Boubacar Sow – Marie Anne Poty

boubacar.sow@uha.fr – marie-anne.poty@uha.fr

Dossier de Spécification du Logiciel

Projet Méthodes Agiles

Table des matières

[1. Objectifs du projet 2](#_Toc505260579)

[2. Fonctionnalités principales 2](#_Toc505260580)

[3. Futurs utilisateurs (& Use Case) 3](#_Toc505260581)

[4. Données à stocker 3](#_Toc505260582)

[5. Définition des IHM (Maquettes) 4](#_Toc505260583)

[6. Contraintes générales 7](#_Toc505260584)

[6.1. Contraintes liées au projet 7](#_Toc505260585)

[6.2. Contraintes matérielles 7](#_Toc505260586)

[7. Temps de réponse / Contraintes mémoire 7](#_Toc505260587)

[8. Fonctionnalités détaillées 7](#_Toc505260588)

[8.1. Fonctionnalités de base 7](#_Toc505260589)

[8.2. Fonctionnalités administrateur 8](#_Toc505260590)

[9. Délais 10](#_Toc505260591)

[Annexes 11](#_Toc505260592)

[Annexe 1 : Interfaces Homme-Machine 11](#_Toc505260593)

[Annexe 2 : Diagramme de Gantt 22](#_Toc505260594)

# Objectifs du projet

L’objectif est développer une application de gestion des cours sur une année scolaire, incluant l’allocation des ressources nécessaires à la tenue d’un cours.

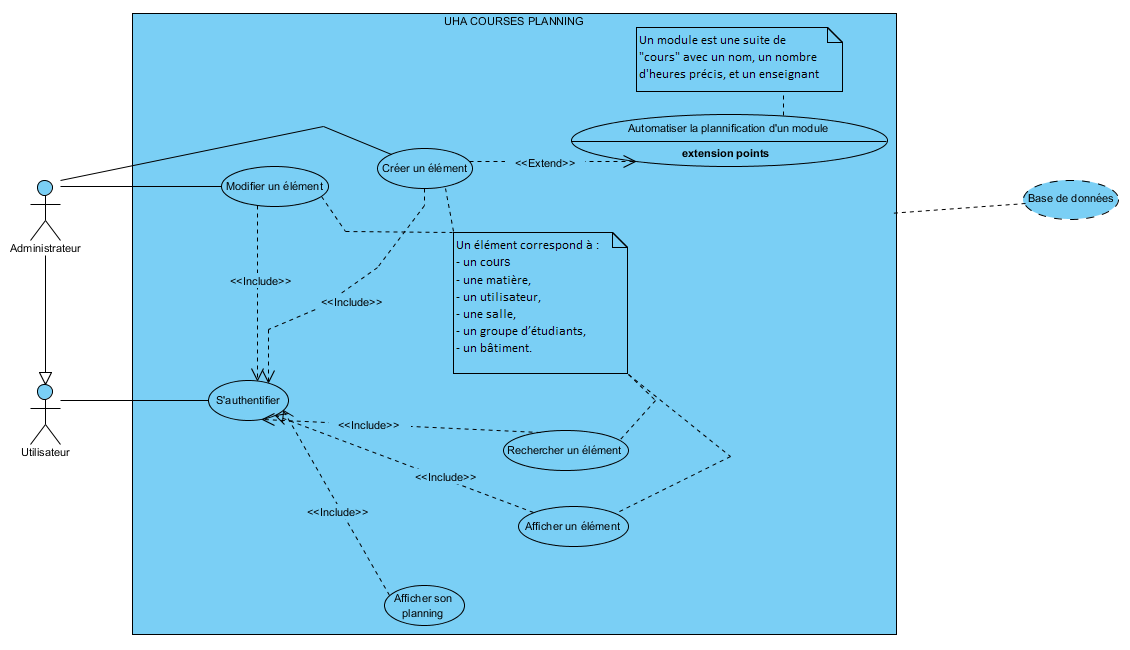
# Fonctionnalités principales

L’application doit pouvoir gérer plusieurs aspects :

* Cours : ajout, modification, suppression, recherche et affichage des cours
* Matière : ajout, modification, suppression, recherche et affichage des matières
* Utilisateur : ajout, modification, suppression, recherche et affichage des utilisateurs, qui peuvent être soit administrateurs, soit simple utilisateurs (enseignant/étudiant)
* Salle : ajout, modification, suppression, recherche et affichage des salles
* Bâtiment : ajout, modification, suppression, recherche et affichage des bâtiments
* Groupe d’étudiants : ajout, modification, suppression, recherche et affichage des groupes d’étudiants
* Planifier un cours (enseignant, salle, date, heure, groupe d’étudiants)
* Afficher le planning d’une semaine pour
  + Un enseignant
  + Un étudiant
* Planifier un module automatiquement en fonction du nombre d’heures et du découpage, de l’enseignant, de la date de début et des groupes d’étudiants concernés.

# Futurs utilisateurs (& Use Case)

L’application s’adresse à des utilisateurs lambdas, aux étudiants, aux enseignants et aux personnes de l’administration (administrateurs). Cependant, seuls deux groupes sont à distinguer, les administrateurs et les utilisateurs, car les utilisateurs lambdas, les étudiants et les enseignants ont accès aux mêmes fonctionnalités.



# Données à stocker

L’application devra disposer d’un module de connexion. Ainsi les données liées aux utilisateurs devront être stockées.

De plus, l’application doit gérer les cours, les matières, les salles, les bâtiments et les groupes d’étudiants. Ces données devront également être stockées dans la base de données.

Les spécifications liées aux données sont expliquées dans le Dossier de Conception Préliminaire.

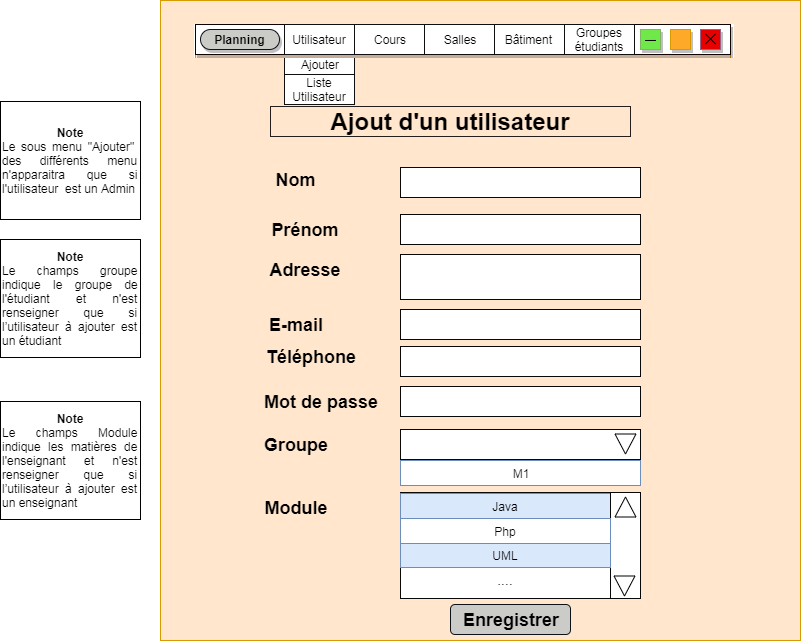
# Définition des IHM (Maquettes)

Les interfaces utilisateur sont définies comme suit :

* La fenêtre d’authentification :



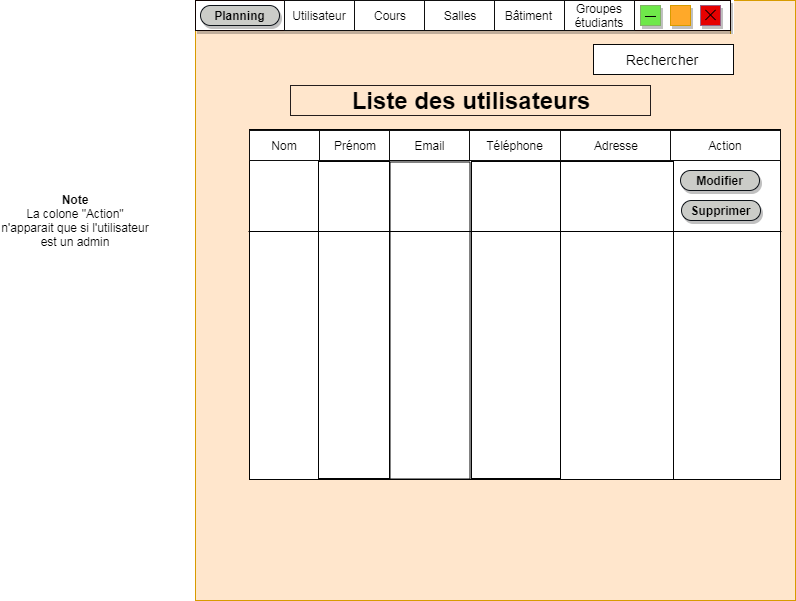
* Les interfaces d’ajout, de modification, de suppression, ainsi que la liste des éléments des différents modules ont la même présentation. Seuls les champs diffèrent.
  + Interface d’ajout d’un utilisateur



* + Interface de modification d’un utilisateur



* + Interface affichage des utilisateurs



Le reste des IHM se trouve en [Annexe 1](#_Annexe_1_:) de ce document.

# Contraintes générales

## Contraintes liées au projet

Au niveau des horaires, un emploi du temps peut démarrer au plus tôt à 8H et terminer à 20H au plus tard, à l’exception du Samedi (12H). Les cours peuvent avoir lieu du Lundi au Samedi.

Les enseignants ne peuvent être disponibles qu’à certain moment. Il faut voir leurs disponibilités avant d’ajouter un cours.

Les salles doivent pouvoir accueillir le nombre d’étudiants dans le(s) groupe(s) associés au cours. Elles doivent également ne pas avoir de cours pour être affectées.

Un cours ne peut pas avoir lieu à deux endroits différents au même moment. De même, il ne peut pas avoir lieu avec deux enseignants différents. En revanche, il peut avoir lieu avec deux groupes d’étudiants (ex : M1 MIAGE + M1 IMR).

## Contraintes matérielles

Le projet devra être fait sous Java, avec l’IDE de votre choix, avec une base de données en MySQL. L’utilisation de PhPMyAdmin sera nécessaire pour la gestion de celle-ci.

# Temps de réponse / Contraintes mémoire

L’application devra fournir une réponse lors d’une requête de l’utilisateur dans un délai de 500 ms. Pour que l’application puisse tourner normalement sur une machine d’un utilisateur, la mémoire maximale utilisée doit être de 200 mo.

# Fonctionnalités détaillées

## Fonctionnalités de base

* Authentification : pour avoir accès à l’application, chaque utilisateur devra s’authentifier. Pour se faire, il devra renseigner un login (email) ainsi qu’un mot de passe.
* Affichage d’un élément (cours, matière, utilisateur, salle, groupe d’étudiants, bâtiment) : en cliquant sur un bouton relatif à un élément, l’utilisateur affichera les caractéristiques de celui-ci. Par exemple, en cliquant sur le bouton « liste des enseignants », l’utilisateur pourra afficher la liste des enseignants.
  + S’il est administrateur, il verra à côté de celle-ci des boutons correspondants respectivement à la modification et à la suppression des éléments.
* Recherche d’un élément : l’utilisateur aura la possibilité de rechercher un élément en renseignant dans le champ correspondant l’objet recherché. Par exemple, s’il recherche un cours, il faudra spécifier soit la date, soit le nom de la matière. S’il recherche un utilisateur, il pourra spécifier le nom ou l’adresse email, etc.
* Affichage d’un planning pour un enseignant/un étudiant : selon l’authentification, l’utilisateur sera alors « détecté » comme étant soit un enseignant (s’il a une liste des modules), soit un étudiant (s’il a un groupe), soit un utilisateur lambda (ni l’un ni l’autre). Selon le cas, le planning pourra alors être affiché.
  + S’il s’agit d’un utilisateur lambda, il devra rechercher la liste des cours associés à enseignant ou un groupe d’étudiants. C.F. « Recherche d’un élément » ;
  + S’il s’agit d’un enseignant, il sera alors affiché tous les cours correspondant aux modules qui lui sont associés ;
  + S’il s’agit d’un étudiant, il sera affiché tous les cours correspondant au groupe d’étudiants auquel il appartient.

## Fonctionnalités administrateur

* Ajout d’un élément (cours, matière, utilisateur, salle, groupe d’étudiants, bâtiment)
* Ajout d’un cours : l’administrateur devra renseigner les champs suivants
  + Nom de la matière
  + Type (TD/TP/CM)
  + Date
  + Horaire
  + Salle
  + Enseignant
  + Groupe d’étudiants

Après ajout, il cliquera sur le bouton « ajouter » pour sauvegarder les données dans la base de données.

Les fonctionnalités d’ajout auront toujours le même principe avec des champs à renseigner différents. Voici ci-après les différents ajouts et leurs champs.

* Ajout d’une matière :
  + Nom
  + Nombre d’heures
  + Niveau (ex : M1 MIAGE)
* Ajout d’une salle
  + Bâtiment
  + Nom de la salle (numéro)
* Ajout d’un bâtiment :
  + Nom du bâtiment
* Ajout d’un groupe d’étudiants
  + Niveau (ex : L1)
  + Formation (ex : MIAGE)
* L’ajout d’un utilisateur se fera de manière spécifique. Tout d’abord, l’administrateur devra choisir entre l’ajout d’un administrateur ou d’un utilisateur basique.
  + S’il choisit un administrateur, alors le profil dans la base de données permettra l’accès aux modifications et suppressions (profil administrateur) ;
  + Sinon, il n’aura accès qu’aux fonctionnalités de consultation, c’est-à-dire la recherche et l’affichage (profil utilisateur).

Ensuite, il devra rentrer les champs suivants :

* + Nom
  + Prénom
  + Adresse
  + Email
  + Mot de passe
  + Téléphone
  + Groupe (éventuellement vide, permet de spécifier un groupe d’étudiant)
    - Un étudiant ne peut faire partie que d’un seul groupe.
    - Caché si création d’un administrateur.
  + Modules (éventuellement vide, permet de spécifier un les matières d’un enseignant)
    - Un enseignant peut avoir une liste de modules à enseigner. Il sera donc possible de sélectionner un ou plusieurs modules dans une liste.
    - Caché si création d’un administrateur.
* Modification d’un élément (cours, matière, utilisateur, salle, groupe d’étudiants, bâtiment) : l’administrateur aura accès à un formulaire reprenant tous les champs de l’élément à modifier. Il pourra les modifier à son gré, et lorsqu’il cliquera sur « OK », la modification sera effectuée en base de données, puis il retournera à l’affichage précédent (liste des éléments).
* Suppression d’un élément (cours, matière, utilisateur, salle, groupe d’étudiants, bâtiment) : l’administrateur cliquera sur un bouton de suppression associé à l’élément concerné. Après confirmation, l’élément sera supprimé.
* Planification automatique d’un module : un module peut-être planifié automatiquement en fonction du nombre d’heures et du découpage, de l’enseignant, de la date de début et des groupes d’étudiants concernés. Un algorithme vérifiera la disponibilité des ressources pour les allouer convenablement selon les contraintes évoquées précédemment.

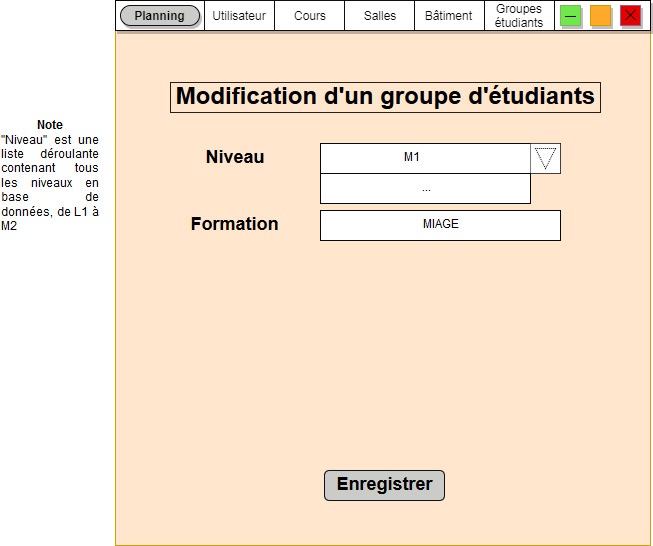
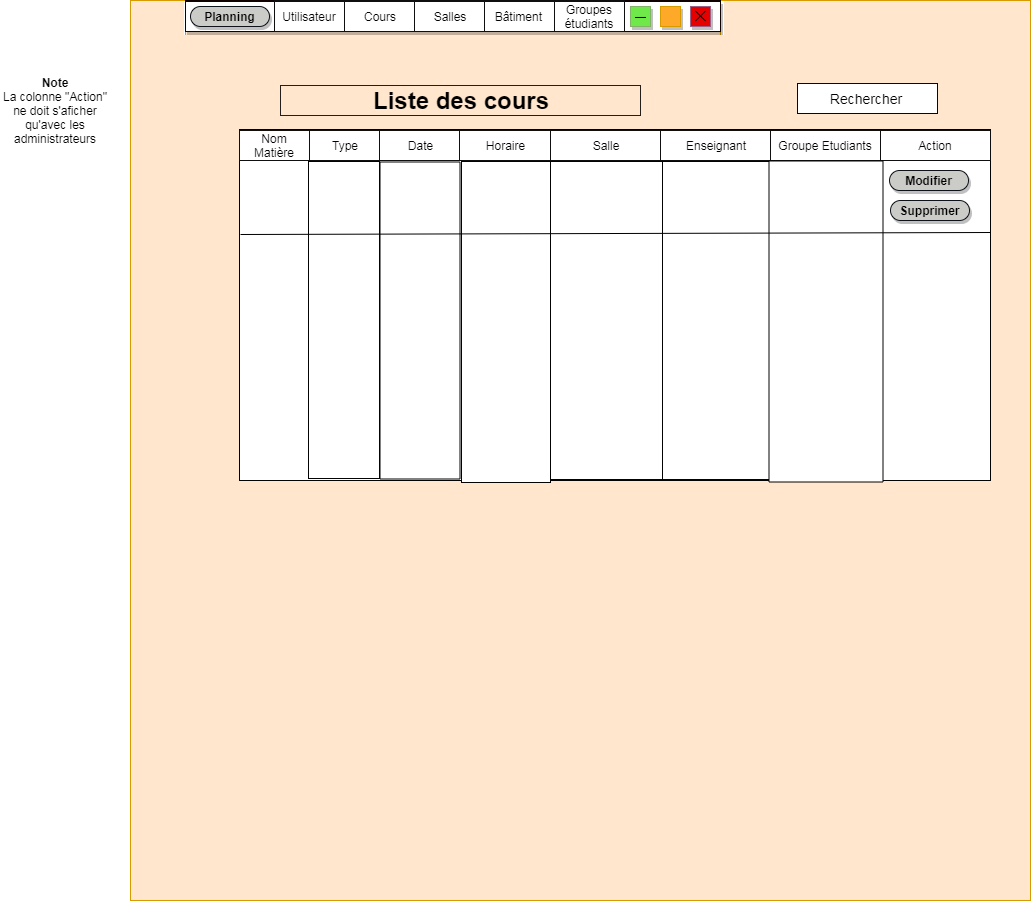
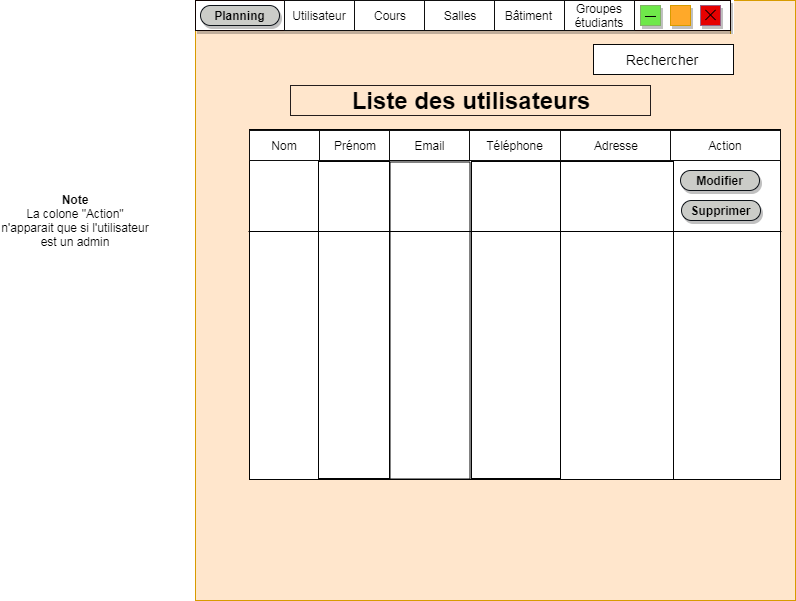
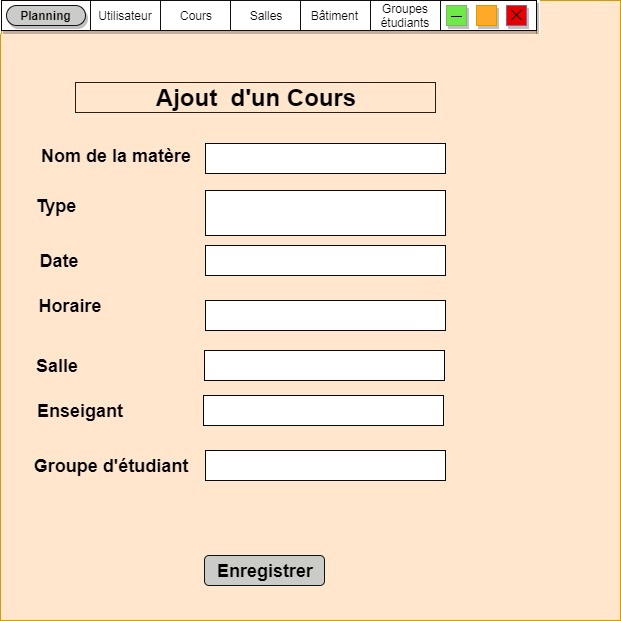
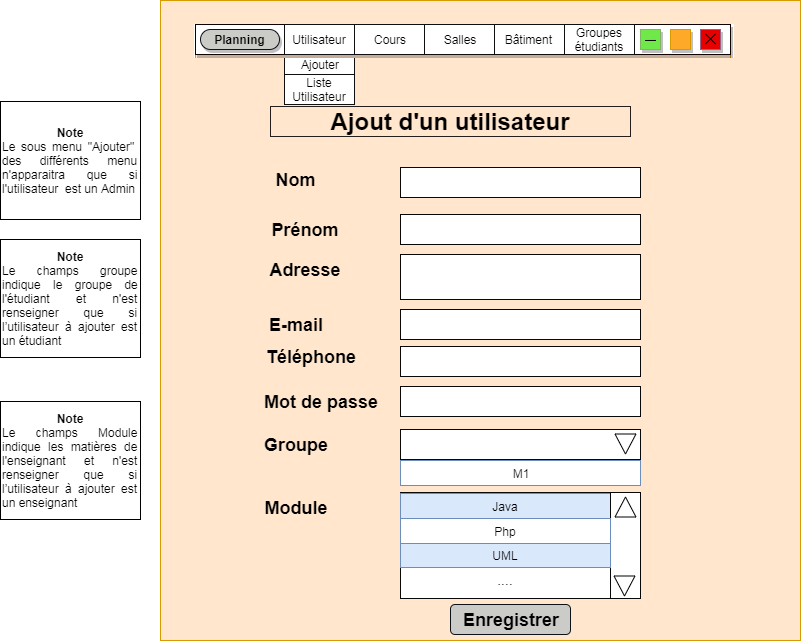
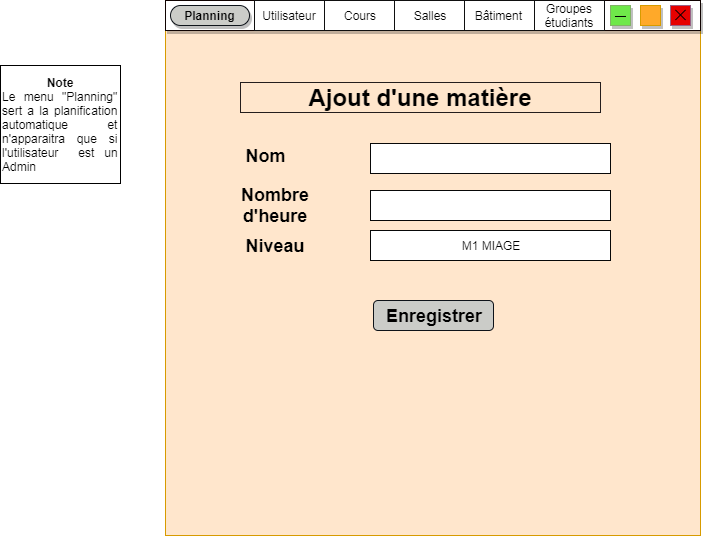
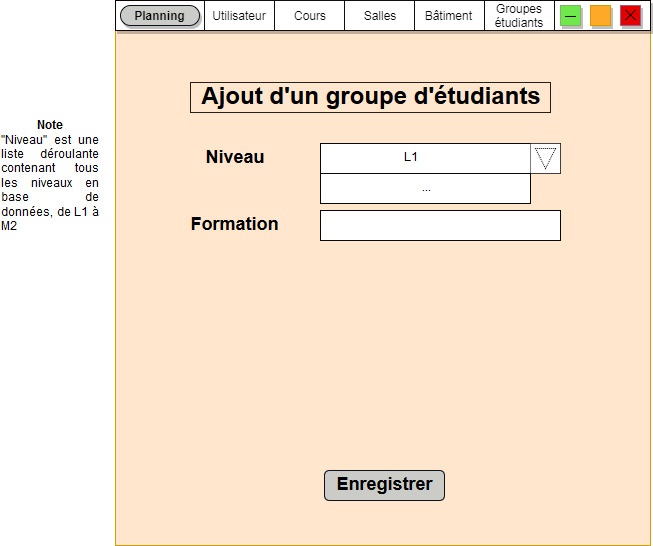
# Délais

L’application devra être livrée sous deux semaines (10 jours ouvrés). Elle devra contenir le maximum de fonctionnalités, comme précisées dans le Plan de Validation du Logiciel.

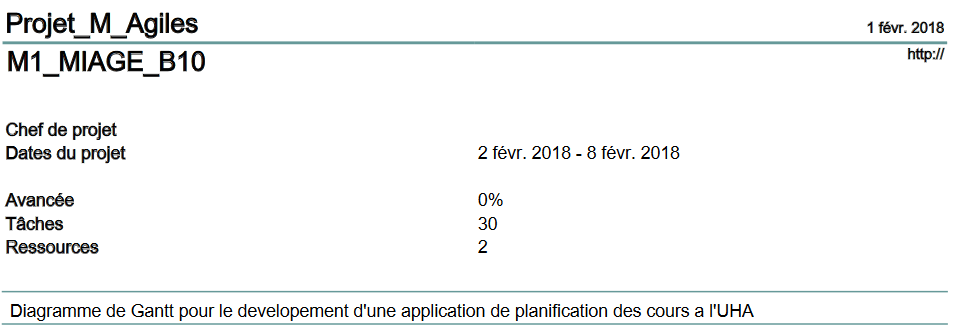
Les délais sont renseignés dans le diagramme de Gantt renseigné en [Annexe 2.](#_Annexe_2_:)

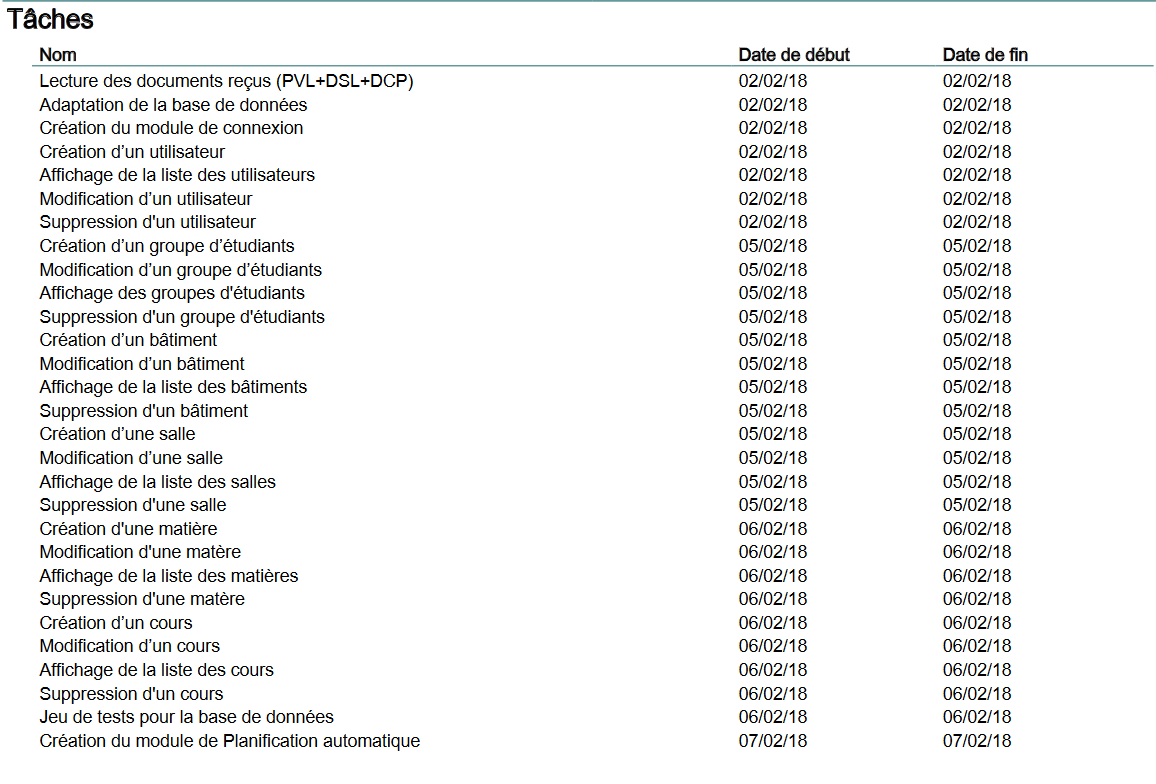
# Annexes

## Annexe 1 : Interfaces Homme-Machine

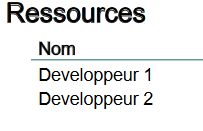
Ci-après se trouvent toutes les IHM imaginées pour ce projet. Elles sont à titre indicatif et ont pour vocation d’aider le développeur à développer tous les éléments nécessaires à chaque module.

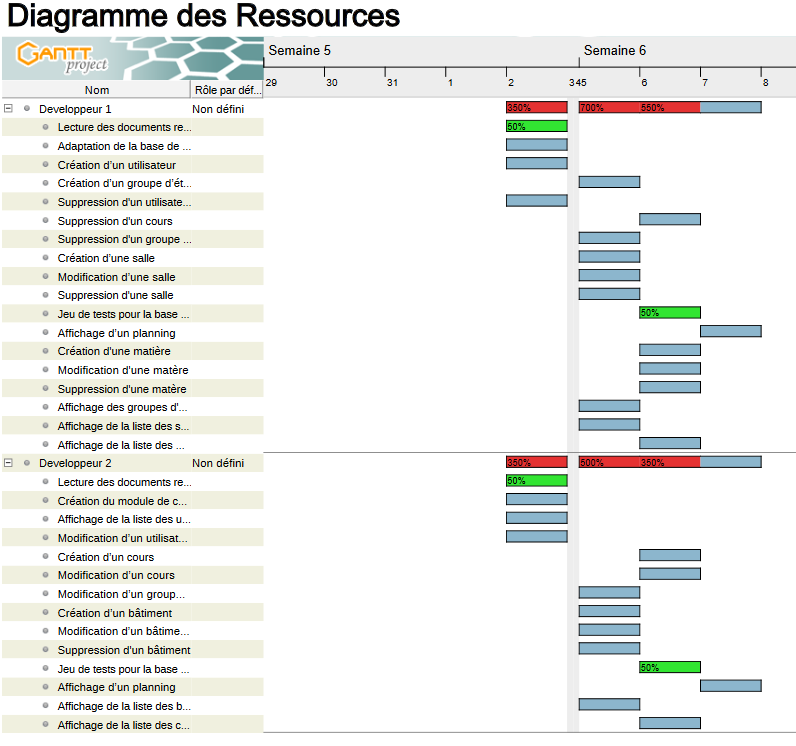
## Annexe 2 : Diagramme de Gantt









NB : les tâches sont découpées en jour. De ce fait, il est possible d’avoir plusieurs tâches à effectuer par jour. Il est donc normal de trouver des pourcentages supérieurs à 100 % (ex : « 700 % » pour 7 tâches dans une journée).