

# Application web de fitness

---

RAPPORT DE TPI

## InfoFit

Thierry Koetschet

CHEMIN DU PERREY 22 | 1670 URSY

THIERRY.KOETSCHET@CPNV.CH

SI-CA2A

## Table des matières

1	Analyse préliminaire .....	3
1.1	Introduction .....	3
1.2	Organisation .....	4
1.3	Méthode de gestion de projet .....	5
1.4	Objectifs.....	5
1.5	Planification initiale .....	6
1.6	Structure du dossier.....	7
2	Analyse / Conception.....	8
2.1	Concept .....	8
2.1.1	Modèle conceptuel de données .....	8
2.1.2	Modèle logique de données.....	9
2.1.3	Base de données.....	10
2.1.4	Moodboard .....	11
2.1.5	Maquettes.....	12
2.2	Stratégie de test.....	16
2.3	Risques techniques .....	16
2.4	Planification .....	17
2.5	Dossier de conception .....	18
3	Réalisation.....	19
3.1	Dossier de réalisation .....	19
3.2	Code source .....	20
3.2.1	Fonction « store » de UserController.php .....	21
3.2.2	Fonction « searchFoodstuff » de FoodController.php.....	22
3.2.3	Fonction « lineChart » de ChartController.php .....	24
3.3	Aperçu de l'application.....	25
3.3.1	Page d'accueil .....	25
3.3.2	Login.....	25
3.3.3	Register .....	26
3.3.4	Profil .....	27
3.3.5	IMC .....	27
3.3.6	Alimentation .....	28
3.3.7	Ajouter un aliment.....	28
3.4	Hébergement sur le serveur web.....	29
3.5	Description des tests effectués .....	29
3.6	Erreurs restantes .....	32
3.7	Liste des documents fournis .....	32
4	Conclusions.....	33
5	Annexes.....	34
5.1	Cahier des charges.....	34
5.2	Horaire de travail .....	37
5.3	Résumé du travail.....	38
5.4	Sources – Bibliographie.....	39
5.5	Journal de travail .....	40
5.6	Glossaire .....	44
5.7	Table des illustrations .....	45
5.8	Manuel d'installation .....	46

# 1 Analyse préliminaire

## 1.1 Introduction

Ce rapport va décrire en détail la réalisation de mon projet de TPI sur une application web de fitness. Cette application permettra aux utilisateurs de s'authentifier grâce un compte et d'accéder à ses fonctionnalités. La première fonctionnalité est le calcul de l'indice de masse corporelle de l'utilisateur grâce à son poids et à sa taille. La deuxième fonctionnalité est un calendrier permettant d'enregistrer les aliments consommés quotidiennement par l'utilisateur afin de calculer le total des calories et des macronutriments journaliers. Toutes ces informations sur les différents aliments absorbés seront accessibles grâce à une API publique. L'application doit être développée avec l'aide d'un framework PHP et avoir une structure de type MVC.

La raison de mon choix de partir sur un tel projet s'explique par le fait que le développement est mon domaine de prédilection en informatique, et spécifiquement le développement web, dans lequel j'aimerais idéalement poursuivre ma carrière professionnelle. De plus, lors des débuts de ma formation d'informaticien à Sainte-Croix, j'ai également développé un attrait particulier pour le sport et la musculation. C'est ainsi que m'est venu l'idée de combiner ces deux domaines pour réaliser un projet intéressant et qui pourrait également m'être utile dans mon parcours sportif.

## 1.2 Organisation

Projet	Date début	Participants		Activités	Tâches
		Nom	Prénom		
Application web de fitness	02.05.2023	Koetschet	Thierry	1 Analyse	11 Analyse du framework Laravel 12 Installation du framework Laravel 13 Analyse de l'API Open Food Facts 14 Moodboard de l'application 15 Maquette de l'application 16 - 17 - 18 - 19 -
				2 Implémentation	21 Conception du MCD 22 Conception du MLD 23 Création de la DB 24 Création du projet 25 Création du gabarit 26 Page d'accueil / Connexion à l'application 27 Page IMC 28 Page données alimentaires journalières 29 Login / Register 30 Connexion à la DB / API
				3 Tests	31 Rédaction de la stratégie de test 32 Rédaction des tests 33 Application des tests 34 - 35 - 36 - 37 - 38 - 39 - 40 -
				4 Documentation	41 Réalisation de la planification initiale 42 Rédaction de la documentation 43 Remplissage du journal de travail 44 - 45 - 46 - 47 - 48 - 49 -
				5 Autres	51 Absence 52 Autres

Image 1 Organisation des tâches

Dans mon organisation personnelle pour la réalisation de mon projet de TPI, j'ai décidé de séparer toutes mes tâches en quatre sprints. Le premier sprint regroupe toutes les tâches concernant la partie analytique de mon travail. Les tâches contenues dans le second sprint concernent la partie pratique du travail d'implémentation de l'application. Le troisième sprint représente la partie tests du travail et enfin le dernier sprint est un peu plus particulier car il contient le travail de rédaction de la documentation et de remplissage du journal de travail qui doit être fait du début à la fin du projet.

### 1.3 Méthode de gestion de projet

J'ai choisi la méthode en cascade pour la gestion de mon projet de TPI. Je trouve que pour un projet en solo, cette méthode répond parfaitement à mes besoins en terme de gestion de projet.

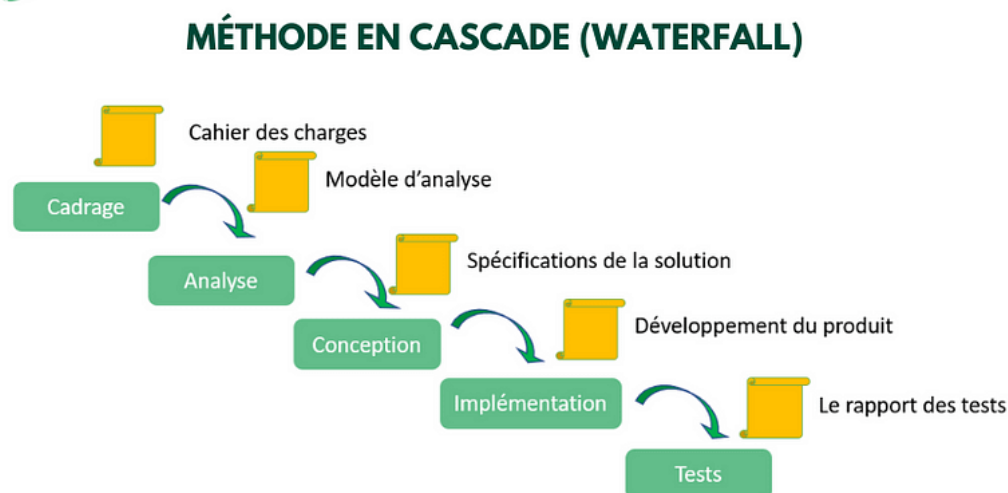


Image 2 Méthode de gestion de projet en cascade

Les cinq étapes de la méthode en cascade correspondent parfaitement à la vision que j'ai pour la réalisation d'un tel projet. En effet, le cadrage en début de projet est primordial afin de structurer le travail et de ne pas trop s'éparpiller. La planification initiale représente parfaitement cette première étape car elle permet de poser une valeur de temps sur les différentes à réaliser dans la suite du projet. La partie analyse correspond au travail préliminaire à réaliser avant de se lancer dans la conception comme l'analyse du framework à utiliser ou encore l'API recommandée dans le cahier des charges. Après la partie analyse, j'ai réalisé la conception de mon application grâce à un moodboard, une maquette, un modèle conceptuel de données et un modèle logique de données. L'étape suivante est d'implémenter le site web, c'est-à-dire d'écrire tout le code nécessaire à son bon fonctionnement. Enfin, les tests permettent de vérifier le fonctionnement général de l'application.

### 1.4 Objectifs

Les objectifs fixés dans le cahier des charges par le chef de projet et les experts sont de réaliser un site web avec l'utilisation d'un framework PHP. L'application web doit également avoir plusieurs fonctionnalités telles qu'un authentificateur sécurisé, un calculateur d'indice de masse corporelle montrant l'évolution du poids des utilisateurs grâce à un graphique et un calendrier permettant de contrôler son alimentation utilisant une API publique.

Personnellement l'objectif principal que j'aimerais atteindre à la fin de ce travail est la maîtrise d'un framework PHP car je pense que cela pourrait peser dans la balance dans la recherche d'un futur emploi dans le développement web.

## 1.5 Planification initiale

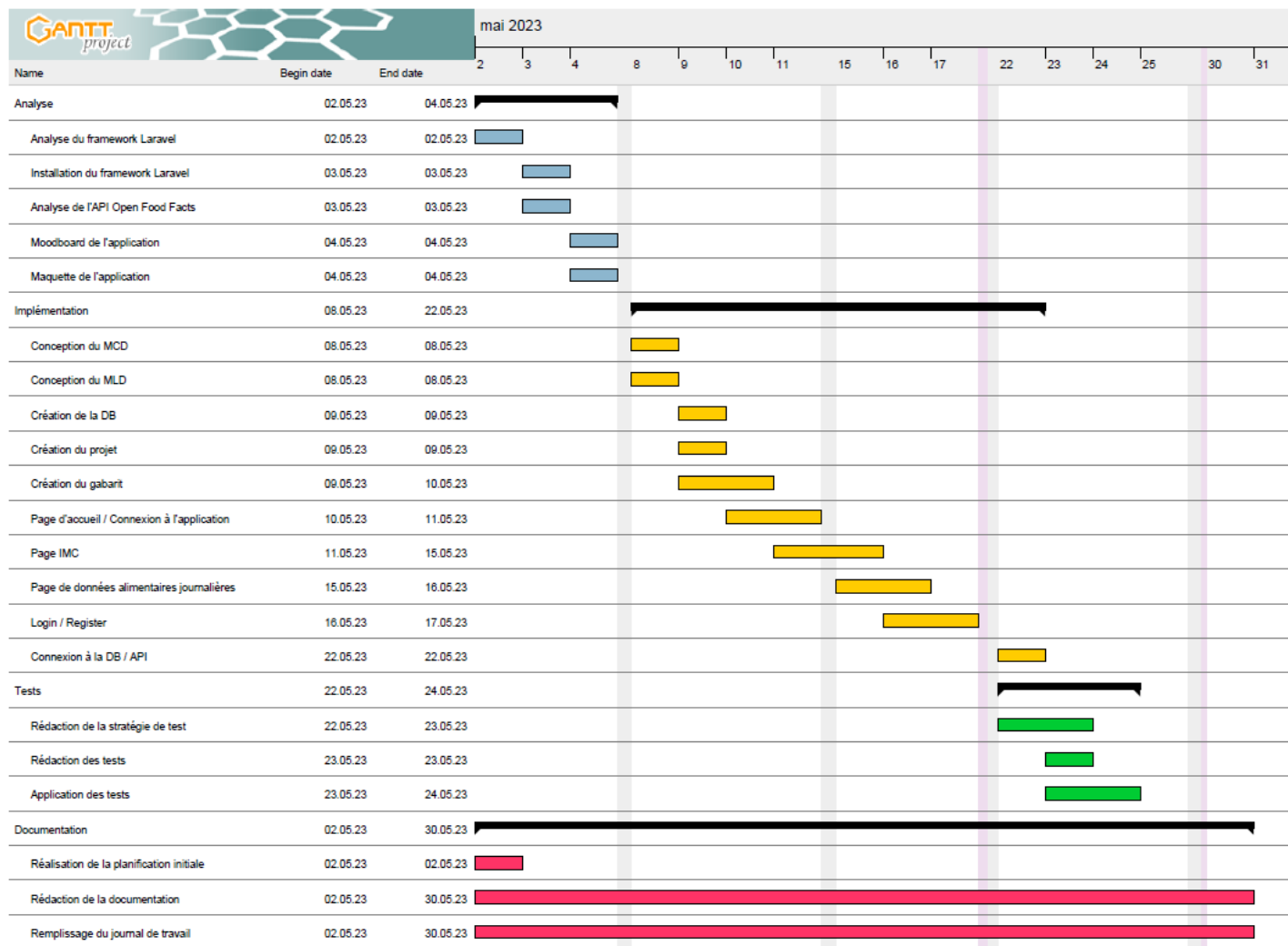


Image 3 Planification initiale du projet de TPI

## 1.6 Structure du dossier

Ce dossier se divise en quatre parties principales.

La première partie se compose de l'analyse préliminaire de ce travail, c'est-à-dire d'une introduction, d'une explication de l'organisation du projet, une description des objectifs visés et puis un bref aperçu de ma planification initiale.

La deuxième englobe toute la partie analyse et conception du projet avec l'élaboration du concept, de la stratégie de test, un compte rendu des risques techniques, une révision de la planification initiale du projet et enfin la partie conception du projet.

La troisième partie représente toute la réalisation pratique du projet commençant par lister tous les fichiers du dossier de réalisation, puis une description des tests effectués et des erreurs restantes et finalement une énumération des documents fournis à la remise du projet.

La dernière décrit les conclusions auxquelles je suis arrivé à la fin de ce projet telles que les objectifs atteints ou non, mon ressenti au fil du projet, les difficultés rencontrées et les améliorations que je pourrais apporter si je devais refaire un tel projet.

## 2 Analyse / Conception

### 2.1 Concept

#### 2.1.1 Modèle conceptuel de données

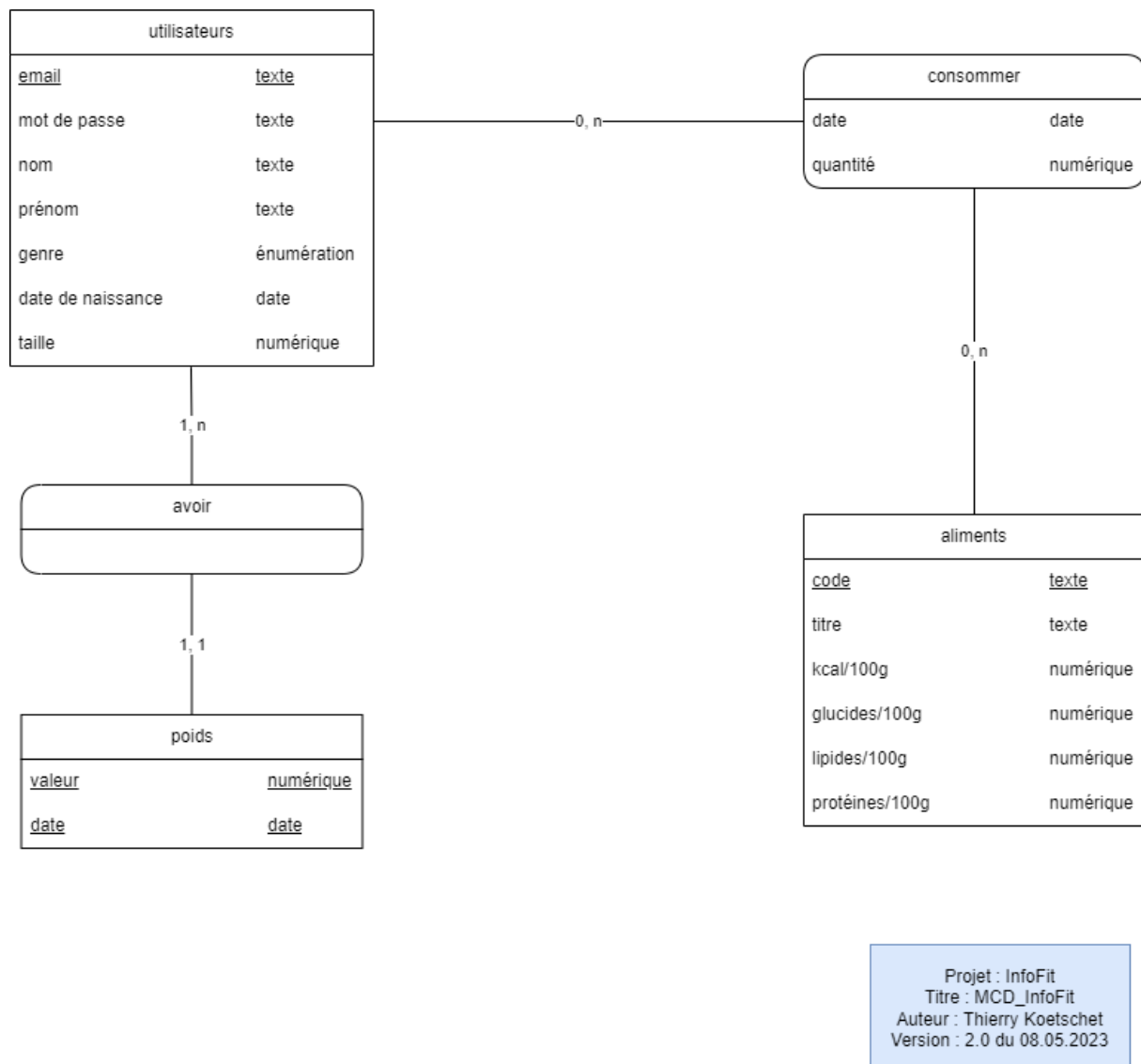


Image 4 MCD de la base de données infofit

Le MCD a été créé avec draw.io. Il s'agit d'un logiciel libre d'édition graphique de diagrammes.

Au lieu d'avoir un simple champ poids dans l'entité « utilisateurs », j'ai ajouté une entité supplémentaire « poids » avec comme attributs « valeur » et « date » afin de permettre aux utilisateurs de mon application de modifier leur poids au fur et à mesure de leur évolution et d'en avoir un historique.



## 2.1.2 Modèle logique de données

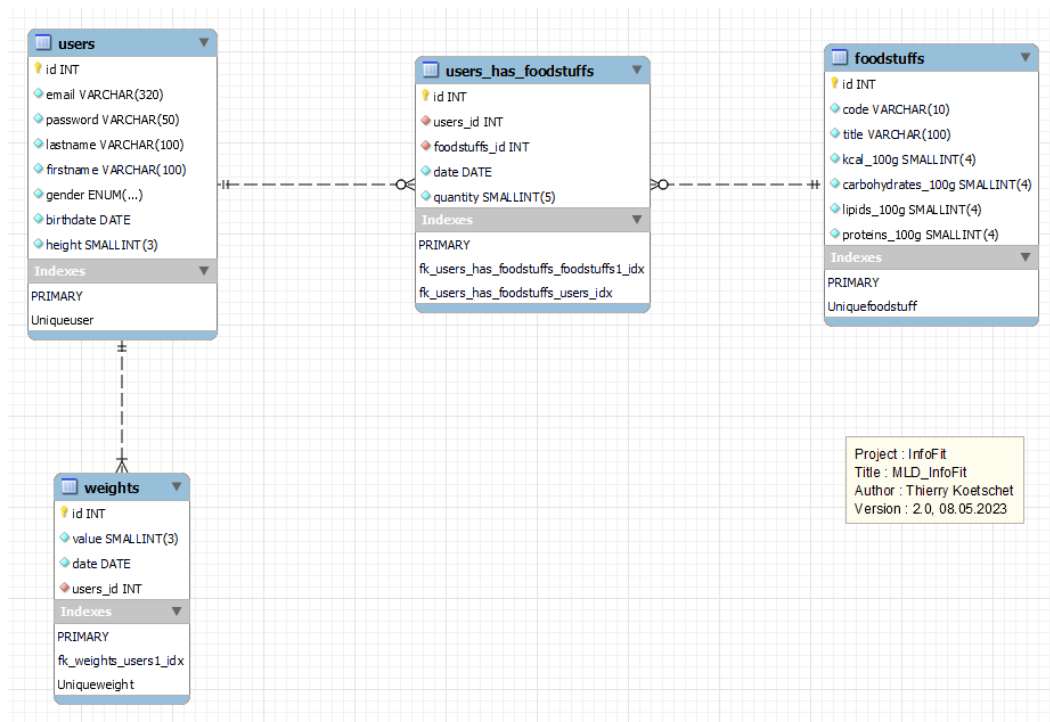


Image 5 MLD de la base de données infofit

Le MLD a été réalisé à l'aide de MySQL Workbench afin de pouvoir générer automatiquement le script de création de la base de données.

### 2.1.3 Base de données

```
-- Listage de la structure de la base pour infofit
CREATE DATABASE IF NOT EXISTS `infofit` /*!40100 DEFAULT CHARACTER SET utf8mb3 */;
USE `infofit`;

-- Listage de la structure de table infofit. foodstuffs
CREATE TABLE IF NOT EXISTS `foodstuffs` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `code` varchar(15) NOT NULL,
  `title` varchar(100) NOT NULL,
  `kcal_100g` float unsigned NOT NULL DEFAULT 0,
  `carbohydrates_100g` float unsigned NOT NULL DEFAULT 0,
  `lipids_100g` float unsigned NOT NULL DEFAULT 0,
  `proteins_100g` float unsigned NOT NULL DEFAULT 0,
  `updated_at` timestamp NULL DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `Uniquefoodstuff` (`code`)
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=utf8mb3;

-- Listage de la structure de table infofit. users
CREATE TABLE IF NOT EXISTS `users` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `email` varchar(320) NOT NULL,
  `password` varchar(64) NOT NULL,
  `lastname` varchar(100) NOT NULL,
  `firstname` varchar(100) NOT NULL,
  `gender` varchar(6) NOT NULL,
  `birthdate` date NOT NULL,
  `height` smallint(3) unsigned NOT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `Uniqueuser` (`email`)
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=utf8mb3;

-- Listage de la structure de table infofit. users_has_foodstuffs
CREATE TABLE IF NOT EXISTS `users_has_foodstuffs` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `users_id` int(10) unsigned NOT NULL,
  `foodstuffs_id` int(10) unsigned NOT NULL,
  `date` date NOT NULL,
  `quantity` float NOT NULL DEFAULT 1,
  `period` varchar(9) NOT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `fk_users_has_foodstuffs_foodstuffs1_idx` (`foodstuffs_id`),
  KEY `fk_users_has_foodstuffs_users_idx` (`users_id`),
  CONSTRAINT `fk_users_has_foodstuffs_foodstuffs1` FOREIGN KEY (`foodstuffs_id`) REFERENCES `foodstuffs` (`id`) ON DELETE CASCADE,
  CONSTRAINT `fk_users_has_foodstuffs_users` FOREIGN KEY (`users_id`) REFERENCES `users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=utf8mb3;

-- Listage de la structure de table infofit. weights
CREATE TABLE IF NOT EXISTS `weights` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `value` smallint(3) unsigned NOT NULL,
  `date` date NOT NULL,
  `users_id` int(10) unsigned NOT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `Uniqueweight` (`value`,`date`,`users_id`),
  KEY `fk_weights_users1_idx` (`users_id`),
  CONSTRAINT `fk_weights_users1` FOREIGN KEY (`users_id`) REFERENCES `users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=16 DEFAULT CHARSET=utf8mb3;
```

Image 6 Scripte SQL de création de la base de données infofit

Voici ci-dessus le scripte de création de la base de données « infofit » également accessible dans mon dépôt GitHub personnel ([https://github.com/ThierryKoetschet/TPI\\_Thierry\\_Koetschet](https://github.com/ThierryKoetschet/TPI_Thierry_Koetschet)). La base de données contient quelques différences par rapport au MCD et au MLD.

Par exemple, on peut remarquer que j'ai dû ajouter à chaque table un attribut « updated\_at » et un attribut « created\_at » au format « timestamp » afin que je puisse insérer des données dans la DB grâce à mon application car Laravel l'exigeait. J'ai également modifié le type des attributs « kcal\_100g », « carbohydrates\_100g », « lipids\_100g » et « proteins\_100g » de la table « foodstuffs » en « float » car j'ai remarqué que les données retournées par l'API Open Food Facts étaient parfois des nombres à virgules et posaient des problèmes pour l'insertion des données dans la base. Enfin, le dernier changement par rapport à la conception de la base de données est l'attribut « period » de la table « users\_has\_foodstuffs » qui permet entre autre de trier les aliments enregistrer par l'utilisateur en fonction de la période dans laquelle il les a consommés.

### 2.1.4 Moodboard



Image 7 Moodboard de l'application

Le moodboard a été réalisé à l'aide d'Adobe Photoshop. Il a principalement servi de croquis pour créer la palette de couleur et les logos utilisé dans le site web.

## 2.1.5 Maquettes

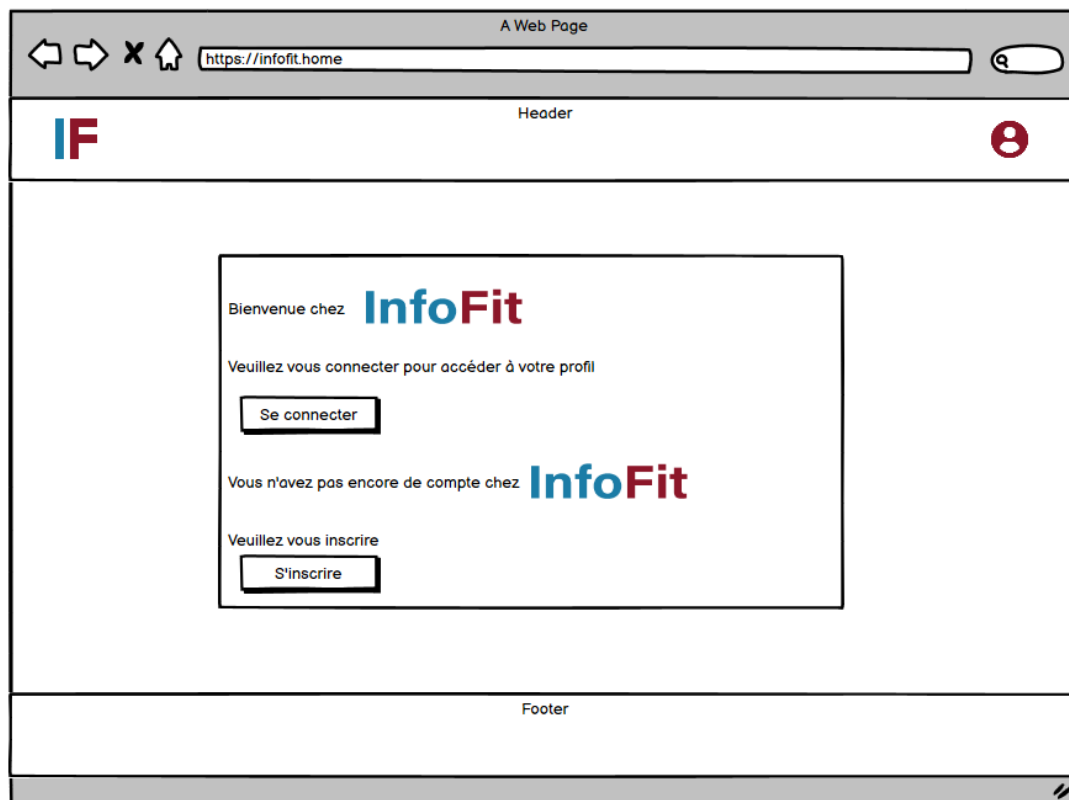


Image 8 Page d'accueil de la maquette

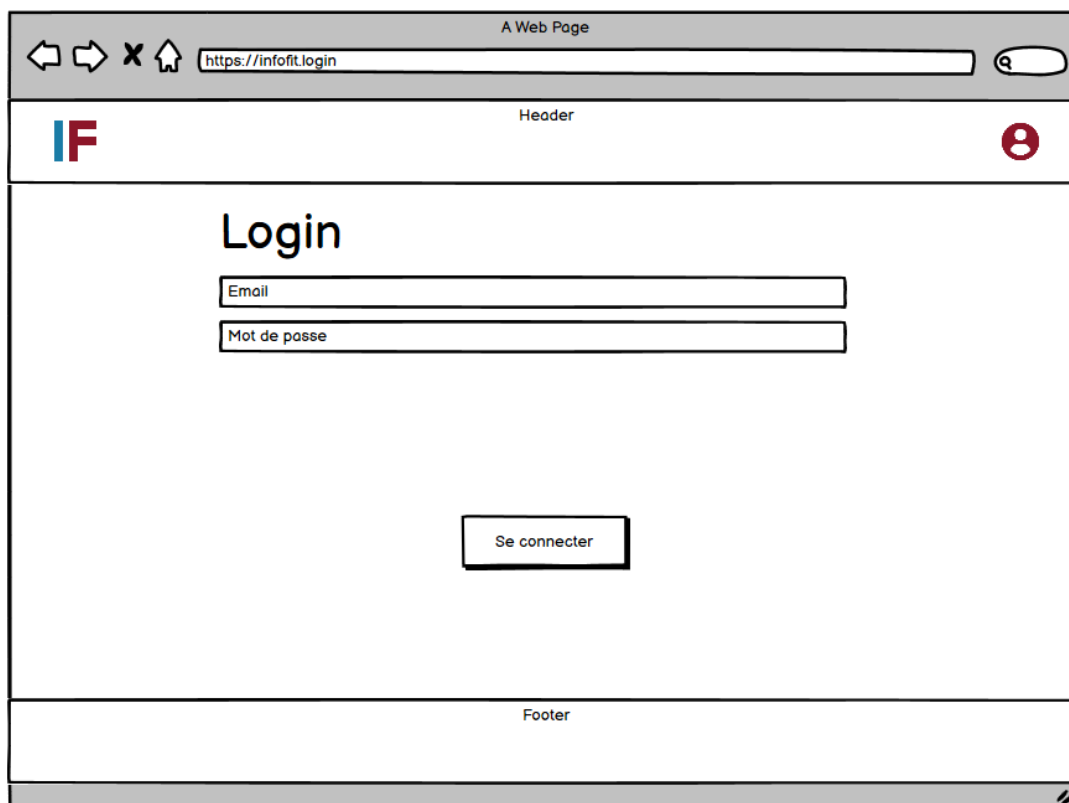
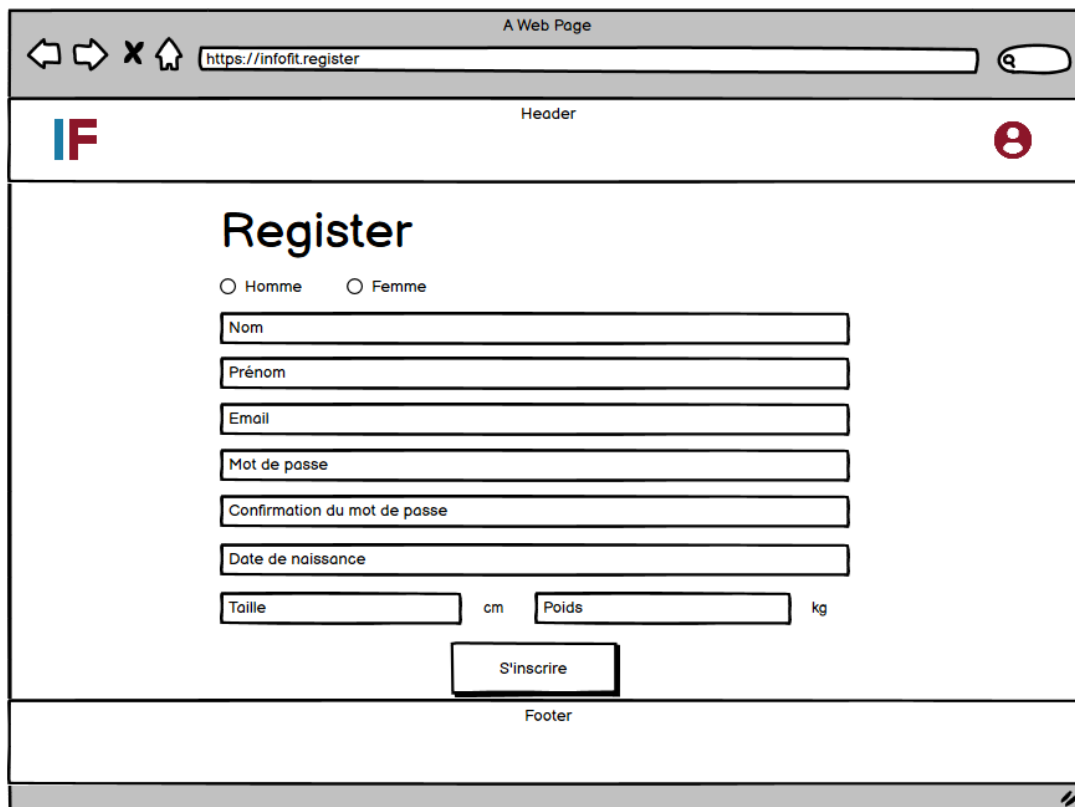


Image 9 Page de login de la maquette



A Web Page

https://infofit.register

Header

IF

Register

☐ Homme ☐ Femme

Nom

Prénom

Email

Mot de passe

Confirmation du mot de passe

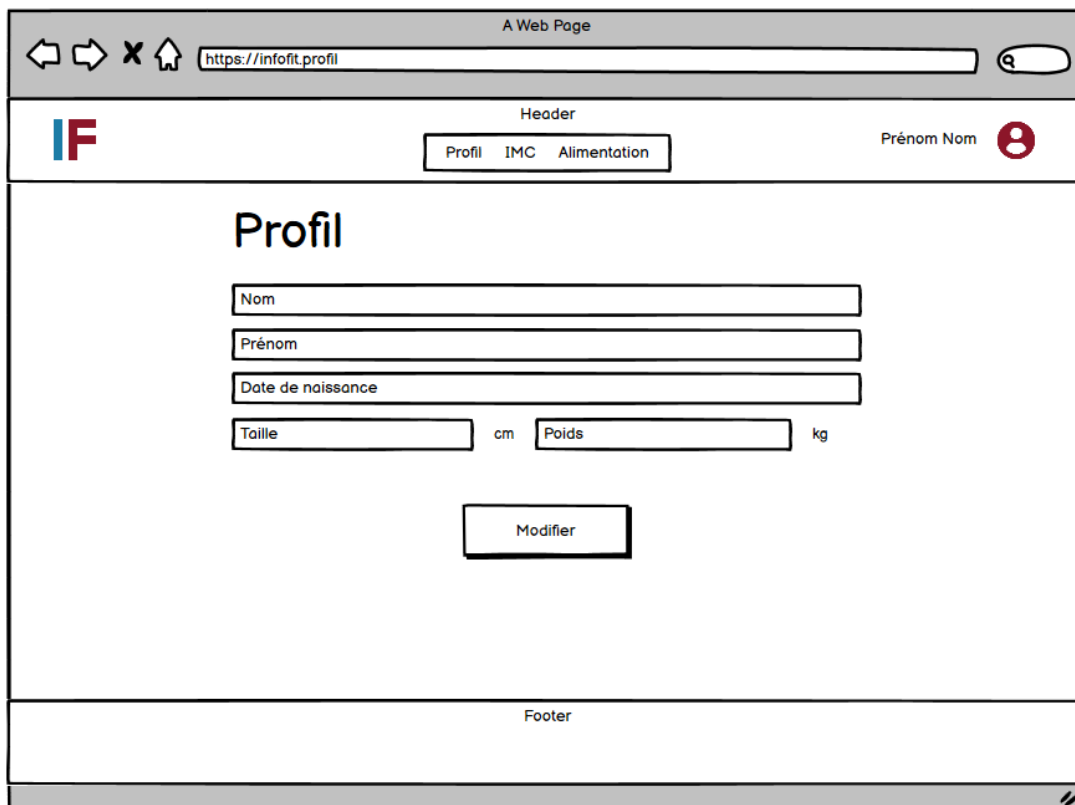
Date de naissance

Taille cm Poids kg

S'inscrire

Footer

Image 10 Page de register de la maquette



A Web Page

https://infofit.profil

Header

IF

Profil IMC Alimentation

Prénom Nom

Profil

Nom

Prénom

Date de naissance

Taille cm Poids kg

Modifier

Footer

Image 11 Page profil utilisateur de la maquette

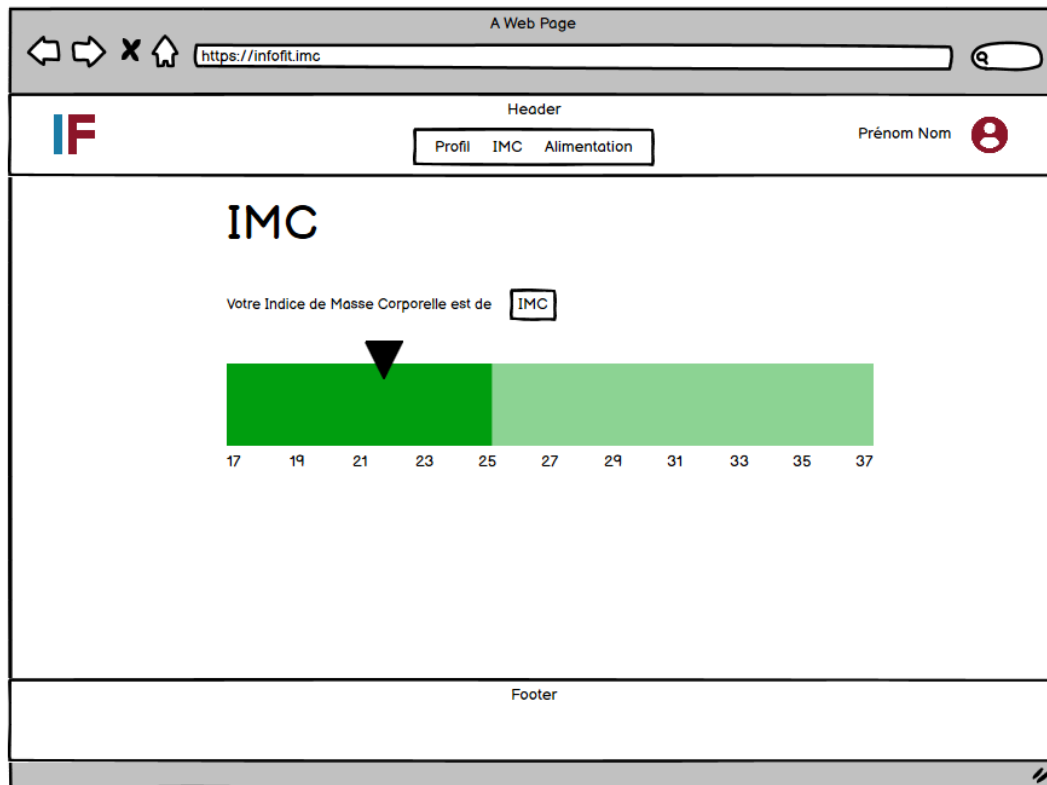


Image 12 Page de calcul de l'IMC de la maquette

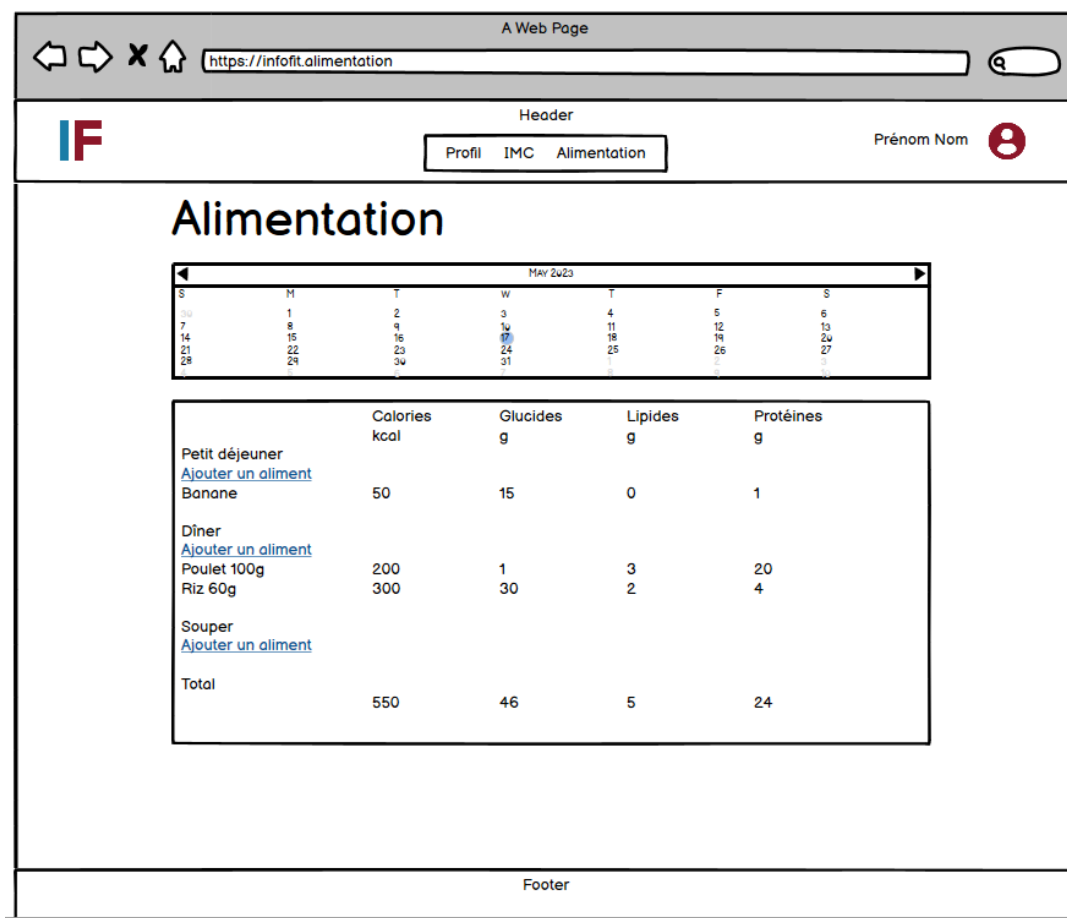


Image 13 Page alimentation de la maquette

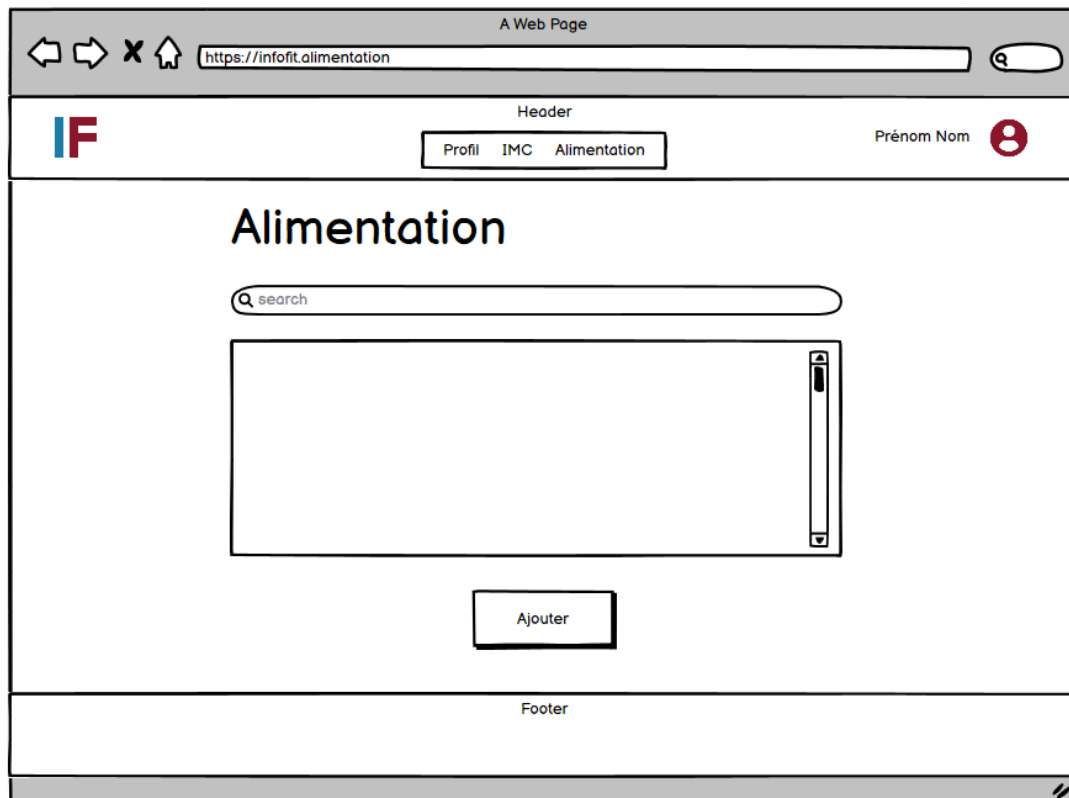


Image 14 Page d'ajout d'aliment de la maquette

Ces maquettes m'ont servi de base tout au long de l'implémentation de mon application, cependant dans la réalisation de certaines fonctionnalités, je me suis rendu compte que je ne pouvais pas copier la maquette complètement. Par exemple, la page sur l'indice de masse corporelle ne ressemble pas à la maquette car j'avais tout d'abord oublié d'y ajouter le graphique d'évolution du poids, puis j'ai également réalisé que mon envie d'implémenter un slider, qui n'était pas mentionner dans mon cahier des charges, allait me demander trop de temps.

## 2.2 Stratégie de test

Dans le cadre de ce projet, je pense qu'une stratégie relativement simple est un choix judicieux. En effet, l'application est composée de trois principales fonctionnalités : un moyen d'authentification en tant qu'utilisateur et de saisie de données personnelles, un calculateur d'indice de masse corporelle et un tableau permettant d'enregistrer la consommation alimentaire journalière de l'utilisateur. Afin de vérifier le fonctionnement de l'application et de la base de données, il suffit d'effectuer des tests fonctionnels sur ces trois fonctionnalités en commençant par l'authentification car les deux autres fonctionnalités ont besoin de certaines informations enregistrées par l'utilisateur.

Je vais également tester les routes de mon application, c'est-à-dire que tous les liens fonctionnent correctement et que toutes les pages du site sont accessibles.

J'ai créé un utilisateur dont je me suis servi au cours de la réalisation de mon application contenant déjà une certaine quantité d'informations. Celui-ci sera utilisé au dans la réalisation de mes tests. Voici ces identifiants :

- Email : [thierry.koetschet@cpnv.ch](mailto:thierry.koetschet@cpnv.ch)
- Mot de passe : Pa\$\$w0rd

## 2.3 Risques techniques

Le risque technique principal est l'apprentissage d'un nouveau framework. En effet, j'ai choisi de réaliser mon site web avec le framework PHP Laravel avec lequel je n'avais encore jamais travaillé. L'apprentissage de l'utilisation d'un tel outil est assez fastidieux et demande beaucoup de temps. Sachant que le temps est une denrée précieuse dans le TPI, il va s'en dire que je prends un paris osé en partant sur cette voie. L'une des raisons qui m'a orienté sur cette décision est que l'un de mes camarades de classe, Pablo Zubieta, avait déjà réalisé son projet de pré-TPI avec Laravel et me l'a vivement recommandé. Il m'a également fait une présentation de l'outil au début du TPI ce qui n'a fait que renforcer mon choix.

Un autre risque auquel je fais généralement face dans de tels projets est que j'ai tendance à trop vouloir en faire, c'est-à-dire à me rajouter du travail supplémentaire, pas forcément nécessaire, simplement car j'ai envie de rendre un produit fini le plus complet possible. Généralement, ce travail supplémentaire impact le temps total à disposition et fait que la fin du projet est un peu tendue.

Le fait de transcrire ces risques techniques de manière écrite me permet d'en prendre conscience et d'y faire particulièrement attention pendant la durée du travail de TPI.



## 2.4 Planification

### Projet

Application web de fitness

	Total
Prévu	90 h 05
Koetschet	95 h 05

### Planification

02.05.23	08.05.23	15.05.23	22.05.23	29.05.23	05.06.23	12.06.23
18 h 20	23 h 55	16 h 45	23 h 55	7 h 10		
18 h 20	23 h 55	16 h 45	23 h 55	12 h 10		

		SEM	18	19	20	21	22	23	24	
<b>1 Analyse</b>										<b>13 h 30</b>
	11 Analyse du framework Laravel	Prévu	3 h 00							3 h 00
		Koetschet	1 h 35							1 h 35
	12 Installation du framework Laravel	Prévu	1 h 30							1 h 30
		Koetschet	1 h 35							1 h 35
	13 Analyse de l'API Open Food Facts	Prévu	3 h 00							3 h 00
		Koetschet	1 h 35							1 h 35
	14 Moodboard de l'application	Prévu	3 h 00							3 h 00
		Koetschet	1 h 35							1 h 35
	15 Maquette de l'application	Prévu	3 h 00							3 h 00
		Koetschet	2 h 25							2 h 25
	16 -	Prévu								
	17 -	Prévu								
	18 -	Prévu								
	19 -	Prévu								
<b>2 Implémentation</b>										<b>37 h 00</b>
	21 Conception du MCD	Prévu		3 h 00						3 h 00
		Koetschet	1 h 05							1 h 05
	22 Conception du MLD	Prévu		2 h 30						2 h 30
		Koetschet	2 h 05							2 h 05
	23 Création de la DB	Prévu		2 h 15						2 h 15
		Koetschet		1 h 35						1 h 35
	24 Création du projet	Prévu		0 h 45						0 h 45
		Koetschet		0 h 35						0 h 35
	25 Création du gabarit	Prévu		5 h 15						5 h 15
		Koetschet		4 h 25						4 h 25
	26 Page d'accueil / Connexion à l'application	Prévu		5 h 30						5 h 30
		Koetschet		4 h 35						4 h 35
	27 Page IMC	Prévu		3 h 00	3 h 00					6 h 00
		Koetschet			8 h 00					8 h 00
	28 Page données alimentaires journalières	Prévu			6 h 40					6 h 40
		Koetschet				6 h 30				6 h 30
	29 Login / Register	Prévu			5 h 15					5 h 15
		Koetschet		5 h 20	5 h 35		1 h 35			12 h 30
	30 Connexion à la DB / API	Prévu				7 h 00				7 h 00
		Koetschet		4 h 00		6 h 25				10 h 25
<b>3 Tests</b>										<b>10 h 05</b>
	31 Rédaction de la stratégie de test	Prévu				1 h 50				1 h 50
		Koetschet								
	32 Rédaction des tests	Prévu				3 h 00				3 h 00
		Koetschet					0 h 50			0 h 50
	33 Application des tests	Prévu				5 h 15				5 h 15
		Koetschet					1 h 35			1 h 35
	34 -	Prévu								
	35 -	Prévu								
	36 -	Prévu								
	37 -	Prévu								
	38 -	Prévu								
	39 -	Prévu								
<b>4 Documentation</b>										<b>22 h 30</b>
	41 Réalisation de la planification initiale	Prévu	3 h 00							3 h 00
		Koetschet	3 h 35							3 h 35
	42 Rédaction de la documentation	Prévu	1 h 30	1 h 30	1 h 30	6 h 30	6 h 30			17 h 50
		Koetschet	2 h 05	1 h 35	2 h 25	7 h 10	5 h 10			18 h 25
	43 Remplissage du journal de travail	Prévu	0 h 20	0 h 20	0 h 20	0 h 20	0 h 20			1 h 40
		Koetschet	0 h 15	0 h 20	0 h 15	0 h 15				1 h 05
	44 -	Prévu								
	45 -	Prévu								
	46 -	Prévu								
	47 -	Prévu								
	48 -	Prévu								
	49 -	Prévu								
<b>5 Autres</b>										<b>7 h 05</b>
	51 Absence	Prévu								
		Koetschet	0 h 30							0 h 30
	52 Autres	Prévu								
		Koetschet		1 h 30	0 h 30	1 h 35	3 h 00			6 h 35
	53 -	Prévu								
	54 -	Prévu								
	55 -	Prévu								
	56 -	Prévu								
	57 -	Prévu								
	58 -	Prévu								
	59 -	Prévu								

Image 15 Planification finale du projet de TPI

La planification finale de mon projet est assez différente de la planification initiale. On remarque que le début du projet c'est assez bien déroulé que la partie analyse a assez vite été terminée, même plus vite que ce qui était planifié. Cependant, la partie implémentation a pris plus de temps que ce qui était prévu et j'ai donc fait le choix de raccourcir le temps prévu pour les tests afin de pouvoir terminer mon application comme je le souhaitais.

## 2.5 Dossier de conception

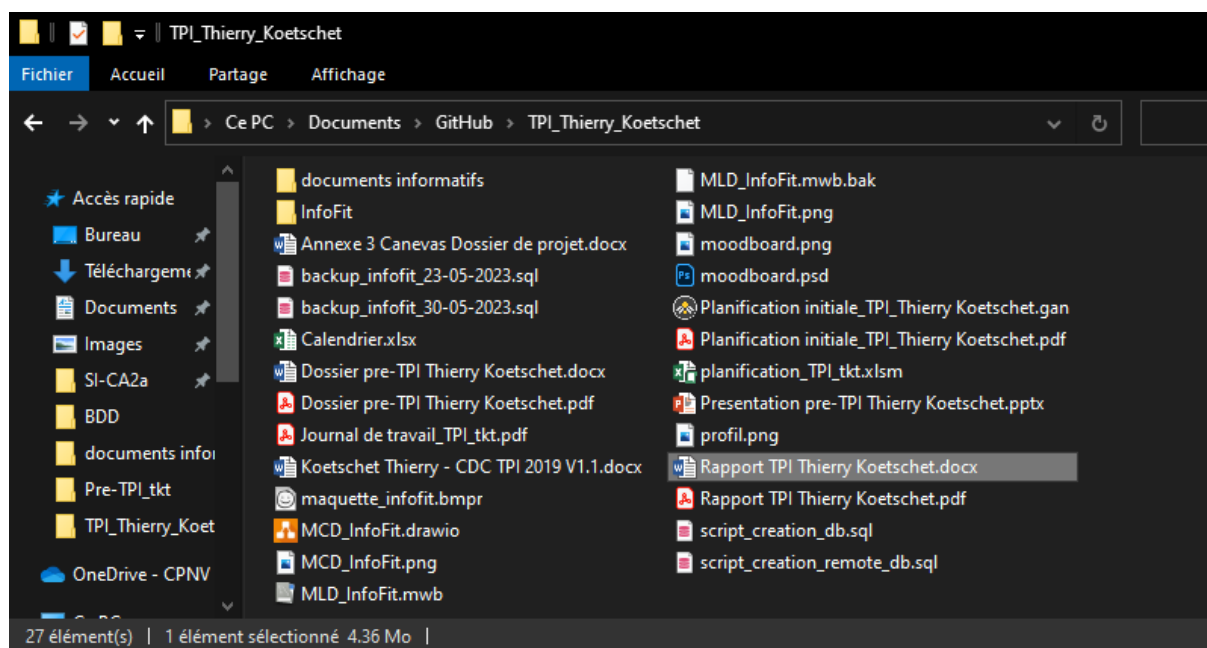


Image 16 Capture du dossier de TPI

La capture d'écran ci-dessus représente tous les fichiers qui ont servi d'une manière ou d'une autre à ce travail.

## 3 Réalisation

### 3.1 Dossier de réalisation

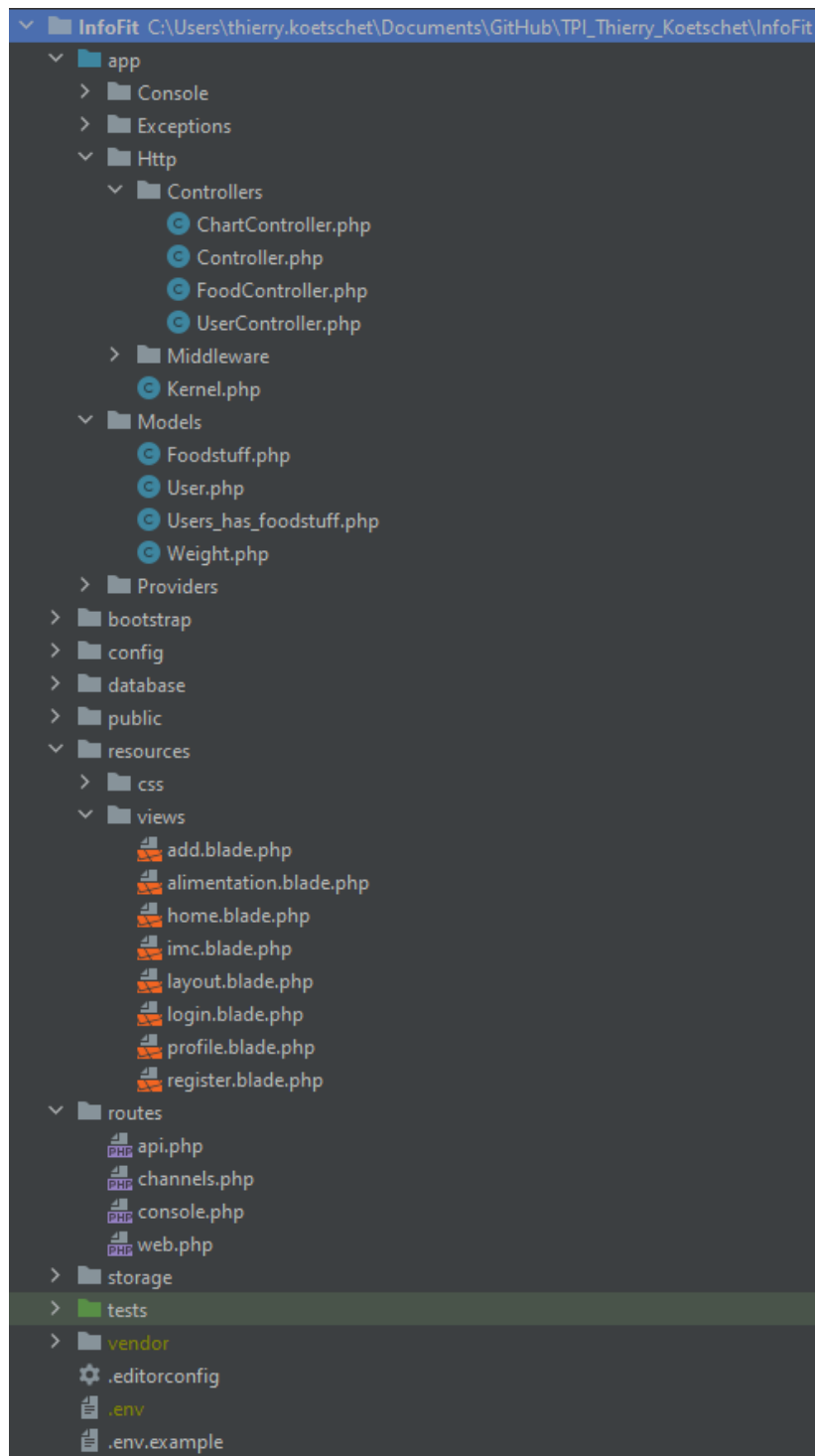


Image 17 Dossier du projet InfoFit

Vous trouverez ci-dessus une capture d'écran du dossier de mon application « InfoFit » avec les principaux dossiers et fichiers visibles.

Tout d'abord, le dossier InfoFit\resources\**views** contient tous les fichiers impactant le front end du site web. Il contient par exemple le gabarit (layout.blade.php) qui affiche le header et le footer de chaque page du site. Le reste des fichiers de ce dossier sont les différentes pages du site comme la page d'accueil (home.blade.php) ou encore la page d'enregistrement d'un utilisateur (register.blade.php). Ces fichiers sont principalement composés de code en HTML et CSS avec un peu de JavaScript et de PHP. Cependant, la majorité du CSS du site est contenu dans les fichiers **styles.css** et **charts.css** du dossier InfoFit\public\css. Les images, donc le logo et les différentes icônes, sont quant à elles contenues dans le dossier InfoFit\public\assets.

Un autre fichier important est le fichier InfoFit\routes\**web.php** car il contient toutes les routes de l'application, c'est-à-dire qu'il permet de faire le lien entre toutes ces vues et permet la transmission de données d'une page à l'autre.

Le dossier InfoFit\app\**Models** contient tous les fichiers permettant les interactions avec la base de données comme le fichier **User.php** qui permet d'insérer, modifier ou supprimer un utilisateur de l'application par exemple.

Le dossier InfoFit\app\Http\**Controllers** contient tous les fichiers servant au fonctionnement global de l'application. Le fichier **ChartController.php** contient les fonctions affichant le graphique d'évolution et calculant l'indice de masse corporelle. Le fichier **FoodController.php** permet d'afficher les aliments ajoutés par l'utilisateur par rapport à la date et la période d'ajout, de rechercher des aliments via une API et de supprimer des aliments ajoutés. Enfin, le fichier **UserController.php** contient une multitude de fonctions telles que la création, l'authentification et la suppression d'un utilisateur, la modification du profil utilisateur pour entre autre pouvoir traquer l'évolution du poids et enfin changer le mot de passe de son compte utilisateur.

### 3.2 Code source

Tout le code source est disponible dans le dossier InfoFit de mon dépôt GitHub ([https://github.com/ThierryKoetschet/TPI\\_Thierry\\_Koetschet](https://github.com/ThierryKoetschet/TPI_Thierry_Koetschet)). J'ai fait une petite sélection de fonctions qui mérite d'avoir une explication dans ce rapport.

### 3.2.1 Fonction « store » de UserController.php

```
public function store(Request $request)
{
    $formFields = $request->validate([
        'gender' => 'required',
        'lastname' => 'required|min:2',
        'firstname' => 'required|min:2',
        'email' => ['required', 'email', Rule::unique( table: 'users', column: 'email')],
        'password' => 'required|confirmed|min:6',
        'birthdate' => 'required',
        'height' => 'required|integer|gt:0',
        'weight' => 'required|integer|gt:0'
    ]);

    //Hash Password
    $formFields['password'] = bcrypt($formFields['password']);

    $user = new User();
    $weight = new Weight();

    $user->gender = $formFields['gender'];
    $user->lastname = $formFields['lastname'];
    $user->firstname = $formFields['firstname'];
    $user->email = $formFields['email'];
    $user->password = $formFields['password'];
    $user->birthdate = $formFields['birthdate'];
    $user->height = $formFields['height'];

    $user->save();
    Auth::login($user);

    $weight->value = $formFields['weight'];
    $weight->date = date( format: "Y-m-d");
    $weight->users_id = Auth::id();

    $weight->save();

    return redirect( to: '/profile')->with('success', 'Enregistrement réussi!');
}
```

Image 18 Capture d'écran de la fonction store

Cette fonction a pour but d'enregistrer un utilisateur qui utilise l'application InfoFit pour la première fois. On remarque qu'elle possède un paramètre « Request \$request » qui permet de récupérer les informations passées par l'envoi du formulaire en méthode post. Elle va alors commencer par valider tous les inputs renvoyés puis, si tous les inputs sont corrects, elle va enregistrer un nouvel utilisateur dans la base de données avec un mot de passe hashé et également un poids correspondant à l'utilisateur. Enfin, on authentifie l'utilisateur puis on retourne la page « profile.blade.php ».

### 3.2.2 Fonction « searchFoodstuff » de FoodController.php

```
//sends the API requests to the online database and translates the data to an array to display in the add page
public function searchFoodstuff(Request $request) {
    $productName = $request['foodstuff'];
    $date = $request['date'];
    $period = $request['period'];
    $infos = ['date' => $date, 'period' => $period];
    $api = file_get_contents( filename: 'https://fr.openfoodfacts.org/categorie/' . $productName . '.json');

    if ($api) {
        $json = json_decode($api, associative: true)['products'];
        $products = array_slice($json, offset: 0, length: 10);
        $productCode = [];
        $productSelection = [];

        foreach ($products as $product) {
            $foodstuff = new Foodstuff();

            $foodstuff->code = $product['code'];

            array_push( $array: $productCode, $foodstuff);
        }

        //this loop generates a 10 items array based on the keyword written in the input
        foreach ($productCode as $product) {
            $api = file_get_contents( filename: 'https://world.openfoodfacts.org/api/v2/product/' . $product->code);
            if ($api) {
                $json = json_decode($api, associative: true)['product'];

                $foodstuff = new Foodstuff();

                $foodstuff->code = $json['code'];
                $foodstuff->title = $json['product_name_fr'];
                if (isset($json['nutriments']['energy-kcal_100g'])) {
                    $foodstuff->kcal_100g = $json['nutriments']['energy-kcal_100g'];
                } else {
                    if (isset($json['nutriments']['energy'])) {
                        $foodstuff->kcal_100g = $json['nutriments']['energy'];
                    } else {
                        $foodstuff->kcal_100g = 0;
                    }
                }
            }
        }
    }
}
```

Image 19 Première partie capture de la fonction searchFoodstuff

```

    } else {
        if (isset($json['nutriments']['carbohydrates'])) {
            $foodstuff->carbohydrates_100g = $json['nutriments']['carbohydrates'];
        } else {
            $foodstuff->carbohydrates_100g = 0;
        }
    }

    if (isset($json['nutriments']['fat_100g'])) {
        $foodstuff->lipids_100g = $json['nutriments']['fat_100g'];
    } else {
        if (isset($json['nutriments']['fat'])) {
            $foodstuff->lipids_100g = $json['nutriments']['fat'];
        } else {
            $foodstuff->lipids_100g = 0;
        }
    }

    if (isset($json['nutriments']['proteins_100g'])) {
        $foodstuff->proteins_100g = $json['nutriments']['proteins_100g'];
    } else {
        if (isset($json['nutriments']['proteins'])) {
            $foodstuff->proteins_100g = $json['nutriments']['proteins'];
        } else {
            $foodstuff->proteins_100g = 0;
        }
    }

    array_push(&array: $productSelection, $foodstuff);
}
else {
    return back()->withErrors(['message' => 'Le produit recherché n\'existe pas']);
}
}
else {
    return back()->withErrors(['message' => 'Le produit recherché n\'existe pas']);
}

return view('view: /add', ['productSelection'=>$productSelection, 'infos'=>$infos]);
}

```

Image 20 Deuxième partie capture de la fonction searchFoodstuff

Tout comme la fonction « store », on récupère les données envoyées dans les inputs du formulaire grâce au paramètre « Request \$request ». Puis elle va faire une première requête API qui va rechercher le code des produits qui ont pour catégorie le texte saisi dans l'input du formulaire. La méthode « file\_get\_contents() » permet de récupérer les informations obtenues grâce à la requête API puis la méthode « json\_encode() » permet de transformer ce grande quantité de texte au format JSON afin d'avoir un tableau associatif lisible en PHP. On réduit le nombre d'objet dans ce tableau à 10 afin de ne pas avoir une trop grande liste puis on effectue une nouvelle requête API produit par produit avec leur code qu'on a récolté précédemment. La fonction va alors créer un nouveau tableau associatif qui regroupera uniquement les informations nécessaires à afficher dans la vue, c'est-à-dire le nombre de kcal, glucides, lipides, protéines par 100 grammes ainsi que le code et nom du produit. Enfin, on renvoie ce tableau ainsi que différentes informations telles que la date sélectionnée et la période dans la vue « add.blade.php ».

### 3.2.3 Fonction « lineChart » de ChartController.php

```
//creates the array that is displayed in the chart
public function lineChart() {
    $users_id = Auth::id();

    //gets all the weights saved by the user
    $weights = Weight::where('users_id', '=', $users_id)->orderby('updated_at', 'asc')->get();

    $users_height = Auth::user()->height;
    $data = [['Date', 'Poids', 'IMC']];

    foreach ($weights as $weight) {
        //array displayed in the chart
        array_push($data, [$weight->date, $weight['value'], $this->calculateImc($weight['value'], $users_height)]);
    }

    return view('view: /imc', ['data'=>$data]);
}
```

Image 21 Capture de la fonction lineChart

Dans la variable « \$weights », on récupère un tableau de la table « weights » dans notre base de données grâce à une requête avec comme critère l'id de l'utilisateur et dans l'ordre ascendant. On enregistre également la taille de l'utilisateur dans la variable « \$users\_height ». On effectue alors une boucle « foreach » afin de créer notre tableau associatif contenant les informations affichées dans le graphique de la vue IMC.



### 3.3 Aperçu de l'application

#### 3.3.1 Page d'accueil



Image 22 Capture de la page d'accueil de InfoFit

#### 3.3.2 Login

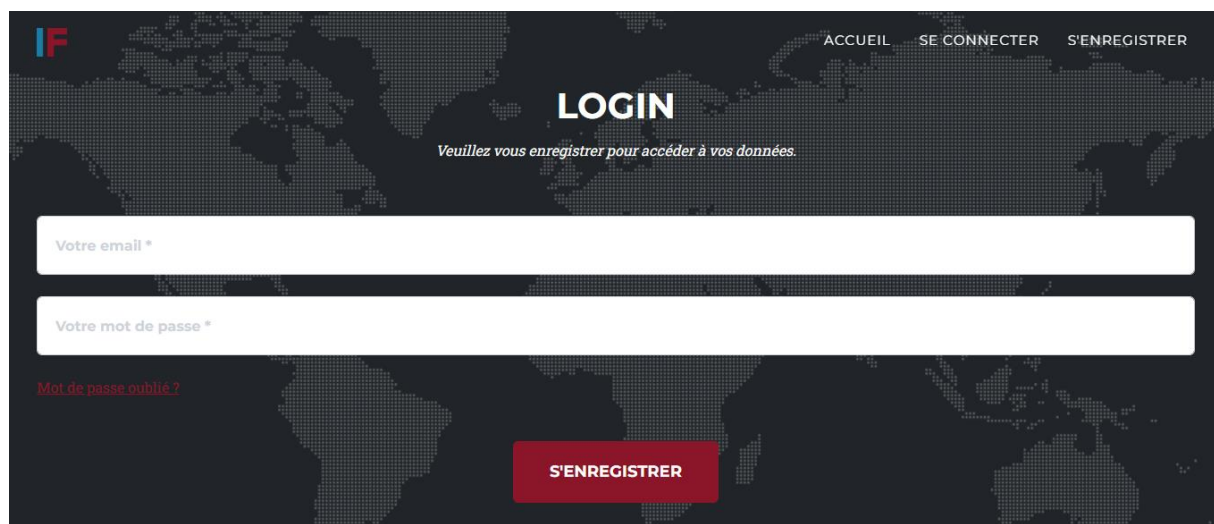
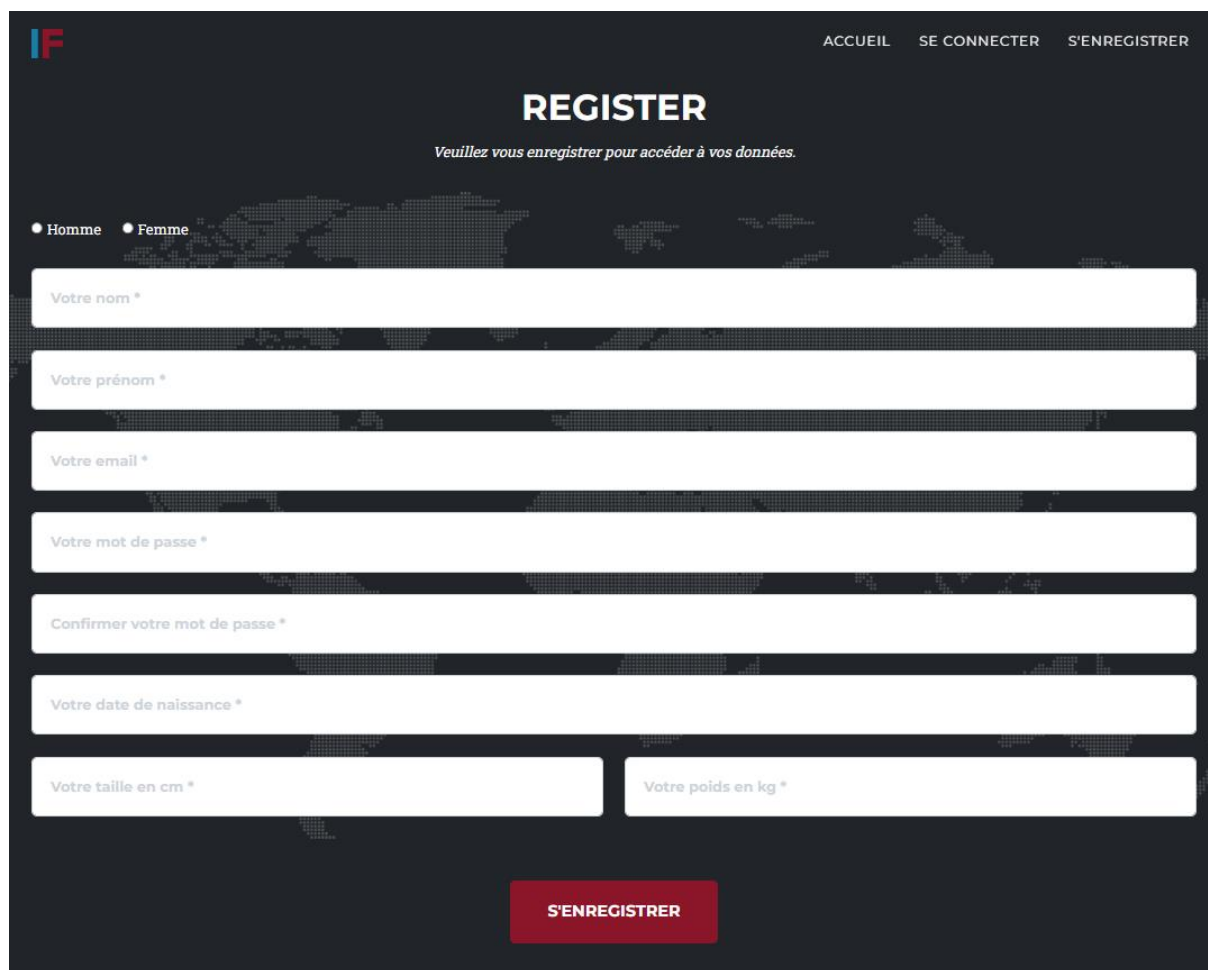


Image 23 Capture de la page de login de InfoFit

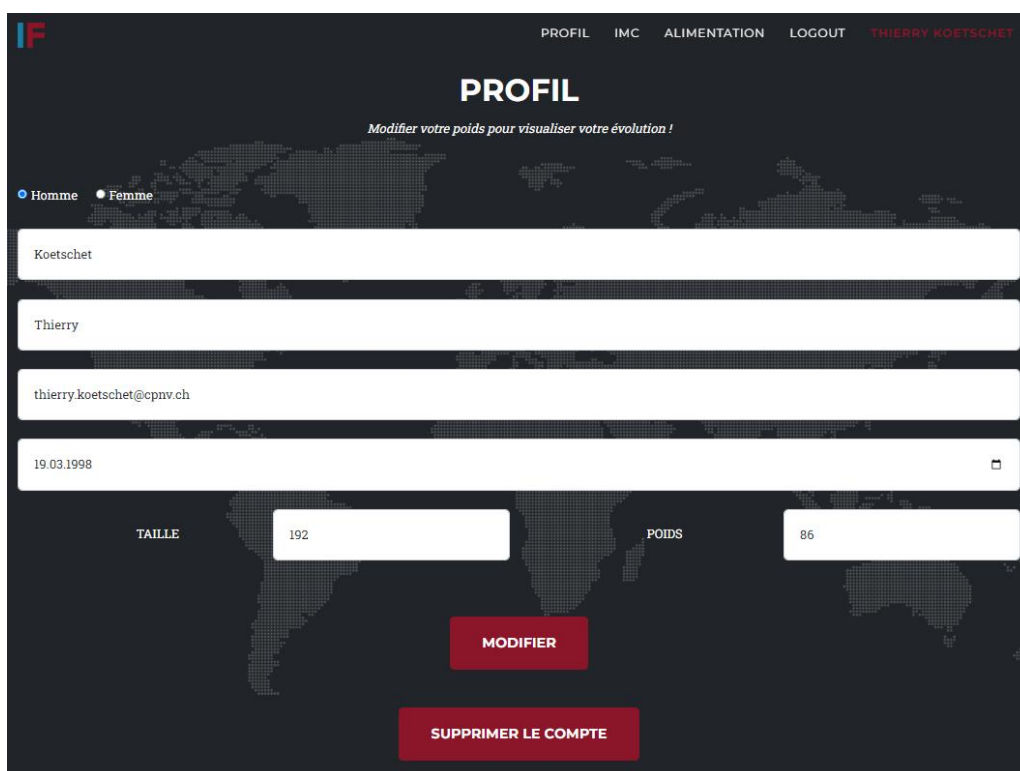
### 3.3.3 Register



The screenshot shows the 'REGISTER' page of the InfoFit application. The page has a dark blue background with a faint world map. At the top left is the 'IF' logo. At the top right are navigation links: 'ACCUEIL', 'SE CONNECTER', and 'S'ENREGISTRER'. The main heading is 'REGISTER' in large white letters, followed by the instruction 'Veuillez vous enregistrer pour accéder à vos données.' Below this are two radio buttons for 'Homme' (selected) and 'Femme'. The registration form consists of several white input fields: 'Votre nom \*', 'Votre prénom \*', 'Votre email \*', 'Votre mot de passe \*', 'Confirmer votre mot de passe \*', 'Votre date de naissance \*', 'Votre taille en cm \*', and 'Votre poids en kg \*'. At the bottom center is a red button labeled 'S'ENREGISTRER'.

Image 24 Capture de la page de register de InfoFit

### 3.3.4 Profil



IF

PROFIL IMC ALIMENTATION LOGOUT THIERRY KOETSCHET

## PROFIL

Modifier votre poids pour visualiser votre évolution !

☒ Homme ☐ Femme

Koetschet

Thierry

thierry.koetschet@cpnv.ch

19.03.1998

TAILLE 192 POIDS 86

MODIFIER

SUPPRIMER LE COMPTE

Image 25 Capture de la page de profil de InfoFit

### 3.3.5 IMC

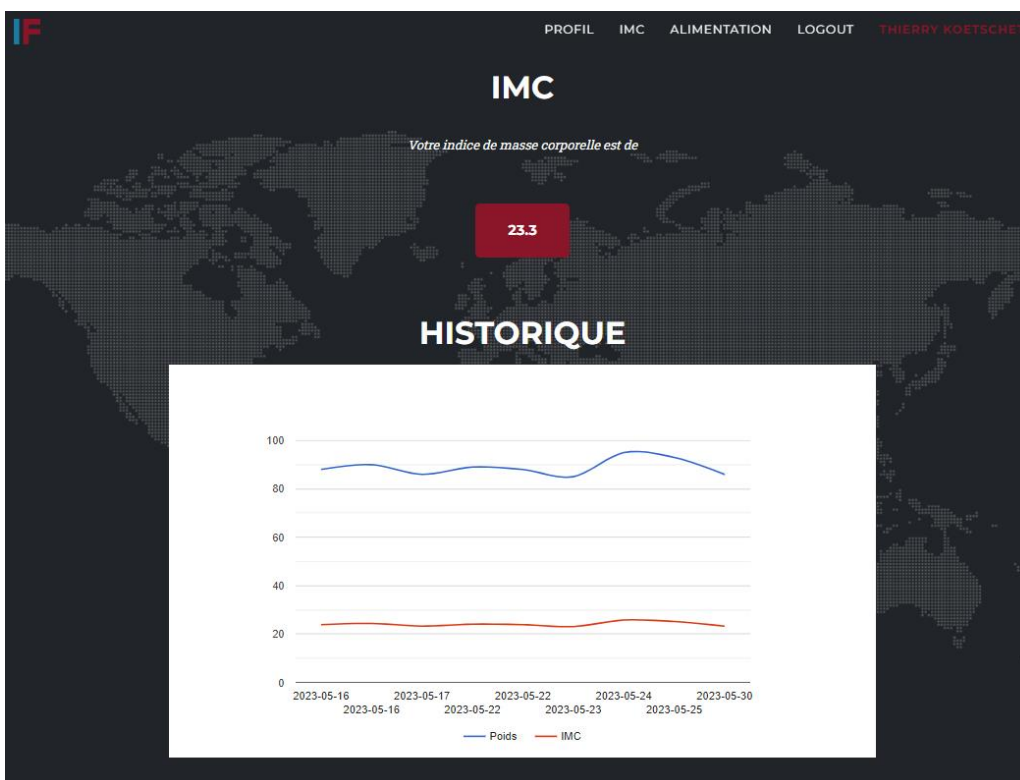


Image 26 Capture de la page d'IMC de InfoFit

### 3.3.6 Alimentation

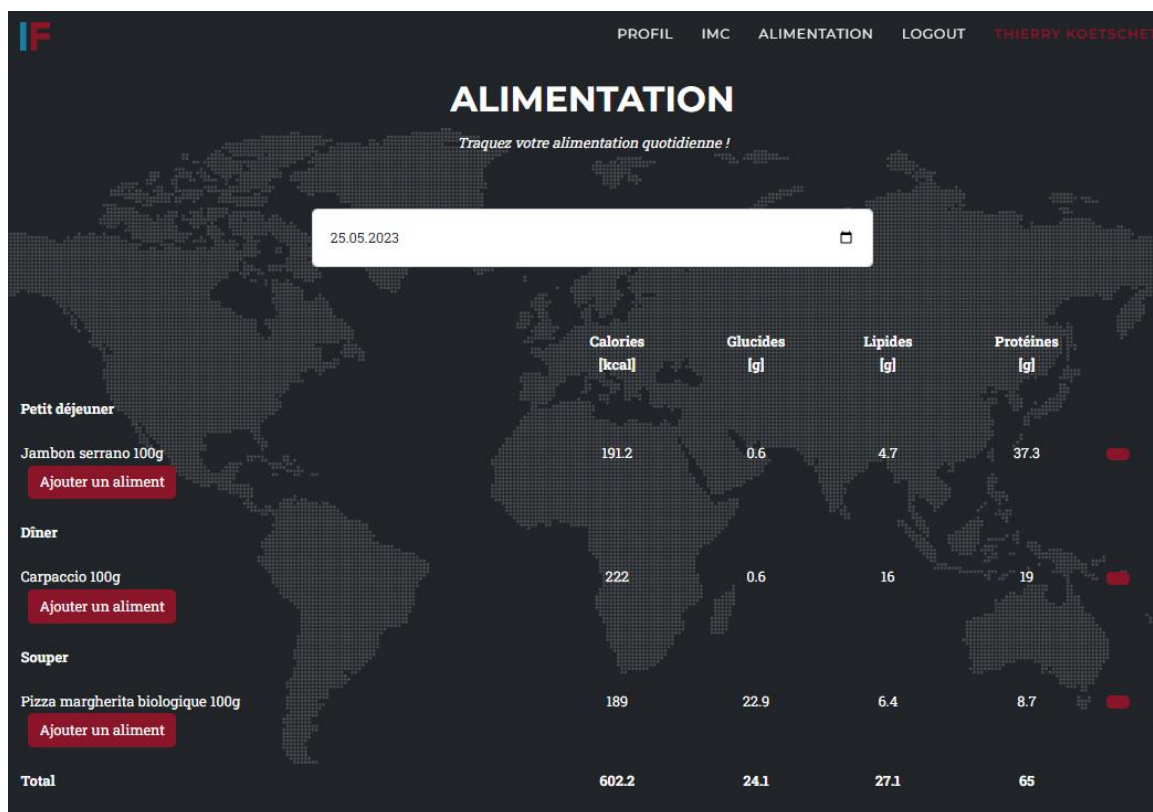


Image 27 Capture de la page d'alimentation de InfoFit

### 3.3.7 Ajouter un aliment



NOM	CALORIES	GLUCIDES	LIPIDES	PROTÉINES	QUANTITÉ
BURRATA DI BUFALA	288	0.8	25	15	1
FROMAGE FONDU NATURE	265	6	21	13	1
CREME DE REBLOCHON	256	1.5	22	1.3	1
FETA	300	2	25	16	1
CANCOILLOTTE IGP À L'AIL	157	0.6	11	14	1
EMMENTAL FRANÇAIS	369	0	29	27	1
MINI MOZZARELLA	245	1.5	18.5	18	1
EMMENTAL	362	2.3	27	27	1
MON PETIT PLAISIR NATURE	201	3.4	17	8.1	1
LE CHEVILLON	413	1	37	10	1

Image 28 Capture de la page d'ajout d'aliment de InfoFit

### 3.4 Hébergement sur le serveur web





Concernant l'hébergement de mon site web sur un serveur web, il y a quelques étapes à réaliser en plus.

Tout d'abord, il faut créer une nouvelle base de données dans la partie MySQL du serveur. J'ai donc adapté mon script de création de ma base de données en changeant le nom pour que tout corresponde et je l'ai exécuté afin de créer la nouvelle base de données. Le script est disponible sur mon dépôt GitHub sous le nom `script_creation_remote_db.sql`.












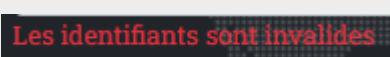

J'ai ensuite transféré le dossier de mon application sur mon serveur web grâce au client FTP FileZilla en m'assurant de modifier le fichier `.env` pour qu'il corresponde à la nouvelle base de données, et ce avant le transfert car le fichier est inaccessible depuis le serveur.

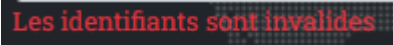












Enfin, il suffit de modifier le répertoire de base du domaine de la configuration Apache du serveur pour que lorsque l'on saisit l'adresse de mon serveur web, <https://tkoetschet.mycpnv.ch/>, l'application InfoFit s'affiche et fonctionne correctement.

### 3.5 Description des tests effectués

Date	Test	Résultat	Validation
<b>Tests des routes</b>			
30.05.2023	Page d'accueil accessible.	La commande <b>php artisan serve</b> ouvre directement la page d'accueil.  Le lien « accueil » de la barre de navigation fonctionne.	
30.05.2023	Page de login accessible.	Le lien dans la barre de navigation fonctionne.  Le bouton de la page d'accueil fonctionne.	
30.05.2023	Page de register accessible.	Le lien dans la barre de navigation fonctionne.  Le bouton de la page d'accueil fonctionne.	
30.05.2023	<a href="#">Mot de passe oublié ?</a>	Le lien fonctionne correctement.	



30.05.2023	Bouton « modifier » de la page profil.	Il renvoie à la page « Profil »..	
30.05.2023	Bouton « supprimer le compte » de la page profil.	Il renvoie à la page d'accueil.	
30.05.2023	Bouton « Profil » de la barre de navigation	Il renvoie à la page « Profil ».	
30.05.2023	Bouton « IMC » de la barre de navigation	Il renvoie à la page « IMC ».	
30.05.2023	Bouton « Alimentation » de la barre de navigation	Il renvoie à la page « Alimentation ».	
30.05.2023	Bouton « Logout » de la barre de navigation	Il renvoie à la page d'accueil.	
30.05.2023	Input de date de la page alimentation	Il renvoie à la page « Alimentation » avec la date sélectionnée.	
30.05.2023	Boutons « Ajouter un aliment » de la page « Alimentation ».	Ils renvoient à la page « Ajouter un aliment ».	
30.05.2023	Bouton « chercher » de la page « Ajouter un aliment ».	Il renvoie à la page « Ajouter un aliment » s'il y a quelque chose de saisi dans l'input, sinon il renvoie une erreur. Parfois, l'API ne trouve pas non plus le texte saisi dans l'input et affiche une erreur dans l'application	
<b>Tests fonctionnels</b>			
30.05.2023	Login		
30.05.2023	Login avec un email inexistant		

30.05.2023	Login avec un faux mot de passe		
30.05.2023	Register		
30.05.2023	Logout	Retour à la page d'accueil et nom d'utilisateur plus affiché au coin de l'écran.	
30.05.2023	Modification du profil	Nom, prénom, genre, email, date de naissance, taille et poids modifiable	
30.05.2023	Mot de passe oublié	Mot de passe réinitialisable sans la demande de l'ancien mot de passe	
30.05.2023	Suppression du compte en cascade	Compte supprimé ainsi que tous enregistrement faits par celui-ci dans la base de données.	
30.05.2023	Graphique d'évolution du poids	Affiche chaque enregistrement correctement	
30.05.2023	Changement de date du tableau alimentation	La page se rafraîchit et enregistre la nouvelle date	
30.05.2023	Recherche d'un aliment via l'API	La recherche ne fonctionne pas à chaque fois. Par exemple, le recherche avec un input avec une valeur nulle provoque une erreur. Certains aliments sont également introuvables via l'API et provoquent une erreur.	
30.05.2023	Ajout d'un aliment dans le tableau alimentation	L'aliment s'ajoute dans le tableau alimentation à la bonne date et à la bonne période.	
30.05.2023	Suppression d'un aliment du tableau alimentation	L'aliment se supprime correctement et l'enregistrement disparaît de la base de données.	

### 3.6 Erreurs restantes

La page « Ajouter un aliment » de l'application contient encore plusieurs erreurs : Cliquez sur le bouton « chercher » avec un input vide va causer une erreur dans l'application. De plus, l'affichage de l'input « quantité » de liste d'aliments retournés n'est pas visible en mode étroit et certains aliments recherchés dans l'API n'existe pas et cause donc une erreur au niveau de l'application.

J'ai également remarqué que certaines pages de l'application posent problème lorsque que l'on n'est pas logué. Par exemple, la page « profile » est problématique si l'on saisit l'URL <http://127.0.0.1:8000/alimentation/profile> dans le navigateur web sans être authentifié.

### 3.7 Liste des documents fournis

Tous les documents fournis au cours ou à la fin de ce projet sont les suivants :

- Une planification initiale fournie le 2 mai 2023
- L'état d'avancée du journal de travail au format PDF chaque mardi et jeudi sur toute la durée du projet
- L'état d'avancée du rapport de TPI au format PDF chaque jeudi sur toute la durée du projet
- Le code source de l'application fourni le 31 mai 2023 sur mon dépôt GitHub ([https://github.com/ThierryKoetschet/TPI\\_Thierry\\_Koetschet](https://github.com/ThierryKoetschet/TPI_Thierry_Koetschet))
- Un rapport de TPI complet et un journal de travail au format PDF fourni le 31 mai 2023 à chaque expert et au chef de projet par email et une version imprimée de chaque pour l'expert numéro 1
- Une procédure d'installation et de mise en service de l'application fournie le 31 mai 2023 en annexe au rapport de TPI
- Un accès à l'application sur la plateforme mycpnv.ch fourni le 31 mai 2023

La date de rendu de certains de ces documents diffère du cahier des charges et est retardée au 31 mai 2023 pour cause de problèmes de connexion internet survenus le 30 mai 2023 pendant une période de 2 heures dus à la casse d'une fibre optique lors de travaux à proximité du bâtiment du CPNV de Sainte-Croix.



## 4 Conclusions

Pour conclure, je pense avoir atteint les objectifs fixés autant dans mon cahier des charges comme les objectifs personnels que je m'étais fixés au début du projet. En effet, toutes les fonctionnalités décrites dans le cahier des charges semblent être respectées. Le site contient un système d'authentification sécurisé, la page de calcul d'indice de masse corporelle fonctionne correctement et enfin la page permettant aux utilisateurs de l'application d'enregistrer leurs données alimentaires journalières est opérationnelle avec des requêtes API allant chercher des informations sur l'API Open Food Facts. Je pense également avoir acquis de nombreuses connaissances sur l'utilisation du framework Laravel qui était mon objectif personnel à atteindre. J'espère que ces connaissances auront une utilité dans ma future carrière professionnelle.

Cependant, je me rends compte qu'il y a certains points de la réalisation de projet que je pourrais améliorer comme la gestion de mon temps qui pourrait être meilleure. La fin de ce projet de TPI fut quelque peu intensive car j'ai consacré un peu trop de temps à l'implémentation de l'application ce qui fait que j'ai manqué de temps pour la rédaction de la documentation et ai dû faire un peu de travail à domicile.

En somme, je suis satisfait du produit que je remets car il s'agit d'un outil je vais utiliser quotidiennement dans mon activité sportive afin de suivre mes progrès et mes informations personnelles. Et comme mon ambition future est de potentiellement lancer mon entreprise d'informatique, je pourrai m'inspirer et utiliser tous ces outils appris lors de ce projet de TPI.

## 5 Annexes

### 5.1 Cahier des charges









CPNV		Filère informatique		Examen - TPI	
<b>1 INFORMATIONS GENERALES</b>					
Candidat :	Nom : <b>KOETSCHET</b>	Prénom : <b>THIERRY</b>			
	 : <a href="mailto:Thierry.KOETSCHET@cpnv.ch">Thierry.KOETSCHET@cpnv.ch</a>	 : 079 194 55 39			
Lieu de travail :	<input type="checkbox"/> CPNV, Rue de la Gare 14, 1450 Sainte-Croix				
Orientation :	<input type="checkbox"/> 88601 Développement d'application <input checked="" type="checkbox"/> 88602 Informatique d'entreprise <input type="checkbox"/> 88603 Technique des systèmes				
Chef de projet :	Nom : SAISON	Prénom : Yann			
	 : <a href="mailto:yann.saison@eduvaud.ch">yann.saison@eduvaud.ch</a>	 : 079 610 08 47			
Expert 1 :	Nom : Malherbe	Prénom : Roger			
	 : <a href="mailto:r.malherbe@rmsoft.ch">r.malherbe@rmsoft.ch</a>	 : 079 230 72 37			
Expert 2 :	Nom : Charmier	Prénom : Grégory			
	 : <a href="mailto:gregory.charmier@gmail.com">gregory.charmier@gmail.com</a>	 : +33 6 45 92 84 86			
Période de réalisation :	Du <b>mardi 2 mai 2023 à 8h00</b> au <b>mardi 30 mai 2023 à 16h50</b>				
Horaire de travail :	Lundi	09h50-12h15	13h30-16h55	Pentecôte 29 mai	
	Mardi	08h00-12h15	13h30-16h55		
	Mercredi	08h00-12h15	-		
	Jeudi	08h00-12h15	13h30-16h55	Ascension 18 mai	
	Vendredi	-	-	Pont de l'Ascension 19 mai	
	Toutes les demi-journées ont une pause obligatoire de 15 minutes, sauf si elles se commencent à 09h50.				
Nombre d'heures :	90 heures				
Planning (en H ou %)	Analyse 20% Implémentation 40% Tests 20% Documentation 20%				
Présentation :	Dates retenues : 12 ou 13 juin 2023				
<b>2 PROCÉDURE</b>					
<p>Le candidat réalise un travail personnel sur la base d'un cahier des charges reçu le 1er jour.</p> <p>Le cahier des charges est approuvé par les deux experts. Il est en outre présenté, commenté et discuté avec le candidat. Par sa signature, le candidat accepte le travail proposé.</p> <p>Le candidat a connaissance de la feuille d'appréciation avant de débiter le travail.</p> <p>Le candidat est entièrement responsable de la sécurité de ses données.</p> <p>En cas de problèmes graves, le candidat avertit au plus vite les deux experts et son CdP.</p> <p>Le candidat a la possibilité d'obtenir de l'aide, mais doit le mentionner dans son dossier.</p> <p>A la fin du délai imparti pour la réalisation du TPI, le candidat doit transmettre par courrier électronique le dossier de projet aux deux experts et au chef de projet. En parallèle, une copie papier du rapport doit être fournie sans délai en trois exemplaires (L'un des deux experts peut demander à ne recevoir que la version électronique du dossier). Cette dernière doit être en tout point identique à la version électronique.</p>					
Fichier : CDC_YSN_Thierry_Koetschet_FrameworkPHP.docx			Version 33 du 30.03.2023 13:27:00		
Auteur :			Dernière modification le 28.04.2023		
			Imprimé le 28.04.2023 12:04:00 à 12:04		

Image 29 Première page du cahier des charges

CPNV

Filière informatique

Examen - TPI

---

**3 TITRE**

Application web de fitness.

---

**4 MATÉRIEL ET LOGICIEL À DISPOSITION**

- 1 PC en configuration standard CPNV (Windows 10) avec accès à internet
- Environnement de développement Web/PHP
- Serveur web local / distant (mycpnv.ch)
- Outil de modélisation de base de données
- Outil de gestion de versions tel git

---

**5 PRÉREQUIS**

- Développement Web (HTML5, CSS, PHP, Javascript)
- Modélisation et gestion de base de données

---

**6 DESCRIPTIF DU PROJET**

L'application finale doit permettre à chaque utilisateur de maîtriser ses données personnelles de santé et d'activités physiques.

**Fonctionnalités :**

- Authentification et données personnelles :
  - Les visiteurs peuvent s'inscrire et ainsi créer un compte utilisateur.
  - Les utilisateurs peuvent s'authentifier de manière sécurisée avec un mot de passe crypté et réinitialiser leur mot de passe en cas d'oubli.
  - Les utilisateurs authentifiés peuvent gérer leurs données personnelles de profil d'utilisateur (email, mot de passe, etc.), ainsi que leurs données physiques de base : sexe, taille, poids, âge, etc.
- IMC (indice de masse corporelle) :
  - L'application doit calculer l'IMC des utilisateurs en fonction de leurs données personnelles et l'afficher sur leur profil.
  - L'application doit afficher un historique de l'IMC (ou du poids) de l'utilisateur de manière simple et compréhensive, idéalement à l'aide de graphiques.
- API Open Food Facts :
  - Les utilisateurs peuvent enregistrer leurs données alimentaires journalières.
  - L'application doit pouvoir récupérer les données de produits alimentaires via l'API Open Food Facts (<https://openfoodfacts.org/data>).
  - En fonction des données alimentaires et de l'API, l'application doit afficher l'historique des apports alimentaires des utilisateurs, idéalement de manière graphique.

**Contraintes technologiques :**

Afin d'assurer la portabilité de l'application, il est demandé d'utiliser un framework PHP/MySQL (MVC) : Yii, Laravel, Symfony, ...

---

Fichier : CDC\_YSN\_Thierry\_Koetschet\_FrameworkPHP.docx  
Auteur :

Page 2 sur 3

Version 33 du 30.03.2023 13:27:00  
Dernière modification le 28.04.2023  
Imprimé le 28.04.2023 12:04:00 à 12:04

Image 30 Deuxième page du cahier des charges

## 7 LIVRABLES

Le candidat est responsable de livrer à son chef de projet et aux deux experts :

- Une planification initiale à la fin de la première journée.
- Chaque jeudi en fin de journée, par email et au format PDF :
  - Un rapport de projet.
  - Un journal de travail.
- Le code source est jour sur un dépôt accessible à chaque fin de journée de travail.
- A la fin du TPI :
  - Un rapport de projet et son journal de travail sous forme imprimé et par email au format PDF.
  - Le code source sur un dépôt (GIT) accessible.
  - Une procédure d'installation et de mise en service
  - Le site sur la plateforme disponible (mycpnv.ch)

## 8 POINTS TECHNIQUES ÉVALUÉS SPÉCIFIQUES AU PROJET

La grille d'évaluation définit les critères généraux selon lesquels le travail du candidat sera évalué (documentation, journal de travail, respect des normes, qualité, ...).

En plus de cela, le travail sera évalué sur les 7 points spécifiques suivants (Point A14 à A20):

1. La qualité de la base de données : relations, intégrité référentielle
2. La pertinence des types de données et leurs bonnes validations dans l'application (modèles).
3. Le respect des standards de développement d'applications : code simple, compréhensible et documenté.
4. La qualité des vues : simplicité, compréhension.
5. La bonne gestion des relations entre données dans l'application.
6. La cohérence des choix technologiques.
7. La bonne implémentation des composants additionnels (graphes, API).

## 9 VALIDATION

	Lu et approuvé le :	Signature :
Candidat :		
Expert n°1 :		
Expert n°2 :		
Chef de projet :		

Image 31 Troisième page du cahier des charges

## 5.2 Horaire de travail

	Lundi	Mardi	Mercredi	Jeudi	Vendredi
08:00 – 09:35					
09:50 – 12:15					
Pause midi					
13:30 – 15:05					
15:20 – 16:55					

### 5.3 Résumé du travail

Ce projet de TPI est l'achèvement d'une formation de deux ans en tant qu'informaticien d'entreprise et consiste en la réalisation d'un travail pratique d'environ 90 heures. Mon cahier des charges me fixait comme objectifs de réaliser un site web avec l'utilisation d'un framework PHP et devant contenir trois principales fonctionnalités : La première fonctionnalité demande à l'utilisateur de s'authentifier afin de pouvoir accéder aux deux autres fonctionnalités. La seconde permet aux utilisateurs de calculer leur indice de masse corporelle en fonction du poids saisi dans leur profil et d'avoir une représentation graphique de leur évolution. La troisième un genre de calendrier alimentaire permettant aux utilisateurs de récupérer de nombreuses informations sur les différents aliments qu'ils ont consommé dans leur journée avec l'aide de requêtes API.

La planification initiale pour la réalisation de ce projet a été faite de la manière suivante : le travail a été divisé en quatre sprint. Le premier sprint se compose de la partie analyse et conception du travail qui représente globalement l'apprentissage de l'utilisation de nouveaux outils tels que le framework Laravel ou encore l'API Open Food Facts ainsi que tout le travail de conception de la base de données, le moodboard et la maquette de l'application. Le sprint suivant est le plus conséquent car il contient toute la partie implémentation du projet, c'est-à-dire l'écriture du code source de l'application. Celle-ci se compose de vues écrites en HTML, CSS et JavaScript, de contrôleurs contenant toutes les fonctions nécessaires au fonctionnement de l'application, de modèles permettant de faire la relation entre le site web et la base de données et enfin de routes faisant le lien entre les différentes vues de l'application. Le troisième sprint concerne tout ce qui se rapporte aux tests de l'application, donc la stratégie de test, la rédaction et l'application des tests. Pour finir, le quatrième sprint contient l'écriture de la documentation et l'organisation du projet. Ce sprint s'étale sur toute la durée du projet.

Le résultat de ces quatre semaines de travail uniquement consacrées à ce projet sont un site web répondant aux critères posés dans le cahier des charges, une base de données qui a connu une certaine évolution au cours de l'implémentation de l'application mais qui fonctionne parfaitement avec celle-ci, un rapport de TPI et un journal de travail contenant chaque tâche effectuée au cours du travail. Je pense également avoir progressé sur le plan personnel au niveau du développement web et plus particulièrement sur l'utilisation de framework PHP qui je pense pourrait intéresser de futurs employeurs.

## 5.4 Sources – Bibliographie

Site Yii Framework :

<https://www.yiiframework.com/> consulté le 02.05.2023

Site Laravel :

<https://laravel.com/> consulté le 02.05.2023

Site Open Food Facts :

<https://ch-fr.openfoodfacts.org/> consulté le 02.05.2023

Site MyFitnessPal :

<https://www.myfitnesspal.com/fr> consulté le 02.05.2023

Site Composer :

<https://getcomposer.org/download/> consulté le 02.05.2023

API denrées alimentaires :

<https://developer.edamam.com/food-database-api-docs> consulté le 02.05.2023

<https://world.openfoodfacts.org/data> consulté le 04.05.2023

Source d'information pour la stratégie de tests :

<https://www.atlassian.com/fr/continuous-delivery/software-testing/types-of-software-testing> consulté le 09.05.2023

Liens pour templates HTML/CSS :

<https://startbootstrap.com/themes> consulté le 09.05.2023

Liens pour graphiques JS :

<https://developers.google.com/chart?hl=fr> consulté le 16.05.2023

## 5.5 Journal de travail

Date	Semaine	Activité	Heures	Description	Remarques
02.05.2023	18	51 Absence	0:30	Présentation du cahier des charges par M. Malherbe	
02.05.2023	18	41 Réalisation de la planification initiale	3:35	Réalisation de la planification initiale	
02.05.2023	18	11 Analyse du framework Laravel	1:35	Analyse des différents frameworks proposés dans le cahier des charges et choix de Laravel	
02.05.2023	18	12 Installation du framework Laravel	1:35	Installation du framework et création d'un projet	Problème rencontré au niveau du fichier php.ini car il manquait une extension mais le problème a été réglé
03.05.2023	18	13 Analyse de l'API Open Food Facts	1:35	Analyse d'API pouvant être utiliser dans mon application	
03.05.2023	18	14 Moodboard de l'application	1:35	Création du moodboard (logos, palette de couleurs)	
03.05.2023	18	15 Maquette de l'application	0:50	Début de la maquette de l'application	Fichier : maquette_infokit.bmp
04.05.2023	18	15 Maquette de l'application	1:35	Fin de la maquette	
04.05.2023	18	42 Rédaction de la documentation	2:05	Début de mise en forme de la documentation et rédaction de l'introduction	
04.05.2023	18	21 Conception du MCD	1:05	Conception du MCD avec draw.io	Je ne sais pas encore comment intégrer toute la partie API à ma base de données et à mon application
04.05.2023	18	22 Conception du MLD	2:05	Conception du MLD avec MySQL Workbench	



<b>04.05.2023</b>	18	43 Remplissage du journal de travail	0:15	Remplissage du journal de travail en fonction du travail réalisé dans la semaine	
<b>08.05.2023</b>	19	23 Création de la DB	0:30	Génération du script de création de la base de données avec MySQL Workbench et exécution du script sur HeidiSQL	
<b>08.05.2023</b>	19	24 Création du projet	0:35	Création de l'application avec Laravel	
<b>08.05.2023</b>	19	52 Autres	0:45	Choix d'un template HTML/CSS pour le site	
<b>08.05.2023</b>	19	26 Page d'accueil / Connexion à l'application	0:35	Début de la page d'accueil et prise en main du framework	
<b>08.05.2023</b>	19	23 Création de la DB	1:05	Mise à jour de la base de données	Après discussion avec M. Saison, j'ai mis à jour les MCD, MLD, et la base de données pour y ajouter une table poids afin d'avoir un historique de l'évolution du poids de chaque user.
<b>08.05.2023</b>	19	25 Création du gabarit	2:05	Création du gabarit	Fichier : layout.blade.php
<b>09.05.2023</b>	19	26 Page d'accueil / Connexion à l'application	4:00	Page d'accueil fonctionnelle	Fichier : home.blade.php
<b>09.05.2023</b>	19	25 Création du gabarit	2:20	Gabarit fonctionnel	
<b>09.05.2023</b>	19	52 Autres	0:45	Visualisation d'une vidéo sur le fonctionnement du login et de la connexion à la base de données avec Laravel	
<b>09.05.2023</b>	19	43 Remplissage du journal de travail	0:05	Remplissage du journal de travail pour le lundi et le mardi	
<b>10.05.2023</b>	19	30 Connexion à la DB / API	4:00	Création de la migration de la base de données et connexion à la base fonctionnelle	

<b>11.05.2023</b>	19	29 Login / Register	5:20	Création d'un enregistrement d'utilisateur	Problème au niveau du modèle User
<b>11.05.2023</b>	19	42 Rédaction de la documentation	1:35	Rédaction de la partie objectif et mise en page	
<b>11.05.2023</b>	19	43 Remplissage du journal de travail	0:15	Remplissage du journal	
<b>15.05.2023</b>	20	29 Login / Register	5:35	Fonctions login et register complètes	
<b>16.05.2023</b>	20	27 Page IMC	4:50	Début sur la page IMC	
<b>16.05.2023</b>	20	52 Autres	0:30	Rencontre avec M. Charmier (expert no.2)	
<b>16.05.2023</b>	20	27 Page IMC	1:50	Avance sur la page IMC	Problème pour insérer des données de la base de données dans le graphique
<b>17.05.2023</b>	20	27 Page IMC	1:20	Page IMC terminée	Problème résolu pour l'insertion de données
<b>17.05.2023</b>	20	42 Rédaction de la documentation	2:25	Ajout du MCD, MLD, des maquettes et rédaction de la partie organisation	
<b>17.05.2023</b>	20	43 Remplissage du journal de travail	0:15	Remplissage journal de travail	
<b>22.05.2023</b>	21	30 Connexion à la DB / API	5:35	Requêtes API fonctionnent à moitié	Problème au niveau de la taille des données récupérées
<b>23.05.2023</b>	21	28 Page données alimentaires journalières	6:55	Pages de données alimentaires journalières et d'ajout d'aliments presque abouties et ajout de la suppression d'utilisateur	
<b>23.05.2023</b>	21	43 Remplissage du journal de travail	0:15	Remplissage journal de travail et envoi mail aux experts	
<b>24.05.2023</b>	21	30 Connexion à la DB / API	0:50	Requêtes API fonctionnent correctement	Correction du bug suite au retour de M. Charmier

<b>24.05.2023</b>	21	42 Rédaction de la documentation	3:10	Avancement de la documentation et modification du journal de travail et du planning pour y ajouter les 5 minutes entre chaque période de 45 minutes	
<b>25.05.2023</b>	21	42 Rédaction de la documentation	4:00	Avancement de la documentation	
<b>25.05.2023</b>	21	28 Page données alimentaires journalières	1:35	Page de données alimentaires journalières complète	
<b>25.05.2023</b>	21	52 Autres	1:35	Page de profil utilisateur complète avec la modification des infos personnelles	
<b>29.05.2023</b>	22	52 Autres	3:00	Avancée de la documentation à la maison	Ecriture de la conclusion, le résumé et du manuel d'installation
<b>30.05.2023</b>	22	42 Rédaction de la documentation	1:35	Avancée de la documentation	
<b>30.05.2023</b>	22	29 Login / Register	1:35	Ajout de l'option mot de passe oublié	
<b>30.05.2023</b>	22	32 Rédaction des tests	0:50	Rédaction des tests à effectuer en fonction de la stratégie de tests	
<b>30.05.2023</b>	22	33 Application des tests	1:35	Application des tests	
<b>30.05.2023</b>	22	42 Rédaction de la documentation	1:35	Avancée de la documentation	
<b>31.05.2023</b>	22	42 Rédaction de la documentation	2:00	Peaufinement de la documentation, impression et envoi aux experts	

## 5.6 Glossaire

Acronyme	Explication
<b>API</b>	Application Programming Interface est un ensemble de composants logiciels facilitant le développement d'application.
<b>CRUD</b>	Acronyme pour « Create Read Update Delete » représentant la relation entre la base de données et l'application.
<b>CSS</b>	Cascading Style Sheets est langage de programmation utile à la mise en page d'un site web.
<b>DB</b>	Database ou base de données
<b>Framework</b>	Ensemble de composants logiciels servant à la programmation d'application.
<b>Front end</b>	Eléments visibles d'une site web
<b>HTML</b>	Hypertext Markup Language est un langage de programmation très utilisé dans le développement web.
<b>IDE</b>	Environnement de développement
<b>JSON</b>	JavaScript Object Notation est un format d'échange de données.
<b>Moodboard</b>	Planche graphique d'ambiance ou d'inspiration.
<b>MVC</b>	Model View Controller est une manière d'organiser la structure d'un projet de développement.
<b>OS</b>	Système d'exploitation
<b>PHP</b>	Langage de programmation principalement utilisé en web.
<b>SGBD(R)</b>	Système de gestion de bases de données (relationnelles).
<b>Sprint</b>	Cycle du projet au cours duquel vont être effectuées des tâches du même style.
<b>URL</b>	Uniform Resource Locator est une adresse web unique.
<b>Use case</b>	Description écrite de la réalisation de tâches utilisée dans le développement logiciel.

## 5.7 Table des illustrations

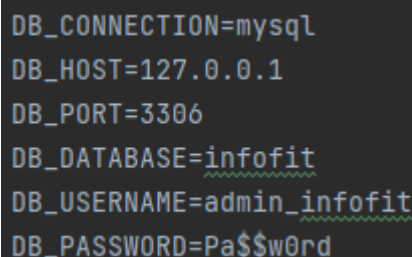
Image 1 Organisation des tâches .....	4
Image 2 Méthode de gestion de projet en cascade .....	5
Image 3 Planification initiale du projet de TPI .....	6
Image 4 MCD de la base de données infofit .....	8
Image 5 MLD de la base de données infofit .....	9
Image 6 Scripte SQL de création de la base de données infofit .....	10
Image 7 Moodboard de l'application .....	11
Image 8 Page d'accueil de la maquette .....	12
Image 9 Page de login de la maquette .....	12
Image 10 Page de register de la maquette .....	13
Image 11 Page profil utilisateur de la maquette .....	13
Image 12 Page de calcul de l'IMC de la maquette .....	14
Image 13 Page alimentation de la maquette .....	14
Image 14 Page d'ajout d'aliment de la maquette .....	15
Image 15 Planification finale du projet de TPI .....	17
Image 16 Capture du dossier de TPI .....	18
Image 17 Dossier du projet InfoFit .....	19
Image 18 Capture d'écran de la fonction store .....	21
Image 19 Première partie capture de la fonction searchFoodstuff .....	22
Image 20 Deuxième partie capture de la fonction searchFoodstuff .....	23
Image 21 Capture de la fonction lineChart .....	24
Image 22 Capture de la page d'accueil de InfoFit .....	25
Image 23 Capture de la page de login de InfoFit .....	25
Image 24 Capture de la page de register de InfoFit .....	26
Image 25 Capture de la page de profil de InfoFit .....	27
Image 26 Capture de la page d'IMC de InfoFit .....	27
Image 27 Capture de la page d'alimentation de InfoFit .....	28
Image 28 Capture de la page d'ajout d'aliment de InfoFit .....	28
Image 29 Première page du cahier des charges .....	34
Image 30 Deuxième page du cahier des charges .....	35
Image 31 Troisième page du cahier des charges .....	36
Image 32 Capture du fichier .env de l'application .....	46

## 5.8 Manuel d'installation

Tout d'abord, ce manuel d'installation explique les étapes nécessaires au lancement du site InfoFit de manière locale et dans un système d'exploitation Windows 10.

Afin de pouvoir ouvrir l'application InfoFit en local, plusieurs installations sont nécessaires. La première installation à faire est la librairie php disponible sur le site <https://www.php.net/>. La deuxième installation à effectuer est Composer (<https://getcomposer.org/>), pour ce faire, ouvrez le fichier php.ini de la librairie php que vous venez d'installer et décommentez la ligne « **extension=php\_fileinfo** », puis enregistrez le fichier. Ensuite, saisissez la commande **composer install** dans une invite de commande.

Une étape importante pour que l'application fonctionne correctement est de créer la base de données localement. Vous pouvez utiliser le SGBDR de votre choix ; j'ai personnellement utilisé Mariadb. Veuillez créer une nouvelle base de données nommée « infofit » et un utilisateur propre à cette base et y possédant tout droits possibles. Exécutez ensuite le script de création de ma base de données disponible sur mon dépôt GitHub ([https://github.com/ThierryKoetschet/TPI\\_Thierry\\_Koetschet](https://github.com/ThierryKoetschet/TPI_Thierry_Koetschet)) sous le nom de **script\_creation\_db.sql**. Enfin, modifiez le fichier **.env.example** du projet InfoFit pour qu'il corresponde aux informations que vous avez saisi lors de la création de la base de données comme l'exemple ci-dessous et renommez-le en **.env**.



```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=infofit
DB_USERNAME=admin_infofit
DB_PASSWORD=Pa$$w0rd
```

Image 32 Capture du fichier .env de l'application

Pour faciliter le lancement de l'application, je conseille l'installation d'un environnement de développement supportant le langage php. Par exemple, Visual Studio Code ou encore PhpStorm fonctionnent parfaitement et j'ai personnellement utilisé dans la réalisation de mon projet l'IDE PhpStorm. Ouvrez donc le projet InfoFit de mon dépôt GitHub dans l'IDE de votre choix ouvrez un terminal. Placez vous à la racine du projet avec un commande **cd** et saisissez ensuite la commande **php artisan serve** afin de lancer l'application de manière locale.