



KuriMediation

(Gestionnaire de cas de médiations)

Chef de projet : M. Dimitrios Lymberis

Expert 1 : M. Nicolas Borboën

Expert 2 : M. Bernard Oberson

Sujet : Développement WEB

Framework : Laravel Breeze – Livewire

Durée : 88 heures

Table des matières

1	Introduction	3
1.1	Titre	3
1.2	Introduction	3
1.3	Matériels à disposition	3
1.4	Prérequis	3
1.5	Livrables	3
2	Analyse / Conception.....	4
2.1	Objectifs.....	4
2.2	Planification initiale	5
2.3	Méthodologie de travail.....	6
2.4	Environnement	6
2.4.1	Laravel	6
2.4.2	MVC.....	6
2.4.3	Eloquent (ORM).....	7
2.4.4	Breeze	7
2.4.5	Tailwind CSS	7
2.4.6	Livewire	7
2.4.7	UwAmp	7
2.5	Conception	8
2.5.1	Schéma fonctionnel	8
2.5.2	Modélisation de la base de données (Méthode MERISE)	9
2.5.3	Justification des types de valeurs dans la modélisation.....	10
2.5.4	Maquettes	11
2.6	Stratégie de test.....	19
2.6.1	Model Factories	19
2.6.2	Seeders	19
2.6.3	Tests manuels	19
2.7	Risques techniques	19
3	Réalisation.....	20
3.1	Mise en place de UwAmp	20
3.2	Installation et mise en place de Laravel 11	20
3.2.1	Connexion à la base de données grâce au fichier « .env »	20
3.3	Mise en place de Breeze	21
3.5	Migrations	22
3.5.1	Table « users »	22
3.5.2	Table « meetings »	23
3.5.3	Table « aftercares »	23
3.5.4	Table « documents »	24
3.5.5	Table « types »	24
3.6	Mise en place des contrôleurs	25
3.7	Mise en place des routes	25
3.7.1	web.php	25
3.7.2	auth.php.....	28
3.7.3	console.php	28
3.8	Authentification	29
3.9	Gestion d'entretiens	30

3.9.1	Enregistrement d'un entretien.....	30
3.9.2	Document PDF	33
3.9.3	Liste d'entretiens.....	33
3.9.4	Statistiques	34
3.10	Gestion des suivis.....	37
3.11	Validation des données et Gestion des erreurs	38
3.12	Insertion de documents PDF liés à un entretien	39
3.12.1	Configuration du disque « pdf »	40
3.12.2	Méthode upload(Request \$request, \$meetingId).....	42
3.12.3	Méthode show(\$meetingId, \$fileName)	42
3.12.4	Méthode destroy(\$meetingId, \$fileName)	44
3.13	Affichage sous format graphique des statistiques	44
3.13.1	Installation et mise en place de Laravel Charts	44
3.13.2	Récupération des valeurs pour les entretiens et ses suivis par mois... ..	46
3.13.3	Récupération des valeurs pour les entretiens et ses suivis par catégorie ..	47
3.13.4	Création des graphiques.....	48
3.13.5	Affichage des graphiques sur la page de graphiques	49
3.14	Export de la page statistique	50
3.14.1	Redimensionnement de la page	50
3.14.2	En-tête et pied de page personnalisé	53
3.15	Exactitudes des valeurs statistiques	56
3.16	Description des tests effectués	56
3.16.1	Test de fonctionnement du site.....	56
3.16.2	Test de migrations	56
3.16.3	Test manuels	56
3.17	Erreurs restantes	56
3.17.1	Routes /home et /home/{year?}	56
3.17.2	Barre de navigation – Lien vers la page des graphiques	57
3.18	Liste des documents fournis	59
4	Conclusions	60
4.1	Bilan de la planification	60
4.2	Bilan des fonctionnalités	62
4.3	Bilan personnel	62
5	Glossaire	63
6	Annexes.....	64
7	Sources	65

1 Introduction

1.1 Titre

Réalisation d'une application web qui a pour but de gérer les cas de médiations au sein d'un établissement scolaire.

1.2 Introduction

Ce projet consiste à mettre en place une application Web permettant de gérer les cas de médiation. Cette application est destinée aux médiateurs souhaitant effectuer un bilan en fin d'année du temps consacré aux médiations en répertoriant non seulement le nombre d'heures mais également le nombre de cas de médiation selon leurs types.

Le but de cette application est de simplifier et optimiser le processus de suivi des entretiens au sein de l'établissement scolaire pour les membres de l'équipe santé (conseillers d'orientation, médiateurs, psychologues, infirmières, etc..).

1.3 Matériels à disposition

- Serveur hébergé par M. Lymberis
- Visual Code avec des extensions facilitant le développement.
- PHP 8.3
- Composer
- UwAmp pour le serveur de base de données de développement en local
- Connexion wifi.

1.4 Prérequis

- Connaître les notions de la Programmation Orienté Objet
- Savoir faire des requêtes SQL
- Savoir coder en langage PHP.
- Avoir suivi les cours des modules ICH-ICT et réalisé différents projets durant la formation d'informaticien CFC à l'ETML (Modules ICT 101, 104, 105, 133 et 151).
- Savoir générer des graphiques avec Highcharts ou D3.js, chart.js.

1.5 Livrables

- Un rapport de projet
- Un journal de travail
- Le code source avec la base de données.

2 Analyse / Conception

Dans ce projet, plusieurs fonctionnalités de bases apprises durant la formation seront à mettre en place. Des opérations CRUD devront être mises en place pour gérer les entretiens ainsi que les suivis, créer des utilisateurs qui auront le droit ou non d'accéder à des pages administratives, générer des graphiques exportables au format PDF basés sur les données récupérées d'entretiens et de suivis.

Après avoir validé l'application, elle sera déployée sur le serveur WEB mis en place par Monsieur Lymberis.

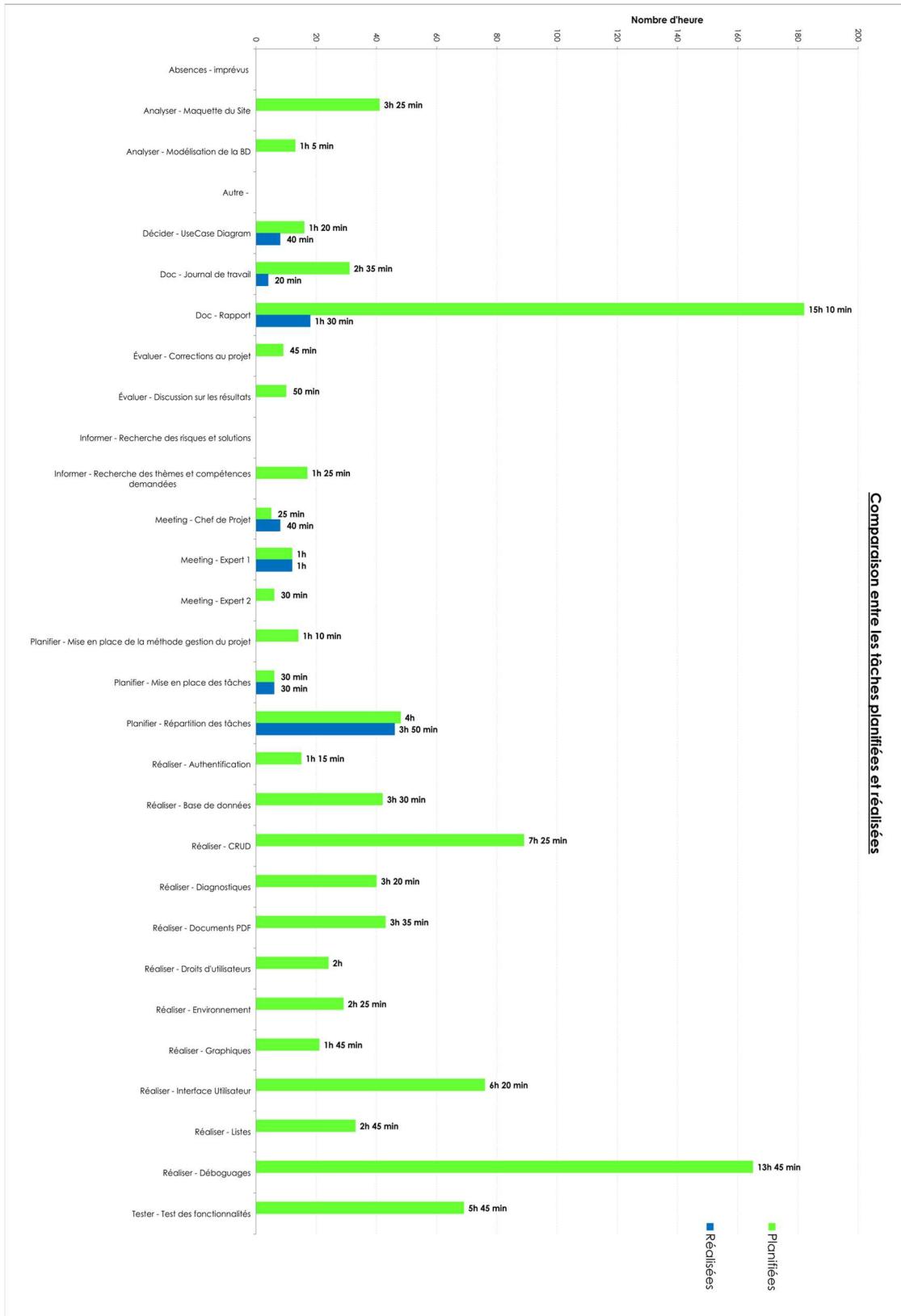
2.1 Objectifs

Ce sous-chapitre est consacré aux objectifs demandés. C'est-à-dire, les fonctionnalités vues avec le maître de stage :

- Opérations CRUD pour les entretiens, les suivis et les années.
- Liste d'entretiens et de ses suivis.
- Filtrage d'entretiens par date, ordre alphabétique ou ordre anti-alphabétique.
- Graphiques représentant la somme de cas de médiations par types, du temps consacré aux entretiens par année, de médiations au total. (Camembert, Histogramme, Barres).
- Téléchargement/Téléversement de fichiers en format PDF.
- Exactitude des valeurs entrées dans les formulaires.
- Ajout/Modification/Suppression de documents liés à un entretien.
- Affichage d'erreurs.

2.2 Planification initiale

Ce chapitre montre la planification du projet. Celui-ci peut être découpé en tâches qui seront planifiées. Il s'agit de la première planification du projet, celle-ci devra être revue après l'analyse. Cette planification sera présentée sous la forme d'un diagramme.



2.3 Méthodologie de travail

Afin de réaliser ce projet de TPI, la méthode qui sera utilisée est la méthode des six pas qui est une méthode très convenable dans le cadre où une limite de temps est imposée.

2.4 Environnement

2.4.1 Laravel

Laravel¹ est un framework PHP open-source, destiné au développement d'applications web en suivant l'architecture MVC (Model – View - Controller) créée par Taylor Otwell. Ce framework fait partie d'un des frameworks PHP les plus populaires grâce à sa simplicité d'utilisation, ses diverses fonctionnalités intégrées et une syntaxe très simpliste. De plus, il évolue à constamment et possède une communauté très active, ce qui fait que la documentation et l'aide en cas de soucis est fortement disponible.

Dans le cas de ce projet, Laravel 11 sera utilisé.

2.4.2 MVC

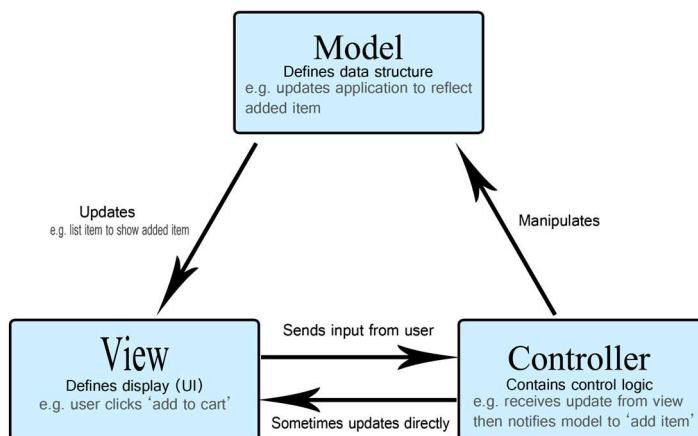
Le modèle MVC² (Model – View – Controller) représente l'architecture centrale utilisé surtout en Laravel. Il vise à simplifier le travail en divisant la charge du travail d'une application web.

Le modèle (Model) représente la partie logique, c'est-à-dire, tout ce qui concerne, la gestion, la récupération, la manipulation ainsi que la sauvegarde de données.

La vue (View) représente la partie visuelle, c'est-à-dire, l'affichage de la page web ainsi que les données. Elle concerne principalement l'interface utilisateur.

Le contrôleur (Controller) représente l'intermédiaire entre le modèle et la vue. Par exemple pour la création d'un utilisateur. Il reçoit les entrées à travers des formulaires puis les traite en interagissant avec le modèle puis, renvoie les résultats à la vue pour qu'ils puissent être affichés.

Cette division des tâches facilite énormément le développement, l'évolution et la maintenance des applications.



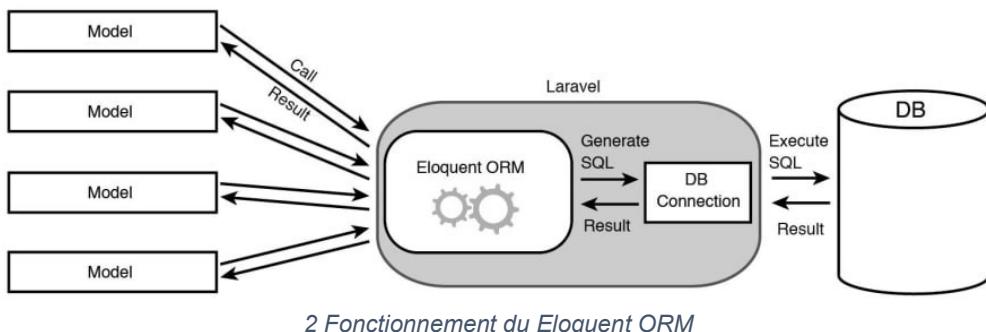
1 Représentation du fonctionnement du MVC

¹ Documentation officielle de Laravel : <https://laravel.com/docs/11.x>

² Documentation du MVC : <https://openclassrooms.com/fr/courses/4670706>

2.4.3 Eloquent (ORM)

Eloquent³ est un ORM (Object-Relational Mapping) intégré à Laravel. C'est un programme qui se place entre une application web et une base de données afin de permettre aux développeurs de travailler avec les données sous formes d'objet à la place d'écrire des requêtes SQL.



2.4.4 Breeze

Breeze⁴ ou Laravel Breeze est un composant supplémentaire de Laravel qui offre une implémentation très simpliste et minimaliste de fonctionnalités d'authentification. Il a pour but de servir de point de départ pour les applications web qui ont besoins de fonctionnalités de bases telles que l'enregistrement, la connexion, la modification de données utilisateurs, la suppression de compte, la vérification d'email etc...

Breeze inclut déjà Tailwind CSS qui est un framework CSS rendant la stylisation de vues plus facile et rapide.

2.4.5 Tailwind CSS

Tailwind CSS⁵ est comme mentionné ci-dessus, un framework CSS qui permet de rendre plus facile la stylisation de vue en évitant de passer par un fichier css.

2.4.6 Livewire

Livewire⁶ est un framework full-stack pour Laravel. Cela facilite la création d'interfaces utilisateur dynamiques directement dans Laravel sans avoir nécessairement besoin de connaissances en JavaScript. Par exemple, à la place de rediriger à chaque appui de boutons vers une autre page contenant un formulaire, avec Livewire, c'est possible d'afficher ou masquer le formulaire sur la page.

2.4.7 UwAmp

UwAmp⁷ est un ensemble de logiciels (Uniform, Windows, Apache, MySQL, PHP) dont le but est de créer un environnement de développement web local sur Windows. Il inclut des outils comme Apache, MySQL et PHP.

³ Documentation sur Eloquent : <https://laravel.com/docs/11.x/eloquent>

⁴ Documentation sur Breeze : <https://laravel.com/docs/11.x/starter-kits#laravel-breeze>

⁵ Documentation sur Tailwind CSS : <https://tailwindui.com>

⁶ Documentation sur Livewire : <https://laravel-livewire.com>

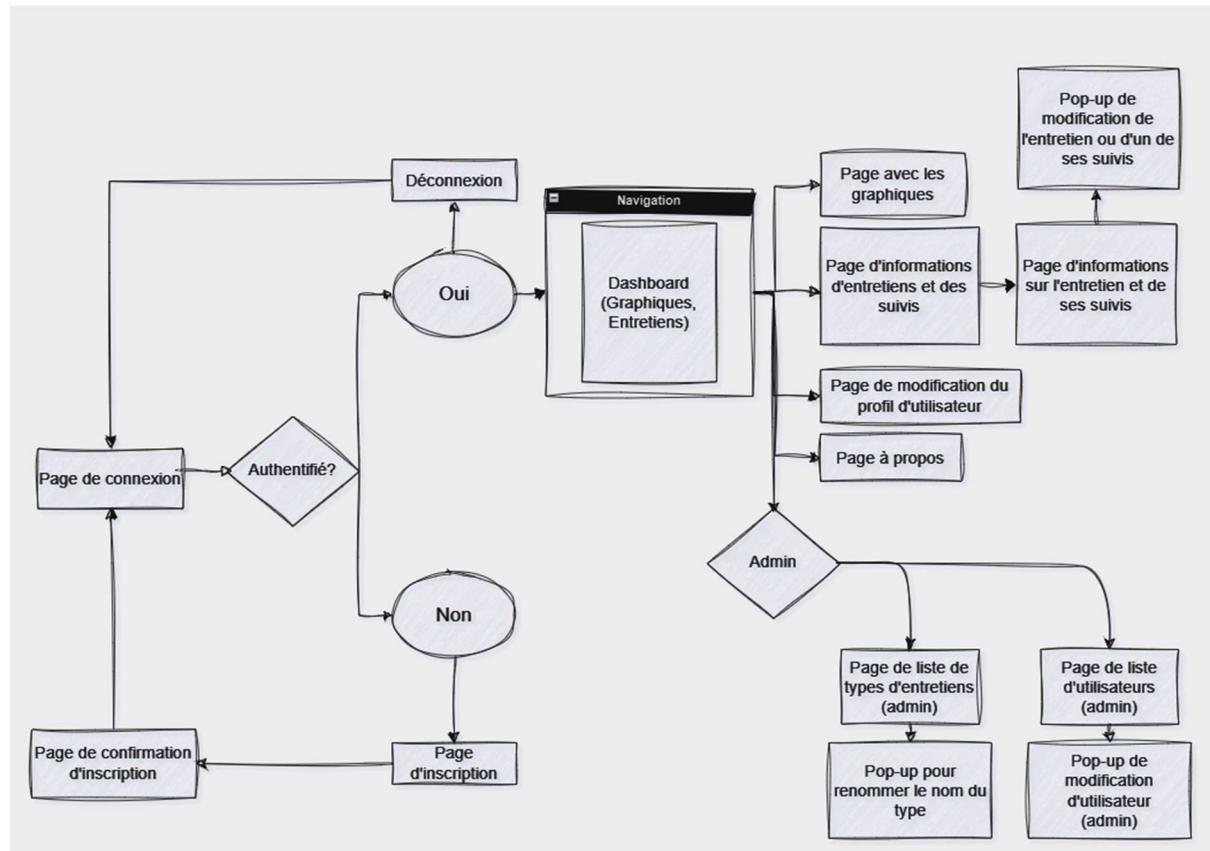
⁷ Documentation sur UwAmp : <https://www.uwamp.com/fr/>

2.5 Conception

Ce chapitre expliquera comment le site fonctionnera à l'aide d'un schéma de principe, des modèles MCD et MLD et des maquettes du site web.

2.5.1 Schéma fonctionnel

Le schéma ci-dessous représente le fonctionnement du site web.



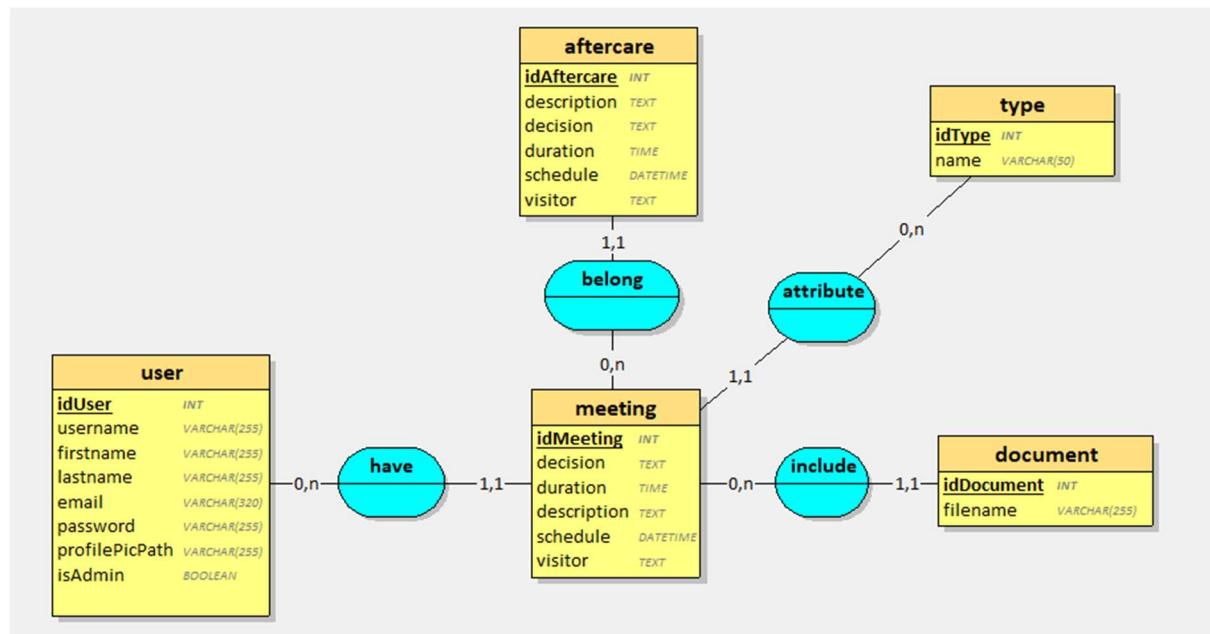
3 Schéma fonctionnel du site

2.5.2 Modélisation de la base de données (Méthode MERISE)

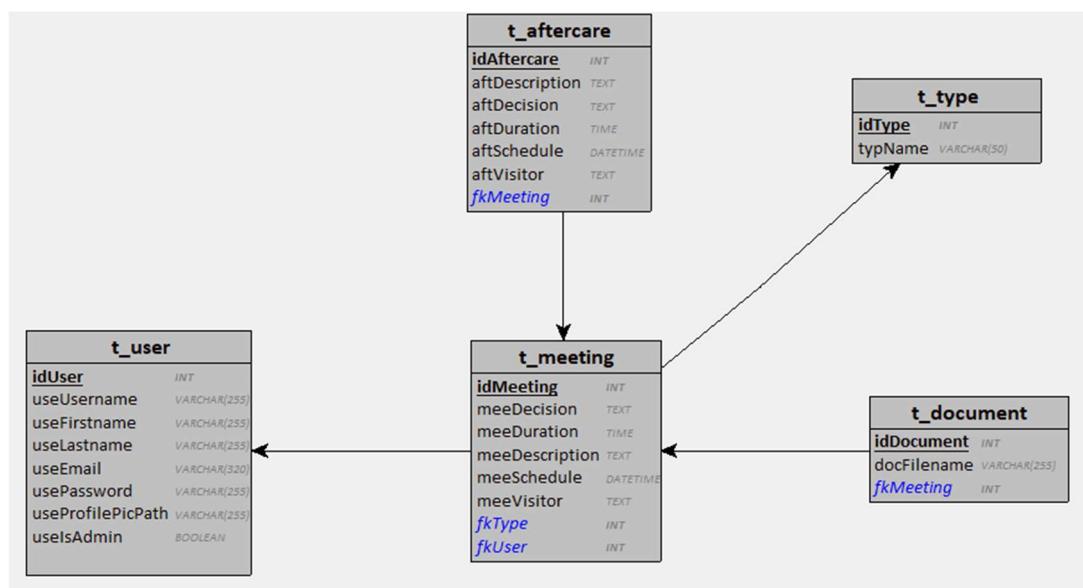
Les MCD et le MLD ci-dessous représentent la structure de la base de données utilisée dans le projet. Ces deux modèles ci-dessous respectent les conventions de nommage de l'ETML.

Cependant, pour une raison de compatibilité avec le framework utilisé, c'est-à-dire, Laravel, les noms des tables et des colonnes ont été adaptés à ses conventions de nommage.

Les entités « year » et « visitor » ont été supprimés car l'année pourrait tout simplement être récupérée dans les propriétés « schedule », puis, « visitor » est remplacé par un attribut dans les entités « meeting » et « aftercare »



4 MCD de l'application (Normes ETML)



5 MLD de l'application (Normes ETML)

2.5.3 Justification des types de valeurs dans la modélisation

Le nom de chaque table et attributs dans le MCD ont été définis en fonction des normes de codage imposées par l'ETML. Cependant dans le MLD, ce sont les normes de Laravel⁸ qui ont été utilisées. La raison de cette différence est qu'il y a des contraintes de compatibilités avec le framework.

2.5.3.1 Entité « USER »

L'entité « USER » est destinée aux utilisateurs. Les types de données et les longueurs des attributs sont définies en se basant sur la table d'utilisateur par défaut de Laravel. La longueur totale d'une adresse électronique est fixée à 320 caractères en raison de la norme RFC 3696⁹.

2.5.3.2 Type « VARCHAR(255) »

La plupart des attributs avec le type VARCHAR() ont une longueur de 255 caractères en raison de la valeur par défaut de Laravel.

2.5.3.3 Type « VARCHAR(300) »

Cette longueur d'attribut est réservée aux emails car elle respecte la norme RFC 3696 comme mentionné dans le chapitre 2.5.3.1.

2.5.3.4 Type « TEXT »

Les attributs ayant ce type est dû au fait que les données entrées seront majoritairement longues. Dans le cas du projet, ce sont des descriptions et ils atteindront une longueur assez grande rapidement.

⁸ Conventions de nommage de Laravel : <https://webdevetc.com/blog/laravel-naming-conventions>

⁹ Norme RFC 3696 : <https://datatracker.ietf.org/doc/html/rfc3696>

2.5.4 Maquettes

Les maquettes suivantes servent à avoir une vision de l'apparence de l'application web.

2.5.4.1 Desktop

2.5.4.1.1 Pages de connexion/inscription

The screenshot shows a login form titled "Connexion à un compte" (Login to an account). It includes fields for "Email ou Pseudonyme" (Email or Pseudonym) and "Mot de passe" (Password), and a "Se connecter" (Log in) button. The top right features "S'inscrire" (Sign up) and "Se connecter" (Log in) buttons. The top left corner displays the "KuriMediation" logo.

6 Page de connexion

The screenshot shows a sign-up form titled "Créer un compte" (Create an account). It includes fields for "Prénom" (First name), "Nom" (Last name), "Pseudonyme" (Pseudonym), "email@domain.com" (Email), and "Mot de passe" (Password), and a "S'inscrire" (Sign up) button. The top right features "S'inscrire" (Sign up) and "Se connecter" (Log in) buttons. The top left corner displays the "KuriMediation" logo.

7 Page d'inscription

2.5.4.1.2 Page d'accueil

The screenshot shows the KuriMediation homepage. At the top, there is a navigation bar with links for Accueil, Graphiques, A propos, Utilisateurs, Types, and a sign-out button. Below the navigation bar, there are three summary boxes: "Nombre d'entretiens au total" (231), "Moyenne d'entretiens par mois" (25), and "Entretiens à venir" (12). The main content area is titled "Entretiens" and lists upcoming appointments by month. The list includes:

- Août: M. John Doe
- Septembre: M. John Doe
- Octobre: Mme. Jane Doe
- Novembre: (empty)
- Décembre: (empty)
- Janvier: (empty)

There is also a search bar labeled "Rechercher..." and a "Trier par Date" button.

8 Page d'accueil

2.5.4.1.3 Page de graphiques

The screenshot shows the KuriMediation page for graphs. It features the same top navigation bar as the homepage. Below the navigation bar, there are three summary boxes: "Nombre d'entretiens au total" (231), "Moyenne d'entretiens par mois" (25), and "Entretiens à venir" (12). The main content area is titled "Graphique 2" and displays a bar chart with nine bars. The y-axis ranges from 0 to 30 in increments of 5. The x-axis has labels 1 through 9. The bars represent the following values:

Month	Value
1	14
2	17
3	14
4	13
5	19
6	28
7	19
8	24
9	18

At the bottom right of the chart area is a blue "Sauvegarder" button.

9 Page de graphiques

2.5.4.1.4 Page de modification du profil

KuriMediation

Modifier le profil

Pseudonyme
JojoDoe

Prénom
John

Nom
Doe

Email
john.doe@hotmail.com

Changer le mot de passe

Mot de passe actuel

Nouveau mot de passe

Confirmer le nouveau mot de passe

Confirmer

Sauvegarder

Annuler

10 Page de modification du profil

2.5.4.1.5 Page d'information sur l'entretien

KuriMediation

Accueil Graphiques A propos

... Se déconnecter

Entretien FIN4

Date : 13 septembre 2024 Type : Harcèlement

Intervenants

John Doe, Jane Doe, Harry Potter

Durée en min: minutes

Description

Harry Potter subit d'harcèlement par John et Jane Doe.

Decision

John et Jane Doe seront convoqués avec leurs parents le 14 septembre afin de discuter à propos de l'harcèlement.

Ajouter

Sauvegarder

Suivis

Suivi du 17 septembre 2024

Intervenants : Harry Potter, John Doe, Jane Doe

Description : Harry Potter est revenu après avoir reçu une bombe à eau à la figure.

Decision : John et Jane sont en heures de colle pendant 4h.

Durée : 1h16

Suivi du 26 septembre 2024

Ajouter

Supprimer

11 Page d'informations sur un entretien

2.5.4.1.6 Formulaire d'ajout d'un nouvel entretien

The screenshot shows a web-based application interface for adding a new interview. At the top, there's a navigation bar with the logo 'KuriMediation', links for 'Accueil', 'Graphiques', and 'A propos', and a 'Se déconnecter' button. Below the navigation is a date selector showing '2024-2025' and '2025-2026'. A sidebar on the left lists months from 'Janvier' to 'Juin'. The main content area has a title 'Ajouter un nouvel entretien' and a subtitle 'Entrez les informations'. It contains several input fields: 'Type intervention' (dropdown), 'Titre' (text), 'Durée en minutes' (text), 'Date' (text), 'Nom des intervenants' (text), 'Description' (text), and 'Décisions' (text). At the bottom are two buttons: 'Ajouter' (blue) and 'Annuler' (black).

12 Formulaire d'ajout d'un nouvel entretien

2.5.4.1.7 Formulaire d'ajout d'un nouveau suivi

This screenshot shows the 'Ajouter un suivi' (Add follow-up) form. At the top, it displays 'Entretien FIN4', the date '13 septembre 2024', and the type 'Harcèlement'. The main form area is identical to the one in screenshot 12, with fields for 'Durée en minutes', 'Date', 'Nom des intervenants', 'Description', and 'Décisions', along with 'Ajouter' and 'Annuler' buttons. Below this form, there are additional buttons: 'Ajouter' (light blue) and 'Supprimer' (red).

13 Formulaire d'ajout d'un suivi

2.5.4.2 Mobile

2.5.4.2.1 Page de connexion/inscription (Mobile)

KuriMediation

Créer un compte

Entrez votre email pour vous inscrire

Prénom Nom

Pseudonyme

email@domain.com

Mot de passe

Se connecter S'inscrire

En appuyant sur Continuer, vous agréez à nos Conditions de Services et Politique de confidentialité

KuriMediation

Se connecter

Entrez vos identifiants

Email ou Pseudonyme

Mot de passe

Se connecter S'inscrire

En appuyant sur Continuer, vous agréez à nos Conditions de Services et Politique de confidentialité

14 Pages d'inscription et de connexion (Mobile)

2.5.4.2.2 Page d'accueil avec barre de navigation (Mobile)

15 Pages d'accueil/Barre de navigation (Mobile)

2.5.4.2.3 Page de graphiques (Mobile)

16 Graphiques (Mobile)

2.5.4.2.4 Page de modification du profil (Mobile)

KuriMediation

Modifier le profil

Pseudonyme
JojoDoe

Prénom Nom
John Doe

Email
johndoe@notmail.com

Changer de mot de passe

Mot de passe actuel
Nouveau mot de passe
Confirmer le nouveau mot de passe
Confirmer

Sauvegarder **Annuler**

17 Modification du profil (Mobile)

2.5.4.2.5 Page d'informations sur l'entretien (Mobile)

KuriMediation

Entretien FIN4

Entretien Date : 13 septembre 2024
Type : Harcèlement Durée en min: minutes

Intervenants
John Doe, Jane Doe, Harry Potter

Description
Harry Potter subit d'harcèlement par John et Jane Doe.

Décisions
Harry Potter subit d'harcèlement par John et Jane Doe.

Preuves

- preuve1.pdf
- preuve2.pdf
- preuve3.pdf

Ajouter

Suivis
Suivi du 17 septembre 2024 **Ajouter**

Intervenants : Harry Potter, John & Jane Doe
Description : Harry Potter est revenu après avoir reçu une bombe à eau à la figure.
Decision : John et Jane sont en heures de

Sauvegarder **Supprimer**

18 Information sur l'entretien (Mobile)

2.5.4.2.6 Formulaire d'ajout d'un nouvel entretien (Mobile)

The screenshot shows a mobile application interface for 'KuriMediation'. At the top, there's a header with three dots and the brand name 'KuriMediation'. Below the header is a navigation bar with the word 'Accueil'. The main content area is titled 'Ajouter un nouvel entretien' and includes a sub-instruction 'Entrez les informations'. It contains several input fields: 'Type intervention' (dropdown), 'Titre' (text input), 'Durée en minutes' (text input), 'Date' (text input), and 'Nom des intervenants' (text input). Below these are two large text areas for 'Description' and 'Décisions'. At the bottom of the form are two buttons: a blue 'Ajouter' button and a black 'Annuler' button.

19 Formulaire Entretien (Mobile)

2.5.4.2.7 Formulaire d'ajout de suivi (Mobile)

This screenshot shows a mobile application interface for 'KuriMediation'. The top part of the screen displays a summary of a previous meeting: 'Entretien FIN4', 'Date : 13 septembre 2024', 'Type : Harclement', and 'Durée en min : minutes'. Overlaid on this summary is a second form for 'Ajouter un nouvel entretien', which has the same structure as the one in the previous screenshot: 'Entrez les informations' with fields for 'Type intervention' (dropdown), 'Titre' (text input), 'Durée en minutes' (text input), 'Date' (text input), 'Nom des intervenants' (text input), 'Description' (text area), and 'Décisions' (text area). At the bottom are the 'Ajouter' and 'Annuler' buttons.

20 Formulaire Suivi (Mobile)

2.6 Stratégie de test

2.6.1 Model Factories

Pour tester quelques tables dans la base de données, l'outil intégré dans Laravel appelé « Model Factories » qui consiste à définir modèle de création exemplaire repris du modèle Eloquent. Puis, c'est aux Seeders qui initient et remplissent la base de données.

Grâce à cela, on peut vérifier que les migrations sont bonnes et en plus de cela, en générant un grand nombre de données, on peut aussi tester le fonctionnement et la stabilité de l'application web.

2.6.2 Seeders

Comme expliqué ci-dessus, les seeders permettent d'automatiser les insertions de données dans la base de données. Combiné avec les « Model Factories », ils facilitent le développement, les tests ainsi que les démonstrations de l'application.

2.6.3 Tests manuels

N'ayant pas voire peu vu les moyens de test de Laravel appelé « Laravel DUSK », les tests manuels ont été effectués. C'est-à-dire que les tests de chaque fonctionnalité ont été faits à la main.

2.7 Risques techniques

L'intégration quelques éléments tels que l'importation et l'exportation de documents PDF, ainsi que la conversion de pages en PDF, pourraient ralentir le développement. Cela est dû au manque de maîtrise de ces technologies et à un apprentissage inachevé.

3 Réalisation

Ce chapitre représente toute la partie « pratique » de ce projet. A travers ce chapitre, le fonctionnement du code de l'application web « KuriMediation » sera expliqué ainsi que justifié. D'abord, la mise en place de l'environnement de travail sera expliquée puis chaque fonctionnalité sera expliquée.

3.1 Mise en place de UwAmp

Afin d'installer et mettre en place UwAmp, il faut tenir en compte les prérequis, c'est-à-dire, dans notre cas, avoir installé « Visual C++ Redistributable 2017 » (à partir de PHP 7.2).

Après avoir fait cela, on peut accéder à PHPMyAdmin en allant sur « localhost/mysql/ » depuis le navigateur. Pour créer la base de données, en faisant les migrations sur Laravel, une invitation proposera de la créer si ce n'est pas fait.

3.2 Installation et mise en place de Laravel 11

Afin d'installer Laravel, il est conseillé d'utiliser l'outil « Composer » qui est un outil en ligne de commande. Il suffit de lancer une commande dans un terminal sur VSCode afin d'installer tous les fichiers nécessaires au fonctionnement.

Cette commande permet de créer un projet Laravel : « composer create-project laravel/laravel KuriMediation »

3.2.1 **Connexion à la base de données grâce au fichier « .env »**

Le fichier « .env » est tout simplement le fichier de configuration de l'application web. Elle contient toutes les valeurs de configuration importantes à la connexion à la base de données.

Cependant, par défaut, le fichier « .env » se nommait « .env.example » car elle contient les informations confidentielles comme le nom de la base de données, l'identifiant, le mot de passe et autres. Il faut donc le renommer et insérer les données. Cela est dû au fait que le fichier fait partie de la liste des fichiers qui sont ignorés par Git.

Afin de pouvoir se connecter à la base de données depuis le projet, il faut remplir quelques champs dans le fichier comme dans l'image sur la prochaine page.

```

APP_NAME=KuriMediation
APP_ENV=local
APP_KEY=base64:/cRepdh51M6S934iy3oozSn8Dj5IPzWd44bHh84YUaM=
APP_DEBUG=true
APP_TIMEZONE=UTC
APP_URL=http://localhost

APP_LOCALE=fr
APP_FALLBACK_LOCALE=fr
APP_FAKEER_LOCALE=en_US

APP_MAINTENANCE_DRIVER=file
APP_MAINTENANCE_STORE=database

BCRYPT_ROUNDS=12

LOG_CHANNEL=stack
LOG_STACK=single
LOG_DEPRECATED_CHANNEL=null
LOG_LEVEL=debug

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=db_kuriMediation
DB_USERNAME=root
DB_PASSWORD=root

```

21 Fichier de configuration .env

3.3 Mise en place de Breeze

Breeze servira de fournir un point de départ avec une page de bienvenue, une page de connexion et une page d'enregistrement.

Pour installer Breeze, il faut lancer la commande dans le terminal depuis le répertoire racine du projet Laravel : « composer require laravel/breeze --dev »

Le terminal proposera quel type de package Breeze souhaité. La version utilisé dans le projet est « Laravel Breeze with Livewire scaffolding ». Ce qui signifie que la bibliothèque « Livewire » est également ajoutée.

Après avoir lancé la commande, il faut configurer l'authentification en lançant la commande : « php artisan breeze:install »

Pour finaliser, il faut installer « npm » ensuite le lancer en lançant la commande :« npm install && npm run dev ». Il est à mémoriser que la commande « npm run dev » devra être lancée à chaque fois pour compiler les ressources frontales et avoir un affichage correct.

Comme cité dans le chapitre 2.4.4, Tailwind CSS est inclus et préconfiguré à l'installation de Breeze.

3.5 Migrations

Pour créer une migration, la commande « `php artisan make:migration *nom de la migration*` » est lancée.

En raison de compatibilité avec Laravel, le nom des tables et des attributs suit la convention de Laravel.

Les différents sous-chapitres qui suivent représentent les tables créées pour les migrations ainsi que des explications de la longueur des attributs.

3.5.1 Table « users »

```
● ● ●  
1 Schema::create('users', function (Blueprint $table) {  
2     $table->id();  
3  
4     $table->string('username');  
5     $table->string('firstname');  
6     $table->string('lastname');  
7     $table->string('email', 320)->unique();  
8     $table->string('password');  
9     $table->string('profile_pic_path')->default('img/defaultprofilepic.jpg');  
10    $table->boolean('isAdmin')->default(false);  
11  
12    $table->rememberToken();  
13    $table->timestamps();  
14});
```

22 Migration de la table "users"

La table « users » est importante car elle contient les informations de l'utilisateur tels que l'email et le mot de passe (password).

L'attribut « email » a une longueur maximum de 320 caractères basé sur la convention RFC 3696 comme mentionné dans le chapitre 2.5.3.1.

L'attribut « isAdmin » sert à déterminer si l'utilisateur est administrateur ou pas. Ce rôle permet d'avoir accès aux pages de modifications d'utilisateurs et de types d'entretiens.

3.5.2 Table « meetings »

```
● ● ●

1 Schema::create('meetings', function (Blueprint $table) {
2     $table->id();
3
4     $table->string('name');
5     $table->text('decision');
6     $table->text('description');
7     $table->integer('duration');
8     $table->dateTime('schedule');
9     $table->text('visitor');
10
11
12    $table->unsignedBigInteger('user_id');
13    $table->foreign('user_id')->references('id')->on('users')->onDelete('cascade');
14    $table->unsignedBigInteger('type_id');
15    $table->foreign('type_id')->references('id')->on('types')->onDelete('cascade');
16
17    $table->timestamps();
18});
```

23 Migration de la table "meetings"

La table « meetings » est une table sur laquelle se base le sujet du projet. Elle contient les différentes données récupérées durant un entretien. L'attribut « visitor » est de type « TEXT » car elle contient les personnes concernées par le cas de médiations et comme convenu avec M. Lymberis, il est plus simple de faire ainsi que de faire plusieurs attributs pour chaque intervenant.

3.5.3 Table « aftercares »

```
● ● ●

1 Schema::create('aftercares', function (Blueprint $table) {
2     $table->id();
3
4     $table->text('decision');
5     $table->text('description');
6     $table->integer('duration');
7     $table->dateTime('schedule');
8     $table->text('visitor');
9
10    $table->unsignedBigInteger('meeting_id');
11    $table->foreign('meeting_id')->references('id')->on('meetings')->onDelete('cascade');
12
13    $table->timestamps();
14});
```

24 Migration de la table "aftercares"

La table « aftercares » est similaire à la table « meetings ». Ce qui change est le fait qu'il n'y ait plus de nom car elle appartient à un entretien.

3.5.4 Table « documents »

```
● ● ●  
1 Schema::create('documents', function (Blueprint $table) {  
2     $table->id();  
3  
4     $table->string('filename');  
5  
6     $table->unsignedBigInteger('meeting_id');  
7     $table->foreign('meeting_id')->references('id')->on('meetings')->onDelete('cascade');  
8  
9     $table->timestamps();  
10});
```

25 Migration de la table "documents"

La table « documents » contiendra le nom du fichier PDF ajouté par l'utilisateur ainsi que l'identifiant de l'entretien. Elle permet d'ajout ou supprimer un document PDF stocké dans le répertoire « public/pdf/*id de l'entretien* ».

3.5.5 Table « types »

```
● ● ●  
1 Schema::create('types', function (Blueprint $table) {  
2     $table->id();  
3  
4     $table->string('name');  
5  
6     $table->timestamps();  
7});
```

26 Migration de la table "types"

La table « types » contient juste un string qui aura comme valeur, le nom du type d'entretien.

3.6 Mise en place des contrôleurs

Dans le projet, il n'y a que des contrôleurs de ressources ont été utilisés. Cependant, il y a aussi des contrôleurs de bases. La différence entre ces deux sont que le contrôleur de ressources contient déjà un modèle contenant les différentes opérations CRUD suivant une convention standard et que le routage est plus simple. Tandis que le contrôleur de base est plus flexible quand on n'a pas besoin d'opérations CRUD.

La mise en place des contrôleurs se fait en lançant la commande : « `php artisan make:controller SampleController` » ou bien si c'est un contrôleur de ressources il faut ajouter « `--resource` » à la fin de la commande. Après la commande lancée, un fichier avec le nom du contrôleur sera créé dans le répertoire « `app/http/Controllers` ».

3.7 Mise en place des routes

Le routage de Laravel fonctionne comme un système central qui gère les requêtes HTTP pour ensuite les diriger vers les contrôleurs ou bien directement vers des vues.

Comme mentionné dans le chapitre précédent, il y a des routes pour les contrôleurs normaux et contrôleurs de ressources.

Pour faire un lien entre les routes et les contrôleurs. Il faut indiquer la classe de la route ainsi que la méthode. Tandis que pour les contrôleurs de ressources, il ne faut pas spécifier la méthode du contrôleur, ainsi que le nom de la méthode à utiliser.

Dans une application Laravel, les routes par défaut sont définies dans plusieurs fichiers. Ces fichiers sont « `web.php` » qui gère les routes web (cookies, sessions, etc...) et « `console.php` » qui gère les commandes Artisan.

Cependant, vu que mon projet Laravel est fait avec Breeze. Ce qui fait qu'un autre fichier appelé « `auth.php` » est ajouté dans le répertoire « `routes` » (à ne pas confondre avec le fichier `auth.php` dans le répertoire « `config` ». Ce fichier sert à gérer les routes d'authentification. Et dans ces routes-là, elles sont séparées en 2 types d'utilisateurs. Ceux qui sont authentifiés (`auth`) et ceux qui ne sont pas authentifiés (`guest`).

3.7.1 web.php

Dans ce fichier se trouvent mes différentes routes pour mes contrôleurs (`MeetingController`, `AftercareController` et `GraphicController`).

3.7.1.1 Authentification

```
// Routes for non-authenticated users
Route::middleware(['guest'])->group(function () {
    Route::get('/', function () {
        if (Auth::check()) {
            return redirect()->route('home');
        } else {
            return redirect()->route('login');
        }
    });

    // Route for the login page
    Route::get('/login', function () {
        return view('livewire.pages.auth.login');
    })->name('login');
});

});
```

27 Fichier web.php

Cette image ci-dessus représente la partie qui redirige vers soit la page d'accueil soit la page de connexion si l'utilisateur n'est pas authentifié.

3.7.1.2 MeetingController

```
// Route for displaying the homepage WITHOUT/WITH existing meetings
Route::get('/home/{year}', [MeetingController::class, 'index'])
    ->name('meeting.index')
    ->middleware(['auth']);
Route::get('/home/{year?}', [MeetingController::class, 'index'])
    ->name('meeting.index')
    ->middleware(['auth']);
Route::post('/meeting/update/{meetingId}', [MeetingController::class, 'update'])
    ->name('meeting.update')
    ->middleware(['auth']);
Route::get('/meeting/destroy/{meetingId}', [MeetingController::class, 'destroy'])
    ->name('meeting.destroy')
    ->middleware(['auth']);
Route::resource('meeting', MeetingController::class)
    ->except('index', 'update', 'destroy');
```

28 Routes pour le MeetingController

Dans la route « meeting.index » le paramètre « year ? » contient un point d'interrogation car il n'est pas obligatoire. Si la valeur est assignée, elle affiche la page avec les valeurs. Dans le cas contraire, elle affiche une page qui ne contient zéro valeur. Cependant une route « /home » a été ajoutée pour des problèmes dans mon contrôleur.

Le contrôleur « MeetingController » est un contrôleur de ressources mais comporte aussi d'autres routes afin qu'ils prennent plus de paramètres que sur le modèle de base. Ce qui explique le « `->except('index', 'update', 'destroy');` ».

La raison pourquoi il y a des autres routes que la route de ressources est pour la mise en forme de l'URL. Car avec la ressource l'url finirait avec « /meeting/*id de l'entretien » alors qu'avec la route personnalisée elle finit comme dans l'image, en « /home/{year?} ».

3.7.1.3 AftercareController

```
// Routes for aftercare resource and store, destroy methods
Route::post('/aftercare/store/{meetingId}', [AftercareController::class, 'store'])
    ->name('aftercare.store')
    ->middleware(['auth']);
Route::get('/aftercare/edit/{meetingId}', [AftercareController::class, 'edit'])
    ->name('aftercare.edit')
    ->middleware(['auth']);
Route::put('/aftercare/update/{meetingId}', [AftercareController::class, 'update'])
    ->name('aftercare.update')
    ->middleware(['auth']);
Route::get('/aftercare/destroy/{aftercareId}', [AftercareController::class, 'destroy'])
    ->name('aftercare.destroy')
    ->middleware(['auth']);
Route::resource('aftercare', AftercareController::class)
    ->except('store', 'edit', 'update', 'destroy');
```

29 Routes pour AftercareController

Comme précisé avec « MeetingController », les routes personnalisées sont que pour l'esthétique de l'URL.

3.7.1.4 GraphicsController

```
// Route that shows the graphics page
Route::get('/graphics/{year?}', [GraphicController::class, 'index'])
    ->name('graphic.index')
    ->middleware(['auth']);
```

30 Route pour GraphicController

3.7.1.5 DocumentController

```
// Routes for the uploading, displaying and deleting the document
Route::post('/document/upload/{meetingId}', [DocumentController::class, 'upload'])
    ->name('document.upload')
    ->middleware(['auth']);
Route::get('/document/show/{meetingId}/{fileName}', [DocumentController::class, 'show'])
    ->name('document.show')
    ->middleware(['auth']);
Route::get('/document/destroy/{meetingId}/{fileName}', [DocumentController::class, 'destroy'])
    ->name('document.destroy')
    ->middleware(['auth']);
```

31 Routes pour DocumentController

Pour ce contrôleur, il est le seul à avoir une route avec un nom différent car elle conçue spécialement pour router à la fonction « upload » qui permet de pourvoir téléverser un fichier PDF au stockage du projet ainsi que d'insérer les données dans la table « documents ».

3.7.2 auth.php

```

Route::middleware('guest')->group(function () {
    Volt::route('register', 'pages.auth.register')
        ->name('register');

    Volt::route('login', 'pages.auth.login')
        ->name('login');

    Volt::route('forgot-password', 'pages.auth.forgot-password')
        ->name('password.request');

    Volt::route('reset-password/{token}', 'pages.auth.reset-password')
        ->name('password.reset');
});

Route::middleware('auth')->group(function () {
    Volt::route('verify-email', 'pages.auth.verify-email')
        ->name('verification.notice');

    Route::get('verify-email/{id}/{hash}', VerifyEmailController::class)
        ->middleware(['signed', 'throttle:6,1'])
        ->name('verification.verify');

    Volt::route('confirm-password', 'pages.auth.confirm-password')
        ->name('password.confirm');
});

```

32 Routes dans le fichier "auth.php"

Ce fichier contient les différentes routes préconfigurées pour les fonctionnalités incluses dans Breeze telles que l'inscription, la connexion, la réinitialisation du mot de passe, la vérification de l'email et plus. Les routes sont groupées dans des « middlewares »

Un middleware un composant intermédiaire qui permet de gérer les requêtes HTTP avant qu'elles n'atteignent les autres routes. Dans le cas de ce projet, cela permet de séparer ce que les utilisateurs authentifiés et non-authentifiés peuvent faire.

3.7.3 console.php

```

Artisan::command('inspire', function () {
    $this->comment(Inspiring::quote());
})->purpose('Display an inspiring quote')->hourly();

```

33 Commandes dans le fichiers "console.php"

Comme mentionné avant, le fichier « console.php » sert à créer des commandes artisan personnalisées.

3.8 Authentification

Comme précisé dans au début du rapport, l'application exploite Laravel avec Breeze. Au moment de l'installation et de la mise en place de Breeze, tout est déjà préconfiguré. Cependant, afin d'aligner les attributs aux besoins de la table « users », plusieurs attributs ont été ajoutées/changées.

```
Schema::create('users', function (Blueprint $table) {
    $table->id();
    $table->string('name');
    $table->string('email')->unique();
    $table->timestamp('email_verified_at')->nullable();
    $table->string('password');
    $table->rememberToken();
    $table->timestamps();
});
```

34 Migrations de la table "users" par défaut



```
1 Schema::create('users', function (Blueprint $table) {
2     $table->id();
3
4     $table->string('username');
5     $table->string('firstname');
6     $table->string('lastname');
7     $table->string('email', 320)->unique();
8     $table->string('password');
9     $table->string('profile_pic_path')->default('img/defaultprofilepic.jpg');
10    $table->boolean('isAdmin')->default(false);
11
12    $table->rememberToken();
13    $table->timestamps();
14});
```

35 Migration de la table "users" après changements

Comme montrés dans les deux images ci-dessus, les changements sont :

- Le prénom et le nom ont été ajoutés.
- La longueur maximale de caractères pour l'email est de 320.
- La booléenne « isAdmin » permet de définir si l'utilisateur est administrateur.
- Le champ « profile_pic_path » a été ajouté initialement afin de permettre à l'utilisateur de changer mais la fonctionnalité n'a pas été faite.

Il ne faut pas oublier que quand on ajoute des attributs il faut aussi aller dans le modèle et ajouter l'attribut dans « \$fillable ». Car, cela permet de remplir de façon simple et rapide plusieurs attributs d'un modèle en utilisant un tableau de données. Si cette étape n'est pas exécutée, les attributs ajoutés ne seront pas pris en compte.

3.9 Gestion d'entretiens

Les entretiens représentent le sujet principal du projet. Ils permettent de mettre en communication les personnes ayant le besoin de s'exprimer avec les médiateurs.

3.9.1 Enregistrement d'un entretien

La page principale contient une liste d'entretiens triés par mois ainsi que trois statistiques représentants : le nombre d'entretiens au total par année, le temps passé au total pour les entretiens et ses suivis et le nombre d'entretiens à venir.

The screenshot shows the KuriMediation application's main dashboard. At the top, there is a navigation bar with links: Accueil (highlighted in blue), Graphiques, A Propos, Utilisateurs, and Types. On the far right, there is a user profile icon labeled "admin". Below the navigation bar, there is a date selector showing "2024". Three large boxes provide summary statistics: "Nombre d'entretiens au total" (1), "Temps passés durant l'année" (04h14min), and "Entretiens à venir" (0). Below these statistics is a table titled "Entretiens" with a list of months from Janvier to Août. Each month entry includes fields for Nom, Intervenants, Description, and Type, along with edit and delete icons. The table has a "Rechercher" (Search) button and a "Trier par" (Sort by) dropdown set to "Date".

36 Page principale du site

A partir de cette page, un bouton bleu permet d'afficher le formulaire pour enregistrer un nouvel entretien. Avant d'expliquer le fonctionnement du formulaire, le fonctionnement du bouton sera d'abord expliqué.

The screenshot shows a web application interface. On the left, there's a sidebar titled 'Nombre d'entretiens au total' with the value '1'. Below it is a section titled 'Entretiens' containing a list of months from 'Janvier' to 'Août'. Under each month, there are input fields for 'Nom:' (e.g., 'fajhw bd'), 'Intervenants:' (e.g., 'dawd'), and 'Date' (e.g., '2024-01-01'). To the right of the sidebar is a main content area. At the top, there's a 'Retour' button and a note: 'Ajouter une nouvel entretien' followed by 'Veuillez insérer les informations ci-dessous'. Below this are several input fields: 'Type d'intervention' (dropdown menu showing 'Harcèlement'), 'Titre de l'entretien' (text input), 'Description' (text input), 'Date' (date input with placeholder 'jj.mm.aaaa'), and 'Nom des intervenants' (text input). At the bottom of this form is a blue 'Ajouter' button. To the right of the form is a sidebar titled 'Entretiens à venir' with the value '0'. It includes a search bar, a 'Trier par' dropdown set to 'Date', and a '+' button. There are also icons for edit and delete.

37 Formulaire pour l'enregistrement d'un entretien

Le bouton bleu est un fichier « livewire » ayant une variable booléenne qui change de valeur à chaque fois que le bouton est pressé. Un « if » vérifiera alors sa valeur.

```
new class extends Component
{
    public $showForm = false;
    public $types;

    public function toggleForm()
    {
        $this->showForm = !$this->showForm;
    }

    public function mount(){
        $this->types = Type::all();
    }
}; ?>
<div>
    {{-- Button that toggles the visibility of the form --}}
    <button wire:click.prevent="toggleForm" class="ml-2 py-2 px-3 rounded-lg border text-white font-extrabold">
        +
    </button>
    @if ($showForm)
        {{-- Le reste du code... --}}
    
```

38 Fonctionnement de la fonction « toggleForm »

Dans le formulaire, on peut constater qu'il manque « la décision » et « la durée » de l'entretien. Cela est tout à normal car le but est que l'entretien puisse être planifié avant, puis la prise de décision et la durée seront remplies à la fin de l'entretien.

```
// Function that creates a new meeting with its parameters
public function store(Request $request){
    // Instanciate a new meeting
    $meeting = new Meeting();
    // Validate the data
    $validatedData = $request->validate([
        'name' => 'required|string|max:255',
        'description' => 'required|string|max:255',
        'schedule' => 'required|date',
        'type_id' => 'required|exists:types,id',
        'visitor' => 'required|string|max:255',
    ]);
    // Define the missing data
    $validatedData['duration'] = 0;
    $validatedData['decision'] = '';
    $validatedData['user_id'] = Auth::id();

    // Inserts the validated data and creates the meeting
    $meeting->create($validatedData);

    // Redirect to the homepage
    return redirect()->route('meeting.index');
}
```

39 Méthode `store()` dans `MeetingController`

Les requêtes sont validées d'abord en vérifiant si le champ est obligatoire, son type et sa longueur maximale. Puis, les valeurs non remplies telles que la durée et la décision sont à leur valeur minimale mais pas vides.

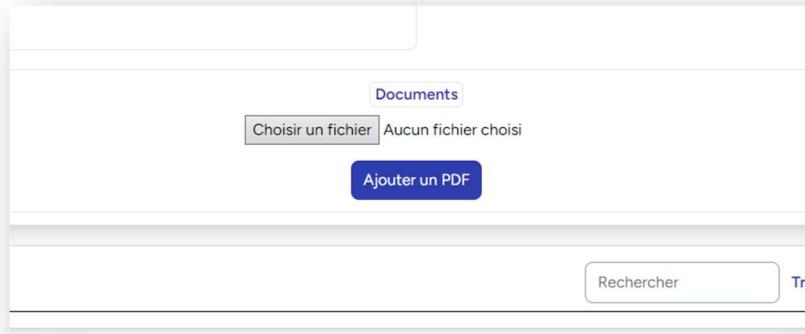
Après avoir créé l'entretien, il est possible d'afficher la page d'information de l'entretien, la page de modification ou bien supprimer l'entretien avec les boutons sur la page principale.



40 Boutons pour gérer l'entretien

3.9.2 Document PDF

Durant ou après un entretien, il arrive souvent qu'on fasse remplir un formulaire par le ou les clients/clients. Afin de pouvoir garder le document dans un endroit sûr et facile à récupérer, il est possible d'insérer des documents PDF. Le fonctionnement sera expliqué dans les prochains points.



41 Bouton pour ajouter un document PDF

3.9.3 Liste d'entretiens

Un des points évalués est le listage. Sur la page d'accueil, une liste d'entretiens triée par mois est affichée.

3.9.3.1 Partie HTML

```
{{-- Checks if the users has meetings or not --}}
@if ($userMeetings != 0)
    @for ($i = 0; $i < 12; $i++)
        <ul>
            <li class="px-2 py-2 border-b shadow-sm rounded-lg border-gray-300 font-extrabold"><?=months[$i]?></li>
        @foreach ($userMeetings[$i] as $meeting)
            <li class="flex justify-between px-4 py-2 border-b rounded-sm text-gray-500">
                <div class="flex flex-col md:block md:space-x-8">
                    <span>Nom: {{ $meeting->name }}</span>
                    <span>Intervenants: {{ $meeting->visitor }}</span>
                    <span>Description: {{ $meeting->description }}</span>
                    @foreach ($types as $type)
                        @if ($type->id == $meeting->type_id)
                            <span>Type: {{ $type->name }}</span>
                        @endif
                    @endforeach
                </div>
                <div class="flex space-x-2 items-center">
                    <span><a href="{{ route('meeting.show', $meeting->id) }}"><i class="fa fa-info-circle fa-lg" ></i></a></span>
                    <span><a href="{{ route('meeting.edit', $meeting->id) }}"><i class="fa fa-edit fa-lg" ></i></a></span>
                    <span>
                        <a href="{{ route('meeting.destroy', $meeting->id) }}" onclick="return confirm('Voulez-vous supprimer cet
                            <i class="fa fa-trash fa-lg"></i>
                        </a>
                    </span>
                </div>
            </li>
        @endforeach
    @endfor
@else
    <li class="px-2 py-2 border-b rounded-sm">Aucune donnée</li>
@endif
```

42 Listage d'entretiens

Dans le « if » si la variable « \$userMeetings » qui est un tableau n'a pas une valeur de 0, elle affiche la liste avec les entretiens. Dans le cas contraire, elle affiche « Aucune donnée ».

3.9.3.2 Méthode getUserAllMeetings(\$year)

Cette méthode permet de récupérer tous les entretiens de l'utilisateur et les mets dans la variable « \$userMeetings » qui est un tableau. Les valeurs de ce tableau sont utilisées pour l'affichage de la liste d'entretiens sur la page d'accueil.

```
/** Function that retrieves all meetings the user has for the chosen year.
 * returns an array sorted by months
 */
public function getUserAllMeetings($year){
    // Gets all user meetings based on the year
    for($i = 0; $i < 12; $i++){
        $userMeetings[$i] = Meeting::where('user_id', Auth::id())
            ->whereYear('schedule', $year)
            ->whereMonth('schedule', $i+1)
            ->get();
    }
    return $userMeetings;
}
```

43 Méthode getUserAllMeetings() dans MeetingController

Dans la méthode on peut voir une boucle « for » qui récupère les entretiens par mois et les entrent dans le tableau « userMeetings » douze fois (Le nombre de mois dans une année).

3.9.4 Statistiques

Après avoir créé des entretiens il est possible de récupérer les données et en faire des statistiques.

3.9.4.1 Méthode countUserTimeSpent(\$year)

Cette méthode permet de calculer et retourner le temps consacré aux entretiens et les suivis dans l'année spécifiée avec la variable « \$year ». Cette valeur sera utilisée pour être affichée sur la page d'accueil. (**Image sur la page 35**)

```
/**
 * Function that returns the total amount of time spent for meetings and its aftercares
 * for the user based on the chosen year
 */
public function countUserTimespent($year){
    $total = 0; // Define the total
    $meetings = Meeting::where('user_id', Auth::id()) // Get all meetings by the year
        ->whereYear('schedule', $year)
        ->get();
    // Sum the values into $total
    foreach($meetings as $meeting){
        $total += $meeting->duration;
        $meetingsAftercares = Aftercare::where('meeting_id', $meeting->id)->get();
        foreach($meetingsAftercares as $aftercare){
            $total += $aftercare->duration;
        }
    }
    $hours = floor($total / 60); // Get the lowest value
    $minutes = $total % 60; // Get the minutes from the hours
    return sprintf('%02dh%02dmin', $hours, $minutes);
}
```

44 Méthode `countUserTimespent()` dans `MeetingController`

3.9.4.2 Méthode `countUserAllMeetings($year)`

Cette méthode permet calculer le nombre d'entretiens et ses suivis de l'année spécifiée avec la variable « `$year` » dont le résultat retourné servira à être affiché sur la page d'accueil.

```
/**
 * Function that returns the total amount of meetings the user has for the chosen year
 */
public function countUserAllMeetings($year){
    $total = Meeting::where('user_id', Auth::id())->whereYear('schedule', $year)->get();
    return $total->count();
}
```

45 Méthode `countUserAllMeetings()` dans `MeetingController`

3.9.4.3 Méthode countUpcomingMeetings(\$year)

Cette méthode permet de calculer le nombre d'entretiens à venir dont le résultat retourné servira à être affiché sur la page d'accueil.

```
/*
 * Function that calculates the number of meetings upcoming based on the current date
 */
public function countUpcomingMeetings($year){
    $meetings = Meeting::where('user_id', Auth::id())->whereYear('schedule', $year)->get();
    // dd($meetings);
    $count = 0;
    foreach($meetings as $meeting){
        if($meeting->schedule > today()){
            $count++;
        }
    }
    return $count;
}
```

46 Méthode countUpcomingMeetings() dans *MeetingController*

Le fonctionnement est de vérifier que la date de chaque entretien est plus tard que la date d'aujourd'hui (fonction `today()` qui retourne la date d'aujourd'hui).

Cependant, par rapport au points évalués, cette statistique n'en fait pas partie. Alors la méthode « `getAvgTimeSpent()` » a été ajoutée.

3.9.4.4 Méthode getAvgTimeSpent(\$year)

Cette méthode permet de calculer la moyenne de temps passée pour les entretiens et ses suivis en une année. Le résultat sera affiché sur la page d'accueil.

```
/*
 * Function that calculates the average time spent on meetings and its aftercares in a year
 */
public function getAvgTimeSpent($year){
    $meetings = Meeting::where('user_id', Auth::id())->whereYear('schedule', $year)->get(); // Get meetings
    $totalTime = 0; // Total of time spent
    $counter = 0; // Counter for the number of meetings and aftercares
    foreach($meetings as $meeting){
        $aftercares = Aftercare::where('meeting_id', $meeting->id);
        $totalTime += $meeting->duration;
        foreach($aftercares as $aftercare){
            if($aftercare->duration > 0){ // If the duration of is higher than 0 it counts
                $totalTime += $aftercare->duration;
                $counter++;
            }
        }
        $counter++;
    }
    $AvgTimeSpent = $totalTime / $counter; // Get the average
    $hours = floor($AvgTimeSpent / 60); // Get the hour leaving the minutes
    $minutes = $AvgTimeSpent % 60; // Get the minutes from the hours

    return sprintf('%02dh%02dmin', $hours, $minutes);
}
```

47 Méthode getAvgTimeSpent()

3.10 Gestion des suivis

Quand le client ou la cliente reviennent pour le même sujet pour la deuxième fois, on appelle cela « un suivi ». Les suivis sont associés à un entretien afin de pouvoir plus facilement retrouver les informations.

Les suivis sont affichés sur la page d'informations d'entretien ou la page de modification de l'entretien.

The screenshot shows a web form for managing an interview. At the top, there are fields for 'Toto' (name), 'Durée:' (duration) set to 0, and a 'Decision' section with a text input field 'Entrez les données'. A 'Sauvegarder' (Save) button is located to the right. Below this, there's a 'Documents' section with a 'Choisir un fichier' (Select file) button and a message 'Aucun fichier choisi' (No file selected). A 'Ajouter un PDF' (Add a PDF) button is also present. At the bottom, a red box highlights the 'Suivis' section, which includes a 'Rechercher' (Search) button, a 'Trier par' (Sort by) dropdown set to 'Date', and a '+' button. Below this section, the interview details are listed: Date: 03.06.2024, Intervenants: Titi, Description: Toto, Durée: 45 mins. To the right of these details are edit and delete icons.

48 Liste de suivis sur la page d'informations de l'entretien

L'ajout du suivi est le même que pour l'entretien. Cependant, ayant eu un problème avec Livewire, le bouton ne marche pas pour afficher le formulaire. Alors, une page de redirection a été faite.

Ajouter une nouveau suivi
Veuillez insérer les informations ci-dessous

Date

Durée

Nom des intervenants

Description

Decision

49 Formulaire d'ajout d'un suivi

Le contrôleur « AftercareController » est simplement un contrôleur de ressources. Les calculs du temps passés s'effectuent dans le contrôleur « MeetingController ».

3.11 Validation des données et Gestion des erreurs

Quand on entre des valeurs dans les formulaires, il est important de vérifier que les valeurs entrées sont correctes et respectent le type de valeurs censé entrer. C'est ce à quoi servent les validateurs de données et la gestion des erreurs.

La gestion des données s'effectue avec la balise « <x-input-error> » qui est un composant « Livewire » inclus avec le package Breeze. Elle vérifie la validation de données dans le contrôleur.

```
<x-input-error class="mt-2" :messages="$errors->get('visitor')"/>
```

50 Exemple d'erreur avec le champ "visitor"

```

@props(['messages'])

@if ($messages)

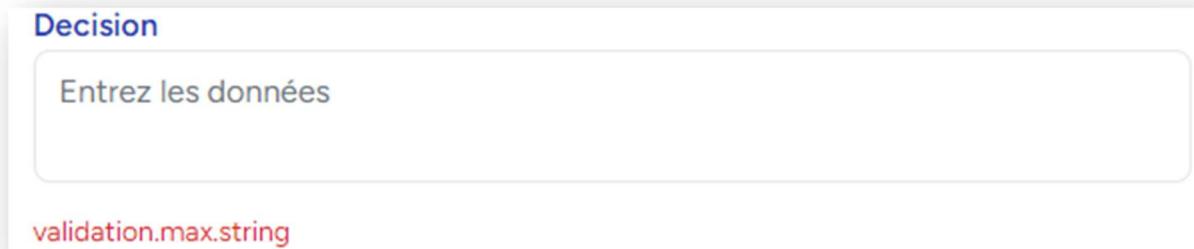


```

51 Contenu du fichier "input-error.blade.php"

La validation des données s'effectue dans le contrôleur

Dès qu'une erreur est présente, elle s'affiche juste à côté du champ avec le message du type d'erreur. Par exemple, avec l'erreur ci-dessous j'ai dépassé la valeur maximal de 255 caractères.



52 Erreur de validation du champ "decision"

3.12 Insertion de documents PDF liés à un entretien

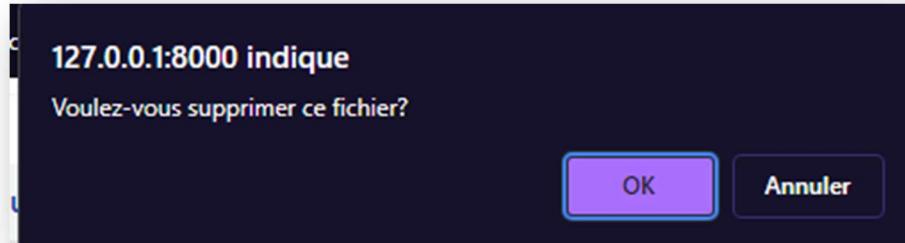
Comme mentionné dans la partie 3.9.2 « Document PDF » il est possible d'importer des fichiers PDF pour le lier à l'entretien.



53 Importation de document PDF

La partie encadrée en rouge représente le nom du fichier. En cliquant dessus, on peut visualiser le PDF grâce à la fonctionnalité intégrée dans le navigateur.

La partie encadrée en violet permet de supprimer le fichier PDF. En cliquant dessus, un message de confirmation apparaîtra en cas d'appuis accidentels.



54 Message de confirmation de suppression

Afin de réussir à mettre en place cette fonctionnalité, la classe « Storage » intégrée dans Laravel a été utilisée.

3.12.1 Configuration du disque « pdf »

Afin de créer le répertoire, la classe « Storage » possède un fichier de configuration dans le chemin « config/filesystems.php ». Ce fichier contient la configuration de « disques » qui sont soit « local » comme les disques « local » et « public » soit « cloud » comme « s3 » qui sont disponibles de base.

```
'disks' => [
    'local' => [
        'driver' => 'local',
        'root' => storage_path('app'),
        'throw' => false,
    ],
    'public' => [
        'driver' => 'local',
        'root' => storage_path('app/public'),
        'url' => env('APP_URL').'/storage',
        'visibility' => 'public',
        'throw' => false,
    ],
    's3' => [
        'driver' => 's3',
        'key' => env('AWS_ACCESS_KEY_ID'),
        'secret' => env('AWS_SECRET_ACCESS_KEY'),
        'region' => env('AWS_DEFAULT_REGION'),
        'bucket' => env('AWS_BUCKET'),
        'url' => env('AWS_URL'),
        'endpoint' => env('AWS_ENDPOINT'),
        'use_path_style_endpoint' => env('AWS_USE_PATH_STYLE_ENDPOINT', false),
        'throw' => false,
    ],
]
```

55 Les trois disques par défaut

Afin d'utiliser que le dossier spécifié pour les documents PDF, un nouveau disque de stockage nommé « pdf » a été configuré. Ce disque est configuré afin que le chemin ne mène pas dans le dossier « public » car pour de raisons de sécurité et de confidentialité, on ne voudrait pas qu'ils soient accessibles par n'importe qui.

```
// PDF Disk
'pdf' => [
    'driver' => 'local',
    'root' => storage_path('app/pdf'),
    'visibility' => 'private',
    'throw' => false,
],
```

56 Disque "pdf"

Après avoir configuré le disque « pdf », un contrôleur appelé « DocumentController » a été créé avec des méthodes de bases comme « show », « upload » et « destroy ». Ces méthodes servent à afficher un document sur un onglet du navigateur, importer un document ou de le supprimer du répertoire et de la base de données.

3.12.2 Méthode upload(Request \$request, \$meetingId)

Cette méthode prend deux paramètres : \$request qui est la requête et \$meetingId qui est l'identifiant de l'entretien car dans la table « documents » appartient à la table « meetings ».

```
public function upload(Request $request, $meetingId)
{
    // Validate the uploaded file
    $request->validate([
        'document' => 'required|mimes:pdf|max:2048',
    ]);

    // Create a folder for the meeting using the meeting's ID if it doesn't exist yet
    $FolderPath = Auth::id() . "/" . $meetingId;
    if (!Storage::disk('pdf')->exists($FolderPath)) {
        Storage::disk('pdf')->makeDirectory($FolderPath);
    }

    // Check if the document is valid
    if ($request->file('document')->isValid()) {
        // Store the uploaded file within the meeting's folder
        $request->file('document')->storeAs($FolderPath, $request->file('document')->getClientOriginalName(), 'pdf');

        // Insert the new document into the database
        Document::create([
            'filename' => $request->file('document')->getClientOriginalName(),
            'meeting_id' => $meetingId,
            'created_at' => now(),
            'updated_at' => now(),
        ]);

        // Redirect to the previous page
        return redirect()->back();
    }

    // Redirect to the previous page with an error
    return back()->with('error', 'Erreur de montage du fichier. Vérifiez que c\'est au format PDF.');
}
```

57 Méthode upload() dans DocumentController

La variable « \$FolderPath » représente le répertoire dans lequel les documents PDF sont stockés. Elle constitue de « Auth::id() » qui est une fonction de Laravel renvoyant l'identifiant de l'utilisateur actuel et de « \$meetingId » qui est l'identifiant de l'entretien auquel le document PDF est lié. Si le répertoire n'existe pas, il est créé.

Ensuite, après que le document soit validé, il est vérifié encore une fois avec la fonction « isValid() », stocké dans le répertoire et dans la base de donnée.

Si tout se passe bien, l'utilisateur est redirigé sur la page précédente. Dans le cas contraire, sur la page précédente avec une erreur.

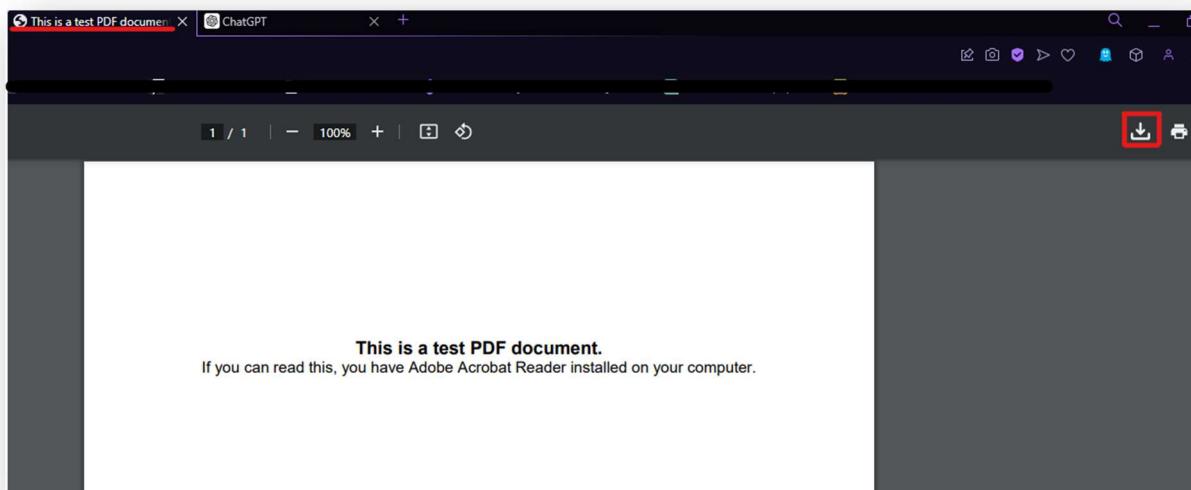
3.12.3 Méthode show(\$meetingId, \$fileName)

La méthode « show() » permet d'afficher le document PDF dans le navigateur. Il est également possible de télécharger le document depuis cette page. Le paramètre « \$meetingId » représente l'identifiant de l'entretien et le paramètre « \$fileName » le nom du fichier que l'on souhaite afficher. (**Image sur la page suivante**).

```
/**  
 * Function that displays the pdf in the browser and make it downloadable  
 */  
public function show($meetingId, $fileName)  
{  
    // Get the pdf path  
    $filePath = Auth::id() . '/' . $meetingId . '/' . $fileName;  
  
    // Check if the file exists  
    if (Storage::disk('pdf')->exists($filePath)) {  
        // Get the path  
        $path = Storage::disk('pdf')->path($filePath);  
  
        // Return the file for download  
        return response()->file($path);  
    }  
    // If the file does not exist, redirect with an error  
    return redirect()->back()->with('error', 'File not found.');
```

58 Méthode show() dans DocumentController

Avant de procéder à l'affichage, la vérification de l'existence du fichier est effectuée. Si le fichier existe la méthode retourne « response()->file(\$path) », une réponse HTTP affichant le fichier. Dans le cas contraire, l'utilisateur est redirigé en arrière avec une erreur.



59 Affichage du document PDF dans le navigateur

On peut voir sur cette image, qu'il est possible de télécharger le document PDF ou bien de l'imprimer directement.

3.12.4 Méthode destroy(\$meetingId, \$fileName)

Cette méthode permet de supprimer le fichier du répertoire ainsi que de la base de données. Elle reprend les mêmes paramètres qu'avec la méthode show().

```
/*
 * Function that deletes the file from the directory and the database
 */
public function destroy($meetingId, $fileName)
{
    // Get the pdf's path
    $filePath = Auth::id() . '/' . $meetingId . '/' . $fileName;
    if (Storage::disk('pdf')->exists($filePath)) {
        Storage::disk('pdf')->delete($filePath);
        // Remove from the database
        Document::where('filename', $fileName)->where('meeting_id', $meetingId)->delete();
        return redirect()->back();
    }
    // If the file does not exist, redirect with an error
    return redirect()->back()->with('error', 'File not found.');
}
```

60 Méthode destroy() dans DocumentController

3.13 Affichage sous format graphique des statistiques

Les graphiques sont utiles afin de représenter visuellement les statistiques et voir l'évolution du nombre d'entretiens passés dans l'espace d'une année.

« Laravel Charts ¹⁰¹¹ » un package crée par une personne appelée « ConsoleTVs » sur GitHub. Elle permet de simplifier la création et la visualisation de graphiques. Ce package est très pratique car il est possible de créer plusieurs types de graphiques en utilisant de diverses bibliothèques telles que « ChartJS », « HighCharts », « FusionCharts » et d'autres encore.

3.13.1 Installation et mise en place de Laravel Charts

Pour l'installation, il faut lancer la commande :

```
chris@Kurisu-Laptop MINGW64 /i/Github/TPI_KuriMedia/KuriMediation (main)
$ composer require consoletvs/charts:6.*
```

61 Installation de Laravel Charts

¹⁰ Github de Laravel Charts : <https://github.com/ConsoleTVs/Charts>

¹¹ Documentation sur Laravel Charts : <https://charts.erik.cat>

Ensuite il faut publier la configuration par défaut de la librairie avec la commande :

```
chris@Kurisu-Laptop MINGW64 /i/Github/TPI_KuriMedia/KuriMediation (main)
$ php artisan vendor:publish --tag=charts_config
```

61 Publication de la configuration de Laravel Charts

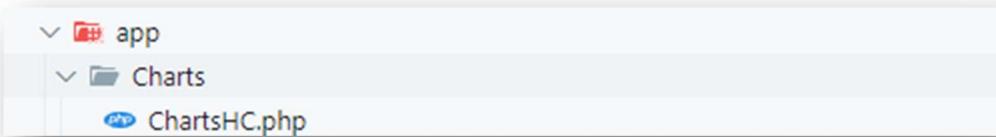
Après avoir fait ceci, la création d'un nouveau graphique se fait avec la commande :

```
chris@Kurisu-Laptop MINGW64 /i/Github/TPI_KuriMedia/KuriMediation (main)
$ php artisan make:chart Exemple
```

63 Crédit d'un nouvel objet Chart

La configuration est définie pour utiliser la librairie « ChartJS » par défaut mais en spécifiant le nom de la librairie souhaitée après avoir inséré le nom de l'objet lors de la création, il est possible de créer un objet Chart spécifiquement pour la librairie que l'on souhaite utiliser.

Après avoir créé le nouvel objet « Charts » un nouveau fichier dans le répertoire « app\Charts » a été ajouté.



64 Nouveau fichier Charts

Afin de pouvoir créer les graphiques, un contrôleur appelé « GraphicController » a été créé. Ce contrôleur servira à afficher la page contenant les graphiques.

Parmi les graphiques, dans le cas de ce projet, il y en a quatre dont deux sont affichés pour les versions mobiles. Le premier et le troisième sont des graphiques affichant la somme de temps passé pour l'entretiens par mois. Le deuxième et le quatrième sont des graphiques affichant le nombre d'entretiens passés par catégorie en une année.

Ayant beaucoup de lignes de code, la méthode « index(\$year) » qui permet d'afficher la page contenant les graphiques a été coupée en plusieurs parties afin de faciliter l'explication.

3.13.2 Récupération des valeurs pour les entretiens et ses suivis par mois

```
// Check if the year has been defined, if so, the user gets to the view with the data.
if($year){
    // Get user's meetings based on the year
    $meetings = Meeting::where('user_id', Auth::id())->whereYear('schedule', $year)->get();

    // Get values of meetings per MONTH
    foreach(self::MONTHS as $monthIndex => $month){
        $valPerMonth[$monthIndex] = 0; // Sets the value to 0 in case there's no value
        foreach($meetings as $meeting){
            // Get aftercares for the meeting
            $aftercares = Aftercare::where('meeting_id', $meeting->id);
            // Checks if the month in the loop matches the month in meetings
            if($monthIndex == $meeting->schedule->format('m')){
                $valPerMonth[$monthIndex] += $meeting->duration;
                foreach($aftercares as $aftercare){
                    $valPerMonth[$monthIndex] += $aftercare->duration;
                }
            }
        }
    }
}
```

65 Partie code : graphiques par mois

La variable « \$year » est un paramètre de la méthode « index(\$year) » car elle définit l'année avec laquelle les entretiens sont récupérées dans la variable « \$meetings ».

Le fonctionnement de cette partie consiste à traverser une boucle « foreach » douze fois en se basant sur la constante « MONTHS » qui contient les mois en format numérique et texte. Puis, la valeur pour chaque mois est initiée à zéro afin le tableau ne soit pas vide. Une autre boucle « foreach » insérera dans la variable « \$valPerMonth » avec son index étant le mois, la somme pour les entretiens ainsi que les suivis.

Il devrait y avoir une méthode plus optimisée afin de faire tout cela, cependant, aucune recherche approfondie n'a été faite.

3.13.3 Récupération des valeurs pour les entretiens et ses suivis par catégorie

Les graphiques par catégorie sont plus complexes car elles affichent deux valeurs sur l'axe X. La première étant le nombre d'entretiens et suivis et la deuxième étant leur somme d'heures passées.

```
// Get values of meetings per CATEGORY
$totalTimeCat = 0; // Total of time spent
$totalNbrMeetingCat = 0; // Total of meetings
foreach($types as $typeIndex => $type){
    $valPerCat[$typeIndex] = 0; // Value per category set to 0
    $nbrPerCat[$typeIndex] = 0; // Number of meetings per category
    $typesNames[$typeIndex] = $type->name; // Set the name of the type
    foreach ($meetings as $meeting){
        $aftercares = Aftercare::where('meeting_id', $meeting->id);
        // If the type matches with the meeting's type, the values are added
        if($type->id == $meeting->type_id){
            // Set the value in the array
            $valPerCat[$typeIndex] += $meeting->duration;
            $nbrPerCat[$typeIndex] += 1;
            $totalTimeCat += $meeting->duration;
            $totalNbrMeetingCat += 1;
            foreach($aftercares as $aftercare){
                $totalTimeCat += $aftercare->duration;
                $totalNbrMeetingCat += 1;
            }
        }
    }
}
```

66 Partie Code : graphiques par catégorie

Cette partie du code contient plusieurs variables :

- \$totalTimeCat : Somme total de temps passés
- \$totalNbrMeetingCat : Nombre total d'entretiens
- \$valPerCat : Somme de temps passés en minutes par catégorie
- \$nbrPerCat : Nombre d'entretiens par catégorie
- \$aftercares = Les suivis.

3.13.4 Création des graphiques

```
// Create the first chart
$chart1PerMonth = new ChartsHC;
$chart1PerMonth->labels(array_values(self::MONTHS));
$chart1PerMonth
    ->dataset
    (
        'Somme du temps passé par mois (heures)',
        'column',
        array_values($this->formatToHoursMinutes($valPerMonth))
    );

// Create the second chart
$chart1PerCategory = new ChartsHC;
$chart1PerCategory->labels(array_values($typesNames));
// First dataset
$chart1PerCategory
    ->dataset
    (
        'Nombre de d\'entretiens passés par catégorie',
        'column',
        array_values($nbrPerCat)
    )->color('rgb(255, 0, 0)');

// Second dataset
$chart1PerCategory
    ->dataset
    (
        'Somme du temps passé par catégorie (heures)',
        'line',
        array_values($this->formatToHoursMinutes($valPerCat))
    )->color('rgba(0, 0, 250, 1)');

```

67 Crédit à la création des graphiques

Le fonctionnement consiste à créer un nouvel objet ChartHC nommé ainsi car c'est un chart utilisant la librairie « HighCharts¹² », ajouter les labels et le nom représentant les données, le type de graphique (varie selon la librairie) et un tableau de valeurs.

La fonction « array_values » permet de récupérer que les valeurs sans les indexs dans un tableau.

Il est également possible de personnaliser la couleur de fond, de l'avant, la bordure et plus encore...

¹² Documentation de HighCharts : <https://www.highcharts.com>

3.13.5 Affichage des graphiques sur la page de graphiques

Après avoir instanciés tous les graphiques et les variables nécessaires, on affiche la vue.

```
// Return the view
    return view('graphics', [
        'types' => $types,
        'years' => $years,
        'months' => self::MONTHS,
        'currentUser' => Auth::user(),
        'currentYear' => intval($year),
        'charts' => $charts,
        'chart1PerMonth' => $chart1PerMonth,
        'chart1PerCategory' => $chart1PerCategory,
        'chart2PerMonth' => $chart2PerMonth,
        'chart2PerCategory' => $chart2PerCategory,
    ]);
]
```

68 Affichage de la vue avec les données

```
@if ($chart1PerMonth != null)
<div class="border border-gray-300 shadow-lg rounded-md p-1 xl:p-8 print hidden md:block">
    {!! $chart1PerMonth->container() !!}
</div>
@endif
@if ($chart1PerCategory != null)
<div class="border border-gray-300 shadow-lg rounded-md p-1 xl:p-8 print hidden md:block">
    {!! $chart1PerCategory->container() !!}
</div>
@endif
```

69 Affichage des graphiques dans « graphics.blade.php »

L'affichage des graphiques se fait en mettant la variable contenant le graphique dans une « <div> » avec une fonction « ->container() » qui permet d'afficher le conteneur.

Avant la balise de fin « </body> », il faut ajouter les scripts dans un balise « <script> » afin d'effectuer la mise en page du graphique par rapport au conteneur. Dans le script on ajoute le lien du CDN (Content Delivery Network) contenant le script JavaScript de la librairie choisie.

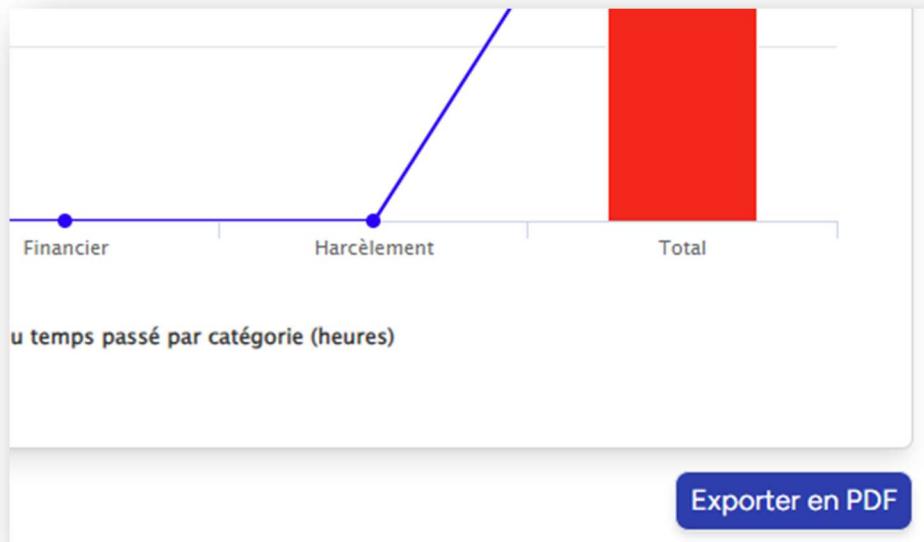
```
<script src="https://cdnjs.cloudflare.com/ajax/libs/highcharts/6.0.6/highcharts.js" charset="utf-8"></script>
@if ($chart1PerMonth != null)
{!! $chart1PerMonth->script() !!}
@endif
@if ($chart1PerCategory != null)
{!! $chart1PerCategory->script() !!}
@endif
@if ($chart2PerMonth != null)
{!! $chart2PerMonth->script() !!}
@endif
@if ($chart2PerCategory != null)
{!! $chart2PerCategory->script() !!}
@endif
```

70 Exécution du script dans « graphics.blade.php »

3.14 Export de la page statistique

L'exportation de la page de statistique est un des points évalués. Cette partie est assez complexe car elle nécessite des connaissances en JavaScript.

Un bouton activant une fonction JavaScript appelée « exportToPDF() » est mise en place sur la page des graphiques.



71 Bouton d'exportation en PDF sur la page « graphics.blade.php »

Afin de pouvoir exporter la page en PDF, la fonction « window.print() » a été choisie en raison de sa simplicité. Cependant, un problème de mise en page était présente.

3.14.1 Redimensionnement de la page

Vu que le format A4 correspond au format mobile, les graphiques pour le format mobile sont affichés, ce qui n'était pas voulu. De plus, vu que la page se redimensionnait, les graphiques quant à eux, ne se redimensionnaient pas. Ayant passé plus de une journée entière à rechercher des solutions afin de résoudre ce problème sans succès, une aide de l'intelligence artificielle « ChatGPT » a été demandée.

ChatGPT a donc, proposé d'utiliser du code JavaScript afin de rapetisser les graphiques, lancer la fonction « window.print() » et ensuite agrandir les graphiques à leurs tailles initiales. (**Image sur la page suivante**).

```

<script defer>
// Function that toggles the printing pop-up
function exportToPDF() {
    resizeCharts();
    // Delay printing to ensure charts are resized
    setTimeout(() => {
        // Initiate printing
        window.print();
        // After printing, resize charts back
    }, 500);
    setTimeout(() => {
        // After printing, resize charts back
        resizeChartsBack();
    }, 500);

}

```

72 Fonction "exportToPDF()" dans le fichier « graphics.blade.php »

La fonction « exportToPDF » est appelée quand le bouton d'exportation est pressé. Cette fonction appelle la fonction « resizeCharts() » qui redimensionne les graphiques, la fonction « window.print() » qui affiche la fenêtre d'impression pour ensuite redimensionner les graphiques à leur taille originelle. La raison des « setTimeout() » qui permettent d'ajouter un délai avant l'exécution de la fonction est car la fenêtre d'impression s'affiche avant que les graphiques aient finit de se redimensionner.

```

// Function that resizes the charts to the desired dimensions
function resizeCharts() {
    // Get references to the charts
    chart1PerMonth = Highcharts.charts[0];
    chart1PerCategory = Highcharts.charts[1];
    // Set the size
    chart1PerMonth.setSize(600, 400);
    chart1PerCategory.setSize(600, 400);
}

```

73 Fonction "resizeCharts()" dans le fichier « graphics.blade.php »

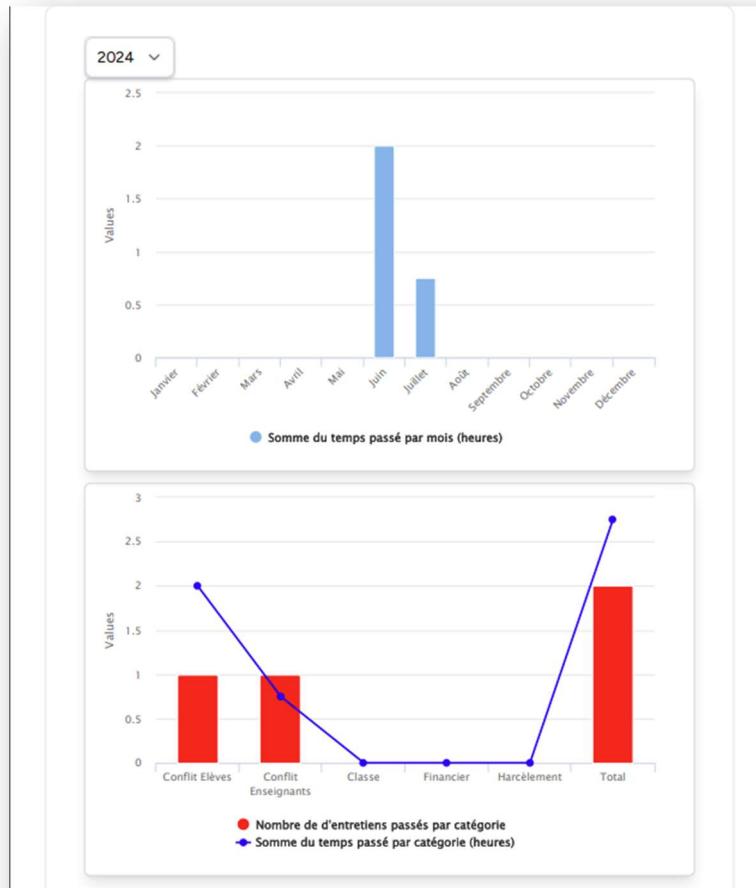
Ces tailles ont été choisies par rapport à la page A4.

```
// Function to resize charts back
function resizeChartsBack() {
    // Resize chart1PerMonth back to original size
    if (chart1PerMonth) {
        chart1PerMonth.setSize(null, null, false);
    }
    // Resize chart1PerCategory back to original size
    if (chart1PerCategory) {
        chart1PerCategory.setSize(null, null, false);
    }
}
```

74 Fonction "resizeChartsBack()" dans le fichier « *graphics.blade.php* »

Les graphiques ayant des tailles définies par rapport à la taille de la balise « <div> » sont mises à null. La valeur « false » représente l'exécution de l'animation ou non. Pour des questions d'optimisations, la valeur est donc, définie à « false ».

Les trois fonctions combinées donnent ce résultat :



75 Fenêtre d'impression

3.14.2 En-tête et pied de page personnalisé

Afin d'afficher ou masquer des éléments au moment de l'exportation en PDF (mode impression). Des styles CSS sont appliqués au moment de l'exécution de la fonction « window.print ».

```
<style>
    /* When the printing is prompted, it changes each class value to be displayed or not. */
    @media print {
        .print {
            display: block;
        }
        .print-hide {
            display: none;
        }
        nav{
            display: none;
        }
    }
</style>
```

76 Style CSS pour la fonction "window.print"

Les éléments ayant la classe « .print » sont affichés au moment de l'impression peu importe s'ils étaient cachés ou bien déjà affichés.

Les éléments ayant la classe « .print-hide », sont quant à eux, cachés.

```
@if ($chart1PerCategory != null)
<div class="border border-gray-300 shadow-lg rounded-md p-1 xl:p-8 print hidden md:block">
    {!! $chart1PerCategory->container() !!}
</div>
@endif
{{-- Displays for mobile --}}
@if ($chart2PerMonth != null)
<div class="border border-gray-300 shadow-lg rounded-md p-1 xl:p-8 print-hide md:hidden">
    {!! $chart2PerMonth->container() !!}
</div>
@endif
```

77 Éléments ayant les classes ".print" et ".print-hide"

La barre de navigation (nav) est cachée afin de laisser place à l'en-tête et le pied de page.

```
<header class="print hidden">
    <div class="flex justify-between text-xs">
        <div>ETML-Médiateur</div>
        <div><?php echo $currentUser->firstname . " " . $currentUser->lastname ?></div>
    </div>
</header>
```

78 En-tête sur la page statistique

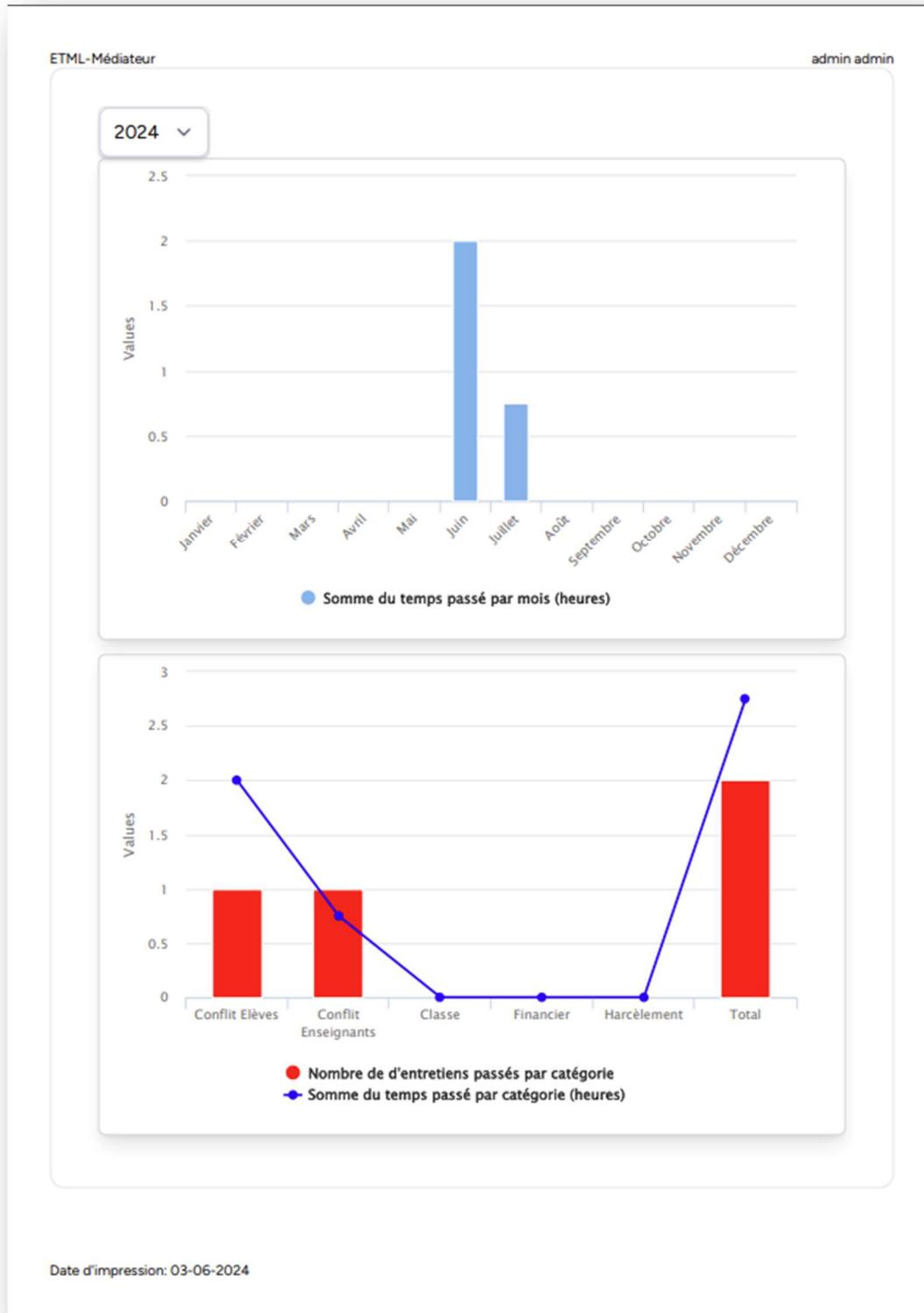
L'en-tête contient un texte « ETML-Médiateur » à gauche et « le prénom et nom » de l'utilisateur actuel.

```
<footer class="print hidden text-xs">
    <?php echo "Date d'impression: " . date('d-m-Y') ?>
</footer>
```

79 Pied de page sur la page statistique

Le pied de page contient la date d'impression en bas à gauche.

Voici donc le résultat de tout ce qui a été mis en place. (**Image sur la page suivante**).



80 PDF de la page statistique

Comme mentionné, ChatGPT a été utilisé afin de pouvoir adapter la mise en page des graphiques.

3.15 Exactitudes des valeurs statistiques

Afin de vérifier que mes valeurs statistiques sont exactes des méthodes spécifiques à chaque point demandé.

Ces méthodes-là sont expliquées en détail au point 3.9.4 « Statistiques ».

De plus, pour encore plus de précisions, des calculs avec les valeurs de la base de données sont effectuées.

3.16 Description des tests effectués

3.16.1 Test de fonctionnement du site

Afin de vérifier si le serveur se lance, se connecte à la base de données et affiche bien la page par défaut de Laravel, la commande « php artisan serve » est lancée.

Vu que le projet contient aussi le package Breeze qui utilise le framework de CSS « Tailwind CSS », il est important de lancer la commande « npm run dev » sur un deuxième terminal.

3.16.2 Test de migrations

Pour s'assurer que les valeurs des attributs sont correctes, il faut tester en faisons les migrations.

3.16.3 Test manuels

N'ayant pas pu voir le fonctionnement de Laravel Dusk, les tests de chaque fonctionnalité de chaque contrôleur ont été effectués.

3.17 Erreurs restantes

3.17.1 Routes /home et /home/{year?}

La route « /home/{year?} » n'ayant pas obligatoirement, une valeur pour « year » devrait fonctionner et rediriger sur la page d'accueil. Cependant, sans la route « /home », une erreur de redirection survenait.

3.17.1.1 Résultat de l'erreur



81 Erreur de redirection

3.17.1.2 Fonctionnement du contrôleur « MeetingControlleur »

```
Checks if the year has been defined, if so, the user gets to the view with the data.
if($year){
    return view('home', [
        'meetingsTotal' => $this->countUserAllMeetings($year), // Get the total of meetings the user has
        'currentUser' => Auth::user(),
        'userMeetings' => $this->getUserAllMeetings($year),
        'timeSpent' => $this->countUserTimeSpent($year),
        'avgTimeSpent' => $this->getAvgTimeSpent($year),
        'upcomingMeetings' => $this->countUpcomingMeetings($year),
        'types' => $types,
        'years' => $years,
        'months' => self::MONTHS,
        'currentYear' => $year,
    ]);
}

// If $year is not assigned, the earliest year from the DB will be assigned to $year and then redirected to the page
elseif(Meeting::where('user_id', Auth::id())->count() > 0){
    $year = Meeting::selectRaw('extract(year FROM schedule) AS year')
        ->distinct()
        ->orderBy('year', 'asc')
        ->first()->year;

    return redirect()->route('meeting.index', $year);
}

// Displaying if there are no meetings in the database
else{
    return view('home', [
        'types' => $types,
        'years' => $years,
        'months' => self::MONTHS,
        'meetingsTotal' => 0,
        'userMeetings' => 0,
        'upcomingMeetings' => 0,
        'avgTimeSpent' => 0,
        'timeSpent' => 0,
        'currentYear' => null,
        'currentUser' => Auth::user(),
    ]);
}
```

82 Fonctionnement de la méthode "index()"

Le fonctionnement de la méthode « index(\$year = null) » est que si l'année est définie, elle affiche la vue avec toutes les valeurs récupérées des méthodes ayant besoin de la variable « \$year ». Cependant, si l'année n'est pas insérée mais que l'utilisateur a des entretiens, la variable « \$year » est définie dans la partie « elseif » pour ensuite rediriger vers la même méthode mais, avec la variable définie. Dans le cas où, l'utilisateur n'a aucun entretien, une page avec des données vides est affichée.

Alors afin de résoudre ce problème, une route supplémentaire, sans paramètres, est ajoutée.

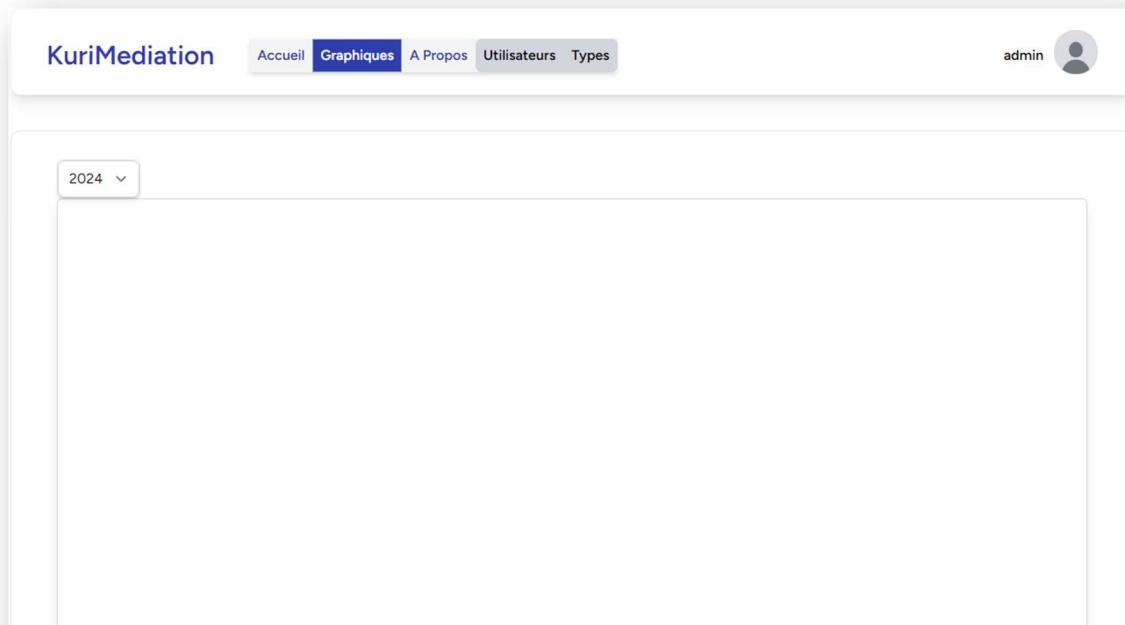
3.17.2 Barre de navigation – Lien vers la page des graphiques

Le fonctionnement de l'affichage de la page des graphiques est similaire à celle de la page d'accueil. Cependant, le problème vient du chargement des graphiques et non des routes.

3.17.2.1 Résultat de l'erreur

```
✖ ▶ Uncaught Error: Highcharts error #16: www.highcharts. VM85 highcharts.js:10  
com/errors/16  
    at a.error (VM85 highcharts.js:10:178)  
    at highcharts.js:9:38  
    at highcharts.js:10:12  
    at highcharts.js:8:103  
    at highcharts.js:8:109
```

83 Erreurs de chargement des graphiques



84 Affichage des graphiques avec l'erreur

Highcharts already defined in the page

This error happens if the `Highcharts` namespace already exists when loading Highcharts or Highstock.

This is caused by including Highcharts or Highstock more than once.

Keep in mind that the `Highcharts.Chart` constructor and all features of Highcharts are included in Highstock, so if you include both, this error will occur.

85 Code erreur : 16

On peut constater un message d'erreur, les graphiques en chargement, et le type d'erreur informé par HighCharts.

3.17.2.2 Fonctionnement de l'affichage des graphiques

Le principe est le même qu'avec l'index du contrôleur « MeetingController », c'est-à-dire que si la variable « \$year » n'est pas instanciée, elle récupère l'année de la table « meetings » la plus récente, et elle retourne dans la méthode « index(\$year = null) » avec la variable instanciée. Le problème ne devrait pas survenir car les graphiques sont instanciés une seule fois dans le « if(\$year) ».

Ayant parlé avec le chef de projet, Monsieur Lymberis, aucune solution n'a été trouvée.

3.18 Liste des documents fournis

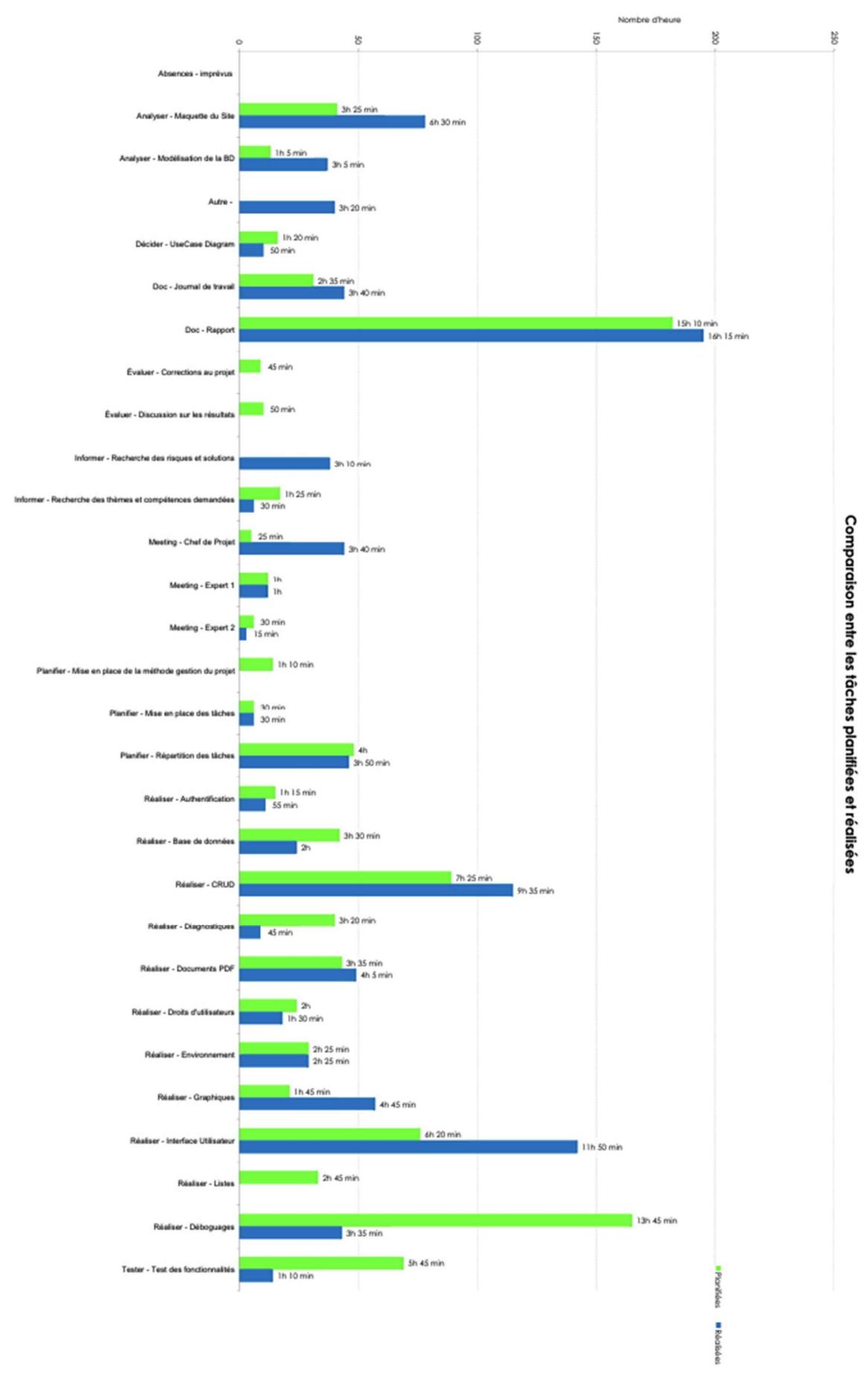
Lister les documents fournis au client avec votre produit, en indiquant les numéros de versions

- *Le rapport du projet – Version 7 - Finale*
- *Le code source sur le dépôt Git (Version 7 - Finale) :*
- *Le journal de travail : Version 7 - Finale*

4 Conclusions

4.1 Bilan de la planification

Au niveau de l'avancement du projet par rapport à la planification, des retards au niveau de l'avancement de la rédaction du rapport à causes de problèmes avec l'affichage des composants « Livewire », de problèmes avec la mise en page des graphiques lors de l'exportation en PDF, de problèmes d'affichage de la page des graphies et de problèmes avec la barre de navigation, ont été rencontrées.



Comparaison entre les tâches planifiées et réalisées

4.2 Bilan des fonctionnalités

Au niveau des fonctionnalités, toutes les fonctionnalités demandées fonctionnent. Cependant, des fonctionnalités supplémentaires telles que la barre de recherches, le tri de la liste d'entretiens et la modification de la photo de profil n'ont pas abouti.

4.3 Bilan personnel

En raison du retard au niveau de la rédaction du rapport, un sentiment de stress était présent constamment. De plus, pendant la rédaction du rapport, des erreurs et des améliorations possibles apparaissent, donnant l'envie de les résoudre.

5 Glossaire

Breeze

Composant supplémentaire de Laravel qui offre une implémentation très simpliste et minimaliste de fonctionnalités d'authentification, 7

Eloquent

Programme se plaçant entre une application web et une base de donnée afin de permettre aux développeurs de travailler avec les données sous formes d'objet, 7

Laravel

Framework PHP, 6

Livewire

Framework full-stack pour Laravel facilitant la création d'interfaces

utilisateur dynamiques sans avoir nécessairement besoin de connaissances en JAvascript, 7

MVC

Architechture (Modèle - Vue - Contrôleur) utilisé en Laravel, 6

Tailwind CSS

Framework CSS qui permet de rendre plus facile la stylisation de vue en évitant de passer par un fichier css., 7

UwAmp

Ensemble de logiciels dont le but est de créer un environnement de développement web local sur Windows, 8

6 Annexes

7 Sources

Afficher message de confirmation à la suppression. (s.d.). Récupéré sur <https://stackoverflow.com/questions/9139075/how-to-show-a-confirm-message-before-delete>

Conversion minutes en heures et minutes. (s.d.). Récupéré sur <https://write.corbpie.com/converting-minutes-to-hours-and-minutes-with-php/>

FontAwesome. (s.d.). Récupéré sur <https://fontawesome.com/v4/get-started/>

Importation de document PDF en Laravel 11. (s.d.). Récupéré sur <https://www.itsolutionstuff.com/post/laravel-11-file-upload-example-tutorialexample.html>

Laravel - Public Disk. (s.d.). Récupéré sur <https://laravel.com/docs/11.x/filesystem#the-public-disk>

Laravel - Value in Textarea. (s.d.). Récupéré sur <https://stackoverflow.com/questions/40340892/laravel-value-in-textarea>

Laravel Debugbar. (s.d.). Récupéré sur <https://github.com/barryvdh/laravel-debugbar>

Pages to PDF. (s.d.). Récupéré sur <https://stackoverflow.com/questions/43140467/page-with-multiple-chart-js-charts-to-pdf>

PHP Form Validation - Errors. (s.d.). Récupéré sur <https://stackoverflow.com/questions/51124049/php-form-validation-and-showing-error-message-beside-input-fields>

RFC 3696. (s.d.). Récupéré sur <https://datatracker.ietf.org/doc/html/rfc3696>

Stackoverflow - Remove no file chosen text. (s.d.). Récupéré sur <https://stackoverflow.com/questions/12035400/how-can-i-remove-the-no-file-chosen-tooltip-from-a-file-input-in-chrome>