTML = Lat_dd;
    document.getElementById("var_Long_dms").innerHTML = Long_dms;
    document.getElementById("var_Long_dd").innerHTML = Long_dd;
    document.getElementById("var_LST_dd").innerHTML = LST_dd;
    document.getElementById("var_HA").innerHTML = HA;
    document.getElementById("var_HA_polaire").innerHTML = HA_polaire;
    document.getElementById("var_RA_hms").innerHTML = RA_hms;
    document.getElementById("var_RA_hd").innerHTML = RA_hd;
    document.getElementById("var_RA_dd").innerHTML = RA_dd;
    document.getElementById("var_RA_polaire").innerHTML = RA_polaire;
    document.getElementById("var_Dec_dms").innerHTML = Dec_dms;
    document.getElementById("var_Dec_dd").innerHTML = Dec_dd;
    document.getElementById("var_Dec_polaire").innerHTML = Dec_polaire;
    document.getElementById("var_Az").innerHTML = Az;
    document.getElementById("var_Alt").innerHTML = Alt;
    document.getElementById("var_Az_polaire").innerHTML = Az_polaire;
    document.getElementById("var_Alt_polaire").innerHTML = Alt_polaire;
    document.getElementById("var_DeltaAz").innerHTML = DeltaAz;
    document.getElementById("var_DeltaAlt").innerHTML = DeltaAlt;
}


function closeMe()
{
    document.getElementById("wrapper_fly").style.display = "none";
}
```

7. Correction des bugs et optimisation de l'utilisation.

7.1. Impossible de réceptionner les données GPS

Lorsque le GPS ne fonctionne pas faute de non réception (ciel inaccessible, endroit clos empêchant la réception satellitaire), on reçoit alors des coordonnées vide :
0°'0",
°'0",
On gérera cet exception dans JavaScript (on bloque la réception GPS)




```
108     if ((Lat_dms.includes("0°'") == true) || (Long_dms.includes("°'") == tru
109     {
110         Lat_dms = "00°00'00\",N ";
111         Long_dms = "000°00'00\",E ";
112         alert("Les coordonnées GPS n'ont pas été reçues.\n\nVérifiez que le
113     }
```

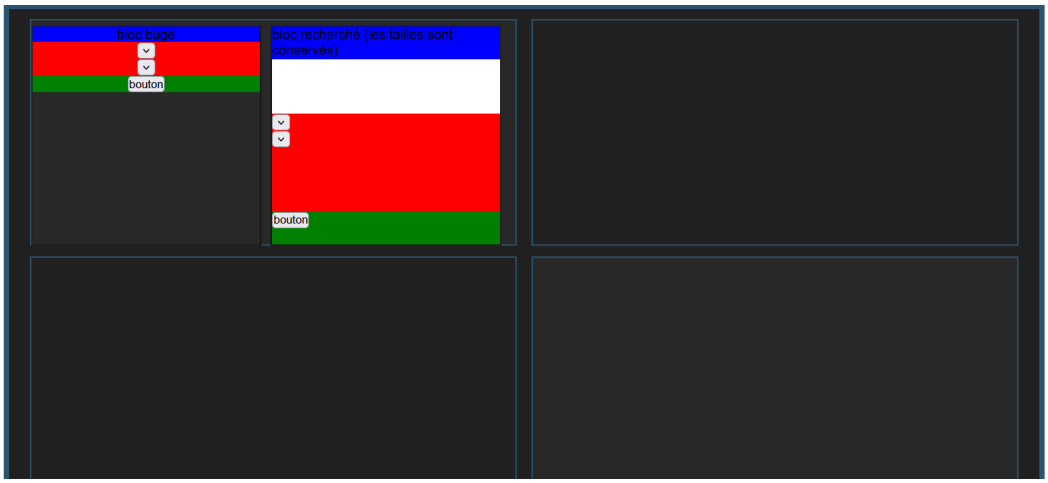
7.2. Tri alphabétique des objets de Messier affichés

En effet, il s'agira de faire une requête plus élaboré pour afficher les objets de Messier dans l'ordre numéroté. Cependant il y a une lettre char afin des nombres qui ne permet pas ce tri facilement.
On isolera ces nombres et on les triera de la sorte :
SELECT Messier FROM catalogue_messier ORDER BY CAST(SUBSTR(Messier, 2) AS INTEGER) ASC;
On arrive à un tri numérique :

7.3. Les tailles des div ne fonctionnent plus dans un form

Les div sont définies par des tailles proportionnel à la taille de la page, hélas lorsqu'elles sont encadrés par une balise <form> on remarque que ces tailles ne sont plus respectés.

Après d'amples recherches et l'isolation du problèmes sur une page il s'agit d'un problème d'affichage qui peut être testé en css avec display pour changer l'ordre et la façon de présenter l'objet sur la page. On utilisera alors display: grid;



7.3. Migration des travaux sur un ordinateur lambda

Afin de pouvoir ouvrir les travaux réalisés dans cette fiche sans nécessiter du matériel du télescope, on prévoit de pouvoir l'ouvrir sur un ordinateur lambda. On prendra pour exemple la dernière version de Wamp qui utilise cependant des versions anciennes de PHP et de Mysql.

- Pour commencer on doit s'assurer que l'utilisateur root a bien le même password que le système, ou bien changer le mot de passer dans le code PHP.
- Puisque nous n'aurons probablement pas de GPS, il faudra gérer l'erreur du fichier introuvable de gestion du GPS :

```php
exec("python3 /home/pi/gps/gps.py",$gps_result); // envoie un array des print réalisés
if (empty($gps_result)) // si il n'y a pas de gps branché
{
    $gps_result[0] = "nop,,,,,,,,,,,,,,,,,,,";
}
 if ($gps_trame == "nop,,,,,,,,,,,,,,,,,,,") // si le gps n'est pas detecté
 {
    $gps_var_date = $gps_var_longitude = $gps_var_latitude = $gps_var_time = "GPS NON DETECTE";
 }
```

- Pour faciliter la détection d'erreur on gérera davantage les exceptions dans la fonction gérant chaque requête de la DB effectué pour détailler les problèmes rencontrés.
- Recréation de certaines requêtes SQL qui utilisaient des fonctionnalités non présente dans les anciennes versions pour utiliser des opérations plus standards :

```php
$request = "SELECT Name,Const FROM catalogue_messier ORDER BY CAST(SUBSTR(Name, 2) AS UNSIGNED) ASC;";
```

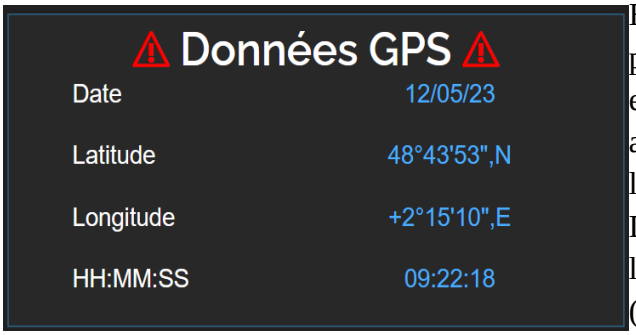
(on remplacera par exemple INTEGER par UNSIGNED)

- Après de nombreuses erreurs et recherches il apparaît que l'isset semble poser problème car plus réstrictif sur l'ancienne version de PHP, on sortira le code de vérification de l'existence des variables du bloc :

```php
if (isset($_GET['objectList'])) // (isset vérifier qu'un objet est selectionné, si c'est sélectionner alors)
{
    $option_objectList = $_GET['objectList'];

    $Telescope_bdr = new BD_communication();
    $request = "SELECT `RA`,`Declinaison`,`Name` FROM `catalogue_messier` WHERE (Name = '".$option_objectList."')
    UNION SELECT `RA`,`Declinaison`,`Name` FROM `catalogue_ngc` WHERE (Name = '".$option_objectList."')
    UNION SELECT `RA`,`Declinaison`,`Name` FROM `catalogue_ic` WHERE (Name = '".$option_objectList."');";
    $result = $Telescope_bdr->connexion($request);
    $row = $result->fetch_assoc();
}
$value_ra = $row["RA"] ?? ""; // renvoie "" si rien trouvé dans la db
$value_d = $row["Declinaison"] ?? "";
$value_catalogue = $db_select1 ?? "";
$value_name = $row["Name"] ?? "";
$value_const = $db_select3 ?? "";
```

- Dans le script JavaScript, l'ancienne version pour `$result→fetch_assoc();` récupérait également la colonne, on s'occupera ainsi de cacher les 3 lignes "Name" de chaque catalogue.

7.4. Gérer l'absence de GPS



En cas de problèmes avec le GPS, deux erreurs sont possibles : le GPS n'est pas détecté ou une erreur est survenu. Glisser le curseur sur le titre fera apparaître un message d'erreur informant l'utilisateur de la résolution possible.
Des données sont ajoutés, l'heure et la date depuis la Raspberry ainsi que des coordonnées par défaut (Lycée de Vilgenis)

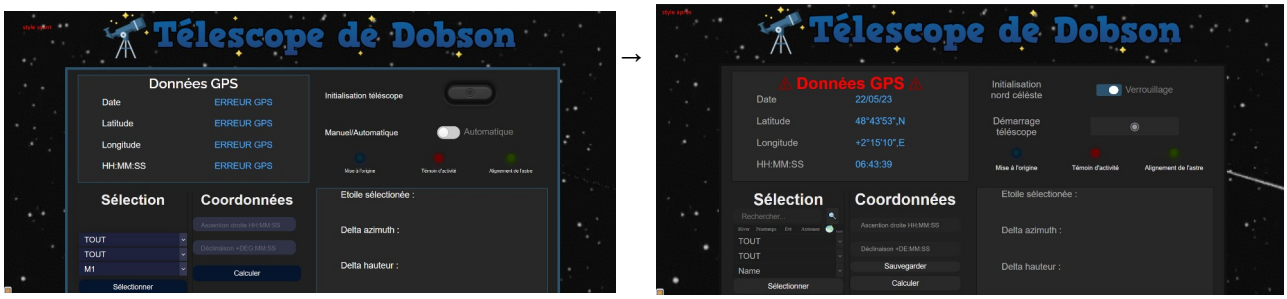
7.4. Rendre le site web responsive et + ergonomique

Après la mise en forme du site avec des unités non influençable, on choisira d'adapter la taille de la page en fonction de la taille de l'écran, pour cela on re-modélisera tout les objets suivants avec des mesures relatives (vw, vh, %, em) en faisant de nombreux test :

- Les boutons
- Les switchs
- Les textes
- Les div, span et formulaires
- Les types d'affichage (flex, grid, etc)
- Les input, select
- Les objets créés (LED)

Afin de rendre l'utilisation de l'IHM plus intuitive, on choisira de :

- Développer une nouvelle mise en forme avec des couleurs dans un thème similaire, on évitera de choisir des couleurs trop différentes
- On éliminera les couleurs mis en évidence pour rien (bord du site, des divs en bleu, titres en bleu etc), on appliquera un thème grisâtre sur l'ensemble des divs et un thème bleu pour les valeurs rentrés par l'utilisateur : textes, select, bouton, placeholder et label doivent être changés + disposition des objets sur le site.



8. Communication avec ESP et format de transmission

L'IHM hébergé sur la Raspberry devra affiché les informations reçu par l'ESP et envoyer les interactions de l'utilisateur sur celui ci. Le protocole utilisé sera TCP afin de privilégier l'état de la donnée transmise.

ENVOI CLIENT RASPBERRY →SERVEUR ESP

L'ESP devra recevoir les interactions de l'utilisateur via l'envoi d'une chaîne de caractère dans le format suivant :

action,données

L'action est un nombre entre 0 et 99 permettant d'identifier le besoin de l'utilisateur, les données transmises seront donc envoyés selon celui ci :

- 00 : Réinitialisation du télescope (bouton STOP)
  Données → NULL
- 01 : Démarrage du télescope en mode manuel (action des codeurs)
  Données → "<nombre_pas_deltaAz>;<nombre_pas_deltaAlt>"
- 02 : Démarrage du télescope en mode automatique (action des moteurs)
  Données → "<nombre_pas_deltaAz>;<nombre_pas_deltaAlt>"

RÉPONSE SERVEUR ESP → CLIENT RASPBERRY

L'ESP devra répondre au client en renvoyant les informations nécessaires.

données

- 00 : Réinitialisation du télescope (bouton STOP)
  Données → renvoie 1 lorsque le télescope est arreté.
- 01 : Démarrage du télescope en mode manuel (action des codeurs)
  Données → "<nombre_pas_restant_deltaAz>;<nombre_pas_restant_deltaAlt>"
- 02 : Démarrage du télescope en mode automatique (action des moteurs)
  Données → "<nombre_pas_restant_deltaAz>;<nombre_pas_restant_deltaAlt>;<erreur>"
  <erreur> : 1 si mouvement parasite, 0 si correct

ENVOI RASPBERRY DEPUIS ESP
On notera un format d'échange de l'ESP à la Raspberry de 9 octets :
temps,décomptage1,décomptage2,numéro_téléscope

- *ASCII → 6 octets : Temps : pour coder l'heure la minute et la secondes*
- *Entier non signé → 2 octets : Décomptage1 nombre pas pour coder jusqu'à 360 possibilités*
- *Entier non signé → 2 octets : Décomptage2 nombre pas pour coder jusqu'à 360 possibilités*
- *Entier non signé → 1 octet : codage du numéro télescope*

ENVOI ESP DEPUIS RASPBERRY
L'ESP devra recevoir les interactions de l'utilisateur via un nombre transmis sur un octet.
nombre,deltaaz,deltaalt

- *Entier non signé → 1 octet : nombre correspondant à l'action à effectuer*
- *Entier non signé → 2 octets : nombre pas delta azimut*
- *Entier non signé → 2 octets : nombre pas delta hauteur*

9. Communication avec l’ESP : sockets

On prépare la trame a envoyé avec javascript : coordonnées, action à réaliser selon le format nécessaire de la page ci dessus. On la transmet a un code client de socket en PHP pour l'envoyer au serveur ESP qui va traiter la trame en conséquence. Celui ci enverra les informations nécessaire (par exemple le décomptage de pas restant) en réponse et sera transmis sur le client PHP qui devra renvoyer à l’IHM à l’aide d’Ajax.

10. Ajout de catalogues dans la base de données

10.1. Préparation et ajout des catalogues dans la base de donnée

Dans la table du catalogue de Messier ajouté on peut remarqué qu'il y a une colonne répertoriant ces objets dans un catalogue plus vaste encore : le catalogue NGC (=New General Catalogue).

Après des recherches pour obtenir ce catalogue mis à jour récememnt, une table composé de ce catalogue et de l'IC (Index Catalogue), un supplément au catalogue NGC a été trouvé sur GitHub.
https://raw.githubusercontent.com/mattiaverga/OpenNGC/master/database_files/NGC.csv
On choisira de séparer ces deux catalogues dans deux tables csv et de mettre les mêmes noms de colonnes (on actualisera le catalogue de Messier récupérer auparavant).

On ajoutera dans la table du catalogue de Messier les abréviations des 88 constellations manquantes pour avoir les mêmes colonnes.

On accède à http://192.168.4.1/phpmyadmin/ et on supprime la table pour rajouter les 3 nouveaux catalogues.

Après de nombreuses erreurs de permissions (suppression manuel de la DB et restauration de la table de Messier), de problèmes d'importations à cause de données ne passant pas sur phpmyadmin, on supprime les dizaines de colonnes inutiles des catalogues et on les importe. Des modifications sont faites dans le code PHP pour mettre à jour les requêtes SQL et les colonnes de chaque table seront renommés pour avoir des noms similaires.

Pour l'affichage sur l'IHM, 3 balises select seront mis en place :

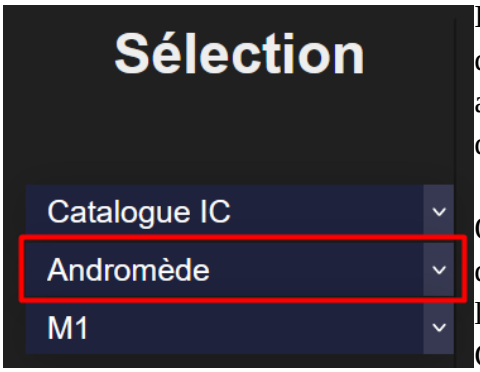
1. Sélection du catalogue
2. Sélection de la constellation (filtre)
3. Sélection de l'astre

La première étape est d'importer les catalogues sur l'IHM, on doit afficher les astres à partir du choix de l'utilisateur. A noter que le terme "selected" ajouté dans la balise permet d'afficher l'option par défaut de l'utilisateur pour afficher son choix précédent comme étant sélectionné à l'actualisation de la page (on garde une trace).

Pour une question d'ergonomie, la sélection de l'utilisateur devra actualiser les résultats des astres proposés sans qu'il y est besoin de cliquer sur un bouton. On commence par réaliser la sélection du catalogue avec le bouton puis une question doit se poser : faut-il réaliser l'actualisation coté serveur ou coté client. Toujours pour économiser des ressources serveurs et pour limiter le temps d'affichage de l'utilisateur on choisira d'importer l'ensemble des noms des 3 catalogues qui sera visible avec tout. Ainsi, JavaScript détectera l'événement de la sélection et cachera les éléments non demandés à l'utilisateur à l'aide d'une balise CSS. Ci dessous un extrait :

```
229  function maskSomeItems()
230  {
231      var selectCatalogue = document.getElementById("db_select1");
232      var selectItems = document.getElementById("db_select2");
233      var catalogue_choice = db_select1.value;
234
235
236      var list_of_items = db_select2.options;
237
238
239      // afficher tout options items
240      if (catalogue_choice == "all")
241      {
242          for (var i=0; i<selectItems.options.length; i++)
243          {
244              selectItems.options[i].style.display = "block";
245          }
246      }
247
248      // masquer des options items que l'utilisateur n'a pas besoin
249      if (catalogue_choice == "catalogue_messier")
250      {
251          for (var i=0; i<list_of_items.length; i++)
252          {
253              var item = list_of_items[i].textContent;
254              if (item.startsWith("IC") || item.startsWith("NGC"))
255              {
256                  list_of_items[i].style.display = "none";
257              }
258              else
259              {
260                  list_of_items[i].style.display = "block";
261              }
262          }
263      }
264
265      if (catalogue_choice == "catalogue_ngc")
266      {
267          for (var i=0; i<list_of_items.length; i++)
268          {
269              var item = list_of_items[i].textContent;
270              if (item.startsWith("IC") || item.startsWith("M"))
271              {
272                  list_of_items[i].style.display = "none";
273              }
```

10.2. Création d’un filtre de constellation selon le catalogue



L’importation d’environ 10 000 astres et l’opération d’affichage est tout de même chronophage, pour cela nous ajouterons des fonctionnalités de filtres, par exemple l’ajout d’un filtre de constellation.

On affichera les 88 constellations dans ce select et on créera de nombreuses opérations d’affichages pour faciliter l’utilisation de l’utilisateur.
Code source détaillé dans le dossier source.

Deux fonctions devront s'exécuter à chaque changement de sélection dans un ordre différent. Puisque l'objet choisi est chargé par PHP, lors de l'actualisation de la page on choisira également de garder cet objet pour se souvenir de la décision de l'utilisateur. Pour cela on utilisera select sur l'objet choisi et une condition se réalisera sur l'affichage de chaque option pour savoir quelle option a été choisi par l'utilisateur.

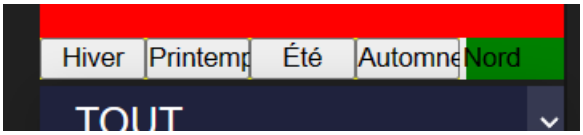
```php
while($row = $result->fetch_assoc())
{
    $selected = ($row['Name'] == $value_name) ? 'selected' : "$objectList";
    echo "<option name=\"".$row['Const']."\" value=\"".$row['Name']."\" ".$selected.">".$row['Name']."</option>";
}
$request = "SELECT Name,Const FROM catalogue_ngc ORDER BY CAST(SUBSTR(Name, 4) AS INTEGER) ASC;";
$result = $Telescope_bdr->connexion($request);
while($row = $result->fetch_assoc())
{
    $selected = ($row['Name'] == $value_name) ? 'selected' : '';
    echo "<option name=\"".$row['Const']."\" value=\"".$row['Name']."\" ".$selected.">".$row['Name']."</option>";
}
$request = "SELECT Name,Const FROM catalogue_ic ORDER BY CAST(SUBSTR(Name, 3) AS INTEGER) ASC;";
$result = $Telescope_bdr->connexion($request);
while($row = $result->fetch_assoc())
{
    $selected = ($row['Name'] == $value_name) ? 'selected' : '';
    echo "<option name=\"".$row['Const']."\" value=\"".$row['Name']."\" ".$selected.">".$row['Name']."</option>";
}
```

Cela permettra donc à l'utilisateur de retrouver l'option qu'il a choisi en premier élément de la liste des astres.

Dans JavaScript on assemblera la fonction des deux filtres après test pour gérer le changement à n'importe quel moment de ces options.

10.3. Création d'un filtre de saison et d'hémisphère

Après comparaison des types d'affichage, on choisit pour une div au dessus affiché en flex, on split en deux div, une de 80 % possédant quatre boutons pour chaque saison, une de 20 % pour un bouton correspondant au choix de l'hémisphère.



Les clics de chaque boutons sont placés dans un formulaire et font appel à des fonctions javascript qui bloque l'envoi du formulaire à php à l'aide de event.preventDefault();.

```javascript
function winter_chooseSeason()
{
    event.preventDefault(); // bouton dans le formulaire, on empeche

    saison = 1;

    restart_season_buttons();

    var bouton = document.getElementById("buttonlikeradio_winter");
    bouton.style.backgroundColor = '#8791ad';
    bouton.classList.add("active");

    chooseSeason();
}
```

En fonction de la saison choisi, on changera la valeur d'une variable globale saison correspondant au bouton cliqué : 1 hiver, 2 printemps, 3 été et 4 automne. On applique le style de base de tout les boutons, on change le style du bouton cliqué et on appelle une fonction pour le filtrage.

La fonction appellé chooseSeason() va permettre dans une première partie de filtrer les 88 constellations, en masquant les constellations qui ne sont pas observable dans la saison actuelle.

```
<div class="box">
    <select name="db_select3" id="db_select3" onchange="maskSomeItems()">
        <option value="all_const" selected>TOUT</option>
        <option value="And">Andromède</option>
        <option value="Ant">Machine pneumatique</option>
        <option value="Aps">Oiseau de paradis</option>
        <option value="Aql">Aigle</option>
```

A l'aide d'un outil de génération de code on crée un tableau répertoriant la meilleure saison d'observation pour les 88 constellations :

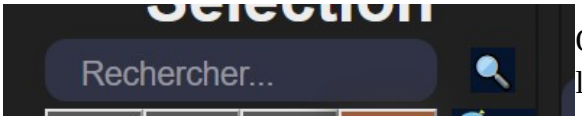
```
const list_constellations_north = [ // 1 hiver, 2 printemps, 3 été, 4 automne, 5 tout
    { abbreviation: "all_const", season: 5 },
    { abbreviation: "And", season: 4 },
    { abbreviation: "Ant", season: 0 },
    { abbreviation: "Aps", season: 2 },
    { abbreviation: "Aql", season: 3 },
```

On en créera un autre pour l'hémisphère sud.

Il sera ensuite nécessaire de filtrer les astres en fonction de la période d'observation des constellations des tableaux. Afin de ne pas provoquer des conflits d'affichage il sera important de suivre un ordre précis (on affiche tout les objets et on masque selon les filtres choisi petit à petit : catalogue > constellations > hemisphere > saison

Après mise en forme des boutons :

10.4. Création d'une barre de recherche



On ajoute une barre de recherche en html/CSS avec l'affichage flex.

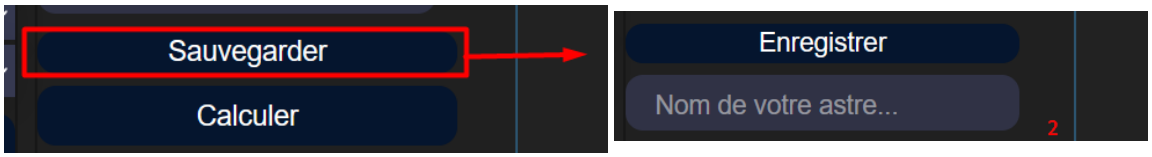
Lorsque l'utilisateur clique, on empêche l'envoi du formulaire, on récupère la valeur de l'input et ci celle-ci existe elle est sélectionné. On provoque l'envoie du formulaire et la sélection de l'astre choisi est bien importé, sinon une alerte est donné informant que l'astre n'est pas disponible (on mettra l'input de l'utilisateur en majuscule pour éviter les conflits de casse).

11. Ajout d'un catalogue personnalisé et la possibilité de sauvegarder des astres (Ajax)

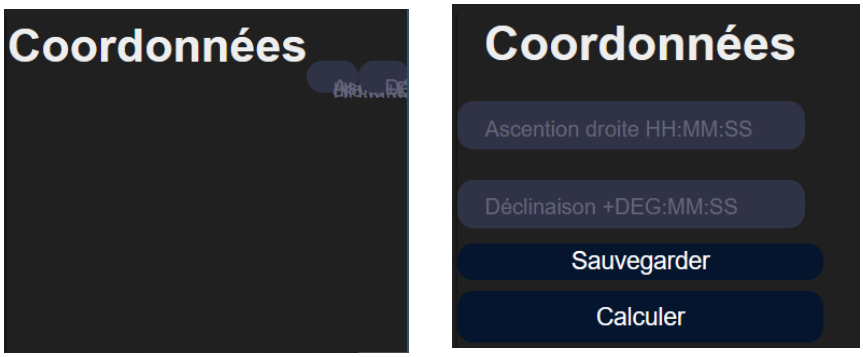
On crée une table csv pour stocker les objets personnalisés, et on l'importe dans MySql à partir de PhpMyAdmin.

	A	B	C	D
1	Name	RA	Declinaison	Const
2				
3				

On crée un nouveau bouton pour permettre à l'utilisateur de sauvegarder son objet



On doit encadrer cette partie dans un formulaire et celui ci va interférer avec les moyens d'affichage des parents (%), pour cela on appliquera un affichage en ligne (display: inline;).




```
event.preventDefault(); // Empêche le rechargement de la page

$.ajax({
    method: "GET",
    url: "save_star.php",
    dataType: "html",
    data: {

    }
})
.done(function(data) {
    alert("ok");
})
```

On implémente un code Ajax, et on affiche une alerte si cela fonctionne. Le code PHP est appelé et on prépare une requête SQL sur PhpMyAdmin pour communiquer avec la DB sur PHP, si celle ci fonctionne alors le code PHP est fonctionnel. On crée alors un code SQL pour préparer une relation d'un catalogue personnalisé :

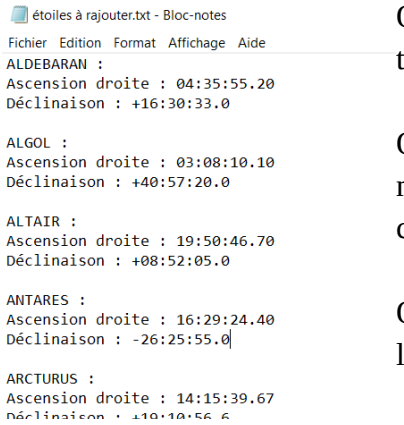
```
CREATE TABLE IF NOT EXISTS `telescope`.`catalogue_custom` (`Name` varchar(64), `RA` varchar(11), `Declinaison` varchar(11), `Const` varchar(9)) DEFAULT (
```

Puis une commande SQL pour ajouter des informations dans cette table :

```
INSERT INTO catalogue_custom VALUES ('ALDEBARAN', '04:35:55.20', '+16:30:33.0', 'all_const');
```

La prochaine étape est donc de transmettre les informations du formulaire pour ajouter des éléments personnalisés dans le catalogue.

Préparation d'un fichier texte d'exemples d'étoiles à rajouter :


étoiles à rajouter.txt - Bloc-notes

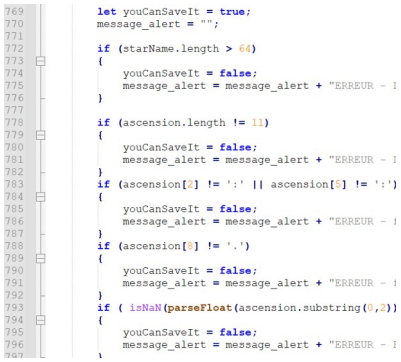
Fichier Edition Format Affichage Aide

```
ALDEBARAN :
Ascension droite : 04:35:55.20
Déclinaison : +16:30:33.0

ALGOL :
Ascension droite : 03:08:10.10
Déclinaison : +40:57:20.0

ALTAIR :
Ascension droite : 19:50:46.70
Déclinaison : +08:52:05.0

ANTARES :
Ascension droite : 16:29:24.40
Déclinaison : -26:25:55.0

ARCTURUS :
Ascension droite : 14:15:39.67
```

On récupère ensuite les données de l'IHM en javascript puis on les transmet dans le bloc Ajax.

On reçoit ces données dans le fichier save_star.php grâce à la méthode GET, et on peut personnaliser l'ajout de l'étoile dans le catalogue.

On traite un minimum les données reçues pour ajouter sans erreur l'étoile de l'utilisateur, on applique des filtres :



```
let youCanSaveIt = true;
message_alert = "";

if (starName.length > 64)
{
    youCanSaveIt = false;
    message_alert = message_alert + "ERREUR - 1
}

if (ascension.length != 11)
{
    youCanSaveIt = false;
    message_alert = message_alert + "ERREUR - 1
}
if (ascension[2] != ':' || ascension[5] != ':')
{
    youCanSaveIt = false;
    message_alert = message_alert + "ERREUR - 1
}
if (ascension[8] != '.')
{
    youCanSaveIt = false;
    message_alert = message_alert + "ERREUR - 1
}
if ( isNaN(parseFloat(ascension.substring(0,2))
{
    youCanSaveIt = false;
    message_alert = message_alert + "ERREUR - 1
```

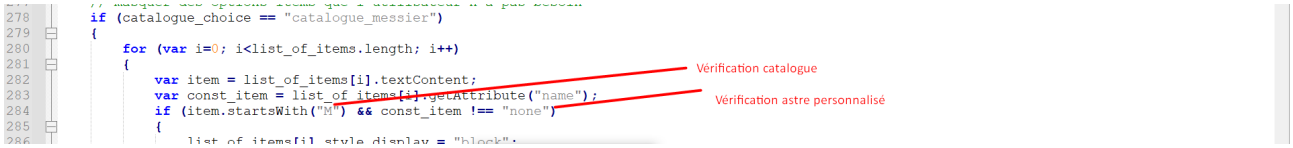
- Nom en majuscule
- Limite de caractères sur le nom de l'étoile
- Limite de caractères sur les coordonnées
- Contrôle des et indication des erreurs de formatage (point virgules, virgule, signe)
- Vérification des valeurs numériques découpés

Afin de ne pas rentrer en conflit avec des noms étoiles déjà présents dans les catalogues astronomiques basiques, on vérifiera que le nom est unique en améliorant la requête SQL :

```
INSERT INTO catalogue_custom (Name, RA, Declinaison, Const)
SELECT '$starName', '$ascension', '$declinaison', 'none'
FROM dual
WHERE NOT EXISTS (
        SELECT * FROM catalogue_custom WHERE Name = '$starName'
) AND NOT EXISTS (
        SELECT * FROM catalogue_messier WHERE Name = '$starName'
) AND NOT EXISTS (
        SELECT * FROM catalogue_ic WHERE Name = '$starName'
) AND NOT EXISTS (
        SELECT * FROM catalogue_ngc WHERE Name = '$starName'
```

On indiquera à l'utilisateur que l'étoile existe déjà par le biais d'une vérification JavaScript, à commencer par ajouter les étoiles du catalogue personnalisé dans la liste des astres observables dans le bloc de sélection de l'IHM web.

Pour cela on rajoute le choix du catalogue personnalisé, et on affiche chaque objet par le biais d'une requête comme les autres catalogues. La modification des filtres est nécessaire : on doit mettre dans chaque condition de choix du catalogue la vérification des astres personnalisés, on peut les reconnaître grâce à une constellation "none" qui leur est donné dans la balise select. Ainsi il nous faut optimiser les conditions pour qu'elles soient plus rapides (les conditions sont maintenant plus compact) et on instaure les nouvelles vérifications.



Il ne reste plus qu'à effectuer la vérification de la présence du nom de l'astre dans la liste des objets sur l'IHM.

Pour cela on réalise une boucle javascript qui compare l'ensemble des options avec l'objet à rajouter, l'utilisateur est informé en plus des erreurs de format précédent en cas de conflit de noms.

On ajoutera une option à l'utilisateur pour pouvoir supprimer l'étoile du catalogue, pour cela on détecte si les coordonnées sont vides, un message de confirmation lui sera alors demandé pour savoir si il veut bien supprimer l'étoile.

Des tests SQL ont été réalisés sur le site et sur PhpMyAdmin. Si un problème survient avec la sauvegarde il est possible d'ouvrir le fichier de sauvegarde seul save_star.php pour regarder l'erreur PHP. Des valeurs par défaut seront alors ajoutés à la place de la réception Ajax pour éviter des erreurs de variables introuvables :

```php
// récupération des données envoyé par Ajax
if (isset($_GET['starName']))
{
    $ascension = $_GET["ascension"];
    $declinaison = $_GET["declinaison"];
    $starName = $_GET["starName"];
}
else // si on execute le fichier a part, pas de données importés par ajax donc on les crée nous mêmes
{
    // $ascension = "14:15:39.67";
    // $declinaison = "-14:15:39.6";
    $ascension = "";
    $declinaison = "";
    $starName = "test";
}
```