Trivia Diamond

*Rapport d’étude préliminaire*

Melissa Boucher et Charlie Laplante

27/02/2015

**Table des matières**

Introduction

Objectif du document 2

Portée du document 2

Références 2

Problématique 2

Définitions 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 de classes UML 8

Diagramme de package 8

Conclusion

évaluer la faisabilité 9

Droits d’auteur et de reproduction 9

Solutions d’implémentation 10

Échéancier d’implémentation 10

Annexes

Modèle Relationnel 11

Modèle Conceptuel 12

Modèle Entité relation 13

Prototype de l’application 14

**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, soit les programmeurs qui devront écrire le programme. Le client (Saliha Yacoub) pourra également y retrouver des explications résumés des fonctionnalités du système pour vérifier si nous avons compris l’énoncé.

## Références

* L’énoncé remis par le professeur
* Période de questions répondu par le professeur
  + Est-ce que les joueurs peuvent abandonner n’importe quand?
  + Clarifier la situation de gagner ? (Nombre de question à répondre par catégorie)
  + Est-ce qu’on doit gérer plusieurs parties ?
  + Doit-on avoir une table score ou pouvons-nous gérer le nombre de bonne réponse avec des counts ?

## Problématique

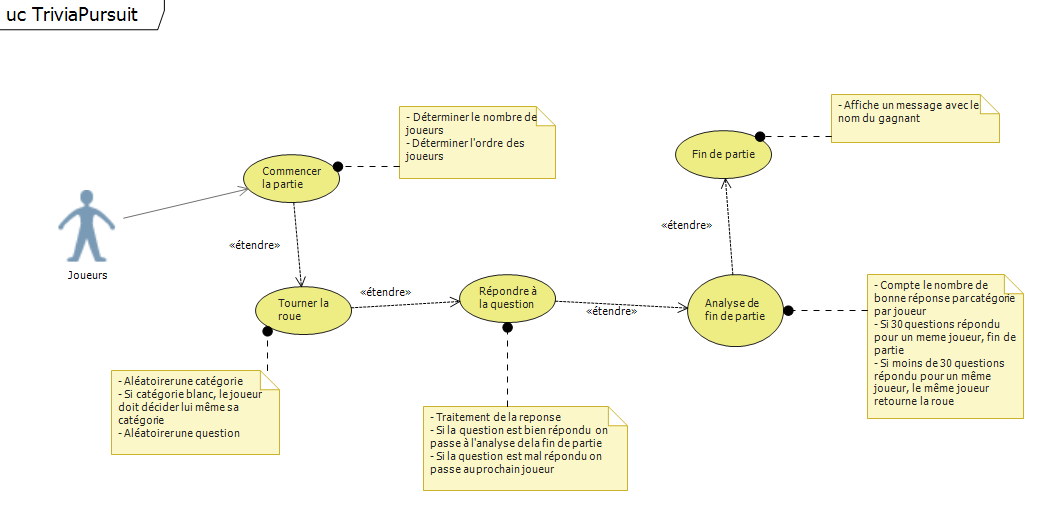
Le client désire un jeu développé nommé Trivia Diamond. Il s’agit d’un jeu questionnaire pouvant être joué de 2 à 4 joueurs. 6 catégories de questions sont disponibles pour divertir les joueurs. Chacun leur tour, les joueurs devront répondre à des questions. Le jeu se termine lorsqu’un joueur réussi à répondre correctement à 5 questions de chaque catégorie. Ce jeu doit être composé d’une base de données Oracle ainsi que d’un programme fait à partir du langage Ado.Net.

***Définitions***

* **ADO.NET** : Bibliothèque logicielle de Microsoft fournissant une interface d’accès aux données dans l’environnement Windows.
* **Implémentation** : Mise en place de fonctionnalité informatique.
* **SQL :** *Structured Query Language* est langage informatique normalisé servant à exploiter des bases de données

**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 Trivial Diamond.

**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 2

**Titre :** Tourner la roulette

**Brève description :** Une catégorie aléatoire est choisis par le programme si la catégorie choisis est blanche demande à l’utilisateur quel catégorie il veut. Ensuite le programme pose une question aléatoire au joueur.

**Acteurs :** Roulette.

**Préconditions :**

* Déterminer le nombre de joueur.
* Déterminer l’ordre des joueurs.

**Post conditions :**

* La question ainsi que sa réponse est gardé en mémoire.

**Enchaînement des opérations**

|  |  |
| --- | --- |
| **Acteurs** | **Système** |
| 1. Le cas début lorsque c’est au tour du joueur à jouer. |  |
| 1. La roulette se met à tourner |  |
|  | 1. Le programme calcule une valeur aléatoire |
| 1. La roulette s’arrête sur une position aléatoire déterminer par le système. |  |

**Scénarios alternatifs :**

**A1 :** La roulette tombe sur la couleur blanche(Système)

- Le joueur choisit sa catégorie. (Acteur)

**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

**ID :** Case 4

**Titre :** Répondre à une question

**Brève description : Un joueur répond à une question.**

**Acteurs :** Le joueur

**Préconditions :**

* La roulette doit être tournée.
* Les joueurs doivent être encore dans la partie.
* Une catégorie doit être choisie.

**Post conditions :**

* Si Bonne réponse on vérifie s’il y a un gagnant.

**Enchaînement des opérations**

|  |  |
| --- | --- |
| **Acteurs** | **Système** |
|  | 1. Le système affiche la question et les choix. |
| 1. Le joueur choisit une réponse. | 1. Le système traite la réponse. |
|  | 1. Le système affiche le résultat (Bonne ou mauvaise réponse) |

**Scénarios alternatifs :**

**A1 :** Le joueur ne choisit pas de réponse

-La réponse est comptée comme étant mauvaise.

-On passe au prochain joueur.

Diagramme de classes UML

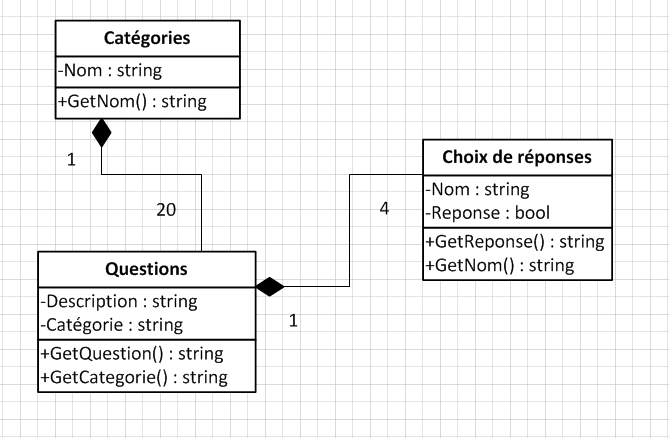
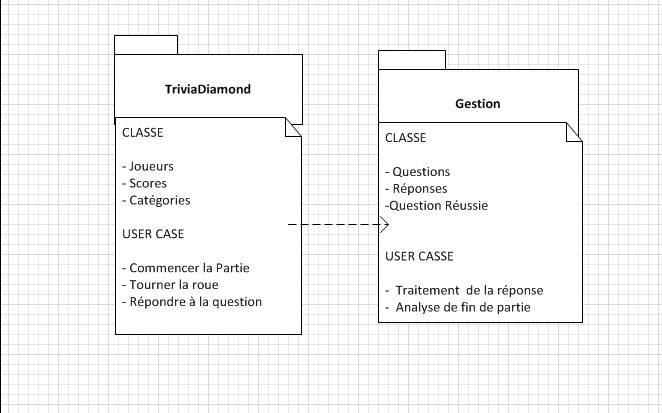
******

Diagramme de packages

**

**Conclusion**

Évaluation de la faisabilité

Le jeu TriviaDiamond est un projet réalisable dans le temps données puisqu’il englobe des connaissances informatiques auquel l’équipe de production (Melissa et Charlie) sont confortable avec. De plus, puisqu’il existe déjà des logiciel similaire sur le marcher nous avons accès à certain information pour nous inspirer et nous guider dans notre création.

Dans le même courant de penser, l’équipe à accès à des outils de programmation à cout presque inexistant. Cet aspect du travail couvert, il est encore plus facile de se concentrer sur la conception du travail.

D’autre part, étant donné que la planification n’est pas compter dans le temps de conception, il ne reste qu’à calculer le temps prévus pour les différentes étapes création ce qui simplifie l’ampleur de la tache globale.

Cependant, il faut prévoir les problèmes que peuvent rencontrer l’équipe en lien avec la connexion réseau nécessaire pour avoir accès à la base de données et à l’application de service web d’hébergement et de gestion de développement de logiciels.

Droits d’auteur et de reproduction

Tous droits réservés. Aucun extrait de ce document ne peut être reproduit, sous quelque forme ou procédé quel qu’il soit.

Solution d’implémentation1

Description de la solution

Pour notre solution numéro un, nous allons commencer par implémenter notre jeu afin qu’il fonction pour un seul joueur. Cette solution nous permettra de faciliter la construction du jeu tout en n’empêchant pas l’ajout de la fonctionnalité multijoueur plus tard. Puisque la plupart des options du jeu peuvent être appliquées qu’à un seul utilisateur, cette solution nous permettra entre autres d’économiser du temps et d’avoir quelque chose à remettre au client en cas de problème avec l’intégration du multijoueur.

Échéancier d’implémentation

|  |  |
| --- | --- |
| **Description des étapes** | **Temps requis** |
| BD | 8 heures |
| Création des tables | 1 heure |
| Insertion des données | 2 heures |
| Création des packages, procédures et fonctions | 5 heures |
| Ado.net et C# | 25 heures |
| Fonctionnement du logiciel | 2.5 heures |
| Intégration des commandes sql | 20 heures |
| Réglage final des bugs / debugge | 2.5 heures |

Solution d’implémentation 2

Description de la solution

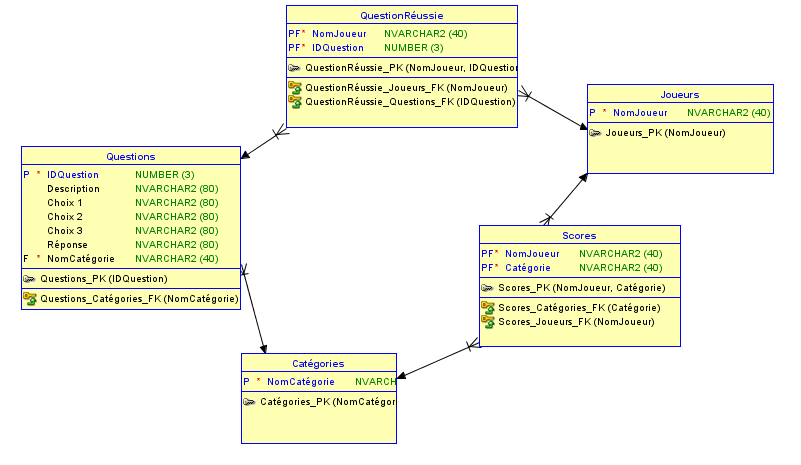
La solution deux consiste en la suite de la solution un. Elle englobe l’ajout de la fonctionnalité multijoueur dans notre jeu et de la peaufinassions de l’interface graphique remise au client. Celle-ci est considérée dans la planification normale comme étant de base, mais peut être abandonnée en cas de manque de temps dû à des raisons «X».

Échéancier d’implémentation

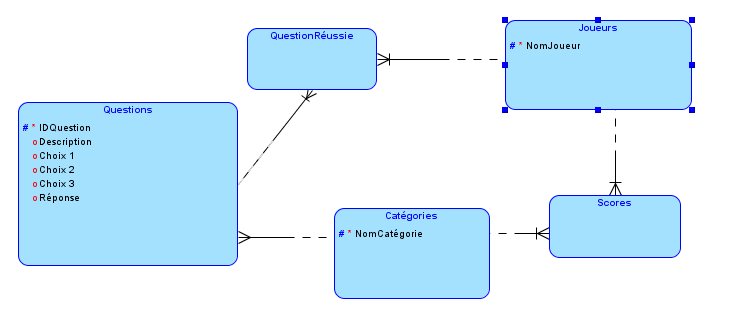
|  |  |
| --- | --- |
| **Description des étapes** | **Temps requis** |
| BD | 8 heures |
| Création des tables | 1 heure |
| Insertion des données | 2 heures |
| Création des packages, procédures et fonctions | 5 heures |
| Interface | 4 heures |
| Développement | 3.5 heures |
| Vérification des entrées au clavier | 0.5 heure |
| Ado.net et C# | 30 heures |
| Fonctionnement du logiciel | 2.5 heures |
| Intégration du multijoueur | 5 heures |
| Intégration des commandes sql | 20 heures |
| Règlage final des bugs / débugage | 2.5 heures |

**Annexes**

Annexe 1 – Modèle Relationnel

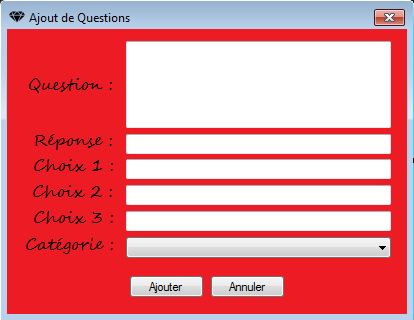
******

Annexe 2 – Modèle Conceptuel

******

Annexe 4 – Modèle Entité Relation

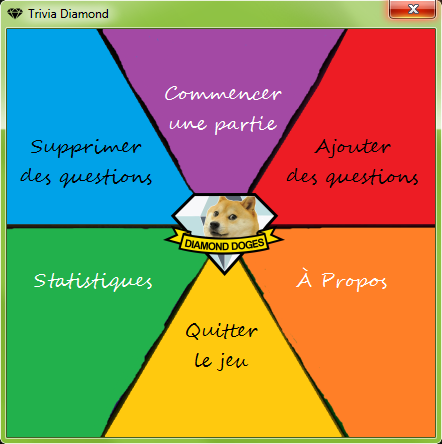
Annexe 3 – Prototype de l’application

******

Windows Form pour ajouter une question dans la base de données selon une catégorie choisie.



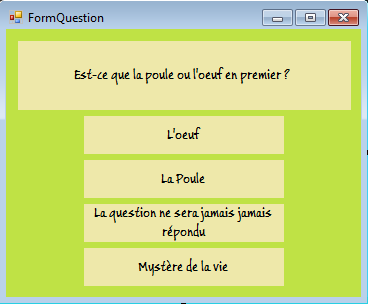
Windows Form pour afficher les informations générales de l’application. Communément appelé À Propos.



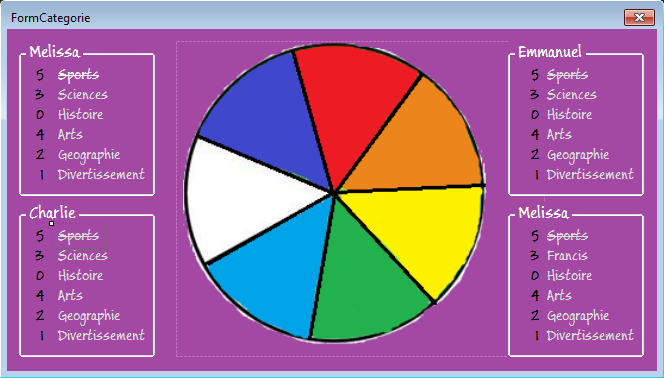
Windows Form du menu de l’application qui va être composé de 6 boutons «Flash» qui permettent d’accéder a chaque partie de l’application.

******

Windows Form pour supprimer une question dans la base de données selon une catégorie choisie.

******

Windows Form qui affiche la question à répondre à l’utilisateur. Les choix de réponses sont des boutons «Flash»

******

Windows Form du jeu avec affichage du nombre de questions répondues et des catégories terminées.