

Rapport d'activité

Atelier De Professionnalisation

Table des matières

Introduction.....	2
1. Résumé du Contexte	2
2. Mission Globale à Effectuer	2
Etapes De Développement.....	2
Première étape.....	2
i. Mise En place de L'environnement logiciel.....	2
ii. Mise en place de la Base de Données	4
Deuxième étape	14
1. Prototypage de l'interface Graphique.....	14
Cas d'utilisation 1 et 2 (Authentification):	15
Cas d'utilisation 2 (Ajout d'un Personnel) :	15
Cas d'utilisation 3 (Suppression d'un Personnel) :	16
2. Création de l'application dans L'IDE.....	17
3. Développement de l'interface.....	19
Troisième étape.....	23
1. Connexion à la BDD	23
2. Codage du modèle.....	25
3. Codage du package « Connexion ».....	27
4. Codage du package « Data Access Layer »	30
5. Génération de la Documentation Technique	31
Quatrième Étape	32
1. Cas D'utilisation 1 (l'authentification).....	33
2. Cas D'utilisation 2 : Ajout d'un personnel	34
3. Cas D'utilisation 3 : Suppression d'un Personnel	34
4) Cas d'utilisation 4 (Modification d'un personnel)	36
5) Cas d'utilisation (affichage des absences).....	38
6) Cas d'utilisation 6 (ajout d'absence)	40
7) Cas d'utilisation 4 (suppression d'une absence)	41
8) Cas d'utilisation 8 (.....	41

Rapport d'activité

Atelier De Professionnalisation

Introduction

1. Résumé du Contexte

Actuellement en poste de développeur junior à l'ESN InfoTech Services 86, J'ai pour mission le développement d'une application de bureau, permettant de gérer le personnel de chaque médiathèque du réseau MediaTek86 (toutes les médiathèques de la ville de Vienne).

2. Mission Globale à Effectuer

Afin d'accomplir cette tâche, il faut déjà choisir la pile applicative qui sera utilisée au cours du développement. J'ai personnellement fait le choix de coder l'application en C#, étant plus à l'aise avec celui-ci qu'avec JAVA. Concernant la partie SGBDR, un environnement WAMP est installé sur poste, avec une base de données en MySQL pour la persistance des informations.

Etapes De Développement

Première étape

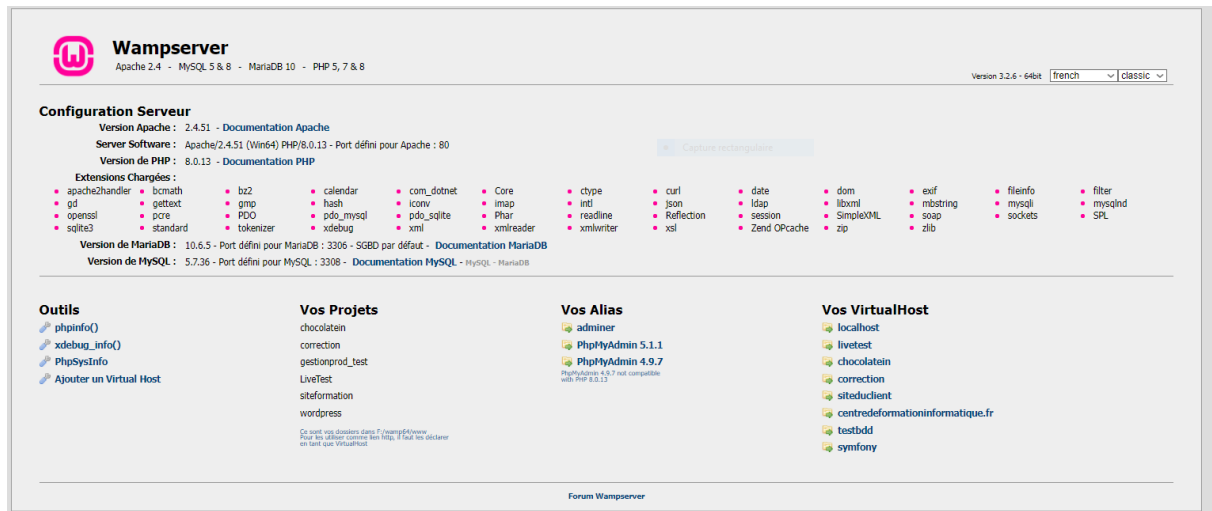
i. Mise En place de L'environnement logiciel

Pour cette première étape, il va tout d'abord falloir mettre en place l'environnement de travail. Voici les différentes étapes pour ce faire :

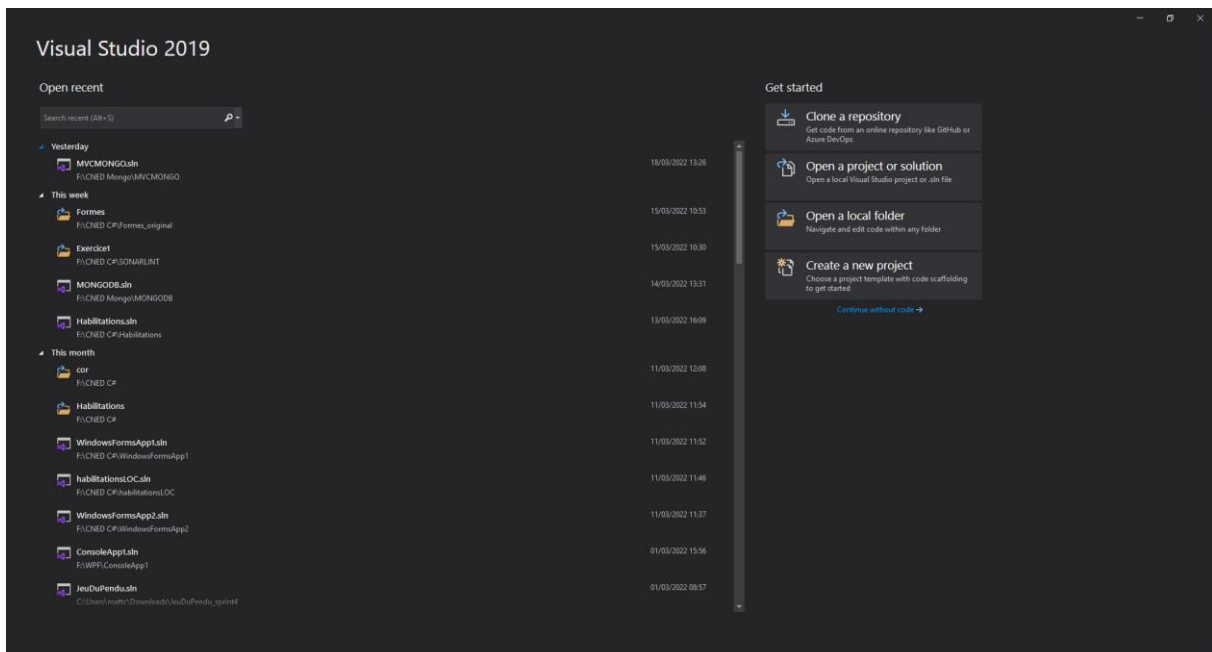
- Il faut installer Wampserver, qui va nous permettre d'héberger localement la BDD et de la gérer avec une interface utilisateur grâce à phpMyAdmin (Si cela est nécessaire).

Rapport d'activité

Atelier De Professionnalisation



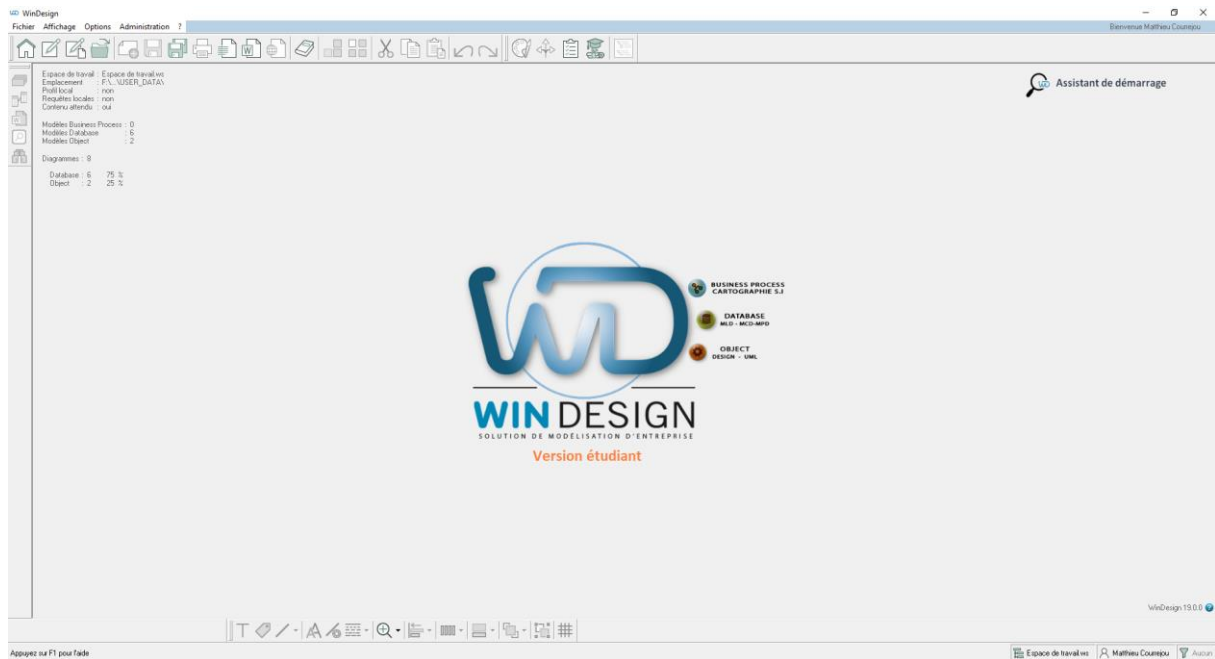
- Utilisant le langage C# pour le développement, il faut installer l'IDE correspondant, qui n'est autre que Visual Studio :



- Il ne reste à installer qu'un logiciel de modélisation, qui nous permettra de visualiser notre modèle conceptuel de données. On choisit ici d'utiliser WinDesign :

Rapport d'activité

Atelier De Professionnalisation

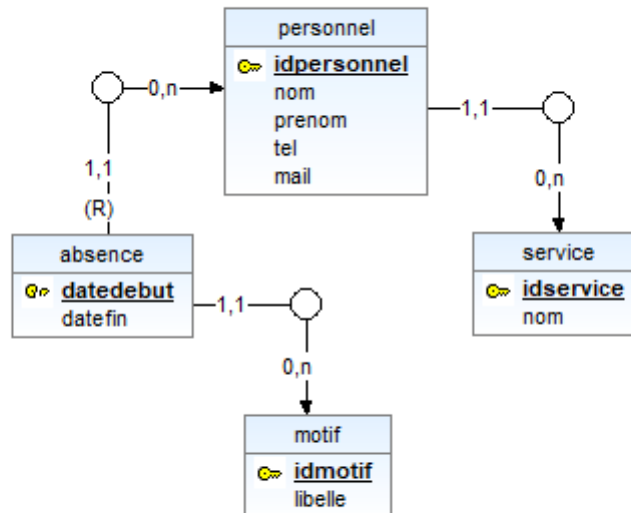


ii. Mise en place de la Base de Données

Nous avons à notre disposition un schéma conceptuel de données préalablement établi (ci-dessous). Le Logiciel Win Design nous permet de générer un script de base de données à partir de celui-ci, et dans le type de base de données désiré.

Rapport d'activité

Atelier De Professionnalisation



Voici la démarche suivie pour obtenir le script au format MySQL :

- Générons tout d'abord le Modèle Logique de Données correspondant au langage cible,

Rapport d'activité

Atelier De Professionnalisation

En corrigeant d'éventuelles « erreurs » de traduction.

Génération d'un Modèle Logique de Données

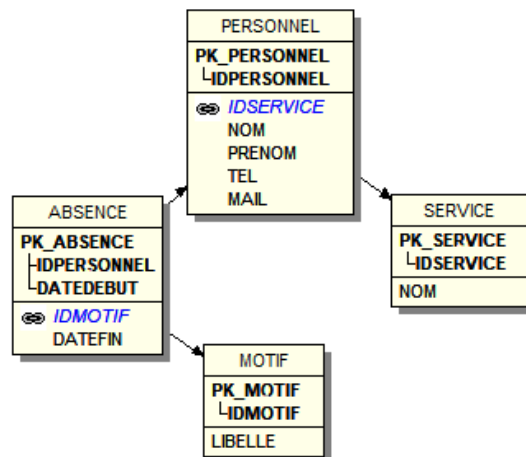


Sélection du modèle destination		OK
<input checked="" type="checkbox"/> Nouveau modèle		Annuler
SGBD courant : MySQL		Aide
SGBD		
Options générales	Options d'historisation	
<input checked="" type="checkbox"/> Conserver les modifications	<input checked="" type="checkbox"/> Montrer les tables d'historisation	
<input checked="" type="checkbox"/> Utilisation des noms logiques	Nom attribut datation : DATE_HISTO	
<input checked="" type="checkbox"/> Recherche objets par codes internes		
<input checked="" type="checkbox"/> Générer les préfixes attributs		
<input type="checkbox"/> Générer table si cardinalité (0,1-x,N)		
<input checked="" type="checkbox"/> Générer les propriétés composées		
<input checked="" type="checkbox"/> Générer les rôles des liens		
<input checked="" type="checkbox"/> Mise à jour des textes libres		
<input checked="" type="checkbox"/> Transformer les informations		
<input checked="" type="checkbox"/> Transformer les règles		
<input checked="" type="checkbox"/> Transformer les vues externes		
<input type="checkbox"/> Utiliser le code relation dans le code des clé étrangères		
<input checked="" type="checkbox"/> Générer tables de code		
Seuil création index : <input type="text"/>		
Seuil multivaluées : <input type="text" value="5"/>		
Sauf...	Contrôle à appliquer	
	Contrôle de base	
	Contrôle documentation	
	Contrôle infos administration	
	Contrôle passage au logique	
	Voir...	
	<input type="checkbox"/> Générer l'attribut clé étrangère en conservant le préfixe de la table mère avec celui de la table fille	

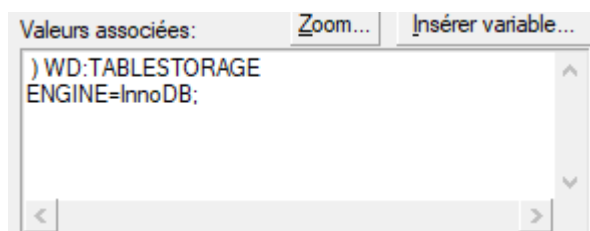
Rapport d'activité

Atelier De Professionnalisation

- Voici le MLD correspondant. Le MCD étant simple, il ne semble pas y avoir de problèmes de traduction. Les relations sont respectées, et tous les champs sont au bon format.



- Nous pouvons désormais générer le script MySQL.
Nous devons nous assurer que nous utilisons bien l'engine InnoDB, et pas celui par défaut de MySQL (MyISAM). En effet, ce-dernier ne gère pas les clés étrangères et les contraintes d'intégrité.



Rapport d'activité

Atelier De Professionnalisation

- Le Script maintenant généré, nous l'ouvrons dans un éditeur de texte (Visual Studio Code) afin d'en vérifier le contenu avant exécution. Vu que nous allons créer la base de données sur phpMyAdmin, nous pouvons supprimer les trois premières lignes qui sont inutiles. Le Script est dorénavant prêt.

```
PERSONNEL.SQL
1 DROP DATABASE IF EXISTS MLR1;
2
3 CREATE DATABASE IF NOT EXISTS MLR1;
4 USE MLR1;
5 # -----
6 #      TABLE : ABSENCE
7 # -----
8
9 CREATE TABLE IF NOT EXISTS ABSENCE
10 (
11     IDPERSONNEL INTEGER(2) NOT NULL ,
12     DATEDEBUT DATETIME NOT NULL ,
13     IDMOTIF INTEGER(2) NOT NULL ,
14     DATEFIN DATETIME NULL
```

- Nous avons créé une base de données MySQL « personnel » sur phpMyAdmin, en veillant à ce que l'encodage soit en utf8_unicode, afin d'éviter toute erreur. Nous créons les tables en important le script généré plus tôt.

The screenshot displays the phpMyAdmin interface with the 'Structure' tab selected. It shows a series of successful SQL execution results for the 'PERSONNEL.SQL' script. Each result line indicates that MySQL returned an empty result set, meaning no data was inserted, only the schema was created. The executed SQL commands are as follows:

- Database Creation:** `CREATE DATABASE IF NOT EXISTS ABSENCE (IDPERSONNEL INTEGER(2) NOT NULL , DATEDEBUT DATETIME NOT NULL , IDMOTIF INTEGER(2) NOT NULL , DATEFIN DATETIME NULL , PRIMARY KEY (IDPERSONNEL,DATEDEBUT)) ENGINE=InnoDB;`
- Index Creation:** `CREATE INDEX I_FK_ABSENCE_PERSONNEL ON ABSENCE (IDPERSONNEL);`
- Table Creation:** `CREATE TABLE IF NOT EXISTS ABSENCE (IDPERSONNEL INTEGER(2) NOT NULL , DATEDEBUT DATETIME NOT NULL , IDMOTIF INTEGER(2) NOT NULL , DATEFIN DATETIME NULL , PRIMARY KEY (IDPERSONNEL,DATEDEBUT)) ENGINE=InnoDB;`
- Index Creation:** `CREATE INDEX I_FK_ABSENCE_PERSONNEL ON ABSENCE (IDPERSONNEL);`
- Table Creation:** `CREATE TABLE IF NOT EXISTS NOTIF (IDMOTIF INTEGER(2) NOT NULL , AUTO_INCREMENT , LIBELLE VARCHAR(128) NULL , PRIMARY KEY (IDMOTIF)) ENGINE=InnoDB;`
- Index Creation:** `CREATE INDEX I_FK_ABSENCE_PERSONNEL ON ABSENCE (IDPERSONNEL);`
- Table Creation:** `CREATE TABLE IF NOT EXISTS SERVICE (IDSERVICE INTEGER(2) NOT NULL , AUTO_INCREMENT , NOM VARCHAR(50) NULL , PRIMARY KEY (IDSERVICE)) ENGINE=InnoDB;`
- Index Creation:** `CREATE INDEX I_FK_ABSENCE_PERSONNEL ON ABSENCE (IDPERSONNEL);`
- Table Creation:** `CREATE TABLE IF NOT EXISTS PERSONNEL (IDPERSONNEL INTEGER(2) NOT NULL , AUTO_INCREMENT , IDSERVICE INTEGER(2) NOT NULL , NOM VARCHAR(50) NULL , PRENOM VARCHAR(50) NULL , TEL VARCHAR(15) NULL , MAIL VARCHAR(128) NULL , PRIMARY KEY (IDPERSONNEL)) ENGINE=InnoDB;`
- Index Creation:** `CREATE INDEX I_FK_PERSONNEL_SERVICE ON PERSONNEL (IDSERVICE);`

Each execution result is followed by a link to 'Créer le code source PHP'.

Rapport d'activité

Atelier De Professionnalisation

- Avant de commencer à insérer des données, ou bien créer des utilisateurs, nous vérifions que les tables correspondent bien aux MLD.
Tout étant conforme aux attentes, il est temps de créer un utilisateur avec accès à la BDD, afin de ne pas travailler avec un compte admin, ce qui est dangereux et fortement déconseillé.

On crée l'utilisateur dans la base de donnée, en indiquant son mot de passe.

```
CREATE USER 'personnel'@'%' IDENTIFIED BY 'motdepasse';|
```

On lui donne l'accès « ALL PRIVILEGES » à toutes les tables de la BDD « personnel ».
ALL PRIVILEGES confère les droits SELECT, INSERT, UPDATE, DELETE sur les champs des tables.

Il permet aussi d'autres manipulations comme la création et la modification des tables.

Nous n'allons pas conférer à cet utilisateur la capacité de transmettre ses droits « GRANT ».

```
1 GRANT ALL PRIVILEGES ON personnel.* TO 'personnel'@'%'
```

Connectons-nous à phpMyAdmin avec se compte, et vérifions nos privilèges :



Nous n'avons accès qu'à la BDD personnel, et nous ne pouvons pas créer d'autres BDD, ni supprimer l'actuelle.

Nous pouvons donc passer à l'étape suivante : Créer une table responsable qui contient 2 champs (login et pwd) :

Rapport d'activité

Atelier De Professionnalisation

```
CREATE TABLE responsable (  
    login VARCHAR(64) NOT NULL,  
    pwd VARCHAR(64) NOT NULL  
);
```

On précise que le ni le login ni le mot de passe ne doivent être nulls, sinon nous risquons de rencontrer des problèmes au moment de l'authentification dans l'application.

```
INSERT INTO responsable VALUES('responsable', SHA2('motdepasse', 256))
```

Le mot de passe est hashé avec la fonction SHA2(_,256) disponible directement dans MySQL. Voilà ce qu'on obtient dans la table « Responsable ».

login	pwd
responsable	967520ae23e8ee14888bae72809031b98398ae4a636773e18f...

Afin de remplir la table « motif », nous effectuons plusieurs INSERT sous cette forme :

```
INSERT INTO motif(libelle) VALUES( 'vacances')
```

Voici la table complétée :

				IDMOTIF	LIBELLE
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	1	vacances
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	2	maladie
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	3	motif familial
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	4	congé parental

Voici la table service complétée, avec des requêtes similaires :

				IDSERVICE	NOM
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	1	administratif
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	2	médiation culturelle
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	3	prêt

Rapport d'activité

Atelier De Professionnalisation

Nous devons maintenant générer des inserts pour les tables « personnel » et « absence ».
Les absences étant identifiées par les employés, nous allons commencer par peupler la table des employés :

```
INSERT INTO `personnel` (`idpersonnel`, `idservice`, `nom`, `prenom`, `tel`, `mail`)
VALUES
(1,2,"Lynn","Nell","01 00 30 32 24","lacus.quisque@aol.com"),
(2,1,"Bruce","Melyssa","09 33 77 81 78","nulla@hotmail.org"),
(3,3,"Ratliff","Lana","08 27 57 22 41","mus.donec.dignissim@outlook.edu"),
(4,1,"Parks","Trevor","07 78 58 16 55","mauris.nulla@icloud.com"),
(5,2,"Mccormick","Stewart","03 91 79 22 14","ut.semper@yahoo.org"),
(6,2,"Harper","Griffin","05 29 16 20 74","justo.praesent@google.org"),
(7,2,"Moon","Reuben","04 62 76 64 62","nullam@aol.org"),
(8,3,"Carlson","Chastity","05 18 71 24 11","aliquam.ultrices@aol.ca"),
(9,1,"Wilkerson","Scarlet","03 17 56 22 38","nullam@hotmail.net"),
(10,2,"Gilliam","Tanya","07 31 11 83 56","ultrices@aol.ca");
```

	IDPERSONNEL	IDSERVICE	NOM	PRENOM	TEL	MAIL
<input type="checkbox"/> Éditer Copier Supprimer	1	2	Lynn	Nell	01 00 30 32 24	lacus.quisque@aol.com
<input type="checkbox"/> Éditer Copier Supprimer	2	1	Bruce	Melyssa	09 33 77 81 78	nulla@hotmail.org
<input type="checkbox"/> Éditer Copier Supprimer	3	3	Ratliff	Lana	08 27 57 22 41	mus.donec.dignissim@outlook.edu
<input type="checkbox"/> Éditer Copier Supprimer	4	1	Parks	Trevor	07 78 58 16 55	mauris.nulla@icloud.com
<input type="checkbox"/> Éditer Copier Supprimer	5	2	Mccormick	Stewart	03 91 79 22 14	ut.semper@yahoo.org
<input type="checkbox"/> Éditer Copier Supprimer	6	2	Harper	Griffin	05 29 16 20 74	justo.praesent@google.org
<input type="checkbox"/> Éditer Copier Supprimer	7	2	Moon	Reuben	04 62 76 64 62	nullam@aol.org
<input type="checkbox"/> Éditer Copier Supprimer	8	3	Carlson	Chastity	05 18 71 24 11	aliquam.ultrices@aol.ca
<input type="checkbox"/> Éditer Copier Supprimer	9	1	Wilkerson	Scarlet	03 17 56 22 38	nullam@hotmail.net
<input type="checkbox"/> Éditer Copier Supprimer	10	2	Gilliam	Tanya	07 31 11 83 56	ultrices@aol.ca

Rapport d'activité

Atelier De Professionnalisation

```
INSERT INTO `absence` (`idpersonnel`,`datedebut`,`idmotif`,`datefin`)
VALUES
  (10,"2021-08-29 00:53:47",2,"2021-04-27 01:08:41"),
  (1,"2022-09-21 07:45:55",4,"2021-09-12 21:48:01"),
  (6,"2022-04-25 08:54:04",3,"2021-07-21 21:25:07"),
  (3,"2021-07-20 10:17:48",1,"2022-01-05 17:23:01"),
  (5,"2022-09-17 01:33:59",3,"2021-06-28 03:54:52"),
  (7,"2021-12-06 05:44:56",1,"2022-11-27 11:54:58"),
  (9,"2022-10-19 10:47:56",1,"2022-01-21 22:06:05"),
  (6,"2022-10-01 14:20:54",3,"2023-03-15 23:53:10"),
  (4,"2022-02-28 01:14:23",1,"2022-07-30 00:14:22"),
  (7,"2021-11-13 00:11:01",2,"2023-02-18 13:44:20");
```

Rapport d'activité

Atelier De Professionnalisation

				IDPERSONNEL	DATEDEBUT	IDMOTIF	DATEFIN
<div> <div>← T →</div> <div> <div>Éditer</div> <div>Copier</div> <div>Supprimer</div> </div> </div> <div>Cliquer sur la flèche pour afficher/masquer la colonne.</div>				1	2022-10-21 19:13:27	2	2021-11-11 07:19:48
				1	2023-03-15 15:33:07	3	2022-09-28 10:20:16
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	2021-09-29 17:10:50	2	2021-12-11 14:08:15
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	2021-10-22 16:43:52	3	2021-06-08 23:03:44
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	2021-11-07 18:45:41	2	2022-06-17 09:16:01
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	2022-06-22 05:27:55	2	2021-12-29 16:27:56
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	2023-01-29 16:06:07	2	2021-09-23 09:28:30
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	2023-03-14 19:07:24	1	2022-06-10 07:36:47
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	2021-04-26 07:26:14	3	2021-07-22 11:31:18
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	2022-01-23 21:16:24	2	2022-04-17 06:54:49
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	2022-04-06 22:35:38	2	2023-01-21 02:58:52
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	2022-06-17 12:49:45	3	2021-04-20 02:51:19
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	2023-01-31 07:50:47	2	2021-11-17 20:17:23
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	2023-02-18 21:55:37	1	2022-02-27 15:57:06
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	2021-05-19 06:31:30	3	2021-07-13 06:04:49
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	2021-08-30 07:49:48	2	2023-02-09 02:25:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	2023-01-22 05:34:46	2	2022-12-17 14:56:07
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	2021-05-05 00:32:42	2	2022-07-10 21:58:54
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	2021-08-01 06:27:12	3	2022-05-01 16:30:06
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	2021-12-05 00:49:47	4	2022-03-19 00:02:27
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	2021-12-17 03:52:03	4	2021-06-06 02:23:04
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	2022-03-16 16:17:12	2	2022-02-09 09:30:17
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	2022-04-12 22:59:56	3	2022-10-29 03:43:48
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	2022-07-20 00:03:49	1	2022-08-15 14:53:01
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	2022-12-29 01:22:22	1	2021-06-17 21:43:35
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	2021-12-25 08:04:57	4	2021-12-22 09:44:00
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	2022-06-08 04:52:40	2	2022-01-09 23:29:50
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	2022-07-11 16:58:15	1	2022-08-01 22:09:41
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	2022-12-06 04:41:48	2	2022-02-07 05:00:09
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	2022-12-10 15:01:26	1	2021-08-20 23:29:24
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	2022-12-21 23:19:59	2	2022-02-18 20:22:38

Bilan :

L'environnement de travail est fonctionnel, la base de données est remplie avec des informations correctes et est sécurisée.

Rapport d'activité

Atelier De Professionnalisation

Deuxième étape

1. Prototypage de l'interface Graphique

Avec l'aide du dossier documentaire, et des cas d'utilisation de l'application, nous pouvons dorénavant dessiner l'interface graphique du logiciel. Ici deux options s'offrent à nous :

- Directement prototyper l'interface sur Visual Studio, car l'onglet « designer » de WinForms est très performant et souple.
- Designer une maquette sur un logiciel spécialisé comme Figma, Sketch ou Pencil.

Dans la mesure où il m'est demandé dans les consignes de l'étape 2 de prototyper sur un logiciel prévu à cet effet, j'opte pour l'option numéro 2.

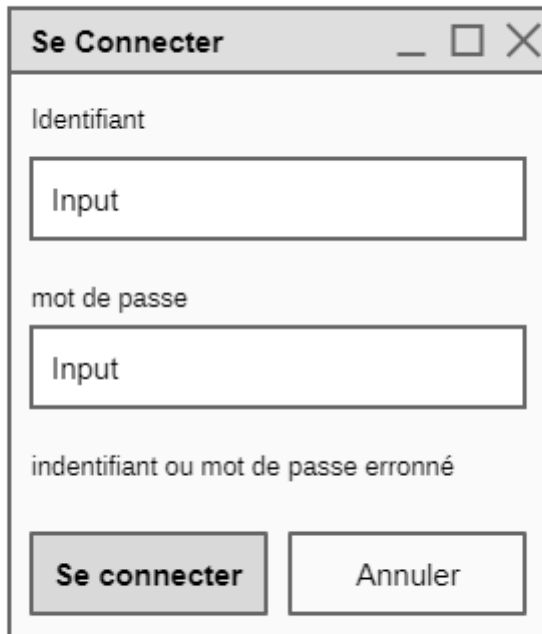
La maquette servira de ligne directrice pour l'interface du projet. En effet, Pencil, le logiciel de maquettage ne disposant pas des composants Windows Forms de Visual Studio, le projet fini ne sera pas identique à la maquette au niveau de la charte graphique.

Toutes les fonctionnalités n'étant disponibles qu'après authentification, commençons par créer une Vue de connexion :

Rapport d'activité

Atelier De Professionnalisation

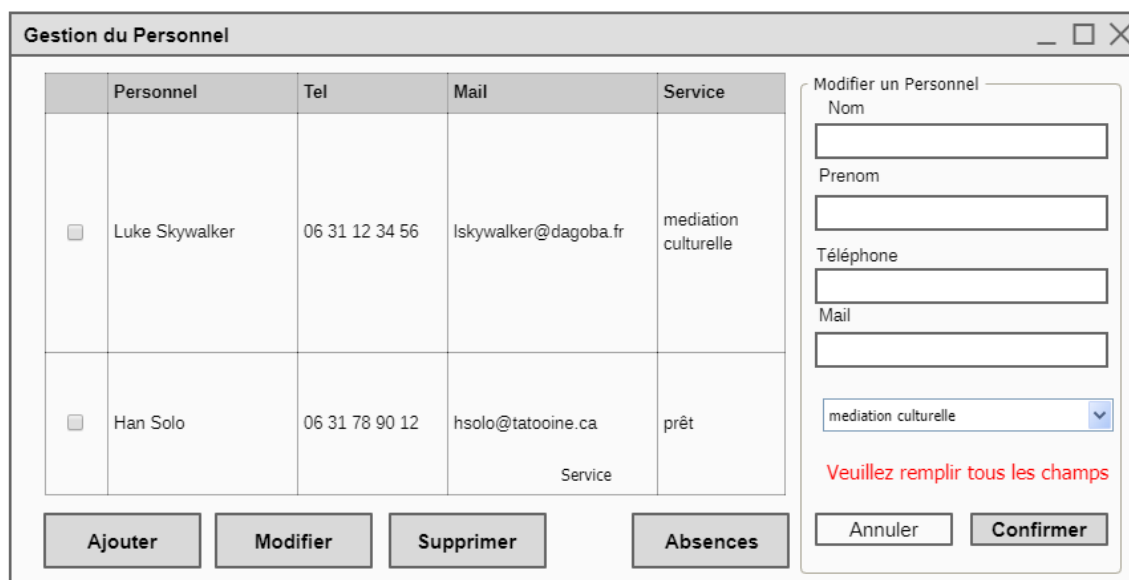
Cas d'utilisation 1 et 2 (Authentication):



A login window titled "Se connecter" with a standard window control bar. It contains two input fields: "Identifiant" and "mot de passe", both with "Input" as placeholder text. Below the inputs is a message "indentifiant ou mot de passe erronné". At the bottom are two buttons: "Se connecter" and "Annuler".

On va regrouper les trois premières fonctionnalités sur une seule Vue, afin de rendre le logiciel plus accessible et simple.

Cas d'utilisation 2 (Ajout d'un Personnel) :



A window titled "Gestion du Personnel" with a standard window control bar. It features a table with columns: Personnel, Tel, Mail, and Service. The table contains two rows: Luke Skywalker (mediation culturelle) and Han Solo (prêt). Below the table are four buttons: "Ajouter", "Modifier", "Supprimer", and "Absences". To the right is a "Modifier un Personnel" panel with input fields for Nom, Prenom, Téléphone, and Mail, a dropdown menu for Service (currently showing "mediation culturelle"), a red message "Veuillez remplir tous les champs", and "Annuler" and "Confirmer" buttons.

	Personnel	Tel	Mail	Service
<input type="checkbox"/>	Luke Skywalker	06 31 12 34 56	lskywalker@dagoba.fr	mediation culturelle
<input type="checkbox"/>	Han Solo	06 31 78 90 12	hsolo@tatooine.ca	prêt

Rapport d'activité

Atelier De Professionnalisation

Cas d'utilisation 3 (Suppression d'un Personnel) :

The screenshot shows the 'Gestion du Personnel' application window. A table lists staff members with columns for 'Personnel', 'Tel', 'Mail', and 'Service'. The first row shows 'Luke Skywalker' with a checked checkbox. A modal dialog box is open in the center, asking for confirmation to delete the selected staff member. The dialog contains a question mark icon and the text: 'Voulez vous vraiment supprimer ce personnel ? Toute suppression est irreversible.' Below the text are 'Annuler' and 'OK' buttons. At the bottom of the application window, there are buttons for 'Ajouter', 'Modifier', 'Supprimer', 'Absences', 'Annuler', and 'Confirmer'.

	Personnel	Tel	Mail	Service
<input checked="" type="checkbox"/>	Luke Skywalker			
<input type="checkbox"/>	Han Solo			

Modifier un Personnel

Nom

Prenom

Téléphone

Mail

mediation culturelle

Aucun personnel sélectionné

Annuler Confirmer

Si aucun personnel n'est sélectionné lors d'une tentative de suppression, un message s'affiche :

The screenshot shows the 'Gestion du Personnel' application window. The table lists staff members with columns for 'Personnel', 'Tel', 'Mail', and 'Service'. The first row shows 'Luke Skywalker' with an unchecked checkbox. The second row shows 'Han Solo' with an unchecked checkbox. A modal dialog box is open in the center, displaying the message 'Aucun personnel sélectionné' in red text. The dialog also contains 'Annuler' and 'Confirmer' buttons. At the bottom of the application window, there are buttons for 'Ajouter', 'Modifier', 'Supprimer', 'Absences', 'Annuler', and 'Confirmer'.

	Personnel	Tel	Mail	Service
<input type="checkbox"/>	Luke Skywalker	06 31 12 34 56	lskywalker@dagoba.fr	mediation culturelle
<input type="checkbox"/>	Han Solo	06 31 78 90 12	hsolo@tatooine.ca	prêt

Modifier un Personnel

Nom

Prenom

Téléphone

Mail

mediation culturelle

Aucun personnel sélectionné

Annuler Confirmer

Rapport d'activité

Atelier De Professionnalisation

Cas d'utilisation 4 (Modification d'un personnel)

The screenshot shows a web application titled "Gestion du Personnel". It features a table with personnel data and a form to modify a selected person.

	Personnel	Tel	Mail	Service
<input checked="" type="checkbox"/>	Luke Skywalker	06 31 12 34 56	lskywalker@dagoba.fr	mediation culturelle
<input type="checkbox"/>	Han Solo	06 31 78 90 12	hsolo@tatooine.ca	prêt

Buttons: Ajouter, Modifier, Supprimer, Absences

Modifier un Personnel

Nom: Skywalker
Prenom: Luke
Téléphone: 0631123456
Mail: lskywalker@dagoba.fr
Service: mediation culturelle (dropdown)
Buttons: Annuler, Confirmer

Cas d'utilisation 5,6,7,8 (Gestion des Absences) :

The screenshot shows the "Gestion des Absences" section of the application. It includes a table for absence entries and a form to add or modify an absence.

	Debut	Fin	Motif
<input type="checkbox"/>	date	date	maladie
<input type="checkbox"/>	date	date	maladie

Buttons: Ajouter, Modifier, Supprimer

Gestion des Absences

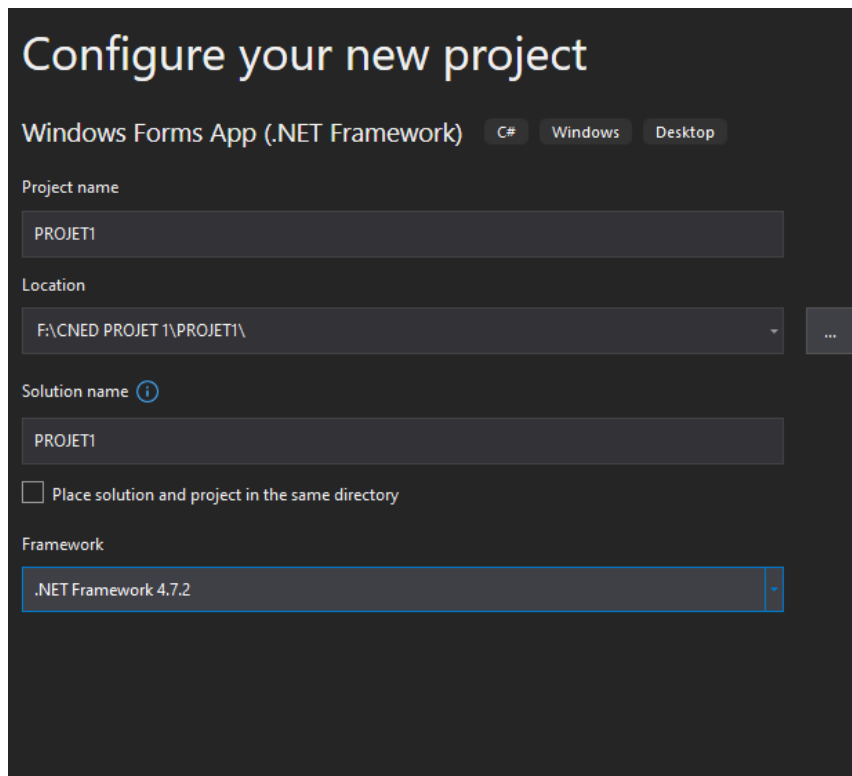
Group: [empty]
Début Absence: 12/02/2022 (dropdown)
Fin Absence: 12/03/2022 (dropdown)
Motif: maladie (dropdown)
Aucune Absence Selectionnée
Buttons: Annuler, Confirmer

2. Création de l'application dans L'IDE

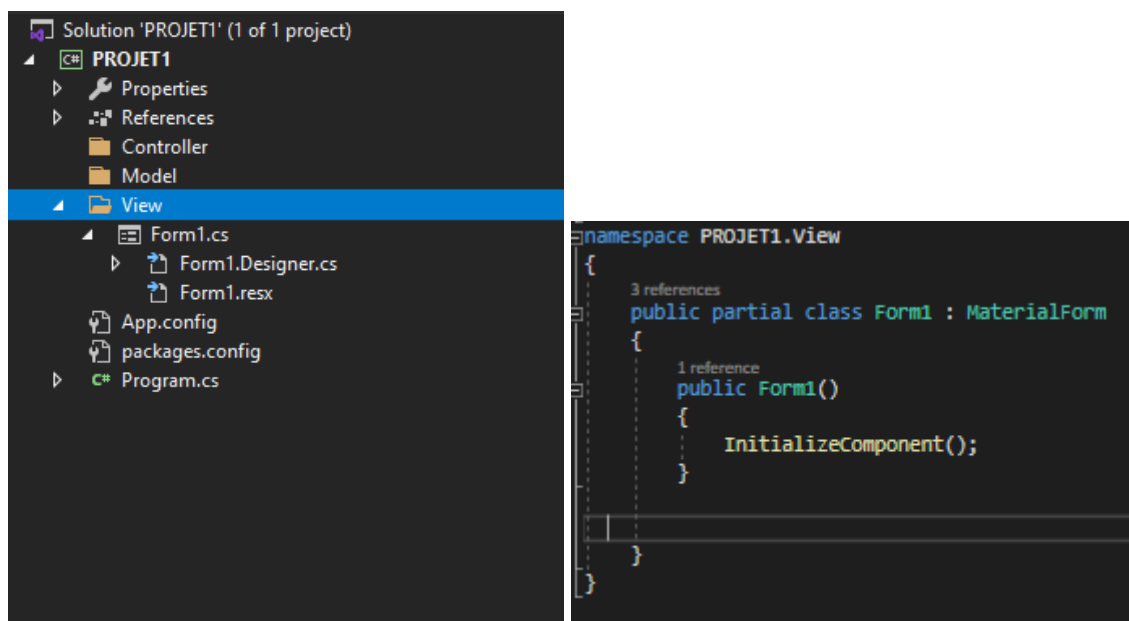
Rapport d'activité

Atelier De Professionnalisation

Nous allons créer un projet Windows Form sous Visual Studio, sous le framework .NET 4.7.2.



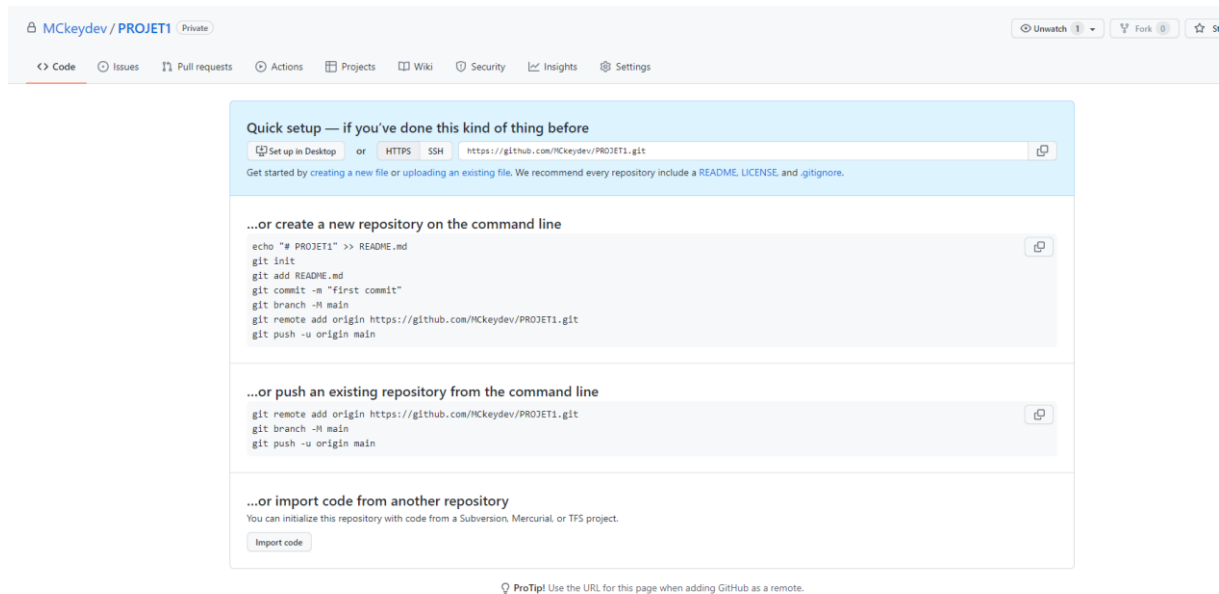
Afin de paramétrer le projet dans le respect de l'architecture MVC, commençons par créer trois packages Model/View/Controller. Déplaçons le formulaire déjà créé dans le package View, en utilisant les options de refactoring pour changer son namespace.



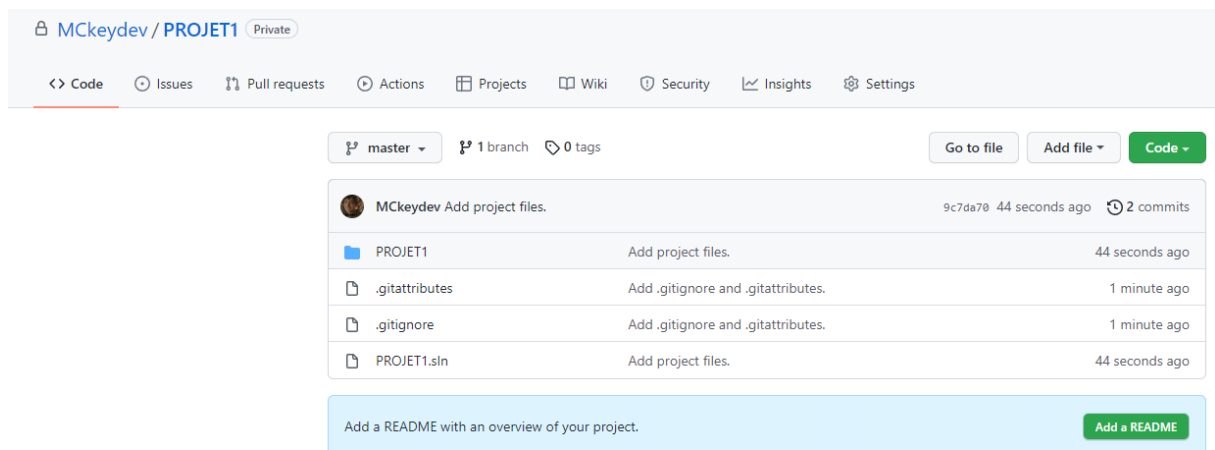
Maintenant que nous disposons d'une base de projet propre, créons un dépôt distant.

Rapport d'activité

Atelier De Professionnalisation



Sous Cissal studio, Github n'est pas disponible nativement. Il faut donc installer l'extension. Cependant, il est déjà installé sur mon poste de travail, cette étape peut donc être passée.



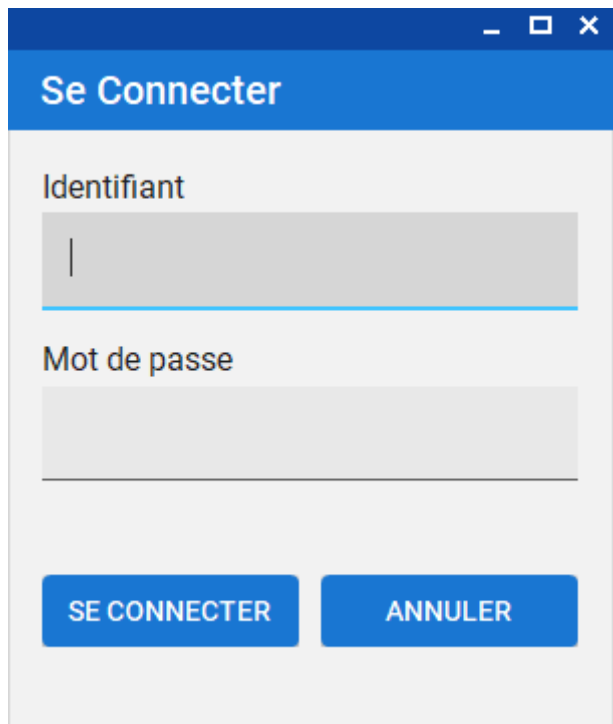
Le dépôt local est dorénavant lié au remote sur Github. Un premier commit d'initialisation est fait.

3. Développement de l'interface

Les étapes de préparations étant remplies, nous pouvons alors créer l'interface et coder les fonctionnalités propres aux vues.

Rapport d'activité

Atelier De Professionnalisation



The image shows a login window titled "Se Connecter". It features two input fields: "Identifiant" and "Mot de passe". Below the fields are two buttons: "SE CONNECTER" and "ANNULER". The window has a blue header bar with the title and standard window control buttons (minimize, maximize, close) in the top right corner.

Se Connecter

Identifiant

Mot de passe

SE CONNECTER ANNULER

Rapport d'activité

Atelier De Professionnalisation

Dessin de la vue d'authentification. TODO : Ajouter les fonctionnalit... [Browse files](#)

...és visuelles.

master

MCkeydev committed 1 minute ago 1 parent 9c7da70 commit 92903de83963ba3af86f4344c907297b255c0d13

Showing 10 changed files with 396 additions and 14 deletions. [Split](#) [Unified](#)

```
18 PROJ1/Controller/Controlele.cs
... @@ -0,0 +1,18 @@
1 + using System;
2 + using System.Collections.Generic;
3 + using System.Linq;
4 + using System.Text;
5 + using System.Threading.Tasks;
6 + using PROJ1.View;
7 + namespace PROJ1.Controller
8 + {
9 +     public class Controlele
10 +     {
11 +         private readonly FrmAuth frmAuth;
12 +         public Controlele()
13 +         {
14 +             this.frmAuth = new FrmAuth(this);
15 +             this.frmAuth.ShowDialog();
16 +         }
17 +     }
18 + }
```

```
26 PROJ1/PROJET1.csproj
... @@ -33,11 +33,12 @@
33 <WarningLevel>4</WarningLevel>
34 </PropertyGroup>
35 <ItemGroup>
36 - <Reference Include="MaterialSkin, Version=1.0.0.0, Culture=neutral, processorArchitecture=MSIL">
37 - <HintPath>..\packages\MaterialSkin.0.2.1\lib\MaterialSkin.dll</HintPath>
36 + <Reference Include="MaterialSkin, Version=2.3.1.0, Culture=neutral, processorArchitecture=MSIL">
37 + <HintPath>..\packages\MaterialSkin.2.3.1\lib\net461\MaterialSkin.dll</HintPath>
38 </Reference>
39 <Reference Include="System" />
```

Gestion du personnel

AJOUT

PERSONNEL

Nom

Prenom

Téléphone

Mail

ANNULER

CONFIRMER

AJOUTER

MODIFIER

SUPPRIMER

GESTIONNAIRE ABSENCES


Rapport d'activité

Atelier De Professionnalisation

Dessin de l'interface principale (vue Main), et des principales fonct...

Browse files

...ionnalités visuelles.

 MCkeydev committed 19 seconds ago 1 parent 92903de commit b3e1dfed9c23f11b7e1edf127cf6a688fda7719c

Showing 7 changed files with 429 additions and 19 deletions.

Split Unified

1 PROJ1/PROJET1.csproj

@@ -39,6 +39,7 @@

39

39

<Reference Include="System" />

40

40

<Reference Include="System.Core" />

41

41

<Reference Include="System.Design" />

42

+

<Reference Include="System.Messaging" />

42

43

<Reference Include="System.Xml.Linq" />

43

44

<Reference Include="System.Data.DataSetExtensions" />

44

45

<Reference Include="Microsoft.CSharp" />

3 PROJ1/Program.cs

@@ -16,7 +16,8 @@ static void Main()

16

16

{

17

17

Application.EnableVisualStyles();

18

18

Application.SetCompatibleTextRenderingDefault(false);

19

-

new Controle();

19

+

Application.Run(new Main());

20

+

//new Controle();

20

21

}

21

22

}

22

23

}

Un commit and push sera effectué après chaque le développement de chaque vue.

Rapport d'activité

Atelier De Professionnalisation

Nous disposons d'une vue de connexion, et d'une vue de gestion du personnel. Il ne manque alors qu'une vue de gestion des absences :

The screenshot shows a web application window titled "Gestion du personnel". Inside, there's a section for "Absences de : COURREJOU Matthieu". To the left is a large grey rectangular area, likely a calendar or list. To the right are three buttons: "AJOUTER" (blue), "MODIFIER" (blue), and "SUPPRIMER" (pink). Below these is a form with two date pickers labeled "Date Début" and "Date fin", both showing "dimanche 20 mars 2022". Below the date pickers is a "Motif" dropdown menu. At the bottom right of the form are two buttons: "ANNULER" (pink) and "CONFIRMER" (blue).

Toutes les vues étant codées et visuellement fonctionnelles, effectuons un dernier commit and push, et passons à l'étape suivante : le développement de la partie modèle et de la connexion à la base de données.

Troisième étape

Cette nouvelle étape va consister à coder la partie modèle de l'application, la connexion à la base de données et la couche d'accès aux données.

Pour ce faire, il nous faut d'abord paramétrer notre IDE afin qu'elle puisse communiquer avec le SGBDR.

Ayant déjà installé les drivers et connecteurs nécessaires à la communication entre Visual Studio et MySQL, il ne me reste qu'à ajouter une nouvelle connexion dans mon IDE.

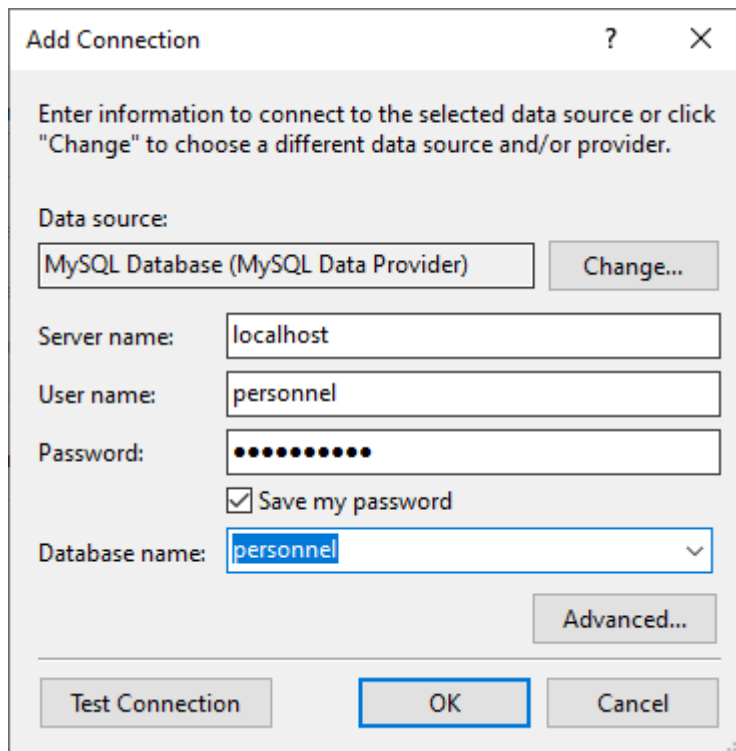
1. Connexion à la BDD

Dans la section « Projet », nous allons sélectionner « new Data Source ». On choisit « DataBase », puis « DataSet ».

Nous devons ensuite créer la connexion avec la BDD :

Rapport d'activité

Atelier De Professionnalisation



Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source: MySQL Database (MySQL Data Provider) Change...

Server name: localhost

User name: personnel

Password: Save my password

Database name: personnel

Advanced...

Test Connection OK Cancel

« Localhost » car le serveur est hébergé localement. Si nous utilisons un serveur distant, cela serait remplacé par l'adresse IP du serveur.

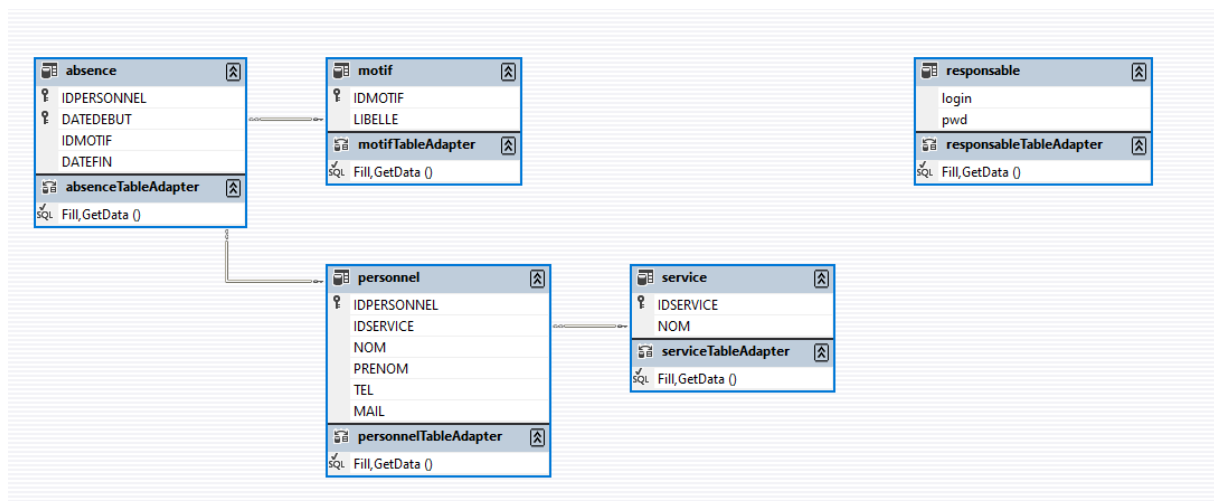
Nous utilisons l'utilisateur créé pour notre BDD afin de travailler avec les privilèges adéquats.

Récupérons la chaîne de connexion qui sera nécessaire dans notre Classe d'accès aux données.

server=localhost;user

id=personnel;password=motdepasse;persistsecurityinfo=True;database=personnel

La connexion est réussie, nous avons même accès à un MDC directement dans l'IDE :



Il va maintenant falloir créer les classes Métiers qui correspondent aux tables de la base de données.

Rapport d'activité

Atelier De Professionnalisation

2. Codage du modèle

Il va de soit de créer une classe pour chaque table de notre base de données. Cependant, les absences étant propres à un personnel, nous pouvons créer une liste d'absences dans la classe Personnel, afin de récupérer plus simplement toutes les absences d'un personnel.

Créons le classe Personnel :

```
/// <summary>
/// Classe représentant chaque personnel.
/// </summary>
1 reference | 0 changes | 0 authors, 0 changes
class Personnel
{
    private readonly int IdPersonnel;
    private readonly int IdService;
    private readonly string Nom;
    private readonly string Prenom;
    private readonly string Tel;
    private readonly string mail;
    private List<Absence> LesAbsences;

    0 references | 0 changes | 0 authors, 0 changes
    public Personnel(int idPersonnel, int idService, string nom, string prenom, string tel, string mail)
    {
        IdPersonnel = idPersonnel;
        IdService = idService;
        Nom = nom;
        Prenom = prenom;
        Tel = tel;
        this.mail = mail;
        this.LesAbsences = new List<Absence>();
    }

    0 references | 0 changes | 0 authors, 0 changes
    public int IdPersonnel1 { get => IdPersonnel; }
    0 references | 0 changes | 0 authors, 0 changes
    public int IdService1 { get => IdService; }
    0 references | 0 changes | 0 authors, 0 changes
    public string Nom1 { get => Nom; }
    0 references | 0 changes | 0 authors, 0 changes
    public string Prenom1 { get => Prenom; }
    0 references | 0 changes | 0 authors, 0 changes
    public string Tel1 { get => Tel; }
    0 references | 0 changes | 0 authors, 0 changes
    public string Mail { get => mail; }
}
```

La Classe Absence :

Rapport d'activité

Atelier De Professionnalisation

```
3 references | 0 changes | 0 authors, 0 changes
class Absence
{
    private readonly int IdPersonnel;
    private readonly string DateDebut;
    private readonly int IdMotif;
    private readonly string DateFin;

    0 references | 0 changes | 0 authors, 0 changes
    public Absence(int idPersonnel, string dateDebut, int idMotif, string dateFin)
    {
        IdPersonnel = idPersonnel;
        DateDebut = dateDebut;
        IdMotif = idMotif;
        DateFin = dateFin;
    }

    0 references | 0 changes | 0 authors, 0 changes
    public int IdPersonnel1 { get => IdPersonnel; }
    0 references | 0 changes | 0 authors, 0 changes
    public string DateDebut1 { get => DateDebut; }
    0 references | 0 changes | 0 authors, 0 changes
    public int IdMotif1 { get => IdMotif; }
    0 references | 0 changes | 0 authors, 0 changes
    public string DateFin1 { get => DateFin; }
}
```

La Classe Service :

```
1 reference | 0 changes | 0 authors, 0 changes
class Service
{
    private int IdService;
    private string Nom;

    0 references | 0 changes | 0 authors, 0 changes
    public Service(int idService, string nom)
    {
        IdService = idService;
        Nom = nom;
    }

    0 references | 0 changes | 0 authors, 0 changes
    public int IdService1 { get => IdService; }
    0 references | 0 changes | 0 authors, 0 changes
    public string Nom1 { get => Nom; }
}
```

Nous avons maintenant toutes les classes correspondant à la base de données. Passons donc au package de connexion.

Rapport d'activité

Atelier De Professionnalisation

La Classe Motif :

```
1 reference | 0 changes | 0 authors, 0 changes
class Motif
{
    private readonly int IdMotif;
    private readonly string Libelle;

    public Motif(int idMotif, string libelle)
    {
        IdMotif = idMotif;
        Libelle = libelle;
    }

    public int IdMotif1 => IdMotif;

    0 references | 0 changes | 0 authors, 0 changes
    public string Libelle1 => Libelle;
}
```

3. Codage du package « Connexion »

Un client n'a besoin que d'une seule connexion à la BDD, en avoir plusieurs seraient inutile et lourd en ressources.

La classe Connexion doit alors appliquer le design pattern d'un singleton. Elle doit nous permettre d'exécuter

Des requêtes SQL, que ce soit d'insertion/Delete/update, ou des requêtes Select.

Toutes les méthodes de requête doivent pouvoir recevoir un dictionnaire de paramètres, afin de préparer les requêtes dans un soucis de sécurité face aux injections SQL.

Rapport d'activité

Atelier De Professionnalisation

4 references | 0 changes | 0 authors, 0 changes

```
public class ConnexionBDD
{
    /// <summary>
    /// Instance unique de notre classe de Connexion.
    /// Initialisation à "null" pour la méthode "GetInstance"
    /// </summary>
    private static ConnexionBDD Instance = null;

    /// <summary>
    /// Object de connexion à la BDD.
    /// </summary>
    private MySqlConnection Connexion;

    /// <summary>
    /// Objet d'exécution de requêtes SQL.
    /// Nécessite une requête SQL et une instance de MySqlConnection.
    /// </summary>
    private MySqlCommand Commande;

    /// <summary>
    /// Curseur permettant de récupérer et lire les résultats de la requête.
    /// </summary>
    private MySqlDataReader Curseur;

    /// <summary>
    /// Constructeur privé qui sera appelé dans GetInstance,
    /// afin de vérifier qu'il n'existe qu'une seule instance de la classe.
    /// </summary>
    /// <param name="chaîneConnexion">Chaîne de connexion à la BDD</param>
    1 reference | 0 changes | 0 authors, 0 changes
    private ConnexionBDD(string chaîneConnexion)
    {
        try
        {
            this.Connexion = new MySqlConnection(chaîneConnexion);
            Connexion.Open();
        } catch (Exception e)
        {
            Debug.WriteLine(e.Message);
        }
    }

    0 references | 0 changes | 0 authors, 0 changes
    public static ConnexionBDD GetInstance(string chaîneConnexion)
    {
        if (Instance is null)
        {
            Instance = new ConnexionBDD(chaîneConnexion);
        }
        return Instance;
    }
}
```

Rapport d'activité

Atelier De Professionnalisation

```
/// <summary>
/// Méthode effectuant requête SQL.
/// Ne retourne rien car résultats disponibles par le Curseur.
/// La requête est préparée en cas de paramètres afin d'éviter une injection SQL.
/// </summary>
/// <param name="requeteSQL">Chaîne de requête SQL</param>
/// <param name="parameters">Dictionnaire de paramètres optionnel</param>
0 references | 0 changes | 0 authors, 0 changes
public void Select(string requeteSQL, Dictionary<string, object> parameters)
{
    try
    {
        Commande = new MySqlCommand(requeteSQL, Connexion);
        if(parameters is object)
        {
            foreach(KeyValuePair<string, object> parameter in parameters)
            {
                Commande.Parameters.Add(new MySqlParameter(parameter.Key, parameter.Value));
            }
        }
        Commande.Prepare();
        Curseur = Commande.ExecuteReader();
    }catch(Exception e)
    {
        Debug.WriteLine(e.Message);
    }
}

/// <summary>
/// Méthode qui exécute une requête non select.
/// Ici donc pas de résultats dans le curseur.
/// </summary>
/// <param name="requeteSQL">Chaîne de requête SQL</param>
/// <param name="parameters">Dictionnaire optionnel de paramètres</param>
0 references | 0 changes | 0 authors, 0 changes
public void Update(string requeteSQL, Dictionary<string, object> parameters)
{
    try{
        Commande = new MySqlCommand(requeteSQL, Connexion);
        if (parameters is object)
        {
            foreach (KeyValuePair<string, object> parameter in parameters)
            {
                Commande.Parameters.Add(new MySqlParameter(parameter.Key, parameter.Value));
            }
        }
        Commande.Prepare();
        Commande.ExecuteNonQuery();
    }
    catch(Exception e)
    {
        Debug.WriteLine(e.Message);
    }
}
```

Rapport d'activité

Atelier De Professionnalisation

```
/// <summary>
/// Méthode qui indique si nous sommes à la fin du curseur
/// </summary>
/// <returns>false si fin du curseur</returns>
0 references | 0 changes | 0 authors, 0 changes
public bool Read()
{
    if(Curseur is null)
    {
        return false;
    }
    try
    {
        return Curseur.Read();
    }
    catch
    {
        return false;
    }
}

/// <summary>
/// Méthode qui retourne le champ désiré dans le curseur;
/// </summary>
/// <param name="champ">Champ voulu</param>
/// <returns>null si le champ n'est pas présent dans le Curseur</returns>
0 references | 0 changes | 0 authors, 0 changes
public object Field(string champ)
{
    if(Curseur is null)
    {
        return null;
    }
    try
    {
        return Curseur[champ];
    }
    catch
    {
        return null;
    }
}

/// <summary>
/// Ferme le Curseur s'il n'est pas null
/// </summary>
0 references | 0 changes | 0 authors, 0 changes
public void Close()
{
    if(Curseur is object)
    {
        Curseur.Close();
    }
}
```

4. Codage du package « Data Access Layer »

La Classe AccesDonnees étant une classe outil, nous n'aurons pas besoin de l'instancier. Ses méthodes seront toutes statiques, et elle ne possèdera pas de constructeur.

Rapport d'activité

Atelier De Professionnalisation

Pour l'instant, nous allons juste la créer, et y stocker la chaine de connexion à la BDD.

```
namespace PROJET1.Dal
{
    /// <summary>
    /// Classe Outil nous permettant d'obtenir les informations
    /// nécessaires dans la BDD.
    /// </summary>
    /// 0 references | 0 changes | 0 authors, 0 changes
    public class AccesDonnees
    {
        private static string chaineConnexion = "server=localhost;user id=personnel;password=motdepasse;persistsecurityinfo=True;database=personnel";
    }
}
```

5. Génération de la Documentation Technique

Commençons par générer la Documentation au format XML.

```
F:\> C:\NET\PROJET1 > PROJET1 > PROJET1 > PROJET1 > bin > Debug > PROJET1.xml

1  [?xml version="1.0"?]
2  <doc>
3      <assembly>
4          <name>PROJET1</name>
5      </assembly>
6      <members>
7          <member name="T:PROJET1.Connexion.ConnexionBDD">
8              <summary>
9                  Classe de connexion à la base de donnée
10                 en Singleton.
11             </summary>
12         </member>
13         <member name="F:PROJET1.Connexion.ConnexionBDD.Instance">
14             <summary>
15                 Instance unique de notre classe de Connexion.
16                 Initialisation à "null" pour la méthode "GetInstance"
17             </summary>
18         </member>
19         <member name="F:PROJET1.Connexion.ConnexionBDD.Connexion">
20             <summary>
21                 Object de connexion à la BDD.
22             </summary>
23         </member>
24         <member name="F:PROJET1.Connexion.ConnexionBDD.Commande">
25             <summary>
26                 Objet d'exécution de requêtes SQL.
27                 Nécessite une requête SQL et une instance de MySqlConnection.
28             </summary>
29         </member>
30         <member name="F:PROJET1.Connexion.ConnexionBDD.Curseur">
31             <summary>
32                 Curseur permettant de récupérer et lire les résultats de la requête.
33             </summary>
34         </member>
35         <member name="M:PROJET1.Connexion.ConnexionBDD.#ctor(System.String)">
36             <summary>
37                 Constructeur privé qui sera appelé dans GetInstance,
38                 afin de vérifier qu'il n'existe qu'une seule instance de la classe.
39             </summary>
40             <param name="chaineConnexion">Chaîne de connexion à la BDD</param>
41         </member>
42         <member name="M:PROJET1.Connexion.ConnexionBDD.Select(System.String,System.Collections.Generic.Dictionary{System.String,System.Object})">
43             <summary>
44                 Méthode effectuant requête SQL.
45                 Ne retourne rien car résultats disponibles par le Curseur.
46                 La requête est préparée en cas de paramètres afin d'éviter une injection SQL.
47             </summary>
48             <param name="requeteSQL">Chaîne de requête SQL</param>
49             <param name="parameters">Dictionnaire de paramètres optionnel</param>
50         </member>
51     </members>
52 </doc>
```

Il est cependant préférable d'avoir une documentation au format HTML avec une mise en page, Afin d'améliorer l'accessibilité aux futurs développeurs du projet. Pour ce faire, nous allons utiliser le logiciel « SandCastle Help File Builder ».

Rapport d'activité

Atelier De Professionnalisation

La documentation est générée :

<div><div>A Sandcastle Documented Class Library</div><div>Namespaces</div><div><div>PROJECT1.Connexion</div><div>PROJECT1.Controller</div><div>PROJECT1.Dal</div><div>PROJECT1.Model</div><div>PROJECT1.View</div></div></div>	<h3>Namespaces</h3> <table><thead><tr><th>Namespace</th><th>Description</th></tr></thead><tbody><tr><td>PROJECT1.Connexion</td><td></td></tr><tr><td>PROJECT1.Controller</td><td></td></tr><tr><td>PROJECT1.Dal</td><td></td></tr><tr><td>PROJECT1.Model</td><td></td></tr><tr><td>PROJECT1.View</td><td></td></tr></tbody></table>	Namespace	Description	PROJECT1.Connexion		PROJECT1.Controller		PROJECT1.Dal		PROJECT1.Model		PROJECT1.View	
Namespace	Description												
PROJECT1.Connexion													
PROJECT1.Controller													
PROJECT1.Dal													
PROJECT1.Model													
PROJECT1.View													

Il ne reste donc qu'à effectuer un commit avec un message clair, et passer à l'étape suivante.

Ajout de la classe Connexion à la base de donnée, ainsi que des class...
...es métiers du Modèle.
Création de la classe AccesDonnes du DAL.

master

MCkeydev committed 21 minutes ago

1 parent 0e6cee3 commit 9542bf6a33e4774bdbb6b478e42f69ec051b33f7

Showing 19 changed files with 432 additions and 237 deletions.

4 .editorconfig

@@ -0,0 +1,4 @@
1 + [*.*cs]
2 +
3 + # S2933: Fields that are only assigned in the constructor should be "readonly"
4 + dotnet_diagnostic.S2933.severity = none

5 PROJECT1.sln

@@ -5,6 +5,11 @@ VisualStudioVersion = 16.0.31829.152
5 5 MinimumVisualStudioVersion = 10.0.40219.1
6 6 Project("{FAE04EC0-301F-11D3-BF48-00C04F79EFBC}") = "PROJECT1", "PROJECT1\PROJECT1.csproj", "{E331846C-5EA2-4E4E-AA45-81CA8DBA9097}"
7 7 EndProject
8 + Project("{2150E333-8FDC-42A3-9474-1A3956046DE8}") = "Solution Items", "Solution Items", "{5F9D4A09-0A3A-41E1-A7B5-E4481D8F1491}"
9 + ProjectSection(SolutionItems) = preProject
10 + .editorconfig = .editorconfig
11 + EndProjectSection
12 + EndProject
13 Global
14 GlobalSection(SolutionConfigurationPlatforms) = preSolution
15 Debug|Any CPU = Debug|Any CPU

Quatrième Étape

Recapitulons le travail déjà accompli :

- Nous avons paramétré l'environnement de Développement.
- Création et remplissage de la base de données correspondant au MCD fourni.
- Maquettage et développement de l'interface graphique du logiciel,
Et mise en place d'un dépôt distant.
- Développement des classes métiers du modèle, ainsi que la classe de connexion à la base de Données.

Rapport d'activité

Atelier De Professionnalisation

Toutes les conditions sont remplies pour commencer le développement des fonctionnalités correspondant aux Cas D'utilisation.

Nous allons respecter l'ordre des cas d'utilisation, le cas d'utilisation suivant ne sera traité qu'après test, et validation du précédent.

1. Cas D'utilisation 1 (l'authentification)

La démarche afin de répondre à ce cas d'utilisation est la suivante :

- Nous allons créer dans « AccesDonnes », une méthode « Authentification » qui reçoit en paramètre l'identifiant et le mot de passe récupérés depuis le formulaire « Auth ».
Celle-ci va appeler la méthode Select() de ConnexionBDD, avec la requête et un dictionnaire de paramètres.
Nous allons ensuite appeler la méthode Read(), qui retourne vrai si le Curseur n'est pas vide.
Donc, si le curseur n'est pas null, et que donc un utilisateur a été trouvé, les identifiants sont corrects.

```
/// <summary>
/// Effectue une requête SELECT sur la table 'responsable'.
/// </summary>
/// <param name="login">identifiant à tester</param>
/// <param name="pwd">mot de passe à tester</param>
/// <returns>true si utilisateur existe dans la BDD</returns>
1 reference | 0 changes | 0 authors, 0 changes
public static bool Authentification(string login, string pwd)
{
    string request = "SELECT * FROM responsable WHERE binary login=@login AND pwd=SHA2(@pwd, 256) ";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@login", login);
    parameters.Add("@pwd", pwd);
    ConnexionBDD connexion = ConnexionBDD.GetInstance(chaineConnexion);
    connexion.Select(request, parameters);

    if (connexion.Read())
    {
        connexion.Close();
        return true;
    }
    else
    {
        connexion.Close();
        return false;
    }
}
```

/*le mot « binary » dans la requête permet de s'assurer que l'authentification est sensible à la casse*/

Rapport d'activité

Atelier De Professionnalisation

Petites précisions : Nous pourrions vider les champs de login à chaque echec d'authentification. Cependant, j'ai choisi de ne pas le faire, afin de ne pas forcer l'utilisateur à renseigner une nouvelle fois toutes ses informations en cas de simple coquille typographique.

Effectuons un commit sur le dépôt distant, après s'être bien assuré que la fonctionnalité est opérationnelle.



2. Cas D'utilisation 2 : Ajout d'un personnel

Tout d'abord, faisons en sorte que le logiciel se lance sans authentification, le temps du développement.

```
6 references | MCkeydev, 8 minutes ago | 1 author, 3 changes
public class Controle
{
    private readonly FrmAuth FrmAuth;
    private readonly Main FrmMain;
    1 reference | MCkeydev, 3 hours ago | 1 author, 2 changes
    public Controle()
    {
        this.FrmMain = new Main(this);
        this.FrmMain.ShowDialog();
        //this.FrmAuth = new FrmAuth(this);
        //this.FrmAuth.ShowDialog();
    }
}
```

Au lancement du programme, la classe Contrôle est instanciée. Celle-ci est censée lancer le formulaire d'authentification afin de vérifier l'identité de l'utilisateur. Ici, le formulaire du programme sera directement lancé.

3. Cas D'utilisation 3 : Suppression d'un Personnel

Le cas d'utilisation ici traité, ne propose que de supprimer un seul personnel à la fois. Cependant, le composant DataGridView, permet de sélectionner plusieurs lignes à la fois, il peut être intéressant pour l'utilisateur de pouvoir supprimer plusieurs personnels s'il le souhaite.

1) Méthode AccesDonnees

```
/// <summary>
/// Supprime une personnel de la base de Données
/// </summary>
/// <param name="idpersonnel"></param>

1 reference | 0 changes | 0 authors, 0 changes
public static void SupprPersonnel(int idpersonnel)
{
    string request = "DELETE FROM personnel WHERE idpersonnel = @idpersonnel";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@idpersonnel", idpersonnel);
    ConnexionBDD c = ConnexionBDD.GetInstance(chaineConnexion);
    c.Update(request, parameters);
}
```

Rapport d'activité

Atelier De Professionnalisation

2) Méthode Contrôleur

```
/// <summary>
/// Appelle la méthode SupprPersonnel D'AccesDonnees
/// </summary>
/// <param name="idpersonnel">ID du personnel à supprimer</param>

public void SupprPersonnel(int idpersonnel)
{
    AccesDonnees.SupprPersonnel(idpersonnel);
}
```

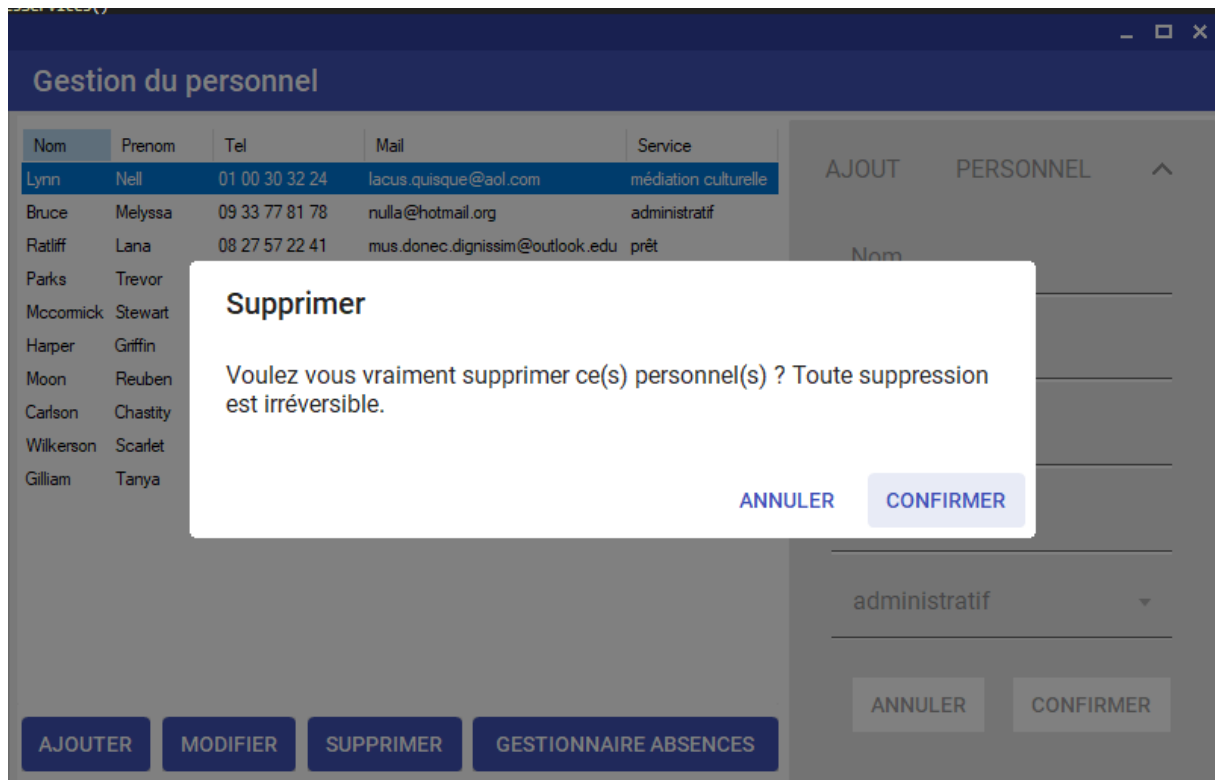
3) Méthode Evènementielle du bouton de Suppression

```
1 reference | 0 changes | 0 authors, 0 changes
private void btnSupprimer_Click(object sender, EventArgs e)
{
    if(dataPersonnel.SelectedRows.Count == 0)
    {
        MaterialSnackBar message = new MaterialSnackBar("Veuillez selectionner un personnel", "Ok", true);
        message.Show(this);
    }
    else
    {
        MaterialDialog materialDialog = new MaterialDialog(this, "Supprimer", "Voulez vous vraiment supprimer ce(s) personnel(s) ? Toute suppression est irréversible.", "Confirmer", true, "Annuler");
        DialogResult result = materialDialog.ShowDialog(this);
        if(((int)result) == 1)
        {
            foreach (DataGridViewRow row in dataPersonnel.SelectedRows)
            {
                int idpersonnel = int.Parse(row.Cells["idpersonnel"].Value.ToString());
                controle.SupprPersonnel(idpersonnel);
            }
            InitialesPersonnels();
        }
    }
}
```

On remarque qu'ici on boucle sur chaque ligne afin d'en récupérer les informations propres à chaque personnel. On appelle alors le contrôleur en lui transmettant l'information.

Rapport d'activité

Atelier De Professionnalisation



Une boîte de dialogue permet à l'utilisateur de confirmer son choix, afin d'éviter toute suppression accidentelle.

4) Cas d'utilisation 4 (Modification d'un personnel)

Ce cas est plus complexe que les précédents, dans la mesure où il utilise les mêmes contrôles que la fonction d'ajout.

Il faut donc créer une propriété booléenne qui permettra de différencier les deux cas d'utilisation :

```
private bool enModif = false;
```

On va d'abord commencer par coder la méthode événementielle du bouton « modifier » :

Rapport d'activité

Atelier De Professionnalisation

```
1 reference | 0 changes | 0 authors, 0 changes
private void btnModifier_Click(object sender, EventArgs e)
{
    if(dataPersonnel.SelectedRows.Count == 0)
    {
        MaterialSnackBar message = new MaterialSnackBar("Veuillez sélectionner un personnel", "OK", true);
        message.Show(this);
    }
    else if(dataPersonnel.SelectedRows.Count > 1) {
        MaterialSnackBar message = new MaterialSnackBar("Veuillez ne sélectionner qu'un personnel", "OK", true);
        message.Show(this);
    }
    else
    {
        enModif = true;
        grpAjout.Enabled = true;
        grpMain.Enabled = false;
        dataPersonnel.Enabled = false;
        Personnel lePersonnel = (Personnel)bdsPersonnel[bdsPersonnel.Position];
        txtNom.Text = lePersonnel.Nom;
        txtPrenom.Text = lePersonnel.Prenom;
        txtMail.Text = lePersonnel.Mail;
        txtTel.Text = lePersonnel.Tel;
        cbService.SelectedIndex = cbService.FindStringExact(lePersonnel.Service);
    }
}
```

Cette méthode s'assure qu'un seul employé est sélectionné.

Elle va ensuite indiquer que nous allons modifier un employé et non pas en ajouter.

Enfin elle transmet les informations du personnel, aux textbox de la zone de modification du personnel.

```
1 reference | MCheydev, 15 hours ago | 1 author, 3 changes
private void btnAjoutConfirmer_Click(object sender, EventArgs e)
{
    if (VerifChampsAjout())
    {
        int idpersonnel = 0;
        if (enModif)
        {
            idpersonnel = int.Parse(dataPersonnel.SelectedRows[0].Cells["idpersonnel"].Value.ToString());
        }

        Service leService = (Service)bdsService.List[bdsService.Position];
        Personnel newPersonnel = new Personnel(idpersonnel, leService.Nom1, leService.IdService1, txtNom.Text, txtPrenom.Text, txtTel.Text, txtMail.Text);

        if (enModif)
        {
            MaterialDialog materialDialog = new MaterialDialog(this, "Modifier", "Voulez vous vraiment modifier ce personnel ? ", "confirmer", true, "Annuler");
            DialogResult result = materialDialog.ShowDialog(this);
            if (((int)result) != 1) return;
            controle.ModifPersonnel(newPersonnel);
        }
        else
        {
            controle.AjoutPersonnel(newPersonnel);
        }

        InitilesPersonnels();
        ResetChampsAjout();

        grpMain.Enabled = true;
        dataPersonnel.Enabled = true;
        enModif = false;
    }
    else
    {
        MaterialSnackBar message = new MaterialSnackBar("Veuillez renseigner tous les champs", "OK", true);
        message.Show(this);
    }
}
```

Voici la méthode du bouton « confirmer », modifiée pour prendre en compte la modification d'un personnel.

Rapport d'activité

Atelier De Professionnalisation

```
/// <summary>
/// Modifie un personnel dans la base de données
/// </summary>
/// <param name="lePersonnel">Personnel avec les nouvelles informations</param>
1 reference | 0 changes | 0 authors, 0 changes
public static void ModifierPersonnel(Personnel lePersonnel)
{
    string request = "UPDATE personnel SET idservice = @idservice, nom = @nom, prenom = @prenom, tel = @tel, mail = @mail WHERE idpersonnel = @idpersonnel";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@idpersonnel", lePersonnel.IdPersonnel);
    parameters.Add("@idservice", lePersonnel.IdService);
    parameters.Add("@nom", lePersonnel.Nom);
    parameters.Add("@prenom", lePersonnel.Prenom);
    parameters.Add("@tel", lePersonnel.Tel);
    parameters.Add("@mail", lePersonnel.Mail);
    ConnexionBDD c = ConnexionBDD.GetInstance(chaineConnexion);
    c.Update(request, parameters);
}
```

Voici la méthode dans la DAL, qui permet de modifier un personnel.

5) Cas d'utilisation (affichage des absences)

La Liste des Absences sera différente pour chaque Personnel. Il n'est donc nécessaire de la remplir qu'en cas de clic sur le bouton « Gestion des Absences ».

Cependant la ComboBox des motifs d'absence ne change jamais. Nous allons donc commencer par remplir celle-ci.

```
/// <summary>
/// Récupère la liste des motifs dans la base de données
/// </summary>
/// <returns>liste des motifs</returns>
1 reference | MCKeydev, Less than 5 minutes ago | 1 author, 1 change
public static List<Motif> GetMotifs()
{
    string request = "SELECT * FROM motif";
    ConnexionBDD c = ConnexionBDD.GetInstance(chaineConnexion);
    c.Select(request, null);
    List<Motif> lesMotifs = new List<Motif>();
    while (c.Read())
    {
        Motif motif = new Motif((int)c.Field("idmotif"), (string)c.Field("libelle"));
        lesMotifs.Add(motif);
    }
    c.Close();
    return lesMotifs;
}
```

*Méthode de la DAL,
qui récupère la liste
des motifs.*

Rapport d'activité

Atelier De Professionnalisation

```
/// <summary>
/// Appelle la méthode GetMotifs d'AccesDonnees.
/// </summary>
/// <returns>List des Motifs</returns>
1 reference | MCkeydev, Less than 5 minutes ago | 1 author, 1 change
public List<Motif> GetMotifs()
{
    return AccesDonnees.GetMotifs();
}
```

Seul le contrôleur peut appeler les méthodes de la DAL, en respect du MVC.

```
/// <summary>
/// Initialise la ComboBox des Motifs.
/// </summary>
1 reference | MCkeydev, Less than 5 minutes ago | 1 author, 1 change
private void InitMotif()
{
    bdsMotif.DataSource = controle.GetMotifs();
    cboMotif.DataSource = bdsMotif;
}
```

Remplissage de la ComboBox. La méthode est appelée au lancement du programme.

Ensuite nous initialisons la Liste d'absence, sur le même mode opératoire que pour la liste des Personnel, à la différence près que l'on utilise l'Id du personnel comme paramètre dans la requête SQL :

```
/// <summary>
/// Recupère la liste des absences d'un personnel dans la BDD.
/// </summary>
/// <param name="lePersonnel">Personnel identifiant les absences</param>
/// <returns>Liste des absences.</returns>
1 reference | MCkeydev, 6 minutes ago | 1 author, 1 change
public static List<Absence> GetAbsences(Personnel lePersonnel)
{
    string request = "SELECT a.*, m.libelle as motif FROM absence a JOIN motif m ON a.idmotif = m.idmotif WHERE idpersonnel = @idpersonnel ORDER BY datedebut DESC";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@idpersonnel", lePersonnel.IdPersonnel);
    ConnexionBDD c = ConnexionBDD.GetInstance(chaineConnexion);
    c.Select(request, parameters);
    List<Absence> lesAbsences = new List<Absence>();
    while (c.Read())
    {
        Absence absence = new Absence((int)c.Field("idpersonnel"), (DateTime)c.Field("datedebut"), (int)c.Field("idmotif"), (DateTime)c.Field("datefin"), (string)c.Field("motif"));
        lesAbsences.Add(absence);
    }
    c.Close();
    return lesAbsences;
}
```

Voici le résultat obtenu :

Rapport d'activité

Atelier De Professionnalisation

Absences de : HOLMES Sherlock

Date Début	Date Fin	Motif
samedi 18 février 2023	dimanche 27 février 2022	vacances
mardi 31 janvier 2023	mercredi 17 novembre 2021	maladie
vendredi 17 juin 2022	mardi 20 avril 2021	motif familial
mercredi 6 avril 2022	samedi 21 janvier 2023	maladie
dimanche 23 janvier 2022	dimanche 17 avril 2022	maladie
lundi 26 avril 2021	jeudi 22 juillet 2021	motif familial

AJOUTER

MODIFIER

SUPPRIMER

Date Début

mardi 22 mars 2022 ▾

Date fin

mardi 22 mars 2022 ▾

Motif

vacances ▾

ANNULER

CONFIRMER

6) Cas d'utilisation 6 (ajout d'absence)

La démarche va être similaire à l'ajout de personnel, à la différence près que deux absences ne peuvent être ajoutées à la même date. Il faudra alors « throw » une erreur, et l'afficher à l'utilisateur.

Voici ce que l'on obtient :

Rapport d'activité

Atelier De Professionnalisation

Gestion du personnel

Absences de : BEZOS Jeff

Date Début	Date Fin	Motif
mardi 22 mars 2022	mercredi 23 mars 2022	vacances

AJOUTER MODIFIER SUPPRIMER

Date Début Date fin

mardi 22 mars 2022 mercredi 23 mars 2022

Motif

vacances

ANNULER CONFIRMER

Impossible d'ajouter deux absences à la même date. OK

7) Cas d'utilisation 4 (suppression d'une absence)

Là pareil, la méthode sera très similaire à la suppression d'un personnel.

La subtilité va être que dans la classe Absence, j'ai fait le choix de retourner la date en chaîne formatée. Donc au moment d'insérer la date dans la requête INSERT, il faut juste la reconvertir en DateTime :

```
/// <summary>
/// Ajoute une absence.
/// </summary>
/// <param name="absence">Object correspondant à l'absence à insérer</param>
1 reference | MKeydev, 45 minutes ago | 1 author, 1 change
public static void AjoutAbsence(Absence absence)
{
    string request = "INSERT INTO absence(idpersonnel, datedebut, idmotif, datefin) VALUES(@idpersonnel, @datedebut, @idmotif, @datefin)";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@idpersonnel", absence.IdPersonnel);
    parameters.Add("@datedebut", DateTime.Parse(absence.DateDebut));
    parameters.Add("@idmotif", absence.IdMotif);
    parameters.Add("@datefin", DateTime.Parse(absence.DateFin));
    ConnexionBDD c = ConnexionBDD.GetInstance(chaineConnexion);
    c.Update(request, parameters);
}
```

8) Cas d'utilisation 8 (