Trivia Pursuit

*Rapport d’étude préliminaire*

Melissa Boucher et Charlie Laplante

25/02/2015

**Table des matières**

Introduction

Objectif du document 2

Portée du document 2

Références 2

Problématique 2

Description de contexte

Diagramme des cas d’utilisation 3

Description des cas d’utilisation 4

Cas 1 4

Cas 2 5

Cas 3 6

Cas 4 7

Diagramme UML 8

Diagramme de classes 9

Modèle relationnel 10

Conclusion

évaluer la faisabilité 11

Solutions d’implémentation 11

Échéancier d’implémentation 11

**Introduction**

## Objectif du document

Ce document servira à prouver que la demande du client a bien été comprise et ainsi démontrer l’analyse qui en a été fait. Une analyse de modélisation, des clarifications de l’énoncé et des besoins du client ainsi que des évaluations de cas et de solutions auront été fait avant l’écriture de ce document afin de permettre de comprendre le projet à qui le veux bien.

## Portée du document

Le document présent s’adresse à toute personne pouvant lire et comprendre des diagrammes de cas d’utilisations et d’UML. Le client pourra également y retrouver des explications résumés des fonctionnalités du système.

## Références

* L’énoncé remis par le professeur
* La documentation du cours fait par le professeur en ligne
* La documentation autre trouvée sur internet
* Les explications du professeur

## Problématique

Le client désire un jeu développé nommé Trivia Pursuit. Il s’agit d’un jeu questionnaire pouvant être joué de 2 à 4 joueurs. 6 catégories de questions sont disponibles pour divertir les joueurs. Ce jeu doit être composé d’une base de données ainsi que d’un programme fait à partir du langage Ado.Net.

**Description de contexte**

## Diagramme de cas d’utilisation

Description des cas d’utilisation

**ID :** Case 1

**Titre :** Commencer une partie

**Brève description :** Des utilisateurs créent une partie de Trivia Pursuit.

**Acteurs :** Les joueurs

**Préconditions :**

* Il y a minimum 2 joueurs présents
* Il y a maximum 4 joueurs présents
* Les joueurs doivent savoir jouer au jeu

**Post conditions :**

* Les joueurs jouent au jeu

**Enchaînement des opérations**

|  |  |
| --- | --- |
| **Acteurs** | **Système** |
| 1. Le cas d’utilisation débute lorsque des joueurs démarrent le programme. | 1. Le système demande un nombre de joueurs valide (2 à 4). |
| 1. Les joueurs entrent leur nom de joueur. | 1. Le système crée ou cherche l’identité des joueurs dans la BD. |
|  | 1. Le système détermine l’ordre des joueurs. |
|  | 1. Le système démarre la partie. |

**Scénarios alternatifs :**

**A1 :** Un joueur décide de quitter ou d’abandonner pendant la création de partie.

* Une nouvelle partie doit être créée

**A2 :** Il n’y a pas assez de joueurs ou trop de joueurs.

* Les utilisateurs doivent par eux même décider quoi faire

**ID :** Case 3

**Titre :** Vérification de fin de partie

**Brève description :** On veut vérifier si un joueur remplit les conditions de gagnant qui mettraient fin à la partie.

**Acteurs :** Un joueur

**Préconditions :**

* Une partie doit avoir été débutée
* Une réponse doit avoir été répondue correctement

**Post conditions :**

* Les joueurs arrêtent de jouer
* Les joueurs continuent de jouer

**Enchaînement des opérations**

|  |  |
| --- | --- |
| **Acteurs** | **Système** |
| 1. Un joueur répond à une question correctement. | 1. Le système met la question répondu dans la table Réussi avec l’ID du joueur et de la question. |
|  | 1. Le système compte le nombre de questions réussies. |

**Scénarios alternatifs :**

**A1 :** Il y a 30 questions réussites dans la table Réussi pour un même joueur.

* La partie se termine

**A2 :** Il n’y a pas 30 questions réussites dans la table Réussi pour un même joueur.

* La partie continue

Diagramme UML

Diagramme de classes

Modèle relationnel

**Conclusion**

Évaluation de la faisabilité

Solutions d’implémentation

Échéancier d’implémentation