



Projet de Génie Logiciel

Création d'une application web intégrant une API

2022 ~ 2023

Bernard Chloé et Degorre Anaïs

Sommaire

Sommaire	1
Introduction	2
Contexte	2
Thématique de l'application.	2
Modélisation des données	2
Installation de l'application	3
Api	5
Documentation de l'API	5
Commandes curl	9
Vues	12
Gestion de projet	14
Bilan	16

Introduction

Dans le cadre du module de Génie Logiciel, nous devons réaliser un projet consistant en la réalisation d'une application web qui constitue le back-end d'une future application mobile. Le choix du contexte métier est libre mais celle-ci doit vérifier des exigences métiers et techniques.

Ce projet se réalise en binôme et l'application fait l'objet d'une gestion des versions avec Git et son code source est partagé via la plateforme GitHub.

Contexte

Thématique de l'application.

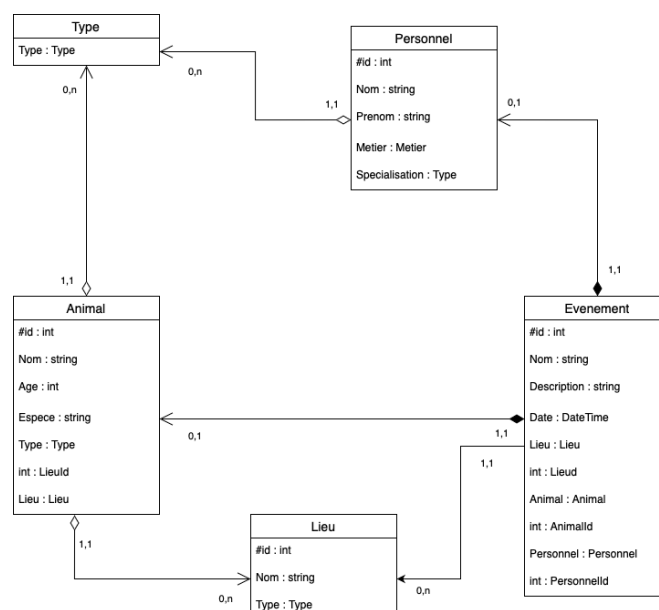
Pour le contexte métier, nous avons choisi de travailler sur la gestion d'une réserve naturelle. L'application contient donc les modèles métiers suivants :

- Animal
- Personnel
- Lieu
- Evenement

L'objectif de l'application est ainsi d'avoir accès aux données de la réserve et de pouvoir les modifier.

Modélisation des données

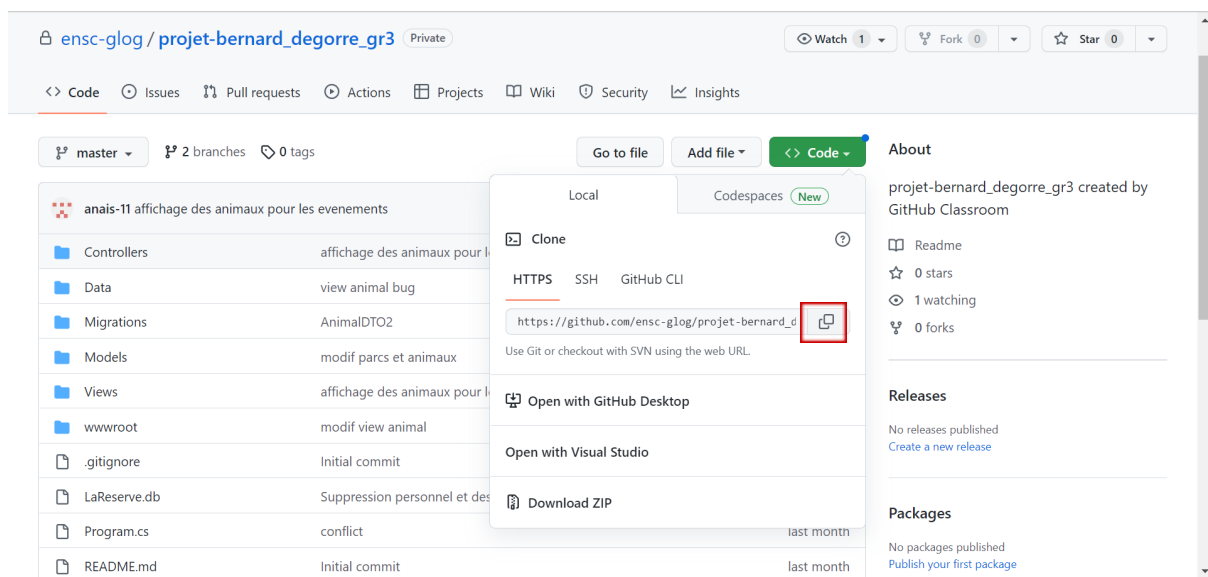
Nous avons choisi de modéliser notre application grâce à un diagramme de classes métier et un modèle relationnel suivants :



Installation de l'application

Pour installer et tester l'application, il est nécessaire d'accéder au dépôt du projet sur GitHub à l'adresse suivante : https://github.com/ensc-glog/projet-bernard_degorre_gr3. Lorsque la page du dépôt est ouverte, il vous suffit de cloner le dépôt tout d'abord en copiant le code en ouvrant l'onglet Code puis en appuyant sur le bouton suivant :

Figure 1 : Page du dépôt git du projet



Quand le lien est copié, ouvrez votre terminal et positionnez dans le dossier que vous voulez en utilisant la commande `cd`. Puis clonez le projet en utilisant la commande `git clone` suivie du lien copié précédemment.

Figure 2 : Terminal avec commandes `cd` et `git clone`

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

Installez la dernière version de PowerShell pour de nouvelles fonctionnalités et améliorations ! https://aka.ms/PSWindows

PS C:\Users\Chloé> cd .\Documents\
PS C:\Users\Chloé\Documents> git clone https://github.com/ensc-glog/projet-bernard_degorre_gr3.git
```

Lorsque ceci est effectué, vous pouvez ouvrir le projet sur Visual Studio Code. Il est nécessaire, suite à cela, d'installer les package suivants directement sur le terminal de Visual Studio Code en utilisant les commandes suivantes :

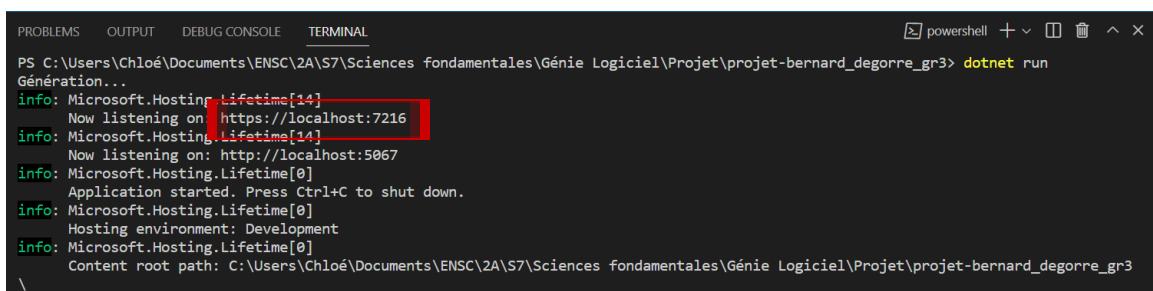
```
dotnet add package Microsoft.EntityFrameworkCore.Design
```

```
dotnet add package Microsoft.EntityFrameworkCore.Sqlite
```

Il vous faut également installer le fichier gitignore avec la commande : `new git ignore`

Vous pouvez maintenant lancer l'application avec la commande : `dotnet run`. Sélectionnez le premier lien affiché lors de l'exécution de la commande :

Figure 3 : Terminal avec la commande `dotnet run`



```
PS C:\Users\Chloé\Documents\ENSC\2A\S7\Sciences fondamentales\Génie Logiciel\Projet\projet-bernard_degorre_gr3> dotnet run
Génération...
info: Microsoft.Hosting.Lifetime[14] Now listening on https://localhost:7216
info: Microsoft.Hosting.Lifetime[14] Now listening on: http://localhost:5067
info: Microsoft.Hosting.Lifetime[0] Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0] Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0] Content root path: C:\Users\Chloé\Documents\ENSC\2A\S7\Sciences fondamentales\Génie Logiciel\Projet\projet-bernard_degorre_gr3
\
```

Api

Documentation de l'API

Voici les différentes méthodes utilisées pour gérer l'API, elles sont présentées par Contrôleur. Globalement, pour les méthodes **POST**, **PUT** et **DELETE**, celles-ci sont utilisées dans tous les contrôleurs pour pouvoir gérer l'API. Notre besoin était différent notamment dans l'affichage des éléments de l'API.

Figure 4 : Méthodes de AnimalApiController

Type de requête	Méthode	Rôle	Données en entrée	Données renvoyées
GET	GetAnimaux	Afficher la liste des animaux	/	Liste d'animaux
	GetAnimal	Afficher les caractéristiques d'un animal identifié par son Id	L'Id de l'animal	L'animal
	GetAnimauxParLieu	Affiche tous les animaux présents dans un lieu	L'Id d'un lieu	Liste d'animaux
	GetPersonnelParAnimal	Affiche tout le personnel pouvant s'occuper de l'animal identifié par son Id	L'Id de l'animal	Liste de Personnel
POST	PostAnimal	Permet d'ajouter un animal à l'API	animalDTO	Renvoie une action d'ajout de l'animal
PUT	PutAnimal	Permet de modifier les caractéristiques d'un animal	L'id de l'animal et animalDTO	Ne retourne rien mais modifie l'animal
DELETE	DeleteAnimal	Permet de supprimer un animal identifié par son id	L'id de l'animal	Ne retourne rien mais supprime l'animal

Comme nous pouvons le constater dans le tableau ci-dessus, nous avons décidé d'afficher tout d'abord la liste des animaux et un animal identifié par son id. Nous avons également

créé des méthodes faisant objet de filtres, notamment l'affichage des animaux par lieu. De plus, une méthode renvoie une liste de personnel selon l'Id d'un animal.

Figure 5 : Méthodes de *EvenementApiController*

Type de requête	Méthode	Rôle	Données en entrée	Données renvoyées
GET	GetEvenements	Afficher la liste des événements	/	Liste d'événements
	GetEvenement	Afficher les caractéristiques d'un événement identifié par son Id	L'Id de l'événement	L'événement
	GetEvenementParLieu	Affiche tous les événements d'un lieu	L'Id d'un lieu	Liste d'événements
POST	PostEvenement	Permet d'ajouter un événement à l'API	evenement DTO	Renvoie une action d'ajout de l'événement
PUT	PutEvenement	Permet de modifier les caractéristiques d'un événement	L'id de l'événement et evenement DTO	Ne retourne rien mais modifie l'événement
DELETE	DeleteEvenement	Permet de supprimer un événement identifié par son id	L'id de l'événement	Ne retourne rien mais supprime l'événement

Pour les événements, nous avons également choisi de les filtrer par lieu et d'afficher comme pour les animaux la liste des événements et un événement identifié par son id.

Figure 6 : Méthodes de LieuApiController

Type de requête	Méthode	Rôle	Données en entrée	Données renvoyées
GET	GetLieux	Afficher la liste des lieux	/	Liste des lieux
	GetEvenement	Afficher les caractéristiques d'un lieu identifié par son Id	L'id du lieu	Le lieu
	GetLieuAvecAnimaux	Affiche tous les lieux contenant des animaux	/	Liste de lieux
POST	PostLieu	Permet d'ajouter un lieu à l'API	Lieu	Renvoie une action d'ajout du lieu
PUT	PutLieu	Permet de modifier les caractéristiques d'un lieu	L'id du lieu et le lieu	Ne retourne rien mais modifie le lieu
DELETE	DeleteLieu	Permet de supprimer un lieu identifié par son id	L'id du lieu	Ne retourne rien mais supprime le lieu

Pour les lieux, nous avons choisi un filtre différent que pour les précédents contrôleurs, en effet, nous décidons d'afficher tous les lieux, un lieu par id mais également tous les lieux contenant des animaux.

Figure 7 : Méthodes de PersonnelApiController

Type de requête	Méthode	Rôle	Données en entrée	Données renvoyées
GET	GetPersonnels	Afficher la liste du personnel	/	Liste du personnel
	GetPersonnel	Afficher les caractéristiques d'une personne identifiée par son Id	L'id du personnel	La personne faisant partie du personnel
	GetAnimauxParPersonnel	Affiche tous les animaux dont peut s'occuper le personnel identifié par son id	L'id du personnel	Liste des animaux
	GetLieuParPersonnel	Affiche tous les lieux où peut travailler un personnel identifié par son id	L'id du personnel	Liste des lieux
POST	PostPersonnel	Permet d'ajouter un personnel à l'API	Personnel	Renvoie une action d'ajout du personnel
PUT	PutPersonnel	Permet de modifier les caractéristiques d'un personnel	L'id du personnel et le personnel	Ne retourne rien mais modifie le personnel
DELETE	DeletePersonnel	Permet de supprimer un personnel identifié par son id	L'id du personnel	Ne retourne rien mais supprime le personnel

Comme pour les animaux, nous affichons le personnel par lieu et nous affichons également les animaux pris en charge par le personnel identifié par son Id.

Commandes curl

Tout d'abord, concernant les commandes proposées ci-dessous, il est nécessaire de vérifier le port du localhost que vous utilisez, en effet, il sera sûrement différent de celui que nous vous proposons dans les différentes requêtes.

Tout d'abord, voici les chemins à utiliser pour les requêtes **GET**.

Figure 8 : Chemins pour accéder aux requêtes permettant d'afficher tous les éléments d'une table

Liste des animaux	https://localhost:7216/api/AnimalApi
Liste des lieux	https://localhost:7216/api/LieuApi
Liste du personnel	https://localhost:7216/api/PersonnelApi
Liste des événements	https://localhost:7216/api/EvenementApi

Puis nous avons décidé d'afficher chaque élément en utilisant leur identifiant en utilisant le chemins suivants :

Figure 9 : Chemins pour accéder aux requêtes permettant d'afficher un élément par Id

Détails d'un animal	https://localhost:7216/api/AnimalApi/{id}
Détails d'un lieu	https://localhost:7216/api/LieuApi/{id}
Détails d'un personnel	https://localhost:7216/api/PersonnelApi/{id}
Détails d'un événement	https://localhost:7216/api/EvenementApi/{id}

Concernant les animaux, nous avons décidé également de les afficher par lieu, pour cela il suffit de prendre le chemin suivant :

<https://localhost:7216/api/AnimalApi/parlieu>

Puis, pour afficher tout le personnel pouvant s'occuper de l'animal identifié par son id :

<https://localhost:7216/api/AnimalApi/{id}/PrisenchargePar>

Afin d'afficher les animaux pouvant être pris en charge par le personnel, il suffit de suivre le lien suivant :

<https://localhost:7216/api/PersonnelApi/{id}/PeutSOccuperDe>

Pour afficher les lieux possédant des animaux :

<https://localhost:7216/api/LieuApi/avecanimaux>

Pour afficher les lieux où peut travailler un personnel :

<https://localhost:7216/api/PersonnelApi/{id}/PeutTravailler>

Les commandes suivantes nous ont permis de tester l'efficacité des requêtes **POST** :

Figure 10 : Commandes CURL pour les requêtes POST

Ajout d'un animal	<code>curl -X POST -k -H 'Content-Type: application/json' -i https://localhost:7216/api/AnimalApi --data '{"Nom": "Lilou", "Age": "3", "Espece": "Lion", "Type": 0, "Lieuld": 2}'</code>
Ajout d'un lieu	<code>curl -X POST -k -H 'Content-Type: application/json' -i http://localhost:20851/api/LieuApi --data '{"Nom": "La Volière", "Type": 4}'</code>
Ajout de personnel	<code>curl -X POST -k -H 'Content-Type: application/json' -i https://localhost:7216/api/PersonnelApi --data '{"Nom": "Boulet", "Prenom": "Bertrand", "Metier": 2, "Type": 4}'</code>
Ajout d'un événement	<code>curl -X POST -k -H 'Content-Type: application/json' -i http://localhost:20851/api/EvenementApi --data '{"Nom": "Le coucher du soleil", "Description": "Profitez d'un beau coucher du soleil dans la savane", "Date": "2023-01-14T18:00:00", "Type": 4, "Lieuld": 2, "Animalld": 4, "Personnelld": 4}'</code>

Nous utilisons une nouvelle fois l'identifiant pour appliquer les requêtes **PUT**, voici des exemples de modifications effectuées :

Figure 11 : Commandes CURL pour les requêtes PUT

Modification d'un animal (Augmenter l'âge de Lili)	<code>curl -X PUT -k -H 'Content-Type: application/json' -i https://localhost:7216/api/AnimalApi/5 --data '{"Id": 5, "Nom": "Lili", "Age": "3", "Espece": "Panda",</code>
---	--

	"Type" : 5, "Lieuid": 2}'
Modification d'un événement (Changement d'horaire pour "La Nuit étoilée")	curl -X PUT -k -H 'Content-Type: application/json' -i https://localhost:7216/api/EvenementApi/4 --data '{ "Id": 4, "Nom": "Nuit étoilée", "Description": "Venez découvrir le spectacle au coeur de la planète des singes lors d'une nuit étoilée...", "Date": "2023-01-14T22:00:00", "Lieuid": 4, "AnimalId":2, "PersonnelId" : 2}'
Modification de personnel (changer la spécialisation de Camerez)	curl -X PUT -k -H 'Content-Type: application/json' -i http://localhost:20851/api/PersonnelApi/7 --data '{"Id": 7, "Nom": "Camerez", "Prenom": "Bob", "Metier": 3, "Specialisation":1 }'
Modification d'un lieu (changer le type d'animaux)	curl -X PUT -k -H 'Content-Type: application/json' -i https://localhost:7216/api/LieuApi/5 --data '{ "Id": 5, "Nom": "La Forêt", "Type": 4}'

Pour finir, nous avons fait des requêtes **DELETE** permettant de supprimer des éléments de l'API, pour vérifier nos requêtes, nous effectuons les commandes suivantes :

Figure 12 : Commandes CURL pour les requêtes DELETE

Suppression d'un animal	curl -X DELETE -k -H 'Content-Type: application/json' -i https://localhost:7216/ api/AnimalApi/{id}
Suppression d'un événement	curl -X DELETE -k -H 'Content-Type: application/json' -i https://localhost:7216/ api/EvenementApi/{id}
Suppression de personnel	curl -X DELETE -k -H 'Content-Type: application/json' -i https://localhost:7216/ api/PersonnelApi/{id}
Suppression d'un lieu	curl -X DELETE -k -H 'Content-Type: application/json' -i https://localhost:7216/ api/LieuApi/{id}

Vues

Pour les Vues, voici les différentes méthodes utilisées dans les contrôleurs et leur utilisation.

Figure 13 : Méthodes de AnimalController

Méthode	Rôle
Index	Afficher la liste des animaux
Details	Afficher les détails sur un animal en fonction de l'Id
Create	Affiche la vue Create de Animal
Create([Bind("Id,Nom,Age,Espece,Type,LieuId")] Animal animal)	Permet d'ajouter un animal
ConfirmDeleteAnimal	Permet la confirmation de suppression de l'animal
DeleteAnimal	Permet de supprimer un animal
Modif	Permet d'afficher la Vue de modification d'un Animal
Modif([Bind("Id,Nom,Age,Espece,Type,LieuId")] AnimalDTO animalDTO)	Permet de modifier un animal
FiltrerLieux(int id)	Permet d'afficher les animaux en fonction d'un lieu

Figure 14 : Méthodes de EvenementController

Méthode	Rôle
Index	Afficher la liste des événements
Details	Afficher les détails sur un événement en fonction de l'Id
Create	Affiche la vue Create de Evenement
Create([Bind("Id,Nom,Description,Date,LieuId,Animaux,Personnel")] Evenement evenement)	Permet d'ajouter un événement
ConfirmDeleteEvenement	Permet la confirmation de suppression de l'animal
DeleteEvenement	Permet de supprimer un événement
Modif	Permet d'afficher la Vue de modification d'un événement
Modif([Bind("Id,Nom,Description,Date,LieuId,Animaux,Personnel")] Evenement evenement)	Permet de modifier l'événement

Figure 15 : Méthodes de LieuController

Méthode	Rôle
Index	Afficher la liste des lieux
Details	Afficher les détails sur un lieu en fonction de l'Id
Create	Affiche la vue Create de Lieu
Create([Bind("Id,Nom,Type")] Lieu lieu)	Permet d'ajouter un lieu
ConfirmDeleteLieu	Permet la confirmation de suppression du lieu
DeleteLieu	Permet de supprimer un lieu
Modif	Permet d'afficher la Vue de modification d'un lieu
Modif([Bind("Id,Nom,Type")] Lieu lieu)	Permet de modifier le lieu

Figure 16 : Méthodes de PersonnelController

Méthode	Rôle
Index	Afficher la liste du Personnel
Details	Afficher les détails sur un personnel en fonction de l'Id
Create	Affiche la vue Create de Personnel
Create([Bind("Id,Nom,Prenom,Metier,Specialisation")] Personnel personnel)	Permet d'ajouter un personnel
ConfirmDeletePersonnel	Permet la confirmation de suppression d'un Personnel
DeletePersonnel	Permet de supprimer un Personnel
Modif	Affiche la vue de modification d'un Personnel
Modif([Bind("Id,Nom,Prenom,Metier,Specialisation")] Personnel personnel)	Permet de modifier les informations sur un personnel

Gestion de projet

Pour ce projet, nous avons tout d'abord utilisé les séances de TD pour mettre en place notre application, nous avons pu avancer notamment sur la création des modèles, de l'API et des Vues. Pour cela, nous avons réparti notre travail en s'occupant chacune de notre côté de deux tables. Nous en avons également profité pour créer le contexte BD permettant de créer la base de données. Par la suite, nous avons décidé de séparer le travail de la manière suivante :

- Création des requêtes de l'API : Chloé
- Création des requêtes des vues : Anaïs

Nous étions toutefois en contact pendant toute la durée du projet pour nous aider mutuellement en cas de problèmes. Nous avons donc pris part d'une certaine manière à tout le projet ce qui nous a permis d'avoir une vue globale de notre avancement.

Voici notre planning :

	Semaine	21/11 - 27/11	28/11 - 4/12	5/12 - 11/12	12/12 - 18/12	19/12 - 25/12	26/12 - 01/01	02/01 - 08/01
API	Réflexion sur le thème							
	Création des modèles							
	Création du contexte BD							
	Ajout de données							
	Création de requêtes d'API							
Vues	Création des vues Index							
	Création des vues Details							
	Création des vues Create							
	Création des vues Delete							
	Création des vues Modif							
Rapport	Rédaction du rapport							

Nous avons eu quelques difficultés à trouver du temps pour programmer pendant les vacances, pour optimiser le temps, nous avons donc commencé à rédiger le rapport.

Bilan

Globalement, nous sommes satisfaites de notre organisation et de la quantité de travail effectuée pour ce projet. Nous avons toutefois eu des difficultés notamment à gérer le modèle métier **Evenement**. En effet, la gestion des listes était plus complexe que prévue, pour obtenir une application qui fonctionne toutefois, nous avons décidé de limiter l'événement à la participation d'un seul animal et d'un seul personnel.

Nous espérons également avoir travaillé de la manière la plus optimale pour continuer ce projet et pour créer l'application mobile reliée à notre API.