



MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER PROFESSIONNEL^(DP)

<i>Nom de naissance</i>	► COSTANZO
<i>Nom d'usage</i>	► COSTANZO
<i>Prénom</i>	► Franck
<i>Adresse</i>	► 3, bd des Bouires, Res Les Comtes Nord BAT F1 13012 MARSEILLE

Titre professionnel visé

Concepteur Développeur d'Applications

MODALITÉ D'ACCÈS :

- Parcours de formation
 Validation des Acquis de l'Expérience (VAE)

DOSSIER PROFESSIONNEL (DP)

Présentation du dossier

Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel.
Ce titre est délivré par le Ministère chargé de l'emploi.

Le DP appartient au candidat. Il le conserve, l'actualise durant son parcours et le présente **obligatoirement à chaque session d'examen.**

Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE. Il est consulté par le jury au moment de la session d'examen.

Pour prendre sa décision, le jury dispose :

1. des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l'entretien professionnel ou de l'entretien technique ou du questionnement à partir de productions.
2. du **Dossier Professionnel (DP)** dans lequel le candidat a consigné les preuves de sa pratique professionnelle.
3. des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d'un parcours de formation
4. de l'entretien final (dans le cadre de la session titre).

[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels du ministère chargé de l'Emploi]

Ce dossier comporte :

- pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ;
- un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d'un titre, d'un diplôme, d'un certificat de qualification professionnelle (CQP) ou des attestations de formation ;
- une déclaration sur l'honneur à compléter et à signer ;
- des documents illustrant la pratique professionnelle du candidat (facultatif)
- des annexes, si nécessaire.

DOSSIER PROFESSIONNEL (DP)

Pour compléter ce dossier, le candidat dispose d'un site web en accès libre sur le site.

 <http://travail-emploi.gouv.fr/titres-professionnels>

Sommaire

Exemples de pratique professionnelle

Concevoir et développer des composants d'interface utilisateur	p.	5
▶ Application Mobile SMOGON Forum	p.	5
▶ AG0630 - Programme Agence Richardson	p.	15
Concevoir et développer la persistance des données	p.	23
▶ Application Mobile SMOGON Forum	p.	23
Concevoir et développer une application multicouche répartie	p.	32
▶ Application Mobile SMOGON Forum	p.	32
Déclaration sur l'honneur	p.	43

DOSSIER PROFESSIONNEL^(DP)

EXEMPLES DE PRATIQUE

PROFESSIONNELLE

DOSSIER PROFESSIONNEL (DP)

Concevoir et développer des composants

Activité-type 1 d'interface utilisateur

Exemple n°1 ▶ Application Mobile SMOGON Forum

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

L'association SMOGON University possède un site internet responsive complet et fonctionnel sans aucune forme de monétisation. Lors de nos échanges, ils ont formulé le besoin d'apporter de la visibilité supplémentaire au site web existant et de faire grandir la communauté de SMOGON.

Sur leur site, on peut voir, entre autre la description suivante de leur communauté :

" Smogon is the most comprehensive and accurate online resource for competitive Pokémon battling. We offer articles and advice via our community forums to help fans of the game compete at every level, while honing their skills in every aspect of competitive Pokémon from team building to battling tactics. Our over-450,000-member organization is growing at an ever increasing rate, constantly expanding our knowledge base and our ability to be at the cutting edge of the game "

Nous avons ainsi décidé de reproduire l'expérience du forum pour ne pas choquer le public habitué de longue date, mais de néanmoins inclure des fonctionnalités supplémentaires. Pour plus de clarté, je vais étayer les principaux objectifs de l'application :

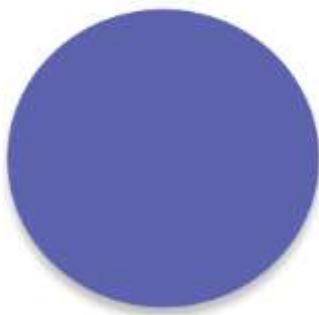
- reproduire les fonctionnalités de base du forum (connexion, gestion de compte, post sur les sujets, création de sujet, réaction aux sujet, etc)
- permettre au utilisateur de sauvegarder des équipes de pokémon sur leur profil (consultable par l'intégralité des membres du forum) en utilisant le format d'équipe mis en place par la communauté de SMOGON et pokemonshowdown, via le site pokepaste
- intégrer les équipes dans les posts mais aussi permettre de les copier rapidement
- intégrer un accès direct depuis l'application au site partenaire pokemonshowdown ou l'utilisateur peut copier ses équipes

DOSSIER PROFESSIONNEL^(DP)

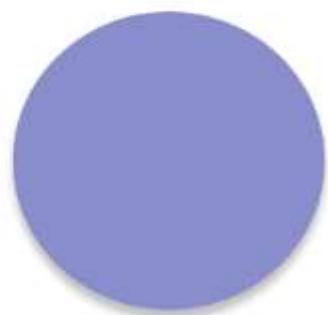
À la suite de la rédaction du cahier des charges, grâce à l'existant j'ai établi la charte graphique suivante :



#ECECEC



#6363B0



#8d8dcf

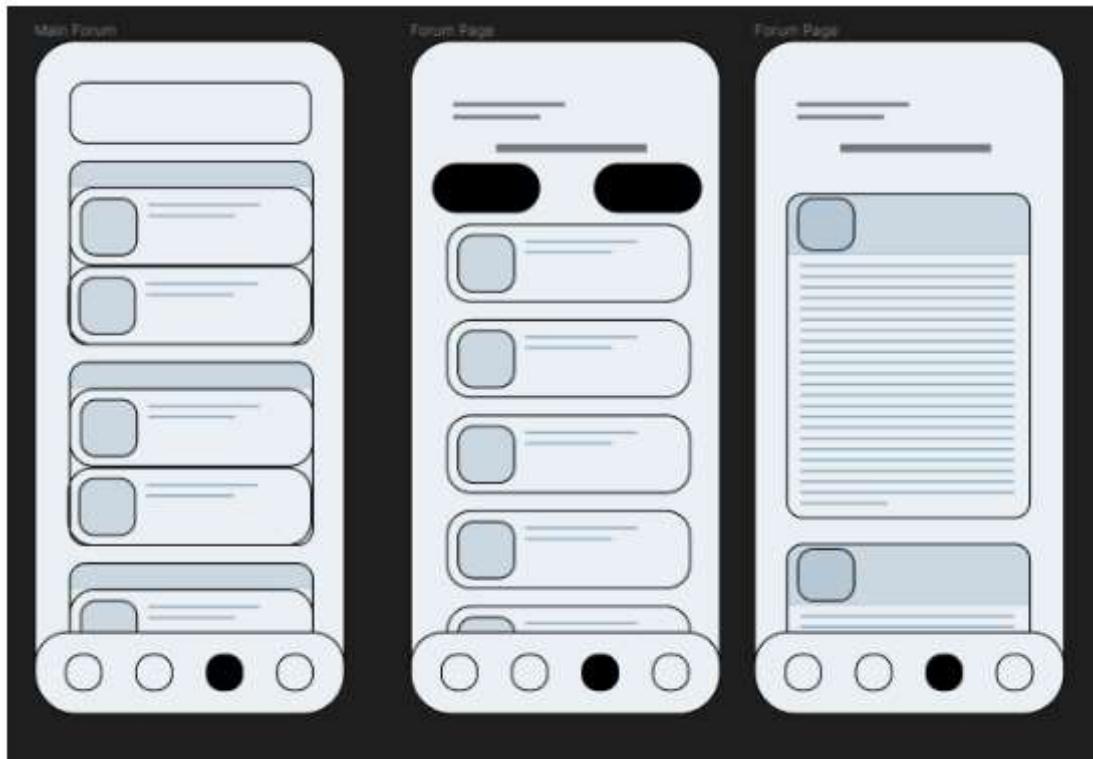
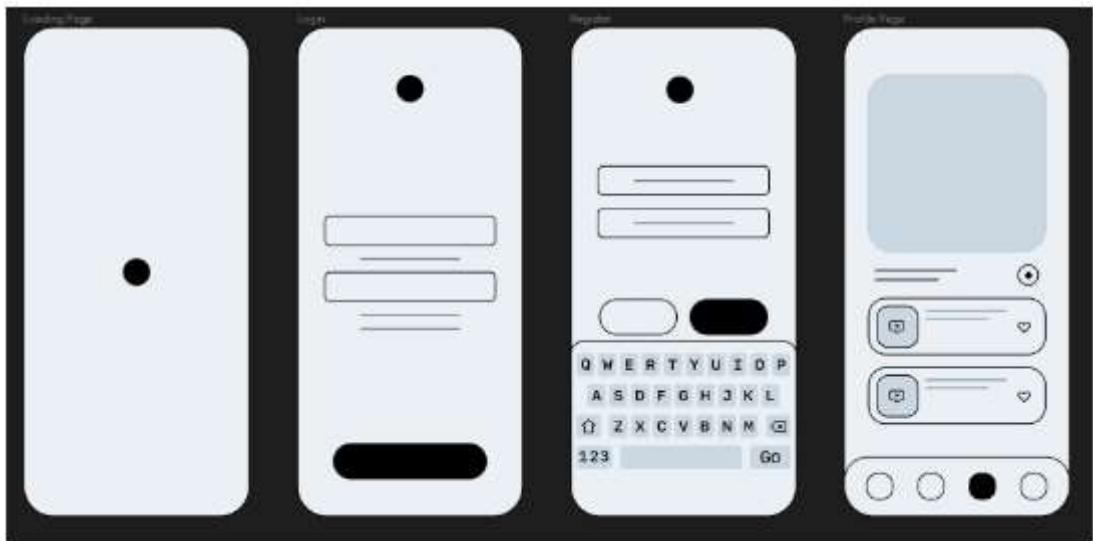
Main Font : Roboto Regular #000000

Exemples, counts, etc : Robot Regular #8C8C8C

Important Text : Roboto Bold #094ABA

DOSSIER PROFESSIONNEL (DP)

Puis en accord avec l'équipe du site nous avons posé le wireframe suivant :

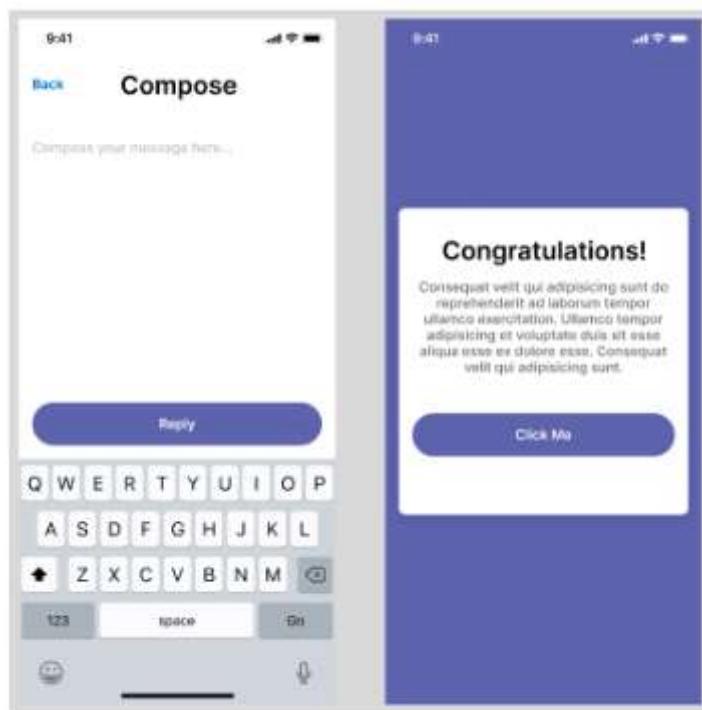
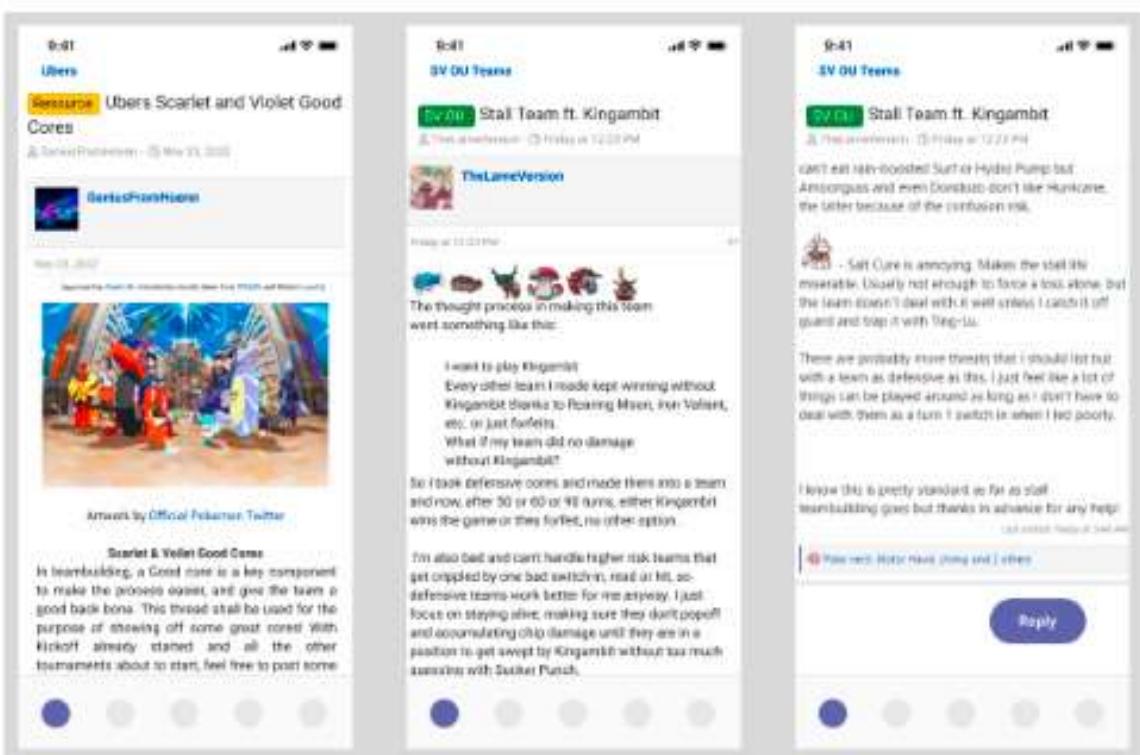


DOSSIER PROFESSIONNEL (DP)

Après validation du wireframe, j'ai établi les maquettes suivante :

The image displays six mobile application screens arranged in two rows of three. The top row shows the Sign Up, Log In, and Profile screens. The Sign Up screen includes fields for Name, Email, Password, and a newsletter checkbox. The Log In screen has fields for Email and Password. The Profile screen features a circular profile picture of Domo-Kun, a blue banner with the name, and a section for Last Posts with three entries. The bottom row shows the Main Forums, Smogon Metagames, and Smogon Metagames categories screens. The Main Forums screen lists Information & Resources, Announcements, and Competitive Play threads. The Smogon Metagames screen lists OverUsed, Ubers, UnderUsed, RarelyUsed, Little Cup, and Doubles OU categories. The Smogon Metagames categories screen lists threads like Ubers Winter League II - Week 1, Kickoff Again - Round 5, and SV Ubers Viability Rankings.

DOSSIER PROFESSIONNEL (DP)



DOSSIER PROFESSIONNEL (DP)

Le framework MAUI fonctionne avec des pages et du code-behind (littéralement code derrière) découpé en deux scripts allant de pair, exemple

```
ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="Smogon_MAUIapp.Pages.Login"
    Title="Login">
    <ScrollView>
        <Grid>
            <Grid.RowDefinitions>
                <RowDefinition Height="3*"/>
                <RowDefinition Height="1*"/>
                <RowDefinition Height="1*"/>
                <RowDefinition Height="1*"/>
                <RowDefinition Height="1*"/>
                <RowDefinition Height="1*"/>
                <RowDefinition Height="1*"/>
            </Grid.RowDefinitions>

            <Grid Grid.Row="0">
                <Grid.RowDefinitions>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                </Grid.RowDefinitions>
                <Label Grid.Row="0"
                    HeightRequest="175" BackgroundColor="#636368"
                    Text="Login" FontAttributes="Bold"
                    TextColor="White" FontSize="30"
                    Padding="20"
                    HorizontalTextAlignment="Center" VerticalTextAlignment="Center"/>
            </Grid>
        </Grid>
    </ScrollView>

```

```
public partial class Login : ContentPage
{
    private UserService userService = new UserService();

    public Login()
    {
        InitializeComponent();
    }

    private async void LoginUser(object sender, EventArgs e)
    {
        if(usernameInput.Text.IsNullOrEmpty() || passwordInput.Text.IsNullOrEmpty())
        {
            await DisplayAlert("Error", "All the fields must be filled in order to login", "OK");
        }
        else
        {
            var token = await userService.LoginUserJWT(usernameInput.Text.Trim(), passwordInput.Text.Trim());
            if(token != null)
            {
                Preferences.Set("token", token.RawData);
                Application.Current.MainPage = new AppShell();
            }
            else
            {
                await DisplayAlert("Error", "We have been unable to log you in with the information provided", "OK");
            }
        }
    }

    private async void PasswordForgottenAsync(object sender, EventArgs e)
    {
        string result = await DisplayPromptAsync("Forgotten Password : ", "Write down your email address :");
    }
}
```

DOSSIER PROFESSIONNEL (DP)

Pour la main page qui affiche les différents forum et qui va chercher ces informations via l'API conçue pour le site (voir partie relative à la persistance des données), l'accès aux données se fait comme suit :

```
<Label Text="Introduction to Smogon" TextColor="#Blue" HorizontalOptions="CenterAndExpand">
    <Label.GestureRecognizers>
        <TapGestureRecognizer Tapped="Intro_Tapped"/>
    </Label.GestureRecognizers>
</Label>

<Label HorizontalOptions="CenterAndExpand"
    Text="For a run-down on everything Smogon, and make sure you take some time to read the " />

<Label Text="Global Rules" TextColor="#Blue" HorizontalOptions="CenterAndExpand">
    <Label.GestureRecognizers>
        <TapGestureRecognizer Tapped="Rules_Tapped"/>
    </Label.GestureRecognizers>
</Label>
</StackLayout>
</Frame>

<StackLayout x:Name="loadingImage"
    IsVisible="False" |
    VerticalOptions="CenterAndExpand"
    HorizontalOptions="CenterAndExpand">
    <Label Text="Loading..." HorizontalTextAlignment="Center"/>
</StackLayout>

<ListView x:Name="mainPageView"
    Margin="20" MinimumHeightRequest="1000" HasUnevenRows="True"
    VerticalScrollBarVisibility="Never">

    <ListView.ItemTemplate>
        <DataTemplate>

            <ViewCell>
                <StackLayout Margin = "0,0,0,30">

                    <Label Text="{Binding name}" FontAttributes="Bold"
                        FontSize="18"
                        HorizontalTextAlignment="Center" VerticalTextAlignment="Center"
                        Background="#8d8ddc" TextColor="White" HeightRequest="50"/>

                    <ListView ItemsSource="{Binding forums}" HasUnevenRows="True"
                        BackgroundColor="#FFFFFF"
                        VerticalScrollBarVisibility="Never">

                        <ListView.ItemTemplate>
```

Une listview est mise en place et on bind certaines informations au template de composant ce qui aura pour effet d'afficher le nombre d'éléments contenu dans la liste qui est attachée à la listview.

DOSSIER PROFESSIONNEL (DP)

Au démarrage de la page “ MainPage ”, le code est comme suit :

```
5 références
public partial class MainPage : ContentPage
{
    #region Properties

        public string SmogonIntroduction = Tools.LinkText.smogonIntroduction;

        public string GlobalRules = Tools.LinkText.globalRules;

    #endregion

    #region Constructor

    1 référence
    public MainPage()
    {
        InitializeComponent();
        ChangeItemSource();
    }
}
```

On construit l'objet, on l'initialise puis on utilise la fonction “ChangeItemSource()” pour attacher une liste à la listview :

```
1 référence
private async void ChangeItemSource()
{
    loadingImage.IsVisible = true;

    TopicService topicService = new TopicService();
    List<Topics> topics = new List<Topics>();
    var aTask = Task.Run(async () =>
    {
        topics = await topicService.GetAllTopics();
    });

    await Task.WhenAll(aTask);

    List<Task> topicsTask = new List<Task>();

    foreach (var item in topics)
    {
        var tempTask = Task.Run(async () =>
        {
            item.forums = await topicService.GetForumsByTopicId(item.topic_id);
        });
        topicsTask.Add(tempTask);
    }

    await Task.WhenAll(topicsTask);

    if (topics != null)
    {
        loadingImage.IsVisible = false;
        mainPageView.ItemsSource = topics;
    }
}
```

DOSSIER PROFESSIONNEL (DP)

Le TopicService suivant est utilisé afin d'accéder aux données de l'API :

```
public class TopicService : InfosAPI
{
    #region Properties

    1 référence
    public string url
    {
        get
        {
            return base.baseUrl;
        }
        set
        {
            base.baseUrl = value;
        }
    }

    private HttpClient client;

    #endregion

    #region Constructor

    1 référence
    public TopicService()
    {
        client = new HttpClient { BaseAddress = new Uri(url) };
    }

    #endregion

    #region Methods

    /// <summary>
    /// Method to get all topics from DB
    /// </summary>
    /// <param name="connString"></param>
    /// <returns></returns>
    1 référence
    public async Task<List<Topics>> GetAllTopics()
    {
        var json = await client.GetStringAsync("topics");
        var topics = JsonConvert.DeserializeObject<List<Topics>>(json);

        return topics;
    }
}
```

```
15 références
public class Topics
{
    #region Properties

    2 références
    public int topic_id { get; set; }

    2 références
    public string name { get; set; }

    1 référence
    public List<Forums> forums { get; set; }

    #endregion

    #region Constructor

    3 références
    public Topics() { }

    0 références
    public Topics(int topic_id, string name)
    {
        this.topic_id = topic_id;
        this.name = name;
    }

    0 références
    public Topics(string name)
    {
        this.name = name;
    }

    #endregion
}
```

On fournit un http client au service et une "baseUrl" (adresse de base) qui est incluse dans le fichier de configuration de l'application, puis le service utilise la route mise à disposition pour l'API pour récupérer les données, les traiter et renvoyer une liste d'objets de type Topics.

DOSSIER PROFESSIONNEL^(DP)

Ce projet m'a permis d'acquérir la compétence du référentiel :

- Maquetter une application
- Développer une interface utilisateur de type Mobile
- Développer des composants d'accès aux données
- Développer la partie front-end d'une interface utilisateur
- Développer la partie back-end d'une interface utilisateur web

2. Précisez les moyens utilisés :

- Réunion avec les membres de SMOGON : discord.com
- Environnement de développement intégré : Visual Studio 2022 - Community Edition
- Gestion des tickets de développement: trello.com
- Editeur de maquette/wireframe : figma.com
- Editeur de MCD/MLD : diagrams.net
- Back-End : C# en .net 6.0
- Front-End : Multi-platform App UI
- Base de donnée : MySql
- Test API : Postman

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul pour le développement mais avec les propriétaires du site en tant que product owner lors des réunions de mise en place et avancée de projet

4. Contexte

Nom de l'entreprise, organisme ou association ▶ SMOGON University

Chantier, atelier, service ▶ Communication Web

Période d'exercice ▶ Du : 01/01/2023 au : En Cours

5. Informations complémentaires (facultatif)

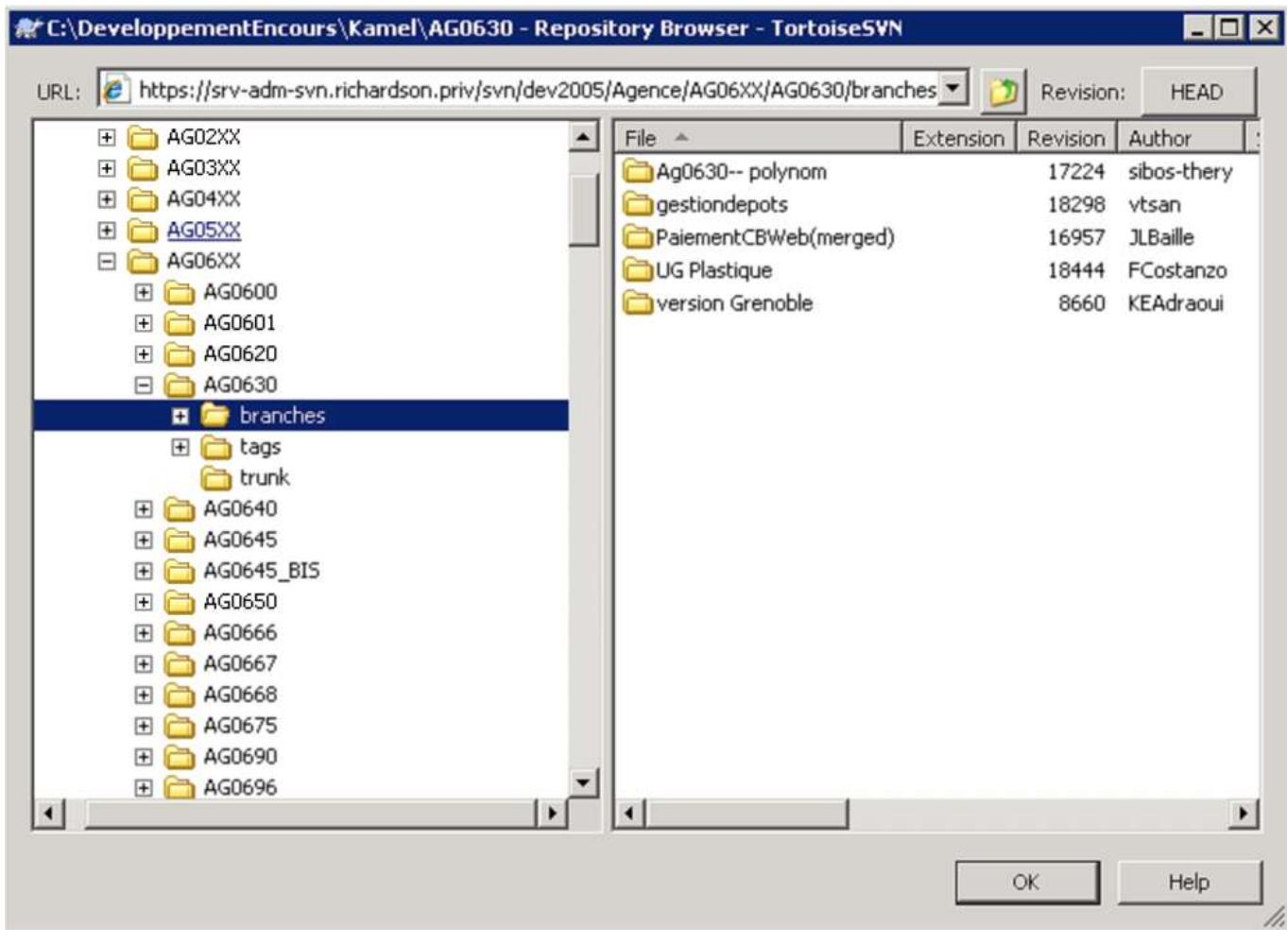
DOSSIER PROFESSIONNEL (DP)

Exemple n°2 ► AG0630 - Programme Agence Richardson

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Durant mon alternance, le besoin de l'entreprise d'ouvrir une nouvelle agence d'achat pour les produits plastiques pour l'intégralité du catalogue est apparu courant 2022. Après mise en place du projet et discussion avec la direction, il a été confié à l'équipe RF (Legacy), de rajouter à un projet existant, l'AG0630, la prise en charge de l'affichage, consultation et altération des données de l'UGA-Plastique (Unité de gestion Achat - plastique).

Ce projet étant fait en équipe, nous avons fonctionné avec un logiciel de versionning appelé tortoise-SVN dont voici une capture relative aux branches de développement du projet :



DOSSIER PROFESSIONNEL (DP)

On peut voir une trace des log de la branche main, appelée "trunk" :

The screenshot shows the TortoiseSVN Log Messages window. The title bar reads "C:\DeveloppementEncours\Kamel\AG0630 - Log Messages - TortoiseSVN". The window displays a list of log entries from revision 16455 down to 18442. The columns are: Revision, Actions, Author, Date, and Message. The message for revision 16455 is highlighted.

Revision	Actions	Author	Date	Message
18442	File	FCostanzo	mercredi 14 juin 2023 12:08:23	correction affichage palette prise en charge nouveau fichier emplzonage
18441	File	FCostanzo	mercredi 14 juin 2023 10:20:18	
17773	File	sdehecq	lundi 12 décembre 2022 08:44:53	correction code article bird
17746	File	sdehecq	mercredi 7 décembre 2022 16:13:35	Afficher la référence fournisseur du fi
17371	File	sibos-thery	mardi 12 juillet 2022 08:07:21	12/07/2022 correction erreur condition
17363	File	sibos-thery	lundi 11 juillet 2022 14:51:03	11/07/2022 force l'agence mere du pa
17321	File	sibos-thery	vendredi 24 juin 2022 11:29:55	ajout paramétrage pour ne pas appelle
17277	File	sibos-thery	mercredi 11 mai 2022 16:54:14	correction bug agence mere
17264	File	sibos-thery	jeudi 5 mai 2022 09:56:43	correction agm et pr pour polynom
17234	File	sibos-thery	lundi 11 avril 2022 09:36:04	11.04.2022 merge de la branche polyr
17193	File	marc	mercredi 23 mars 2022 14:35:41	article optionnel UGA
17191	File	sibos-thery	mercredi 23 mars 2022 10:47:23	23032022 Remise des champs uga pré
16956	File	JLBaille	lundi 6 décembre 2021 11:54:02	merge de la branche PaiementCBWeb
16455	File	KEAdrao...	mercredi 17 mars 2021 16:46:...	affichage de la qté calculée selon

Below the table, a large text area displays the detailed message for revision 16455:

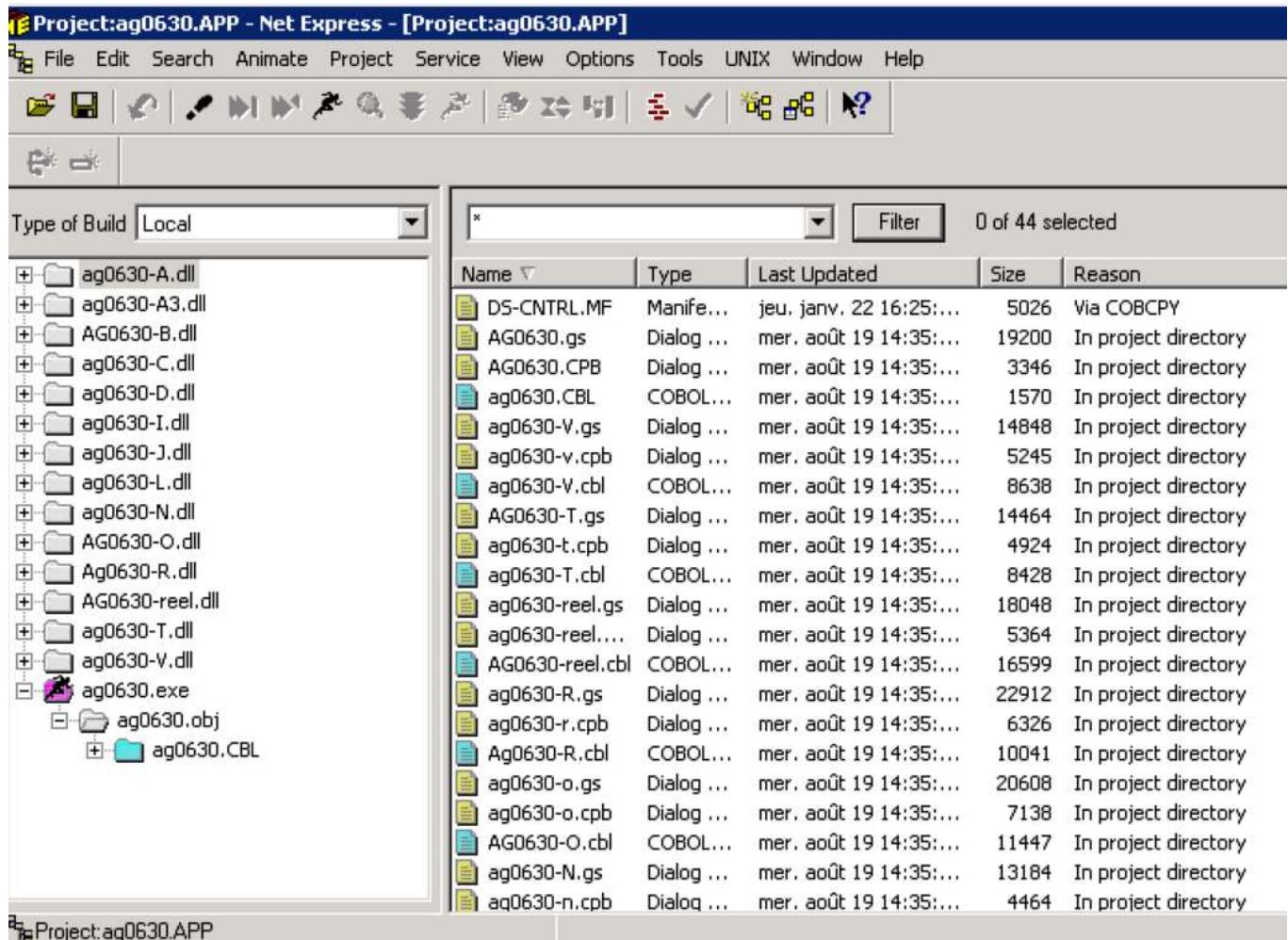
```
correction affichage palette
prise en charge nouveau fichier emplzonage
```

At the bottom of the window, there are several buttons and checkboxes:

- Path, Action, Copy from path, Revision buttons
- Show 89 revision(s), from revision 1378 to revision 18442 - 1 revision(s) selected, showing 2 changed paths
- Show only affected paths, Stop on copy/rename, Include merged revisions
- Statistics, Help, OK buttons
- Show All, Next 100, Refresh buttons

DOSSIER PROFESSIONNEL (DP)

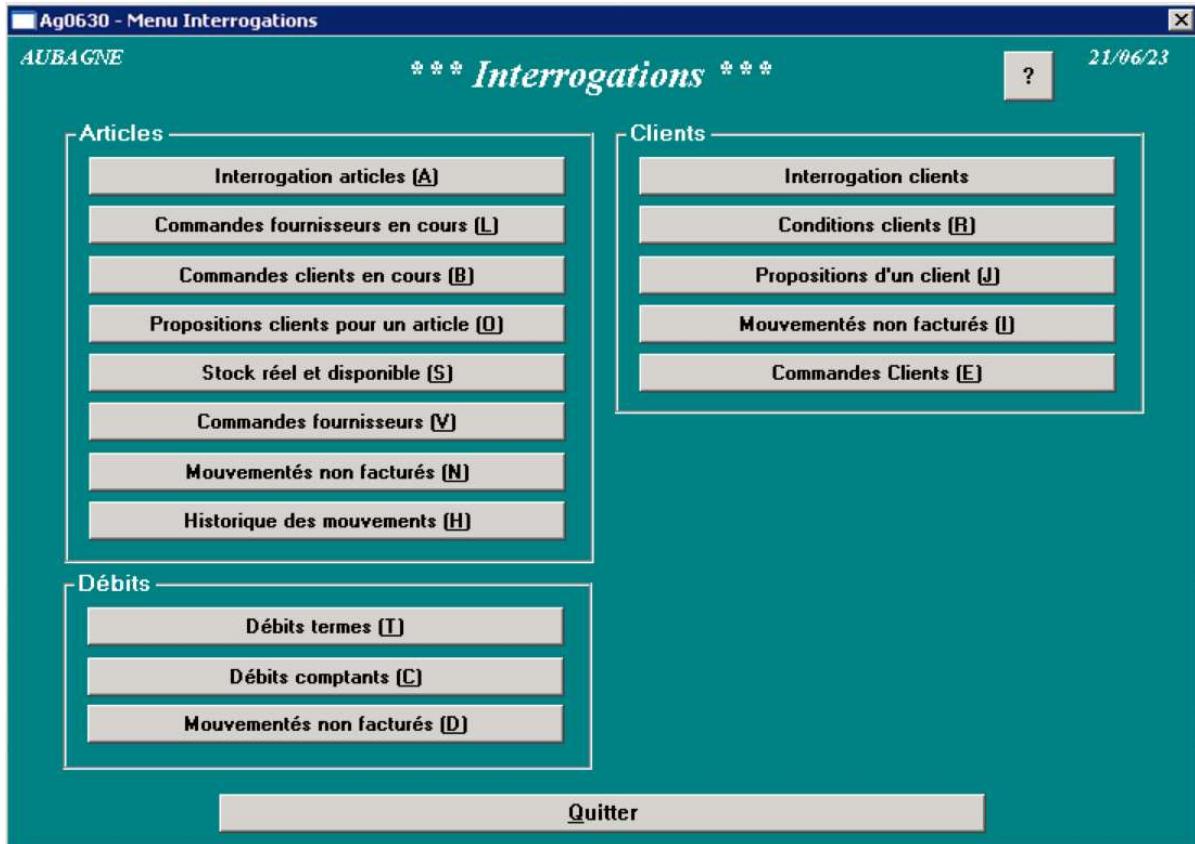
L'organisation des fichiers est faite sous forme d'un seul dossier avec un .app qui est le point d'entrée de l'application :



Les scripts qui nous intéressent sont l'AG0630-A.cbl et AG0630-A.gs, qui correspondent respectivement à la logique de calcul et l'affichage, soit le back-end et le front-end.

DOSSIER PROFESSIONNEL (DP)

Lors du démarrage l'AG0630 se présente comme suit :



Au démarage, on y trouve plusieurs choix qui correspondent à des sous-projets. Celui qui nous intéresse est le premier choix Interrogation articles (A).

Nous allons nous concentrer sur l'affichage du stock relatif à un article qui serait présent dans l'UGA-Plastique. Par soucis de confidentialité, la fonction ne sera montrée qu'en version debug, privée de toutes formes d'informations réelles.

DOSSIER PROFESSIONNEL (DP)

AG0630 - Interrogation Articles

***** Interrogation articles *****

X(12), Le 00/00/00

* Mode déconnecté * (F1, Double-Clic, Alt-A = Recherche Alpha)

Code article	X(13)	Rech. Alph.	Chiffre clé X(6)	Code accès	Panier
Service	9	X(12)	X(9)	Fiche de Contrôle	Nb. art. zzz9 Mt. Ht. zzzbz9.9(2)
Famille	X(3)	X(24)			Stocks
Fournisseur	X(7)	X(30)			Info zzzz9- zzzz9- zzzz9-
Groupe	X(3)	X(20)	X(1) A liquider		Réel zzzz9- zzzz9- zzzz9-
	X(90)				Dispo. zzzz9- zzzz9- zzzz9-
Désignation de l'article				UT	Créat.
X(45)				X(2)	99/99
X(20)		Prix Public Rich		TTC : zzzz9.99	
Prix Moy. Pond.	X(9)	PU. HT.	Date Maj Prix	Remise Prof.	Vétusté
z(5)9.99	z(5)9.99	z(5)9.99	z9/99/99	z9.99	z9
Meilleur prix d'achat : --- (Alt-P, Double-Clic = Détail de la chaîne de prix)					
Ventes 12mois	Moy. Mens.	Moy. Mens. Ecr.	Couv. théorique	Stock Sécur.	
z(8)9-	zzzzzz9-	zzzzz9-	zzzz9-	zzzz9	
(Alt-Q, Double-Clic = Détail des commandes fin.)					
- Quantité en commande	z(7)9.99- à partir du z9/99/99		Stock Prévis.		
(Direct usine exclue)					
Cond. fournisseur - Qté	zzz9.99	zzz9.99	- Libellé X(6)	Ventes UG	
(F1, Double-Clic, Alt-C = Recherche Alpha)					
Code client	X(5)	X(25)	Rech. Alph.		
<input type="checkbox"/> A potentiel					
Conditions client	29.99	29.99	Groupe X(3)	Prop. Fixe X(4)	
Condition Type 9999	Meilleure offre client existante		x(16)	x(15)	
Cdes Clients (B)	Propo. Clients (Q)	Cdes Fournis. (V)	Mouvtés Non Fact. (N)	Historique Mvts (H)	Quitter
Prix Net		Ensemble	Promo		
zz9.99		%MB	zz9.99	ZZZZZ9.99	
Tarif National					

Jusqu'à maintenant les deux zones encerclées ne permettait d'afficher que les valeurs relatives à l'UGA normale.

Comme dans les framework modernes, le framework DialogSystem offre des composant dont les champs sont identifiés par variable locale au composant et globale par rapport à l'intégralité de la fenêtre. Nous allons voir la variable locale, qui est reprise par la fenêtre globale.

DOSSIER PROFESSIONNEL (DP)

AG0630 - Interrogation Articles

***** Interrogation articles *****

X(12), Le 00/00/00

* Mode Déconnecté * (F1, Double-Clic, Alt-A = Recherche Alpha)

Code article X(13) Rech. Alph. Chiffre clé X(6) Code accès X(6) Panier Nb. art. zz9 Mt. Ht. zzzbz9.9[2]

Service	X(9)	X(12)
Famille	X(3)	X(24)
Fournisseur	X(7)	X(30)
Groupe	X(3)	X(20)
X(90)		

Désignation de l'article
X(45)
X(20)

Prix Moy. Pond. X(9) PU. HT
z(5)9.99 z(5)9.99 z(5)9.99

Meilleur prix d'achat : --- [Alt-P, Doub]

Ventes 12mois Moy. Mens. Moy. N
z(8)9- zzzzzzz9- zz9 [Alt-Q, Doub]

- Quantité en commande z(7)9.99- à partir du z9/99/99 Stock Prévis.
- (Direct usine exclue)

Cond. fournisseur - Qté zzz9.99 zzz9.99 - Libellé X(6) Ventes UG
[F1, Double-Clic, Alt-C = Recherche Alpha]

Code client X(5) X(25) Rech. Alph.
 A potentiel

Conditions client 29.99 29.99 Groupe X(3) Prop. Fixe X(4)

Condition Type 9999 Meilleure offre client existante z(16) z(15)

Cdes Clients (B) Propo. Clients (Q) Cdes Fournis. (V) Mouvtés Non Fact. (N) Historique Mvts (H) Quitter

Entry Field Properties

General Options More Options

Name EF56
Master field LIB-STK-UGA Master field...

Initial state Enabled Disabled
Picture string X(12)
Blank when zero Justified right

OK Connections... Cancel Help

Oct. Nou. Déc.
(*: Sorties du mois en cours)

Prix Net Ensemble Promo
zz9.99 zzzbz9.99 zzzbz9.99
%MB Tarif National

Ici le champ est donc LIB-STK-UGA et il correspond en COBOL à un PIC X(12), soit un 12 caractères alphanumériques. Lors de l'interrogation des fichiers indexés qui servent de base de donnée, si l'application trouve le code article dans le fichier qui liste l'intégralité des articles de l'UGA-Plastique, l'affichage sera alors différent. COBOL étant un langage expensif, montrer l'intégralité des tests, lectures et ouvertures de fichier semble trop volumineux aussi je me suis concentré sur le moment où le programme interroge le fichier des articles de l'UGA-Plastique.

DOSSIER PROFESSIONNEL (DP)

AFF-ART-UGPLAST.

```
MOVE SPACES TO UGA-NEG-LE2 .
MOVE ART-STK TO ART-UGAPLAST-OPT .
READ STOCKUGAPLASTOPT INVALID KEY
    MOVE SPACE TO LIB-STK-UGA *> UGA-NEG-LE2
    MOVE "-----" TO STK-UGA-LE2
    MOVE 0 TO SHOW-COND AFFICHE-BORDER
    GO TO SAFF-ART
END-READ .
```

Si l'article EST gérée en OPT a l'UGAchat ET NEG A

```
IF NEG-UGAPLAST-OPT = 1
    MOVE "STK R UGP : " TO LIB-STK-UGA *> UGA-NEG-LE2 .
    MOVE 1 TO SHOW-COND AFFICHE-BORDER
    MOVE CONDVTE-UGAPLAST-OPT TO COND-UGA-LE2
    MOVE 55 TO STK-METHODE
    MOVE ART-STK TO STK-PARAMS-55
    CALL "CALLDOTNET" USING STK-METHODE
        STK-PARAMS-55 STK-RESULTAT
    MOVE STK-PARAMS-55(1:9) TO STK-UGA-LE2
ELSE
    MOVE 0 TO SHOW-COND AFFICHE-BORDER
    MOVE SPACE TO LIB-STK-UGA .
```

Ce test étant le dernier de la liste, il détermine en fin de chaîne, si l'article existe ou non dans le catalogue. Le read sert comme dans les langages modernes à exécuter une action de lecture sur le fichier avec un renvoi de status du fichier (présent, absent, lisible, corrompu, etc). Si la clé est invalide, c'est à dire que le code article est absent du fichier, on met des espaces dans l'affichage que l'on avait ciblé plus haut et on met des "-----" dans la quantité de stock. Si le code négocié est à 1, on considère que l'article est présent et qu'il est disponible, et on affiche ainsi "STK R UGP" (qui était avant "STK R UGA") dans la variable d'affichage LIB-STK-UGA et le stock dans la boîte attenante.

DOSSIER PROFESSIONNEL^(DP)

Ce projet m'a permis d'acquérir la compétence du référentiel suivante :

Activité 1 :

- Développer une interface utilisateur de type Desktop
- Développer des composants d'accès aux données
- Développer la partie front-end d'une interface utilisateur
- Développer la partie back-end d'une interface utilisateur

Activité 3 :

- Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement

2. Précisez les moyens utilisés :

- Réunion avec les membres de SMOGON : discord.com
- Environnement de développement intégré : Visual Studio 2022 - Community Edition
- Gestion des tickets de développement: trello.com
- Editeur de maquette/wireframe : figma.com
- Editeur de MCD/MLD : diagrams.net
- Back-End : C# en .net 6.0
- Front-End : Multi-platform App UI
- Base de donnée : MySql
- Test API : Postman

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul pour le développement mais avec les propriétaires du site en tant que product owner lors des réunions de mise en place et avancée de projet

4. Contexte

Nom de l'entreprise, organisme ou association ▶ SMOGON University

Chantier, atelier, service ▶ Communication Web

Période d'exercice ▶ Du : 01/01/2023 au : En Cours

DOSSIER PROFESSIONNEL (DP)

Activité-type 2 Concevoir et développer la persistance des données

Exemple n° 1 ▶ Application Mobile SMOGON Forum

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Suite aux réunions relatives à l'aspect visuel et en parallèle aux décisions, j'ai commencé par choisir les technologies que j'allais utiliser. Mon but était d'offrir une application cross-platform simple d'installation, facile d'utilisation et surtout maintenable par la communauté.

Notre choix s'est porté sur C#, .net, SQL et MAUI pour les raisons suivantes :

- **C#** est un langage de programmation puissant et expressif, permettant de développer un code clair et maintenable.
- **MAUI** (Multi-platform App UI) est un framework qui permet de créer des interfaces utilisateur multiplateformes attrayantes et réactives, avec un code partagé entre les différentes plateformes.
- **SQL** est un système de gestion de base de données robuste et largement utilisé, offrant une grande flexibilité pour la gestion des données de l'application.
- La **communauté de SMOGON** est une communauté qui compte beaucoup de développeurs bénévoles et de néophytes habitués à bidouiller des applications desktop et des softs autour de l'univers du jeux-vidéo, qui sont très souvent en **C#**, le front est souvent en **WPF** et ils savent aussi souvent utiliser des **bases de données en SQL** (server ou mysql)

Nous avons ensuite établi les différentes fonctionnalités du site :

- **Comptes utilisateurs** : Les utilisateurs doivent pouvoir créer et gérer leurs propres comptes, y compris la configuration de leur profil et la mise à jour de leurs informations personnelles.
- **Fils de discussion et messages** : Le cœur de tout forum réside dans la possibilité de créer des fils de discussion (sujets) et de publier des messages (réponses) dans ces fils de discussion. Les utilisateurs doivent pouvoir créer de nouveaux fils de discussion et publier des messages, ainsi que modifier ou supprimer leurs propres messages.
- **Recherche** : Les utilisateurs doivent pouvoir rechercher des fils de discussion ou des messages spécifiques dans le forum.
- **Modération** : Des outils doivent être mis en place pour permettre aux modérateurs de gérer le contenu du forum, y compris la possibilité de supprimer ou modifier des messages, d'exclure des utilisateurs et d'établir des règles pour le forum.

DOSSIER PROFESSIONNEL^(DP)

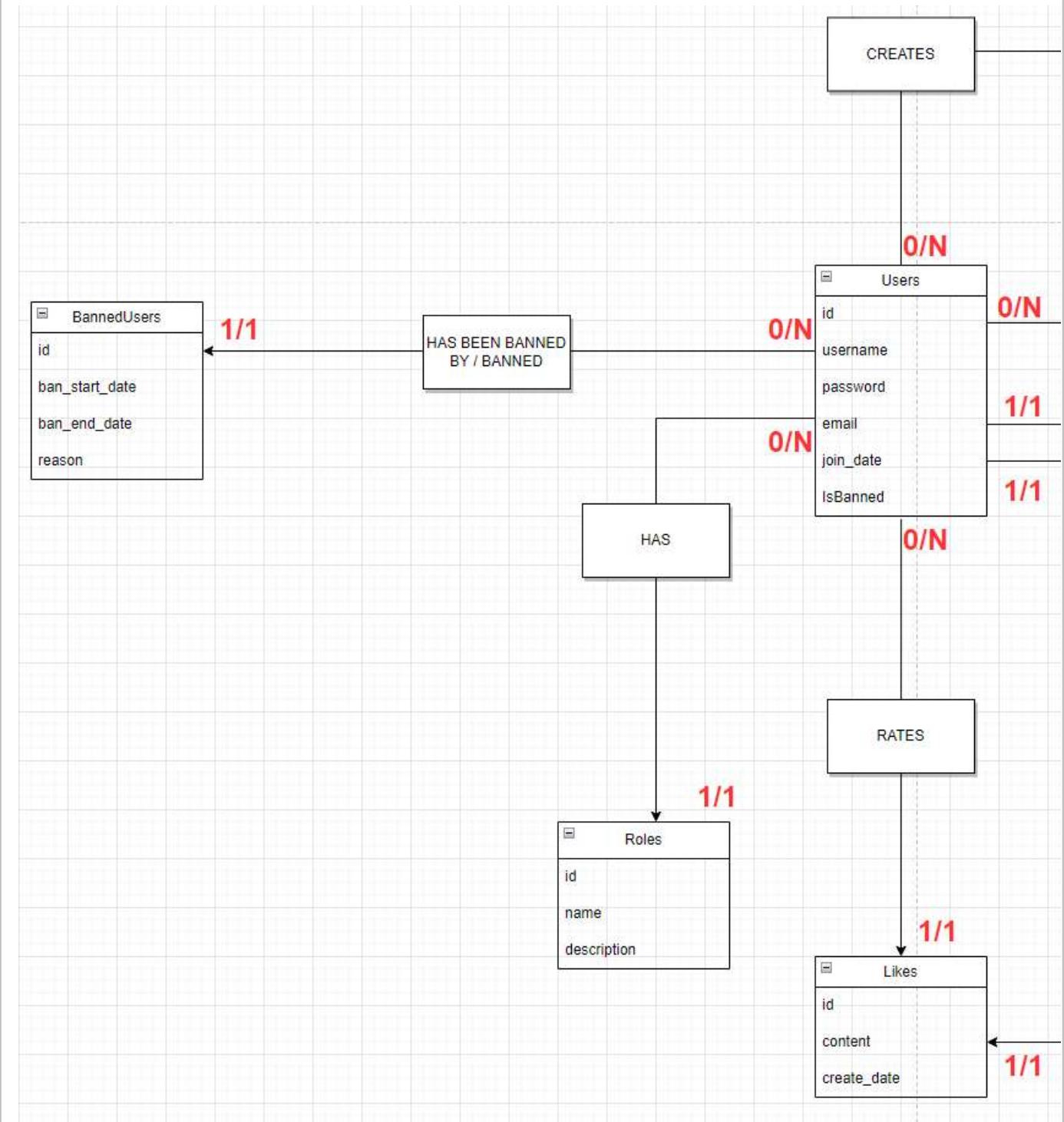
- **Notifications** : Les utilisateurs doivent pouvoir recevoir des notifications lorsqu'une personne répond à l'un de leurs messages ou les mentionne dans un fil de discussion.
- **Profils d'utilisateurs** : Chaque utilisateur doit disposer de sa propre page de profil qui affiche son activité sur le forum, y compris les fils de discussion et les messages qu'il a créés.
- **Pièces jointes d'images et de fichiers** : Les utilisateurs doivent pouvoir joindre des images et des fichiers à leurs messages.
- **Création d'équipe** : Les utilisateurs doivent pouvoir publier la composition de leur équipe à partir d'un formulaire de texte.
- **Évaluation des messages** : Les utilisateurs doivent pouvoir voter positivement pour les messages.

J'ai ensuite conçu les modèles de données en me basant sur ces besoins. Durant cette phase, J'ai mis en place les différents modèles, en m'appuyant et épurant le dictionnaire de données. Voici les différents modèles que j'ai réalisés dans l'ordre suivant :

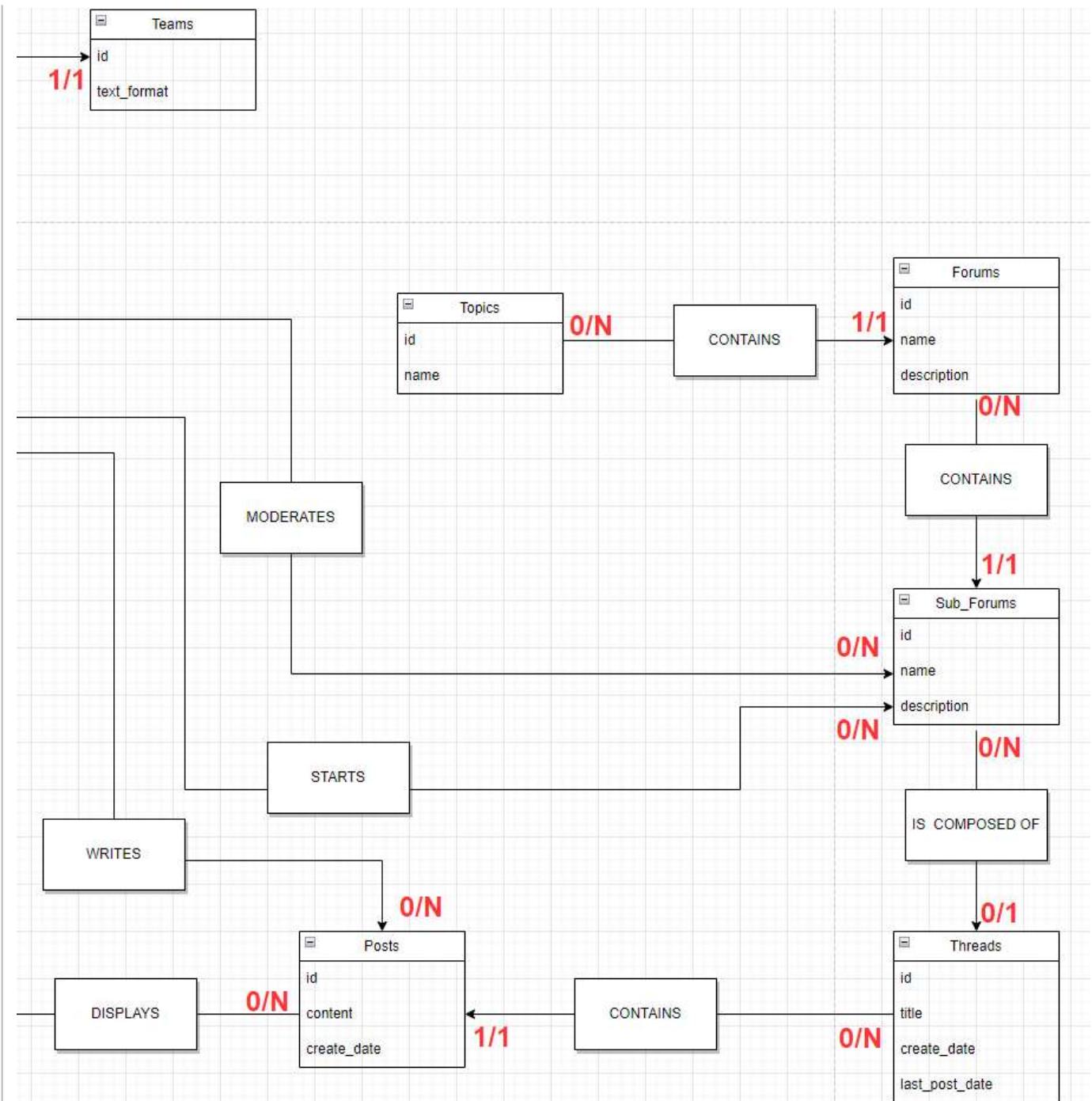
- Le modèle conceptuel de données (MCD) du type modèle entité-association. Qui permet de décrire le système d'information à l'aide d'entités.
- Modèle logique de données (MLD), qui consiste à décrire la structure selon laquelle les données seront stockées en dans la base de données.

DOSSIER PROFESSIONNEL (DP)

Pour le MCD :

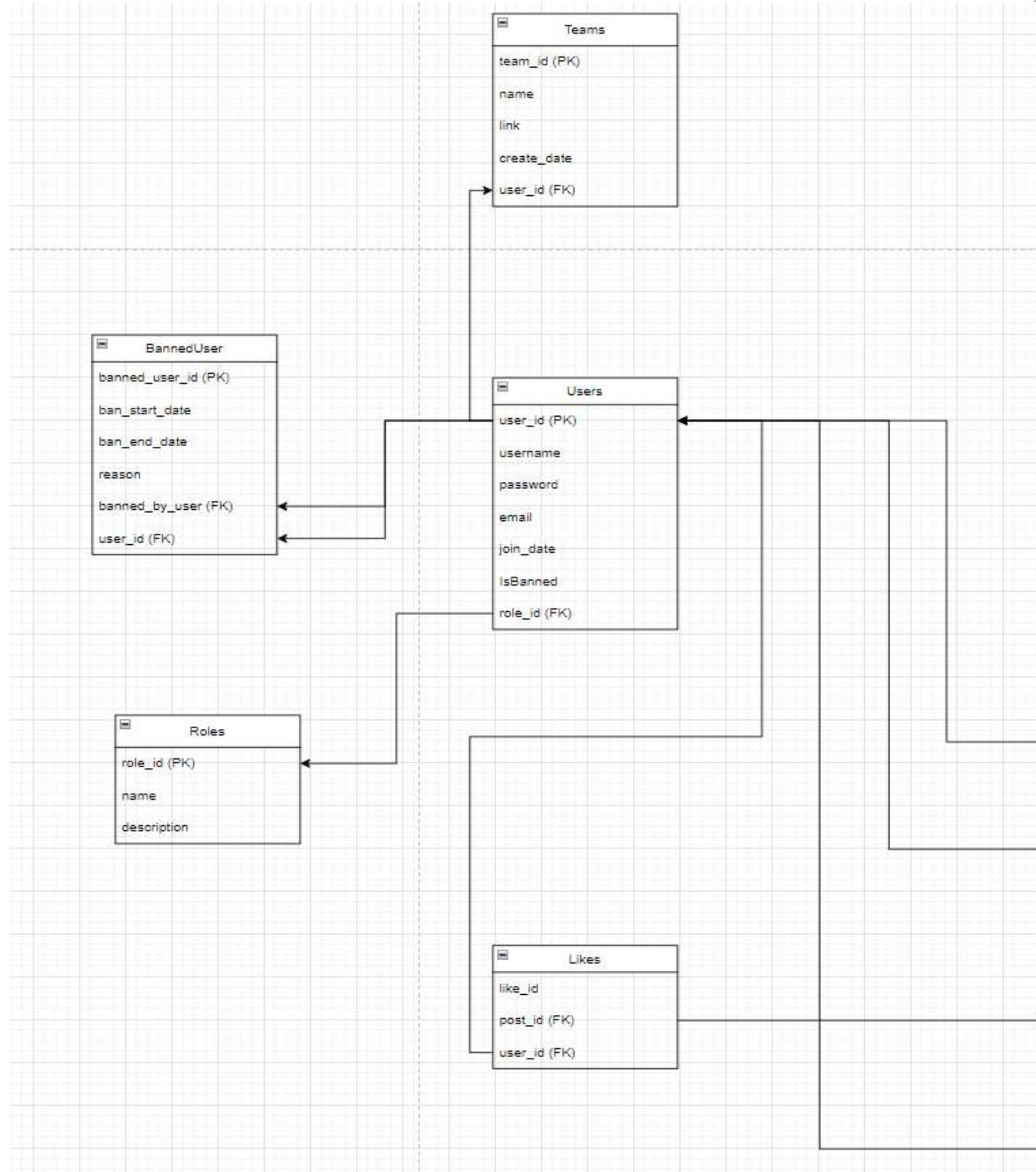


DOSSIER PROFESSIONNEL (DP)

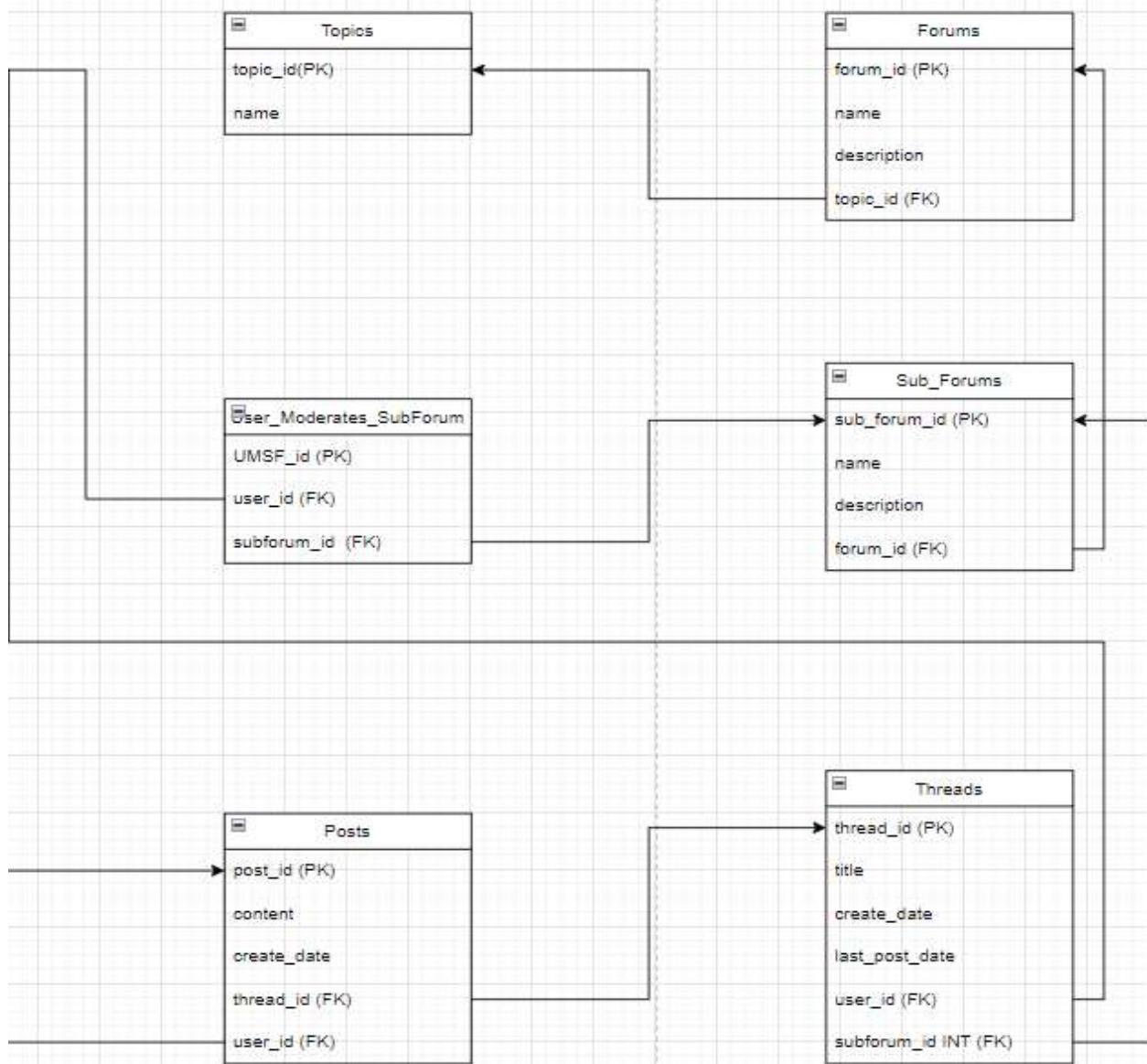


DOSSIER PROFESSIONNEL (DP)

Pour le MLD :



DOSSIER PROFESSIONNEL (DP)



DOSSIER PROFESSIONNEL (DP)

A la suite de ces conceptualisations, je me suis attelé à la rédaction du script de la base de données comme suite l'exemple suivant :

```
1 CREATE SCHEMA IF NOT EXISTS `Smogon_Forum` DEFAULT CHARACTER SET utf8mb4 ;
2
3 USE `Smogon_Forum` ;
4
5 -- Roles table
6 CREATE TABLE Roles (
7     role_id INT AUTO_INCREMENT NOT NULL,
8     name VARCHAR(255) NOT NULL,
9     description VARCHAR(255) NOT NULL,
10    PRIMARY KEY (role_id)
11 );
12
13 -- Users table
14 CREATE TABLE Users (
15     user_id INT AUTO_INCREMENT NOT NULL,
16     username VARCHAR(255) NOT NULL,
17     password VARCHAR(255) NOT NULL,
18     email VARCHAR(255) NOT NULL,
19     avatar_url VARCHAR(255),
20     join_date DATE NOT NULL,
21     isBanned BOOLEAN,
22     role_id INT NOT NULL,
23     PRIMARY KEY (user_id),
24     CONSTRAINT FK_Users_role_id_Roles FOREIGN KEY (role_id) REFERENCES Roles(role_id) ON DELETE CASCADE
25 );
26
27 -- Teams table
28 CREATE TABLE Teams (
29     team_id INT AUTO_INCREMENT NOT NULL,
30     name VARCHAR(255) NOT NULL,
31     link VARCHAR(255) NOT NULL,
32     date_created DATE NOT NULL,
33     user_id INT NOT NULL,
34     PRIMARY KEY (team_id),
35     CONSTRAINT FK_Teams_user_id_Users FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE
36 );
```

DOSSIER PROFESSIONNEL (DP)

Puis j'ai créé les modèles relatifs à ces tables (exemple Roles et Users) :

```
13 références
public class Roles
{
    #region Properties

    [Key]
    [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
    1 référence
    public int role_id { get; set; }
    2 références
    public string name { get; set; }
    2 références
    public string description { get; set; }

    0 références
    public List<Users> users { get; set; } = new List<Users>();

    #endregion

    #region Constructors

    4 références
    public Roles()
    {
        0 références
        public Roles(int role_id, string name, string description)
        {
            this.role_id = role_id;
            this.name = name;
            this.description = description;
        }
    }

    0 références
    public Roles(string name, string description)
    {
        this.name = name;
        this.description = description;
    }

    #endregion
}

public class Users
{
    #region Properties

    public int user_id { get; set; }
    public string username { get; set; }
    public string password { get; set; }
    public string email { get; set; }
    public DateTime join_date { get; set; }
    public string avatar_url { get; set; }
    public bool isBanned { get; set; }
    public int role_id { get; set; }
    public List<Posts> posts { get; set; }
    public List<Teams> teams { get; set; }

    #endregion

    #region Constructor

    3 références
    public Users()
    {
        0 références
        public Users(int user_id, string username, string password, string
        {
            this.user_id = user_id;
            this.username = username;
            this.password = password;
            this.email = email;
            this.join_date = join_date;
            this.avatar_url = avatar_url;
            this.role_id = role_id;
        }
    }
}
```

Ce projet m'a permis d'acquérir la compétence du référentiel :

- Concevoir une base de données
- Mettre en place une base de données
- Développer des composants dans le langage d'une base de données

DOSSIER PROFESSIONNEL^(DP)

2. Précisez les moyens utilisés :

- Réunion avec les membres de SMOGON : discord.com
- Environnement de développement intégré : Visual Studio 2022 - Community Edition
- Gestion des tickets de développement: trello.com
- Editeur de maquette/wireframe : figma.com
- Editeur de MCD/MLD : diagrams.net
- Back-End : C# en .net 6.0
- Front-End : Multi-platform App UI
- Base de donnée : MySql
- Test API : Postman

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul pour le développement mais avec les propriétaires du site en tant que product owner lors des réunions de mise en place et avancée de projet.

4. Contexte

Nom de l'entreprise, organisme ou association ► SMOGON University

Chantier, atelier, service ► Communication Web

Période d'exercice ► Du : 01/01/2023 au : En Cours

5. Informations complémentaires (*facultatif*)

DOSSIER PROFESSIONNEL (DP)

Activité-type

Concevoir et développer une application multicouche répartie

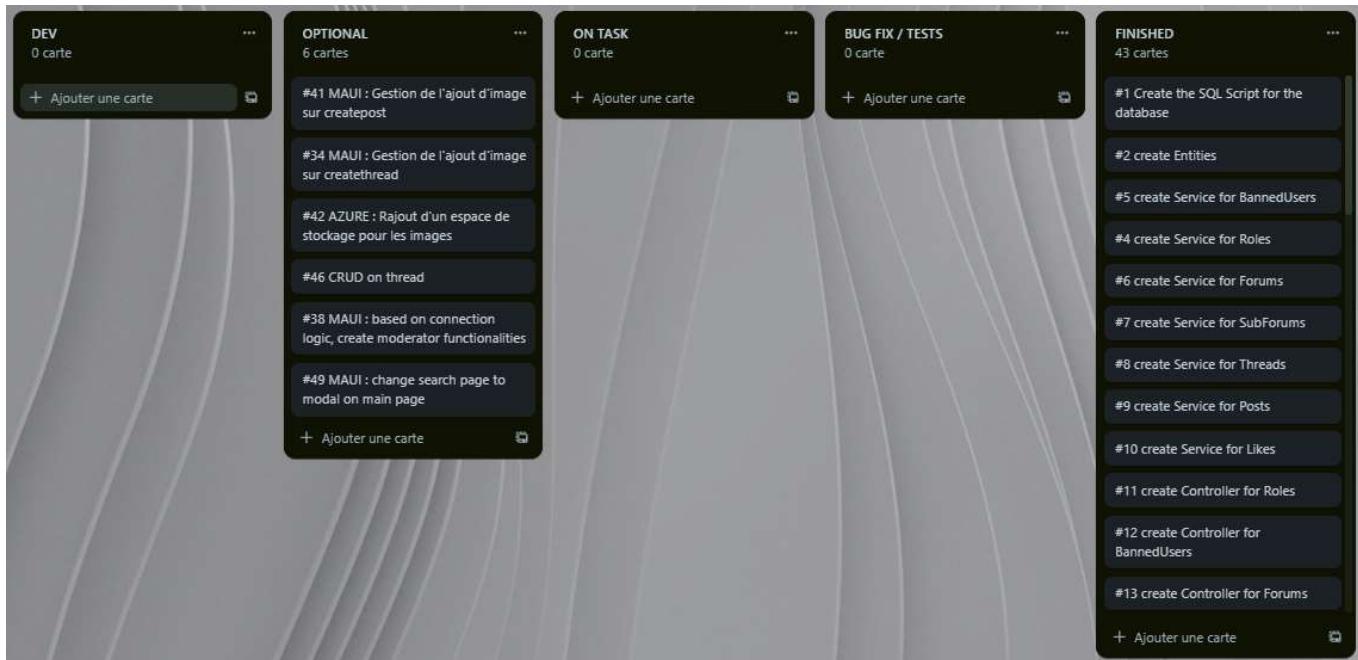
Exemple n° 1 ▶ Application Mobile SMOGON Forum

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Comme expliqué plus haut, le choix des technologies s'est fait sur la base des connaissances et pratiques des divers membres de la communauté SMOGON. De plus, il a été décidé que dans l'optique d'une **maintenabilité** et d'une **lisibilité du projet**, on utiliserait le **minimum possible de librairies**, et, que l'on utiliserait une **norme pour l'écriture du code** que je vais détailler dans ce chapitre.

Pour que les "products owner" puissent avoir une vision du projet, il a été versionné sur Github et nous avons mis en place un espace de travail trello qui permettait la suivie du projet et des tickets.

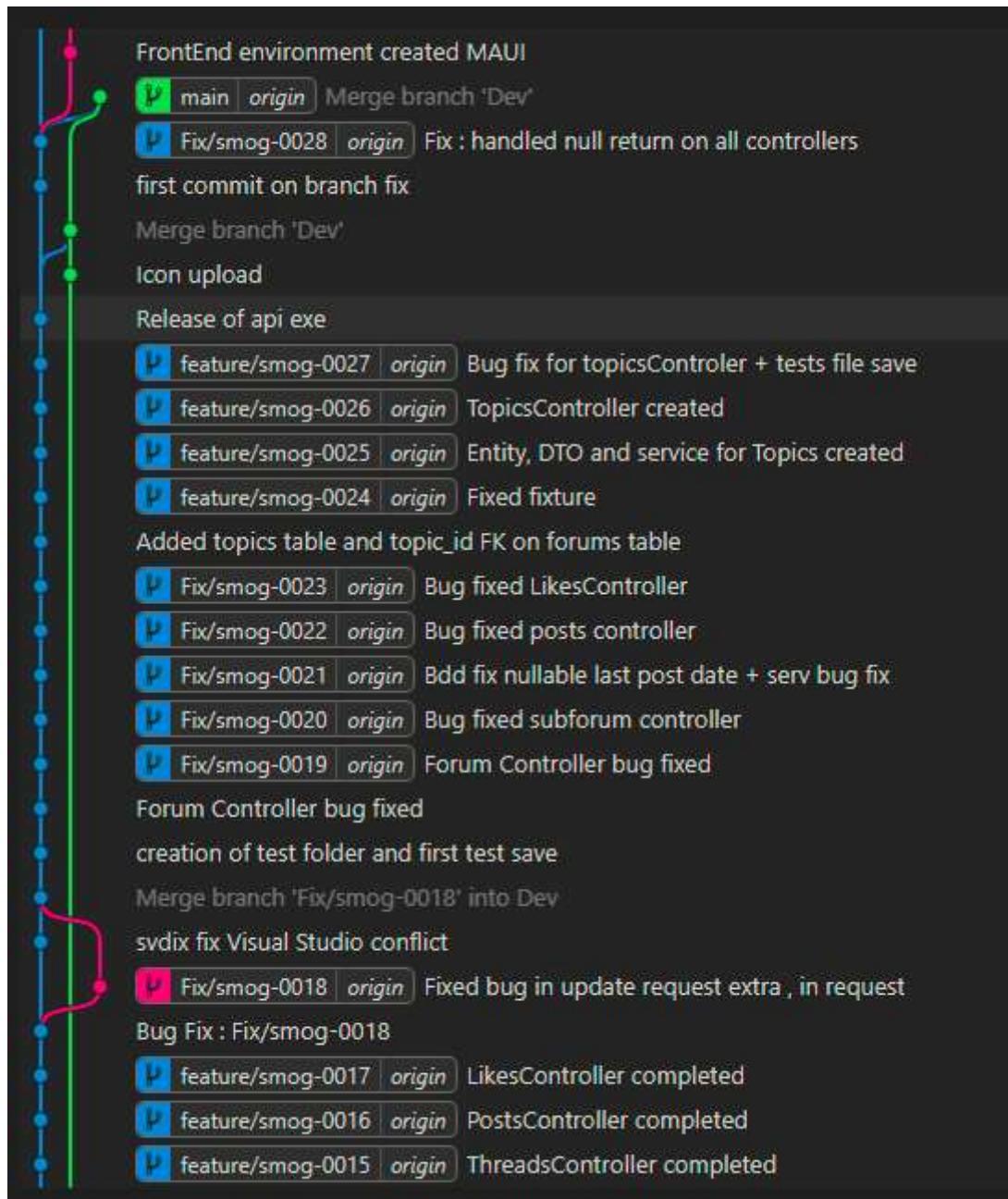
Voici une capture d'écran de la page trello à la fin de la phase 1 de développement :



Chacun des tickets porte un numéro qui correspond à une fonctionnalité, qui correspond à une branche de développement sur Github.

DOSSIER PROFESSIONNEL (DP)

Ici, un exemple du fil de l'avancée du projet grâce à une visualisation par branche :



On retrouve les numéros de tickets, dans les cadres, et les commentaires servent à comprendre la tâche accomplie.

DOSSIER PROFESSIONNEL (DP)

Nous avons opté pour l'utilisation d'une API REST (Representational State Transfer), car elles offrent une architecture simple, basée sur les standards du web tels que HTTP et JSON. Elles permettent une communication efficace entre différentes parties de l'application, facilitent l'indépendance des déploiement, favorisent la flexibilité et l'évolutivité. Elles permettent de développer des applications modulaires et réutilisables, tout en suivant des bonnes pratiques de conception et de développement. En résumé, elles sont une approche fiable, largement adoptée et adaptée aux besoins actuels de développement d'applications.

Ainsi nous avons établi les routes suivantes pour cartographier l'intégralité des fonctionnalités :

- Utilisateurs :

GET /api/users	# Récupère la liste de tous les utilisateurs
GET /api/users/:id	# Récupère l'utilisateur par ID
POST /api/users	# Crée un nouvel utilisateur
PUT /api/users/:id	# Met à jour un utilisateur existant
DELETE /api/users/:id	# Supprime un utilisateur
POST /api/users/:id/login	# Connecte un utilisateur
GET /api/users/:id/threads	# Récupère tous les fils de discussion par ID utilisateur
GET /api/users/:id/posts	# Récupère tous les messages créés par ID utilisateur

- Rôles :

GET /api/roles	# Récupère la liste de tous les rôles
GET /api/roles/:id	# Récupère le rôle par ID
POST /api/roles	# Crée un nouveau rôle
PUT /api/roles/:id	# Met à jour un rôle existant
DELETE /api/roles/:id	# Supprime un rôle
GET /api/roles/:id/users	# Récupère tous les utilisateurs par ID de rôle

- Forums :

GET /api/forums	# Récupère la liste de tous les forums
GET /api/forums/:id	# Récupère le forum par ID
POST /api/forums	# Crée un nouveau forum
PUT /api/forums/:id	# Met à jour un forum existant
DELETE /api/forums/:id	# Supprime un forum
GET /api/forums/:id/subforums	# Récupère tous les sous-forums par ID de forum
GET /api/forums/:id/threads	# Récupère tous les fils de discussion par ID de forum

DOSSIER PROFESSIONNEL (DP)

- Sous-Forum :

GET /api/subforums # Récupère la liste de tous les sous-forums
GET /api/subforums/:id # Récupère le sous-forum par ID
POST /api/subforums # Crée un nouveau sous-forum
PUT /api/subforums/:id # Met à jour un sous-forum existant
DELETE /api/subforums/:id # Supprime un sous-forum
GET /api/subforums/:id/threads # Récupère tous les fils de discussion par ID de sous-forum

- Sujets:

GET /api/threads # Récupère la liste de tous les fils de discussion
GET /api/threads/:id # Récupère le fil de discussion par ID
POST /api/threads # Crée un nouveau fil de discussion
PUT /api/threads/:id # Met à jour un fil de discussion existant
DELETE /api/threads/:id # Supprime un fil de discussion
GET /api/threads/:id/posts # Récupère la liste de tous les messages d'un fil de discussion

- Posts :

GET /api/posts # Récupère la liste de tous les messages
GET /api/posts/:id # Récupère le message par ID
POST /api/posts # Crée un nouveau message
PUT /api/posts/:id # Met à jour un message existant
DELETE /api/posts/:id # Supprime un message
GET /api/posts/:id/likes # Récupère la liste de tous les likes d'un message

- Likes :

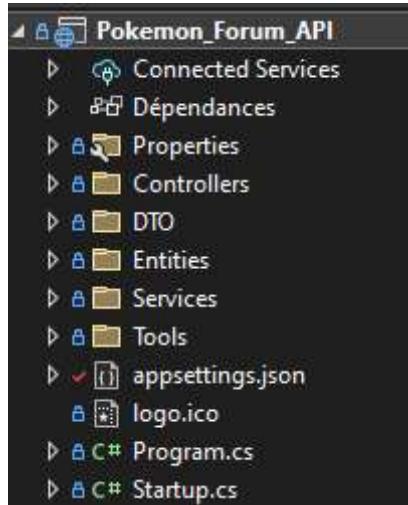
GET /api/likes/:id # Récupère le like par ID
POST /api/posts/:id/likes # Permet à un utilisateur d'aimer un message
DELETE /api/posts/:id/likes/:like_id # Permet à un utilisateur de ne plus aimer un message

- Utilisateurs Bannis:

GET /bannedusers # Récupère tous les utilisateurs bannis
GET /bannedusers/:id # Récupère un utilisateur banni spécifique par ID
POST /bannedusers # Crée un nouvel utilisateur banni
PUT /bannedusers/:id # Met à jour un utilisateur banni existant
DELETE /bannedusers/:id # Supprime un utilisateur banni existant

DOSSIER PROFESSIONNEL (DP)

Nous avons choisi l'architecture suivante le projet en microservices comme suit :



Nous allons nous focaliser sur l'aspect connexion de l'utilisateur pour mettre en exergue plusieurs aspects fondamentaux du développement.

Le modèle de base pour un utilisateur est comme suit :

```
57 références
public class Users
{
    #region Properties

    [Key]
    [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
    public int user_id { get; set; }
    13 références
    public string username { get; set; }
    7 références
    public string password { get; set; }
    6 références
    public string email { get; set; }
    4 références
    public DateTime join_date { get; set; }
    3 références
    public string avatar_url { get; set; }
    4 références
    public bool isBanned { get; set; }
    5 références
    public int role_id { get; set; }
    1 référence
    public List<Posts> posts { get; set; }
    1 référence
    public List<Teams> teams { get; set; }

    #endregion

    #region Constructor

    4 références
    public Users(){}
    6 références
    public Users(int user_id, string username, string password, string email, Da
    {
        this.user_id = user_id;
        ...
    }
}
```

DOSSIER PROFESSIONNEL (DP)

Ensuite le contrôleur relatif à la route User et nous allons nous focaliser sur la fonction de connexion :

```
[ApiController]
[Route("/users")]
1 référence
public class UsersController : ControllerBase
{
    string connectionString = Tools.Tools.connectionString;
    //string connectionString = Utils.ConnectionString;
    //string connectionString = $"Server=127.0.0.1;User ID=root;Password=;Database=saigon_forum;";
    UserService userService = new UserService();

    0 références
    public UsersController() {}

    [HttpGet]
    0 références
    public async Task<ActionResult<List<Users>>> GetAllUsers()
    {
        var users = await userService.GetAllUsers(connectionString);
        if (users == null)
            return BadRequest("An error occurred while getting all the users. Please check your request and try again.");
        return Ok(users);
    }
}
```

```
[HttpPost("login")]
0 références
public async Task<IActionResult> Login([FromBody] UserDtoLogin userDto)
{
    var token = await userService.LoginJWT(connectionString, userDto.username, userDto.password);
    if (token == null)
    {
        return Unauthorized();
    }

    return Ok(new { token = new JwtSecurityTokenHandler().WriteToken(token) });
}
```

On peut voir sur le screenshot ci-dessus que la méthode de login utilise le principe de token JWT, que je vais détailler un peu plus bas pour parler d'un aspect sécurité.

L'objet permettant l'accès aux données (nous avons choisi de différencier la classe de base du DTO pour une question de compréhension pour les néophytes) :

```
1 référence
public class UserDtoLogin
{
    [Required]
    [StringLength(20, MinimumLength = 3, ErrorMessage = "username length must be between 3 and 20")]
    1 référence
    public string username { get; set; }
    [Required]
    [StringLength(20, MinimumLength = 8, ErrorMessage = "password length must be between 8 and 20")]
    1 référence
    public string password { get; set; }
}
```

Une première couche de limitation est imposée par le DTO et une prise en charge d'erreur.

DOSSIER PROFESSIONNEL (DP)

Ensuite, le service relatif à l'utilisateur :

```
19 références
public class UserService
{
    string connectionString = Tools.Tools.connectionString;
    9 références
    public UserService() {}

    /// <summary>
    /// Method to login User using JWT tokens
    /// </summary>
    /// <param name="connString"></param>
    /// <param name="_username"></param>
    /// <param name="_password"></param>
    /// <returns></returns>
    1 référence
    public async Task<SecurityToken> LoginUserJWT(string connString, string _username, string _password)
    {
        Users user = new Users();

        try
        {
            using (MySqlConnection conn = new MySqlConnection(connString))
            using (MySqlCommand cmd = new MySqlCommand("SELECT * FROM users where username=@username", conn))
            {
                await conn.OpenAsync();
                cmd.Parameters.AddWithValue("@username", _username);

                using (var reader = await cmd.ExecuteReaderAsync())
                {
                    while (await reader.ReadAsync())
                    {
                        int id = reader.GetInt32(0);
                        string username = reader.GetString(1);
                        string password = reader.GetString(2);
                        string email = reader.GetString(3);
                        DateTime join_date = reader.GetDateTime(5);
                        string avatar_url = reader.IsDBNull(4) ? "www.exemple.fr/imagedefouf" : reader.GetString(5);
                        bool isBanned = reader.GetBoolean(6);
                        int role_id = reader.GetInt32(7);

                        if (BCrypt.Net.BCrypt.Verify(_password, password))
                        {
                            user = new Users(id, username, password, email, join_date, avatar_url, role_id, isBanned);
                        }
                        else
                        {
                            return null;
                        }
                    }
                }
            }
        }
    }
}
```

La fonction admet trois paramètres: le chaînon de connexion, et deux chaînes de caractères relatifs au mot de passe et au nom d'utilisateur. Pour pallier aux injections SQL, les requêtes sont préparées puis exécutées avec les paramètres établis.

Parmis le peu de librairies utilisées nous avons choisi la librairie Bcrypt pour la prise en charge de la vérification des token JWT(JSON Web Tokens), qui sont utilisés pour l'authentification et l'autorisation dans les applications web et mobiles.

DOSSIER PROFESSIONNEL (DP)

Les tokens JWT offrent une sécurité garantie par leur signature numérique et leur capacité à contenir toutes les informations nécessaires pour vérifier l'identité et les autorisations d'un utilisateur. Ils permettent une transmission efficace via divers moyens tels que les URL, les en-têtes HTTP ou les cookies. Les tokens JWT peuvent être configurés avec une durée de validité pour renforcer la sécurité. Grâce à leur auto-validation, ils améliorent les performances et facilitent le passage à l'échelle sans nécessiter des consultations constantes d'une base de données. Les tokens JWT sont compatibles avec diverses technologies et peuvent être personnalisés pour inclure des informations spécifiques à l'application.

Voici la deuxième partie de la fonction :

```
if (user.user_id != 0)
{
    //Getting the key from app setting
    IConfigurationBuilder builder = new ConfigurationBuilder()
        .SetBasePath(Directory.GetCurrentDirectory())
        .AddJsonFile("appSettings.json", optional: true, reloadOnChange: true);

    var configuration = builder.Build();

    JwtSettings jwtSettings = configuration.GetSection("Jwt").Get<JwtSettings>();

    var key = Encoding.ASCII.GetBytes(jwtSettings.Key);

    //Token shaping
    var tokenDescriptor = new SecurityTokenDescriptor
    {
        Subject = new ClaimsIdentity(new Claim[]
        {
            new Claim("User_id", user.user_id.ToString()),
            new Claim("Role_id", user.role_id.ToString())
        }),
        Expires = DateTime.UtcNow.AddDays(90),
        SigningCredentials =
            new SigningCredentials( new SymmetricSecurityKey(key), SecurityAlgorithms.HmacSha256Signature ),
        Issuer = jwtSettings.Issuer,
        Audience = jwtSettings.Audience
    };

    var tokenHandler = new JwtSecurityTokenHandler();
    var token = tokenHandler.CreateToken(tokenDescriptor);

    return token;
}
else
{
    return null;
}
}
catch (Exception ex)
{
    return null;
}
```

DOSSIER PROFESSIONNEL (DP)

Cette partie sert à créer le token lors de l'authentification de l'utilisateur pour lui faciliter la connexion et l'interaction avec l'application. En cas de retour null, un message est alors envoyé par l'API pour dire que la connexion n'a pas pu être établie.

A l'issue du développement, nous avons écrit des scripts de test à utiliser sur l'application postman dont voici un extrait :

```
{
  "info": {
    "_postman_id": "049307ed-187d-4c9d-91b6-9171783bc96e",
    "name": "TESTS",
    "schema": "https://schema.getpostman.com/json/collection/v2.1.0/collection.json",
    "_exporter_id": "21702646"
  },
  "item": [
    {
      "name": "/users",
      "item": [
        {
          "name": "Get all users",
          "event": [
            {
              "listen": "test",
              "script": {
                "exec": [
                  ""
                ],
                "type": "text/javascript"
              }
            }
          ],
          "protocolProfileBehavior": {
            "disableBodyPruning": true
          },
          "request": {
            "method": "GET",
            "header": [],
            "body": {
              "mode": "raw",
              "raw": "{\r\n  \"name\" : \"aurelie\",\r\n  \"age\" : 25\r\n}",
              "options": {
                "raw": {
                  "language": "json"
                }
              }
            }
          }
        }
      ]
    }
  ]
}
```

Il suffisait alors à divers utilisateurs de renseigner l'adresse de l'api et de lancer les test puis de retourner les fichiers de log produit suite aux test.

DOSSIER PROFESSIONNEL^(DP)

La publication de l'API s'est fait en deux phases :

Nous avons tout d'abord opté pour une publication sur azure de part la simplicité de l'interface de publication de Visual Studio, mais il s'est avéré que le coût était farameineux (+ de 200€ pour 1 semaine d'hébergement).

Puis nous avons choisi de prendre un serveur virtuel privé (VPS) avec la société IONOS. La publication de l'API s'est révélée tout autant facile grâce à l'interface Plesk. Cependant, au lieu de ne cliquer que sur un seul bouton, il nous faut compiler l'application puis la publier sur le site.

Ce projet m'a permis d'acquérir la compétence du référentiel :

- Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement
- Concevoir une application
- Développer des composants métier
- Construire une application organisée en couches
- Développer une application mobile
- Préparer et exécuter les plans de tests d'une application
- Préparer et exécuter le déploiement d'une application

2. Précisez les moyens utilisés :

- Réunion avec les membres de SMOGON : discord.com
- Environnement de développement intégré : Visual Studio 2022 - Community Edition
- Gestion des tickets de développement: trello.com
- Editeur de maquette/wireframe : figma.com
- Editeur de MCD/MLD : diagrams.net
- Back-End : C# en .net 6.0
- Front-End : Multi-platform App UI
- Base de donnée : MySql
- Test API : Postman

3. Avec qui avez-vous travaillé ?

DOSSIER PROFESSIONNEL^(DP)

J'ai travaillé seul pour le développement mais avec les propriétaires du site en tant que product owner lors des réunions de mise en place et avancée de projet.

4. Contexte

Nom de l'entreprise, organisme ou association ► SMOGON University

Chantier, atelier, service ► Communication Web

Période d'exercice ► Du : 01/01/2023 au : En Cours

5. Informations complémentaires (*facultatif*)

DOSSIER PROFESSIONNEL (DP)

Déclaration sur l'honneur

Je soussigné(e) Franck Costanzo

déclare sur l'honneur que les renseignements fournis dans ce dossier sont exacts et que je suis
l'auteur des réalisations jointes.

Fait à Marseille

le 21/06/2023

pour faire valoir ce que de droit.

Signatu

