

**Projet tuteuré : Base de la conception orientée objets et IHM**

**Par Pierre BOURDIAUX et Erwan THERON**

Table des matières

[Introduction 2](#_Toc74492055)

[Persona et User story 3](#_Toc74492056)

[Sketchs et description : 6](#_Toc74492057)

[Storyboards 8](#_Toc74492058)

[Diagramme de cas d'utilisation et description 11](#_Toc74492059)

[Considérations ergonomiques 18](#_Toc74492060)

[**Prise en compte de l’accessibilité** 19](#_Toc74492061)

[Diagramme de paquetage 20](#_Toc74492062)

[Diagramme de classes 21](#_Toc74492063)

[**Diagramme de séquence** 29](#_Toc74492064)

[**Partie ajoutée** 31](#_Toc74492065)

# Introduction

Tout d’abord, le but de ce projet est de créer un gestionnaire de cocktails, simple et facile d’utilisation pour tout le monde et donc de recouvrir une grande mixité d’utilisateurs. Le principe est simple, il y a une liste de cocktails déjà présents dans l’application et l’utilisateur peut trouver un cocktail en particulier (avec les détails de la recette, ingrédient, etc.…) grâce aux recherches filtrées, il pourra aussi ajouter des favoris pour les retrouver plus facilement s’il le souhaite. De plus l’application permettra d’enregistrer ses propres créations, en effet si une personne souhaite enregistrer sa recette de cocktail numériquement et la retrouver facilement il pourra le faire. Cette application sera donc la solution pour les fans de cocktails ou ceux qui souhaite découvrir de nouvelles saveurs. Enfin il existe aussi des cocktails sans alcool, c’est pour ça que certain cocktail ont une section sans alcool pour faire plaisir à tout le monde !

# Persona et User story

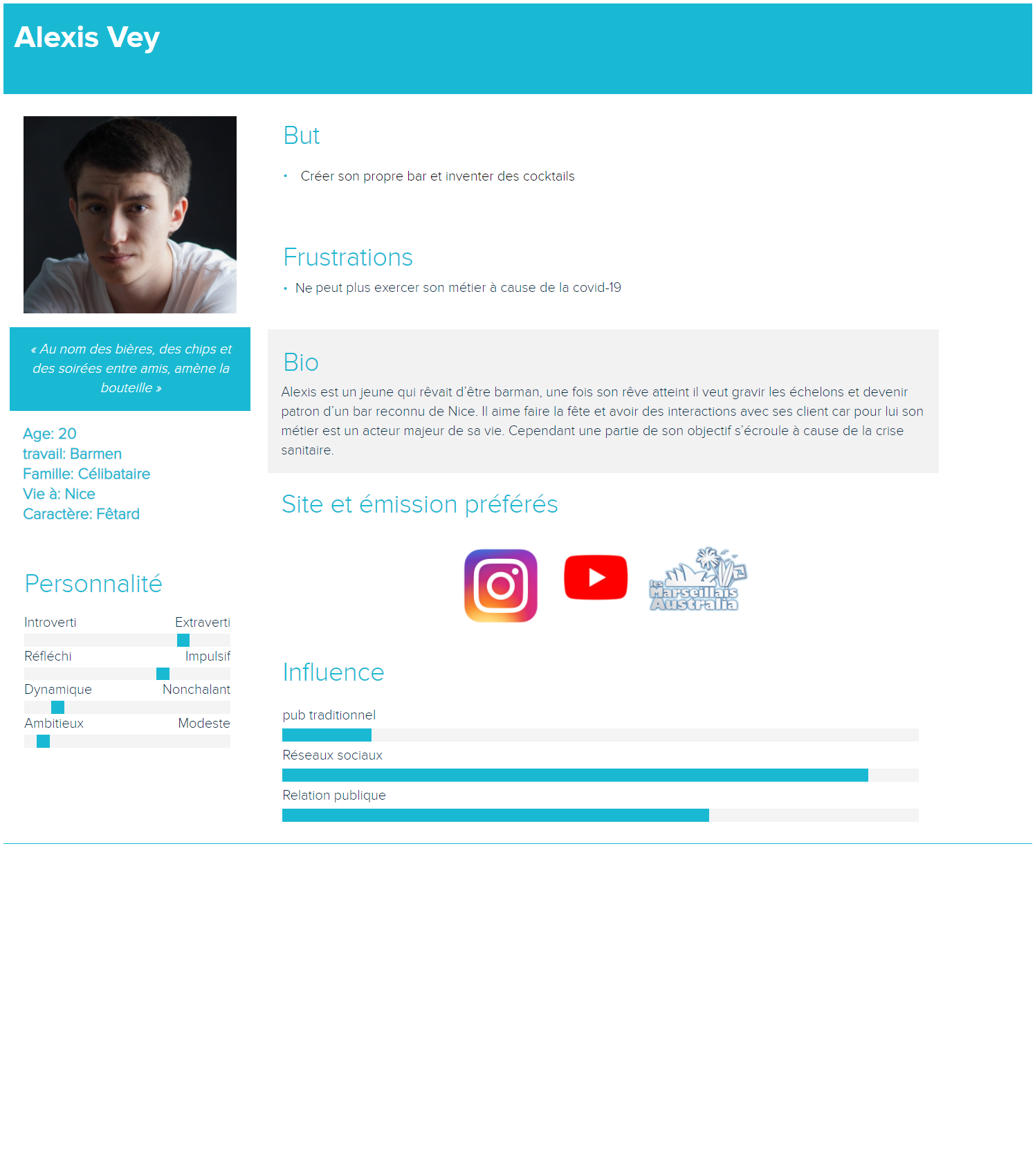
Une image contenant texte

Description générée automatiquement

Un ami d’Isabelle lui a conseillé notre application, elle se connecte dessus et trouve le menu déroulant, un peu perdue elle remarque les zones de recherche. Elle rentre dans la zone de recherche des alcools qu’elle a chez elle et trouve une sélection de cocktail, se rappelant qu’une de ses amies est en plein régime, elle regarde donc les calories des cocktails et en remarque certain avec peu de calorie. Elle ajoute ces alcools à ces favoris pour pouvoir les retrouver plus tard. Elle a plutôt apprécié le site et surtout le fait de voir les calories et les cocktails possibles de faire avec l’alcool à sa disposition. Elle utilisera certainement cette application.



René veut impressionner sa femme pour leurs noces d’Or et refaire le même repas que pour leur rencontre cependant il ne se souvient plus de la recette du cocktail qu’ils avaient pris. René se rappelle l’application qu’un de ses amis lui avait conseillé lors d’un apéritif. Une fois sur l’application, René remarque qu’il y a une section de recherche par nom de cocktail, il rentre donc le nom qu’il se rappelle, “Sex on the beach" et il tombe sur le cocktail qu’il recherche, il clique dessus et a donc accès à toutes les informations sur ce dernier (nom, ingrédient, doses, recette, calories, etc.…). René pourra donc épater sa femme et revivre une soirée comme au bon vieux temps.

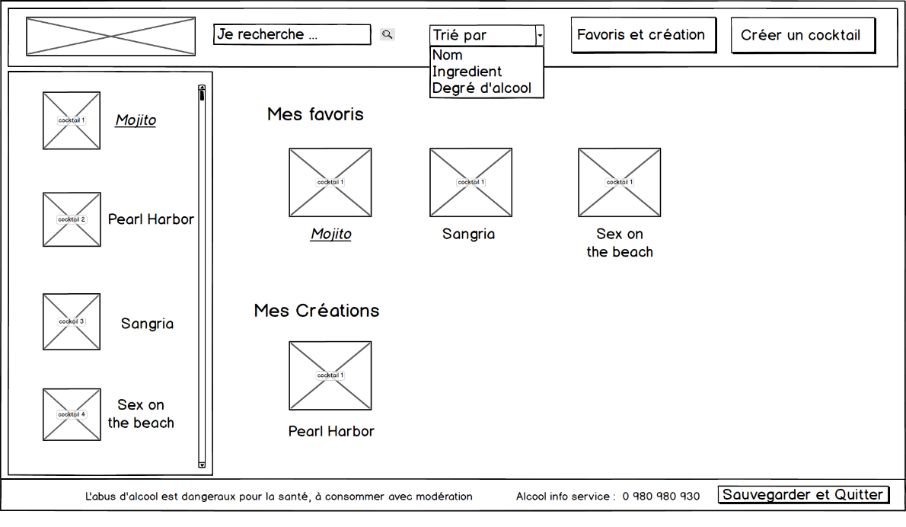


Alexis Vey sort de sa formation de barman juste avant le confinement. Ne voulant pas perdre la main, il décide de s'entraîner à la confection de cocktails. Il trouve une «app» qui référence les cocktails déjà existants et qui permet d’en créer de nouveaux. Il aime bien la partie recette qui décrit bien la façon dont il faut préparer les cocktails et la partie création, qui lui permet de tester de nouvelles idées. Alexis trouve que la section avec les détails l’aide à trouver des cocktails qu’il pourrait proposer à ses clients.

# Sketchs et description :

La description des sketchs se fait de la manière suivante, on suit un fil conducteur et chaque partie indiquée (bouton, menu, …) dans le texte est surligné d’une couleur, cette couleur est reportée sur les sketchs à l’endroit de la partie en question.

Lorsque l’utilisateur arrive sur l’application il arrive sur cette page, il a accès à la liste de ses cocktails favoris et ses créations et il pourra y retourner grâce au bouton situé en haut. Sur la gauche il aura accès à la liste de tous les cocktails triés par nom et, si le nom est souligné, l’information qu’il existe une version soft du cocktail. S’il clique sur l’un des cocktails, il sera redirigé sur cette page.



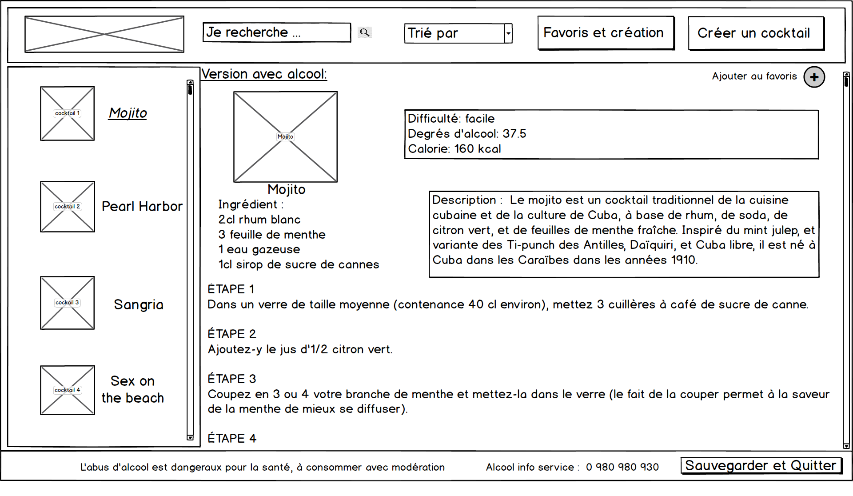
L’utilisateur aura donc le détail de la recette,

Il pourra cliquer sur ce bouton pour l’ajouter à ses favoris.

L’utilisateur voit donc la recette avec ses ingrédients, les étapes, une description, etc…

S’il existe une version soft du cocktail la recette sera renseignée en dessous.

Si le cocktail est une de ses créations, il pourra le supprimer ou le modifier. Si le cocktail n’est pas une création les boutons ne seront pas accessibles.



Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

L’utilisateur peut décider de rechercher un cocktail à tout moment sur différents critères (nom, degrés d’alcool et ingrédient) grâce au menu déroulant. Il peut rentrer son mot clé et cliquer sur la loupe pour lancer la recherche.

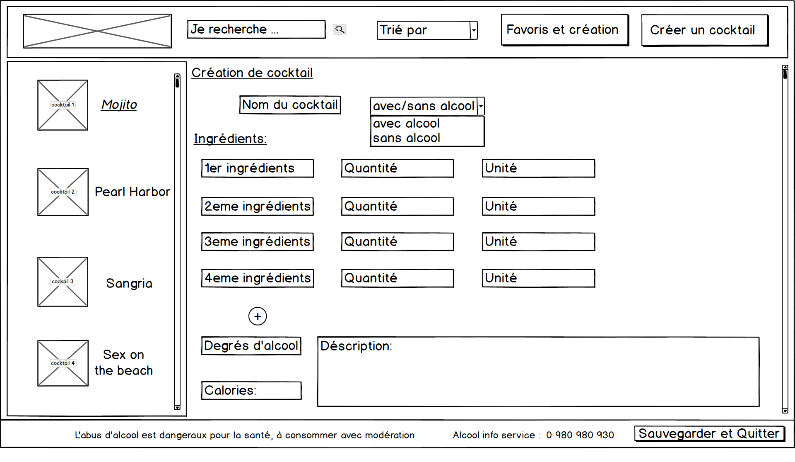
Il aura donc la liste des cocktails correspondant à sa recherche (indication de la difficulté et une description du cocktail), S’il clique sur l’un des cocktails il sera redirigé sur la page vue précédemment.

À tout moment l’utilisateur peut créer un cocktail en cliquant sur ce bouton.

Il est redirigé sur cette page.

Il peut rentrer les différentes informations de sa création (nom, s’il contient de l’alcool, les ingrédients et leurs quantité/unités, le degré d’alcool, les calories, la description des étapes de conception). S’il lui faut plus d’ingrédients, il clique sur ce bouton pour en ajouter d’autres.

Si l’utilisateur est satisfait de la recette il peut l’enregistrer ou l’annuler.

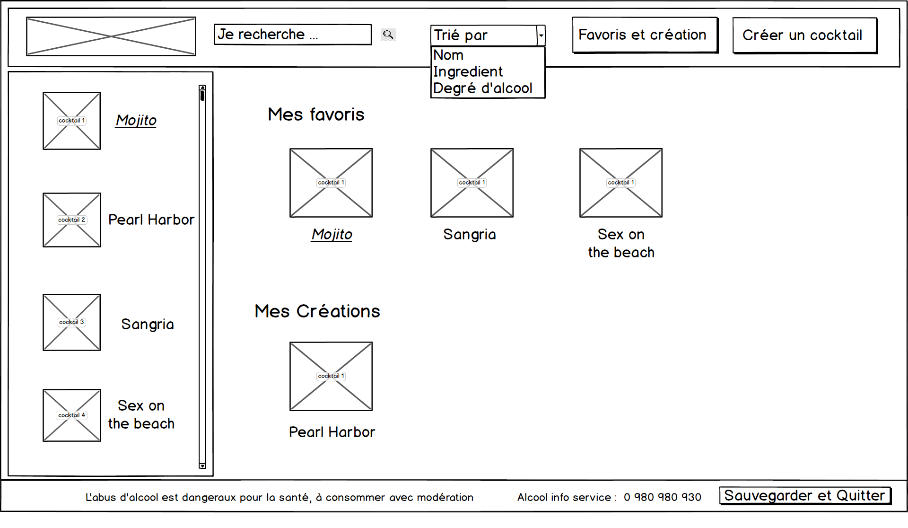


Une image contenant texte

Description générée automatiquement

# Storyboards

Recherche de cocktails :



Si l’utilisateur souhaite faire la recherche d’un cocktail dont il connait le nom il peut chercher en scrollant dans la liste de cocktail trié par ordre alphabétique et faire clic gauche sur celui qu’il désire.

Clic gauche pour lancer la recherche

Ecris le mot clé de sa recherche en fonction du filtre

Clic gauche pour choisir le filtre

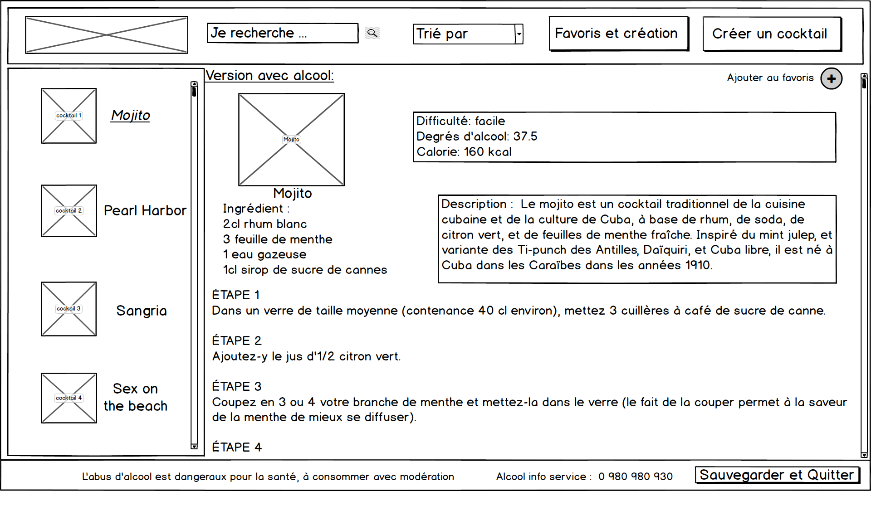
Lorsque l’utilisateur arrive sur l’application il sera sur cette page

L’utilisateur arrive donc sur cette page avec les cocktails les plus pertinents par rapport à sa recherche

Une image contenant texte

Description générée automatiquement

Il peut scroller et faire clic gauche sur le cocktail de son choix

Une image contenant texte

Description générée automatiquement

L’utilisateur arrive donc sur cette page qui détaille le cocktail choisi

Si le cocktail est une de ses créations ces deux boutons seront accessibles

L’utilisateur peut donc scroller pour avoir tous les détails de la recette

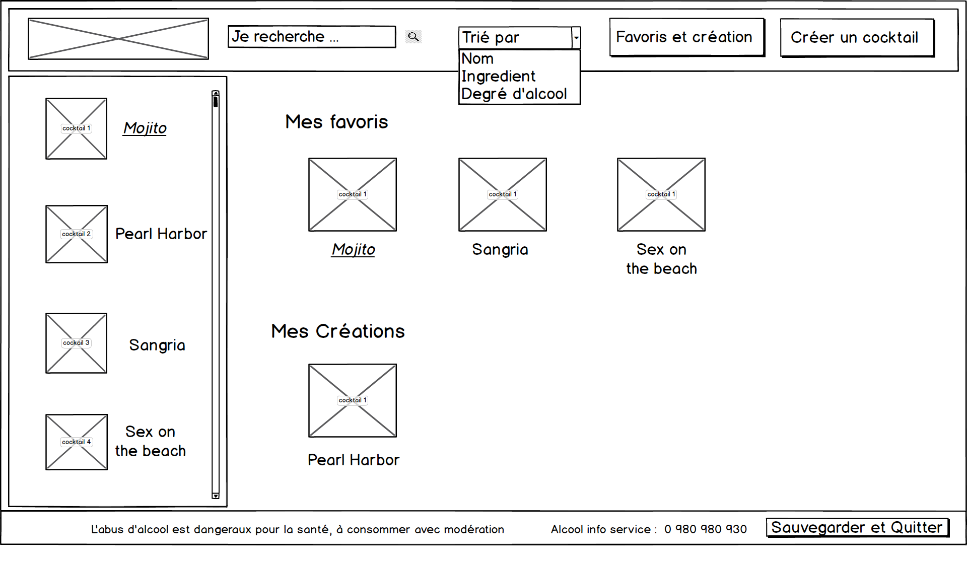
Il pourra avoir accès à ses favoris et création en faisant clic droit ici

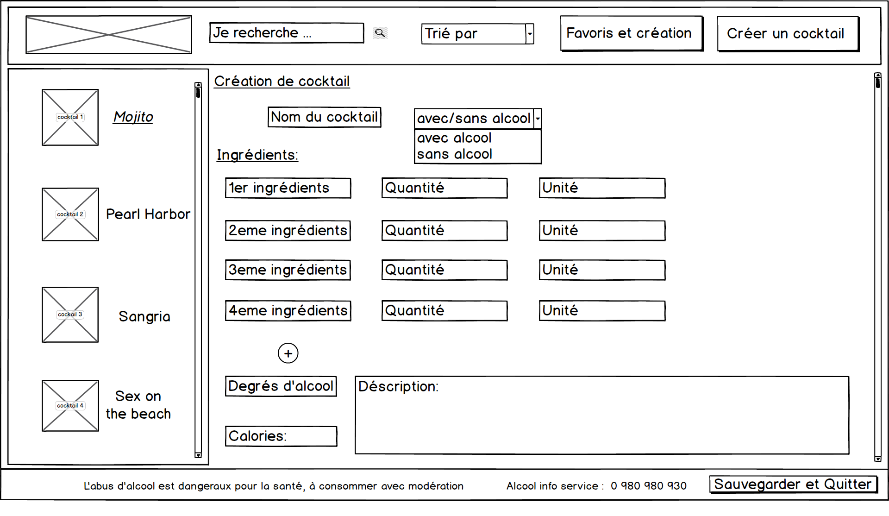
Il peut ajouter le cocktail à ses favoris en faisant clic gauche ici

Création de cocktail :

L’utilisateur veut créer son cocktail,

Il fait un clic gauche sur ce bouton



Une image contenant texte

Description générée automatiquement

S’il est satisfait il enregistre sa recette en cliquant sur ce bouton, il sera donc redirigé sur la recette de ce cocktail

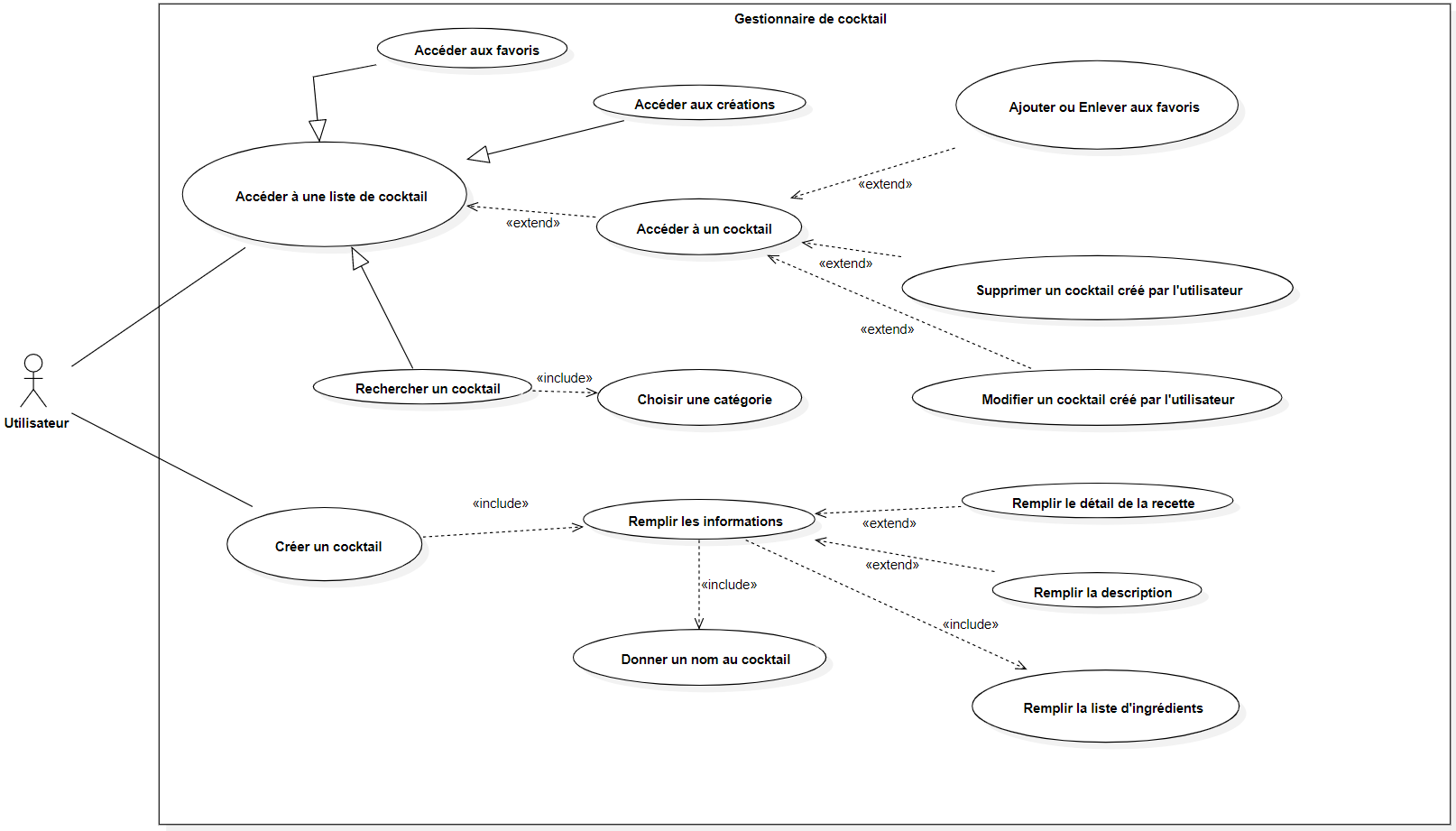
Si l’utilisateur veut annuler la création il clique sur ce bouton, il sera alors redirigé sur la page d’accueil (favoris et créations)

S’il n’a plus de place pour rentrer l’ingrédient il peut en rajouter ici avec un clic gauche

Il rentre donc toutes les informations nécessaires

L’utilisateur arrive donc sur cette page

# Diagramme de cas d'utilisation et description



**Cas “Accéder à une liste de cocktail”.**

|  |  |
| --- | --- |
| Nom | Accéder à une liste de cocktail |
| Objectif | Montrer une liste de cocktail |
| Acteur principal | Utilisateur |
| Condition initiale |  |
| Scénario d’utilisation | -L’utilisateur ouvre l'application  -L’utilisateur fait une recherche. |
| Condition de fin | -L'utilisateur rentre sur la page d’un cocktail.  -Accéder à la création d’un cocktail |

**Cas “Accéder aux favoris”.**

|  |  |
| --- | --- |
| Nom | Accéder aux favoris |
| Objectif | Montrer une liste de cocktails que l’utilisateur a apprécié et enregistrer |
| Acteur principal | Utilisateur |
| Condition initiale | -L’utilisateur a déjà utilisé l’application et a enregistré des favoris |
| Scénario d’utilisation | -L’utilisateur ouvre l’application.  -L’utilisateur clique sur le bouton favoris et création. |
| Condition de fin | -L’utilisateur recherche un cocktail.  -L’utilisateur rentre sur la page d’un cocktail.  -Accéder à la création d’un cocktail |

**Cas “Accéder aux créations”.**

|  |  |
| --- | --- |
| Nom | Accéder aux favoris |
| Objectif | Montrer une liste de cocktails que l’utilisateur a créé |
| Acteur principal | Utilisateur |

Même Condition initiale, Scénario d’utilisation, Condition de fin que “Accéder aux favoris”.

**Cas “Accéder à un cocktail”.**

|  |  |
| --- | --- |
| Nom | Accéder à un cocktail |
| Objectif | Avoir un page qui décrit le cocktail et la façon de le faire |
| Acteur principal | Utilisateur |
| Condition initiale | -Avoir accès à une liste de cocktail |
| Scénario d’utilisation | -Sélectionner un cocktail dans une liste |
| Condition de fin | -Faire une nouvelle recherche  -Retourner sur une liste  -Accéder à la création de cocktail |

**Cas “Ajouter ou Enlever aux favoris”.**

|  |  |
| --- | --- |
| Nom | Ajouter ou Enlever aux favoris |
| Objectif | Ajouter ou enlever un cocktail de la liste des favoris |
| Acteur principal | Utilisateur |
| Condition initiale | -Être sur la page d’un cocktail |
| Scénario d’utilisation | -Ajouter le cocktail dans la liste des favoris  -Enlever le cocktail dans la liste des favoris |
| Condition de fin | -Faire une nouvelle recherche  -Retourner sur une liste  -Accéder à la création de cocktail |

**Cas “Supprimer un cocktail créé par l’utilisateur”.**

|  |  |
| --- | --- |
| Nom | Supprimer un cocktail créé par l’utilisateur |
| Objectif | Supprimer un cocktail de la liste des créations |
| Acteur principal | Utilisateur |
| Condition initiale | -Être sur la page d’un cocktail que l’on a créé |
| Scénario d’utilisation | -Supprime le cocktail |
| Condition de fin | -Quand le cocktail est supprimé |

**Cas “Modifier un cocktail créé par l’utilisateur”.**

|  |  |
| --- | --- |
| Nom | Modifier un cocktail créé par l’utilisateur |
| Objectif | Modifier un cocktail de la liste des créations |
| Acteur principal | Utilisateur |
| Condition initiale | -Être sur la page d’un cocktail que l’on a créé |
| Scénario d’utilisation | -Change les informations du cocktail créé |
| Condition de fin | -Quand le cocktail est modifié  -Abandon de la modification |

**Cas “Rechercher un cocktail”.**

|  |  |
| --- | --- |
| Nom | Rechercher un cocktail |
| Objectif | Trouver un cocktail qui pourrait plaire |
| Acteur principal | Utilisateur |
| Condition initiale | -Accéder à la zone de recherche |
| Scénario d’utilisation | -Rentre la recherche à faire |
| Condition de fin | -Annuler la recherche  -Accéder à une liste liée à la recherche |

**Cas “Choisir une catégorie".**

|  |  |
| --- | --- |
| Nom | Choisir une catégorie |
| Objectif | Trier la recherche par catégorie |
| Acteur principal | Utilisateur |
| Condition initiale | -Accéder à la zone de recherche |
| Scénario d’utilisation | -Choisir la catégorie nom et faire la recherche  -Choisir la catégorie Ingrédient et faire la recherche  -Choisir la catégorie degrés d’alcool et faire la recherche |
| Condition de fin | -Annuler la recherche  -Accéder à une liste liée à la recherche |

**Cas “Créer un cocktail”.**

|  |  |
| --- | --- |
| Nom | Créer un cocktail |
| Objectif | Créer son propre cocktail |
| Acteur principal | Utilisateur |
| Condition initiale | -Accéder à la zone de création de cocktail |
| Scénario d’utilisation | -Créer un cocktail avec les ingrédients disponibles |
| Condition de fin | -Annuler la création  -Sauvegarder la création |

**Cas “Remplir les informations”.**

|  |  |
| --- | --- |
| Nom | Remplir les informations |
| Objectif | Remplir les informations du cocktail qu’on crée |
| Acteur principal | Utilisateur |
| Condition initiale | -Accéder à la création d’un cocktail |
| Scénario d’utilisation | **Obligatoire:**  -Donner un nom au cocktail (**cas**)  -Remplir la liste d’ingrédients (**cas**)  **Optionnel:**  -Remplir le détail de la recette (**cas**)  -Remplir la description (**cas**) |
| Condition de fin | -Annuler la création  -Sauvegarder la création si la partie obligatoire est remplie |

# Considérations ergonomiques

Notre application respecte certaine règle pour optimiser l’ergonomie de cette dernière. Nous pouvons citer la règle des trois clics, qui est une règle non officielle qui consiste à ce que l’utilisateur ai accès à toutes les fonctionnalités en maximum trois clics de souris. Prenons plusieurs cas d’utilisation, si un utilisateur recherche un cocktail en particulier par le nom, il rentre le nom qu’il souhaite cherche, choisi la condition de recherche (1 clic), lance la recherche (2 clics) et a accès à tous les cocktails correspondants. Si maintenant l’utilisateur souhaite voir ses favoris et ses créations, il a juste à cliquer sur le bouton « Favoris et créations » en haut de la page et il y est. Si l’utilisateur souhaite ajouter un cocktail, il clic sur « Créer un cocktail », rentre les informations de son cocktail et clic sur

« Ajouter ». Et cette règle est respecté pour toutes les fonctionnalités de l’application.

De plus, nous avons essayé de prendre en compte la loi de Fitts au maximum, c’est une loi qui donne l’indice de la difficulté à accéder à une tâche, en particulier sur un écran d’ordinateur, dont l’image ci-dessous la représente

Une image contenant texte, équipement électronique, clipart, capture d’écran

Description générée automatiquement

On peut donc voir que les endroits facilement accessibles sont dans les coins ainsi que le centre et les endroits accessible sont tout autour de l’écran tandis que le reste de l’écran est atteignable moins facilement.

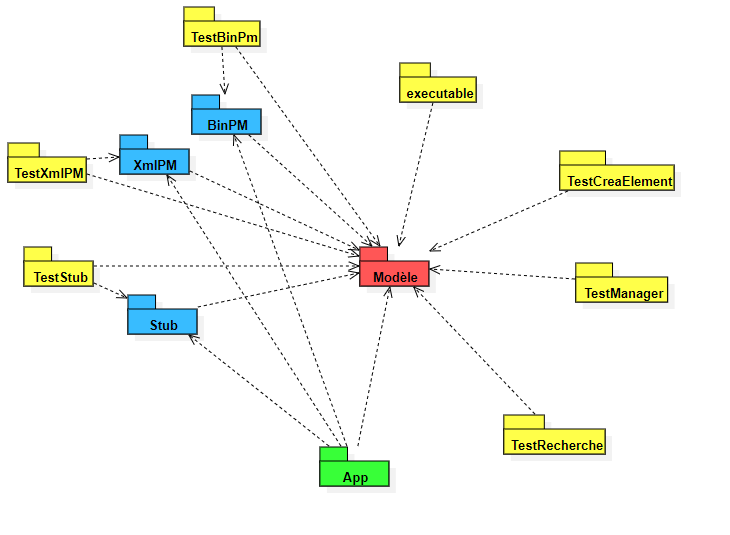
****

Sur notre application on peut voir que toutes les fonctionnalités se trouvent sur les bords de l’écran, elles sont donc accessibles sans trop de problèmes pour l’utilisateur. Les partis les moins accessible sont constitué de la description d’un cocktail, il n’y a donc aucune interaction possible à cet endroit.

# **Prise en compte de l’accessibilité**

Pour notre application, nous avons fais en sorte que même les personnes atteint de problème visuel puissent l’utiliser. En effet les couleurs choisis sont simples : noir, blanc, gris clair et gris foncé. Ces couleurs sont juste les contrastes les unes des autres ce qui facilite la distinction des différentes parties. Le seul endroit où les personnes atteintes de daltonisme pourraient avoir des problèmes c’est le bandeau de prévention rouge en bas de la page, c’est pour cela que nous avons rajouté un icone « Attention » pour que la personne voit que ce qui est écrit ici est important.

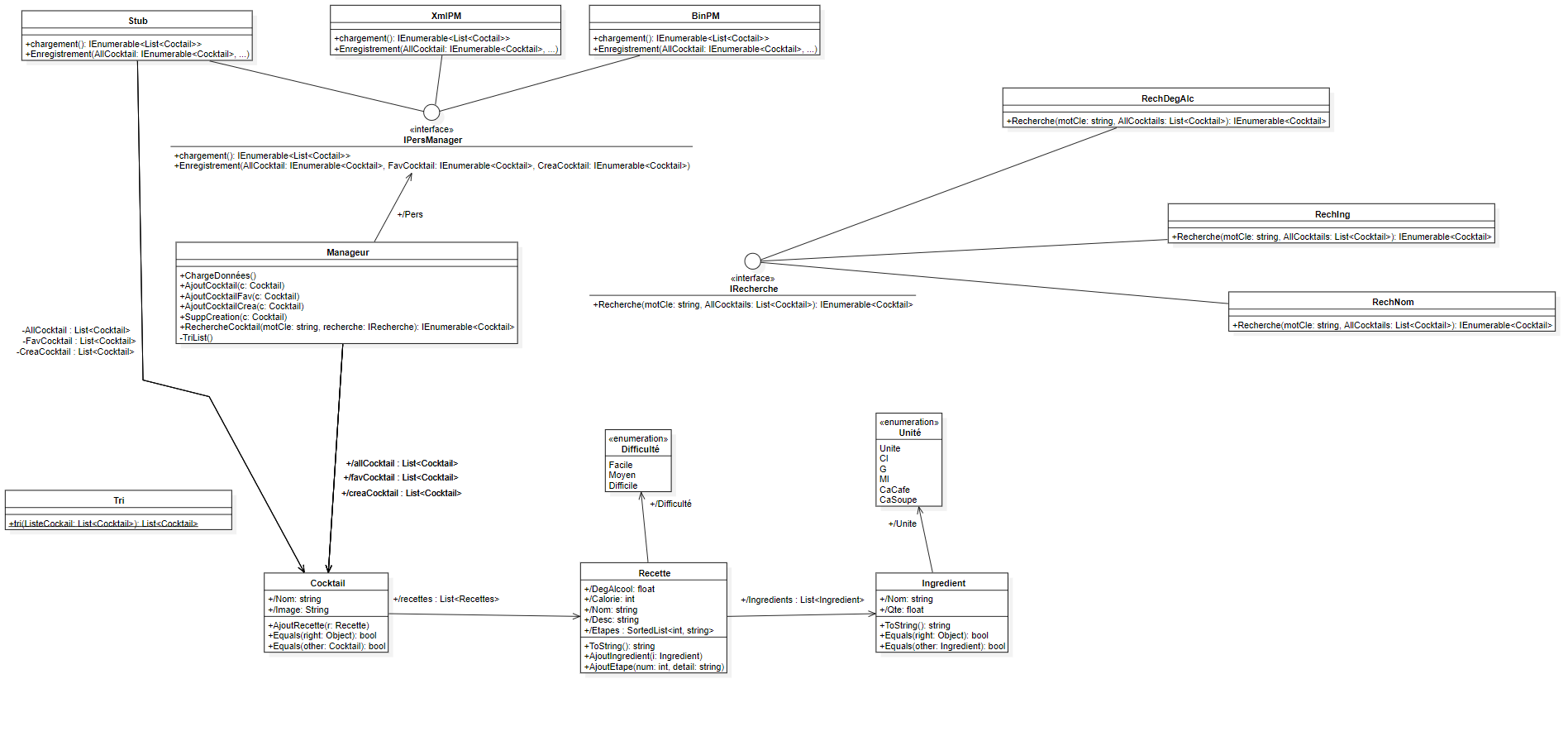
# Diagramme de paquetage



Voici le diagramme de paquetage de l’application, pour les couleurs, le vert représente l’application, le rouge tout le modèle derrière cette application, le bleu toutes les possibilités de persistance (chargement et d’enregistrement des données, choix grâce à une stratégie) et enfin le jaune qui représente tous les tests des différentes classes, persistance etc…

On peut remarquer que tous les paquetages ont une dépendance avec « Modèle » qui se trouve au milieu, pour les tests, ils sont effectués sur des méthodes / constructeur /… du modèle, pour les différents moyens de persistance, chacun d’entre eux sont là pour charger ou enregistrer les données du modèle et les tests de persistances ont une dépendance envers eux car ils utilisent leurs méthodes. Le paquetage app a une dépendance envers « Modèle » mais aussi avec toutes les persistances possibles. Ce choix de persistance sera fait à l’instanciation du manager dans le paquet « App » grâce à une injection de dépendance. On peut donc dire que « Modèle » est au cœur de l’application, tout passe par lui.

# Diagramme de classes



Voici le diagramme de classes de notre application, nous allons voir étape par étape tout le fonctionnement de notre modèle. Dans un premier temps, on verra toute la partie principale du diagramme (classe Ingrédient, Cocktail, Manager, etc..). Puis nous détaillerons la partie de recherche et enfin la partie persistance. Pour chaque partie nous étudierons en détails l’architecture du modèle (patrons de conception, dépendances, …)

**Partie principale**

Une image contenant texte, capture d’écran, intérieur

Description générée automatiquement

*Nous allons détailler le diagramme de la droite vers la gauche, tous les constructeurs ne sont pas renseigné pour ne pas surcharger le diagramme.*

Tous d’abord nous avons la classe « Ingredient » qui contient les informations d’un ingrédient (nom, quantité, unité). Les unités sont renseignées grâce à une énumération, qui sera utile lors de la création de cocktail car chaque possibilité de cette énumération sera accessible dans une « comboBox » pour chaque ingrédient.

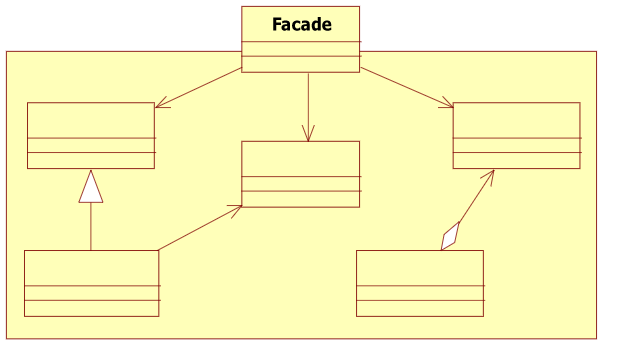
Ensuite nous avons la classe « Recette » qui contient les informations d’une recette. Pour la liste d’ingrédient il y a une composition d’ingrédient grâce à une liste. Chaque ingrédient de la recette peut être ajouté via la méthode « AjoutIngrédient ».

Il y a aussi une liste d’étapes, au début de la conception nous voulions créer une classe dédiée mais finalement nous avons décidé d’utiliser la collections avancée « SortedList » car cette dernière simplifie le code et les listes sont dans l’ordre renseigné, lorsque l’on parcourt la liste, toutes les étapes sont dans l’ordre de leurs numéros respectif et non dans l’ordre avec lequel on les a ajoutées. Chaque étape est ajoutée grâce à la méthode « AjoutEtape ». Enfin nous avons crée une énumération pour renseigner la difficulté.

Après cela, il y a la classe cocktail, renseignant elle aussi toutes les informations sur un cocktail précis, comme son nom, le nom de sont image mais aussi la liste de toutes les recettes disponibles de ce cocktail (pouvant être ajouté grâce à la méthode « AjoutRecette »). Nous avons fait ce choix car chaque cocktail a plusieurs recette possible (avec plus ou moins d’ingrédients, différentes quantités, etc.). Cette liste est utile notamment pour les cocktails qui ont une version sans alcool. Prenons l’exemple du Mojito, il a une recette avec alcool mais aussi une recette nommée « Virgin mojito » qui elle ne contient pas d’alcool.

Puis la plus grosse partie du modèle se situe dans la classe « Manager », cette classe suit le patron de conception « Façade », voyons en détail ce que cela signifie.

Le diagramme UML représentant ce patron de conception est sous cette forme



Source : <http://goprod.bouhours.net/>

Ce patron est de type structurel, la façade est une interface de haut niveau qui permet de gérer plus facilement le sous-système, il est en quelque sorte la porte d’entré du modèle, tout ce qui est extérieur au modèle passe par cette façade, c’est elle qui agit sur les composants du sous-système, leurs propriétés etc. Sachant que le sous-système ne connaît pas l’existence de cette façade, aucun des composant n’a une référence à cette façade.

Dans notre cas, la façade « Manager » sera la porte d’entrée du modèle pour l’application, elle contient toutes les méthodes pour gérer les informations, ajouter des cocktails, les supprimer, les trier, le chargement / enregistrement des données (détaillé plus tard), la recherche de cocktail (détaillé plus tard). Cette classe contient aussi les listes de cocktails, une qui référence tous les cocktails, une autre seulement les favoris et une dernière avec toutes les créations. Chaque liste a une méthode pour qu’on lui ajoute un cocktail, dans ces méthodes on vérifie d’abord si le cocktail est éligible à la liste (si un cocktail est déjà en favoris par exemple pour éviter les doublons), tout comme la méthode de suppression de cocktail qui vérifie si le cocktail est une création car dans le cas échéant la suppression est impossible. La classe manager a aussi un attribut de type « IPerManager » mais ont détaillera cette partie plus tard comme la méthode de recherche. Enfin pour que ces listes soit triées, « Manager » a une méthode « triList » qui trie les trois listes par rapport au nom des cocktails par ordre alphabétique. Cette méthode appelle la méthode statique de la classe tri qui elle aussi est statique. Cette méthode prend une liste en paramètre et la rend triée.

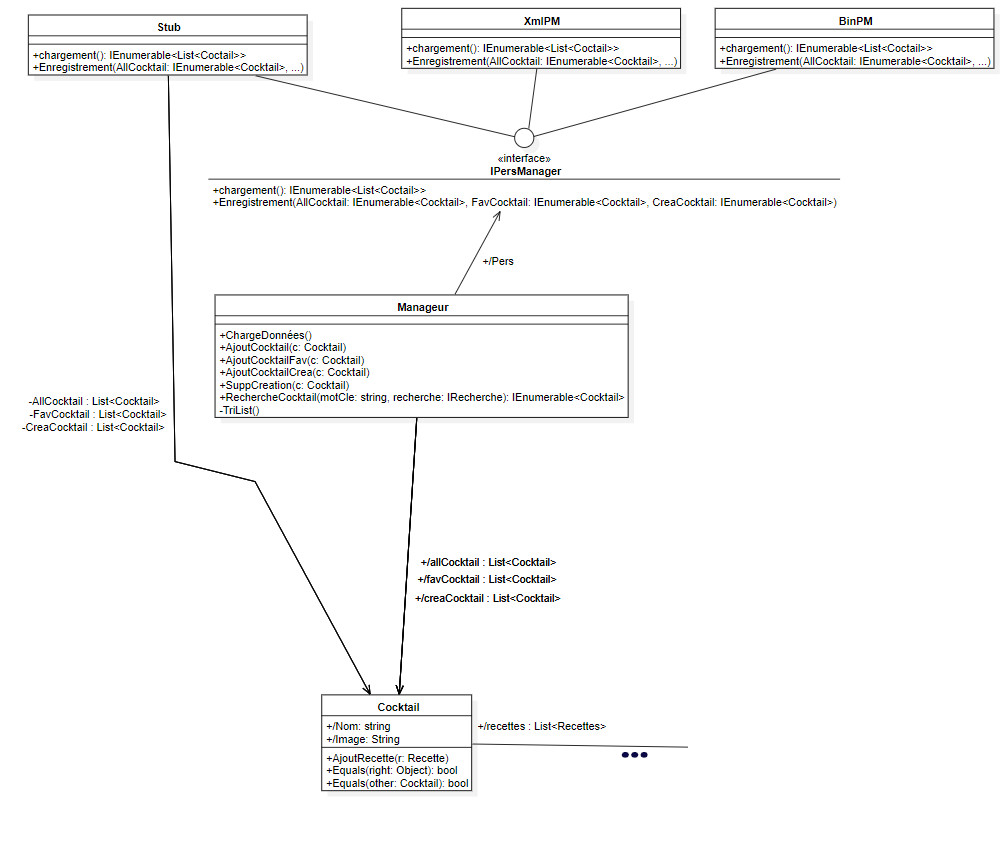
**Recherche de Cocktail**

Une image contenant texte, capture d’écran

Description générée automatiquement

Dans cette partie du diagramme de classes nous verrons en détail la recherche de cocktail. Pour cette partie nous avons trouvé pertinent d’utiliser l’abstraction grâce à une interface. En effet toutes les recherches sont faites à partir d’un mot clé pour avoir une liste de cocktail en sortie, la seule chose qui diffère c’est la condition de recherche, soit par nom, degrés d’alcool ou par ingrédient. Pour choisir la condition on utilise une injection de dépendance grâce à la méthode « rechercheCocktail » du Manager qui, en fonction de la classe de recherche passé en paramètre (donc passant par l’interface IRecherche) effectuera la recherche avec le mot clé renseigné. Le choix se fera donc lors de l’appel de la méthode du manager, elle pourra être lié à une comboBox par exemple. Ce choix d’architecture est utile car si on veut implémenter d’autre possibilité de trie, il suffit d’écrire la classe de trie, l’ajouter à la comboBox et donc pas besoin de modifier du code, seulement de la création. Cette technique ressemble au patron de conception « Stratégie » que nous verrons dans la prochaine partie.

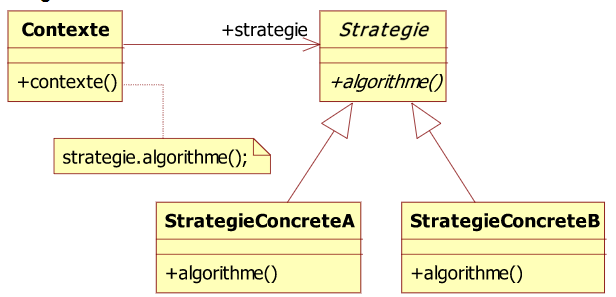
**Choix de Persistance**



Enfin voici la partie « Persistance » du diagramme de classes, cette partie est là pour le chargement / enregistrement des données et les rendre persistantes dans l’application (sauf le stub qui charge des données pour les tests du modèle).

Pour cette partie nous avons utilisé le patron de conception nommé « stratégie », détaillons ce patron de conception.

Voici le diagramme UML de la stratégie.



Source : <http://goprod.bouhours.net/>

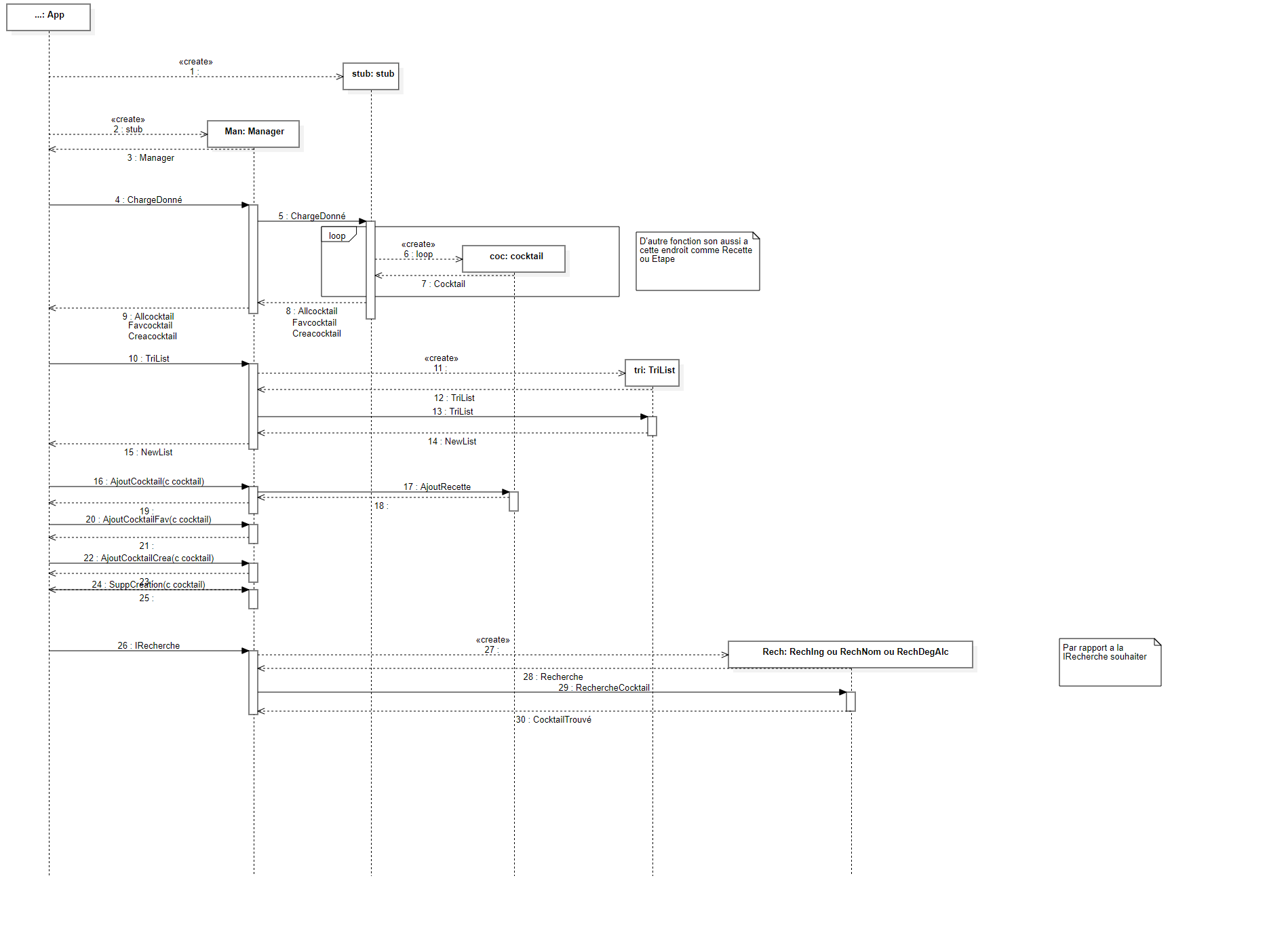
Ce patron de conception est de type Comportemental.

Le principe est de créer une interface, la stratégie (dans notre modèle c’est IPersManager) puis plusieurs stratégie concrète (ici nos différents moyens de persistance comme le stub, la persistance XML etc.) qui hérite de la stratégie principale. On pourrait prendre l’exemple des lacets de chaussure, la stratégie serait faire ses lacets et les stratégie concrète les différents nœuds possibles. Ensuite pour choisir la stratégie que l’on souhaite utiliser, on déclare un attribut dans le contexte du type Stratégie et on l’instancie du type de la stratégie concrète que l’on souhaite.

Si on prend notre cas, la stratégie est « IperManager » et les stratégies concrètes sont « Stub », « XmlPM » et « BinPM ». Pour choisir entre ces stratégie concrètes, on a un attribut Pers qui est définie lors de l’instanciation du manager grâce à une injection de dépendance, ce qui consiste à définir dynamiquement la dépendance entre la persistance et le manager, l’un n’est pas dépendant de l’autre. L’avantage d’utiliser cela est que l’on peut créer par la suite d’autre moyen de persistance et il y aura juste le type de persistance à modifier lors du lancement de l’application.

Chaque stratégie concrète hérite de IPerManager, ce qui veut dire qu’elles ont toutes redéfinis les deux méthodes Chargement et Enregistrement, elles font le même travail mais d’une façon différente (à par le Stub qui ne peut pas enregistrer), Chargement rend trois listes de cocktail (seulement une est renseigner sur le diagramme pour éviter de surcharger) et Enregistrement les enregistre dans un fichier. Ces stratégie concrètes créent aussi trois listes de cocktails lors du chargement de données, sur le diagramme nous avons seulement représenté celles du Stub pour ne pas surcharger le diagramme.

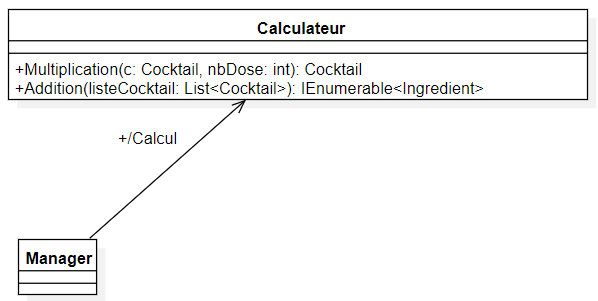
# **Diagramme de séquence**



Pour ce diagramme de séquence on s'est concentré sur les interactions entre “l’application“ et la classe “manager”. Pour commencer on peut voir que l'application commence par créer le stub pour le passer en paramètre à manager lors de sa création. La première fonction qu’on appelle est “ChargeDonné”, la fonction est transmise à “stub” qui grâce à une boucle vas créer les différents cocktails avec leurs descriptions, leurs recettes, les ingrédients etc. Puis vas les intégrer dans différentes listes (Allcocktail pour tous les cocktails / FavCocktail pour les favoris / CreaCocktail pour les créations) puis on les retournes à “manager” qui vas lui aussi les retourner à “App”. On peut voir ensuite que la fonction “TriList” est appelée est créer par “manager”, cette fonction sert a trier les listes qu’on lui donne et les retourne a “App”. Ensuite nous avons quelques fonctions d’ajout et une de suppression. Nous avons “AjoutCocktail” qui permet d'ajouter un cocktail à la liste AllCocktail (Ne peut être fait que par les créateurs de l’app), “AjoutFav” qui ajoute un cocktail dans la liste des favoris et “AjoutCrea” qui permet d'ajouter une création fait par un utilisateur. La fonction “suppCrea” permet de supprimer un cocktail si c’est un cocktail créé par l’utilisateur qui veut le supprimer. Pour ces quatre fonctions, elles sont transmises de “App” à "Manager" et ne retournent rien.

Pour finir, Notre dernière classe de créé dans ce diagramme est une fonction de recherche, cette Classe “Rech” se créer en fonction de la recherche souhaitée, cette condition est envoyée de “App” à “Manager” et ensuite “Manager” envoi la fonction de recherche à la nouvelle classe créer, Cette classe renvoie à “App” une liste de cocktail qui correspond à la recherche souhaitez.

# **Partie ajoutée**



La partie ajouté de notre application est un calculateur, ce dernier permettrais de faire différentes opérations sur un ou plusieurs cocktails. La multiplication prend en paramètre un seul cocktail et un nombre, la méthode retournera une copie du cocktail a l’exception que les doses des ingrédients sont multipliées par le nombre passé en paramètre car toutes les doses renseignées sont par personne, si l’utilisateur veut faire un cocktail mais pour 20 personnes il augmente la quantité de personne et la méthode calculera pour chaque ingrédient la nouvelle quantité. La deuxième méthode est celle de l’addition, elle permettrait de recevoir une liste de cocktail et ressortir une liste d’ingrédient, celle des ingrédients nécessaires pour faire tous les cocktails passé en paramètre