

Documentations Projet

SPACE LEARNER



SOMMAIRE

1- Doc IHM

I - Contexte

II - Personas

III - Sketchs

IV - Story-board

V - Diagramme de cas d'utilisations

VI - Choix de l'ergonomie justifiée

VII - Prise en compte de l'accessibilité

2 - Doc Conception Orienté Objet

I - Diagramme de paquetage

II - Diagramme de classe

III - Diagramme de classe orienté

IV - Diagramme de séquence

V - Description de l'architecture

VI - Vidéo du projet

1- Doc IHM

I - Contexte

L'espace étant un sujet vaste et encore plein de secret, nous avons trouvé pertinent de construire une application autour de ce sujet. Nous voulons mettre en avant la beauté de l'univers et vous donner le goût de vouloir en découvrir plus et vous faire rêver sur la magie de l'espace. Pour cela vous allez pouvoir découvrir des milliers d'astres que vous pourrez consulter à tout moment.

Cette application a pour but de faire découvrir l'espace à un néophyte. Elle permettra à n'importe qui de n'importe quel âge de découvrir l'espace peu importe le niveau de connaissance qu'il a sur le sujet et proposera de manière intuitive de pouvoir découvrir des astres ou de se divertir. Les différents astres présents sur l'application seront variés, vous aurez donc le choix ! Si vous vous intéressez seulement au trou noir vous en aurez ! Vous aurez la possibilité de voir tous les trous noirs entrés sur l'application et ainsi pouvoir tous les consulter ! Même si l'application est conçue pour faire découvrir l'espace de manière générale, sans prérequis. Si vous voulez donc accéder à tous les astres sans préférences, aucun problème.

L'utilisateur pourra gérer les astres de l'application. En effet, il pourra accéder à une gestion avancée qui lui permettra d'ajouter ou de modifier les astres de l'application, ce qui lui permettra de posséder le contrôle complet du contenu de l'application, il pourra être responsable des informations renseignées. Grâce à ce fonctionnement, on peut imaginer un fonctionnement communautaire, où chaque personne peut ajouter des informations à l'application, ce qui pourrait rendre l'application extrêmement fiable et posséder une infinité de données à jour.

Suite à l'étude du marché, nous avons remarqué qu'il n'existe pas d'application en rapport à l'espace aussi complète, proposant autant de fonctionnalités. Les applications portant sur le même sujet sont souvent payantes et se contentent uniquement de montrer les astres sous forme de carte 2d ou 3d. Généralement les applications concurrentes n'ont que très peu d'intérêt pédagogique, elles ont donc uniquement un public connaisseur.

Le public visé est donc très divers mais il aura comme point commun d'être intéressé à l'astronomie.

II - Personnas



Rémi

16 ans, lycéen

“L’espace m’as intéressé depuis toujours”

Aisance numérique : ★★★★★

Expertise domaine : ★★☆☆☆

Fréquence d’usage : ★★★★★

Rémi est en Terminal Scientifique. Il est très intéressé par l’astrologie depuis toujours, du peu qu’il connaît il trouve le domaine extrêmement passionnant. En sortant des cours, il adore passer des heures entières à apprendre des choses à propos de l’espaces en se documentant sur internet..

☒ Buts clés

- Être astronaute
- Apprendre des choses sur l’espace
- Obtenir son bac

Personnalité

- Sportif
- Aime les jeux vidéos
- Aime apprendre



Christian

42 ans, père de famille

“ Mon rêve est d’aller dans l’espace ! “

Aisance numérique : ★☆☆☆☆

Expertise domaine : ★★★★★

Fréquence d’usage : ★★★★★

Christian est père de 3 enfants qui passent leur temps sur les jeux vidéo. Christian est passionné par l’espace et passe la majorité ces week-end à observer l’espace. Il aimerait partager sa passion avec ses enfants et ainsi les faire sortir de leurs bulle.

☑ Buts clés

- Faire plaisir à sa famille
- Apprendre des choses à ses enfants
- Être intelligent

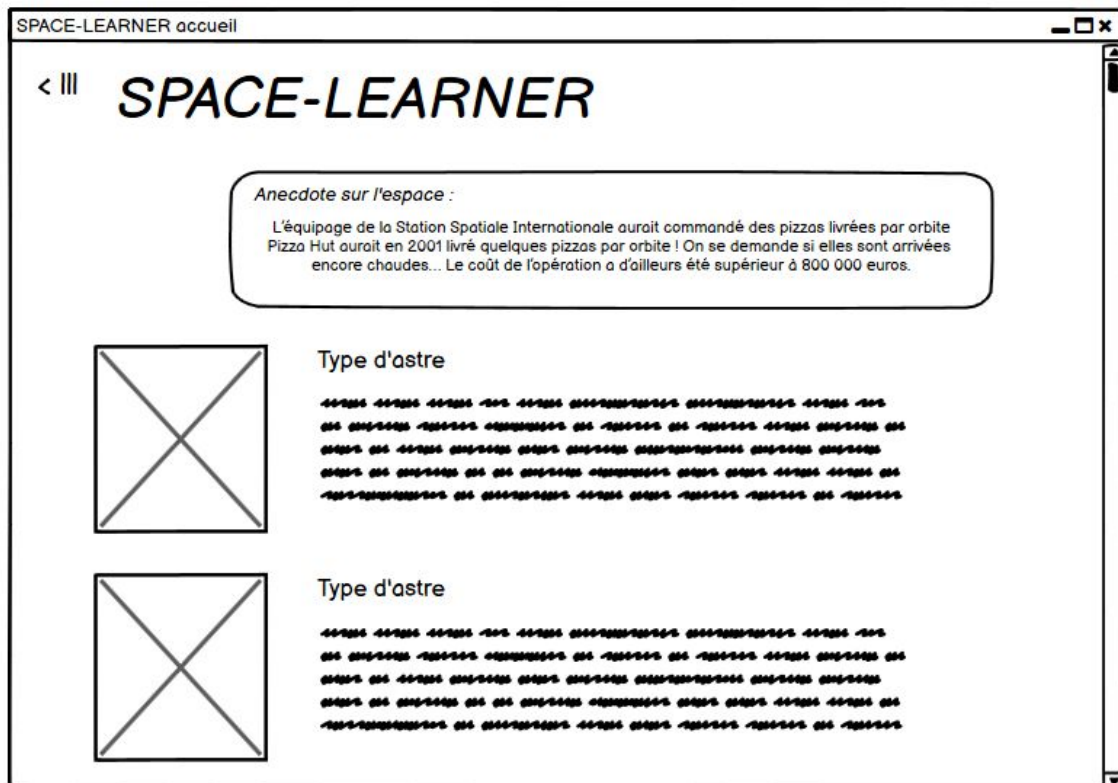
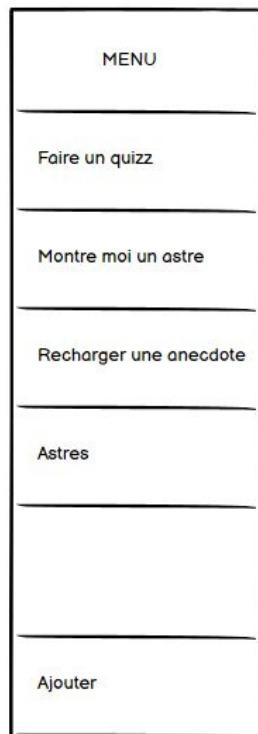
😊 Personnalité

- Papa cool
- Aime passer du temps dehors la nuit
- Aime apprendre

III - Sketchs

Menu

page d'accueil



Recherche avancée dans la liste des astres

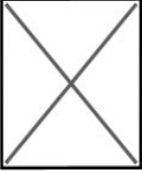
SPACE-LEARNER accueil

< |||

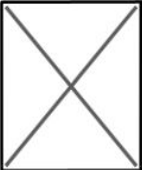
SPACE-LEARNER

Q Rechercher


Trier par



Nom de l'astre



Nom de l'astre



Nom de l'astre

Trier par

Etoile

Planete

Galaxie

Satellite

Système solaire

Trou Noir

Affichage d'un astre et de ces informations

SPACE-LEARNER Nom de l'astre

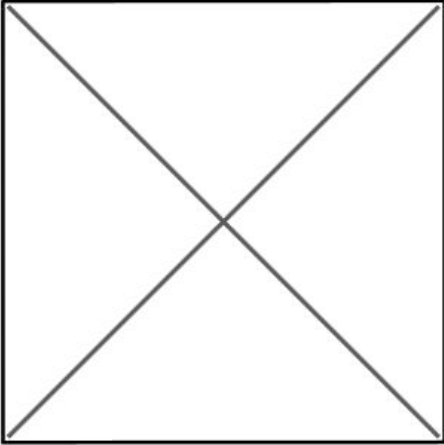
< |||

SPACE-LEARNER

Supprimer

Modifier

NOM DE L'ASTRE



Taille :

Masse :

Distance terre-astre :

Température :

Surface :

1 jours sur l'astre :

Luminosité :

Formulaire de création d'astre

SPACE-LEARNER Ajouter un astre

< ||| **SPACE-LEARNER**

Type ▾

Non de l'astre :

Taille :

Masse :

Distance terre-astre :

Température :


Surface :

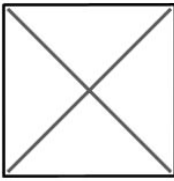
1 jours sur l'astre :


Luminosité :

Mission vers l'astre :

Description :

Uploader une image 





Confirmer

pop up de confirmation de création

SPACE-LEARNER

Voulez vous confirmer la création de cet astre ?

Annuler Modifier

pop up de confirmation

SPACE-LEARNER

Voulez vous vraiment supprimer cet astre ?

Annuler Confirmer

formulaire de modification d'un astre

SPACE-LEARNER Modifier un astre

< ||| **SPACE-LEARNER**

Type ▾

Non de l'astre :

Taille :

Masse :

Distance terre-astre :

Température :


Surface :

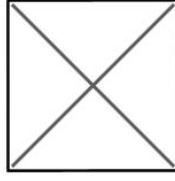
1 jours sur l'astre :

Luminosité :

Mission vers l'astre :

Description :

Changer l'image 



Confirmer

pop up d'erreur

SPACE-LEARNER

Une erreur c'est glissée dans le formulaire

Annuler Modifier

Création d'un Satellite

SPACE-LEARNER Ajouter un astre

< |||

SPACE-LEARNER

Satellite

Non de l'astre :


Taille :

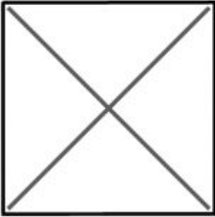
Masse :


Distance terre-astre :

Astre Orbité :

Description

Uploader une image 





Confirmer

Création d'une Galaxie

SPACE-LEARNER Ajouter un astre

< |||

SPACE-LEARNER

Galaxie

Non de l'astre :


Taille :

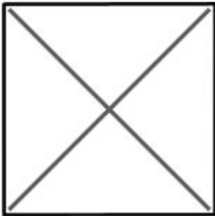
Masse :


Distance terre-astre :

Nombre d'étoiles :

Description

Uploader une image 





Confirmer

Création d'un Trou Noir

SPACE-LEARNER Ajouter un astre

< III

SPACE-LEARNER

Trou Noir

Non de l'astre :


Taille :

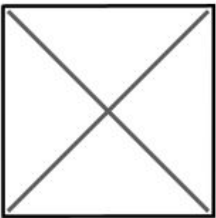
Masse :


Distance terre-astre :

Type :

Description

Uploader une image 





Confirmer

Création d'une Etoile

SPACE-LEARNER Ajouter un astre

< III

SPACE-LEARNER

Etoile

Non de l'astre :

Taille :


Masse :

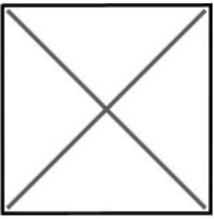
Distance terre-astre :


Luminosité :

Masse :

Description

Uploader une image 





Confirmer

Création d'un Système Solaire

SPACE-LEARNER Ajouter un astre

< ||| **SPACE-LEARNER**

Système solaire ▾

Non de l'astre :

Taille :


Masse :

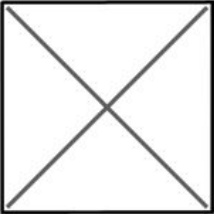
Distance terre-astre :


Nombre de planète :

Nombre de soleil :

Description

Uploader une image 





Confirmer

Création d'une Planète

SPACE-LEARNER Ajouter un astre

< ||| **SPACE-LEARNER**

Planète ▾

Non de l'astre :

Taille :

Masse :


Distance terre-astre :

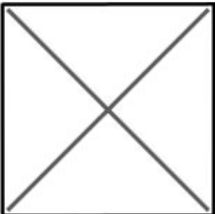
Est vivable ? :


Age :

Masse :

Description

Uploader une image 





Confirmer

Question 5 du quizz

SPACE-LEARNER QUIZZ

< |||

SPACE-LEARNER

QUIZZ de culture de l'espace : niveau moyen

Question du quizz en rapport a l'espace

REPONSE 1

REPONSE 2

REPONSE 3

REPONSE 4

Réponses juste : 3/10

Question 5/20

Résultat du quizz bon

SPACE-LEARNER résultat quizz

< |||

SPACE-LEARNER

QUIZZ de culture de l'espace : niveau moyen

Merci d'avoir participé à notre Quizz sur l'espace !

Retour à l'accueil

Reessayer

Bravo, vous avez obtenu : 8/10

Question 5/20

Résultat mauvais du quizz

SPACE-LEARNER résultat quizz

< |||

SPACE-LEARNER

QUIZZ de culture de l'espace : niveau moyen

Merci d'avoir participé à notre Quizz sur l'espace !

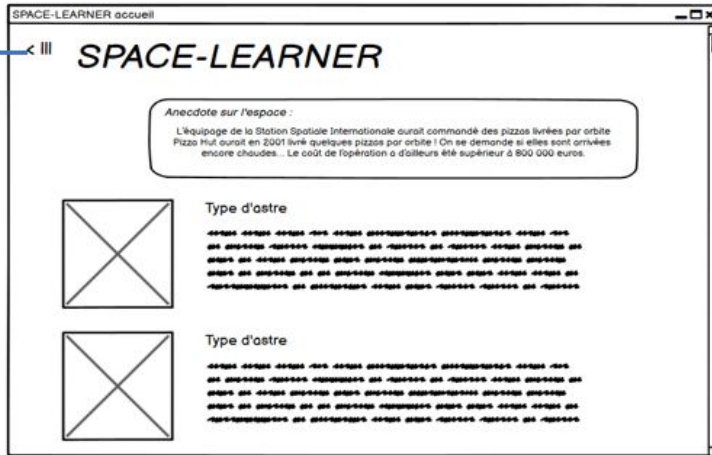
Retour à l'accueil

Reessayer

Dommage, vous avez obtenu : 4/10

Question 5/20

IV - Story board

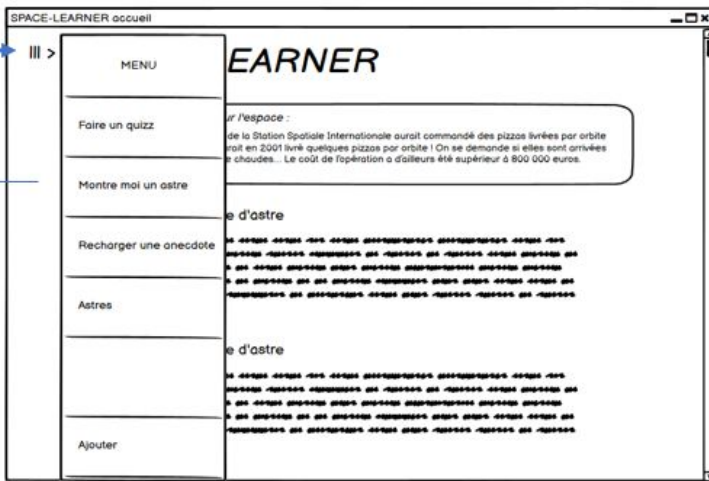


Sur la page d'accueil de l'application on retrouve une liste de catégorie d'astre ainsi que la définition correspondante décrivant les éléments basiques nécessaires à la compréhension du type de l'astre. Cette définition est associée à une illustration de l'astre sous forme d'image.

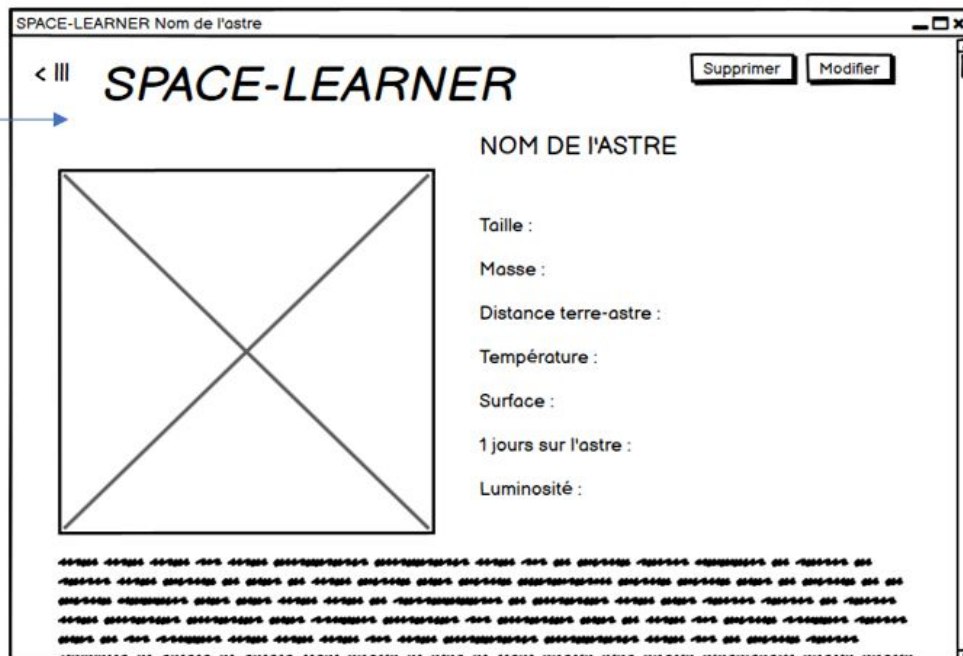
En haut de la page, on retrouve une anecdote tirée au sort dans une liste d'anecdotes prédéfinies. Cette anecdote change à chaque rechargement de la page. La fenêtre possède donc un scroll-bar, en haut on retrouve le nom de l'application ainsi que le mot clé accueil.

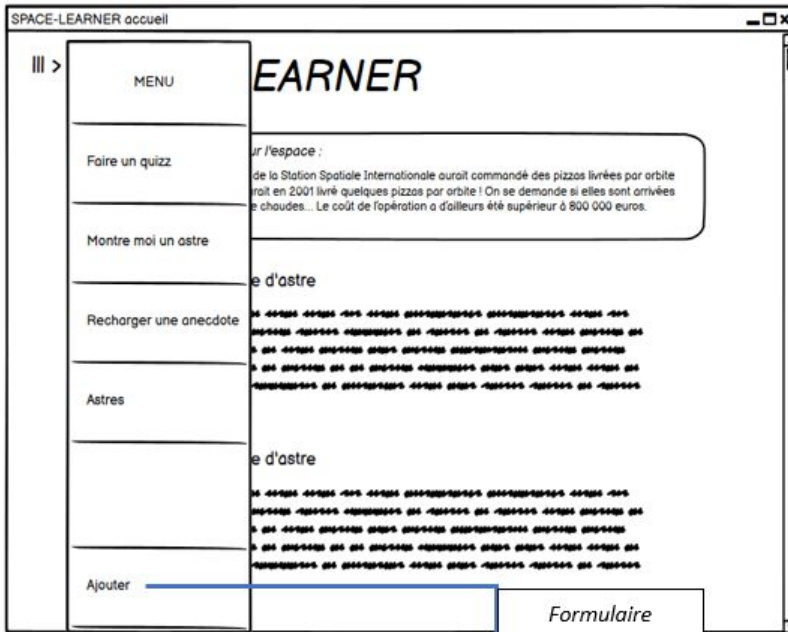
En haut à gauche il y a un bouton cliquable permettant d'accéder au menu de l'application.

Ouverture du menu déroulant



Le bouton "Montre-moi un astre" permet de renvoyer l'utilisateur sur la page détaillée d'un astre tiré aléatoirement dans la liste complète des astres, cette fonctionnalité sert à apprendre à l'utilisateur l'existence d'un astre qu'il ne connaît probablement pas.





Tout en bas, il y a bouton nommé "ajouter" qui permettra à un utilisateur d'ajouter un astre. Cette fonctionnalité est destinée à un utilisateur averti, il faudra qu'il fasse attention aux informations qu'il rentre car ces modifications seront enregistrées dans un fichier.

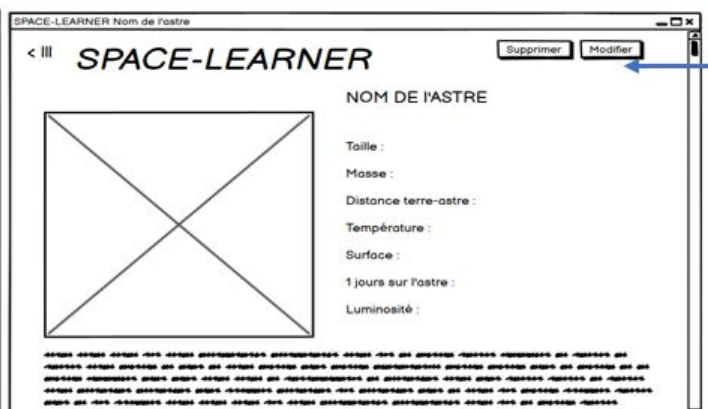
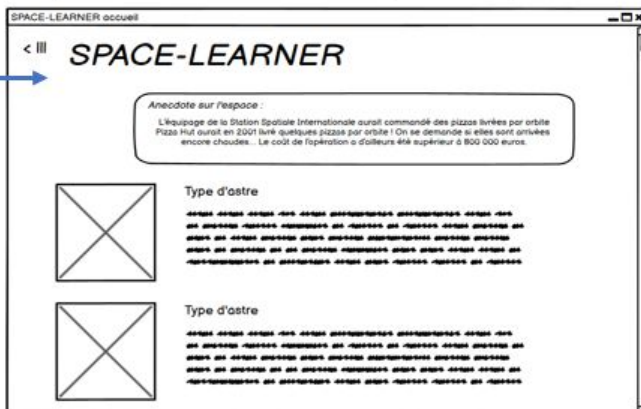
Formulaire dépend du type de l'astre choisi

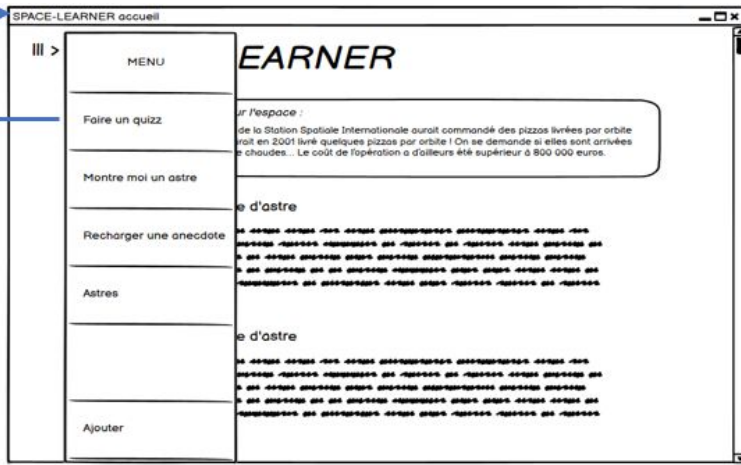
Une page s'ouvre avec un formulaire à remplir, si l'utilisateur remplit tous les champs, la page nouvellement créée s'ouvre. S'il manque des informations, il y a un message d'erreur qui s'affiche, l'utilisateur a pour option d'annuler ou de modifier les données précédemment ajoutées.

Retour à l'accueil

Formulaire erronée : erreur

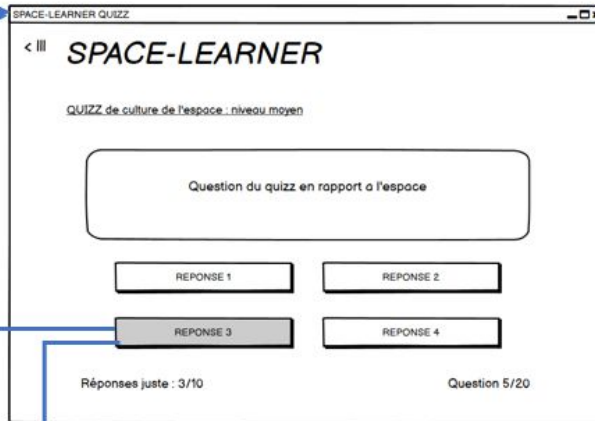
Formulaire sans erreurs : création





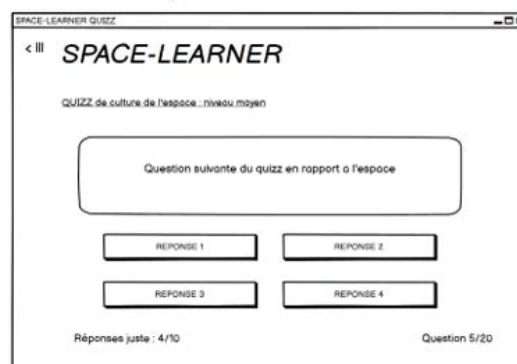
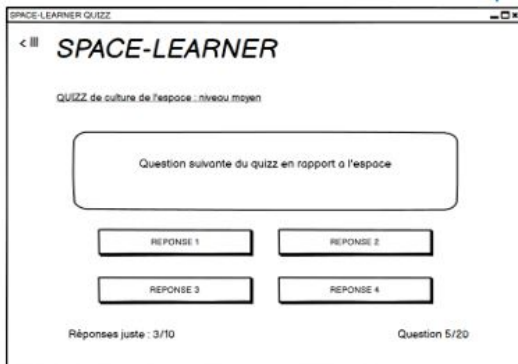
L'utilisateur ayant cliqué sur le bouton "faire un quizz", il pourra donc participer à un quizz, lorsqu'il se trompe, la mauvaise réponse sera en rouge et la bonne en vert, en revanche s'il ne se trompe pas, uniquement la bonne réponse sera en vert. Une moyenne est calculée à partir des résultats.

A la fin des vingt questions, il obtiendra son résultat ainsi qu'une appréciation, s'il a plus de 5 points, elle sera : "BRAVO, vous avez obtenu 5/10 !", on proposera donc à l'utilisateur de faire un autre quizz. Sinon elle sera : "DOMMAGE, vous avez obtenu 4/10 !" et il sera proposé à l'utilisateur de réessayer.



Mauvaise réponse : point pas ajouté

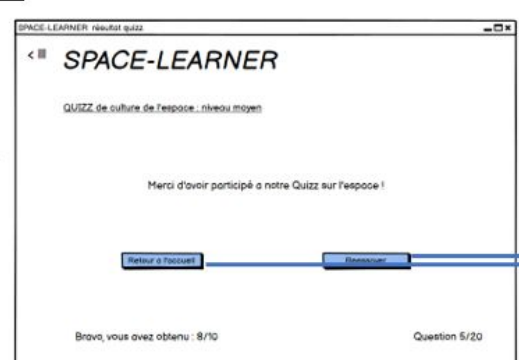
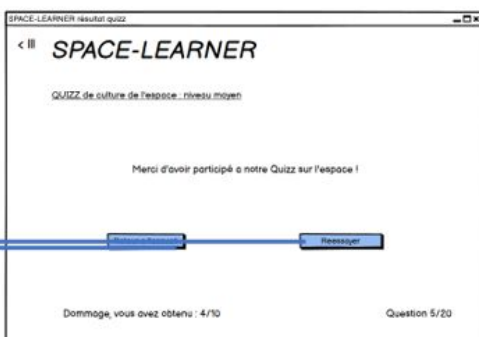
Bonne réponse : point ajouté

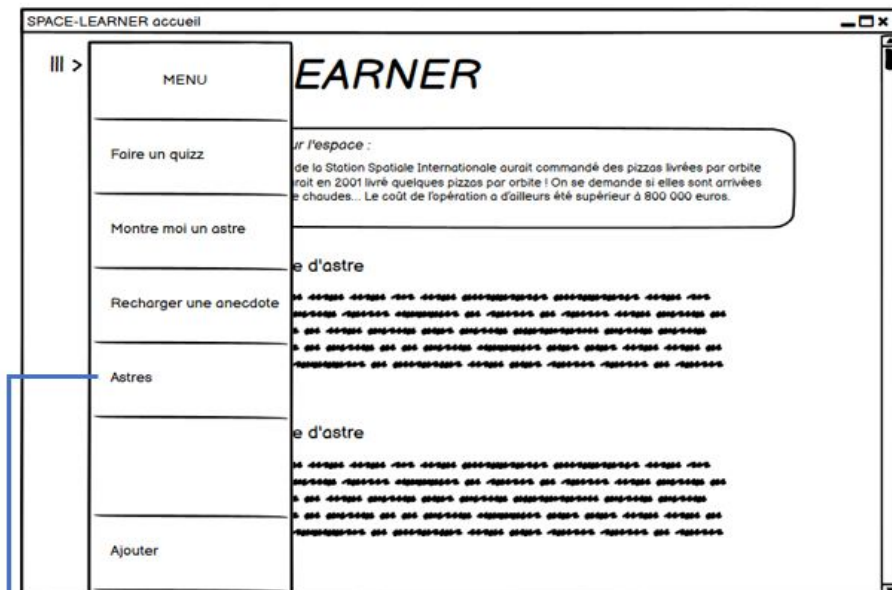


Moyenne inférieure à 5 :

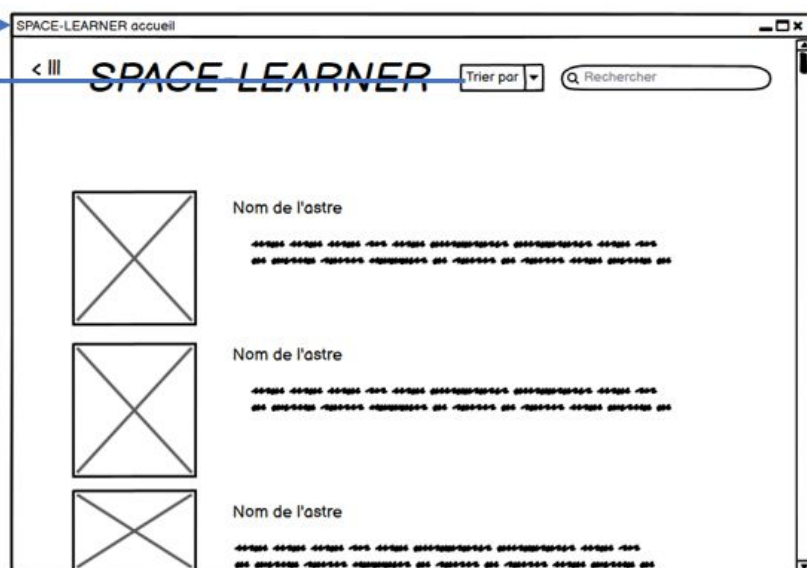
Au bout de 10 questions

Moyenne supérieure à 5 :

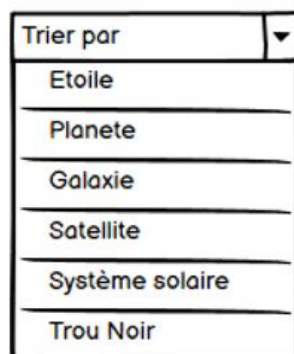




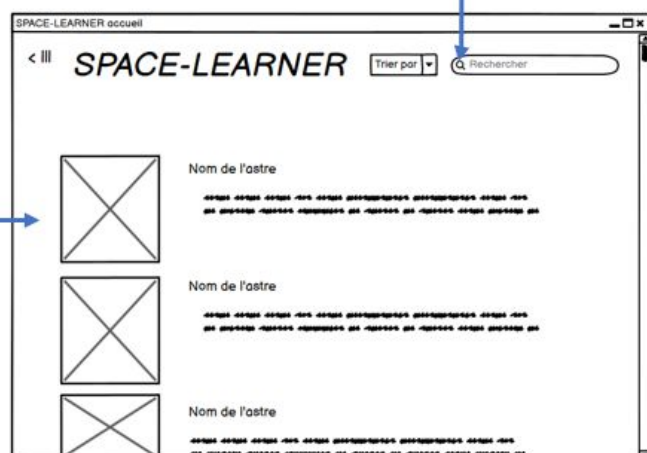
Si on appuie sur le bouton "Astres" du menu, cela ouvre une nouvelle page contenant la liste de tous les astres ainsi qu'un aperçu de la description de l'astre en question ainsi qu'un aperçu de l'astre en question. Lorsqu'on clique sur un astre, on accède à la page détaillée d'un astre. En haut de la page Astres, il y a un Combobox contenant une liste de critères pour rechercher des astres le type de l'astre. En haut à droite, on retrouve une barre de recherche qui affiche une liste d'astres possédant les lettres inscrites dans leurs noms.

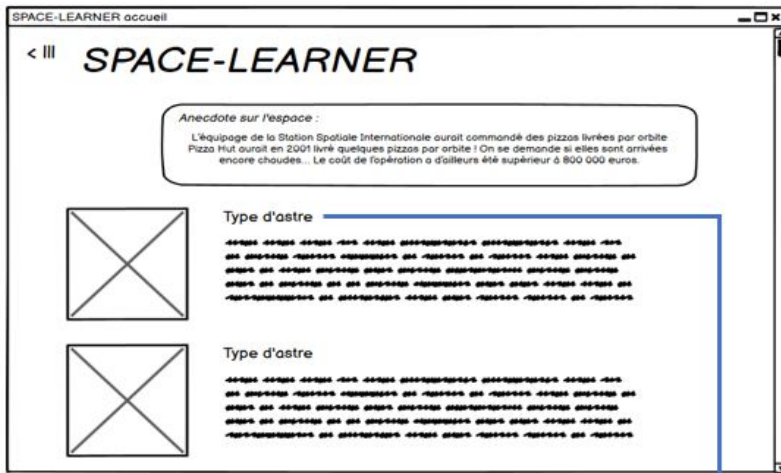


Recherche par
mot clé

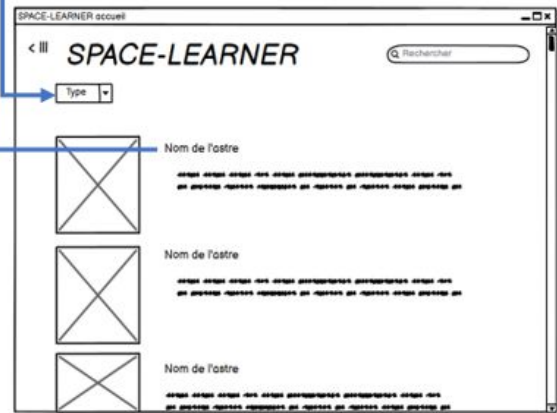
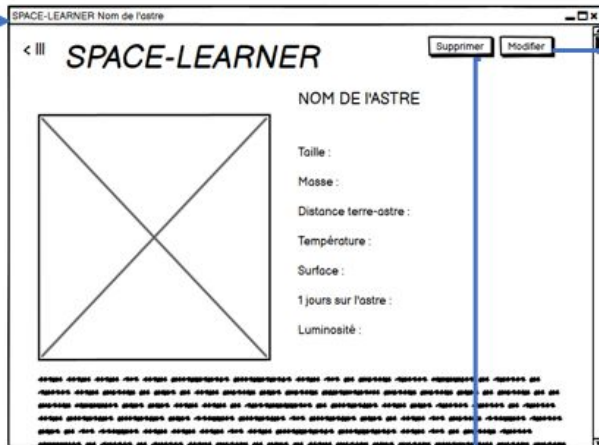


Changement
d'affichage des
astres



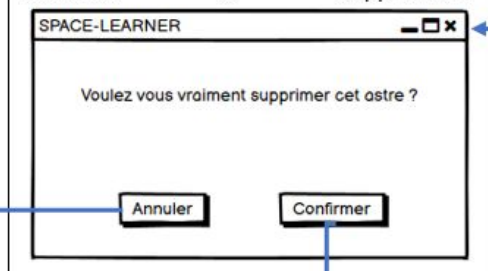


Chaque type d'astre décrit sur la page d'accueil est cliquable, elle renvoie sur une recherche avancée avec comme critère le type d'astre choisi juste avant. Elle contient une image en grands, à droite on aura quelque détail précis sur l'astre tel que sa taille sa masse... certain seront facultatif car on ne peut pas connaître certaines informations notamment par exemple la température d'un trou noir. On aura ensuite une description sur l'astre, son histoire...

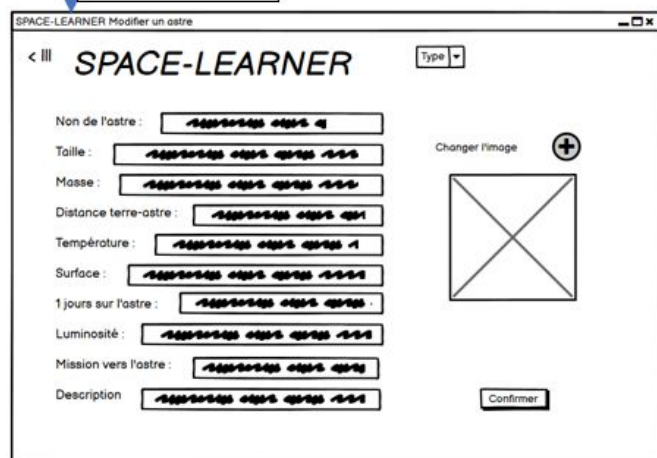
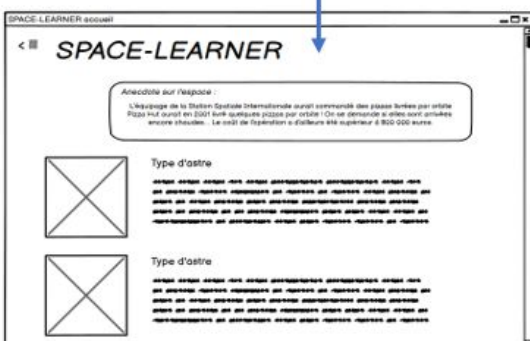


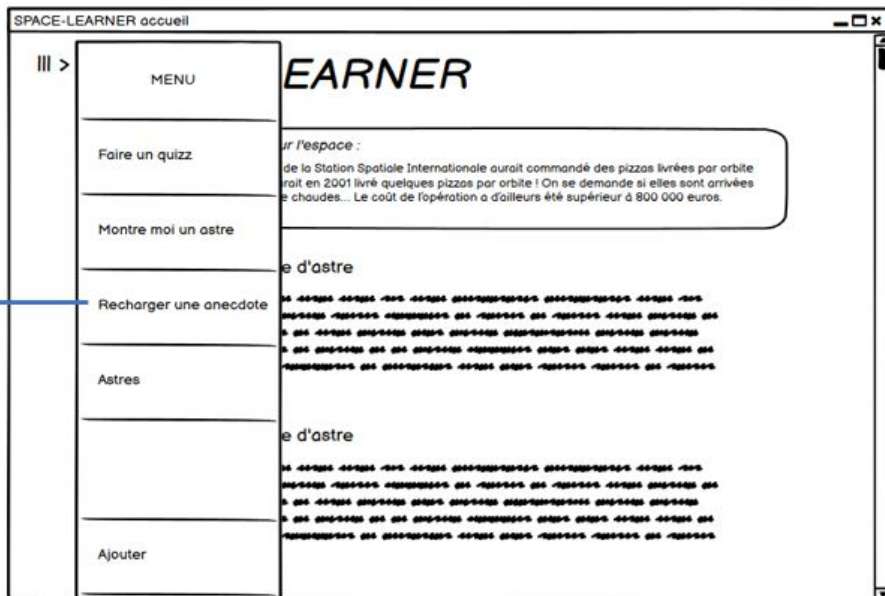
Supprimer est très simple, il suffit de cliquer sur le bouton supprimer, il y a un pop-up lui demandant de confirmer, s'il annule, il retombe sur la page de l'astre, sinon, l'utilisateur sera reconduit à la page d'accueil de l'application.

Si l'utilisateur clique sur modifier, il obtient le même formulaire que quand il a ajouté l'astre mais cette fois ci, le formulaire est prérempli, lorsqu'il confirme, si les informations sont bonnes, l'utilisateur atterri sur la page de l'astre. S'il y a une erreur, l'application affichera une fenêtre pop-up lui proposant d'annuler ou de modifier.



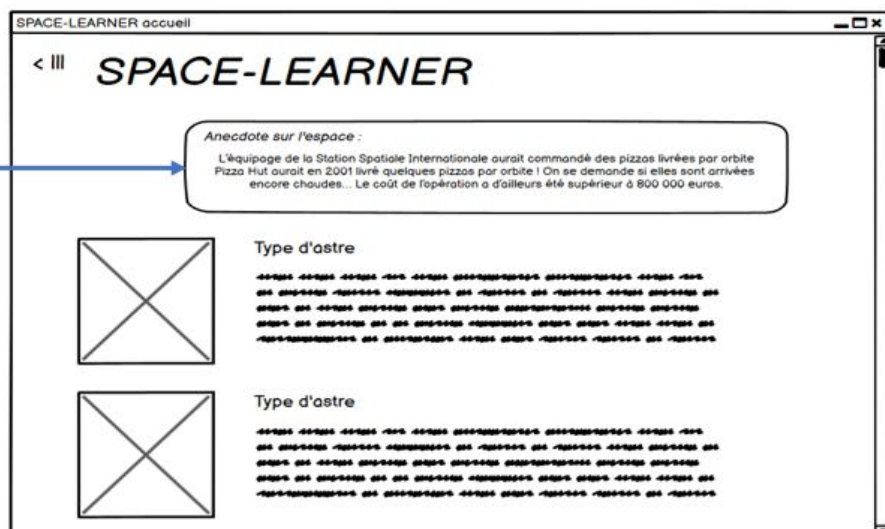
Formulaire dépend du type de l'astre



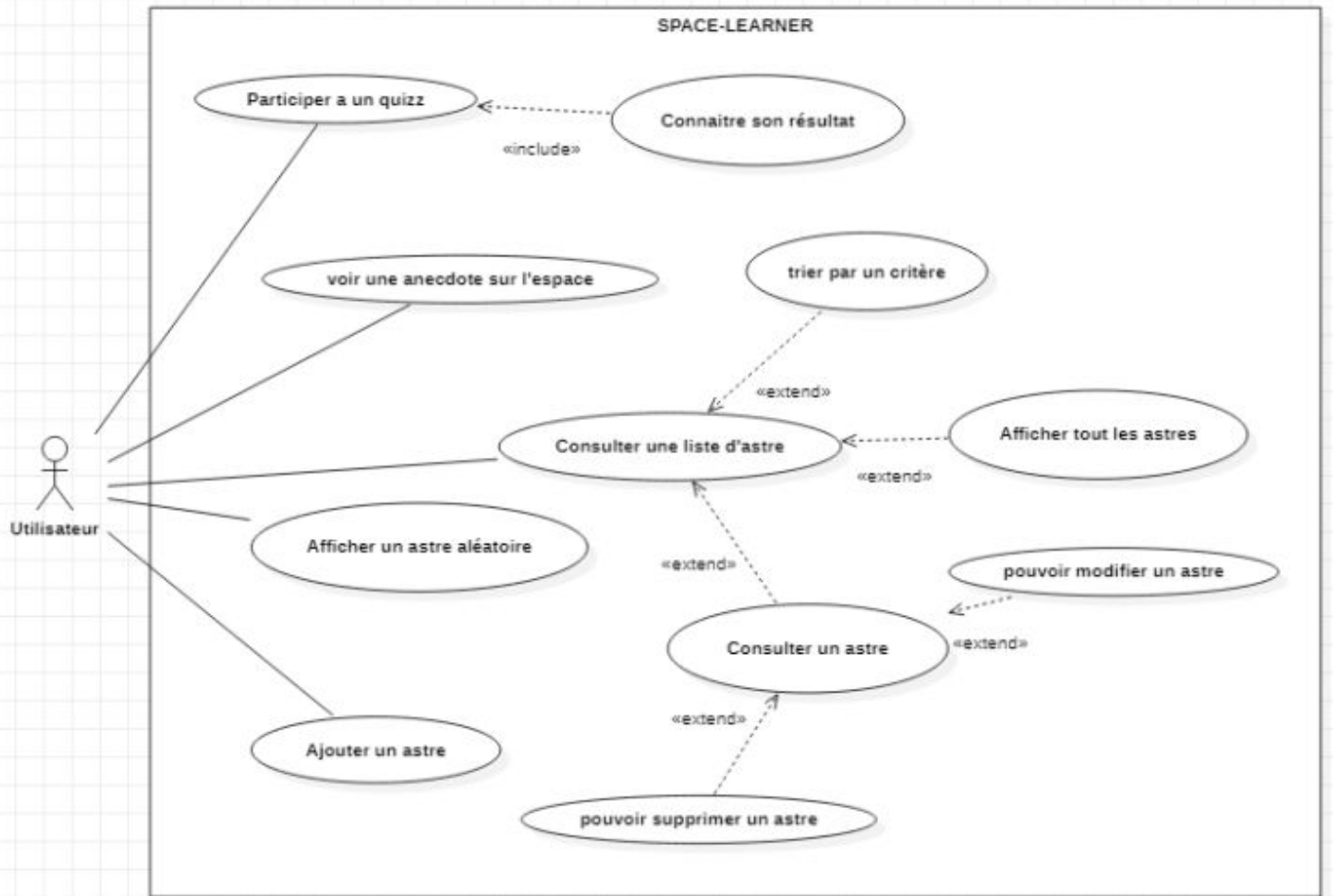


Actualisation de
l'anecdote

Si l'utilisateur clique sur le bouton « Recharger une anecdote », l'anecdote en haut de la page changera, une nouvelle sera tirée au sort parmi la base de données contenant les anecdotes.



V - Diagramme de cas d'utilisations :



nom: réaliser un quizz

objectif: pouvoir répondre à des questions et valider sa réponse.

acteurs principaux: l'utilisateur

acteurs secondaire: -

condition initiales: -l'utilisateur peut à tout moment réaliser un quizz

scénario d'utilisation: -l'utilisateur doit répondre à 20 questions et à chaque fois valider sa réponse.

conditions de fin: l'utilisateur peut connaître son score.

nom: pouvoir connaître son score

objectif: l'utilisateur pourra savoir son score afin de s'améliorer et aura un commentaire sur le score qu'il a eu.

acteurs principaux: l'utilisateur

acteurs secondaire:-

condition initiales: l'utilisateur viens de réaliser un quizz et vient de valider la 20eme question.

scénario d'utilisation:l'utilisateur pourra voir son score et le commentaire et ensuite il pourra se déplacer comme il le souhaite dans l'application

conditions de fin:l'utilisateur a pris conscience de son score et du commentaire.

nom: voir une anecdote sur l'espace.

objectif:l'utilisateur pourra lire une anecdote dans la page d'accueil de l'application

acteurs principaux: l'utilisateur

acteurs secondaire: -

condition initiales: l'utilisateur peut se trouver n'importe où dans l'application

scénario d'utilisation:l'utilisateur doit aller dans la page d'accueil pour lire une anecdote, pour qu'il y est une nouvelle anecdote l'utilisateur doit recharger la page.

conditions de fin:l'utilisateur a pu lire une anecdote

nom: ajouter un astre

objectif: l'utilisateur peut ajouter à sa base de donnée un nouvel astre.

acteurs principaux:l'utilisateur

acteurs secondaire: -

condition initiales: l'utilisateur peut se trouver n'importe où dans l'application.

scénario d'utilisation : l'utilisateur doit aller dans "nouvel astre" dans l'application puis remplir un formulaire et le valider.

conditions de fin: si l'utilisateur n'a pas rempli quelques cases obligatoires tel que le nom, l'astre ne sera pas enregistré. L'utilisateur doit remplir les cases obligatoires puis retenter de valider. sinon l'astre sera enregistrer

nom: consulter un astre

objectif: l'utilisateur peut voir la description de l'astre choisis

acteurs principaux: l'utilisateur

acteurs secondaire: -

condition initiales: l'utilisateur doit être dans la liste des astres(triés ou non) ou a dû consulter un astre random.

scénario d'utilisation: l'utilisateur doit sélectionner un astre ou a dû sélectionner un astre random. il peut ensuite lire la description de l'astre, voir sa photo...

conditions de fin: l'utilisateur a pu avoir accès à la description d'un astre.

nom: trier par critère

objectif: l'utilisateur pourra accéder à seulement un type d'astre.

acteurs principaux: l'utilisateur

acteurs secondaire: -

condition initiales: l'utilisateur peut se trouver n'importe où dans l'application.

scénario d'utilisation: l'utilisateur doit pouvoir sélectionner le type d'astre qu'il veut et accéder à la liste de ces astres.

conditions de fin: l'utilisateur peut sélectionner un astre plus facilement si il recherche un type précis d'astres.

nom: afficher un astre aléatoire

objectif: l'utilisateur pourra consulter n'importe quel astre de n'importe quel type.

acteurs principaux: l'utilisateur

acteurs secondaire: -

condition initiales: l'utilisateur peut se trouver n'importe où dans l'application.

scénario d'utilisation: l'utilisateur pourra sélectionner grâce au menu un astre aléatoire, il pourra ensuite le consulter

conditions de fin: l'utilisateur a pu consulter un astre pris aléatoirement.

nom: supprimer un astre

objectif: supprimer un astre de la base de données

acteurs principaux: l'utilisateur

acteurs secondaire: -

condition initiales: l'utilisateur doit se trouver sur la page d'un astre.

scénario d'utilisation: l'utilisateur doit sélectionner un bouton supprimer, un message s'affichera pour confirmer la suppression. il devra ensuite cliquer sur annuler ou ok.

conditions de fin: si l'utilisateur a cliqué sur ok alors l'astre à été supprimé de la base de données sinon il est revenu en arrière sur la page de l'astre.

nom: modifier un astre

objectif: l'utilisateur peut modifier n'importe quel astre de l'application

acteurs principaux: l'utilisateur

acteurs secondaire: -

condition initiales: l'utilisateur doit se trouver sur la page d'un astre.

scénario d'utilisation: l'utilisateur doit sélectionner le bouton modifier. Le formulaire de l'astre s'affichera et il pourra modifier l'astre. il devra ensuite valider la modification.

conditions de fin: l'utilisateur à modifier ou non son astre et à valider la modification, l'utilisateur n'a pas supprimé des cases obligatoires, les modifications de l'astre ont donc été enregistrées.

Si l'utilisateur a supprimé des cases obligatoires, les modifications ne seront pas enregistrées et il devra les remplir ou revenir en arrière pour annuler les modifications qu'il a pu faire.

nom: consulter une liste d'astre

objectif: l'utilisateur peut voir tous les astres de l'application

acteurs principaux: l'utilisateur

acteurs secondaire: -

condition initiales: l'utilisateur peut se trouver n'importe où dans l'application.

scénario d'utilisation: l'utilisateur peut consulter la liste des astres d'un seul type ou peut avoir la liste de tous les astres

conditions de fin: l'utilisateur peut cliquer facilement sur l'astre qu'il recherchait.

nom: afficher tous les astres

objectif: l'utilisateur peut consulter la liste de tous les astres

acteurs principaux:l'utilisateur

acteurs secondaire: -

condition initiales: l'utilisateur peut se trouver n'importe où dans l'application.

scénario d'utilisation:l'utilisateur doit pouvoir sélectionner un bouton pour avoir la liste de tous les astres

conditions de fin: l'utilisateur peut regarder la liste de tous les astres et consultez celui qu'il veut.

VI - Choix de l'ergonomie justifiée :

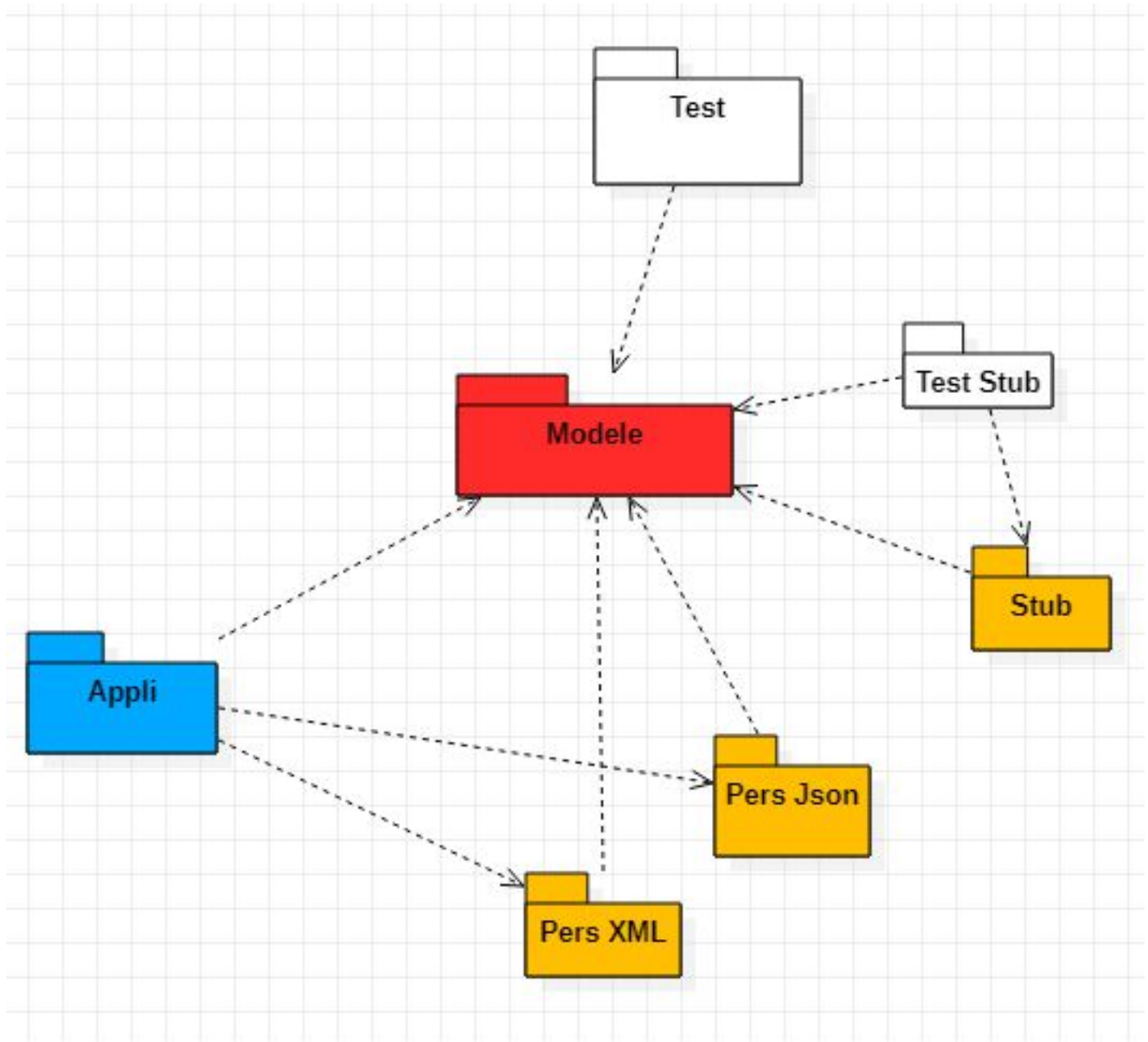
Nous avons voulu faire une application assez simple pour que toutes personnes ayant un âge différent puissent y accéder facilement. Le point centrale de l'application est le menu, celui-ci est accessible depuis n'importe quel endroit grâce à un symbole logique. Il permet facilement à l'utilisateur d'accéder à n'importe quelle fonction de l'application en moins de trois clics à part les fonctions modifier et supprimer qui se trouvent sur la page de chaque astre car nous avons jugées que ces fonctionnalités étaient avancées et qu'il n'était pas important de les mettre en avant pour éviter entre autre les mauvaises manipulations. Nous avons aussi utilisé des mots souvent utilisés ou très explicites tels que "supprimer" "modifier"...

VII - Prise en compte de l'accessibilité :

Ensuite pour le côté esthétique, nous avons voulu rester sur le thème de l'espace, avec des couleurs un peu sombres et des teintes bleutées mais en gardant une bonne lisibilité de manière à économiser au maximum la vision de nos lecteurs afin qu'il passent le plus de temps possible sur l'application. Pour donner une idée de ce à quoi ressemble l'astre, nous avons mis une image de celui-ci (si possible), cela pourra faire aussi rêver l'utilisateur de contempler la beauté de certains astres et leur donnera envie d'avoir plus d'informations sur celui-ci.

2 - Doc Conception Orienté Objet

I - Diagramme de paquetage :



Description :

On voit sur le diagramme de paquetage différents packages contenant tous des éléments interchangeables.

- Le Package Modele contenant toutes les classes :

Il doit gérer toutes les entités de l'application, en partant de la Classe Astre et de ces Classes filles jusqu'à l'incrémentation des Anecdotes grâce à la classe Anecdote. Le fonctionnement de l'application dépend donc en grande partie du Modèle.

- Le Package Appli contenant les vues de l'application :

Il rassemble toutes les parties visuelles de l'application notamment toutes les userControls. Elle gère également l'algorithme d'affichage et de gestion d'événements des Questions d'un quizz. L'intérêt de dissocier l'Appli du reste des packages est de rendre interchangeable les tests et les autres morceaux de l'application. Ainsi ce découpage permet l'extension de l'appli. En effet, ainsi, je peux créer plus tard une nouvelle méthode de Persistance, j'ai simplement à implémenter la méthode IPersistance depuis Modèle.

- Le Package Stub implémente l'interface IPersistance :

Le Stub a pour rôle temporaire de créer et d'instancier les éléments nécessaires au fonctionnement du Manager, il crée des données plus ou moins factices pour tester les fonctionnalités des autres projets. Il sera effectivement remplacé par de la Persistance à long terme.

- Le package PersXML implémente l'interface IPersistance :

Contient les méthodes nécessaires à la persistance de données notamment Sauvegarder, Charger. Le tout ayant pour but de restituer les données depuis un fichier. Ce Package aura pour but de remplacer le Stub, il est donc facultatif à condition d'utiliser le Stub.

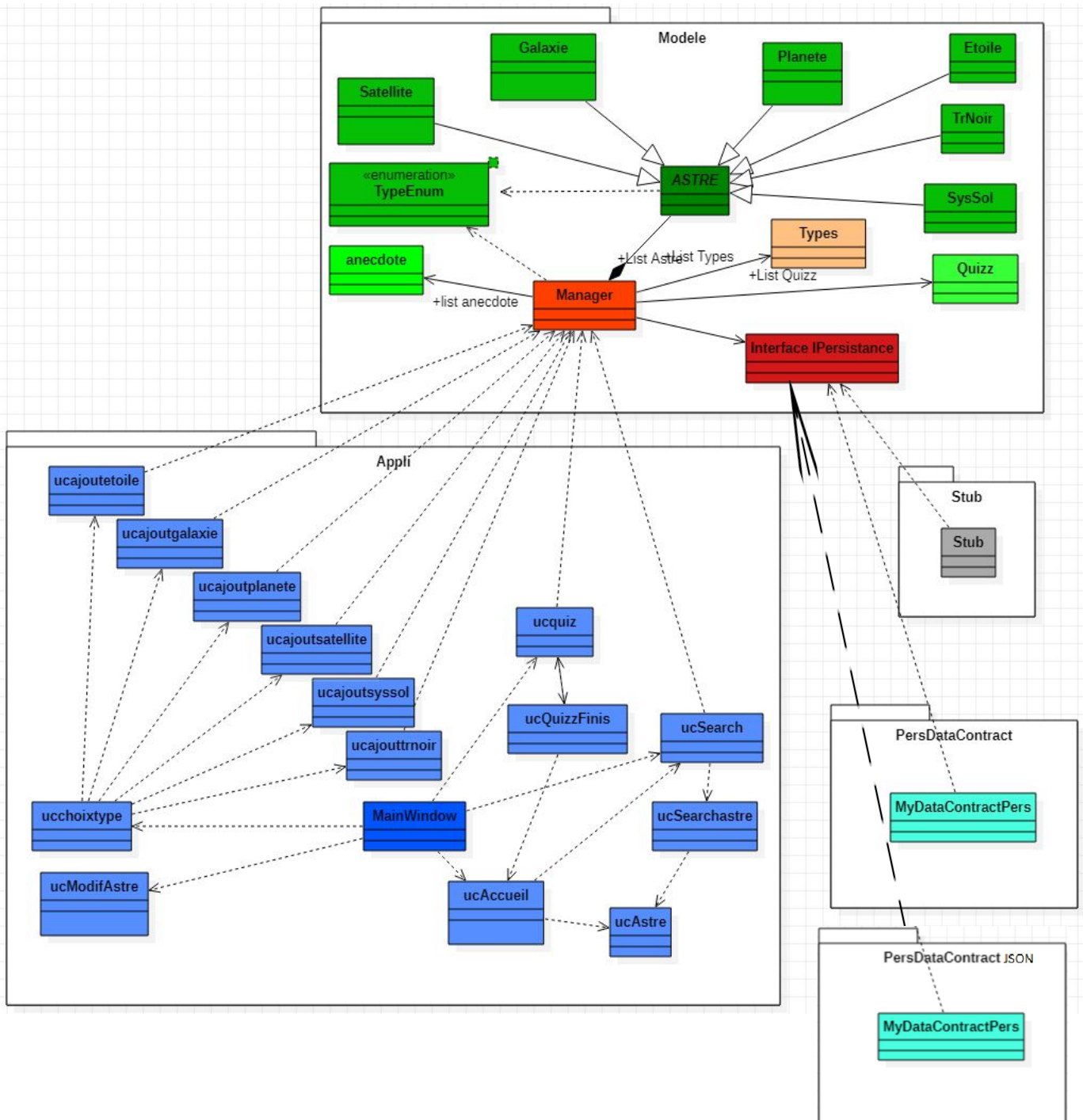
- Le package PersLinq implémente l'interface IPersistance :

Contient les méthodes nécessaires à la persistance de données notamment Sauvegarder, Charger. Le tout ayant pour but de restituer les données depuis un fichier. Ce Package aura pour but de remplacer le Stub, il est donc facultatif à condition d'utiliser le Stub.

- Le package Test n'a pas besoin du Stub pour fonctionner :

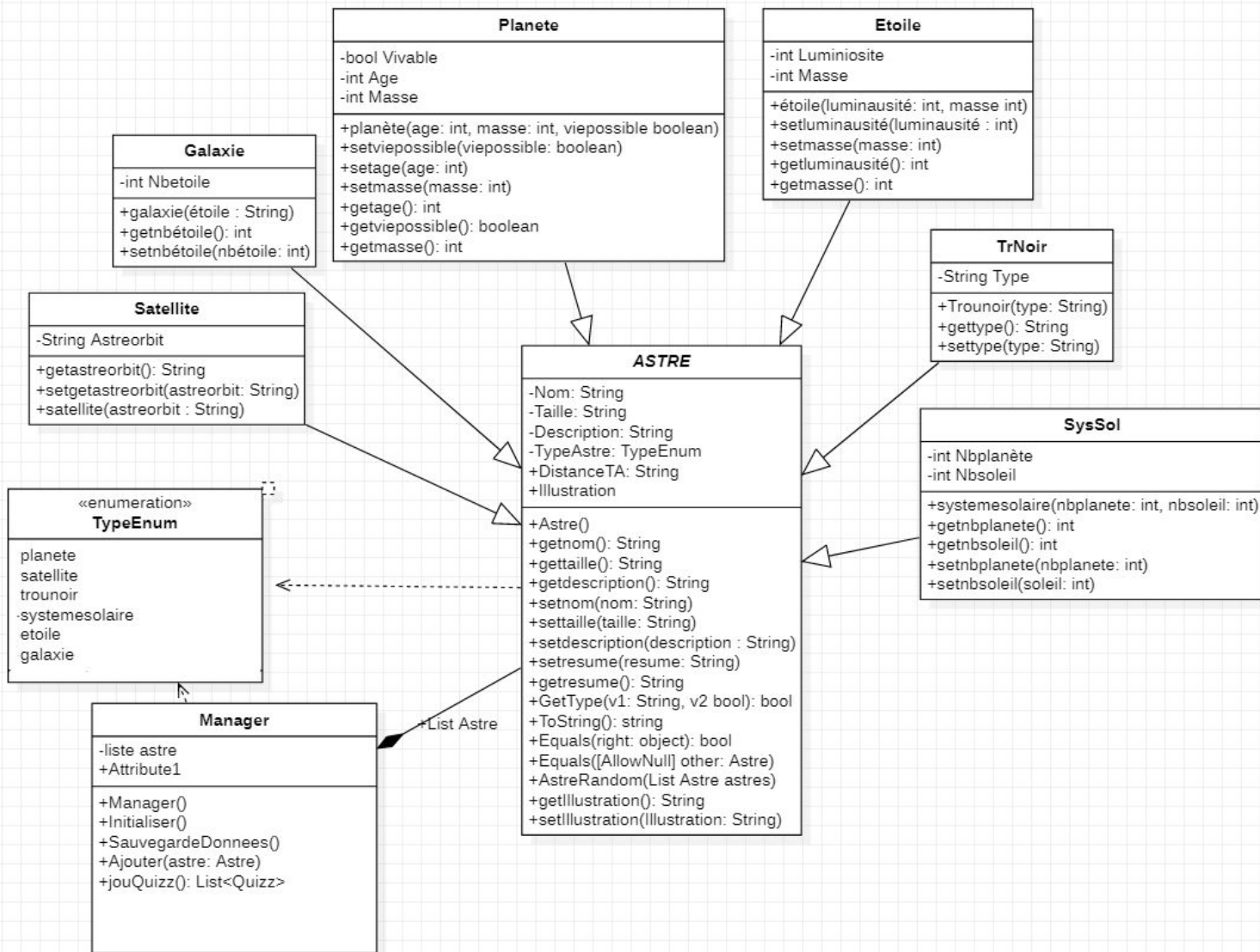
Comme son nom l'indique, il a pour but de tester les fonctionnalités des Méthodes indépendamment mais plus précisément la partie Appli. Il doit tester si la sécurité des entrées-sortie de données est complète ainsi que la fonctionnalité parfaite des éléments.

Diagramme de classes simplifié :



III - Description écrite de l'architecture :

Astre :



Classes Étendues Astres

La classe principale de l'application est la classe abstraite **Astre**. Implémentant l'interface **IEquatable < Astre >**, elle possède plusieurs Attributs notamment les caractéristiques basiques des Astres, peu important leurs types incluant **Nom**, **Taille**, **Description**, **Illustration**, La distance Terre-Astre et pour Finir son type stocké dans un **Type Enum**. Grâce à cet ensemble de classes, il est aisé de créer un **Astre** de n'importe quelle type.

Nous avons choisi de créer une classe par type d'Astre pour plusieurs raisons. Tout d'abord, cette méthode permet de stocker simplement les différentes informations des astres notamment le nombre d'étoile pour une Galaxie, cela permet également de ne pas proposer à l'utilisateur d'entrer par exemple la luminosité d'une planète. Grâce à cette disposition, il est également plus aisé de trier les Astres par type.

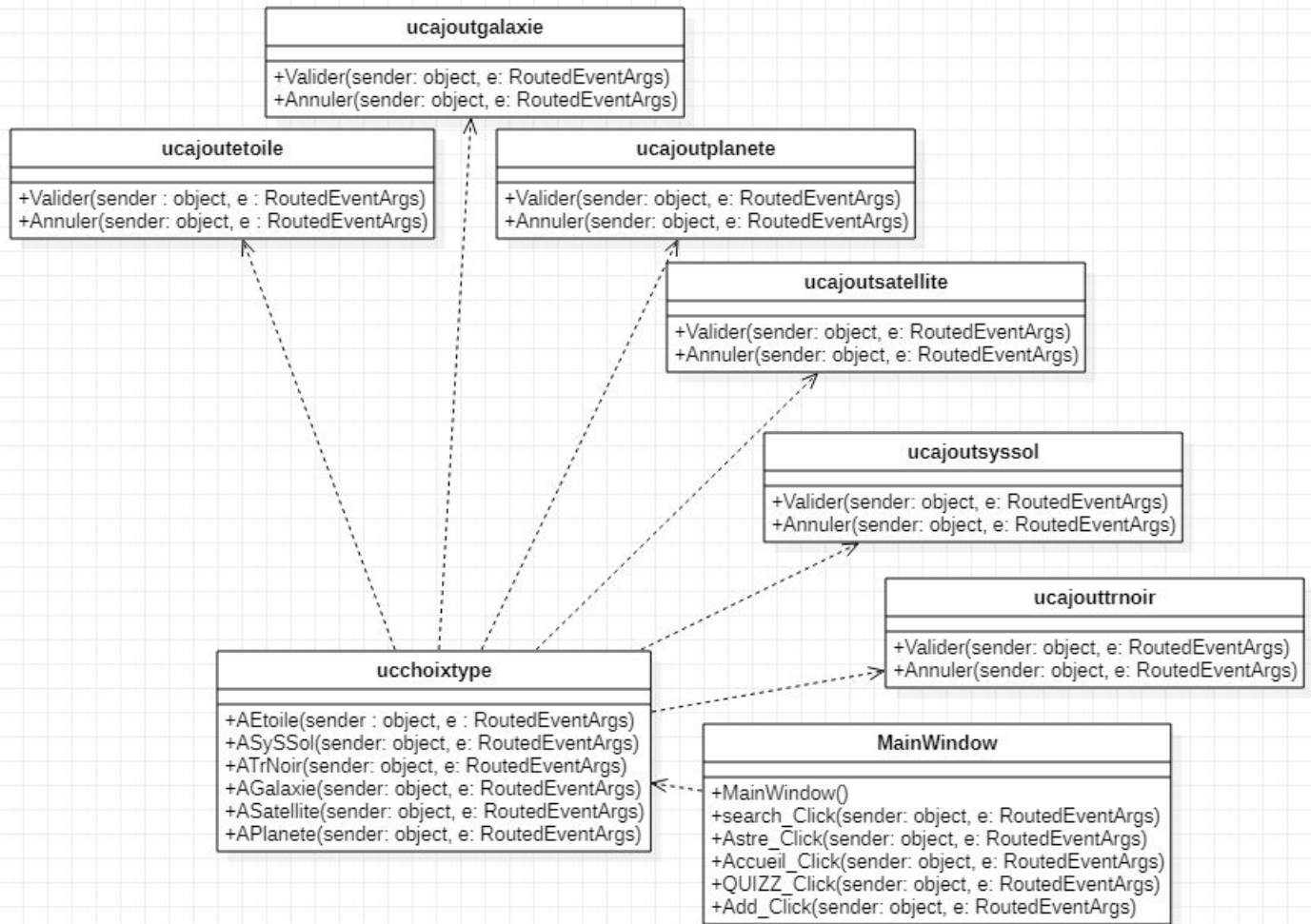
Chaque astre peut être modifié dans notre application. Il nous fallait donc tous les get et set de chaque attribut pour pouvoir voir et modifier les attributs.

Ensuite nous avons deux méthodes pour créer un astre, une qui nous permet d'entrer des astres dans la méthode initialiser de manager et un autre qui nous sert dans l'ajout des astres. En effet lorsque nous voulons ajouter un astre, un astre est créé avec des valeurs tel que "aucun nom" pour le nom, ensuite, l'astre est mis à jour avec les informations rentrées par l'utilisateur.

La méthode "AstrRandom" nous nous en servons pour afficher un astre aléatoire lorsque nous cliquons dans le menu "afficher un astre random".

Les méthodes equals servent dans l'affichage des astres.

Partie ajout d'un astre:



Cette partie est la partie qui nous sert à ajouter des astres de différents types. Les méthodes incluent choix type nous servent à afficher les users contrôles des différents types dans un espace réservé sur ChoixType grâce à plusieurs boutons.

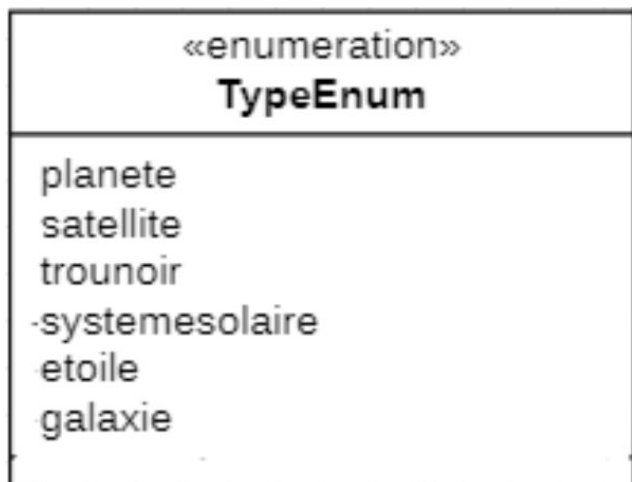
Ensuite dans chaque ajout d'un type nous avons deux méthodes. Une qui contrôle un bouton validé qui sert à ajouter le nouvel astre dans la liste d'astre puis de revenir à l'accueil et le deuxième sert à contrôler un bouton annulé permettant de ne pas sauvegarder l'astre créé et de revenir à l'accueil.

L'ajout d'astre étant une pièce importante dans notre application nous avons décidé de pouvoir y accéder grâce au menu depuis l'accueil. Grâce au menu, nous pouvons accéder à l'affichage de "choix type" qui est un user control permettant de choisir le type de l'astre que l'on veut ajouter.

Nous avons ensuite créé six usercontrols pour pouvoir entrer chaque type d'astre. Nous avons fait ce choix pour que cela soit plus pratique et plus efficace pour ajouter un astre, en effet, l'utilisateur peut voir tous les types d'astres qu'il peut ajouter et a juste à cliquer sur un bouton pour avoir accès au formulaire et il pourra à tout moment changer de type. Pour cela nous affichons le usercontrol dans une grid de choix type. Cette façon de faire permet également une clarté d'affichage et facilite l'entrée

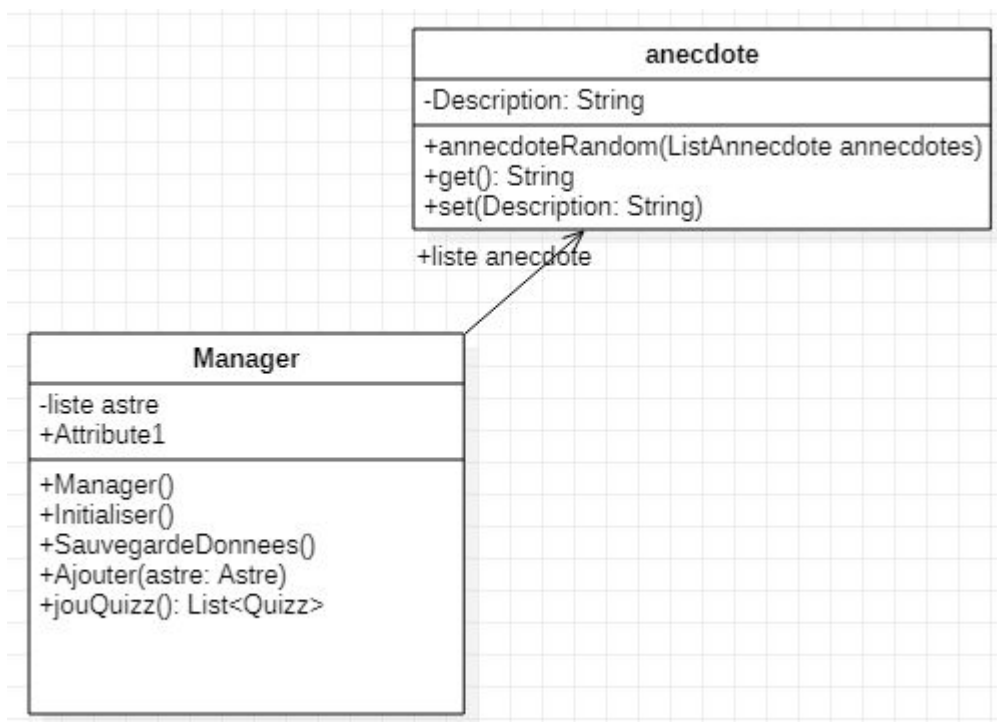
Ensuite lors de l'ajout, nous avons un astre déjà créé par défaut avec des informations tel que "aucun nom" pour son nom. Les attributs de l'astre sont reliés dans les textbox avec le binding dans le xaml et au datatcontext = le nouvelle astre pour que le binding fonctionne. L'utilisateur pour entrer tout ce qu'il souhaite puis il lui suffit de valider. Le bouton Valider fera en sorte de garder l'astre créé en l'ajoutant à la liste d'astre grâce à la méthode ajoute du manager. Si l'utilisateur souhaite ne pas sauvegarder cet astre, il lui suffira de cliquer sur le bouton annuler. Cela le ramène dans l'accueil.

La classe enumeration TypeEnum



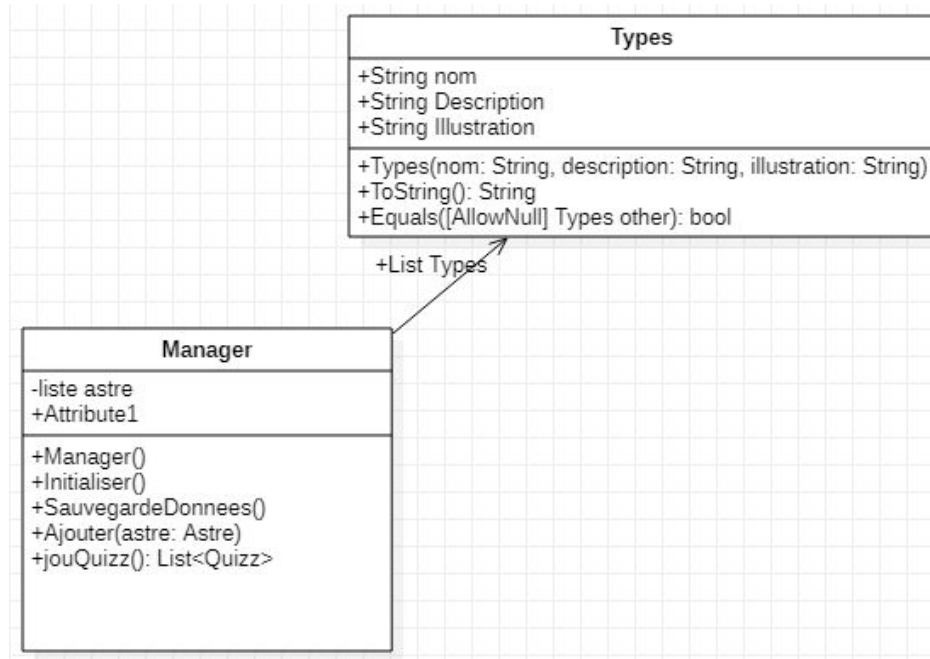
La classe enumeration EnumType est utilisée pour l'affichage de la liste d'astre. Nous affichons les types dans une combobox puis nous testons pour voir quel type est sélectionné et nous affichons les astres qui ont ce type.

La classe anecdote



Manager possède une liste d'anecdotes. Une anecdote est un String appelé Description. La méthode `anecdoteRandom` permet de renvoyer une anecdote tirée aléatoirement dans la liste du manager.

La classe Types

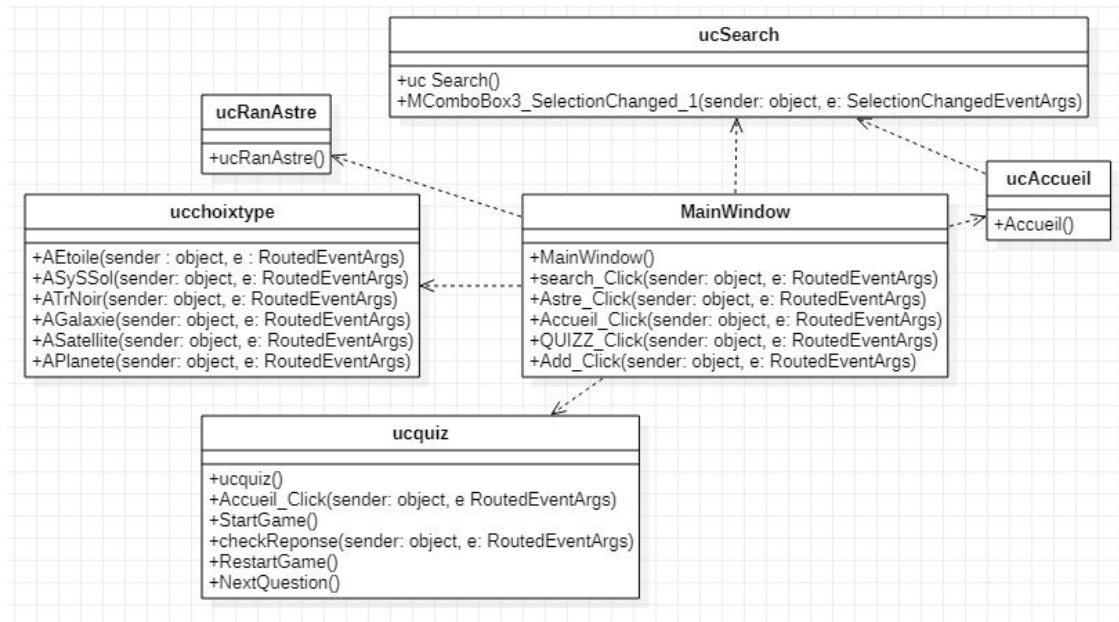


La classe Types nous sert à donner une définition d'un type. Un Types possède un nom (ex : planète), une définition et une illustration pour pouvoir voir à quoi ressemble ce type d'astre.

Le constructeur Types permet de créer un type. Nous les avons créés dans le manager qui possède la liste des types qui restera fixe.

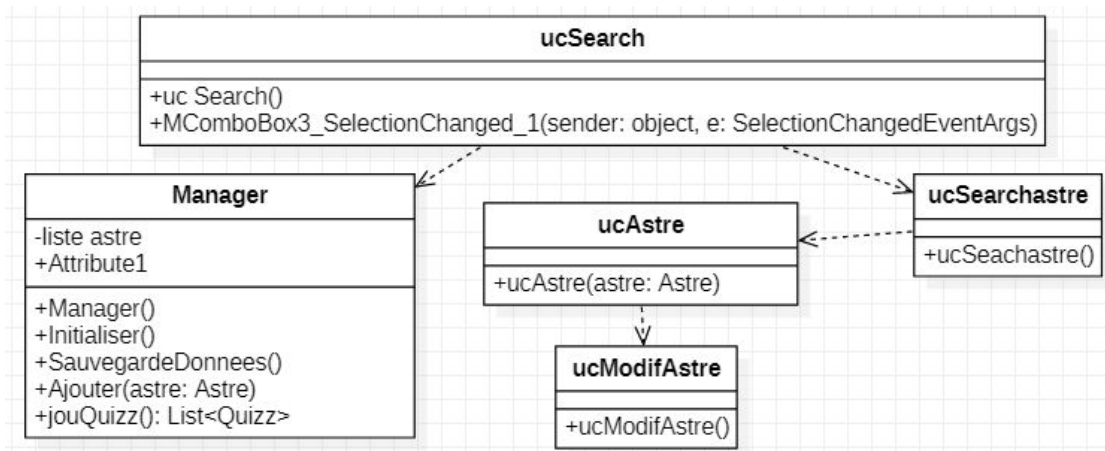
Les autres méthodes de types nous servent dans leurs affichages dans la page accueil.

MainWindow

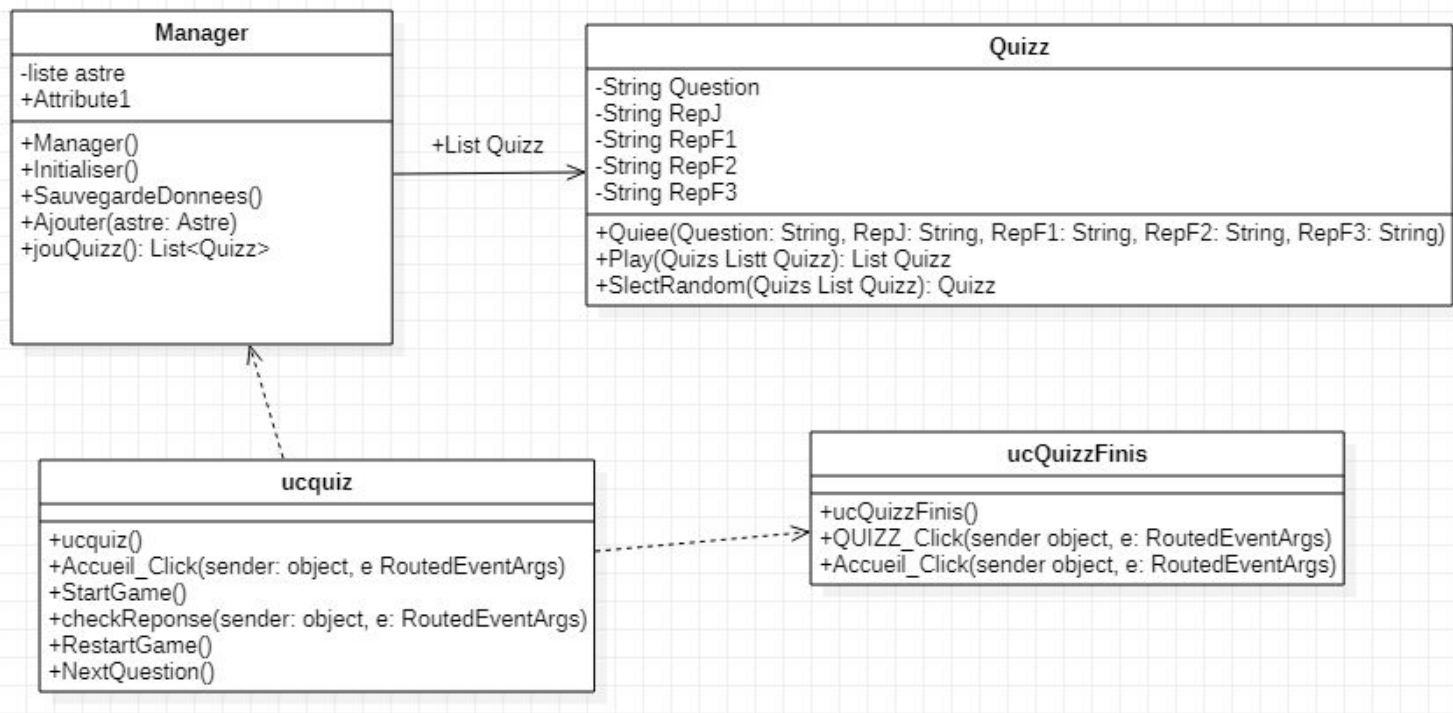


ici MainWindow est la fenêtre dans laquelle, les usercontrols s'affichent.

Elle est constituée d'un menu dans lequel on peut cliquer et ainsi faire apparaître certaines fenêtres grâce aux méthodes de cette classe.



Cette classe a été créée pour pouvoir afficher une liste d'astre qui ont le même type choisi dans le combobox. Si c'est le cas alors elles sont affichées.



La classe quizz est une classe qui possède 5 attributs : trois mauvaises réponses, une bonne réponse et la question. Le Manager possède une liste de ces quizzs.

Réalisation d'un quizz

Ici nous avons les classes qui vont nous permettre de réaliser un quizz. Tout d'abord on prend 10 questions au hasard tiré une à une grâce à la méthode `SelectRandom` dans `Quizz` puis que l'on met dans une liste de quizzs dans la méthode `Play` de `Quizz`.

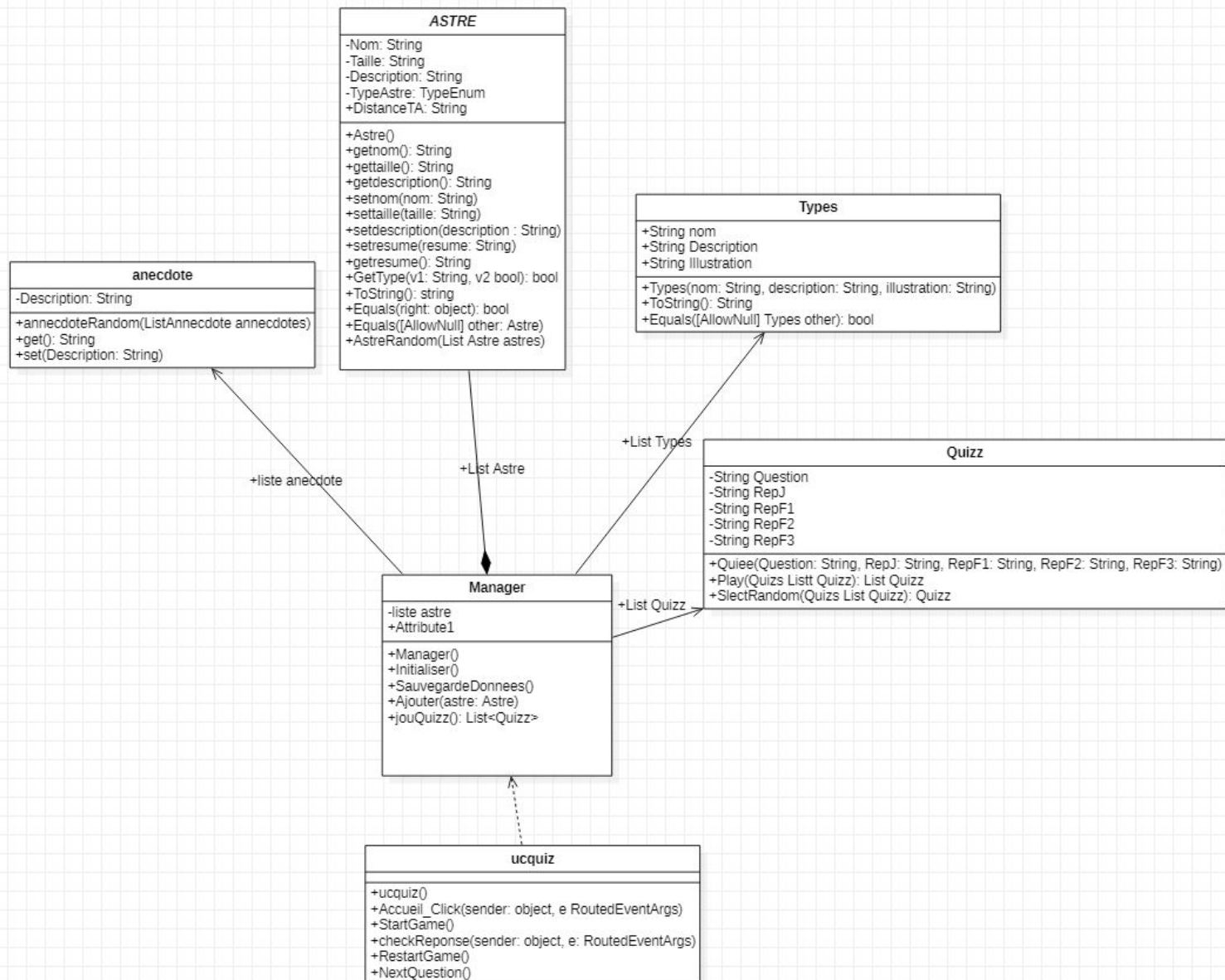
Ensuite, on va faire appel à la méthode `StartGame` qui commence la partie en prenant une question au hasard des 10 questions. Puis on va voir quelle question l'utilisateur a choisie grâce à la méthode `checkReponse`, et en fonction de cela, on va ajouter des points ou non au score du joueur. Ensuite cette méthode fait appel à `Nextquestion` qui va afficher une nouvelle Question.

Enfin, quand les 10 questions sont passées on va aller sur le usercontrol `ucQuizzFinis` qui va nous afficher notre score. Les deux méthodes qu'il possède servent soit à retourner à l'accueil ou refaire sur un nouveau quizz.

Pour le menu nous avons choisi le mettre dans mainwindow. Ensuite lorsque nous cliquons sur l'une des parties du menu, elle s'affiche dans le mainwindow, à côté du menu, il reste alors toujours visible est accessible peu importe où nous sommes.

Le Manager

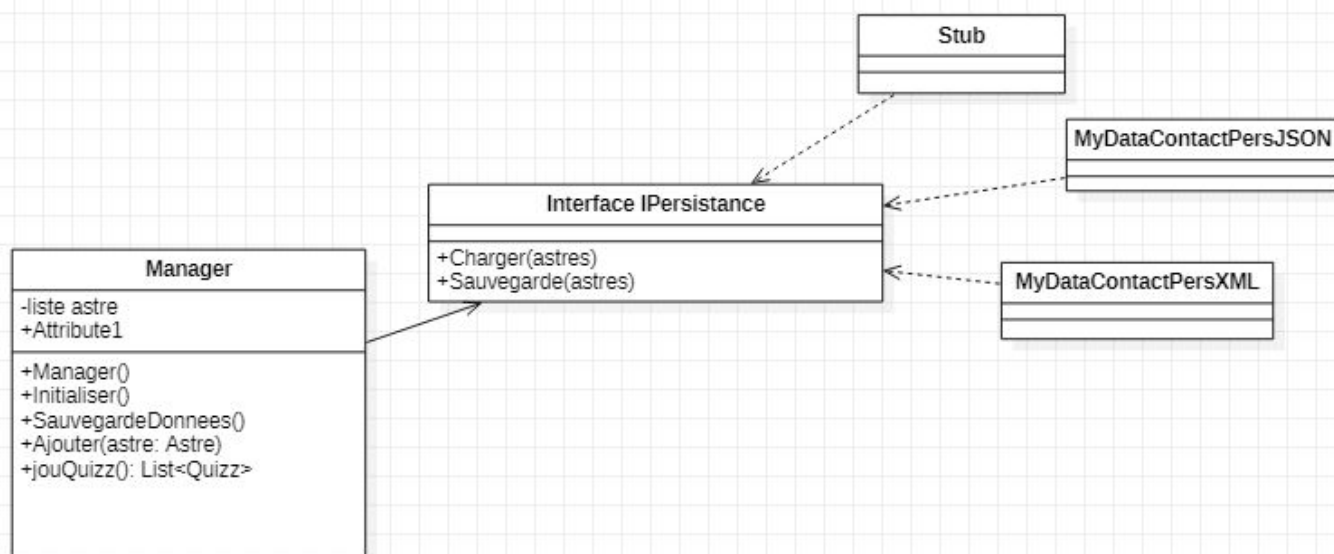
MainWindow
+MainWindow() +search_Click(sender: object, e: RoutedEventArgs) +Astre_Click(sender: object, e: RoutedEventArgs) +Accueil_Click(sender: object, e: RoutedEventArgs) +QUIZZ_Click(sender: object, e: RoutedEventArgs) +Add_Click(sender: object, e: RoutedEventArgs)



Un Manager est une classe public.

Elle possède quatre attributs primaires :

- List <astres> La liste des Astres
- List <anecdotes> Liste des anecdotes
- List <quizz> Liste des quizz
- List <types> liste des Types



On trouve également dans le Manager l'interface :

- persistance du type IPersistence : une instance d'une classe fille de l'interface qui constitue notre persistance

Cette classe sert uniquement d'intermédiaire entre les classes énumérées ci-dessus, ainsi que toutes les vues de l'application modifier, ajouter ou supprimer un Astre. Il possède aussi les méthodes Charger et Sauvegarder utile à la persistance

convRechlmg

+Convert(value: object, targettype: Type, parameter: object, culture: CultureInfo): object
+ConvertBack(value: object, targettype: Type, parameter: object, culture: CultureInfo): object

Création du dossier Convertisseur :

Afin de pouvoir, si amélioration existe, nous avons créé un dossier nommé "Convertisseur" contenant pour le moment uniquement une classe "convRechImg.cs" permettant de convertir une chaîne de caractères contenant le nom d'une image comme "IllLune" en un chemin d'accès vers cette image. Cela nous permet depuis n'importe quelle classe d'appeler une image sans se soucier d'où se situe-t-elle. En revanche, si l'appel de l'image échoue, une image par défaut nommé "illAstre.jpg" s'affiche à la place de l'image inexistante. Pour simplifier le Diagramme de classe, nous avons décidé de ne pas mettre toutes les dépendances de cette classe car elle est utilisée partout ou l'application utilise des images notamment à l'ajout d'astre avec la méthode ConvertBack () ou lors de l'affichage détaillé d'un Astre avec Convert (), elle implémente l'interface IValueConverter.

Patron de Conception :

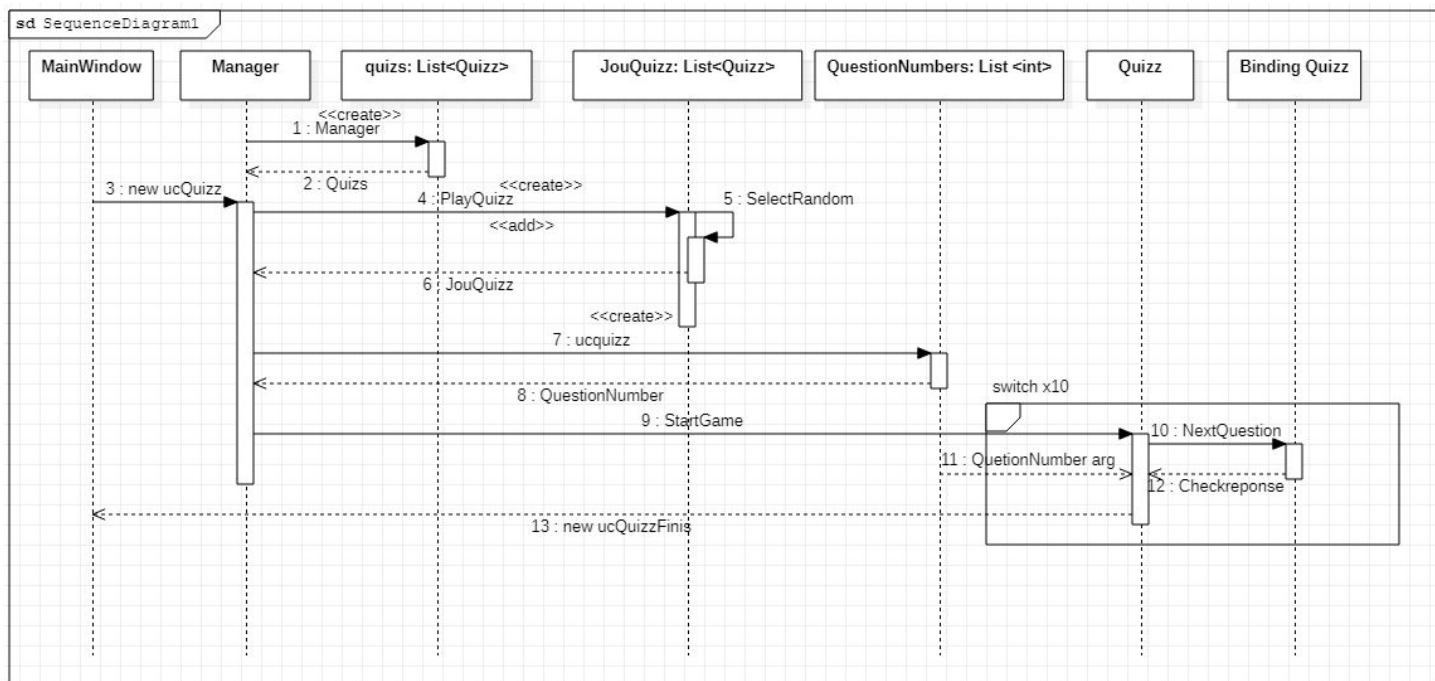
Adaptateur :

La Classe convRechImg permet de convertir le nom d'une image en un chemin d'accès. Grâce à cette classe, peu importe où nous nous trouvons dans l'appli, si on appelle le Convertisseur grâce au nom d'une image, celle-ci sera automatiquement transformé en chemin d'accès et utilisable pour afficher une image par exemple.

Stratégie:

Le patron de Conception stratégie consiste à créer des éléments interchangeable, ici nos deux méthodes de persistance le sont. On peut également imaginer par la suite la création de nouvelles méthodes de persistance interchangeable.

IV - Diagramme de séquence :



Au lancement d'un userControl ucQuizz, une liste de vingt questions est créée par le manager sous forme d'une liste nommé jouQuizz. Depuis la Classe Quizz, sont tirées alors au hasard dix questions depuis la Liste quizzes. Ensuite, une liste contenant des nombres de un à dix est créé et appelé questionNumber puis mélangé aléatoirement dans la méthode StartGame. Les deux listes sont alors accessibles par la méthode NextQuestion qui possède un gros switch contenant dix cases appelées dans l'ordre aléatoire créé précédemment avec la liste questionNumber. Est alors Binder dans l'affichage xaml de l'interface graphique la question puis quatre boutons de réponses dix fois d'affilée en prenant bien le soins de noter le score et de l'afficher à l'utilisateur. À la Fin qui quiz, est appelée l'userControl ucQuizzFinis proposant à l'utilisateur de retourner à l'accueil ou de refaire un quiz, alors la boucle recommence.

Complément d'après création : Considérations ergonomiques

L'application est organisée de manière à être le plus ergonomique possible :

- L'ergonomie Organisationnel : L'application a été conçue pour permettre à l'utilisateur d'accéder à n'importe quelle fonction de l'application en moins de trois cliques excepté les fonctionnalités avancées tel que modifier ou supprimer se trouvant, elle sur les pages de chaque astre. Le menu, accessible depuis n'importe quelle endroit de l'application joue le rôle de point de repère à l'utilisateur, s'il se sent perdu, il saura naturellement ou aller voir pour retrouver son chemin.
- L'ergonomie Cognitive : L'application ayant un but pédagogique, il a été prévu mainte et maintes répétition d'informations dissimulées partout. Par exemple sur la page d'accueil, les anecdotes ont l'aire bénigne, mais l'intérêt que l'utilisateur dévoue à ces anecdotes lui procure l'apprentissage des informations sans qu'il ne s'en rende compte. Le quizz est également un parfait exemple d'ergonomie. En effet le fait de jouer à un mini jeu sur l'espace permet à l'utilisateur de jouer en apprenant. Ainsi, cette manière d'apprendre est l'une des plus efficaces pour le grand public puisqu'il peut facilement mener à un dépassement de soit suivant un objectif, avoir 10/10 au quizz de Space Learner.

Complément d'après création : prise en compte de l'accessibilité :

Lors de la réalisation de notre application nous avons fait en sorte que l'application soit agréable aux yeux avec fond noir. Pour les boutons nous avons voulu les mettre en couleur (bleu) pour qu'il soit facilement visible par l'utilisateur s'il est mal voyant dont les personnes âgées qui font partie du public visé pour notre application. Notre menu est assez gros, il peut donc être facilement lisible et on peut facilement choisir ou aller. La police de notre écriture ne reste jamais trop petite pour que tout soit lisible. Les images sont seulement petites dans la liste des astres mais lorsqu'on clique dessus, nous avons tous les détails et la photo en grand. L'utilisateur n'aura donc aucun mal à avoir accès à tout les possibilités de notre application.

V - Vidéo Fonctionnel du projet