

2025

# API et ASP.NET Core

## Web API

E-GESCOM  
MATEO SAGE

PRO A PRO | 1419 Av. d'Italie, 82000 Montauban, France

## Table des matières

Introduction .....	2
Contexte du stage .....	2
Présentation de l'entreprise .....	2
Objectifs du projet .....	2
I. Architecture et Technologies Utilisées .....	3
Backend : API Web ASP.NET .....	3
Organisation du projet .....	3
Gestion de la base de données (DbContext) .....	12
Authentification et autorisation (BearerAuthorizationMiddleware.cs) .....	13
Exposition des Services (WS) .....	15
Structure des endpoints de l'API REST .....	16
Exemples d'endpoints GET .....	16
Connexion et Déploiement .....	19
Tunnel SSH pour l'accès et le port forwarding .....	19
Configuration des URLs des services .....	19
II. Tests .....	20
Résultats .....	20
Conclusion .....	24
Bilan du projet .....	24
Compétences acquises .....	24

# Introduction

## Contexte du stage

Durant mon stage, j'ai été chargé d'établir un API interne du site E-Gescom de l'entreprise "Pro à Pro", en migrant l'intégralité (commencer), existante du backend d'une architecture PHP vers une API Web ASP.NET RESTful en C#, en adoptant une architecture MVC pour les employés internes.

## Présentation de l'entreprise

Pro à Pro est un acteur majeur de la **distribution alimentaire pour les professionnels de la Restauration Hors Domicile (RHD)** en France. Filiale du groupe METRO, l'entreprise se positionne comme un grossiste alimentaire qui accompagne et livre quotidiennement des milliers de clients variés, incluant des établissements de santé, scolaires, d'entreprise, des maisons de retraite, ainsi que des restaurants traditionnels.

Elle propose une large gamme de produits couvrant l'épicerie, le frais, le surgelé, les boissons et les produits non-alimentaires, s'engageant à livrer ses clients sur l'ensemble du territoire français, souvent en 24h. Pro à Pro met en avant une dimension humaine et des valeurs fortes, favorisant l'intégration et l'épanouissement de ses collaborateurs.

Durant mon stage, j'ai intégré la **Direction des Systèmes d'Information (DSI)**, spécifiquement au sein du service de **Développement Web**. Ce service, crucial pour l'évolution et la performance des outils internes de l'entreprise, était placé sous la responsabilité de **Madame Yurelis**, qui m'a supervisé durant ma période où j'étais présent.

## Objectifs du projet

L'objectif principal de ce projet de stage était de **concevoir, développer et livrer une API Web RESTful en ASP.NET Core pour le site interne "E-Gescom" de Pro à Pro**. Cette nouvelle API visait à moderniser et à optimiser la gestion de l'ensemble des processus liés au secteur **agro-alimentaire** de l'entreprise, en offrant une interface de communication performante et sécurisée pour les applications internes utilisées par les employés.

**Migrer l'API existante** d'une technologie PHP vers un environnement C#/.NET, garantissant ainsi une meilleure maintenabilité, évolutivité et performance.

**Structurer l'API** selon les principes de l'architecture MVC, en développant des Modèles, des Contrôleurs et des Repositories robustes.

**Sécurité** (via authentification par jeton Bearer) pour permettre aux différents employés internes de Pro à Pro puissent interagir avec les données.

# I. Architecture et Technologies Utilisées

## Backend : API Web ASP.NET

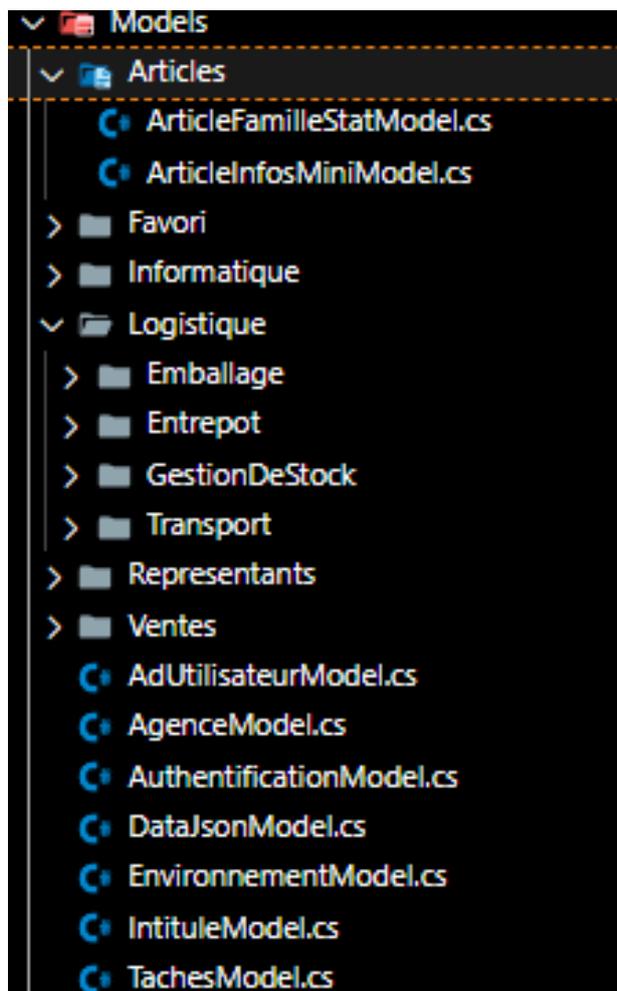
L'API a été développée en **ASP.NET Core**, un framework moderne et performant de Microsoft, choisi pour sa robustesse, sa flexibilité et sa capacité à construire des services web RESTful efficaces. Ce choix a permis de bénéficier de l'écosystème .NET et de ses nombreux avantages en termes de développement et de maintenance par rapport à l'architecture PHP préexistante.

Le projet a été organisé selon les principes de l'architecture **MVC (Model-View-Controller)**.

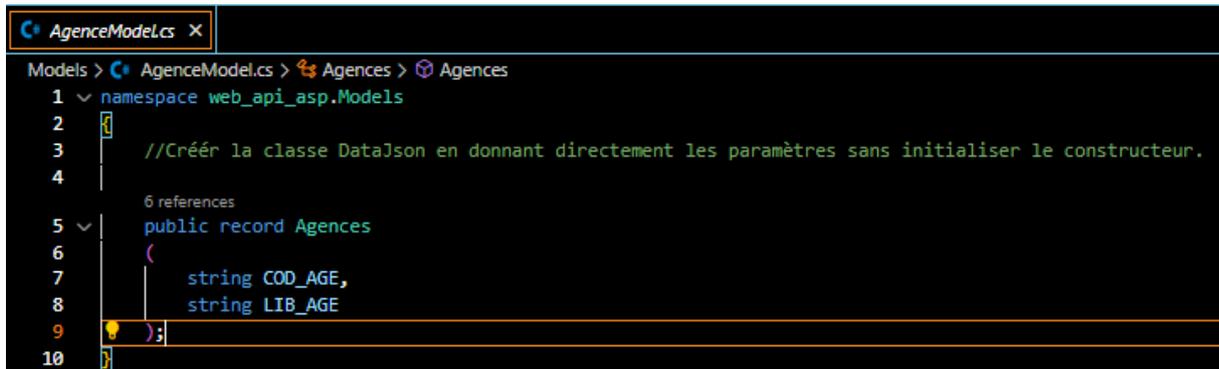
### Organisation du projet

#### Modèles (Models) :

Représentent la structure de données de l'application et les entités métier. Ils reflètent la base de données. Ils peuvent contenir des validations simples.

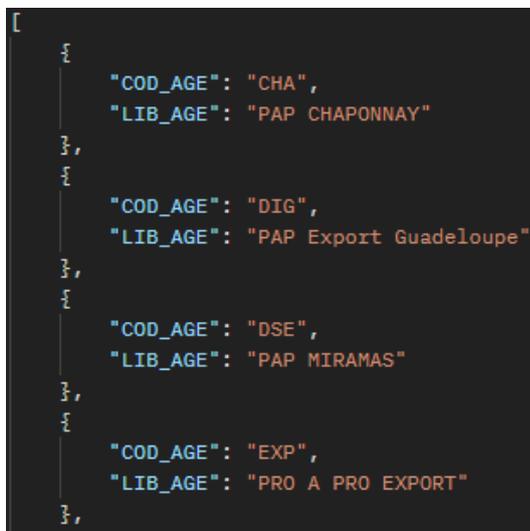


Exemple de modèles (AgenceModel.cs, AuthenticationModel.cs, etc.)

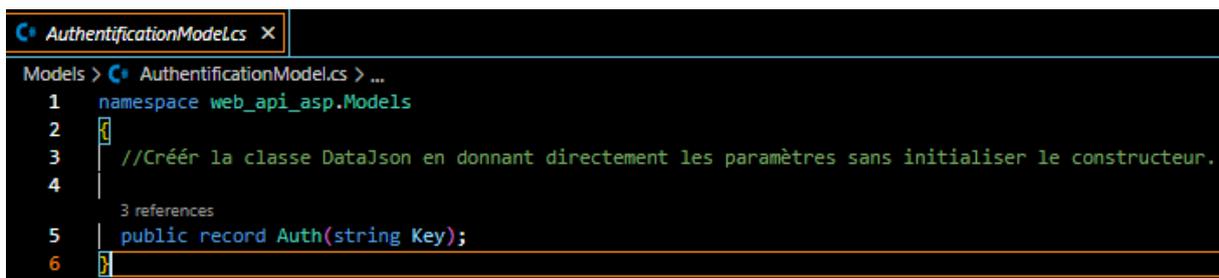


```
AgenceModel.cs x
Models > AgenceModel.cs > Agences > Agences
1 namespace web_api_asp.Models
2
3 //Créer la classe DataJson en donnant directement les paramètres sans initialiser le constructeur.
4
5 public record Agences
6 (
7     string COD_AGE,
8     string LIB_AGE
9 );
10
```

Structure attendue :

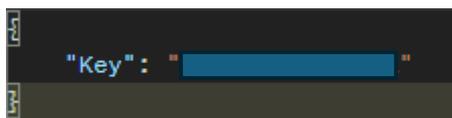


```
[
  {
    "COD_AGE": "CHA",
    "LIB_AGE": "PAP CHAPONNAY"
  },
  {
    "COD_AGE": "DIG",
    "LIB_AGE": "PAP Export Guadeloupe"
  },
  {
    "COD_AGE": "DSE",
    "LIB_AGE": "PAP MIRAMAS"
  },
  {
    "COD_AGE": "EXP",
    "LIB_AGE": "PRO A PRO EXPORT"
  }
]
```



```
AuthenticationModel.cs x
Models > AuthenticationModel.cs > ...
1 namespace web_api_asp.Models
2
3 //Créer la classe DataJson en donnant directement les paramètres sans initialiser le constructeur.
4
5 public record Auth(string Key);
6
```

Résultat attendue :



```
{
  "Key": ""
}
```

C'est le mot de passe qui permet d'accéder à l'environnement WS (AS400).

```

DataJsonModel.cs 5 X
Models > DataJsonModel.cs > Agence
1 using System.Text.Json.Serialization;
2
3 namespace web_api_asp.Models
4 {
5
6 //Créer la classe DataJson et Agence en donnant directement les paramètres sans initialiser le constructeur.
7
8 /*
9 public record DataJson(string Name, string Description, List<Agence> Agences);
10 public record Agence(string COD_AGE, string LIB_AGE);
11 */
12
13 4 references
14 public class DataJson
15 {
16     [JsonPropertyName("name")]
17     1 reference
18     public string Name { get; set; }
19
20     [JsonPropertyName("description")]
21     1 reference
22     public string Description { get; set; }
23
24     [JsonPropertyName("agences")]
25     1 reference
26     public List<Agence> Agences { get; set; }
27 }
28
29 26 references
30 public class Agence
31 {
32     /*
33     Permet de mettre des propriétés individuelles en utilisant cet attribut => [JsonPropertyName("")]
34     Dans notre requête les paramètres seront en majuscule
35     */
36
37     25 references
38     public string COD_AGE { get; set; }
39
40     25 references
41     public string LIB_AGE { get; set; }
42 }
43 }

```

WS / /datajson / /datajson

GET

Params Headers Body

Query Params

Key	Value
Key	Value

Body Headers (4)

{ } JSON

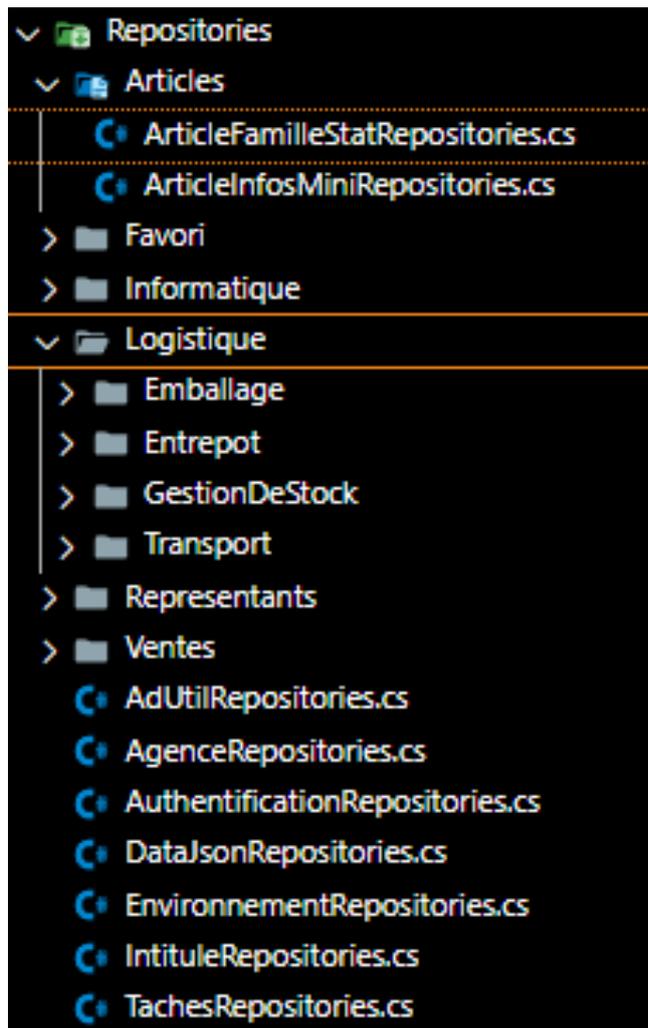
```

1 {
2   "name": "datagescom",
3   "description": "Liste de données en dur pour les applications",
4   "agences": [
5     {
6       "COD_AGE": "MTB",
7       "LIB_AGE": "MONTAUBAN"
8     },
9     {
10      "COD_AGE": "PER",
11      "LIB_AGE": "PERPIGNAN"
12    },
13    {
14      "COD_AGE": "DSE",
15      "LIB_AGE": "MIRAMAS"
16    },
17    {
18      "COD_AGE": "DOL",
19      "LIB_AGE": "DOLE"
20    },
21    {
22      "COD_AGE": "BZH",
23      "LIB_AGE": "SAINT GILLES"
24    },
25    {
26      "COD_AGE": "HAR",
27      "LIB_AGE": "CHALETTE"
28    },
29    {
30      "COD_AGE": "DID",
31      "LIB_AGE": "RUNGIS"
32    }
33  ]
34 }

```

**Repositories (Couche d'accès aux données) :**

Encapsulent la logique d'accès aux données, servant d'intermédiaire entre les Contrôleurs et la source de données (base de données, services externes, etc.). Ils définissent les méthodes pour interagir avec les entités (CRUD : Create, Read, Update, Delete). L'utilisation de repositories permet d'isoler la logique d'accès aux données.



Exemple de repositories (AgenceRepositories.cs, AuthenticationRepositories.cs, etc.)

Contient les méthodes pour récupérer des informations sur les agences.

```
AgenceRepositories.cs x
Repositories > AgenceRepositories.cs > ...
1  using Dapper;
2  using web_api_esp.Models;
3  using web_api_esp.Tools;
4
5  namespace web_api_esp.Repositories
6  {
7      4 references
      public class AgenceRepositories
8      {
9          3 references
          private DatabaseContext _databaseContext;
10
11         0 references
          public AgenceRepositories(DatabaseContext databaseContext)
12         {
13             _databaseContext = databaseContext;
14         }
15
16         // Retournera la liste de toutes les agences.
          1 reference
          public async Task<IEnumerable<Agences>> ListAgencesAsync()
17         {
18             return await _databaseContext.DbConnection.QueryAsync<Agences>(  
19                 @"SELECT IAAGE AS COD_AGE,  
20                 | COD_AGE.INLIB AS LIB_AGE FROM ADAGID  
21                 | LEFT JOIN DCINT COD_AGE ON COD_AGE.INARG = IAAGE AND COD_AGE.INCOD = 'AGE'  
22                 | WHERE IAPAY = 'FR' and IAANN = 'N'");  
23             }
24
25         // Retournera la liste des agences d'un utilisateur.
          1 reference
          public async Task<IEnumerable<Agences>> ListAgencesByUserAsync(string user)
26         {
27             return await _databaseContext.DbConnection.QueryAsync<Agences>(  
28                 @"SELECT DISTINCT COD_AGE, LIB_AGE  
29                 | FROM VUE_DROITS  
30                 | JOIN sysschemas ON schema_name  
31                 | LIKE 'SPF_%' AND substr(schema_name, 5, 3) = COD_AGE  
32                 | WHERE TACHE = 'ADPAS' AND (USR = @User or G_USR = @User)  
33                 | ORDER BY COD_AGE",  
34                 new { User = user }  
35             );  
36         }
37     }
38 }
39
40 }
```

## Gère la logique d'authentification des utilisateurs

```
AuthenticationRepositories.cs 2 X
Repositories > AuthenticationRepositories.cs > AuthenticationRepositories > ValidateTokenAsync
1 using System.Drawing;
2 using System.Security.Claims;
3 using System.Text;
4 using Microsoft.EntityFrameworkCore.Metadata.Internal;
5 using web_api_asp.Models;
6 using web_api_asp.Tools;
7
8 namespace web_api_asp.Repositories
9 {
10     5 references
11     public class AuthenticationRepositories
12     {
13         3 references
14         DatabaseContext _databaseContext;
15
16         0 references
17         public AuthenticationRepositories(DatabaseContext databaseContext)
18         {
19             _databaseContext = databaseContext;
20         }
21
22         2 references
23         public async Task<Auth?> AuthenticateAsync(string username, string password)
24         {
25             try
26             {
27                 // Utiliser la méthode CreateDbConnection pour vérifier les informations d'identification dans la base de données
28                 _databaseContext.CreateDbConnection(username, password);
29
30                 // Si l'authentification est réussie, retourner un objet Auth avec les identifiants
31                 return new Auth(username + ":" + password);
32             }
33             catch
34             {
35                 // Si une erreur survient, retourne null (échec authentification)
36                 return null;
37             }
38         }
39
40         // Méthode pour valider le token sans chiffrement.
41         1 reference
42         public async Task<bool> ValidateTokenAsync(string token)
43         {
44             var pp = token.Split(':');
45             try
46             {
47                 // Vérifie la validité du token dans la base de données
48                 _databaseContext.CreateDbConnection(pp[0], pp[1]);
49                 return true;
50             }
51             catch
52             {
53                 return false;
54             }
55         }
56     }
57 }
```

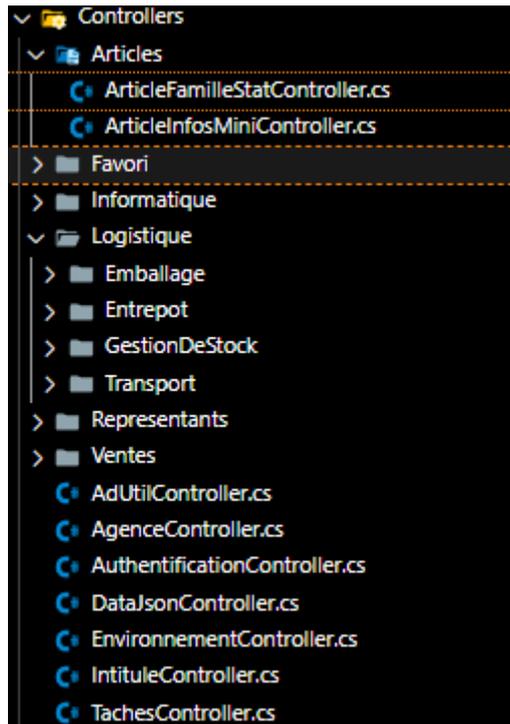
Permet d'interagir avec les données des articles

```
ArticleFamilleStatRepositories.cs x
Repositories > Articles > ArticleFamilleStatRepositories.cs > ...
1  using Dapper;
2  using web_api_esp.Models.Articles;
3  using web_api_esp.Tools;
4
5  namespace web_api_esp.Repositories.Articles
6  {
7      3 references
8      public class ArticleFamilleStatRepositories(DatabaseContext databaseContext)
9      {
10         3 references
11         private DatabaseContext _databaseContext = databaseContext;
12
13         // Retournera la liste de tous les articles famille stat.
14         1 reference
15         public async Task<IEnumerable<Article>> ListArticleFamilleStatAsync()
16         {
17             return await _databaseContext.DbConnection.QueryAsync<Article>(
18                 @"SELECT DISTINCT ATSF AS code, LIB_FAS AS libelle
19                 FROM CTARST
20                 | JOIN DIC_FAS ON CODE_FAS = ATSF
21                 WHERE ATSTCL = 'AST' AND ATSTAN = 'N'
22                 ORDER BY ATSF "
23             );
24
25         // Retournera la liste de tous les articles sous famille stat.
26         1 reference
27         public async Task<IEnumerable<Article>> ListSFamilleStatArtAsync(string cod_fam_stat)
28         {
29             return await _databaseContext.DbConnection.QueryAsync<Article>(
30                 @"SELECT DISTINCT ATSSF AS code, LIB_SFAS AS libelle
31                 FROM CTARST
32                 | JOIN DIC_SFAS ON CODE_SFAS = ATSSF
33                 WHERE ATSTCL = 'AST' AND ATSTAN = 'N' AND ATSF = @CodFamStat
34                 ORDER BY ATSSF ",
35                 new { CodFamStat = cod_fam_stat }
36             );
37
38         // Retournera la liste de tous les articles sous sous famille stat.
39         1 reference
40         public async Task<IEnumerable<Article>> ListSSFamilleStatArtAsync(string cod_fam_stat, string cod_s_fam_stat)
41         {
42             return await _databaseContext.DbConnection.QueryAsync<Article>(
43                 @"SELECT DISTINCT ATSSSF AS code, LIB_SSFAS AS libelle
44                 FROM CTARST
45                 | JOIN DIC_SSFAS ON CODE_SSFAS = ATSSSF
46                 WHERE ATSTCL = 'AST' AND ATSTAN = 'N' AND ATSF = @CodFamStat AND ATSSF = @CodSfamStat
47                 order by ATSSSF ",
48                 new { CodFamStat = cod_fam_stat, CodSfamStat = cod_s_fam_stat }
49             );
50
51         }
52     }
}
```

### Contrôleurs (Controllers) :

Reçoit les requêtes HTTP des clients, interagissent avec les Repositories pour récupérer ou manipuler les données, et renvoie les réponses HTTP (généralement au format JSON).

Ils contiennent la logique de haut niveau de l'API.



Exemple de controllers (*AgencesController.cs*, *AuthenticationController.cs*, etc.)

Gère les requêtes liées aux agences

```
AgenceController.cs x
Controllers > AgenceController.cs > AgenceController > AgenceController
1 using Microsoft.AspNetCore.Mvc;
2 using web_api_asp.Repositories;
3 using web_api_asp.Models;
4
5 namespace web_api_asp.Controllers
6 {
7
8     // Modèle de la route itineraire {{WS_URL}}
9     [Route("")]
10    [ApiController]
11
12
13    // Constructeur primaire.
14    0 references
15    public class AgenceController(AgenceRepositories agenceRepositories) : ControllerBase
16    {
17        2 references
18        private readonly AgenceRepositories _AgenceRepositories = agenceRepositories;
19
20        // GET Récupère toutes les agences .
21        // GET /toutesagences
22        [HttpGet("toutesagences")]
23        public async Task<ActionResult<Agences>> GetAllAgences()
24        {
25            var allAgences = await _AgenceRepositories.ListAgencesAsync();
26
27            return Ok(allAgences); // Retourne la variable qui contient la méthode pour retourner toutes les agences.
28
29        // GET Récupère les agences d'un utilisateur
30        // GET /agences/{user}
31        [HttpGet("agences/{user}")]
32        0 references
33        public async Task<ActionResult<Agences>> GetAgencesByUser(string user)
34        {
35            var agencesByUser = await _AgenceRepositories.ListAgencesByUserAsync(user);
36
37            return Ok(agencesByUser); // Retourne la variable qui contient la méthode pour retourner les agences d'un utilisateur.
38        }
39    }
40 }
```

Endpoint de connexion (get /login) et gère la génération des jetons d'authentification.

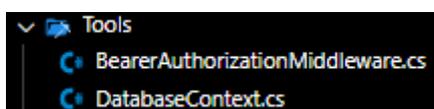
```
AuthenticationController.cs x
Controllers > AuthenticationController.cs > ...
1 using web_api_asp.Repositories;
2 using Microsoft.AspNetCore.Mvc;
3 using web_api_asp.Models;
4 using System.Text;
5 using Microsoft.AspNetCore.Authorization;
6 using Microsoft.EntityFrameworkCore.Storage.ValueConversion;
7
8 namespace web_api_asp.Controllers
9
10 // Modèle de la route itinéraire {WS_URL}
11 [Route("")]
12 [ApiController]
13 // Constructeur primaire.
14
15 0 references
16 public class AuthenticationController(AuthenticationRepositories authenticationRepositories) : ControllerBase
17 {
18     1 reference
19     private readonly AuthenticationRepositories _authenticationRepositories = authenticationRepositories;
20
21     // GET Récupère le login (token)
22     // GET /login
23     [HttpGet("login")]
24     [AllowAnonymous]
25     0 references
26     public async Task<ActionResult<Auth>> GetAuthAsync()
27     {
28         var authHeader = Request.Headers["Authorization"].ToString();
29         if (!string.IsNullOrEmpty(authHeader) && authHeader.StartsWith("Basic ", StringComparison.OrdinalIgnoreCase))
30         {
31             // Extrait les informations d'identification encodées en Base64 (après le préfixe "Basic ")
32             var encodedCredentials = authHeader.Substring("Basic ".Length).Trim();
33
34             // Décode la chaîne Base64 pour obtenir username:password
35             var decodedCredentials = Encoding.UTF8.GetString(Convert.FromBase64String(encodedCredentials));
36
37             // Divise en username et password
38             var credentials = decodedCredentials.Split(':');
39             if (credentials.Length == 2)
40             {
41                 var token = await _authenticationRepositories.AuthenticateAsync(credentials[0], credentials[1]);
42                 if (token != null)
43                 {
44                     return Ok(token);
45
46                     // Encoder les informations d'identification (username:password) en Base64
47                     // var encodedResponse = Convert.ToBase64String(Encoding.UTF8.GetBytes($"{credentials[0]}:{credentials[1]}"));
48                     // return Ok(new { Key = encodedResponse });
49                 }
50             }
51             return Unauthorized("Invalid credentials");
52         }
53     }
54 }
```

## Consulter ou manipuler les articles

```
ArticleFamilleStatController.cs x
Controllers > Articles > ArticleFamilleStatController.cs > ArticleFamilleStatController > ArticleFamilleStatController
1 using Microsoft.AspNetCore.Mvc;
2 using web_api_esp.Models.Articles;
3 using web_api_esp.Repositories.Articles;
4
5 namespace web_api_esp.Controllers.Articles
6 {
7     // Modèle de la route itinéraire {{WS_URL}}
8     [Route("")]
9     [ApiController]
10
11     // Constructeur primaire.
12     0 references
13     public class ArticleFamilleStatController(ArticleFamilleStatRepositories articleFamilleStatRepositories) : ControllerBase
14     {
15         3 references
16         private readonly ArticleFamilleStatRepositories _ArticleFamilleStatRepositories = articleFamilleStatRepositories;
17
18         // GET Récupère la liste des articles famille stat.
19         // GET /articles/article-famille-stat
20         [HttpGet("articles/article-famille-stat")]
21         0 references
22         public async Task<ActionResult<Article>> GetFamilleStatArt()
23         {
24             var resultat = await _ArticleFamilleStatRepositories.ListArticleFamilleStatAsync(); ;
25             return Ok(resultat); // Retourne la variable qui contient la méthode pour retourner la liste des articles famille stat.
26         }
27
28         // // GET Récupère la liste des articles sous famille stat.
29         // // GET /articles/article-sfamille-stat/{cod_fam_stat}
30         [HttpGet("articles/article-sfamille-stat/{cod_fam_stat}")]
31         0 references
32         public async Task<ActionResult<Article>> GetSFamilleStatArt(string cod_fam_stat)
33         {
34             var resultat = await _ArticleFamilleStatRepositories.ListSFamilleStatArtAsync(cod_fam_stat);
35             return Ok(resultat); // Retourne la variable qui contient la méthode pour retourner la liste des articles sous famille stat.
36         }
37
38         // // GET Récupère la liste des articles sous sous famille stat.
39         // // GET /articles/article-ssfamille-stat/{cod_fam_stat}/{cod_s_fam_stat}
40         [HttpGet("articles/article-ssfamille-stat/{cod_fam_stat}/{cod_s_fam_stat}")]
41         0 references
42         public async Task<ActionResult<Article>> SSFamilleStatArt(string cod_fam_stat, string cod_s_fam_stat)
43         {
44             var resultat = await _ArticleFamilleStatRepositories.ListSSFamilleStatArtAsync(cod_fam_stat, cod_s_fam_stat); ;
45             return Ok(resultat); // Retourne la variable qui contient la méthode pour retourner la liste des articles sous sous famille stat.
46         }
47     }
48 }
```

## Gestion de la base de données (DbContext)

Le DbContext est la **passerelle principale** l'application C# et la base de données. Il représente la **session avec la base de données** et est responsable de la **traduction des requêtes LINQ (Language Integrated Query)** en requêtes SQL. J'ai utilisé le service **DapperNTI**.



```
DatabaseContext.cs 1 x
Tools > DatabaseContext.cs > DatabaseContext > DbConnection
1 using System;
2 using System.Data;
3 using System.Security.Claims;
4 using Aumerial.Data.Nti;
5 using Dapper;
6
7 namespace web_api_asp.Tools;
8
9 43 references
10 public class DatabaseContext
11 {
12     4 references
13     private NTiConnection _dbConnection;
14
15     32 references
16     public NTiConnection DbConnection {
17         get {
18             return _dbConnection;
19         }
20     }
21
22     2 references
23     public void CreateDbConnection(string username, string password)
24     {
25         var connectionString = $"Server=10.183. user={username};password={password};naming convention=1;";
26         _dbConnection = new NTiConnection(connectionString);
27         _dbConnection.Open();
28
29         // TODO: => ajouter agence (_MTB) à partir d'une variable
30         _dbConnection.Execute("CALL ('(' (')')");
31     }
32 }
33
34 }
```

Ce qui est caché sont les **bibliothèques AS/400** utilisées.

## Authentification et autorisation (BearerAuthorizationMiddleware.cs)



La sécurité de l'API était primordiale pour garantir que seuls les employés internes autorisés puissent accéder aux données sensibles de l'E-Gescom. Pour cela, un middleware personnalisé, **BearerAuthorizationMiddleware.cs**, a été développé pour gérer le processus d'authentification et d'autorisation des requêtes.

Son rôle est de s'intercaler dans le pipeline de traitement des requêtes HTTP d'ASP.NET Core. Son objectif est d'inspecter chaque requête entrante avant qu'elle n'atteigne le contrôleur cible, afin de vérifier les informations d'authentification présentes dans l'entête Authorization.

## Visuallisation de la méthode

```
0 references
public async Task InvokeAsync(HttpContext context, AuthenticationRepositories authenticationRepositories)
{
    var endpoint = context.GetEndpoint();
    if (endpoint?.Metadata.GetMetadata<AllowAnonymousAttribute>() != null)
    {
        // Skip authentication if the [AllowAnonymous] attribute is present
        await _next(context);
        return;
    }

    // Get the token from the Authorization header
    var authorizationHeader = context.Request.Headers["Authorization"].FirstOrDefault();
    var auth = authorizationHeader?.Split(' ');
    if (auth == null || auth.Length != 2)
    {
        context.Response.StatusCode = 401; // Unauthorized
        await context.Response.WriteAsync("Missing or invalid Authorization header.");
        return;
    }

    var authMethod = auth[0].ToLower();
    var authData = auth[1];

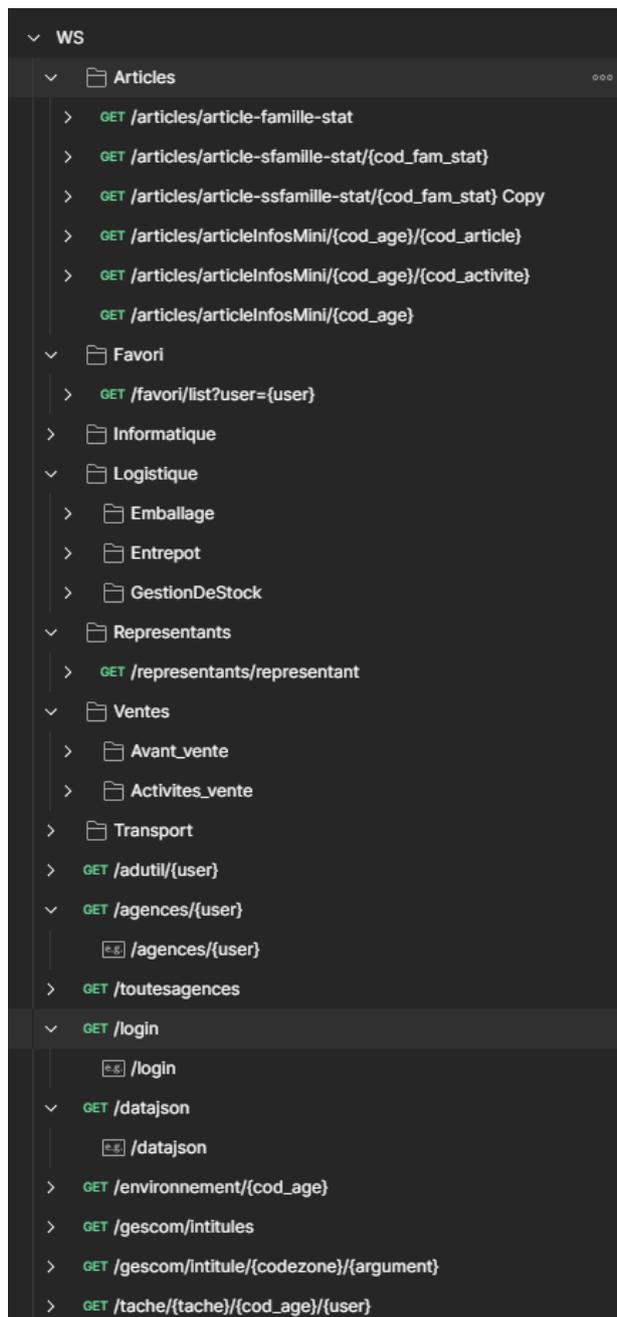
    if (authMethod == "basic")
    {
        var decodedCredentials = Encoding.UTF8.GetString(Convert.FromBase64String(authData));
        var credentials = decodedCredentials.Split(':');
        if (await authenticationRepositories.AuthenticateAsync(credentials[0], credentials[1]) == null)
        {
            context.Response.StatusCode = 403; // Forbidden
            await context.Response.WriteAsync("Invalid credentials.");
            return;
        }
    }
    else if (authMethod == "bearer")
    {
        if (!await authenticationRepositories.ValidateTokenAsync(authData))
        {
            context.Response.StatusCode = 403; // Forbidden
            await context.Response.WriteAsync("Invalid or expired token.");
            return;
        }
    }
    else
    {
        context.Response.StatusCode = 403; // Forbidden
        await context.Response.WriteAsync("Invalid credentials.");
        return;
    }

    // Proceed to the next middleware
    await _next(context);
}
```

## Exposition des Services (WS)

L'API RESTful développée expose un ensemble de services web structurés et organisés par domaines métier, permettant aux applications clientes internes de Pro à Pro (notamment le site E-Gescom) d'interagir avec les données et les fonctionnalités de gestion agro-alimentaire. Chaque "folder" de la structure des WS correspond généralement à un contrôleur ou à un ensemble de fonctionnalités regroupées logiquement.

### Visualisation des WS (Postman)



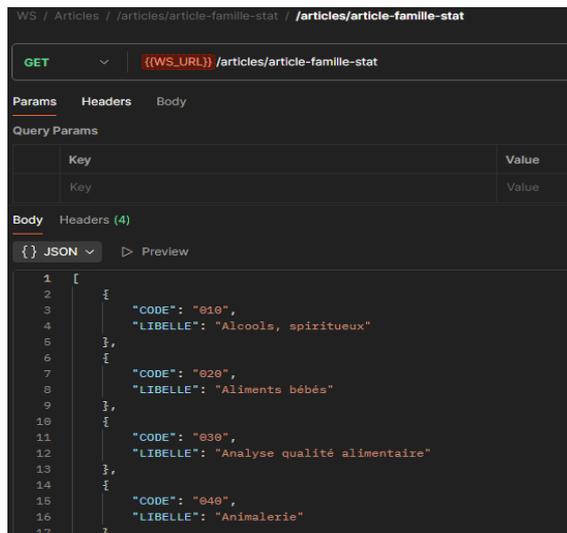
## Structure des endpoints de l'API REST

L'API Web RESTful développée pour le site E-Gescom interne de Pro à Pro expose une série d'endpoints, organisés de manière logique par domaines métier. Cette structure permet aux applications clientes internes de Pro à Pro d'interagir de manière claire et cohérente avec les données et les fonctionnalités de gestion agro-alimentaire et d'administration interne.

## Exemples d'endpoints GET

**Articles** : Ces endpoints gèrent l'accès et la manipulation des informations relatives aux articles et à leurs classifications.

**GET /articles/article-famille-stat** : Permet d'obtenir les différentes familles statistiques d'articles.



```
WS / Articles / /articles/article-famille-stat / /articles/article-famille-stat
GET {{WS_URL}} /articles/article-famille-stat
Params Headers Body
Query Params
Key Value
Key Value
Body Headers (4)
JSON Preview
1 [
2   {
3     "CODE": "010",
4     "LIBELLE": "Alcools, spiritueux"
5   },
6   {
7     "CODE": "020",
8     "LIBELLE": "Aliments bébés"
9   },
10  {
11   "CODE": "030",
12   "LIBELLE": "Analyse qualité alimentaire"
13  },
14  {
15   "CODE": "040",
16   "LIBELLE": "Animalerie"
17  },
18 ]
```

**GET /articles/article-sfamille-stat/{cod\_fam\_stat}** : Retourne les sous-familles statistiques associées à une famille statistique donnée.

```

WS / Articles / /articles/article-sfamille-stat/{cod_fam_stat} /articles/article-sfamille-stat/{cod_fam_stat}
GET {{WS_URL}} /articles/article-sfamille-stat/050
Params Headers Body
Query Params
Key Value
Body Headers (4)
JSON Preview
1 [
2   {
3     "CODE": "027",
4     "LIBELLE": "Anisés"
5   },
6   {
7     "CODE": "063",
8     "LIBELLE": "Base plantes, fruits et amers"
9   },
10  {
11   "CODE": "064",
12   "LIBELLE": "Base vins"
13  },
14  {
15   "CODE": "423",
16   "LIBELLE": "Sans alcool"
17  }
18 ]

```

**GET /articles/articleInfosMini/{cod\_age}/{cod\_article}** : Fournit des informations synthétiques sur un article spécifique en fonction du code agence et du code article.

```

WS / Articles / /articles/articleInfosMini/{cod_age}/{cod_article} /articles/articleInfosMini/{cod_age}/{cod_article}
GET {{WS_URL}} /articles/articleInfosMini/MTB/112567
Params Headers Body
Body Headers (4)
JSON Preview
1 [
2   {
3     "CODE_ART": "112567",
4     "AGENCE": "MTB",
5     "CODE_ACT": "EPI",
6     "LIBACTIVITE": "Produits d'épicerie",
7     "SERVICE_PREPA": "SEC",
8     "UV": "BTE",
9     "LIBUV": "Boite",
10    "DESIG_COMMERC": "Ail en cubes lyophilisé 200 g",
11    "CONDITIONNEMENT": "C6",
12    "LIBCONDITIONNEMENT": "Carton de 6",
13    "CODE_MARQUE": "DUCRO",
14    "MARQUE": "Ducros",
15    "MOT_DIRECTEUR": "AIL",
16    "BIO": "N",
17    "EGALIM": "N",
18    "CHEMINVISUEL": "http://papas400pi.prooapro.info/DAM/VISUELS/V0000247433.JPG",
19    "CHEMINFICHETECHNIQUE": "https://prooapro.fr/media/product/0/2/4/1/T0000241235.PDF",
20    "NOMFICHETECHNIQUE": "T0000241235.PDF",
21    "CHEMINFICHESECURITE": "",
22    "NOMFICHESECURITE": "",
23    "CHEMINFICHELOGISTIQUE": "https://prooapro.fr/media/product/0/2/4/1/L0000241236.PDF",
24    "NOMFICHELOGISTIQUE": "L0000241236.PDF",
25    "STATUT": ""
26   }
27 ]

```

**GET /articles/articleInfosMini/{cod\_age}/{cod\_activite}** : Fournit des informations synthétiques sur un article en fonction du code agence et du code d'activité (peut-être lié à un type de produit ou service).

```

/articles/articleInfosMini
+
WS / Articles / /articles/articleInfosMini(cod_age)/(cod_activite) /articles/articleInfosMini(cod_age)/(cod_activite)
GET {{WS_URL}} /articles/listeArticleInfosMiniParActivite/MTB/FRA
Params Headers Body
Body Headers (4)
JSON Preview
1 [
2   {
3     "CODE_ART": "37400",
4     "AGENCE": "MTB",
5     "CODE_ACT": "FRA",
6     "LIBACTIVITE": "Produits Frais",
7     "SERVICE_PREPA": "FRA",
8     "UV": "KG",
9     "LIBUV": "Kilogramme",
10    "DESIG_COMMERC": "Jambon sec sup. FR désossé rectangle 7 M 5,5 kg",
11    "CONDITIONNEMENT": "C11K",
12    "LIBCONDITIONNEMENT": "Carton de 11 Kg",
13    "CODE_MARQUE": "SALPE",
14    "MARQUE": "Saloir du Périg",
15    "MOT_DIRECTEUR": "JAMBON SEC",
16    "BIO": "N",
17    "EGALIM": "N",
18    "CHEMINVISUEL": "http://papas400p1.proopro.info/DAN/VISUELS/V0800415641.JPG",
19    "CHEMINFICHETECHNIQUE": "https://proopro.fr/media/product/0/0/1/4/T0800014849.PDF",
20    "NOMFICHESECHNQUE": "T0800014849.PDF",
21    "CHEMINFICHESECURITE": "",
22    "NOMFICHESECURITE": "",
23    "CHEMINFICHELOGISTIQUE": "",
24    "NOMFICHELOGISTIQUE": "",
25    "STATUT": ""
26  },
27  {
28    "CODE_ART": "24118",
29    "AGENCE": "MTB",
30    "CODE_ACT": "FRA",

```

**Agences** : Offre un accès à la liste complète de toutes les agences.

```

/toutesagences
+
WS / /toutesagences / /toutesagences
GET {{WS_URL}} /toutesagences
Params Headers Body
Body Headers (4)
JSON Preview
1 [
2   {
3     "COD_AGE": "BZH",
4     "LIB_AGE": "PAP SAINT GILLES"
5   },
6   {
7     "COD_AGE": "CFR",
8     "LIB_AGE": "PAP LONGUEIL"
9   },
10  {
11    "COD_AGE": "CHA",
12    "LIB_AGE": "PAP CHAPONNAY"
13  },
14  {
15    "COD_AGE": "DID",
16    "LIB_AGE": "PRO A PRO RUNGIS"
17  },
18  {
19    "COD_AGE": "DIG",
20    "LIB_AGE": "PAP Export Guadeloupe"
21  },
22  {
23    "COD_AGE": "DIM",
24    "LIB_AGE": "PAP MARTINIQUE"
25  },
26  {
27    "COD_AGE": "DOL",
28    "LIB_AGE": "PAP DOLE"
29  },
30  {
31    "COD_AGE": "DOM",
32    "LIB_AGE": "Agence PAP D.O.M."
33  },
34  {
35    "COD_AGE": "DSE",
36    "LIB_AGE": "PAP MIRAMAS"
37  },
38  {
39    "COD_AGE": "EXP",
40    "LIB_AGE": "PRO A PRO EXPORT"
41  },

```

## Connexion et Déploiement

### Tunnel SSH pour l'accès et le port forwarding

Pour le développement, les tests, la mise en place d'un accès sécurisé et d'une méthode de communication entre les environnements était essentielle. Cette section décrit l'utilisation d'un tunnel SSH avec redirection de ports pour établir cette connectivité.

*Objectif du Tunnel SSH* : Le tunnel SSH (Secure Shell) a été utilisé pour établir une connexion sécurisée vers la machine distante (10.183.X.1).

Il permet de :

- **Accéder au serveur distant** : En se connectant avec mon identifiant AS400 (M.....G@10.183.X.1).
- **Sécuriser les communications** : Toutes les données transitent via un canal chiffré.

*Redirection de Port Inversée (-R 5056:localhost:5056)* : La commande SSH spécifique utilisée était :

```
C:\Users\matheo.sage-ext>ssh -R 5056:localhost:5056 M G@10.183. 1
```

### Configuration des URLs des services

Les URLs utilisées envoyant des requêtes HTTP aux services externes (par exemple, pour récupérer des données métiers depuis un système AS/400 via une API tierce, ou pour communiquer avec des services PHP existants pendant la phase de migration).

Dans le fichier **config.inc.php** nous y mettons nos urls

```
config.inc.php [x]
const BBB_PASS_DEFAULT = 'G@10.183.X.1';
// const BASE_PATH_WS = 'http://10.183.X.1:11100/ws';
// const BASE_PATH_WS = 'http://10.183.X.1:11106';
```

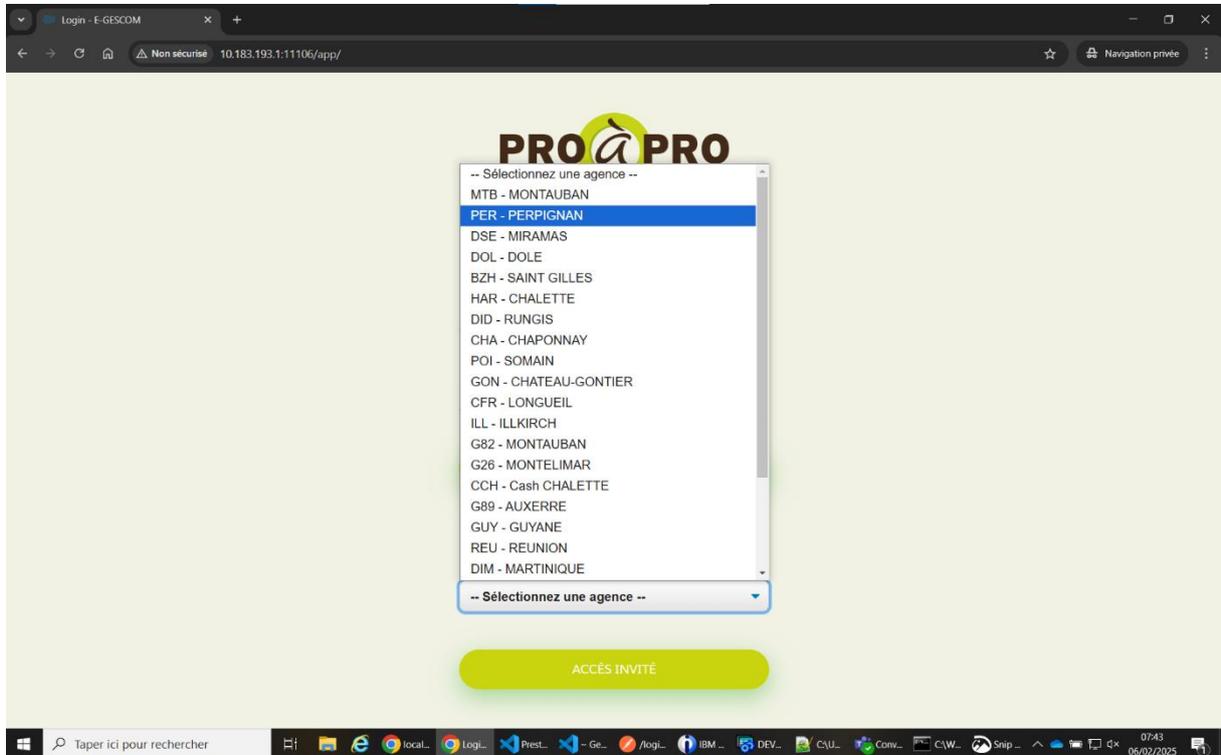
'http://10.183.X.1:11100/ws' => adresse et ports spécifiques pour utiliser et joindre les "Web Services" existants (serveur développement).

'http://10.183.X.1:11106' = adresse et port qui représente mon vhost personnel de développement pour tester mes requêtes.

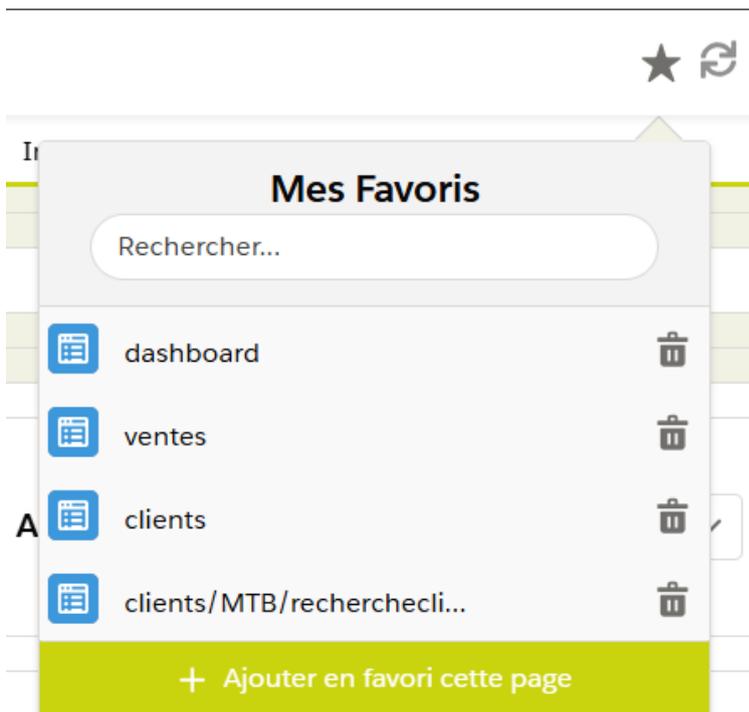
## II. Tests

### Résultats

Login :



Favoris part rapport à un utilisateur :



### Profil utilisateur plusieurs résultats :

**Module Profil Utilisateur**

Matricule: MASAG | Nom: SAGE MATEO | Service: Inconnu | Téléphone: Inconnu | Email: Inconnu

Accueil > Informatique > Support > Profil Utilisateur

#### Liste des tâches de l'utilisateur

Afficher 10 éléments

Niveau de droit	Agence	Nom tache	Groupe tache	Libellé
G	MTB	[Redacted]	[Redacted]	MAJ du 2xNet et 3xNet 0 par le PR
G	MTB	[Redacted]	[Redacted]	Ed. achats par articles/fournisseur
G	MTB	[Redacted]	[Redacted]	Edition achats par fournisseur/articles
G	MTB	[Redacted]	[Redacted]	Edition achats par fournisseur/articles
G	MTB	[Redacted]	[Redacted]	Etat des modifications en pointage de factures
G	MTB	[Redacted]	[Redacted]	Etat des modifications en pointage de factures
G	MTB	[Redacted]	[Redacted]	Détermination articles à calculer stock mini.
G	MTB	[Redacted]	[Redacted]	Gestion des articles emballages
G	MTB	[Redacted]	[Redacted]	Gestion articles fournisseurs alerte réappro.
G	MTB	[Redacted]	[Redacted]	Arrêt/Redémarrage "AFA1" et "AFA2"

Affichage de 1 à 10 sur 1.546 éléments

**Module Profil Utilisateur**

Matricule: MASAG | Nom: SAGE MATEO | Service: Inconnu | Téléphone: Inconnu | Email: Inconnu

Accueil > Informatique > Support > Profil Utilisateur

#### Liste des tâches de l'utilisateur

Afficher 10 éléments

Rechercher: MTB\_TOUS

Niveau de droit	Agence	Nom tache	Groupe tache	Libellé
G	MTB	[Redacted]	MTB_TOUS	Ed. achats par articles/fournisseur
G	MTB	[Redacted]	MTB_TOUS	Edition achats par fournisseur/articles
G	MTB	[Redacted]	MTB_TOUS	Etat des modifications en pointage de factures
G	MTB	[Redacted]	MTB_TOUS	Edition des alertes de réappro.
G	MTB	[Redacted]	MTB_TOUS	Edition des alertes de réappro. prévisionnelle
G	MTB	[Redacted]	MTB_TOUS	Articles non au catalogue et réputés actifs
G	MTB	[Redacted]	MTB_TOUS	Etat de préparation des fromages à la coupe
G	MTB	[Redacted]	MTB_TOUS	Edition des marchandises à recevoir le..
G	MTB	[Redacted]	MTB_TOUS	Edition codes articles personnalisés/client
G	MTB	[Redacted]	MTB_TOUS	Fichier articles : recherche par mot directeur

Affichage de 1 à 10 sur 307 éléments (filtrés de 1.546 éléments au total)

Profil utilisateur recherché :

Module Profil Utilisateur

Matricule: RPO03704P, Nom: PONTELLO Remi, Service: Inconnu, Téléphone: Inconnu, Email: dummy\_mail@proapro.fr

Accueil > Informatique > Support > Profil Utilisateur

Liste des tâches de l'utilisateur

Afficher 10 éléments

Niveau de droit	Agence	Nom tâche	Groupe tâche	Libellé
G	MTB		INF_DEM	MAJ du 2xNet et 3xNet 0 par le PR
G	MTB		MTB_INFOR	MAJ du 2xNet et 3xNet 0 par le PR
G	MTB		MTB_INFOR	Ed. achats par articles/fournisseur
G	MTB		MTB_TOUS	Ed. achats par articles/fournisseur
G	MTB		INF_DEM	Edition achats par fournisseur/articles
G	MTB		MTB_INFOR	Edition achats par fournisseur/articles
G	MTB		MTB_TOUS	Edition achats par fournisseur/articles
G	MTB		INF_DEM	Etat des modifications en pointage de factures
G	MTB		MTB_INFOR	Etat des modifications en pointage de factures
G	MTB		MTB_TOUS	Etat des modifications en pointage de factures

Section Articles => Recherche article :

Module Recherche Article

Accueil > Articles > Recherche Article

Liste | Visuels | Extraction Excel

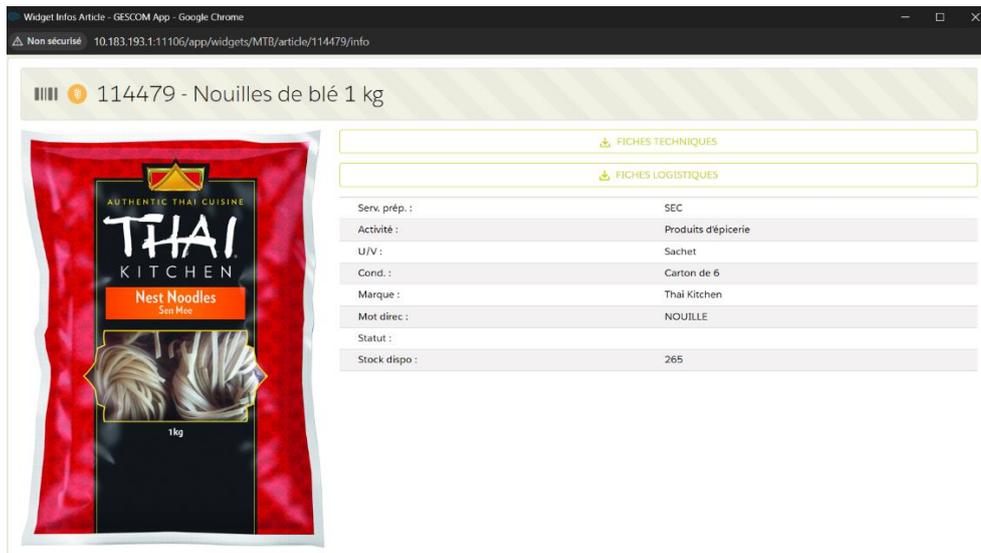
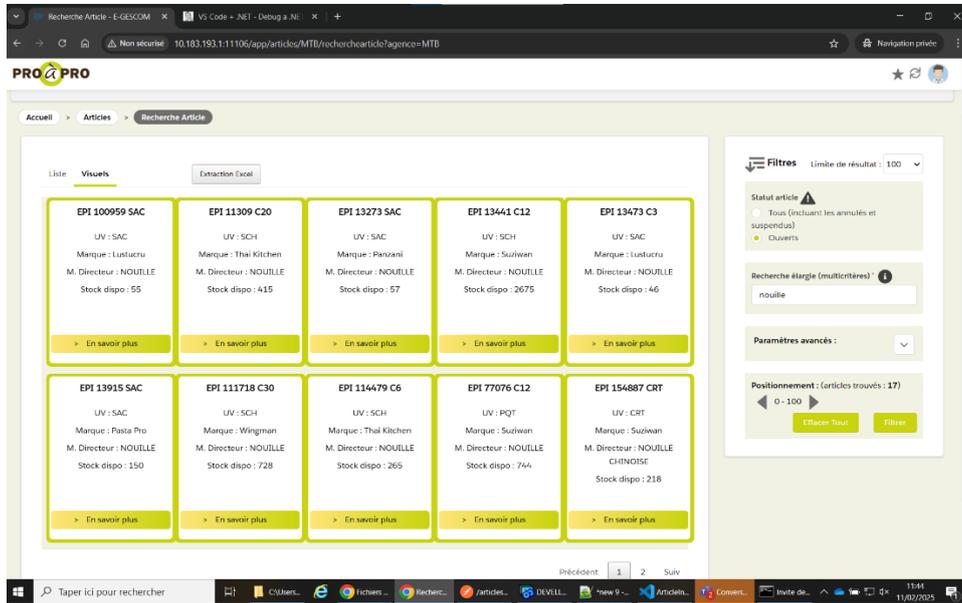
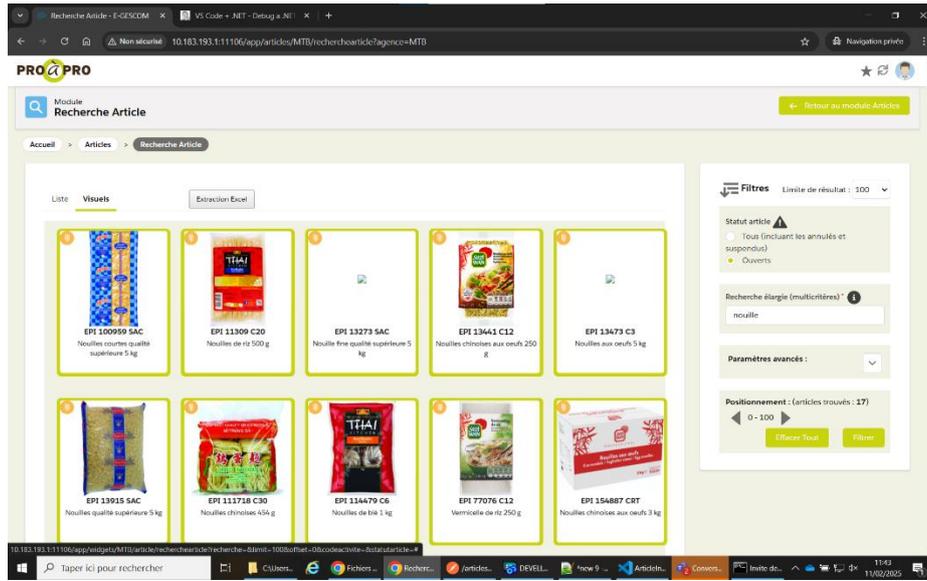
Actions	Code Article	Activité	UV	Marque	Designation Commerciale	Conditionnement	Mot Directeur	Statut Article
	100959	EPI	SAC	Lustucru	Nouilles courtes qualité supérieure 5 kg	SAC	NOUILLE	
	11309	EPI	SCH	Thai Kitchen	Nouilles de riz 500 g	C20	NOUILLE	
	19179	EPI	SAG	Rivoire&Garret	Nouilles 5 kg	SAG	NOUILLE	A
	19209	EPI	SAG	Méditerranée	Nouilles - 5 Kg	SAG	NOUILLES	A
	13273	EPI	SAC	Panzani	Nouille fine qualité supérieure 5 kg	SAC	NOUILLE	
	13441	EPI	SCH	Suziwan	Nouilles chinoises aux oeufs 250 g	C12	NOUILLE	

Filtres | Limite de résultat : 100

Statut article:  Tous (incluant les annulés et suspendus)  Ouverts

Recherche élargie (multicritères): nouille

Positionnement: (articles trouvés: 51) 0 - 100



etc...

# Conclusion

## Bilan du projet

Le projet de développement d'une API Web ASP.NET Core RESTful pour le site interne E-Gescom de Pro à Pro a été une réussite pour moi. Cette initiative a permis de concrétiser la migration d'une architecture PHP existante vers une solution en C#, apportant des améliorations notables en termes de performance, de maintenabilité et de sécurité.

**Sécurité renforcée :** L'implémentation d'un middleware d'authentification par jeton (Bearer Token) a permis de sécuriser l'accès à l'API pour les utilisateurs internes, garantissant la confidentialité et l'intégrité des données sensibles.

**Amélioration des performances :** La transition vers ASP.NET Core devrait apporter des gains de performance par rapport à l'ancienne API PHP, améliorant la réactivité des applications utilisées par les employés.

Certes j'ai eu beaucoup de mal à démarrer mais ensuite avec des recherches sur internet et de l'aide de la part de toutes l'équipes de la DSI j'ai pu commencer et avancer sur ce projet qui m'a été donné pendant ma période de stage.

## Compétences acquises

Compréhension de l'architecture MVC améliorée et du langage c#.