Ticket JO Paris 2024

**Documentation technique**

Table des matières

[**1.** **Introduction** 2](#_Toc180355157)

[**2.** **Conception et architecture** 2](#_Toc180355158)

[**3.** **Spécification technique** 8](#_Toc180355159)

[**4.** **Partie sécurité** 11](#_Toc180355160)

# **Introduction**

Le présent document technique est fait pour présenter le projet Ticket jeux olympique Paris 2024.

Ce projet a été a été conçu pour offrir une expérience utilisateur fluide et sécurisée, en intégrant les dernières technologies web afin de garantir des performances optimales et une navigation intuitive sur tous les appareils.

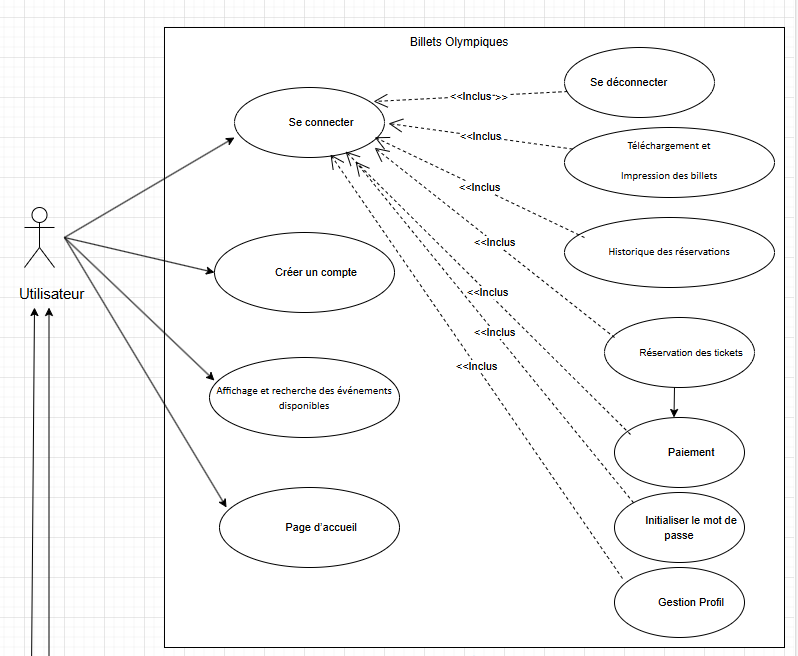
# **Conception et architecture**

a**– Diagramme de cas d’utilisation**

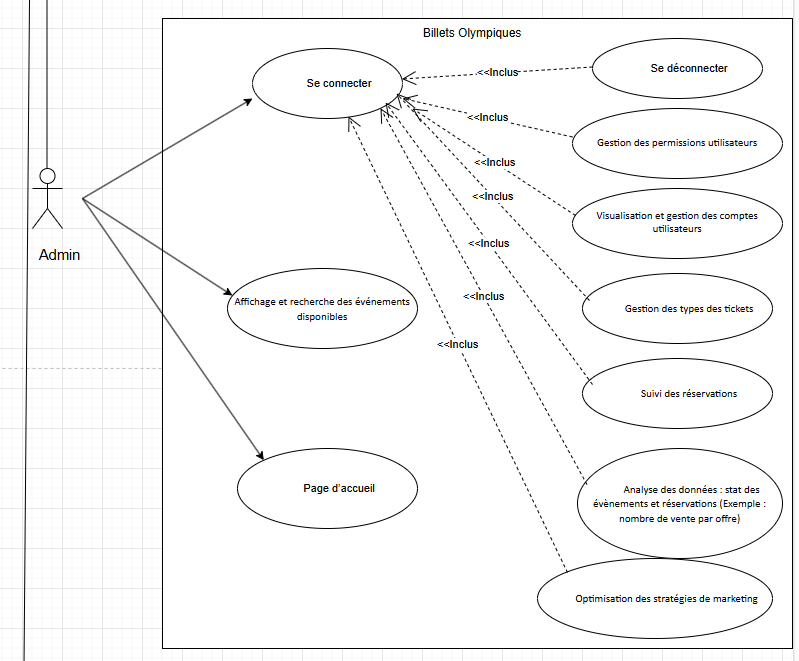
* **Fonctionnalités coté utilisateur**

1. Gestion Compte utilisateur : inscription ou connexion
   1. Création d’un compte utilisateur avec validation par e-mail
   2. Possibilité de réinitialiser le mot de passe.
2. Gestion des tickets :
   1. Affichage et recherche des événements disponibles
   2. Réservation des tickets avec un mode de paiement sécurisé.
   3. Téléchargement et impression des tickets.
   4. Historiques des réservations avec la possibilité d’annulation ou transfert des tickets.

* Gestion Profil :
  + Mise à jour des informations personnelles.

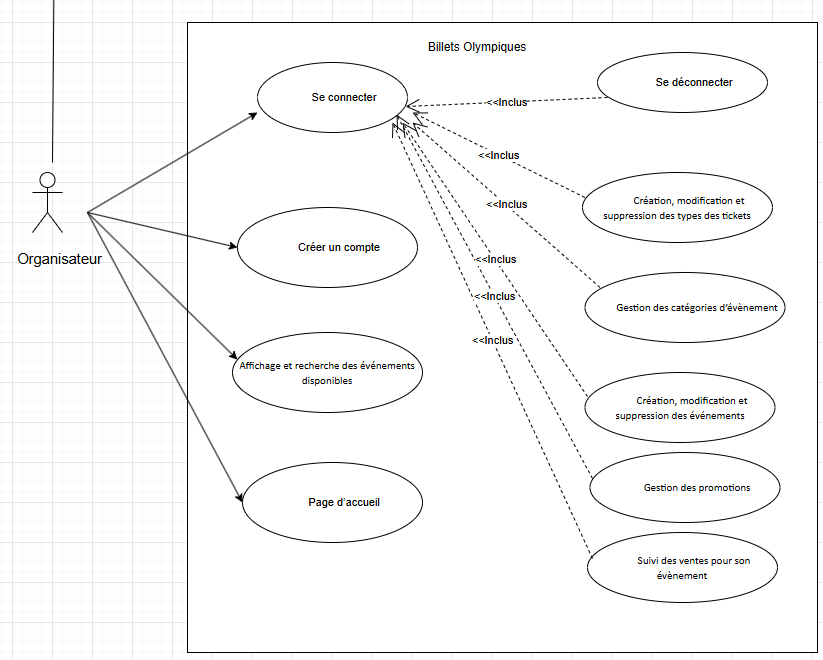


* **Fonctionnalités coté Administrateur**
* Gestion des utilisateurs :
  + Gestion des permissions des utilisateurs
  + Visualisation et gestion des comptes utilisateurs
* Gestions des événements :
  + Paramétrage des statuts des tickets.
  + Suivi des réservations.
* Rapport et analyses
  + Gestion des rapports des réservations et des événements
  + Analyse des données et optimisations des stratégies de marketing.



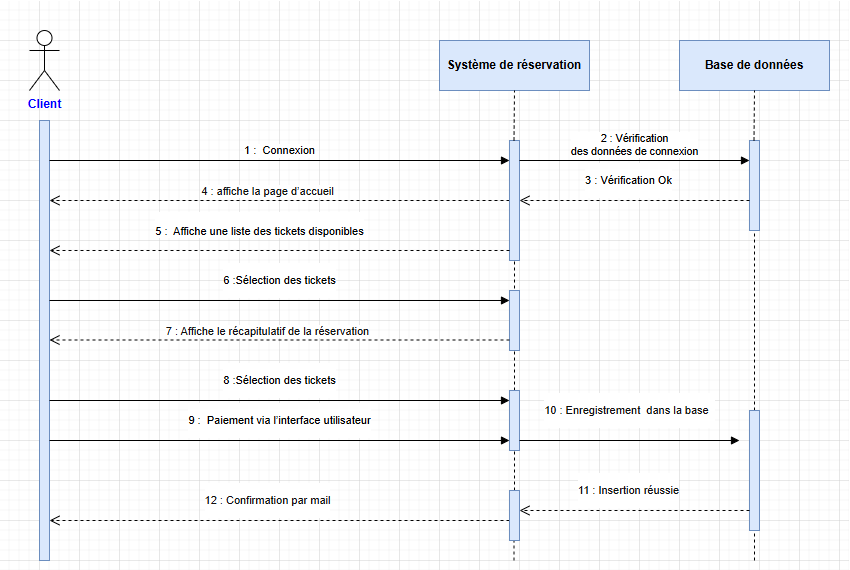
Un compte administrateur ne peut être crée directement dans l’application

* **Fonctionnalités coté Organisateur**
* Gestions des événements :
  + Création, modification et suppression des événements.
  + Paramétrage des statuts des tickets.
  + Gestion des disciplines des événements.
  + Gestion des offres
* Rapport et analyses
  + Suivi des ventes pour son événement

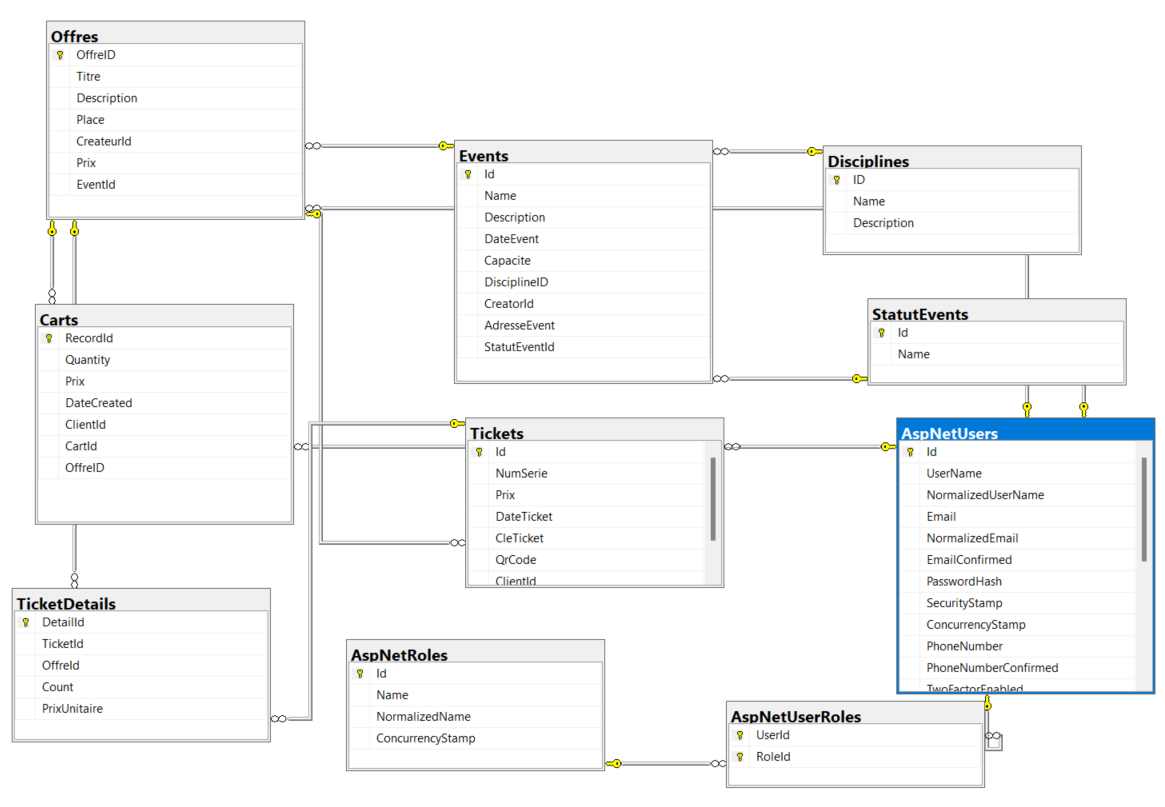


* **Fonctionnalités techniques :**
* Gestion de la Sécurité :
  + Chiffrement des données sensibles : mode de passe et informations de paiement.
  + Protection contre les attaques.
  + Surveillance des accès et des activités suspectes.
* Gestion de la performance
  + Capacité de gérer un grand nombre d’utilisateur.
  + Optimisation des requêtes sql.
  + Utilisation des caches pour améliorer le chargement des pages.
* Intégration :
  + Intégration de la plateforme de paiement : Stripe
  + Intégration des réseaux sociaux pour partager les promotions sur les tickets.
* Sauvegarde
  + Backup journalier de l’application et la base de données sur des serveurs différents.

**b- Diagramme de séquence du processus de vente**



**c- Modèle Conceptuel de Données (MCD)**

****

* **Les tables créent par ASP.net**

Parmi les tables crées par ASP.NE les plus importantes :

* La table **AspNetUser** représente tous les utilisateurs de l’application. Elle hérite de la table Identity User
* La table **AspNetRole** représente les différents rôles que peut prendre un utilisateur.
  + Cette table peut être paramétrée à partir de l’application par l’administration via le menu : page d’accueil 🡺 Gestion utilisateurs

Le rôle « Admi », ne peut être affiché ni crée

* La table **AspNetUserToken** représente un jeton d’utilisation pour un utilisateur donnée. Cette table permet de gérer les sessions des utilisateurs de l’application
* La table **AspNetUserRole** est une table qui permet d’associer les utilisateurs et les rôles.

Un utilisateur et Rôle sont liés par une relation Many To Many qui créé l’entité d’association UserRole.

L’administrateur peut paramétrer les rôles pour certains utilisateurs : comme les organisateurs des événements, des commerciaux ….

Page d’accueil 🡺 Gestion utilisateurs 🡺 Gestion des rôles utilisateur

* **Les tables crées par le model entity Framework**
* La table **Event** : correspond à la liste des événements. Ces événements peuvent être crée par l’administrateur ou un organisateur.

Un utilisateur (Admin ou organisateur) peut crée plusieurs événement, mais un événement appartient à un seul utilisateur d’où la relation On-ToMany

La création des événements se fait via le bouton disponible au niveau de la gestion des événements : visible que pour l’administrateur et l’organisateur.

* La table **Disciplines** : correspond à la différente catégorie de sports : un événement appartient à une catégorie et une catégorie contient plusieurs évènements : One-Many

L’utilisateur peut ajouter directement la discipline au moment de la saisie

* La table Offre : Correspond aux différents offre lié à un événement.

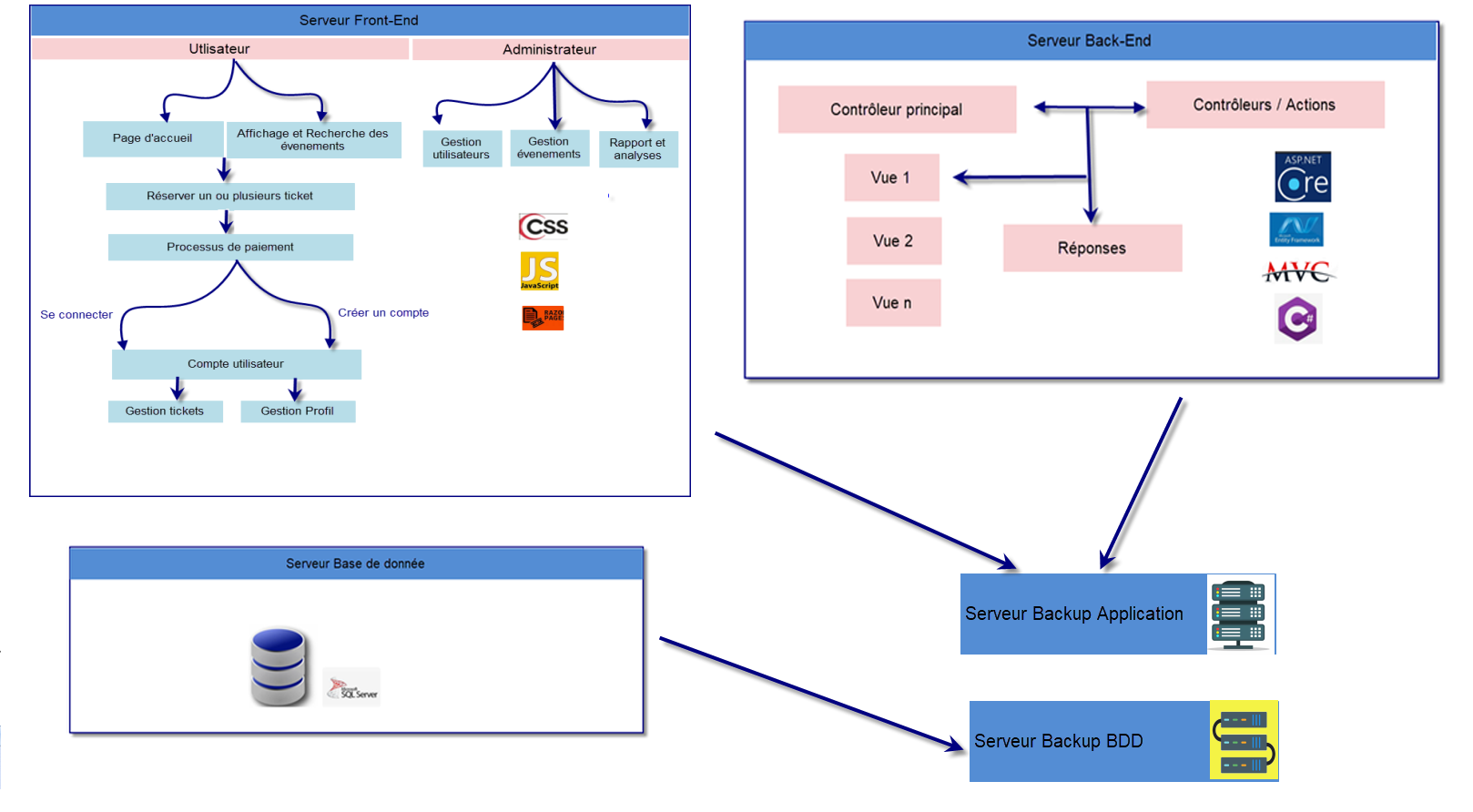
Un évènement peut avoir plusieurs offres, et une offre appartient à un seul événement 🡺 On-ToMay

* La table Tickets : Correspond aux différents tickets liés à une offre donnée d’où la relation une offre peut avoir plusieurs tickets mais un ticket correspond à une offre 🡺 On-ToMay

Le ticket est lié à un utilisateur donné, et un utilisateur peut avoir plusieurs tickets 🡺 On-ToMay

* La table Carts : elle décrit le panier utilisateur, liée à l’utilisateur et les offres sélectionnées

**c- Architecture**



L’architecture est composée de trois parties

* + - Partie Font-end
    - Partie-back-end
    - Base de données

# Avec des serveurs de Backup : un pour l’application, et le deuxième pour la base de donnée

# **3. Spécification technique**

# Environnement de travail

Le développement a été fait sous Windows 10

# Outils de développement

* IDE : Visual studio 2022 : parfaitement adapté pour tout projet C#.

Il existe pour tout type de système d’exploitation.

# Coté Back-end

- C# :

Il bénéficie d'un écosystème d'outils de développement extrêmement performant, notamment avec **Visual Studio** et **Visual Studio Code.**

Un langage orienté objet, fortement typé, puissant et complet pour le développement des applications.Net

- Le Framework ASP.NET CORE (7.0.20) :

Multiplateforme, Il propose une structure MVC (Modèle, Vues, Contrôleur) qui permet de bien structurer et organiser son code. Il dispose d’une bibliothèque riche et complète pour assurer la sécurité et gérer les authentifications.

- L'ORM (Object-Relational Mapping) Entity Framework (7.0.0) : **simplifier l'accès aux bases de données relationnelles**. Au lieu d'écrire des requêtes SQL brutes pour interagir avec la base de données, on utilise du code C# (ou un autre langage .NET) pour manipuler les données sous forme d'objets.

1. Coté Font-end

- HTML 5 :

Il contient des éléments permettent de structurer le contenu de manière plus intuitive et plus **sémantique**, facilitant le référencement (SEO) et l'accessibilité.

Il introduit de nombreuses **API JavaScript natives** qui permettent d'ajouter des fonctionnalités

- CSS 5 / Bootstrap (5.3.3) :

Le CSS offre une grande flexibilité pour personnaliser les éléments HTML, et en maîtrisant bien ses concepts (mise en page, transitions, animations, etc.), en créant des interfaces très complexes.

. Le framework Bootstrap offre une large palette de composants et de styles prêts à l'emploi permettant de **gagner du temps** en proposant des composants et des styles préconstruits, tout en rendant le site **réactif** par défaut.

- Javascript: pour ajouter de l'interactivité à une page.

- Razor page : Integrées avec ASP.NET Core, conçu pour faciliter la création d'applications web dynamiques. Il offre une approche simplifiée par rapport au modèle traditionnel MVC (Model-View-Controller) en permettant de regrouper la logique de présentation et la logique de traitement des données dans une seule page.

# Base de donnée

* Sql serveur.

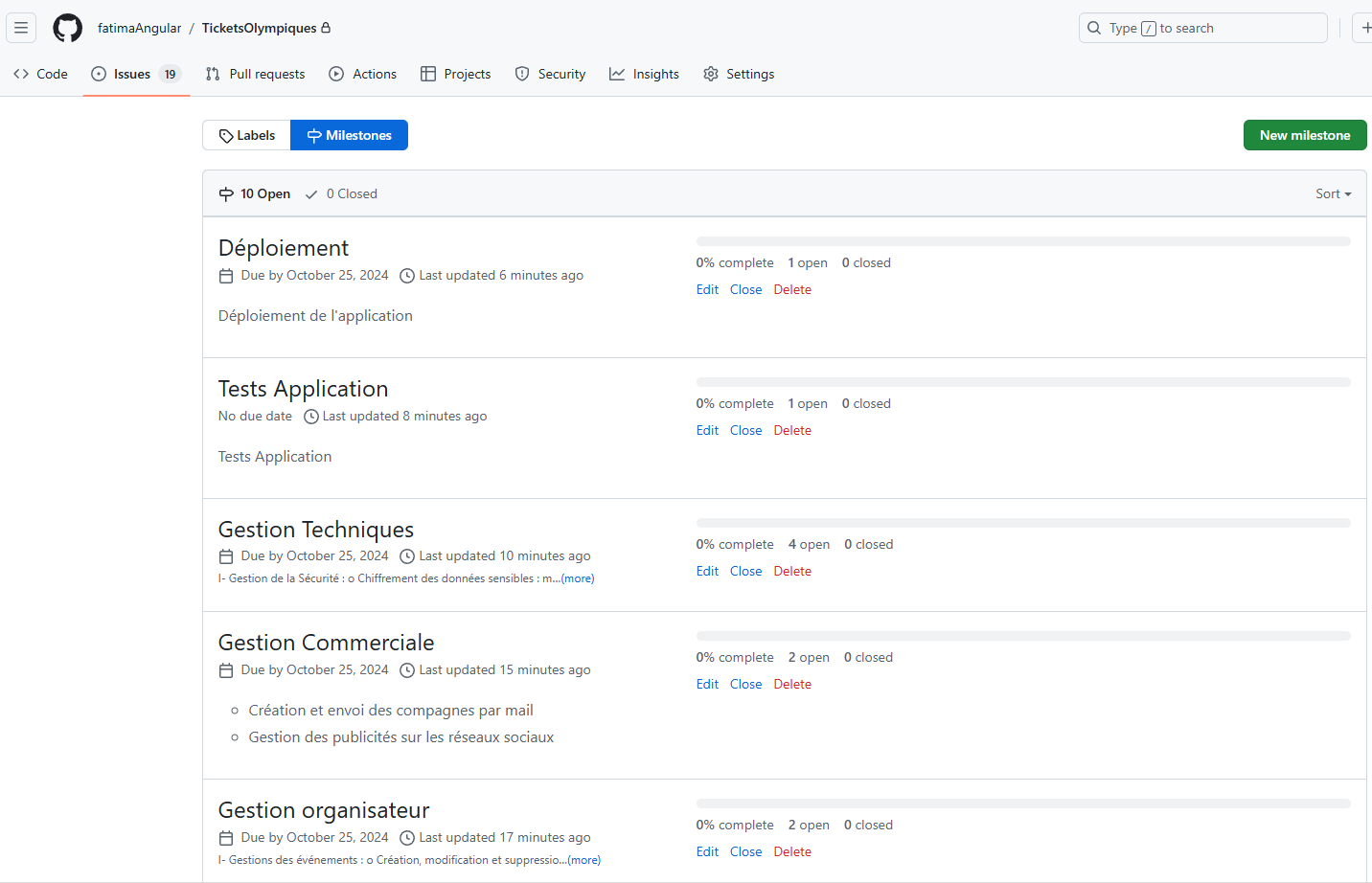
# Outils de versionning

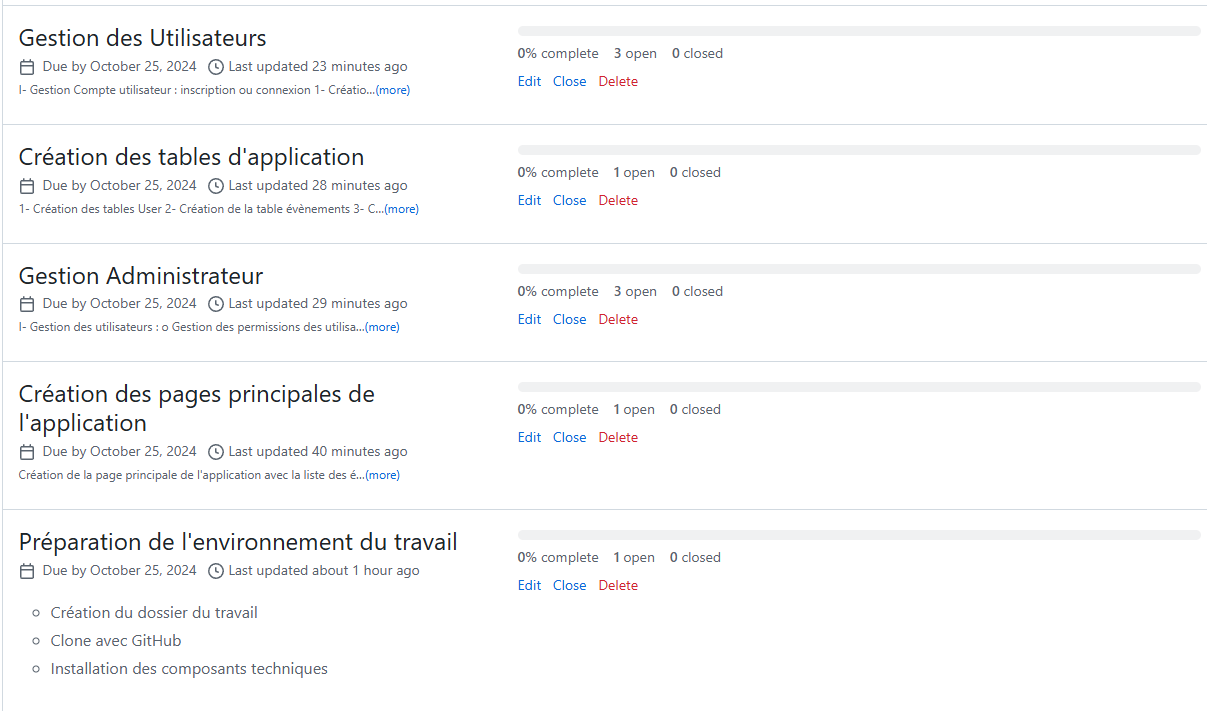
**Git et Github**

# Gestion de projet

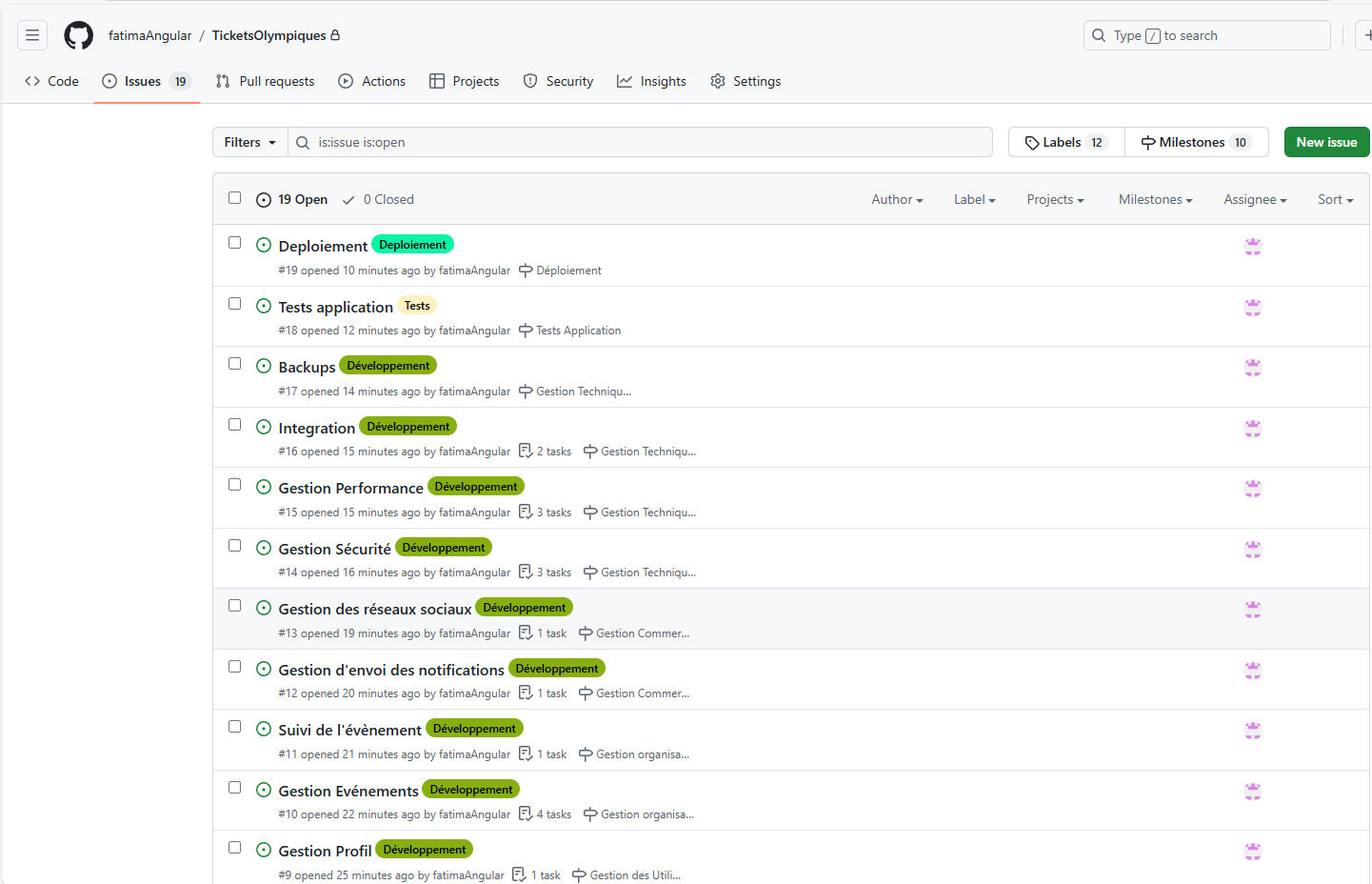
La gestion du projet a été faite au niveau de GiHub par le tableau Kanban

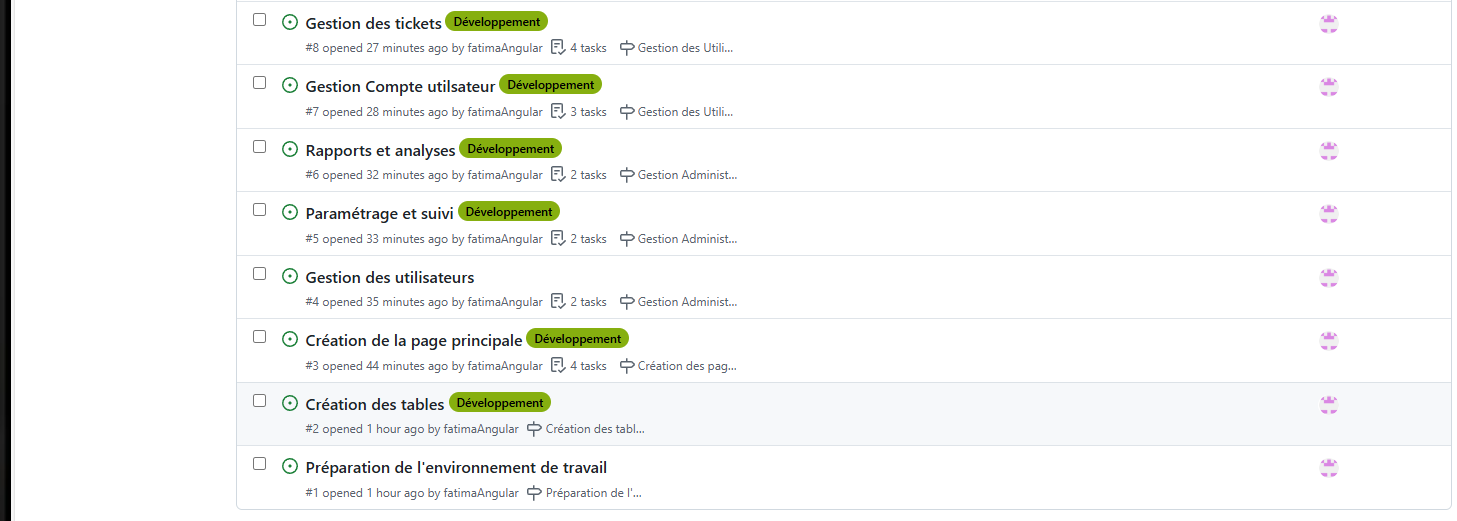
1. Création d’un Repository intitulé « [**TicketsOlympiques**](https://github.com/fatimaAngular/TicketsOlympiques)**»**
2. Création d’un Project intitulé : « **Gestion tickets des jeux olympiques** »
3. Création de la liste des milestones (10 en total)

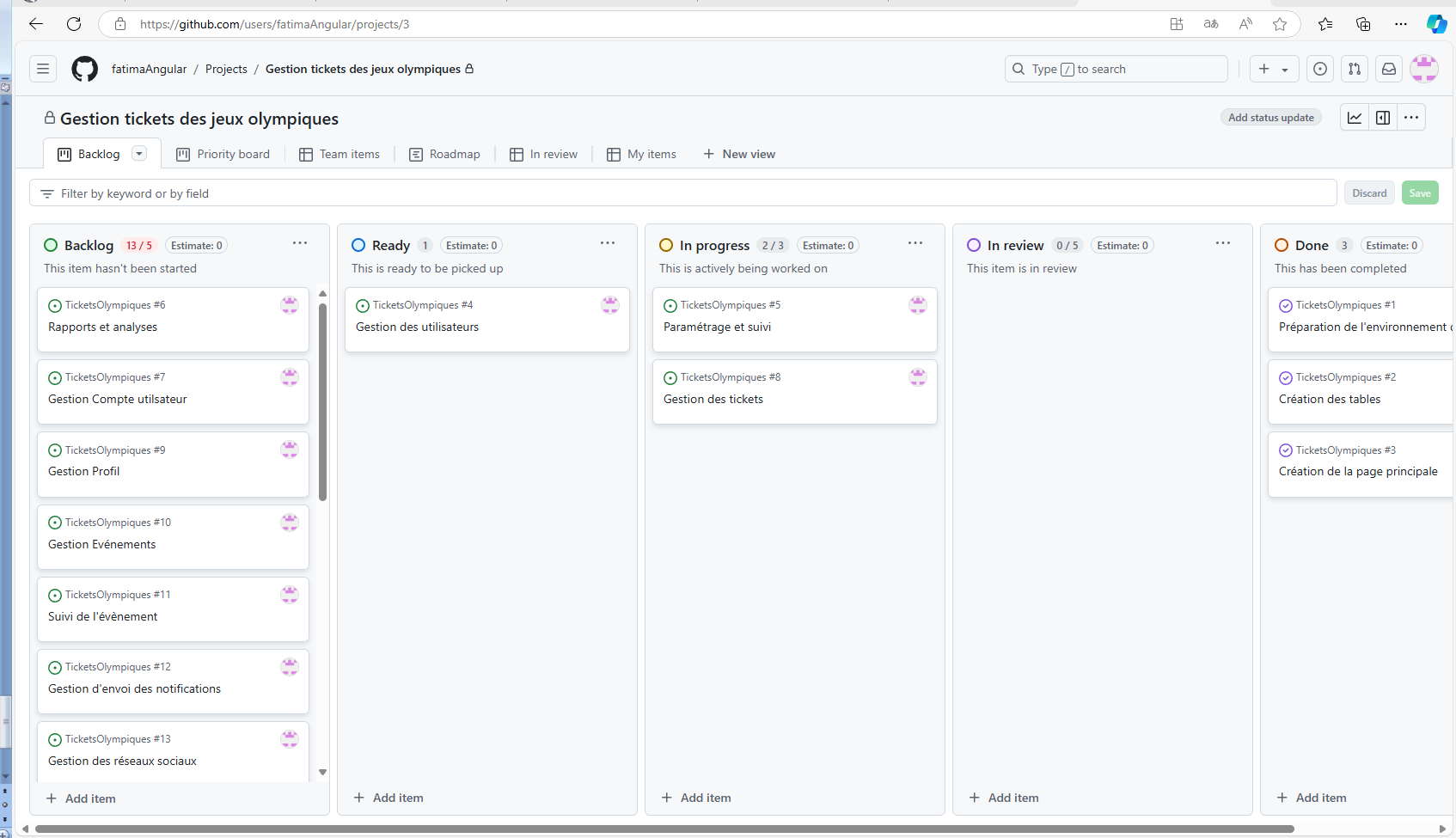




Création de la liste des issues (19 en total)

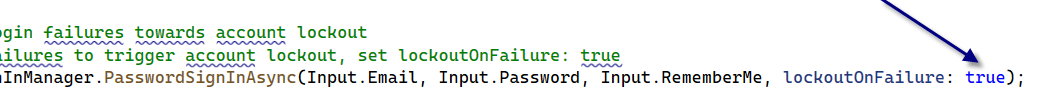




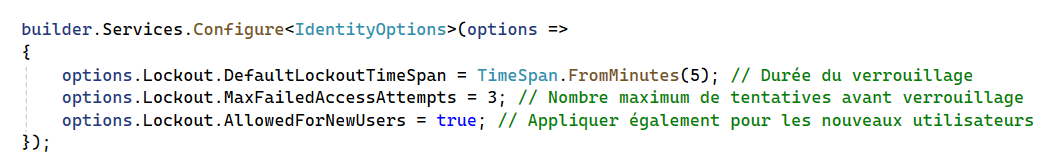


# **Partie sécurité**

* Exiger aux utilisateurs un mot de passe robuste composé des caractères majuscule, minuscules, des caractères spéciaux, et des chiffres, afin d’éviter les **attaques par force brute**, et bloquer la connexion au bout de trois tentative de connexion, ce contrôle est en mis en place par la méthode PasswordSignInAsync de Asp.net en mettant le paramètre lockoutonFailure à true



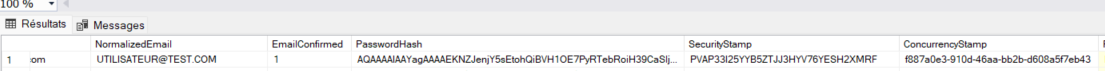
En ajoutant le service correspondant dans la classe Program.cs





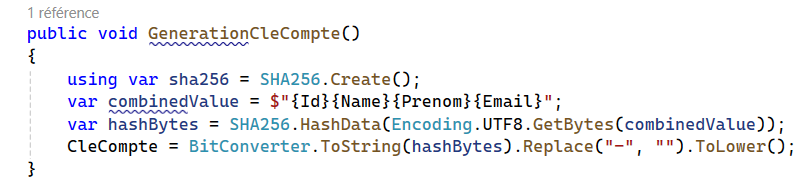
* Confirmation d’inscription par mail : utilisation des fonctions proposées par ASP.net comme : SendEmailAsync
* Chiffrer les données sensibles directement dans la base de données en utilisant les bibliothèques de chiffrement coté ASP.NET : (System.Security.Cryptography).

* Utilisation de l’ASP.net core MVC avec Identity donne plusieurs fonctions permettant le hachage des mots de passe ce qui permet une protection importante de cette données dans la base.



* Utilisation d’une clé unique générée automatiquement par l’application qui permet la confirmation de l’identité de l’utilisateur.

Cette clé est généré dans l’application par la fonction GenerationCleCompte() dans la classe User qui hérite de la classe IdentityUser



* On peut aussi utiliser Protection contre les injections SQL :
  + Utilisation du Entity FrameWork pour communiquer avec la base de données.
  + Utilisation des requêtes paramétrées pour éviter les injections sql.
* Chiffrement des communications avec la base de données
* Chiffrement de la base de donnée sql : il faut utiliser la fonction Transparent Data Encryption (TDE).

Cette fonction est native dans Microsoft SQL Server Entreprise, avec un chiffrement en AES 256.

* Création de la clé principale : Pour TDE, une clé principale doit être créée dans la base de données [master] (maître). Chaque serveur de base de données ne peut disposer que d'une clé principale qui est partagée par toutes les bases de données utilisateur. Le mot de passe doit être fourni dans le script dans cette étape. Si la clé principale n'existe pas encore, elle sera créée avec le mot de passe fourni.

USEmaster;

GO

CREATE MASTER KEY ENCRYPTION BY PASSWORD='\*\*\*\*\*\*\*';

GO

* Vérification mot de passe de la master key :

DECLARE @MKPassword NVARCHAR(255)= '\*\*\*\*\*\*\*'

DECLARE @sql NVARCHAR(4000)

BEGINTRY

-- PRINT 'INFO: Open master key withspecifiedpassword'

SET @sql='USE master; OPEN MASTER KEY DECRYPTION BY PASSWORD = '''+ @MKPassword+''''

EXECsp\_executesql@sql

END TRY

BEGIN CATCH

PRINT'ERROR 2: '+ERROR\_MESSAGE();

RETURN

END CATCH

Si le mot de passe correspond : Commandes réussies.

Si le mot de passe est erroné : ERROR 2: La clé n'est pas chiffrée avec le déchiffreur spécifié.

## Creation du certificat de chiffrement

Il suffit de créer un certificat qui s’appellera : Certificatdechiffrement par exemple

Renseigner les dates de début et de fin du certificat :

UseMaster

GO

CREATE CERTIFICATE Certificatdechiffrement

WITH SUBJECT='Le certificat qui permettra de chiffrer les bases et les sauvegardes',

START\_DATE='20190507',

EXPIRY\_DATE='20301231';

* + Coté client dans et dans la chaine de connexion, il faut ajouter l’option : Encrypt=True pour crypter les communications entre ASP.NET et la base de donnée.
* Gestion des rôles et permissions : définir les rôles te les permissions pour chaque utilisateurs et restreindre l’accès aux fonctionnalités en fonction des rôles.
  + Role admin : impossible de le créer dans l’application, il est crée manuellement via des script sql
    - Le premier script permet d’inserer les differents roles dans la table AspNetRoles :

INSERT INTO AspNetRoles (Id, Name, NormalizedName, ConcurrencyStamp)

VALUES

(NEWID(), 'User', 'User', NEWID()),

(NEWID(), 'Organizer', 'ORGANIZER', NEWID()),

(NEWID(), 'Admin', 'ADMIN', NEWID());

* + - Le deuxiéme permet de crer le compte Admi comme demandé par le client

INSERT INTO [dbo].[AspNetUsers]

([Id]

,[UserName]

,[NormalizedUserName]

,[Email]

,[NormalizedEmail]

,[EmailConfirmed]

,[PasswordHash]

,[SecurityStamp]

,[ConcurrencyStamp]

,[PhoneNumber]

,[PhoneNumberConfirmed]

,[TwoFactorEnabled]

,[LockoutEnd]

,[LockoutEnabled]

,[AccessFailedCount]

,[Discriminator]))

VALUES

(Newid()

,’Administrateur@TicketJO.com'

,'ADMINISTRATEUR@TICKETJO.COM'

,'Administrateur@TicketJO.com'

, ADMINISTRATEUR@TICKETJO.COM'

,1

,'AQAAAAIAAYagAAAAEMTxqiU9q1Dc6EruywAhDIn6t6ghFBuGx5Bm56SrxJSfmtQQplFOUWTzN8eIOnoeFg=='

,'M6JIG62D4VNOXEO4POTF4MBGANNQX4FC'

,'9e570f2e-3305-4172-a0d6-99bf444452b2'

,Null

,0

,0

,NULL

,1

,0

,'User' )

* + - Le troisième permet d’affecter le role »Admin » à la personne concerné

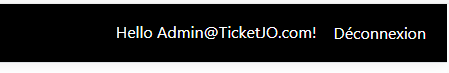
DECLARE @userId1 nvarchar(450) = (SELECT Id FROM AspNetUsers WHERE Email = 'Administrateur@TicketJO.com ')

DECLARE @roleId1 nvarchar(450) = (SELECT Id FROM AspNetRoles WHERE Name = 'Admin')

INSERT INTO AspNetUserRoles (UserId, RoleId)

VALUES (@userId1, @roleId1)

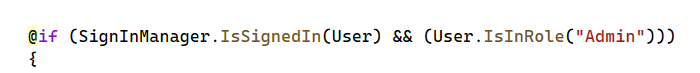
L’administrateur peut se connecter dans l’application



L’administrateur peut accéder à un menu spécial pour gérer les utilisateurs, gérer les événements, les statistiques



L’affichage de ce menu est contrôle par la ligne suivante :

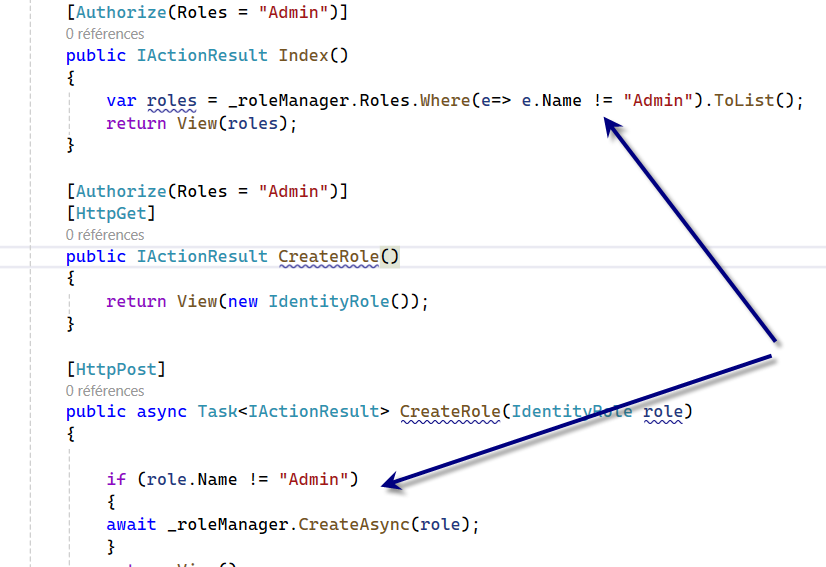


L’administrateur peut aussi créer un utilisateur de type « Organisateur » qui peut s’occuper de la gestion des événements

Son menu général est comme suit :



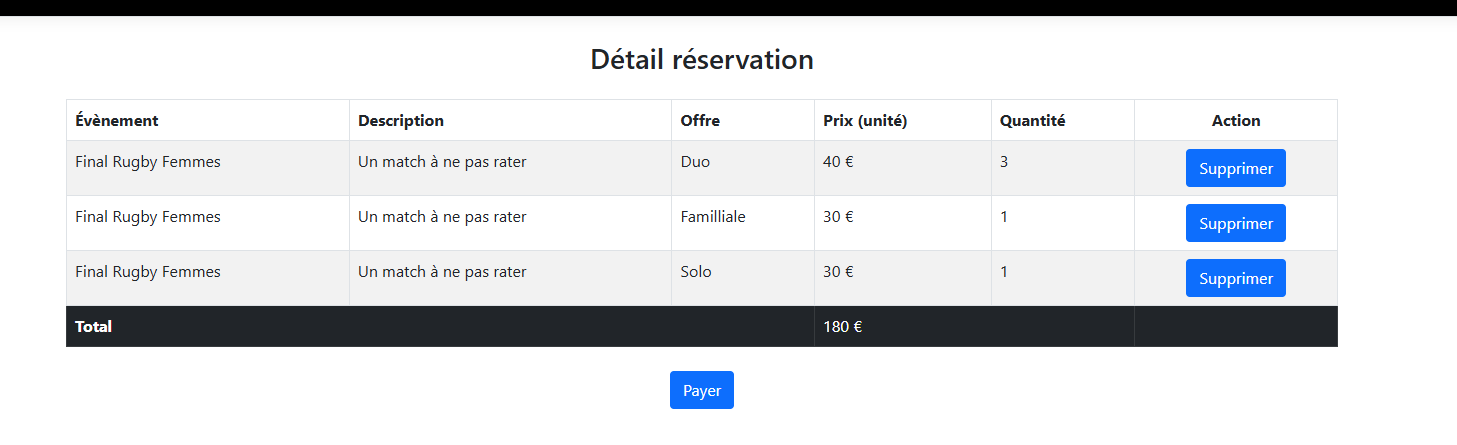
Remarque : L’affectation du Rôle « Admin » est impossible au niveau de l’application même par l’administrateur suite au plusieurs contrôles inséré au niveau du contrôleur correspondant :



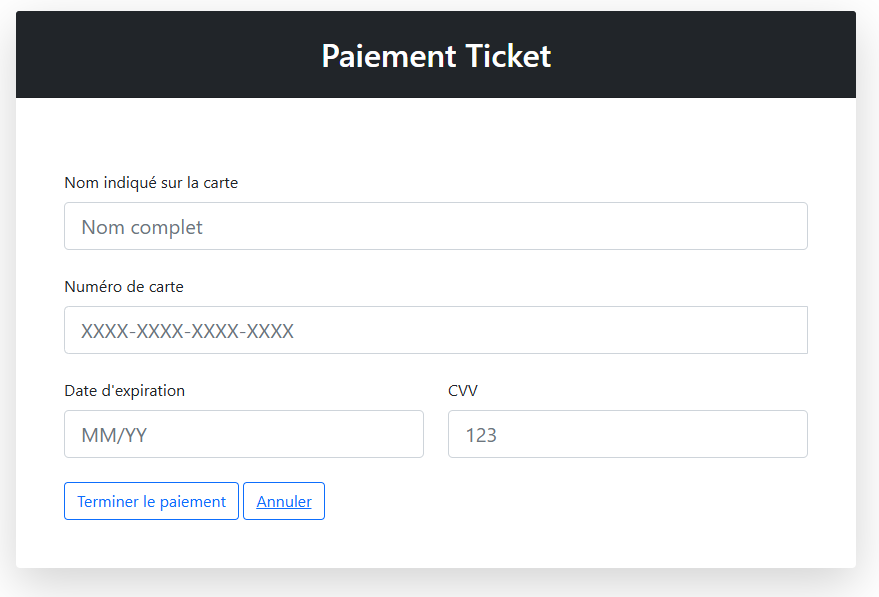


Pour un utlisateur normal : il a accès à son historique et son profil

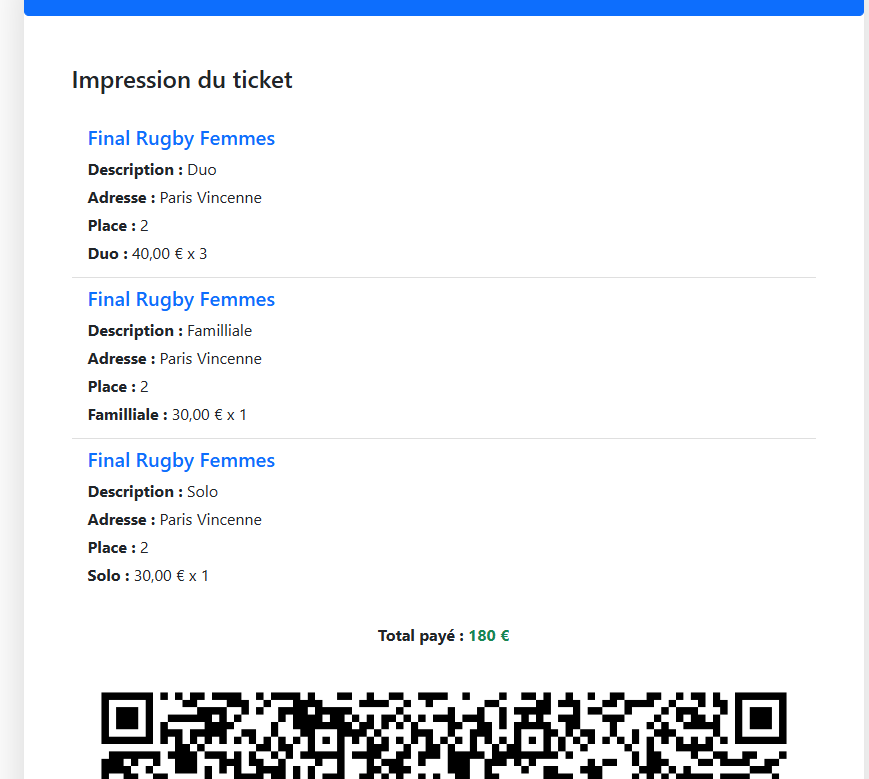
* + - Il peut réserver des tickets selon les offres



* + - Payer sa commande



* + - Imprimer son ticket



# **Futures évolutions**

1. Il faut ajouter l’archive des événements : les clôturer puis les archiver pour qu’ils ne restent pas dans la liste d’actualité.
2. Le même principe pour les offres.
3. Ajouter des différentes statiques pour savoir lequel des événements est plus rentable.
4. Jouter un nouveau utilisateur de Rôle « Commerciale » qui peut gérer les notifications et les réseaux sociaux pour les nouvelles offres