



# CAHIER DES CHARGES

Nom de l'entreprise : UPJV

Nom du projet : Grille de rappel

Personnes à contacter : Mme Rosselle Maryline

# Sommaire

<b>I.</b>	<b>Introduction.....</b>	<b>3</b>
<b>II.</b>	<b>Partie 1 : Cahier des charges.....</b>	<b>3</b>
<b>III.</b>	<b>Partie 2 : Ebauche de planification-Décomposition en tâches et jalons.....</b>	<b>10</b>
<b>IV.</b>	<b>Partie Analyse.....</b>	<b>11</b>
<b>V.</b>	<b>Partie conception détaillée.....</b>	<b>18</b>
<b>VI.</b>	<b>BILAN.....</b>	<b>24</b>
<b>VII.</b>	<b>Référence.....</b>	<b>24</b>

## I) Introduction

Ce cahier des charges contient toutes les informations et ressources nécessaires à l'élaboration du projet « Grille de rappel ». Le but de ce projet est d'obtenir une application interactive où l'utilisateur va pouvoir tester ses connaissances sur ce qu'il sait ou non. L'utilisateur doit obtenir le maximum de points en 5 minutes tout en utilisant sa mémoire rappel et reconnaissance. Les questions posées rapportent plus de points en fonction de la date de la question ! (une question datant d'un mois rapportera plus de point qu'une question datant d'une semaine par exemple).

## II) Partie 1 : Cahier des charges

### A) Besoins et objectifs

L'organisme demandeur est Mme Marilyn Rosselle, enseignante et chercheuse à l'Université de Picardie Jules Verne située au pôle scientifique Saint-Leu, 33 rue Saint-Leu, 80039 Amiens Cedex 1. Pour contacter l'organisme nous avons le numéro de téléphone qui est le 03 22 82 59 21 mais aussi le mail [marilyne.rosselle@u-picardie.fr](mailto:marilyne.rosselle@u-picardie.fr).

Notre client souhaite obtenir une application interactive nommée « Grille de rappel ». Cette application est destinée à être utilisée par les étudiants de l'UPJV en premier lieu, afin de vérifier s'ils connaissent bien leurs cours en s'entraînant de manière ludique.

Par exemple si un étudiant en informatique veut réviser son cours de réseau, il doit pouvoir créer une nouvelle session en rentrant des questions du cours dans un fichier. Il pourra ouvrir ce fichier plus tard pour réviser cette partie à nouveau.

L'application devra pouvoir être utilisée par des enfants.

### 1. Moyens et contraintes

Deux étudiants travailleront sur ce projet. Ils travailleront sur leur machine personnelle.

Cette application ne doit fonctionner qu'à l'aide de la souris pour que même un enfant puisse jouer avec. Nous devons créer une application ludique où 12 questions devront apparaître aléatoirement. Ces questions pourront être renseignées sur un fichier de questions. La question des moyens n'a pas été abordée. Le projet est susceptible d'être poursuivi sur plusieurs années. Cependant notre client souhaite obtenir une première version pour avril 2021.

## 2. Contrainte technique et architecture

Nous ne devons pas créer une application web mais une application interactive de la technologie que l'on souhaite. Le langage de programmation est au choix. Notre application devra en premier lieu fonctionner sur une machine personnelle. Par la suite, il pourra être possible de l'installer sur les serveurs de l'UPJV. Ainsi notre application devra pouvoir s'installer sur les machines de l'UPJV sous le système d'exploitation Linux.

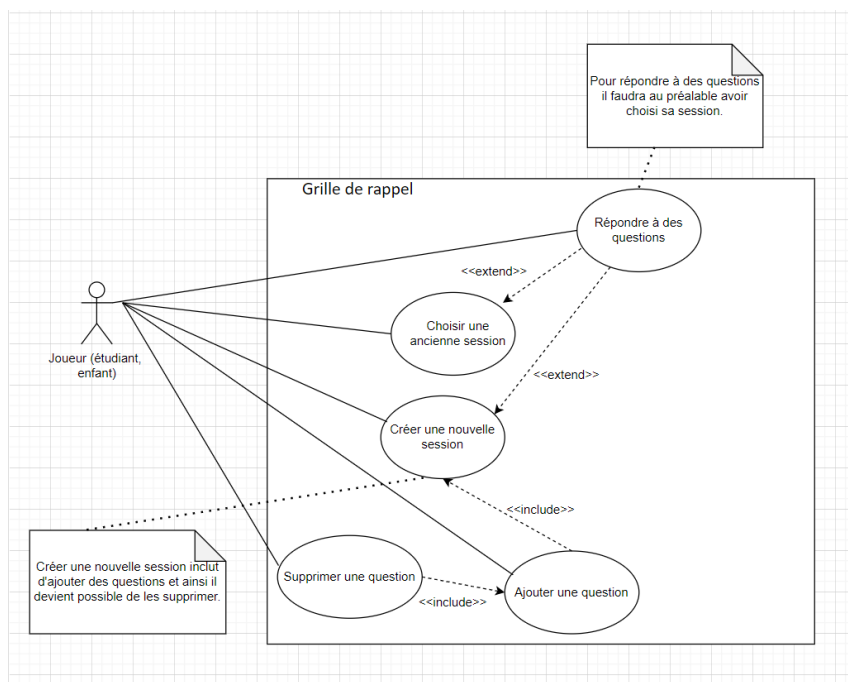
### B) Analyse de l'existant, concurrence et positionnement

Il n'existe pas vraiment d'applications web qui permettent de réaliser ce que l'on souhaite. En effet nos clients demandent une application web (et pas autre chose), plus des réponses précises. On est donc les premiers sur le marché !

Cependant il existe déjà des sites web qui permettent aux utilisateurs de créer leurs propres grilles de rappel.

Jeretiens (1) est un exemple de site web qui explique à l'utilisateur comment réaliser ses propres grilles de rappel.

### C) Offre fonctionnelle de votre produit



### D) Description de chacune des fonctionnalités

Nom fonctionnalité : Répondre à des questions

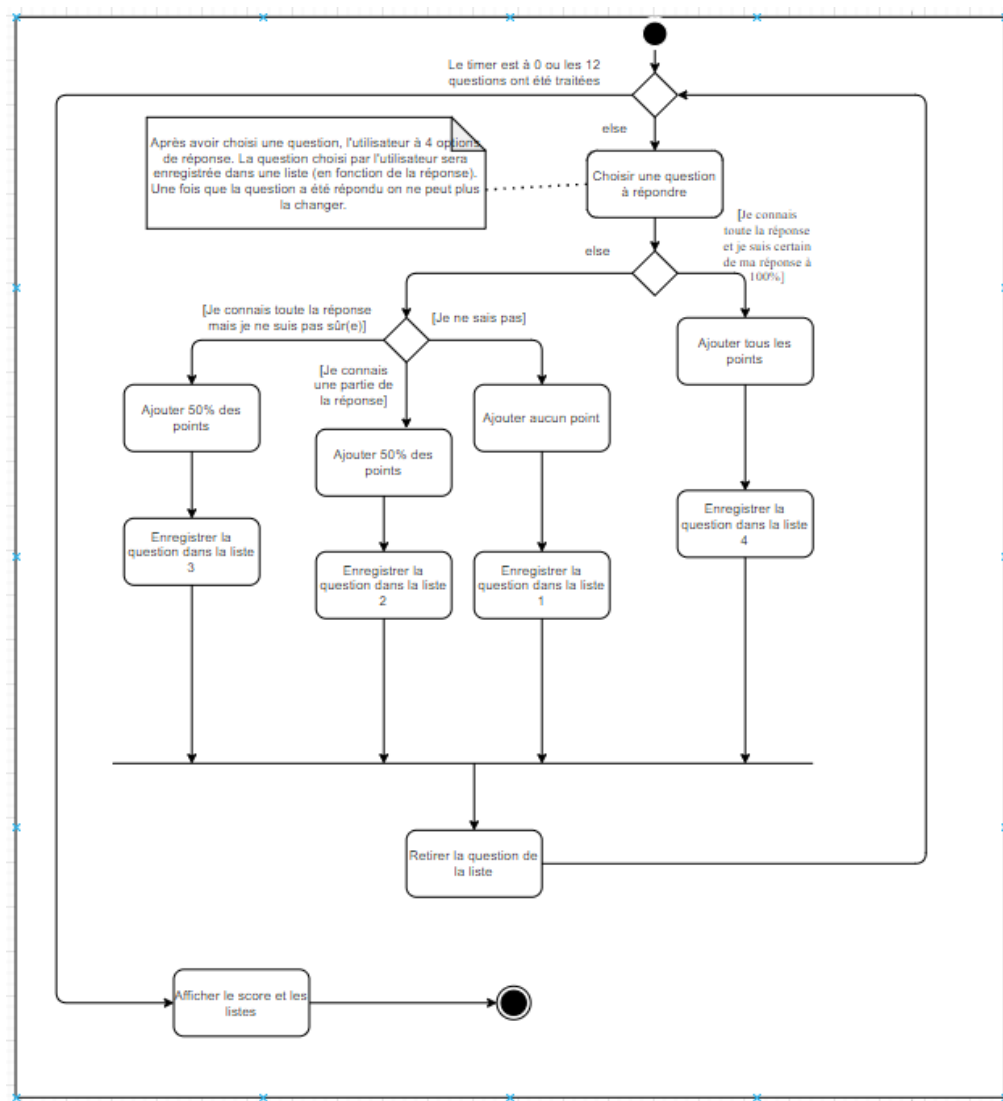
Utilisateurs : Joueur (étudiant, enfant)

Description fonctionnalité : Lorsque le joueur lance une partie, un tableau de 12 questions aléatoires apparaîtra. Le joueur a 5 minutes pour obtenir un maximum de points. Lorsque le joueur clique sur une question, il a 4 choix possibles :

- Je ne sais pas
- Je connais une partie de la réponse
- Je connais toute la réponse, mais je ne suis pas sûr(e)
- Je connais toute la réponse et je suis certain de ma réponse à 100%

Des points sont attribués en fonction de la réponse donnée ci-dessus.

Plus la question est ancienne, plus elle rapporte de points. A la fin, le jeu fait un affichage des questions pour lequel le joueur a su répondre, des questions où il n'a pas su répondre, ...



Règles de gestion : Tout doit pouvoir se faire à la souris !

Des questions doivent être rentrées pour pouvoir lancer une partie. Sinon un message d'erreur doit être renvoyé invitant le joueur à rentrer des questions.

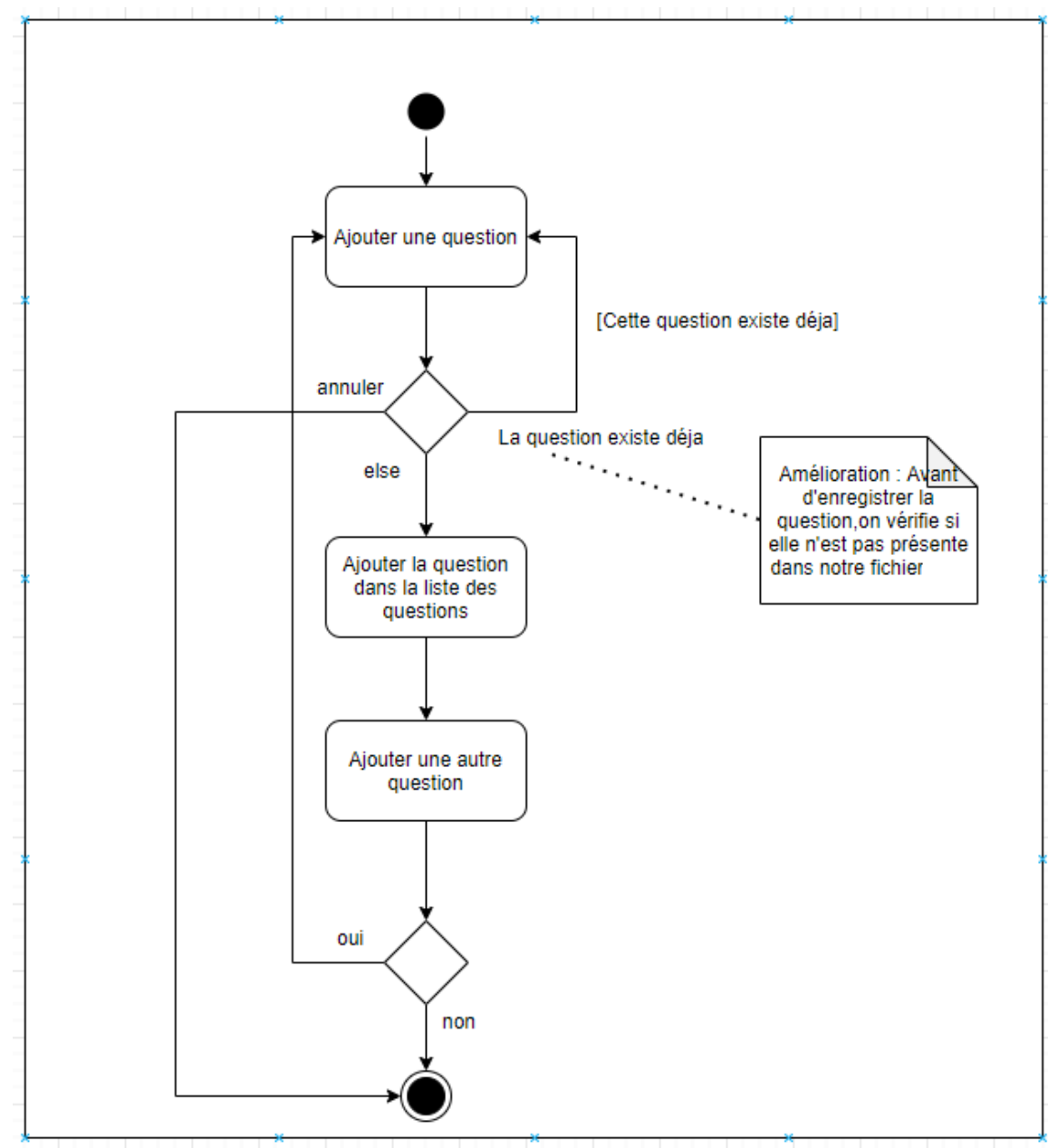
Autres contraintes :

Priorité : M : C'est la fonctionnalité la plus importante de l'application (le cœur). Sans celle-ci on ne respecte pas le projet !

Nom fonctionnalité : Ajouter une question

Utilisateurs : Joueur (étudiant, enfant)

Description fonctionnalité : Le joueur pourra choisir de créer une nouvelle question.



Amélioration : Il faut vérifier que la question n'existe pas dans notre fichier.

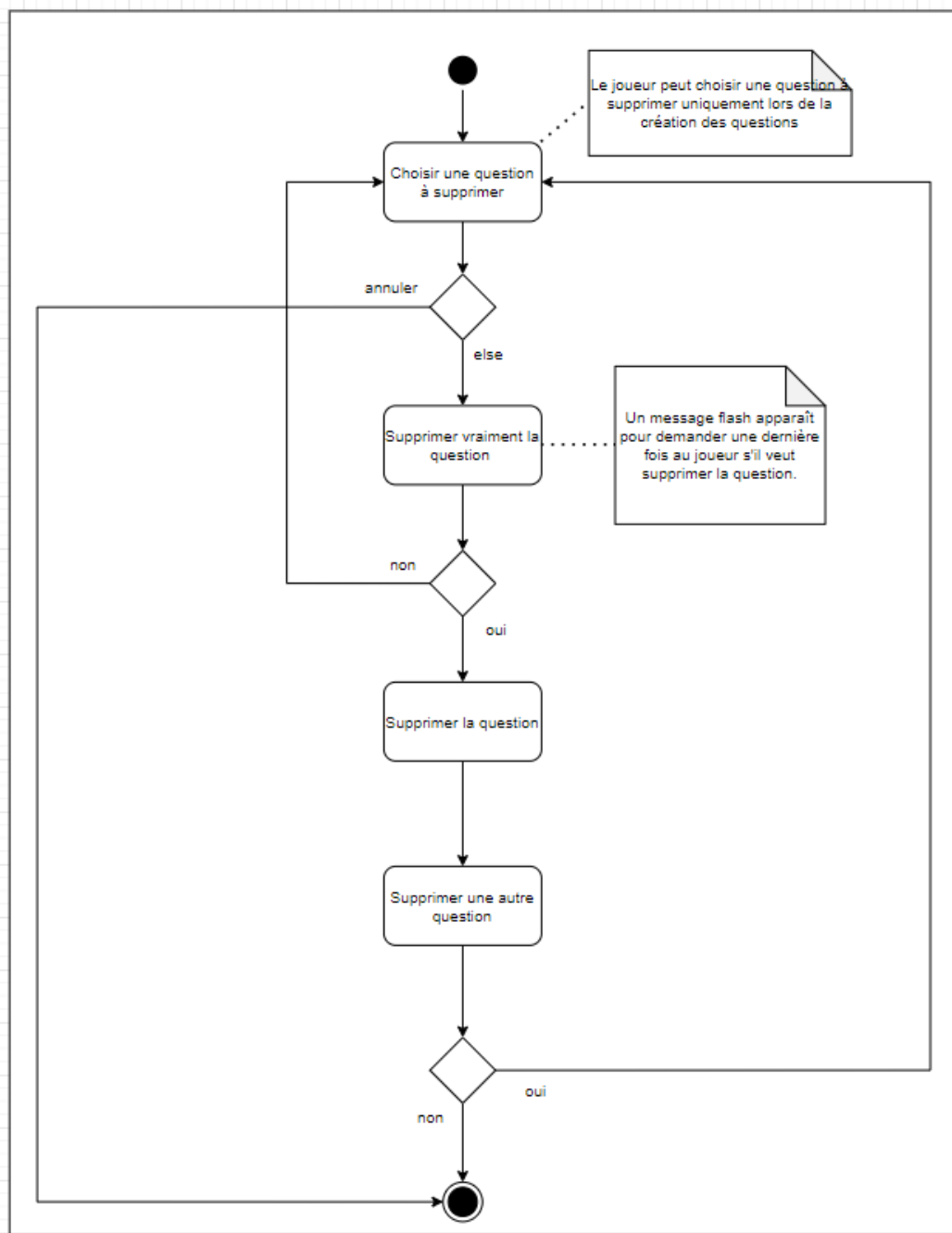
Autres contraintes :

Priorité : M : Une fonctionnalité très importante puisqu'il faut implémenter des questions pour commencer à répondre à des questions.

Nom fonctionnalité : Supprimer une question

Utilisateurs : Joueur (étudiant, enfant)

Description fonctionnalité : Le joueur pourra choisir de supprimer une question existante.



Règles de gestion :

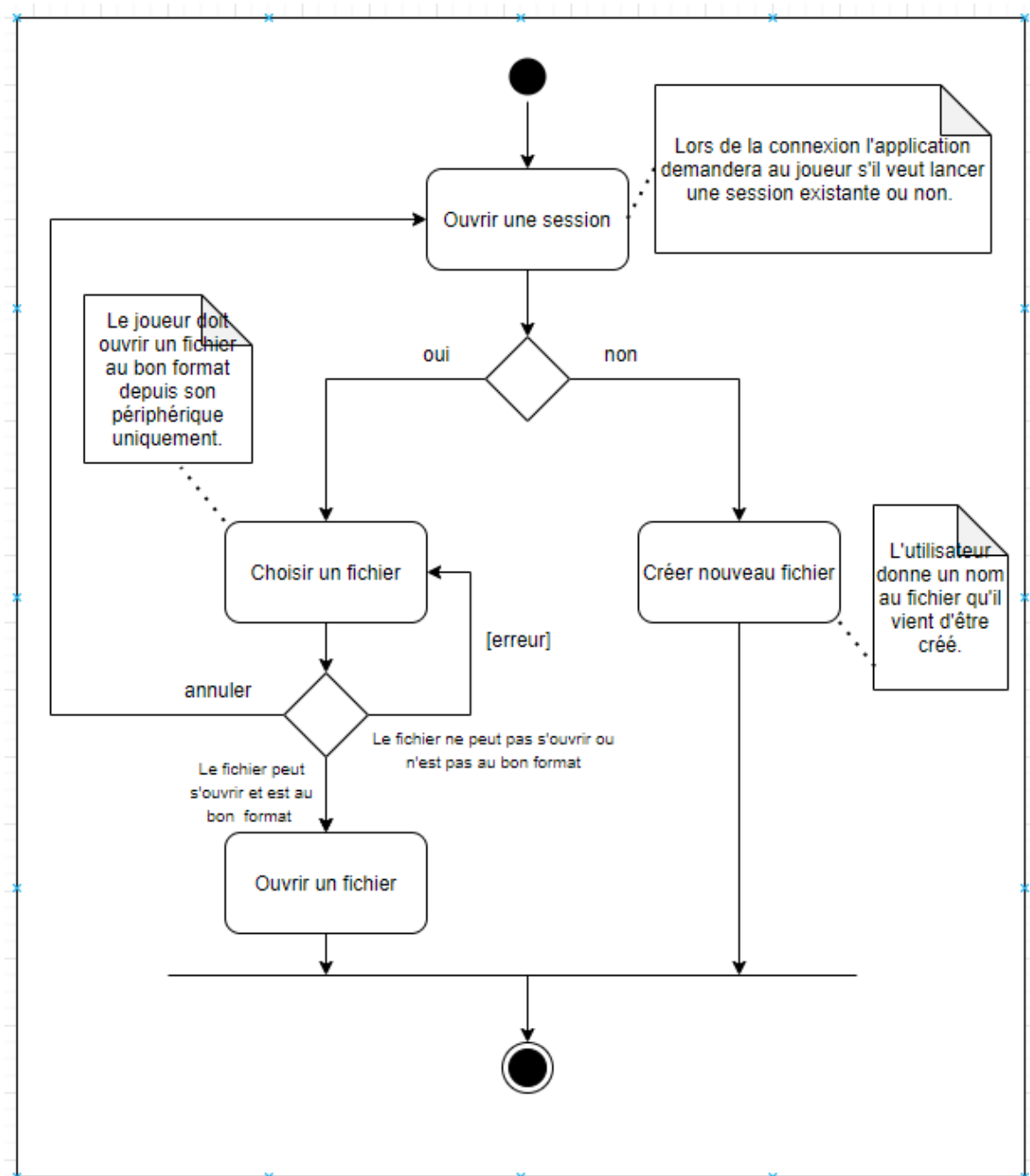
Autres contraintes :

Priorité : M : Une fonctionnalité à implémenter. En effet l'utilisateur doit pouvoir supprimer une question qui juge ne pas être utile.

Nom fonctionnalité : Choisir une nouvelle session

Utilisateurs : Joueur (étudiant, enfant)

Description fonctionnalité : Le joueur peut choisir lors du lancement de l'application de créer une nouvelle session. Un fichier sera créé et il pourra inscrire des questions.



Règles de gestion : Le fichier créé est au bon format.

Autres contraintes :

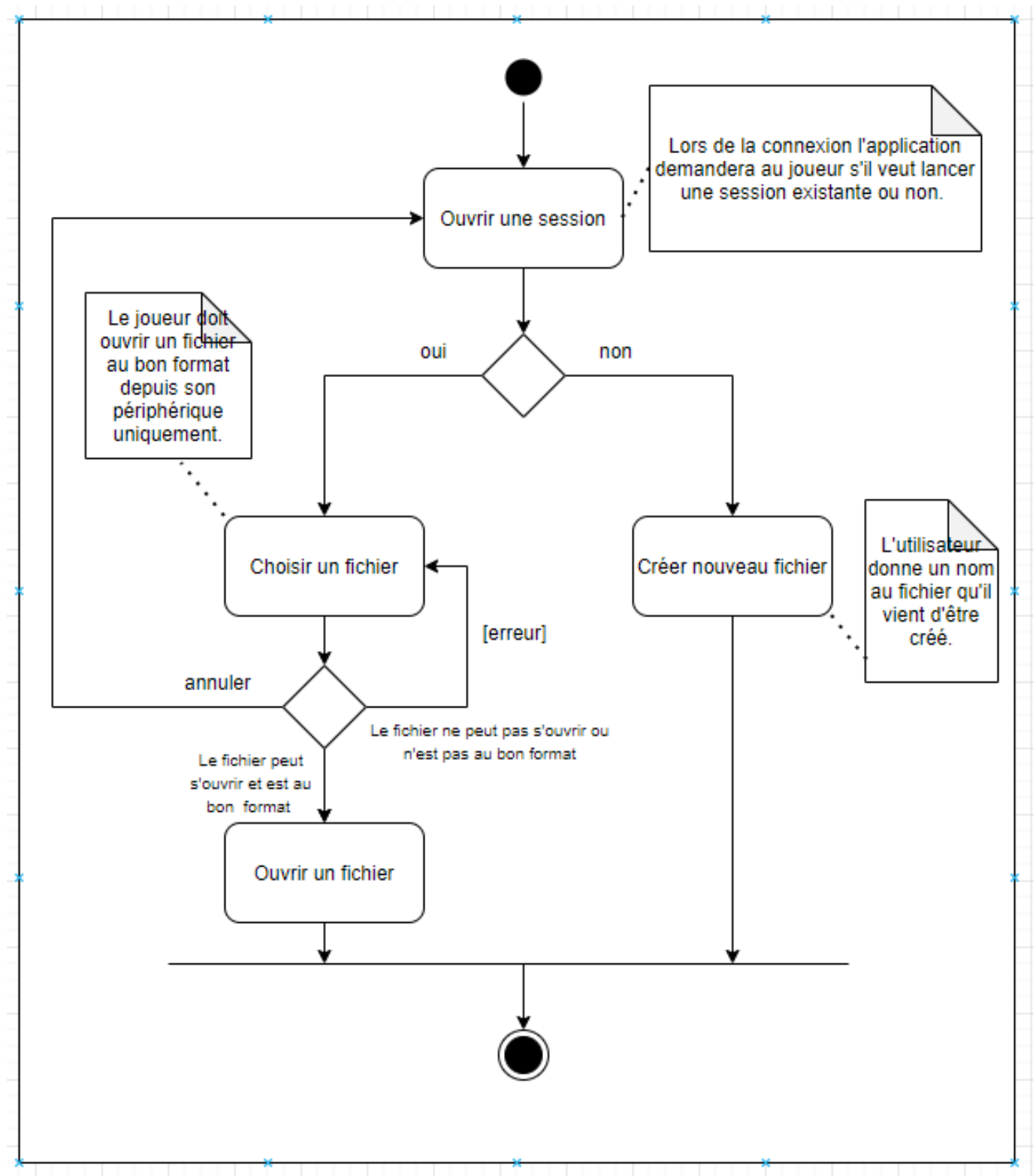
Priorité : M : L'utilisateur doit pouvoir créer une session quand il le souhaite ! S'il ne peut pas créer ses propres questions, ce n'est pas intéressant...



Nom fonctionnalité : Choisir une ancienne session

Utilisateurs : Joueur (étudiant, enfant)

Description fonctionnalité : Le joueur pourra choisir d'ouvrir un fichier json contenant des questions.



Règles de gestion : Le fichier doit être au bon format et lisible par l'application.

Autres contraintes :

Priorité : M : Si on ne peut pas lire de fichier, on ne peut pas lancer le jeu.

### III) Partie 2 : Ebauche de planification-Décomposition en tâches et jalons

#### A. Partie obligatoire

<u>Lettre</u>	<u>Description de la tâche</u>	<u>Durée allouée</u>	<u>Tâches antérieures</u>
A	Choisir une nouvelle session	3 jours	
B	Choisir une ancienne session	3 jours	
C	Ajouter/Supprimer des questions	3 jours	A, B
D	Répondre à des questions	1 jours	C
E	IHM	5 jours	A, B, C, D

#### B. Partie optionnelle : Le GANT

Voir livrable « GANT »

### Conclusion

Ce cahier des charges doit permettre la mise en place des différentes fonctionnalités que doit comporter notre application. La première version devra contenir au moins les fonctionnalités présentées tout en respectant les moyens et contraintes.

## IV) Partie analyse

### 1) Introduction

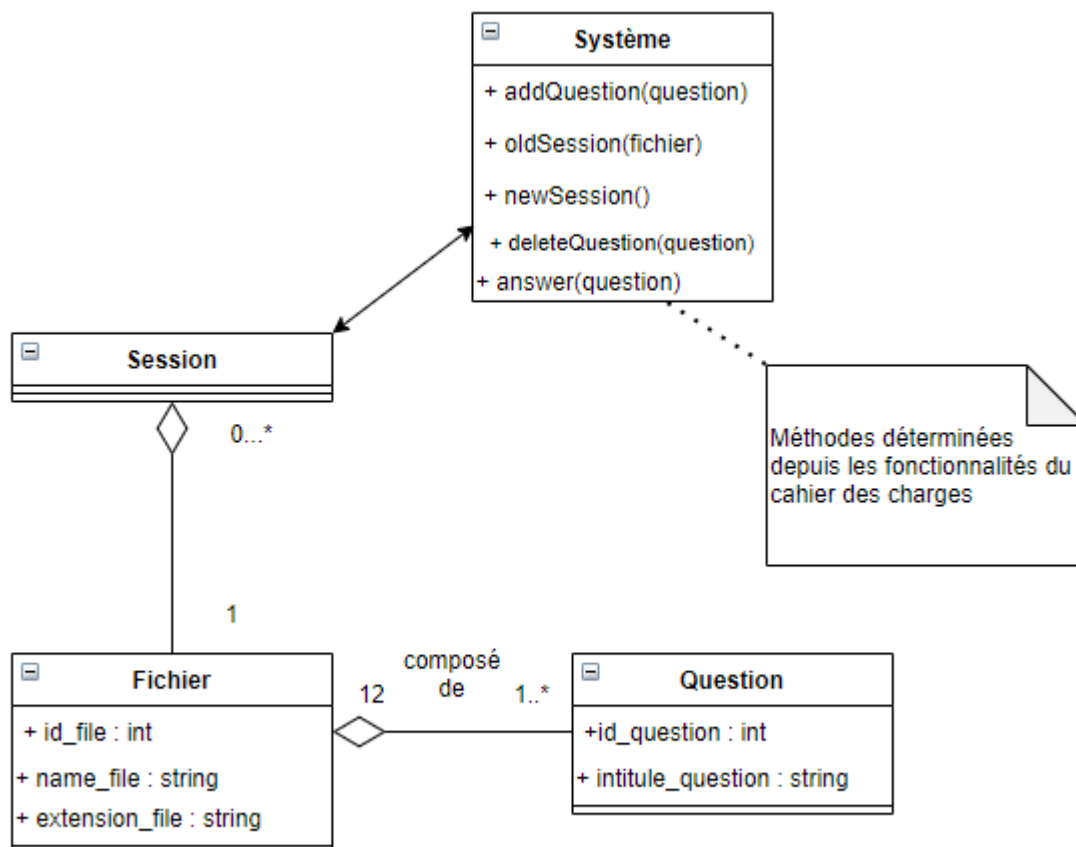
Le cahier des charges décrit les différentes caractéristiques et fonctionnalités que souhaite notre client pour le projet de la « grille de rappel ».

Pour rappel le but de ce projet est d'obtenir une application interactive où l'utilisateur va pouvoir tester ses connaissances sur ce qu'il sait ou non.

Dans ce livrable nous allons aborder la partie analyse en répondant à la question « Avec quoi allons nous réaliser ce projet ? ».

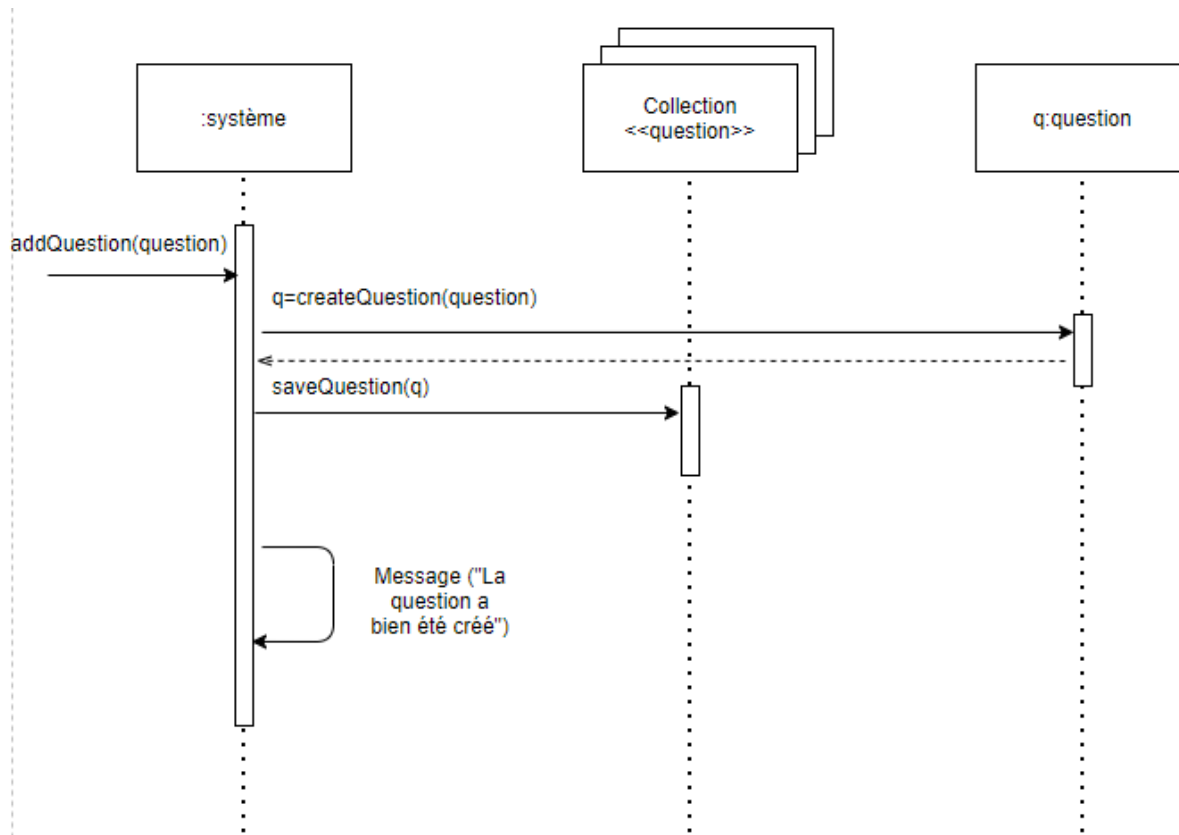
### 2) Analyse

#### A) Diagramme d'analyse initial

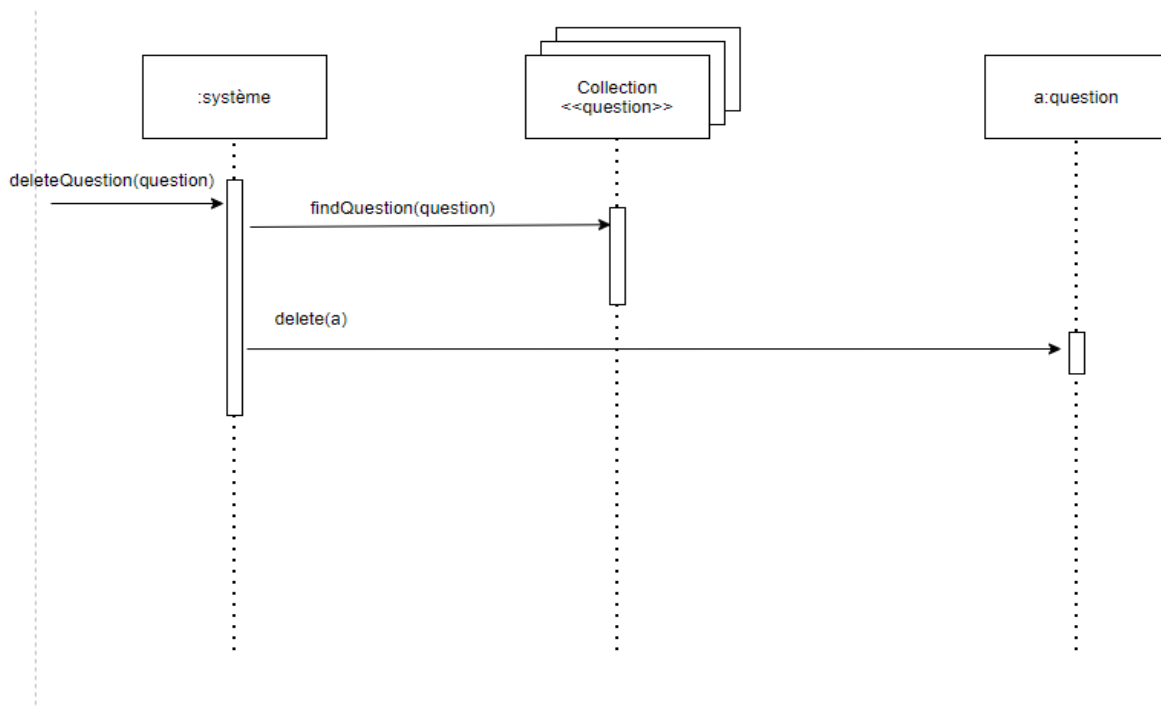


## B) Diagramme de séquence d'analyse

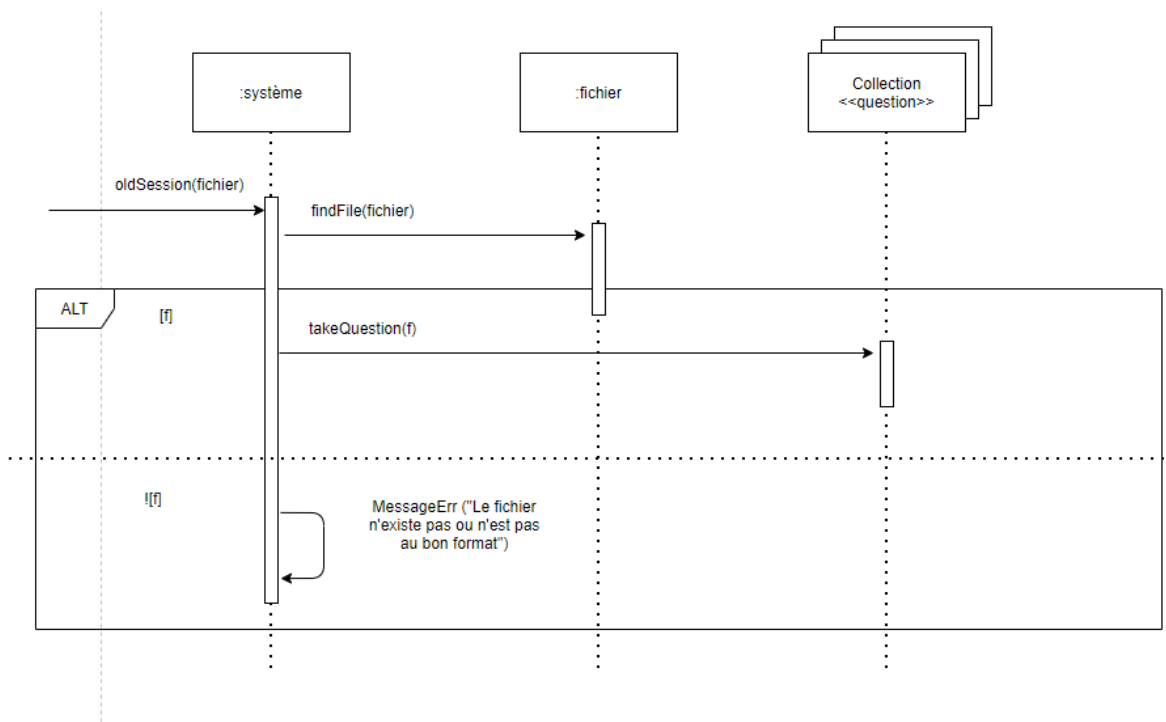
- addQuestion(question)



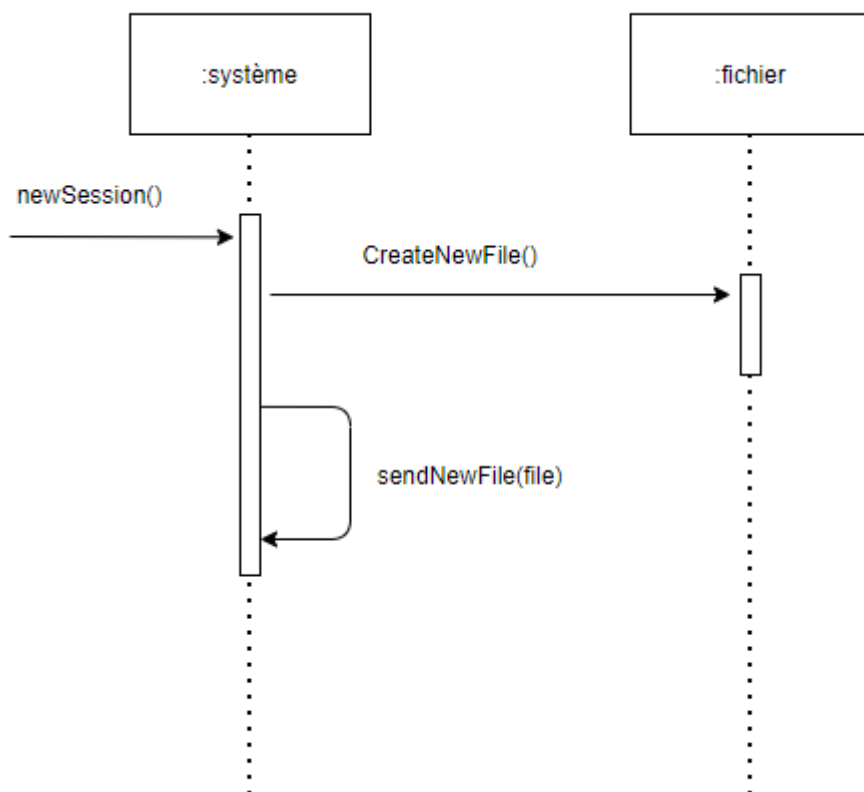
- deleteQuestion(question)



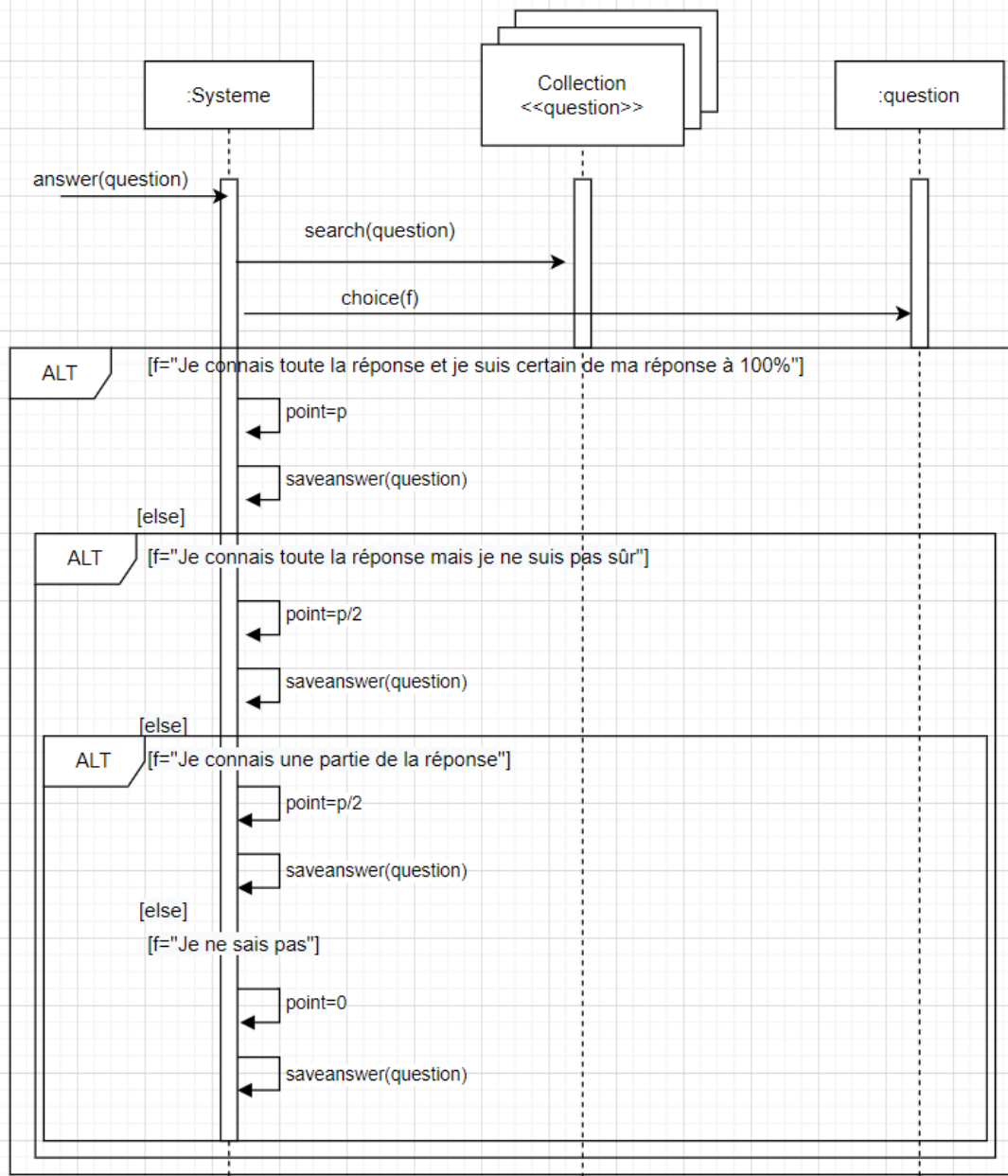
- oldSession(fichier)



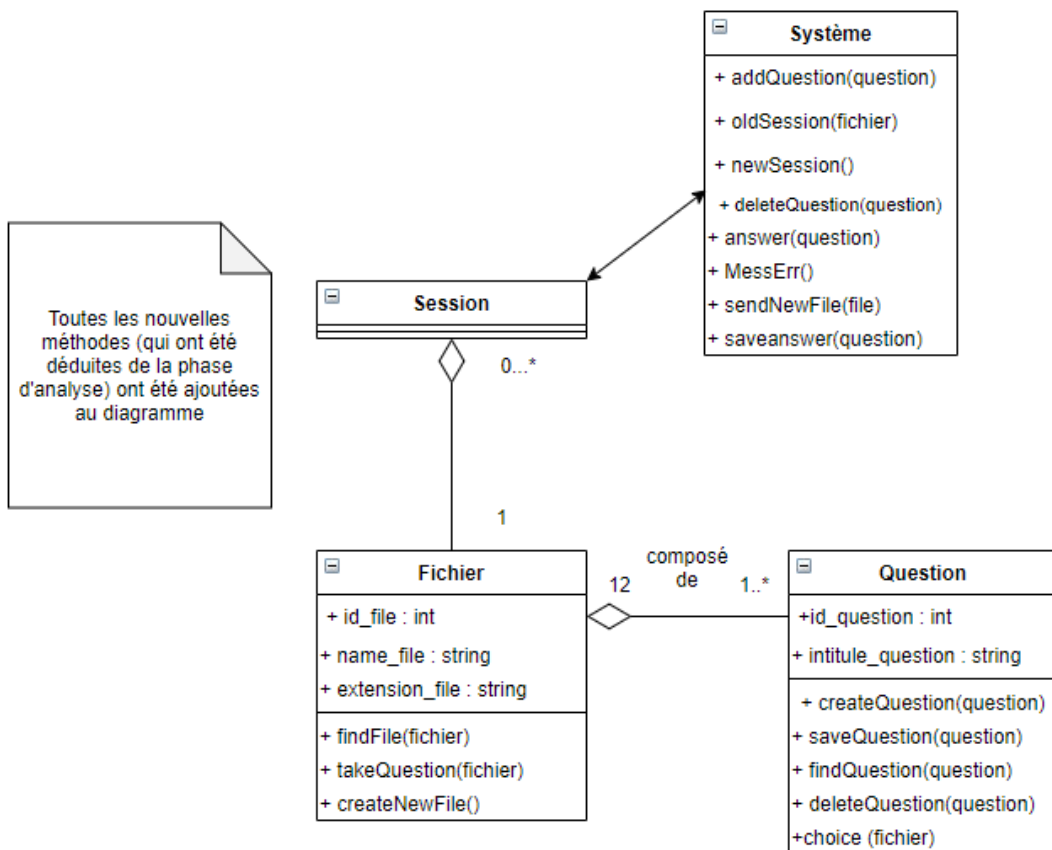
- newSession()



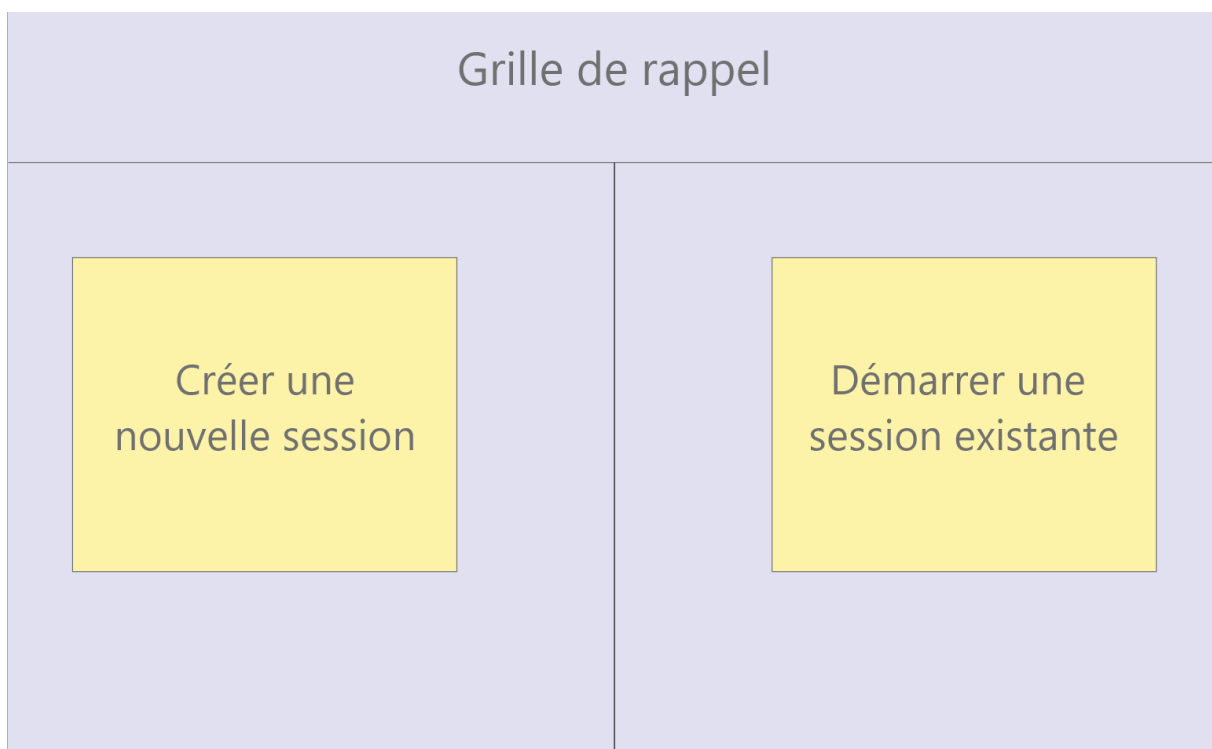
- answer(question)



### c) Diagramme d'analyse amélioré



### 3) Maquettage



Nom du fichier

Question ...

Entrer le nom du fichier existant



00:00

Tic Tac vous avez 5 minutes...

Points : 0

Question 1 Une question	Question 2 Une question	Question 3 Une question	Question 4 Une question
Question 5 Une question	Question 6 Une question	Question 7 Une question	Question 8 Une question
Question 9 Une question	Question 10 Une question	Question 11 Une question	Question 12 Une question

Votre réponse

☐ Je ne sais pas

☐ Je connais une partie de la réponse

☐ Je connais la réponse mais je ne suis pas sûr(e)

☐ Je connais toute la réponse et je suis certain(e) de ma réponse à 100%

Retour

Valider

Fin de la partie !

SCORE : 0

Je ne sais pas	Je connais une partie de la réponse	Je connais la réponse mais je ne suis pas sûr	Je connais la réponse et je suis certain à 100%

## 4) Conclusion

Le client ne souhaite pas de programmation web. Le choix le plus simple serait de programmer notre application en JAVA. Le C n'étant pas un langage de programmation très adapté pour réaliser du graphisme. De plus le Python n'est pas le langage de programmation le plus pratique pour nous car c'est un langage que nous n'avons pas pratiqué depuis quelques années.

## V) Partie conception détaillée

### 1) Introduction

Dans ce livrable nous allons désormais rentrer plus en profondeur dans le projet. Nous allons premièrement présenter les choix que nous avons réalisés à la suite du cahier des charges et de la partie analyse. Puis nous montrerons un peu de code que nous jugeons pertinent pour notre application. Enfin nous réaliserons un cahier de tests afin de montrer à notre client comment mettre en pratique certaines fonctionnalités de l'application.

### 2) Conception détaillée

La grille de rappel est programmée en JAVA pour des raisons de facilité que nous avons évoquée lors de la phase d'analyse.

Deux boutons sont présents sur l'écran d'accueil : un pour ouvrir un fichier existant, l'autre pour créer un fichier de questions.

Les questions sont enregistrées dans un fichier texte que l'application génère. Le joueur rentre ses questions une par une dans ce fichier. Elles sont enregistrées dans une ArrayList . Elles sont écrites dans le fichier à l'aide de la fonction Path file (voir partie codage).

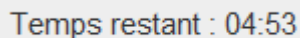
Si un fichier existe déjà, il est possible de le récupérer via une fonction permettant d'ouvrir le répertoire du périphérique sur lequel l'application est lancée.

La classe RandomAccessFile permettra de récupérer des lignes, de les lire et de les afficher à partir d'un fichier.

La résolution de l'application est de 1300/800.

Le timer se situe dans la partie supérieure droite de l'écran lorsque le jeu est lancé. Le temps s'écoule jusqu'à atteindre 00 : 00.

Il s'affiche de la façon suivante :



Temps restant : 04:53

Un score apparaîtra sur l'écran de jeu. Il s'incrémentera en fonction du nombre de points que rapporte la question et du choix de proposition.

L'évènement MouseListener permet d'ouvrir une petite fenêtre, par-dessus la fenêtre du jeu, qui contient toutes les propositions à une question (exemple : «Je ne sais pas », ...) lorsque l'utilisateur réalise un click gauche dans un rectangle qui contient une question. Les rectangles sont réalisés avec la classe « Graphics ».

Les réponses de l'utilisateur sont enregistrées dans le fichier texte correspondant à la réponse choisie. A la fin, le contenu de ces fichiers est affiché sur l'écran de fin de partie. Lorsque la partie sera terminée il suffira de supprimer ces fichiers, puis de les recréer avec les mêmes noms.

### **3) Un peu de code**

```

public void actionPerformed(ActionEvent ae) {
    String fileName = field.getText();
    String encoding = "UTF-8";
    try{
        PrintWriter writer = new PrintWriter(fileName, encoding);
        writer.close();
        JOptionPane.showMessageDialog(null,
            "Fichier créer !");
        valid = true;
    }
    catch (IOException e){
        System.out.println("An error occurred.");
        e.printStackTrace();
    }
}

```

La fonction New\_File permet de créer un fichier texte. Nous allons récupérer le texte du JTextField et à l'aide de la fonction PrintWriter un nouveau fichier vierge est créé.

```

try {
    Path file = Paths.get(fileName);
    Files.write(file, lines, StandardCharsets.UTF_8);
}
catch (IOException e){
    System.out.println("An error occurred.");
    e.printStackTrace();
}

```

La fonction Write\_File permet d'écrire dans le fichier. Nous allons récupérer le nom du fichier à l'aide de la classe Path. La fonction File.write, prend en paramètres le nom du fichier, la liste des questions et l'écriture en UTF8. Ainsi un fichier texte rempli des 12 questions en 12 lignes est créé.

```

public void Count() {

    timer = new Timer(1000, new ActionListener() {

        @Override
        public void actionPerformed (ActionEvent e) {

            seconde--;
            SSeconde = Format.format(seconde);
            SMinute = Format.format(minute);
            l.setText("Temps restant : " + SMinute + ":" + SSeconde);

            if (seconde==0) {
                seconde= 59;
                minute--;
                SSeconde = Format.format(seconde);
                SMinute = Format.format(minute);
                l.setText("Temps restant : " + SMinute + ":" + SSeconde);
            }
            if (minute==0 && seconde == 0) {
                timer.stop();
                System.exit(1);
            }
        }
    });
}

```

La fonction Count() est la fonction qui gère le timer. Elle s'occupe de faire les modifications nécessaires. Par exemple lorsque le timer arrive à 04 : 00, il doit faire passer le timer à 03 : 59 ! Pour le moment l'application se ferme lorsque le timer arrive à 00 : 00 (cela sera modifié plus tard). Les affichages se font dans une autre partie du code !

```

public class FileChooser1 {

    public static void main(String[] args) {

        JFileChooser jfc = new JFileChooser(FileSystemView.getFileSystemView().getHomeDirectory());

        int returnValue = jfc.showOpenDialog(null);

        if (returnValue == JFileChooser.APPROVE_OPTION) {
            File selectedFile = jfc.getSelectedFile();
            System.out.println(selectedFile.getAbsolutePath());
        }

    }

}

```

La fonction fileChooser1() est la fonction qui s'occupe d'obtenir le répertoire du périphérique sur lequel se trouve l'utilisateur actuellement. On peut alors afficher le chemin du fichier sélectionné.

## 4) Cahier de tests

Imaginons qu'un étudiant utilise notre application pour réviser un cours d'informatique. Il accède à la page d'accueil où deux choix s'offrent à lui :

- Créer une session
- Ouvrir une session

Il décide de créer une session. Une nouvelle page s'affiche et il rentre le nom du fichier texte qu'il veut créer et une fois cela fait il rentre les questions une par une en prenant soin de valider à chaque fois. Par exemple il rentre le nom « RevisionMO.txt » et rentre plusieurs questions comme « Qu'est ce qu'un diagramme d'activité ? », « Comment construire un diagramme d'activité » et ainsi de suite.

Retour au menu où il ouvre une session, il choisit le fichier qu'il vient de créer et la page de questions s'ouvre. 12 cases cliquables apparaissent en cliquant sur chacune des cases une petite fenêtre s'ouvre faisant apparaître les 4 choix de réponses de la question. Une fois les 5 minutes écoulées ou les 12 questions répondues, les révisions prennent fin et le score s'affiche ainsi que les questions rangées dans un tableau en fonction de la réponse de l'étudiant. Il a ensuite la possibilité de revenir au menu et de recommencer.

## **5) Conclusion**

A travers ce livrable le client peut avoir une idée beaucoup plus précise du projet. Les fonctionnalités ainsi que l'application sont décrites avec des détails « en profondeur ». On se penche désormais sur le cœur même du sujet. Ainsi, il est maintenant possible de se faire une idée de à quoi ressemblera l'application en général.

## **VI) BILAN**

Toutes ces parties constituent une première approche pour le client pour qu'il puisse avoir une première idée de l'application. Ce projet nous a permis de nous avoir une idée de notre projet et des outils que nous allons utiliser. Les débuts de code fonctionnelle nous ont permis de nous familiariser avec le JAVA et d'être en capacité de présenter un premier prototype contenant les premières fonctionnalités.

## **VII) Références diverses**

La méthode MoSCoW : [https://fr.wikipedia.org/wiki/Méthode\\_MoSCoW](https://fr.wikipedia.org/wiki/M%C3%A9thode_MoSCoW)

Le diagramme de Gantt : <https://www.gantt.com/fr/>

Jeretiens : <https://jeretiens.net/memorisez-grace-aux-tables-de-rappel/> (1)