“Project Stream”

Ce document résulte d’un travail personnel ayant pour but d’explorer un domaine de la programmation afin de se spécialiser dans un secteur particulier. Des heures de cours étaient dédiées à ce travail sur une période de 2 mois.

Mon projet: «Project Stream»

L’objectif du projet était de développer sur Unity un jeu de carte tel Hearthstone, tout en utilisant une base de données d’utilisateurs déjà existante et appartenant au site learn2play.fr, qui est un site permettant à des personnes de trouver des coéquipiers sur des jeux tel que League Of Legends.
Ma partie au sein du projet était de développer la connexion à son compte, ainsi que de gérer la collection de cartes des utilisateurs.
La réalisation de ce projet nécessitait donc d’avoir des connaissances en base de données, en REST API, en programmation orientée objet et en sérialisation de données.
Il était également nécessaire d’avoir un projet bien architecturé et un code optimisé puisqu’il s’agit essentiellement de manipulation de données et de requêtes web.

Le jeu:

Chaque carte est représentait par un streamer «Twitch.tv» et par un champion du jeu League Of Legends. Elles possèdent toutes des talents, des rôles, et des caractéristiques différentes.

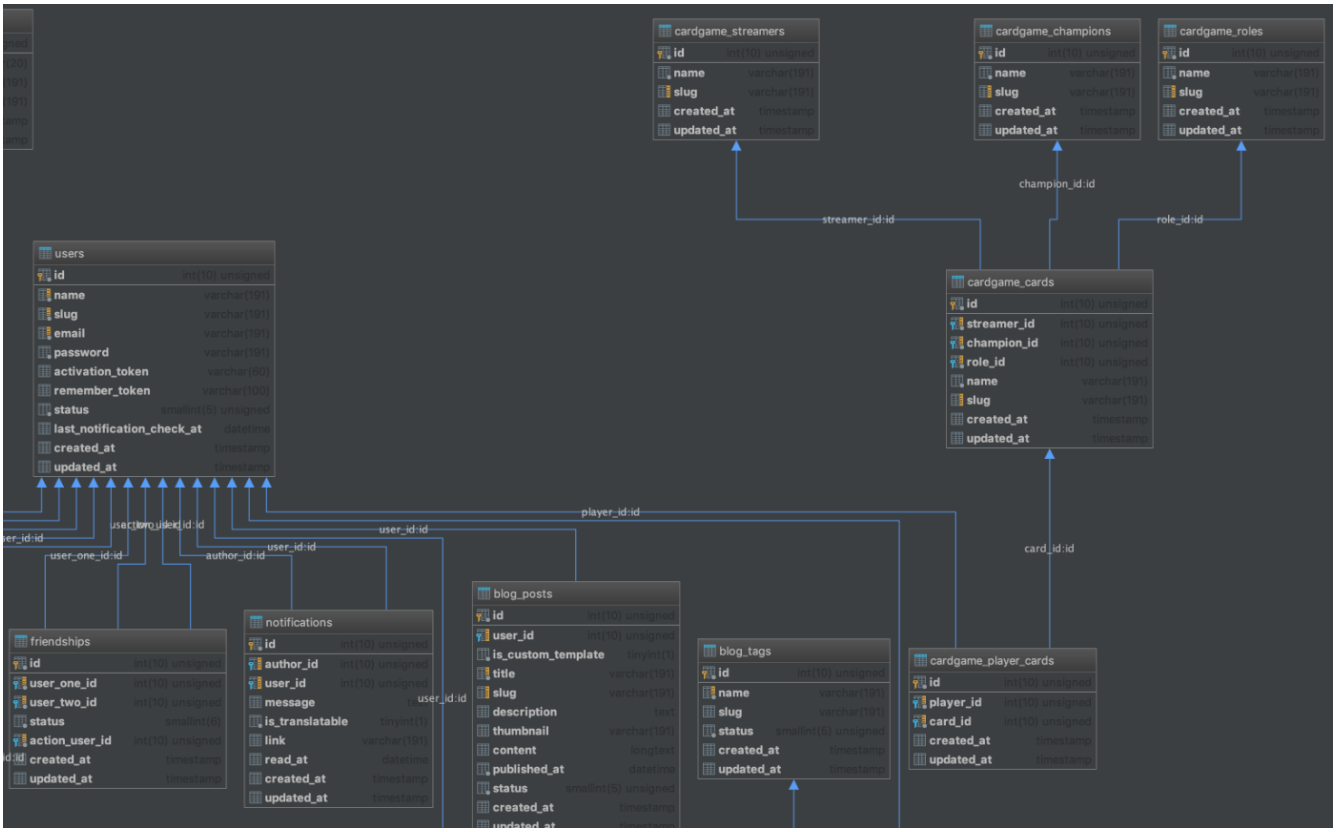
Pourquoi ce projet?

Je souhaiterais travailler dans le domaine du serious gaming, ou dans le développement d'applications 3D à but marketing, éducatif, informatif…
La base de données est une partie qui reviens souvent dans ce genre d'application puisque par exemple, les données des clients sont souvent stockées par les entreprises travaillant dans ce domaine.
Je suis intéressé par la manipulation de données, ayant apprécié travailler sur de la base de données lors de mes deux derniers stages.
J'avais également l'envie de découvrir des aspects du développement web et des API.
Ce projet avait aussi un gros potentiel puisque j'ai été à la base contacté par un streamer Twitch français, «Narkuss», qui avait l'idée de développer ce jeu de cartes. Sa portée médiatique pourrait permettre à ce jeu d'attirer des joueurs à l'essayer.

Comment ai-je abordé ce projet?

Dans un premier temps, nous avons réfléchis au concept du jeu, ses règles, les effets que les cartes pourraient avoir, pour ensuite architecturer la base de données et l'implémenter.
J'ai travaillé avec un développeur web afin d'élaborer cette base de données, puis j'ai ensuite pu développer l'application Unity afin de la lier aux utilisateurs de la base avec leurs collections de cartes.

La base de données correspondante au jeu de carte:



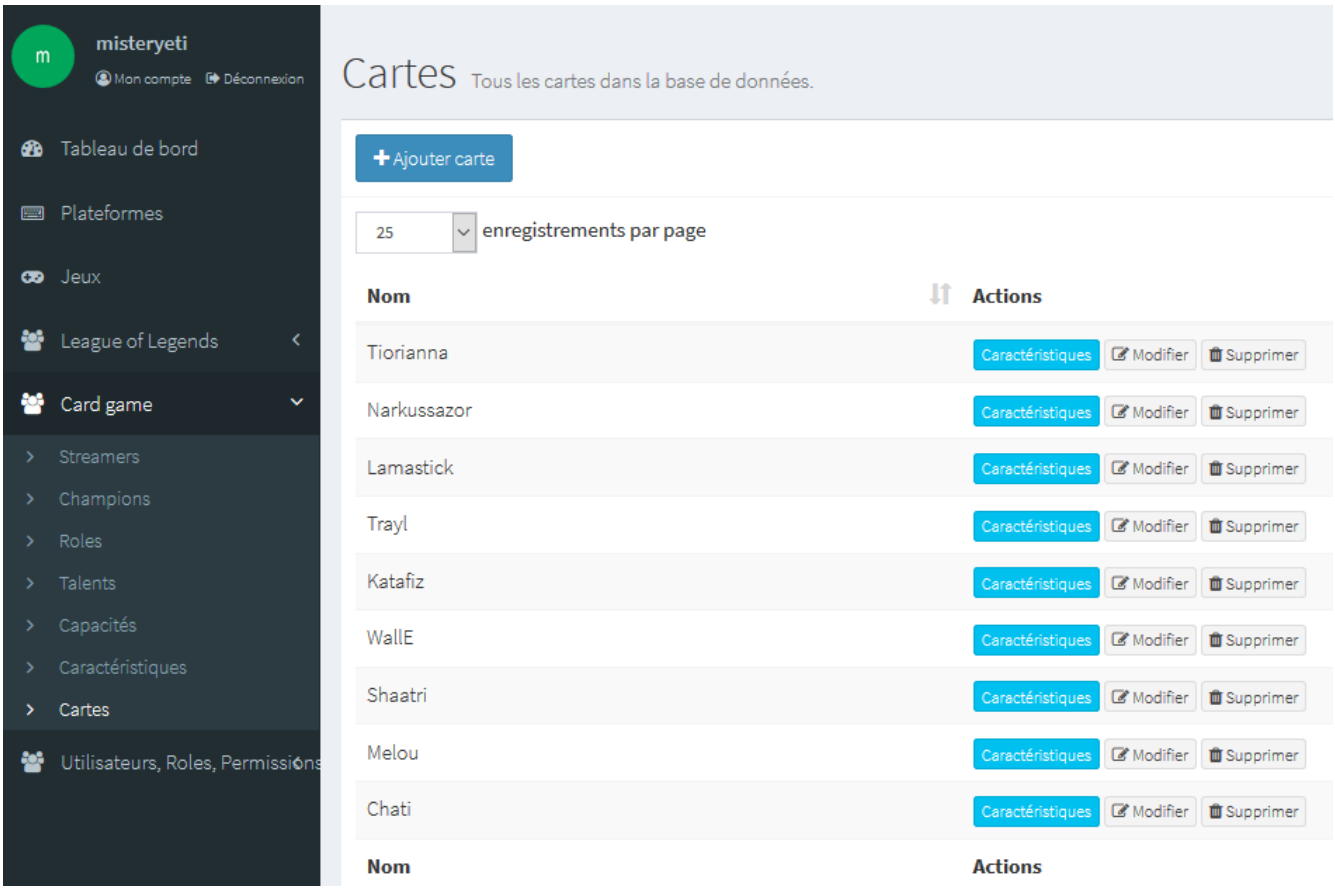
Les routes actuelles de l'API:

```php
// Card game routes

    $router->group([
        'prefix'    => 'cardgame',
        'namespace' => 'CardGame',
    ], function () use ($router) {
        $router->get('card/{page?}/{limit?}', 'CardController@index');
        $router->post('register-card/{card}/{player}', 'CardController@register');
        $router->get('player/{player}/{page?}/{limit?}', 'PlayerController@index');

        $router->get('streamer', 'DataController@streamer');
        $router->get('champion', 'DataController@champion');
        $router->get('role', 'DataController@role');
        $router->get('talent', 'DataController@talent');
        $router->get('ability', 'DataController@ability');
        $router->get('caracteristic', 'DataController@caracteristic');
    });

    (new GameRoute())->gameApiRoutes();
});
```

Exemple du code php permettant d'ajouter une carte à un joueur:

```php
public function register(Card $card, Player $player): Response
    {
        $card->players()->attach($player->id);

        return $this->player($player);
    }
```

Le développeur web avec lequel j'ai travaillé à également créer un dash board permettant de créer facilement des cartes et ses caractéristiques correspondantes, comme les streamers, les champions de League Of Legends, etc...



Les requêtes GET & POST sur Unity:

Problèmes:

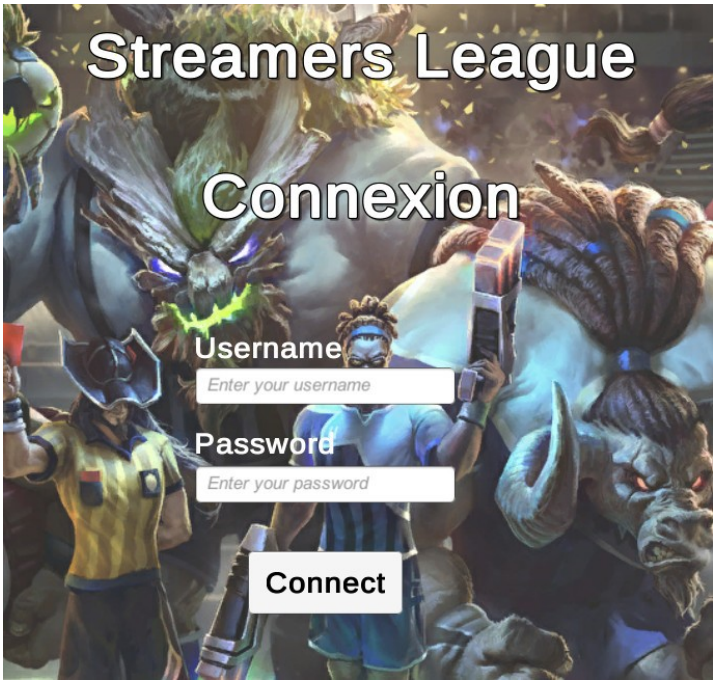
- Comment exécuter des requêtes GET & POST sur Unity
- Comment les rendre génériques ?
- Comment retourner des données tout en gardant une certaine généricité.
- Comment exécuter des méthodes :
  avant/après l'exécution de la requêtes web
  dans le cas d'un succès/échec de la requête web

Mes solutions:

- Utiliser des delegates
- Utiliser des coroutines
- Utiliser des objets génériques.
- Convertir des données JSON en objets
- Exécuter des méthodes en fonction du résultat de la requête web

```
//Get generic method
   public static IEnumerator RequestCoroutineGet<T>(string url, Action<T> getObject = null, Action actionFail =
null, Action actionBefore = null, Action actionAfter = null)
   {

      if (actionBefore != null)
         actionBefore();

      UnityWebRequest www = UnityWebRequest.Get(url);
      yield return www.SendWebRequest();

      if (www.isNetworkError || www.isHttpError)
      {
         actionFail();
      }
      else
      {
         string jsonResult =
                  System.Text.Encoding.UTF8.
                  GetString(www.downloadHandler.data);
         T obj = JsonHelper.GetJsonObject<T>(jsonResult);
         getObject(obj);

         if (actionAfter != null)
            actionAfter();
      }

   }
```

Exemple, l'authentification au jeu:



Que veux-t-on qu'il se passe lorsque le joueur clique sur «Connect» ?

1) Désactiver le bouton et faire apparaître un logo de chargement.
2) Exécuter la requêtes :
   - Si la requête renvoie un code d'erreur → Ne pas connecter l'utilisateur, ré-activer le bouton et retirer le logo de chargement.
   - Si la requêtes s'exécute correctement → Charger la collection de cartes, puis connecter l'utilisateur, et retirer le logo de chargement

Chaque partie doit être traitée séparément → Une seule tâche à la fois

```
public void Authentication()
{
    StartCoroutine(AuthenticateCoroutine());
}

IEnumerator AuthenticateCoroutine()
{
    WWWForm form = new WWWForm();
    form.AddField("identifier", inputFieldUserName.text);
    form.AddField("password", inputFieldPassword.text);

    yield return Request.RequestCoroutinePost<RootObjectUser>(
        "http://learn2play.fr/api/authenticate",
        form,
        GetUser,
        RequestFailed,
        BlockButton,
        LoadCards
        );
}
```

Collecter les données
Échec de la requête
Avant l'exécution de la requête
Après l'exécution de la requête

```
void BlockButton()
{
    button.interactable = false;
    GameManager.Instance.SetLoadingLogo(true);
}
```

Avant l'exécution de la requête

```
void RequestFailed()
{
    Display(connexionInvalid);
    button.interactable = true;
    GameManager.Instance.SetLoadingLogo(false);
}
```

Dans le cas échéant de la requête

```
void GetUser(RootObjectUser rootUser)
{
    dataManager.RootUser = rootUser;
}
```

Dans le cas où la requête s'est bien exécutée

```
void LoadCards()
{
    StartCoroutine(LoadCardsCoroutine());
}

IEnumerator LoadCardsCoroutine()
{

    yield return Request.RequestCoroutineGet<CardGame.RootObjectCard>(
        "http://learn2play.fr/api/cardgame/card",
        GetCards,
        null,
        null,
        ConnexionSuccess
        );

}
```

Dans le cas où la requête s'est bien exécutée également

=> Dans ce cas, on doit charger la collection de carte de l'utilisateur qui vient de se connecter, et pour se faire, il faut exécuter une nouvelle requête.

Grâce à la généricité de mon code, on peut donc exécuter une requête après l'exécution d'une autre requête tout en exécutant d'autres méthodes, avant, ou après chaque requête, et ainsi de suite.
Chaque tâche est donc traitée séparément, aux moment voulus. La généricité de ce code permet de coder l'exécution d'une requête très simplement.

Qu’ai-je appris grâce à ce projet?

- Quelques aspects du développement web
- Exécuter des requêtes API au travers d’Unity
- Développer une partie générique
- Architecturer un projet
- Travailler avec une équipe à distance

Conclusion:

Le développement de ce projet m’a permis de faire face à des problèmes dont je n’avais pas les solutions au premier abord.
J’ai appris à me former dans un domaine dont je n’avais que les bases, j’ai pris du plaisir à chercher des solutions, et le fait d’arriver à la finalité de ce projet en respectant mon cahier des charges établi au départ m’a motivé à encore plus pousser mes recherches.

Ce projet me conforte également dans l’idée de travailler dans le domaine du serious game.
Je suis très satisfait de ces 2 mois de travail.