

# Application web de fitness

---

RAPPORT DE TPI

# InfoFit

**Thierry Koetschet**

CHEMIN DU PERREY 22 | 1670 URSY  
THIERRY.KOETSCHET.1998@GMAIL.COM  
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
2.2	Stratégie de test.....	16
2.3	Risques techniques .....	16
2.4	Planification .....	17
2.5	Dossier de conception .....	17
3	Réalisation.....	17
3.1	Dossier de réalisation .....	17
3.2	Description des tests effectués .....	18
3.3	Erreurs restantes .....	19
4	Conclusions .....	19
5	Annexes.....	20
5.1	Cahier des charges.....	20
5.2	Résumé du travail.....	20
5.3	Sources – Bibliographie.....	21
5.4	Journal de travail .....	22
5.5	Glossaire .....	22
5.6	Manuel d'installation .....	22

# 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, sa taille et son genre. 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 ma quête du corps de mes rêves.

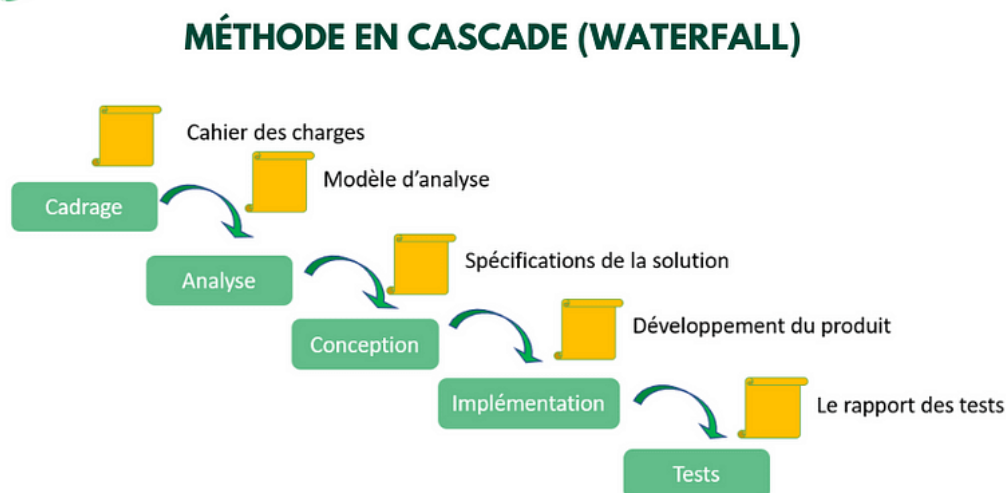
## 1.2 Organisation

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

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.



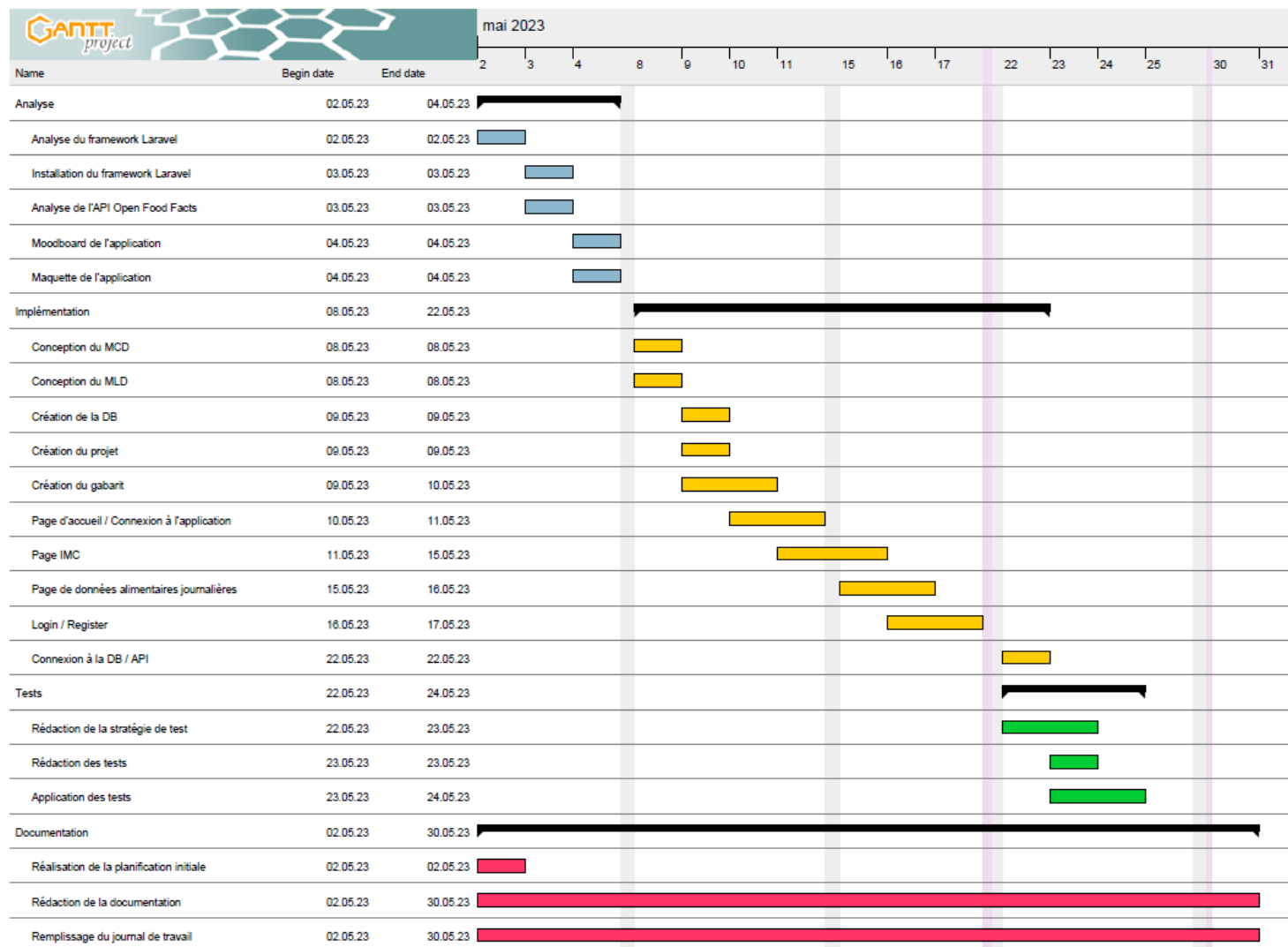
Les cinq étapes de la méthode en cascade correspondent parfaitement à la vision que j'ai pour la réalisation d'un tel projet. En effet, le cadrage en début de projet est primordial afin de structurer notre travail et de ne pas trop s'éparpiller. La planification initiale représente parfaitement cette 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 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



## 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.

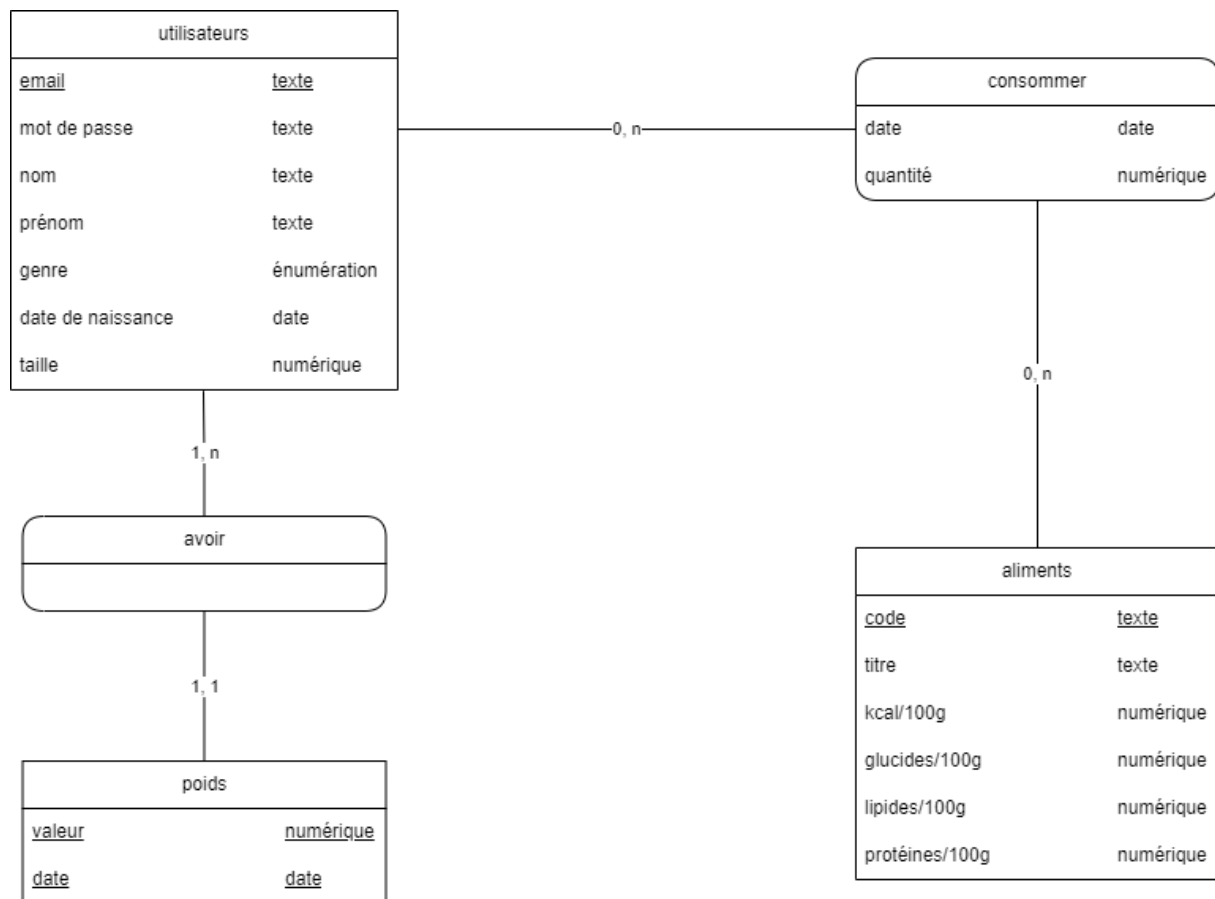
La troisième partie représente toute la réalisation pratique du projet commençant par lister tous les fichiers du dossier de réalisation, puis une description des tests effectués et des erreurs restantes et finalement une énumération des documents fournis à la remise du projet.

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



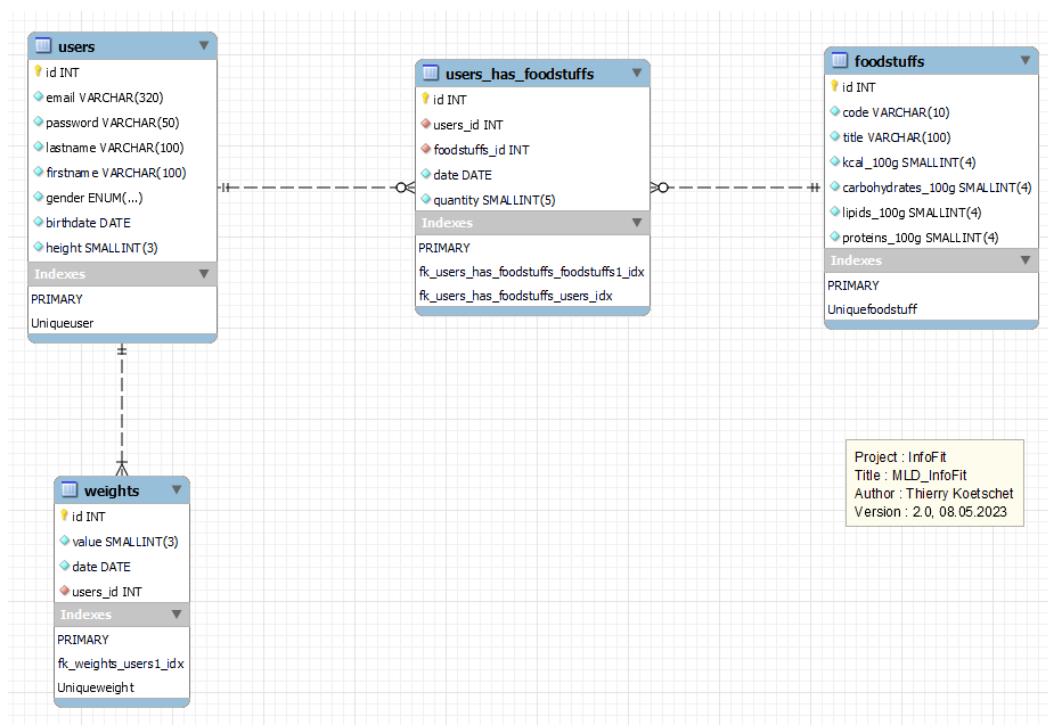
Projet : InfoFit  
 Titre : MCD\_InfoFit  
 Auteur : Thierry Koetschet  
 Version : 2.0 du 08.05.2023

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



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

```

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=11 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=9 DEFAULT CHARSET=utf8mb3;

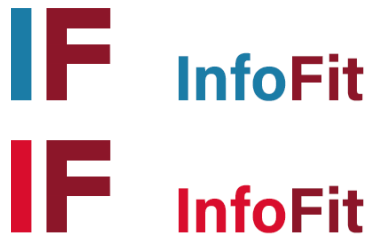
-- Listage de la structure de table infofit. users_has_foodstuffs
CREATE TABLE IF NOT EXISTS `users_has_foodstuffs` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `users_id` int(10) unsigned NOT NULL,
  `foodstuffs_id` int(10) unsigned NOT NULL,
  `date` date NOT NULL,
  `quantity` float NOT NULL DEFAULT 1,
  `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=4 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=9 DEFAULT CHARSET=utf8mb3;

```

Voici ci-dessus le script 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)). Il y a 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.

### 2.1.4 Moodboard

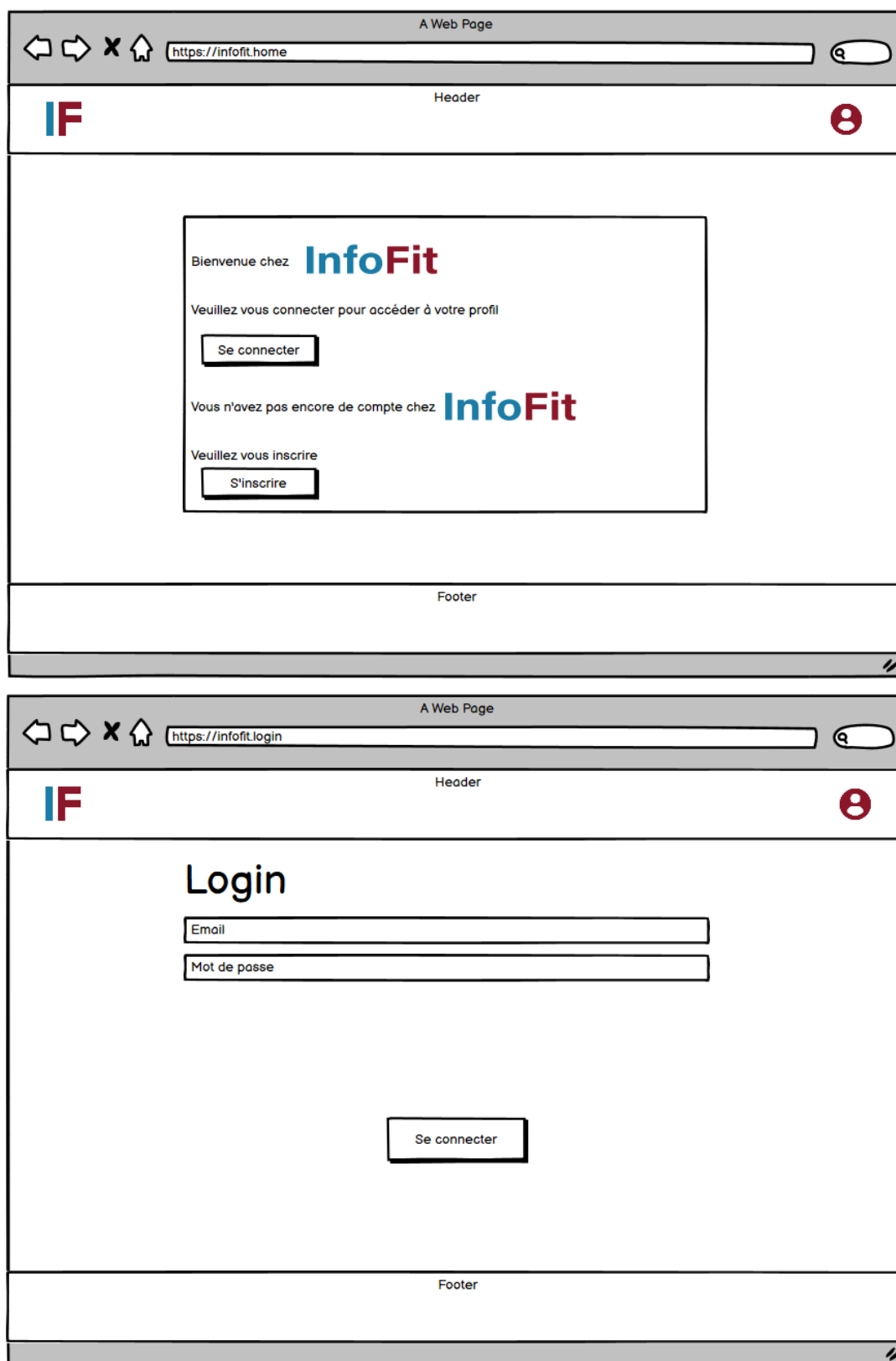


Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Quis ipsum suspendisse ultrices gravida. Risus commodo viverra maecenas accumsan lacus vel facilisis.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Quis ipsum suspendisse ultrices gravida. Risus commodo viverra maecenas accumsan lacus vel facilisis.

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

## 2.1.5 Maquettes



A Web Page

https://infofit.register

IF

Header

## Register

☐ Homme ☐ Femme

Nom

Prénom

Email

Mot de passe

Confirmation du mot de passe

Date de naissance

Taille cm Poids kg

S'inscrire

Footer

A Web Page

https://infofit.profil

IF

Header

Profil IMC Alimentation

Prénom Nom

## Profil

Nom

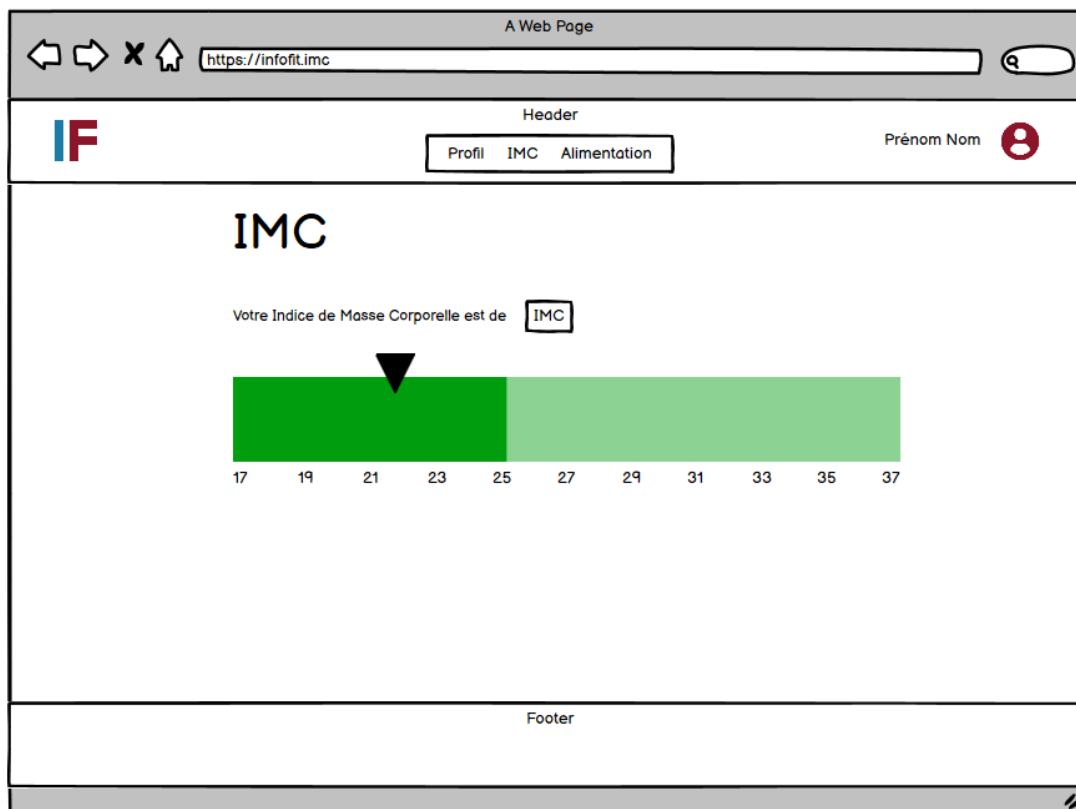
Prénom

Date de naissance

Taille cm Poids kg

Modifier

Footer



A Web Page

https://infofit.alimentation

Header

IF

Profil IMC Alimentation

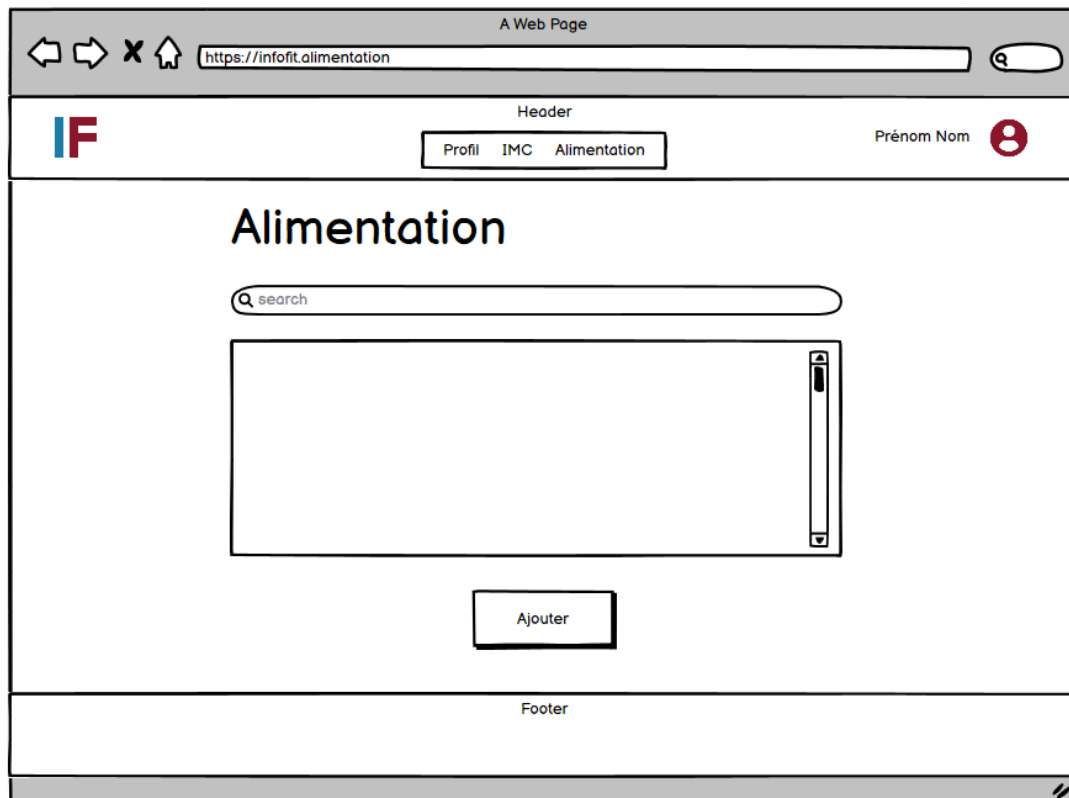
Prénom Nom

## Alimentation

May 2023						
S	M	T	W	T	F	S
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

	Calories kcal	Glucides g	Lipides g	Protéines g
Petit déjeuner				
<a href="#">Ajouter un aliment</a>				
Banane	50	15	0	1
Dîner				
<a href="#">Ajouter un aliment</a>				
Poulet 100g	200	1	3	20
Riz 60g	300	30	2	4
Souper				
<a href="#">Ajouter un aliment</a>				
Total	550	46	5	24

Footer



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.

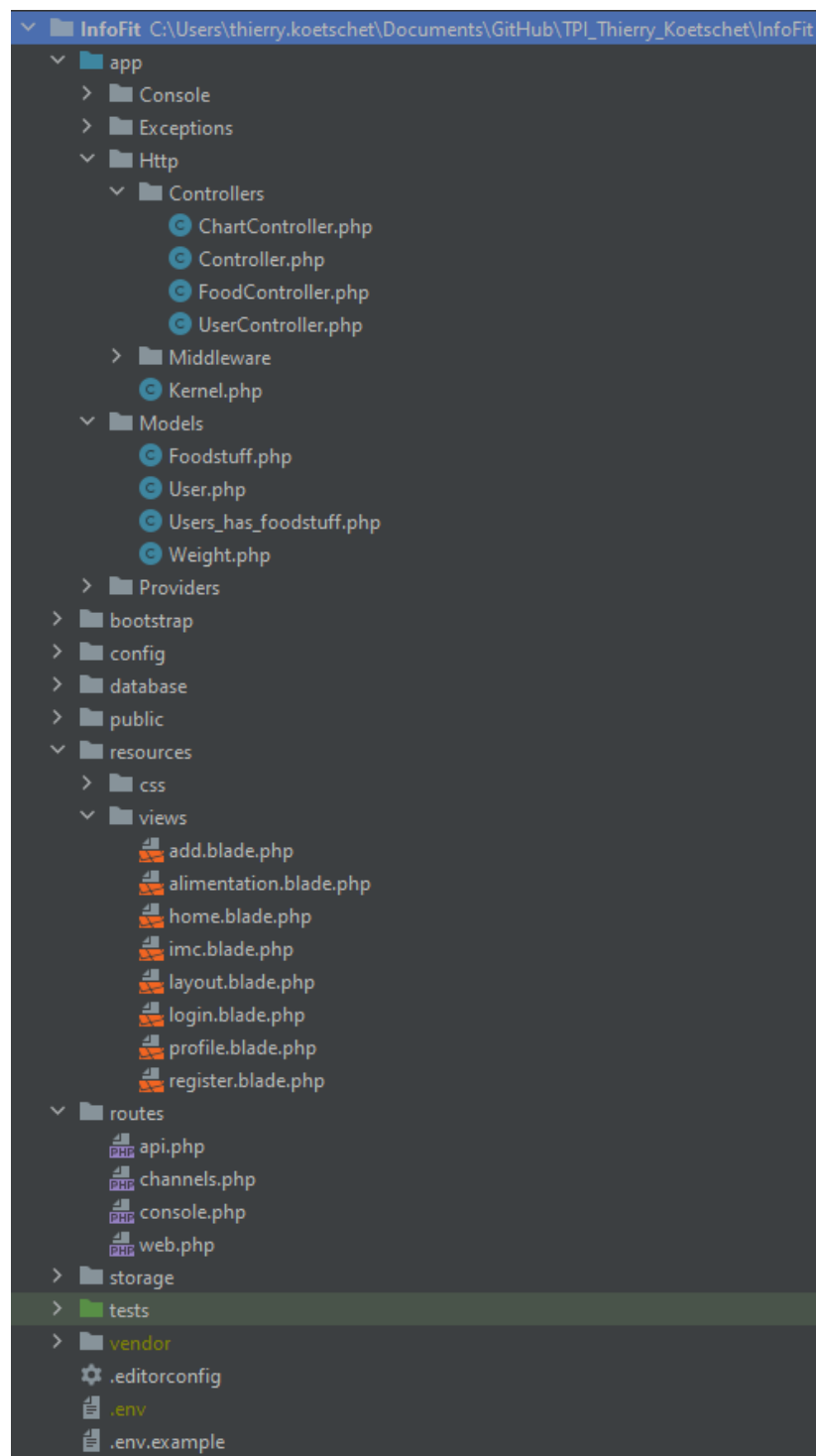


## 2.4 Planification

## 2.5 Dossier de conception

# 3 Réalisation

## 3.1 Dossier de réalisation


















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.

### 3.2 Description des tests effectués

Date	Test	Résultat	Validation
Tests unitaires			
28.03.2023			
28.03.2023			
28.03.2023			
28.03.2023			
28.03.2023			
28.03.2023			
Tests d'intégration			

28.03.2023			
<b>Tests fonctionnels</b>			
28.03.2023			
28.03.2023			
28.03.2023			
28.03.2023			
28.03.2023			
28.03.2023			
28.03.2023			
28.03.2023			

### 3.3 Erreurs restantes

### 3.4 Liste des documents fournis

## 4 Conclusions

## 5 Annexes

### 5.1 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

## 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

## 5.6 Glossaire

API	Application Programming Interface est un ensemble de composants logiciels facilitant le développement d'application.
CRUD	Acronyme pour « Create Read Update Delete » représentant la relation entre la base de données et l'application.
CSS	Cascading Style Sheets est langage de programmation utile à la mise en page d'un site web.
DB	Database ou base de données
Framework	Ensemble de composants logiciels servant à la programmation d'application.
Front end	Eléments visibles d'une site web
HTML	Hypertext Markup Language est un langage de programmation très utilisé dans le développement web.
IDE	Environnement de développement
Moodboard	Planche graphique d'ambiance ou d'inspiration.
MVC	Model View Controller est une manière d'organiser la structure d'un projet de développement.
OS	Système d'exploitation
PHP	Langage de programmation principalement utilisé en web.
Use case	Description écrite de la réalisation de tâches utilisée dans le développement logiciel.

## 5.7 Manuel d'installation