HELHa
Haute École Louvain en Hainaut

What do you know about it ?

Giorgio Caculli LA196672, Guillaume Lambert LA198116, Tanguy Taminiau LA199566
Groupe B05

29 avril 2021

ÉCONOMIQUE

Table des matières

Introduction

Dans le cadre du cours de “Projets” de l’UE 210, nous allons devoir créer un jeu similaire à “Tu te mets combien?” (TTMC). Il s’agit d’un jeu comportant des cartes comportant des questions. Il existe quatre thèmes repérable par leurs couleurs respectives, la couleur :

- mauve est attribué aux cartes ayant pour thème “improbable”
- orange pour “plaisir”
- bleu pour “informatique”
- vert pour “scolaire”

Chaque carte possède un thème, un sujet en rapport avec le thème, et quatre questions.
Les question sont numéroté de un à quatre et triées par ordre croissant de difficulté.
Le jeu commence avec un joueur qui tire une carte, il pose la question "tu te mets combien en (sujet de la carte) ?" à l’autre joueur, l’autre joueur lui répond un chiffre entre un et quatre, le joueur ayant tiré la carte pose alors la question correspondante au nombre donné par l’autre joueur. En cas de bonne réponse, le joueur ayant répondu gagne un nombre de points équivalent au numéro de la question. En cas de mauvaise réponse, le joueur ne gagne aucun point. C’est ensuite au joueurs ayant répondu de tirer une carte et d’interroger l’autre.

1 Product backlog

US-01	En tant qu'utilisateur je voudrais savoir mon score.
US-02	En tant qu'utilisateur je voudrais savoir si j'ai bien répondu.
US-03	En tant qu'utilisateur je voudrais savoir si j'ai mal répondu.
US-04	En tant qu'utilisateur je voudrais savoir quelle était la bonne réponse.
US-05	En tant qu'utilisateur je voudrais savoir mettre mon jeu sur pause.
US-06	En tant qu'utilisateur je voudrais savoir reprendre mon jeu où je l'avait laissé.
US-07	En tant qu'utilisateur je voudrais savoir arrêter mon jeu à tout moment.
US-08	En tant qu'utilisateur j'aimerais joué en multi joueur localement.
US-09	En tant qu'administrateur je dois pouvoir ajouter une nouvelle carte au deck.
US-10	En tant qu'administrateur je veux pouvoir supprimer une carte du deck.
US-11	En tant qu'administrateur je veux pouvoir modifier une carte existante.
US-12	En tant qu'utilisateur j'aimerais avoir une musique de fond.
US-13	En tant qu'utilisateur j'aimerais pouvoir gérer le volume de la musique.
US-14	En tant qu'utilisateur j'aimerais pouvoirs activer ou désactiver la musique de fond.
US-15	En tant qu'utilisateur je voudrais pouvoir choisir mon propre pséudonyme.
US-16	Création d'un plateau de jeu.
US-17	Implémentation du plateau de jeu.
US-18	Creation des pions.
US-19	Implémentation des pions de jeu et animation de mouvements des pions.
US-20	Investiguer sur le multi joueur en ligne.
US-21	Pion personnalisé.
US-22	En tant que joueur, je voudrais communiquer avec d'autre joueurs.
US-23	En tant que joueur, j'aimerais jouer avec d'autre joueurs en ligne.
US-24	En tant que joueur, j'aimerais rejoindre une partie en ligne.
US-25	En tant que joueur, j'aimerais héberger une partie en ligne.

2 Tableau Trello

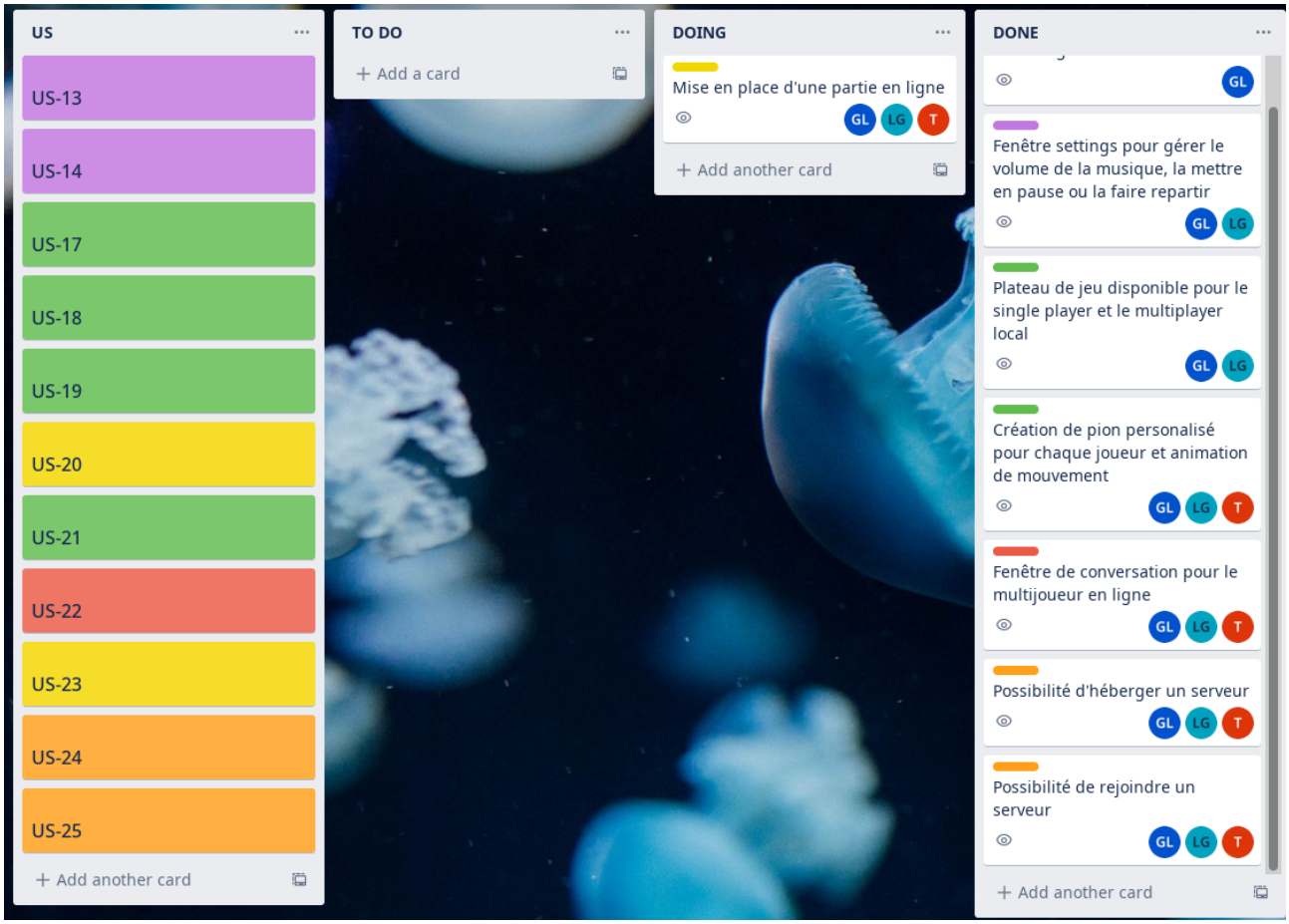

Figure 1 – Tableau Trello Sprint 3

3 Diagramme de classe

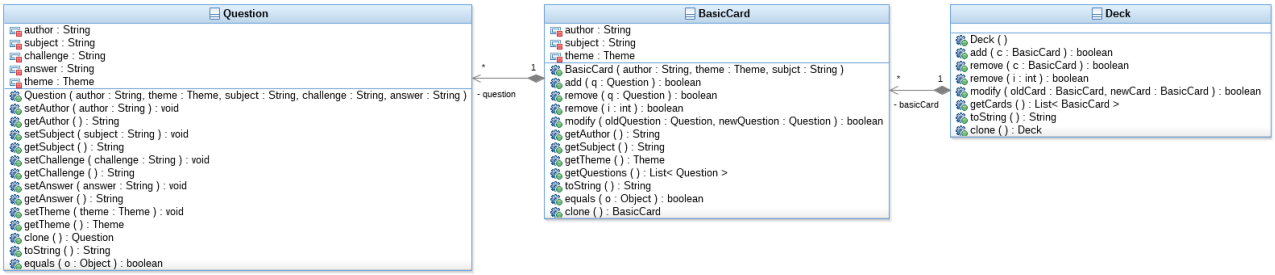

FIGURE 2 – Diagramme de classe du modèle

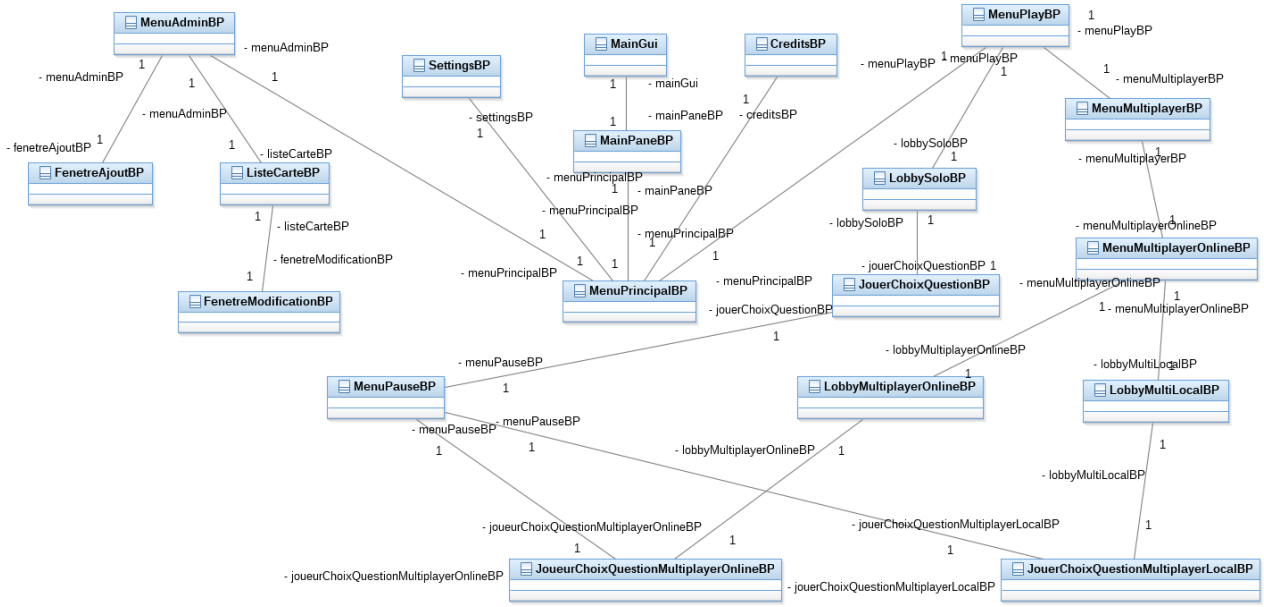

FIGURE 3 – Diagramme de classe de la vue

Nous avons décidé de travailler avec le MainGui en position principale. Lorsque nous changeons de menu via le click d'un bouton, nous modifions la scène via différents StackPane inclus dans les différentes classes. Ce qui nous permet d'avoir une répartition plus flexible des menus ainsi qu'une plus grande liberté lors des instantiations de ceux-ci. Grâce aux StackPanes, on a su faire plusieurs réinstantiations d'un même menu, ce qui nous a permis de démarrer des nouvelles parties à chaque instance d'une partie solo ou multi joueur.

4 Code le plus représentatif

Code les plus représentatifs du Sprint 3

```
1  package be.helha.ttmc.ui.gui.play;
2
3  import java.util.ArrayList;
4  import java.util.List;
5  import java.util.Random;
6
7  import be.helha.ttmc.model.Deck;
8  import be.helha.ttmc.ui.Settings;
9  import javafx.scene.control.Label;
10 import javafx.scene.layout.AnchorPane;
11 import javafx.scene.layout.BorderPane;
12 import javafx.scene.paint.Color;
13 import javafx.scene.shape.Rectangle;
14
15 public class PlateauBP extends BorderPane
16 {
17     private int nbCases;
18     private List< Rectangle > cases;
19     private List< Label > num;
20     private PionCircle pions[];
21
22     private final double WIDTH_RECT = 35.;
23     private final double HEIGHT_RECT = 30.;
24     private final double MOUV_X = WIDTH_RECT + 1;
25     private final double MOUV_Y = HEIGHT_RECT + 1;
26
27     private Settings s;
28
29     public PlateauBP( Deck d, int nbJoueurs, Settings s )
30     {
31         this.s = s;
32         nbCases = d.getCards().size();
33         AnchorPane anch = new AnchorPane();
34         getCases();
35         getNum();
36         int test = 0;
37         int rangY = 0;
38         String text;
39         for ( int i = 0; i < cases.size(); i++ )
40         {
41             double mouvX = ( i + 1 ) * MOUV_X;
42             text = ( i + 1 ) + "";
43             Label lab = new Label( text );
44             if ( ( i + 1 ) * MOUV_X > s.getWidth() - WIDTH_RECT * 3 )
45             {
46                 if ( ( int ) ( ( ( i + 1 ) * MOUV_X ) / ( s.getWidth() - WIDTH_RECT * 3 ) ) > rangY )
47                 {
48                     rangY++;
49                     test = 0;
50                 }
51
52                 mouvX = ( test + 1 ) * MOUV_X;
53                 test++;
54             }
55             double mouvY = rangY * MOUV_Y;
56             getCases().get( i ).setY( mouvY );
57             lab.setTranslateY( mouvY );
58
59             if ( i == d.getCards().size() - 1 )
60                 lab.setText( text + "\nFinish" );
61             lab.setLayoutX( mouvX );
62             getNum().add( lab );
63             getCases().get( i ).setX( mouvX );
64             anch.getChildren().addAll( getCases().get( i ), getNum().get( i ) );
65         }
66         pions = new PionCircle[ nbJoueurs ];
67         for ( int i = 0; i < nbJoueurs; i++ )
68         {
69             Random randColorR = new Random();
70             Random randColorG = new Random();
71             Random randColorB = new Random();
72             pions[ i ] = new PionCircle( getCases().get( 0 ).getX() + WIDTH_RECT / 2,
73                     getCases().get( 0 ).getY() + HEIGHT_RECT / 2, WIDTH_RECT / 3,
74                     Color.rgb( randColorR.nextInt( 255 ), randColorG.nextInt( 255 ), randColorB.nextInt( 255
       ) ) );
75             anch.getChildren().add( pions[ i ] );
76         }
77         setCenter( anch );
78     }
```

```
79
80      public double getHEIGHT_RECT()
81      {
82          return HEIGHT_RECT;
83      }
84
85      protected List< Rectangle > getCases()
86      {
87          if ( cases == null )
88          {
89              cases = new ArrayList<>();
90              for ( int i = 0; i < nbCases; i++ )
91              {
92                  Rectangle rect = new Rectangle( WIDTH_RECT, HEIGHT_RECT );
93                  rect.setStroke( Color.BLUE );
94                  rect.setFill( Color.ALICEBLUE );
95                  cases.add( rect );
96              }
97          }
98          return cases;
99      }
100
101     public double getWIDTH_RECT()
102     {
103         return WIDTH_RECT;
104     }
105
106     private List< Label > getNum()
107     {
108         if ( num == null )
109             num = new ArrayList<>();
110         return num;
111     }
112
113     public PionCircle getPion( int id )
114     {
115         return pions[ id ];
116     }
117 }
```

5 Sprint review

Durant ce troisième et dernier sprint, quelques nouveautés ont fait leurs apparitions.
Un plateau de jeu a été ajouté en solo ainsi qu'en multi local. Le multi joueur local accepte plus de deux joueurs (contrairement à avant où deux était le maximum).

- Objectif atteint
  - Connaitre son score.
  - Savoir si la réponse donnée était la bonne.
  - Savoir si la réponse donnée était mauvaise.
  - Savoir la bonne réponse si la réponse donnée était mauvaise.
  - Possibilité de mettre le jeu en pause.
  - Reprendre mon jeu là où je l'avais laissé.
  - Possibilité de quitter mon jeu à tout môment.
  - Présence d'un multi-joueur local.
  - Possibilité d'ajouter une nouvelle carte.
  - Possibilité de retirer une carte.
  - Possibilité de modifier une carte.
  - Présence d'une musique de fond.
  - Utilisation de pseudo créé par l'utilisateur.
  - Création et utilisation d'un plateau de jeu.
  - Création de pion de couleur différente pour chaque joueur.
  - Création d'un chat en multi joueur en ligne.
  - Possibilité de régler le volume sonore.
  - Possibilité de désactiver la musique de fond.
  - Création d'un launcher permettant de télécharger les librairies nécessaires au fonctionnement du jeu. Il permet aussi de lancer le jeu.
- Objectif non atteint
  - Jeu en multi joueur en ligne opérationnel.

6 Sprint retrospective

En ce qui concerne le possitif de ce sprint,
- Les différentes tâches du projet ont été bien distribuée entre tous les membres de l'équipe.
- L'entraide entre tous les membres de l'équipe.

En ce qui concerne le négatif de ce sprint,
- Etant donnés le temps court entre les 2 sprint, nous avons été surchargés, ce qui a donné une démotivation à l'équipe.
- Le manque de documentation de la programmation réseau à rendue ardue le développement du jeu en ligne, ce qui nous a frustré.

En ce qui concerne les améliorations pouvant être faite,
- recherche approfondie sur de la documentation anglaise pour la programmation réseau.

7 Design pattern

Afin d'assurer un certain niveau de securité lors des manipulations avec les cartes du deck, on a envisagé la possibilité de mettre en place un Proxy. Ceci, permettrai d'intéragir avec les carte à travers une interface qui aurait le seul et unique droit d'accéder à toutes les cartes présentes dans le Deck. Ce serait la seule qui aurait le droit d'écrire dans le deck.

ÉCONOMIQUE