Planning web



(Une image originale représentant le projet)

Sacha Grenier – Cin4b

ETML - Lausanne

Durée : 105 heures

Chef de projet : Gruaz Gilbert

Experts : Mottier André, Borboen Nicolas

Table des matières

[1 Spécifications 4](#_Toc482166682)

[1.1 Titre 4](#_Toc482166683)

[1.2 Description 4](#_Toc482166684)

[1.3 Matériel et logiciels à disposition 4](#_Toc482166685)

[1.4 Prérequis 4](#_Toc482166686)

[1.5 Cahier des charges 4](#_Toc482166687)

[2 Planification Initiale 5](#_Toc482166688)

[3 Analyse 6](#_Toc482166689)

[3.1 Opportunités 6](#_Toc482166690)

[3.2 Document d’analyse et conception 6](#_Toc482166691)

[3.2.1 Page principale 7](#_Toc482166692)

[3.2.2 La page Planning des utilisateurs 9](#_Toc482166693)

[3.2.3 Page utilisateurs 9](#_Toc482166694)

[3.3 Conception des tests 12](#_Toc482166695)

[3.4 Planification détaillée 12](#_Toc482166696)

[4 Réalisation 13](#_Toc482166697)

[4.1 Dossier de Réalisation 13](#_Toc482166698)

[4.1.1 Mise en place de l’environnement 13](#_Toc482166699)

[4.2 Modifications 13](#_Toc482166700)

[5 Tests 13](#_Toc482166701)

[5.1 Dossier des tests 13](#_Toc482166702)

[6 Conclusion 13](#_Toc482166703)

[6.1 Bilan des fonctionnalités demandées 13](#_Toc482166704)

[6.2 Bilan de la planification 14](#_Toc482166705)

[6.3 Bilan personnel 14](#_Toc482166706)

[7 Divers 14](#_Toc482166707)

[7.1 Journal de travail 14](#_Toc482166708)

[7.2 Bibliographie 14](#_Toc482166709)

[7.3 Webographie 14](#_Toc482166710)

[8 Annexes 14](#_Toc482166711)

[8.1 Script SQL obtenu avec l’application jMerise 15](#_Toc482166712)

# Spécifications

## Titre

Plateforme de planification hebdomadaire d’ateliers et de travailleurs

## Description

Inspiré d’une situation réelle vécue au GRAAP-F (fondation qui s’occupe de travailleurs AI), l’idée de ce projet est de réaliser une plateforme Web qui permet, pour un chef d’ateliers, de faire une planification hebdomadaire de plusieurs ateliers dans lesquels travaillent des employés au bénéfice de l’AI.

Une réalisation générique de ce projet permettra d'utiliser la plateforme pour planifier des classes avec des cours modulaires, ou des élèves inscrits dans des activités de camp de sport, etc.

## Matériel et logiciels à disposition

1 ordinateur standard ETML, avec la structure habituelle

A mettre en œuvre par le candidat :

* Une architecture WAMP (Xampp),
* Un framework PHP Laravel
* Un dépôt GIT

## Prérequis

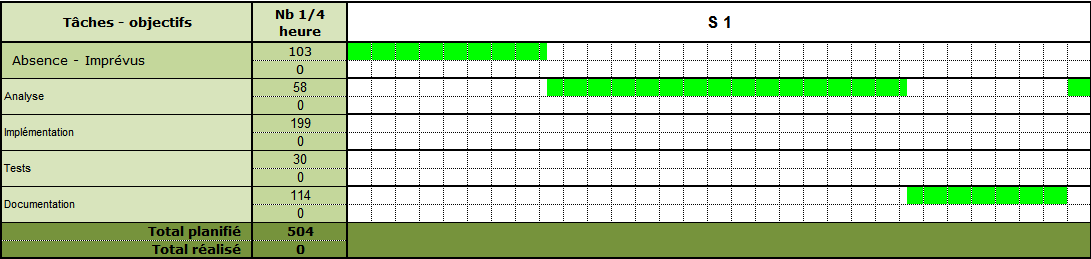
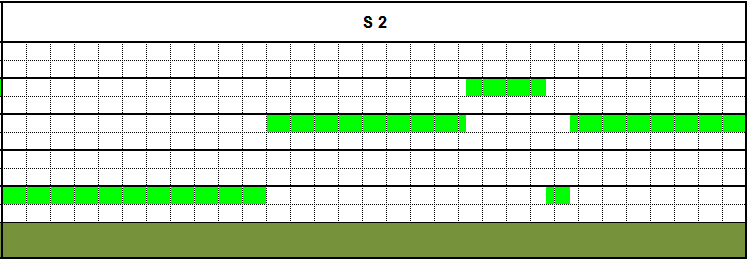
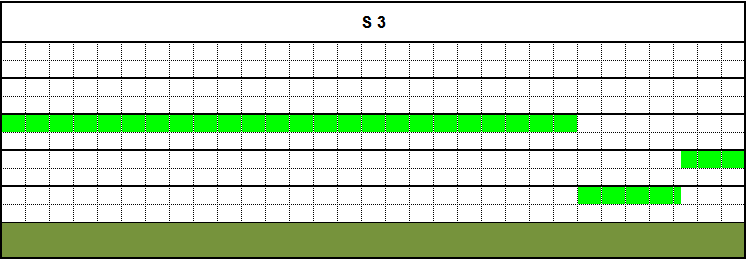
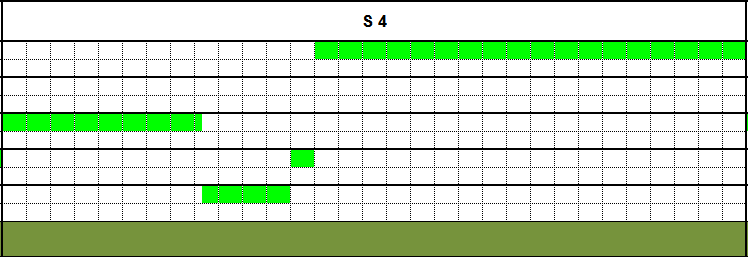
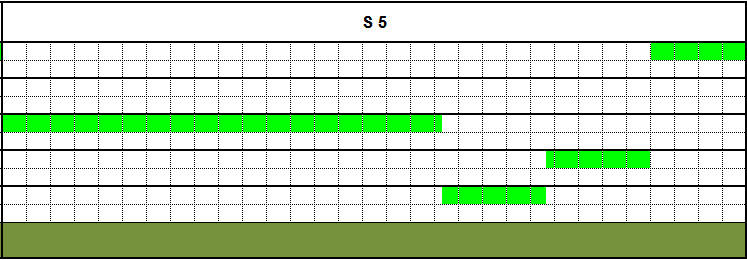
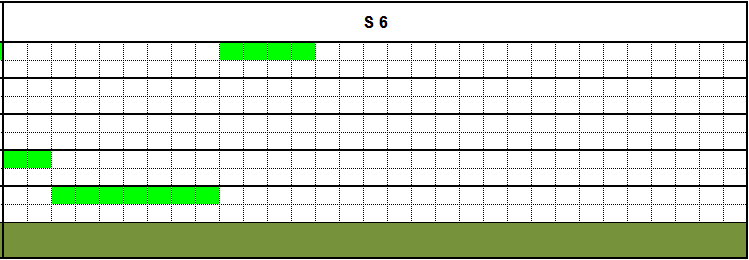
Avoir suivi les modules ICH à l'ETML, les projets et effectué des stages

## Cahier des charges

Le document fourni par le chef de projet fait foi. Il doit être mis en annexe 🡺 Lien sur CDC

# Planification Initiale

Voici le Gantt que j’ai obtenu en faisant ma planification initiale :



# Analyse

## Opportunités

La création de ce site web va me permettre de :

* M’améliorer avec le framework Laravel
* M’améliorer en Javascript, jQuery et AJAX
* Créer une grille de planning et la rendre dynamique

Les difficultés seront :

* Le temps à disposition
* Créer la grille sans utiliser de librairie

Ainsi que les solutions possibles

Si les spécifications de départ ne laissent pas de doutes sur la manière de réaliser un projet, ce chapitre ne fera que renvoyer le lecteur aux spécifications.

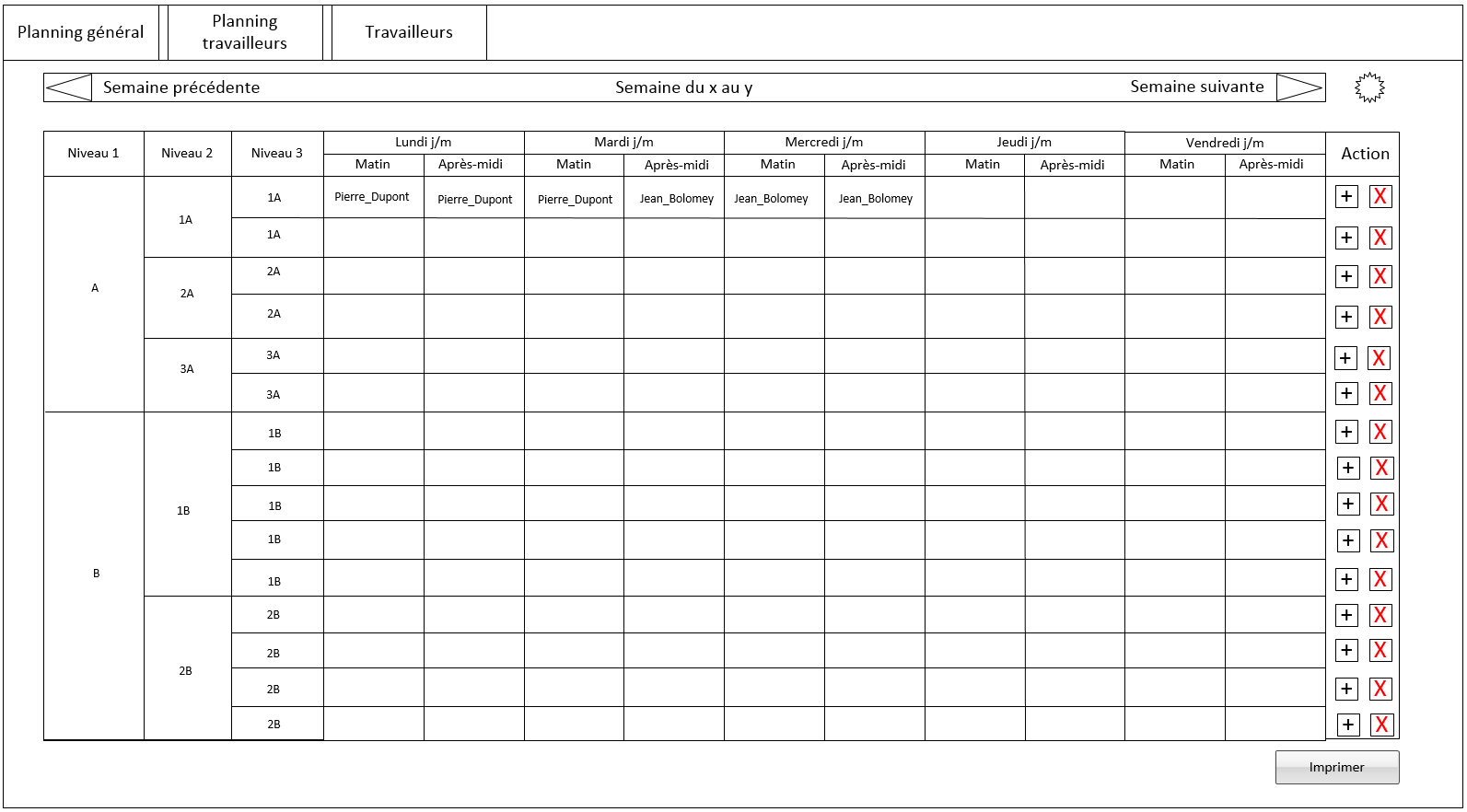
## Document d’analyse et conception

Le but du site est de créer un planning générique qui pourra être utilisé dans plusieurs domaines. Pour me mettre en situation, j’ai choisi le domaine donné dans le cahier de charges qui est le GRAAP-F. L’objectif est donc d’attribuer des tâches à des utilisateurs et de les implémenter dans un planning sous forme de grille hebdomadaire. Il doit être possible d’attribuer des tâches à des utilisateurs lors de la semaine courante, mais aussi lors des semaines suivantes. Les semaines précédentes seront affichées mais ne pourront pas être modifiées, elles feront office « d’historique ».

En ce qui concerne les pages du site, j’ai créé des maquettes concernant à chaque page du site.

### Page principale

Cette page affichera le planning général, avec diverses possibilités d’ajout, suppression et modification de données.



1

2

3

4

5

6

7

8

Description :

1) Menu d’entête, comportant les liens vers les 3 pages du site.

2)Barre de navigation indiquant la semaine actuellement affichée dans la grille, avec deux boutons en forme de triangle pour se déplacer entre les semaines.

3)Boutons + (ajouter) et x (supprimer) qui affecteront les lignes. Normalement ces boutons seront cachés et visibles uniquement lorsque l’utilisateur passe sa souris sur la ligne souhaitée. Le bouton + ajoutera une ligne en dessous et prendra exactement la même catégorie et la même tâche. Le bouton X supprimera la ligne du tableau, avec confirmation de l’utilisateur.

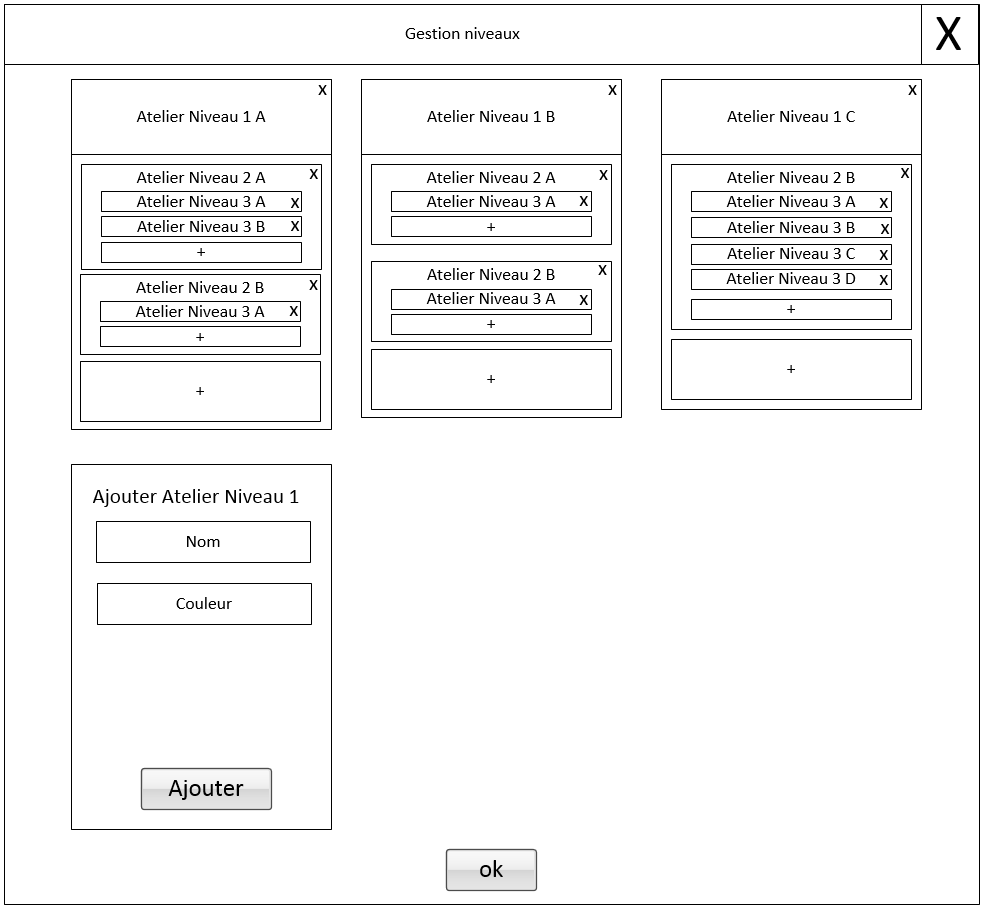
4)Colonne correspondant au Mardi, qui se sépare en Matin et Après-midi.

5)Les catégories et tâches qui en héritent seront affichées ici. Au moins une ligne par tâche sera toujours affichée.

6) Exemple de cellule, correspondant au mardi après-midi. Ici, l’utilisateur pourra cliquer pour afficher champ de recherche, qui une fois complété de quelques lettres proposera les différents utilisateurs correspondants (Autocomplete).

7) Bouton d’impression pour imprimer (imprimer ou enregistrer sous format PDF) la semaine affichée.

8)Bouton qui ouvrira un menu sous forme de pop-up, et qui servira à modifier les catégories et les tâches.

Menu : 

1

4

3

2

1

Description :

1)Boutons pour fermer la pop-up

2)Formulaire pour ajouter un atelier de niveau 1

3)Bouton + pour ajouter un atelier de niveau 2. Une fois le bouton cliqué, un champ texte apparaitra et appuyer sur la touche entrée ajoutera l’atelier.

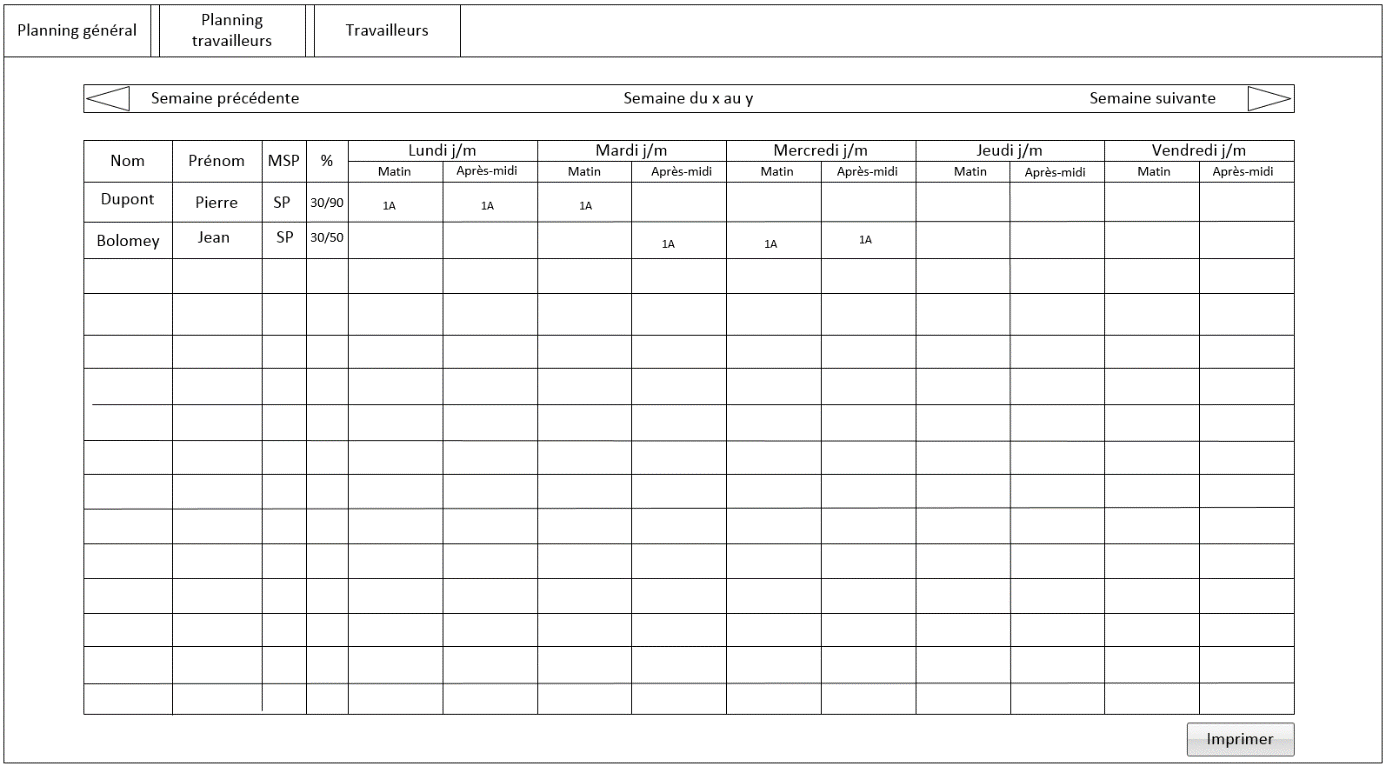
4)Idem que pour le point 3) mais pour ajouter un atelier de niveau 3

Peu importe les niveaux, tous les ateliers seront supprimables à l’aide de la croix qui se situe à droite de chaque atelier. Bien évidemment, si un atelier de niveau 2 est supprimé, tous les ateliers de niveau 3 associés le seront aussi.

### La page Planning des utilisateurs

Cette page sera uniquement en lecture et rien de sera modifiable. Avec les boutons en forme de triangle, il sera possible de se déplacer à travers les semaines et d’y afficher le planning. Elle représente la même grille que pour la page principale mais sous une autre forme. Voici le modèle :

2



4

3

5

1

Description :

1) Barre de navigation indiquant la semaine actuellement affichée dans la grille, avec deux boutons en forme de triangle pour se déplacer entre les semaines.

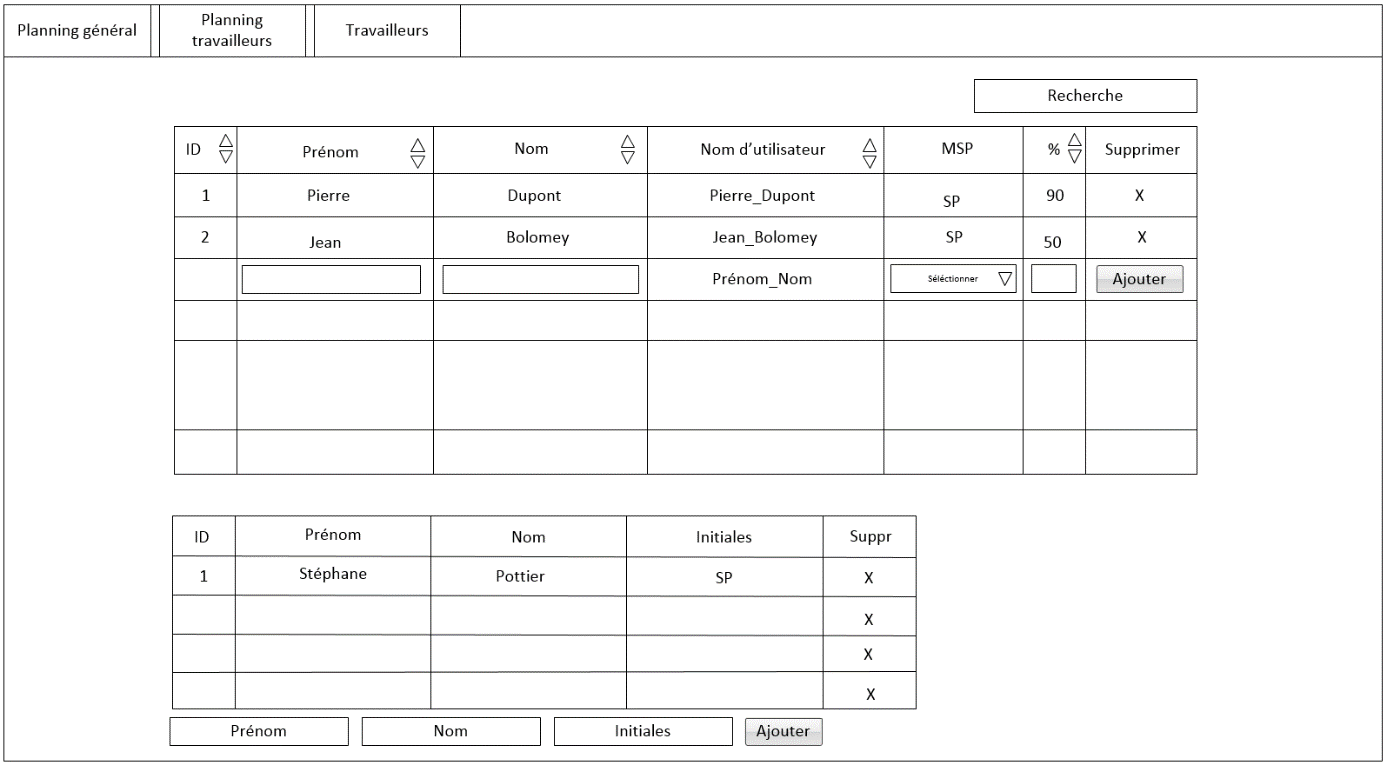
2)Colonnes indiquant le nom et le prénom des utilisateurs

3)Colonne MSP, indiquant le Maitre Sociaux Professionnel

4)Collonne %, indiquant le pourcentage effectif/prévu   
5) Bouton d’impression pour imprimer (imprimer ou enregistrer sous format PDF) la semaine affichée.

### Page travailleurs

Cette page affichera la liste des travailleurs sous forme de tableau. Pour ce faire, je vais utiliser la libraire DataTables (<https://datatables.net/>) qui propose un tableau dynamique où il est possible de trier les données et qui utilise une pagination.



3

2

1

Description :

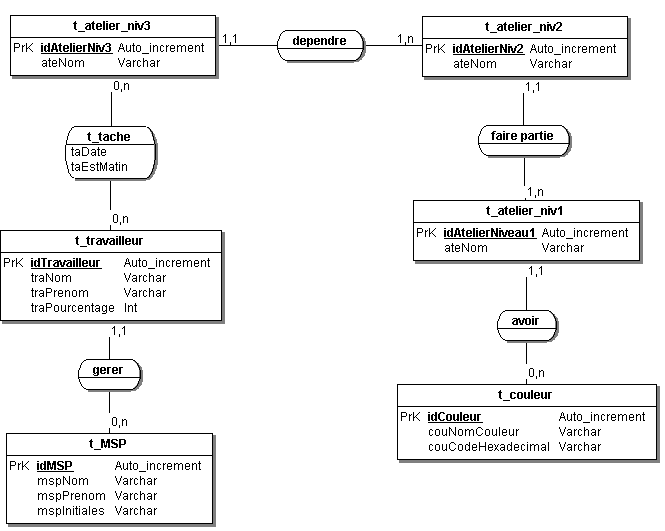
1)Tableau dynamique comportant la liste des travailleurs généré par DataTables. La dernière ligne est sous forme de formulaire pour ajouter un travailleur dans la liste.

2)Liste des Maitres sociaux professionnels, sous forme de tableau simple, sans utiliser DataTables.

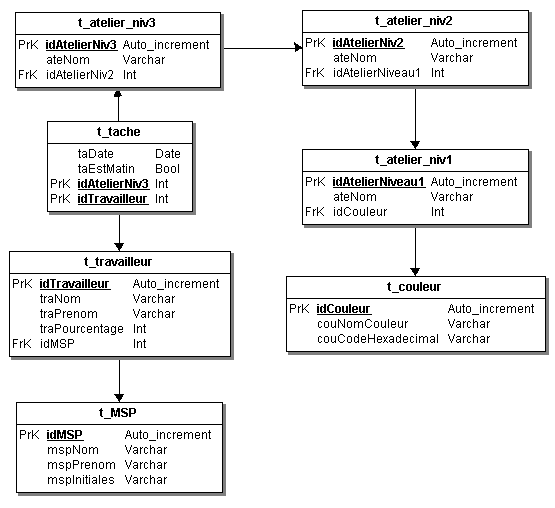
3)Formulaire pour ajouter un MSP à la liste

### Base de données

Pour la base de données, je vais utiliser MySQL. Laravel fourni quelques outils pour aider à la construire, notamment un système de migrations que je vais utiliser pour créer mes tables. Le MCD, MLD et le script SQL suivants ont été réalisés à l’aide de l’application jMerise, que vous pouvez consulter ici : <http://www.jfreesoft.com/JMerise/>. Cet outil permet de créer un MCD qui se transforme ensuite en MLD et en un MPD, sous forme de script SQL. J’ai donc dû nommer mes tables spécifiquement pour que le MLD construit par jMerise corresponde aux normes ETML.

On y retrouve donc 5 tables basiques et une table intermédiaire.   
Voici le MCD : 

Etant donné que les cardinalités entre la table « t\_atelier\_niv3 » et « t\_travailleur » sont de 0,N, j’ai donc prévu la table intermédiaire « t\_tache ». Elle correspond à un atelier X assigné à un travailleur Y.

S’en suit le MLD : 

Suite à ceci l’application jMerise m’a proposé un script SQL pour créer la base de données. Vous le trouverez dans les annexes, ou en cliquant ici : [Script SQL](#_Script_SQL_obtenu)

Et enfin le MPD, dans lequel j’ai modifié le nom des champs pour respecter les conventions du framework Laravel (snake case) et que vous pouvez retrouver ici : <https://laravel.com/docs/5.4/eloquent#eloquent-model-conventions>



## Conception des tests

Ce paragraphe permet de spécifier la stratégie de test qui sera menée au point 5.1

Qui, quand, avec quelles données, dans quel ordre, etc.

Avec quels matériels, quels services, etc.

## Planification détaillée

A ce stade, après l’analyse complète du projet, un planning détaillé et complet (avec tâches, sous-tâches, dépendances, durée, …) peut être finalisé.

Le planning détaillé doit s’inscrire dans le planning initial. Il faut que l’on puisse situer cette planification détaillée par rapport à la planification initiale.

# Réalisation

## Dossier de Réalisation

### Mise en place de l’environnement

Cette partie permet de reproduire ou reprendre le projet par un tiers.

Pour chaque étape, il faut décrire sa mise en œuvre. Typiquement :

Versions des outils logiciels utilisés (OS, applications, pilotes, librairies, etc.)

Configurations spéciales des outils (Equipements, PC, machines, outillage, etc.)

Code source commenté des éléments logiciels développés.

Modèle physique d’une base de données.

Arborescences des documents produits.

Schémas, plans d’adressages, plan de nommage, etc.

Il faut décrire le parcours de réalisation et justifier les choix.

## Modifications

Historique des modifications demandées (ou nécessaires) aux spécifications détaillées.

Date, raison, description, etc.

# Tests

## Dossier des tests

On dresse le bilan des tests effectués (qui, quand, avec quelles données…) sous forme de procédure. Lorsque cela est possible, fournir un tableau des tests effectués avec les résultats attendus et obtenus, ainsi que les actions à entreprendre en conséquence (et une estimation de leur durée).

Si des tests prévus dans la stratégie n'ont pas pu être effectués :

raison, décisions, etc.

Liste des bugs répertoriés avec la date de découverte et leur état:

Corrigé, date de correction, corrigé par, etc.

# Conclusion

## Bilan des fonctionnalités demandées

Il s’agit de reprendre point par point les fonctionnalités décrites dans les spécifications de départ et de définir si elles sont atteintes ou pas, et pourquoi.

Si ce n’est pas le cas, estimer en « % » ou en « temps supplémentaire » le travail qu’il reste à accomplir pour terminer le tout.

## Bilan de la planification

Distinguer et expliquer les tâches qui ont généré des retards ou de l'avance dans la gestion du projet. Indiquer les différences entre les planifications initiales et détaillées avec le journal de travail.

## Bilan personnel

Si c’était à refaire:

Qu’est-ce qu’il faudrait garder ? Les plus et les moins ?

Qu’est-ce qu’il faudrait gérer, réaliser ou traiter différemment ?

Qu’est que ce projet m’a appris ?

Suite à donner, améliorations souhaitables, …

Remerciements, signature, etc.

# Divers

## Journal de travail

Date, activité (description qui permet de reproduire le cheminement du projet), durée, liens et références sur des documents externes. Lorsqu’une activité de recherches a été entreprise, il convient d’énumérer ce qui a été trouvé, avec les références.

## Bibliographie

Références des livres, revues et publications utilisés durant le projet.

## Webographie

Références des sites Internet consultés durant le projet.

# Annexes

Cahier des charges

Listing du code source (partiel ou, plus rarement complet)

Guide(s) d’utilisation et/ou guide de l’administrateur

Etat ou « dump » de la configuration des équipements (routeur, switch, robot, etc.).

Extraits de catalogue, documentation de fabricant, etc.

## Script SQL obtenu avec l’application jMerise

#------------------------------------------------------------

# Script MySQL.

#------------------------------------------------------------

#------------------------------------------------------------

# Table: t\_atelier\_niv3

#------------------------------------------------------------

CREATE TABLE t\_atelier\_niv3(

idAtelierNiv3 int (11) Auto\_increment NOT NULL ,

ateNom Varchar (25) NOT NULL ,

idAtelierNiv2 Int NOT NULL ,

PRIMARY KEY (idAtelierNiv3 )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: t\_atelier\_niv2

#------------------------------------------------------------

CREATE TABLE t\_atelier\_niv2(

idAtelierNiv2 int (11) Auto\_increment NOT NULL ,

ateNom Varchar (25) NOT NULL ,

idAtelierNiveau1 Int NOT NULL ,

PRIMARY KEY (idAtelierNiv2 )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: t\_travailleur

#------------------------------------------------------------

CREATE TABLE t\_travailleur(

idTravailleur int (11) Auto\_increment NOT NULL ,

traNom Varchar (25) NOT NULL ,

traPrenom Varchar (25) NOT NULL ,

traPourcentage Int NOT NULL ,

idMSP Int NOT NULL ,

PRIMARY KEY (idTravailleur )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: t\_atelier\_niv1

#------------------------------------------------------------

CREATE TABLE t\_atelier\_niv1(

idAtelierNiveau1 int (11) Auto\_increment NOT NULL ,

ateNom Varchar (25) NOT NULL ,

idCouleur Int NOT NULL ,

PRIMARY KEY (idAtelierNiveau1 )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: t\_couleur

#------------------------------------------------------------

CREATE TABLE t\_couleur(

idCouleur int (11) Auto\_increment NOT NULL ,

couNomCouleur Varchar (25) NOT NULL ,

couCodeHexadecimal Varchar (40) NOT NULL ,

PRIMARY KEY (idCouleur )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: t\_MSP

#------------------------------------------------------------

CREATE TABLE t\_MSP(

idMSP int (11) Auto\_increment NOT NULL ,

mspNom Varchar (25) NOT NULL ,

mspPrenom Varchar (25) NOT NULL ,

mspInitiales Varchar (4) NOT NULL ,

PRIMARY KEY (idMSP )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: t\_tache

#------------------------------------------------------------

CREATE TABLE t\_tache(

taDate Date ,

taEstMatin Bool ,

idAtelierNiv3 Int NOT NULL ,

idTravailleur Int NOT NULL ,

PRIMARY KEY (idAtelierNiv3 ,idTravailleur )

)ENGINE=InnoDB;

ALTER TABLE t\_atelier\_niv3 ADD CONSTRAINT FK\_t\_atelier\_niv3\_idAtelierNiv2 FOREIGN KEY (idAtelierNiv2) REFERENCES t\_atelier\_niv2(idAtelierNiv2);

ALTER TABLE t\_atelier\_niv2 ADD CONSTRAINT FK\_t\_atelier\_niv2\_idAtelierNiveau1 FOREIGN KEY (idAtelierNiveau1) REFERENCES t\_atelier\_niv1(idAtelierNiveau1);

ALTER TABLE t\_travailleur ADD CONSTRAINT FK\_t\_travailleur\_idMSP FOREIGN KEY (idMSP) REFERENCES t\_MSP(idMSP);

ALTER TABLE t\_atelier\_niv1 ADD CONSTRAINT FK\_t\_atelier\_niv1\_idCouleur FOREIGN KEY (idCouleur) REFERENCES t\_couleur(idCouleur);

ALTER TABLE t\_tache ADD CONSTRAINT FK\_t\_tache\_idAtelierNiv3 FOREIGN KEY (idAtelierNiv3) REFERENCES t\_atelier\_niv3(idAtelierNiv3);

ALTER TABLE t\_tache ADD CONSTRAINT FK\_t\_tache\_idTravailleur FOREIGN KEY (idTravailleur) REFERENCES t\_travailleur(idTravailleur);