

Royaume du Maroc
UNIVERSITÉ MOHAMED V – RABAT

ECOLE NATIONALE SUPERIEURE D'INFORMATIQUE
ET D'ANALYSE DES SYSTEMES



Rapport de Projet de Fin de 2-ème année

Génération d'emploi du temps interactif par satisfaction des contraintes

Filière : Génie logiciel (GL)

Réalisé par :

Ikrame El arfaoui

Asmaa El azhar

Sous la direction de :

Mr. Mahmoud Elhamlaoui

Membres de jury :

Mr. Mahmoud EL HAMLAOUI

Mr. Mahmoud NASSAR

Année universitaire : 2020 – 2021

Résumé

Le présent document est le fruit de notre travail dans le cadre de projet de fin de première année. Ce projet avait comme but de réaliser une conception et une réalisation d'une application de génération d'emploi du temps interactif avec gestion de CSP.

Grâce à l'aide de notre encadrant, on a appris à utiliser différents outils et découvert plusieurs concepts qui nous ont aidé à réaliser cette application.

Durant notre projet, nous avons pour mission dans une première étape de faire une analyse du problème d'emploi du temps et de satisfaction des contraintes ainsi qu'une conception de cette application où nous avons analysé le sujet, décidé d'une approche crédible et modélisé l'application.

Dans la deuxième étape, nous avons pour une mission le développement et la réalisation de l'application où nous avons commencé à mettre au point l'application en créant la base de données et le site web en question grâce aux langages Django, Python, HTML, CSS.

Abstract

This document is the result of our work as part of the end of the first year project.

The goal of this project was to design and build an interactive schedule generation application with CSP management.

Thanks to the help of our supervisor, we learned to use different tools and discovered several concepts that helped us make this application.

During our project, our mission was in a first step to make an analysis of the problem of use of time and satisfaction of constraints as well as a design of this application where we analyzed the subject, decided on a credible approach and modeled the application.

In the second stage, we had for a mission the development and the realization of the application where we started to develop the application by creating the database and the website in question thanks to the languages Django, Python, HTML , CSS.

Table des matières

| | |
|---|-----------|
| Abstract | 0 |
| Introduction générale | 4 |
| Chapitre 1 : Présentation du projet | 5 |
| 1.1. Contexte général | 6 |
| 1.2. Motivation et planification | 6 |
| 1.3. Cadre d'étude | 7 |
| 1.4. Les ressources..... | 7 |
| 1.5. Les contraintes..... | 8 |
| Chapitre 2 : Résolution de problèmes de satisfaction de contraintes avec des algorithmes adaptés | 10 |
| 2.1. introduction | 11 |
| 2.2. Qu'est ce qu'un CSP | 11 |
| 2.3. Concepts de base d'un CSP: | 11 |
| 2.4. Méthodes de résolution des CSPs : | 14 |
| 2.4.1. Méthodes exactes : | 14 |
| 2.4.2. Les méthodes approchées : | 17 |
| 2.4.3. Emploi du temps et CSP : | 17 |
| Chapitre 3 : Analyse et Conception..... | 19 |
| 3.1. Analyse et approche | 20 |
| 3.2. Conception UML..... | 20 |
| 3.2.1. Diagramme de cas d'utilisation..... | 20 |
| 3.2.2. Diagramme des classes | 21 |
| Chapitre 4 : Réalisation | 23 |
| 4.1. Backend | 24 |
| 4.1.1. Django Framework : | 24 |

| | |
|-------------------------------------|----|
| 4.1.2. Design pattern MVC/MVT | 25 |
| 4.1.2.1. ORM | 25 |
| 4.1.2.2. MVC | 25 |
| 4.1.2.3. MVT | 26 |
| 4.2. Frontend..... | 27 |
| Conclusion..... | 34 |
| Webographie..... | 35 |

Table des figures

| | |
|---|----|
| Figure 1 : Diagramme de cas d'utilisation | 20 |
| Figure 2 : Digramme de classes | 21 |
| Figure 3 : Schéma de l'architecture MVT | 26 |
| Figure 4 : Page accueil | 27 |
| Figure 5 : Liste des emplois du temps..... | 27 |
| Figure 6 : Emploi du temps d'une filière | 28 |
| Figure 7 : Télécharger emploi du temps..... | 28 |
| Figure 8 : Liste des matières..... | 29 |
| Figure 9 : Supprimer matière..... | 29 |
| Figure 10: Ajouter matière | 30 |
| Figure 11 : Gestion des professeurs | 30 |
| Figure 12 : Liste des professeurs | 31 |
| Figure 13 : Ajouter un nouveau professeur | 31 |
| Figure 14 : Liste des salles | 32 |
| Figure 15 : Ajouter une salle | 32 |
| Figure 16 : Liste des filières | 33 |
| Figure 17 : Ajouter une filière | 33 |

Introduction générale

Chaque année, les responsables pédagogiques des universités et des écoles d'ingénieurs ont pour mission d'organiser les emplois du temps des différentes formations ou filières en essayant, au mieux, de satisfaire les contraintes « humaines » des enseignants et des étudiants, les contraintes pédagogiques imposées par la progression des enseignements et en tenant compte des contraintes « physiques » liées aux ressources matérielles (les salles, les équipements, etc.).

Il existe bien des outils de génération des emplois du temps dans les laboratoires de recherche, mais ces outils souffrent d'un manque d'interfaces conviviales et accessibles. Par exemple, certains outils peuvent prendre en compte différentes contraintes que doit exprimer l'utilisateur dans un langage spécifique. Cependant, les contraintes concernant les souhaits des responsables, des enseignants et des étudiants sont difficiles à exprimer. De plus, les responsables pédagogiques avouent que les difficultés ne viennent pas seulement de la génération des emplois du temps mais aussi de leur manipulation. On constate, en effet, que lors de la création d'un emploi du temps, les responsables partent généralement d'un emploi du temps existant auxquels ils apportent des modifications. La méthode ne consiste donc pas à créer mais à adapter un emploi du temps. D'où la nécessité de disposer d'outils interactifs facilitant ces adaptations. Contrairement aux outils de génération automatique, les outils recherchés sont « centrés utilisateur » ; il convient de revenir à des outils totalement sous le contrôle de l'utilisateur, adaptés aux démarches de travail selon différents points de vue liés aussi bien à la conception, qu'à la visualisation (consultation et recherche) et la modification des emplois du temps. Dans cette optique, l'utilisateur redevient l'acteur central de la résolution du problème, et les outils s'articulent autour de son activité.

L'outil recherché doit donc avant tout être interactif, souple, ouvert et doit présenter des qualités et satisfaire plusieurs contraintes. Ces raisons nous ont motivés à réfléchir sur la conception d'une application web d'aide à la manipulation des emplois du temps suffisamment ouvert pour permettre de nombreuses extensions.

Chapitre 1 : Présentation du projet

Ce chapitre a pour objectif de présenter le contexte du projet, les problèmes à résoudre, les objectifs à atteindre, et la démarche suivie.

1.1. Contexte général

Les différentes filières, les salles, les professeurs, les horaires, les cours,... autant d'informations essentielles liées à la planification des emplois du temps des étudiants de chaque filière et de chaque année scolaire. Cette application servira donc à stocker, gérer et manipuler toutes ces informations correctement et avec plus de facilité.

1.2. Motivation et planification

Notre projet de fin d'année consiste donc à développer une application pour générer les emplois du temps d'une école.

Nos objectifs sont les suivants :

- Générer les emplois du temps.
- Visualiser la liste des emplois du temps générés et télécharger l'emploi du temps par filière, par professeur et par année.
- Visualiser la liste des matières , mettre à jour, ajouter ou supprimer une matière.
- Visualiser la liste des professeurs ,ajouter ou supprimer une matière.
- Visualiser la liste des salles , mettre à jour, ajouter ou supprimer une salle.
- Visualiser la liste des filières , mettre à jour, ajouter ou supprimer une filière.

L'objectif sera également d'avoir une approche originale et différente par rapport au sujet et d'utiliser au mieux les outils dont nous disposons pour satisfaire les contraintes.

Nous avons commencé par une phase où nous avons analysé le sujet, décidé d'une approche crédible et modélisé l'application, ainsi que le choix de l'algorithme qu'in va adapter pour manipuler les données et éviter le maximum des conflits liés aux problème de planification.

Ainsi qu'une phase de développement où nous avons commencé à mettre au point l'application en créant la base de données, l'algorithme choisi et le site web en question.

1.3. Cadre d'étude

Chaque année, les responsables pédagogiques de la direction des études ont pour mission de concevoir les emplois du temps des différentes filières en essayant, au mieux, de satisfaire les contraintes « humaines » des enseignants et des étudiants, les contraintes pédagogiques imposées par la progression des enseignements et en tenant compte des contraintes « physiques » liées aux ressources matérielles (les salles, les équipements, etc.). Ainsi, l'idée de mettre en place un système de gestion des emplois du temps est née, dont l'objectif est:

- La génération automatique des emplois du temps.
- Consultation et suivi des emplois du temps.

1.4. Les ressources

Les ressources considérées sont les entités physiques nécessaires à l'élaboration des emplois du temps. Il s'agit des salles, des enseignants, des groupes, des étudiants et des matériels. Les ressources sont caractérisées par des données abstraites et des données spécifiques. Les données abstraites caractérisent chaque ressource et sont constituées d'un code, qui permet de la différencier des autres ressources, son calendrier qui précise quels sont Les jours de disponibilité et d'indisponibilité et sa description. En plus de ces données abstraites, chaque ressource possède des caractéristiques spécifiques qui dépendent du type de la ressource. L'intérêt de distinguer ces deux types de caractéristiques est que l'outil peut Très facilement évoluer pour prendre en compte de nouveaux types de ressources. Dans la suite nous décrivons les caractéristiques spécifiques des ressources considérées Dans l'étude.

- **Les ressources de type « salle »**

Une salle est un lieu dans lequel sont assurés des enseignements. Le type d'une salle est une indication sur le type d'enseignement qu'on peut y faire.

- **Les ressources de type « enseignant »**

L'enseignant désigne une personne pouvant assurer des enseignements. Chaque enseignant est caractérisé par : son nom et son prénom, son grade ,sa spécialité. La spécialité renseigne sur les matières que peut enseigner un enseignant.

- **Les ressources de type « groupe »**

Un groupe est un ensemble d'étudiants. Il se compose d'autres groupes. En général les étudiants d'une filière sont décomposés en groupe et chaque groupe est décomposé en plusieurs.

- **Les entités temporelles**

Pour modéliser le temps, les entités : date, heure, durée, créneau et calendrier sont définies.

- **Les séances et les réservations**

Une séance correspond à une instance temporelle d'un enseignement à une date donnée, pendant un créneau précis. L'ensemble des séances d'une ressource est appelé planning de cette ressource. Il s'agit de l'ensemble des séances dans lesquelles apparaît la ressource. Ainsi, il est possible de considérer le planning d'une salle au même titre que celui d'un enseignant. La notion de planning apparaît également au niveau des enseignements : cela permet ainsi de manipuler l'évolution dans le temps d'un enseignement particulier en vue. Une réservation correspond à un créneau ajouté au planning d'une ressource. Elle correspond à une option posée sur l'occupation de cette ressource. Par rapport à une séance une réservation n'est pas associée à un enseignement. Les réservations apportent de la souplesse dans l'utilisation de l'outil notamment lors de la phase de création des emplois du temps par les différents responsables.

1.5. Les contraintes

L'analyse sur le terrain montre que les données gérées doivent vérifier certaines contraintes pour garantir leur cohérence. De manière abstraite, les contraintes à respecter peuvent être classées en deux groupes : les contraintes physiques et les contraintes pédagogiques.

- **Les contraintes physiques**

Ces contraintes ne doivent pas être violées sinon cela conduirait à des situations conflictuelles. on dira qu'il y a un « conflit physique de ressource » entre deux séances si ces deux séances ont une ressource en commun pendant une durée non nulle.

Voici-les contraintes physiques:

- 1- une ressource ne peut pas être occupée en même temps dans deux séances différentes.
- 2- on ne peut pas mettre plus d'étudiants qu'il n'y a de places dans une salle.
- 3- Un professeur ne peut pas enseigner deux matières différentes dans le même horaire.

- **Les contraintes pédagogiques**

Les contraintes pédagogiques diffèrent des contraintes physiques par le fait qu'elles peuvent éventuellement être violées, dans ce cas on obtient des emplois du temps de moins bonne qualité d'un point de vue pédagogique. Typiquement ces contraintes sont utilisées pour exprimer ce que doit être un « bon » emploi du temps. Voici quelques exemples de ces contraintes :

- 1- homogénéiser la durée des séances d'un enseignement.

- 2- éviter les séances creuses dans l'emploi du temps.
- 3- éviter de placer plusieurs séances d'un même enseignement dans une journée.
- 4- éviter de finir après 17h50.
- 5- Eviter d'affecter des séances durant un créneau horaire déclaré temps libre du professeur.
- 6- Le nombre maximum des matières enseignées par jour ne peut pas dépasser 6 matières.

Dans ce premier chapitre, nous avons donné une idée claire sur le projet, son contexte, les objectifs à atteindre, et la démarche suivie.

Chapitre 2 : Résolution de problèmes de satisfaction de contraintes avec des algorithmes adaptés

Ce chapitre a pour objectif de formuler le problème de satisfaction des contraintes et de déterminer les différentes solutions.

2.1. introduction

Le problème d'emploi du temps est un problème défini en termes de contraintes (de temps, d'espaces ou plus généralement de ressources comme d'ailleurs d'autres problèmes tels que :

- les problèmes de planification et d'ordonnancement : planifier une production, gérer un trafic ferroviaire...
- les problèmes d'affectation du personnel à des tâches, des entrepôts à des marchandises,...

Ces différents problèmes fortement contraint ayant la particularité commune d'être d'être caractérisés par une très forte combinatoire, peuvent être considérés comme des problèmes de satisfaction de contraintes (**CSP** : **C**onstraint **S**atisfaction **P**roblems) car ces problèmes se formulent aisément en CSP quand ils ne nécessitent que des contraintes fortes.

L'utilisation d'une technique de satisfaction de contraintes est adaptée aux problèmes très contraints où une exploration de l'espace de recherche est envisageable.

Dans ce type de problème, la difficulté est de trouver une solution satisfaisant toutes les contraintes et non de trouver une solution minimisant ou maximisant une fonction, on fait d'ailleurs souvent abstraction de cette fonction en ne tenant compte que des contraintes : on cherche une solution réalisable et non pas la meilleure solution.

2.2. Qu'est ce qu'un CSP

Un CSP est défini comme étant un ensemble de contraintes impliquant un ensemble de variables, dont chacune est définie sur son domaine propre. L'objectif consiste à trouver un ensemble de valeurs, choisies dans les domaines susmentionnés, à affecter à ces variables de sorte que toutes les contraintes soient satisfaites.

2.3. Concepts de base d'un CSP:

Plus formellement, un problème de satisfaction de contraintes est défini par le triplet (X, D, C) tel que :

- $X = (X_1, X_2, \dots, X_n)$ est l'ensemble des n variables du problème.
- $D = (D_1, D_2, \dots, D_n)$ est un ensemble de n domaines finis dont chacun est associé à une variable de X . C'est à dire le domaine D_i est associé à la
- $C = (C_1, C_2, \dots, C_m)$ est un ensemble de m contraintes. Chaque contrainte C_i est défini par un couple (v_i, r_i) tel que :
 - $v_i = \{X_{i1}, \dots, X_{ini}\}$ est un ensemble de n_i variables sur lesquelles porte la contrainte C_i ; n_i est appelé arité de la contrainte.
 - r_i est une relation définie par un sous-ensemble de produits cartésiens $D_{i1} \times \dots \times D_{ini}$ des domaines associés aux variables de v_i . Elle représente les n -uplets de valeurs autorisées pour ces variables.

Définitions :

Pertinence : on dit qu'une variable X_i est pertinente pour la contrainte C_k , si C_k porte sur la variable X_i .

Arité de la contrainte : l'arité de la contrainte C_k est le nombre de variables pertinents pour C_k .

CSP binaire : un CSP binaire est un CSP $P = (X, D, C)$ dont toutes les contraintes $C_k \in C$ ont une arité égale à 2. En d'autres termes, chaque contrainte a exactement 2 variables pertinentes. Notons que tout CSP n -aire peut être ramené à un CSP binaire équivalent.

Matrice de contraintes : une matrice de contraintes est une matrice Mat à m lignes (m étant le nombre de contraintes) et n colonnes (n étant le nombre de variables du CSP) telle que :

$$Mat[i, j] = \begin{cases} 1 & \text{si } X_i \text{ est pertinent pour } C_i \\ 0 & \text{sinon} \end{cases}$$

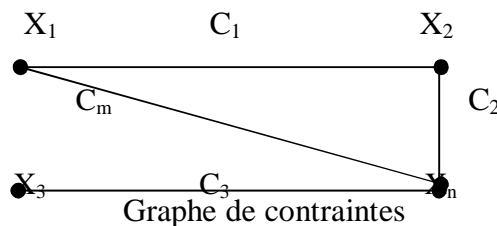
Variables

$$\text{contraintes} \begin{matrix} C1 \\ C2 \\ C3 \\ Cm \end{matrix} \begin{pmatrix} X_1 & X_2 & X_3 & \dots & X_n \\ 1 & 1 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 1 \\ 0 & 0 & 1 & \dots & 1 \\ 1 & 0 & 0 & \dots & 1 \end{pmatrix}$$

Matrice de contraintes Mat[i,j]

Sur cet exemple, on peut lire : X_1 et X_2 sont pertinentes pour C_1 , X_2 et X_n sont pertinentes pour C_2 etc...

Graphe de contraintes : une contrainte relie les deux variables pour lesquelles ellesont pertinentes par une arête, les sommets du graphe étant les variables.



Instanciation : étant donnée un CSP (X,D,C) , on appelle instanciation du CSP I une application qui associe à chaque variable $X_i \in X$ une valeur I_i , telle que $X_i \in D_i$.

Instanciation partielle : une instanciation partielle I_p de $X_p = \{X_{p1}, \dots, X_{pj}\} \subseteq X$ est une application qui associe à chaque variable $X_{pi} \in X_p$ une valeur $I_p(X_{pi}) \in D_{pi}$ (D_{pi} est le domaine de X_{pi}).

Satisfaction de contraintes : soit P un CSP, une instanciation partielle I_p satisfait la contrainte $C_i = (v_i, r_i)$ et on notera $(I_p \models C_i)$ ssi $v_i \in X_p$ et $I_p(v_i) \in r_i$. On dira alors que I_p viole C_i ssi $v_i \notin X_p$ et $I_p(v_i) \notin r_i$.

Instanciation consistante : soit P un CSP, une instanciation partielle I_p des variables est dite consistante ssi $C_i = (v_i, r_i) \in C$ telle que $v_i \in X_p$ et $I_p \models C_i$.

Solution d'un CSP : une solution S d'un CSP P est une instanciation consistante de toutes les variables (c'est à dire $X=X_p$). On dira que S satisfait $P (S \models P)$.

2.4. Méthodes de résolution des CSPs :

Plusieurs approches sont utilisées pour résoudre les CSPs, on distingue les méthodes exactes (ou complètes) et les méthodes approchées (ou incomplètes).

Les méthodes exactes font une recherche arborescente en instanciant les variables une par une et effectuent des réparations en cas d'échec. Les méthodes approchées font une réparation d'une configuration en parcourant d'une manière aléatoire l'espace de recherche.

2.4.1. Méthodes exactes :

La plupart des algorithmes de recherche exacte pour les problèmes de satisfaction de contraintes sont basés sur l'algorithme Backtrack Standard. De nombreuses améliorations majeures ont été proposées pour cet algorithme de base.

a) L'énumération (Backtrack) :

Cette méthode consiste à énumérer l'ensemble des solutions potentielles et à vérifier que chaque contrainte est satisfaite. Cette méthode donne une bonne réponse à tous les coups, mais conduit à développer un grand nombre de solutions potentielles (croissance exponentielle avec la taille des données) rendant souvent les traitements très lourds. Pour trouver systématiquement une solution, l'algorithme affecte au fur et à mesure une valeur de D à chaque variable correspondant dans X . il vérifie évidemment que la valeur est correcte par rapport aux contraintes. A chaque affectation, les domaines des variables restantes diminuent. Il arrive qu'à l'affectation d'une variable X_n , l'algorithme ne puisse trouver de valeur pour X_n car $D(X_n)$ est vide. A ce stade, l'algorithme revient en arrière, à la dernière variable affectée X_{n-1} . il modifie la valeur de X_{n-1} en espérant que le domaine de la variable X_n suivante ne sera plus nul. Si après avoir essayé toutes les valeurs du domaine $D(X_{n-1})$, il n'a pas toujours de solutions pour X_n , il recule et modifie la variable précédent X_{n-2} . s'il n'existe pas de solution, il reculera jusqu'à X_0 , sinon il trouve obligatoirement la solution.

Le défaut de cet algorithme est qu'il peut tester toutes les valeurs possibles avant de trouver une solution. Compte tenu de la nature des problèmes gérés, ses dimensions deviennent rapidement ingérables, car elles augmentent

selon une loi de type n puissance p . si jamais la première bonne solution se trouve dans les dernières possibilités ou si jamais il n'y a pas de solutions, le temps de calcul sera très long.

b) Amélioration du Backtrack :

✓ Algorithmes avec retour arrière non chronologique :

Quand il rencontre un échec, l'algorithme Backtrack remet en cause le dernier choix et essaie alors une nouvelle valeur pour la variable courante x . lorsque toutes les valeurs ont été utilisées, il revient en arrière sur la variable précédente. Cependant, rien ne garantit que cette variable soit en cause dans les différents échecs rencontrés lors de l'affectation de x . par exemple, si cette variable n'est pas liée à x par une contrainte, elle ne peut être impliquée dans ces échecs. Par conséquent, essayer de nouvelles valeurs pour cette variable conduira aux mêmes échecs au niveau de l'affectation de x . pour pallier ce défaut du Backtrack, plusieurs méthodes dotées de retour arrière non chronologiques ont vu le jour. Le retour arrière non chronologique consiste à analyser les causes des échecs et à revenir sur la variable la plus profonde qui est en cause dans l'échec. C'est la notion de saut en arrière ou backjump. La façon d'analyser les causes des échecs détermine alors l'algorithme. Parmi les algorithmes exploitant une technique de retour arrière non chronologique, on cite :

- l'algorithme Backjumping (BJ): lorsque toutes les extensions de l'affectation courante avec une valeur de x sont inconsistantes, BJ revient en arrière sur la variable la plus profonde dans l'arbre dont la valeur est en conflit avec une valeur de x .
- l'algorithme Graph-based Backjumping (GBJ) : le retour arrière repose sur le graphe de contraintes. Si l'affectation courante ne peut être étendue de façon consistante à une variable x , alors GBJ revient en arrière sur la variable la plus profonde du voisinage de x . soit y cette variable. Si toutes les valeurs du domaine de y ont été essayées, GBJ revient sur la variable la plus profonde qui appartienne soit au voisinage de y , soit au voisinage de toute variable instanciée après y qui soit une cause potentielle de l'échec d'une extension de l'affectation courante (x est en une), et ainsi de suite.
- L'algorithme Conflict-directed Backjumping (CBJ) : pour chaque variable instanciée x , CBJ maintient un ensemble dit ensemble des conflits. Cet ensemble contient toutes les variables instanciées avant x avec lesquelles x est en conflit pour au moins une de ses valeurs ainsi que les variables qui sont en cause dans l'échec d'une extension d'une affectation contenant x .

lorsqu'un échec survient ou lorsque toutes les valeurs ont été essayées, on revient en arrière jusqu'à la variable la plus profonde de l'ensemble des conflits de la variable courante.

✓ **Algorithmes avec filtrage avant :**

L'algorithme Backtrack, teste la consistance de l'affectation courante en vérifiant qu'elle ne viole aucune contrainte liant x et une variable affectée. Autrement dit, le test de consistance s'effectue en fonction des variables déjà instanciées. C'est le concept de consistance en arrière (ou look-back scheme). Une autre technique consiste à exploiter le concept de consistance en avant (ou look-ahead scheme). Suivant ce concept, à chaque affectation d'une variable x , les valeurs des variables non instanciées qui ne sont pas compatibles avec la valeur de x sont supprimées. Le calcul des valeurs à supprimer dépend alors du niveau de filtrage utilisé. Un algorithme utilisant ce concept cherche à étendre une affectation consistante A . dans ce but, il choisit une variable non instanciée et lui affecte une valeur v du domaine. Le choix des variables et des valeurs à instancier fait généralement appel à des heuristiques. Une fois l'affectation construite, il supprime du domaine de chaque variable non affectée les valeurs qui ne sont pas compatibles avec v selon le niveau de filtrage utilisé. Si un domaine devient vide, alors l'affectation ne possède pas d'extension consistante. Il faut alors essayer une nouvelle valeur pour x . si toutes les valeurs ont été essayées, l'algorithme revient en arrière sur la variable affectée juste avant x . Si tous les domaines possèdent au moins une valeur, alors l'algorithme tente d'étendre en procédant comme précédemment. La recherche se termine quand toutes les variables sont instanciées (découverte d'une solution) ou si toutes les possibilités ont été étudiées (preuve de l'inconsistance du problème).

• **Algorithmes avec mémorisation :**

Les algorithmes avec retour arrière non chronologique permettent d'éviter certaines redondances dans l'arbre de recherche. Toutefois, il subsiste tout de même des redondances. Aussi, pour ne pas visiter plusieurs fois les mêmes sous arbres, une solution consiste à mémoriser des informations portant sur le travail déjà réalisé. Ces informations sont généralement mémorisées sous la forme de contraintes induites désignées généralement par le terme nogood. Un nogood correspond à une affectation qui ne peut être étendue en une solution. Leur mémorisation permet donc d'interdire certaines affectations et ainsi d'éviter de reproduire plusieurs fois les mêmes sous arbres. Parmi ces méthodes, on peut citer :

- 1- Jump-back learning
- 2- Nogood recording

3- Dynamic backtracking

2.4.2. Les méthodes approchées :

Les méthodes approchées réparent une configuration donnée en parcourant de manière aléatoire l'espace de recherche. La méthode min-conflits [MIN92] est la plus répandue. Elle est construite autour d'une heuristique de réparation locale avec minimisation de conflits. L'heuristique consiste à choisir une variable intervenant dans une contrainte non satisfaite. Et à choisir pour cette variable, une valeur qui minimise le nombre de contraintes violées.

Les méthodes stochastiques sont aussi des méthodes approchées, mais elles ont une approche différente de celles citées précédemment. Les méthodes stochastiques commencent par une instanciation initiale et elles essaient de la réparer en faisant des changements de valeurs de certaines variables. Contrairement aux autres méthodes approchées, celles-ci admettent une dégradation de la solution, d'une façon qui suit certaines règles qui font que, l'algorithme puissent sortir d'un optimum local. Parmi ces méthodes on cite le recuit simulé, la méthode tabou, les algorithmes génétiques et la méthode basée sur le SMA(Système Multi Agent) : les colonies de fourmis.

2.4.3. Emploi du temps et CSP :

Une approche énumérative est utilisée pour résoudre le problème de l'emploi du temps. Ce problème d'optimisation d'emploi du temps se définit par six ensembles :

Un ensemble Professeurs = $\{P_1, \dots, P_p\}$, un ensemble Formations = $\{F_1, \dots, F_f\}$, un ensemble Enseignements = $\{E_1, \dots, E_e\}$, un ensemble Salles = $\{S_1, \dots, S_s\}$, un ensemble Créneaux = $\{C_1, \dots, C_c\}$ et un ensemble Contraintes contenant l'ensemble des contraintes entre les variables des cinq ensembles précédents. La connaissance du problème permet pour chaque cours de fixer quelques variables. Ainsi pour chaque professeur P_i de l'ensemble Professeurs, est fixé quel enseignement et à quelle formation il le dispensera. Il reste donc qu'à fixer pour chaque cours la salle et le créneau qui seront utilisés et qui satisfassent les contraintes. Le CSP proposé se présente de la façon suivante :

$V = \{P_1, \dots, P_p, F_1, \dots, F_f, E_1, \dots, E_e, S_1, \dots, S_s, C_1, \dots, C_c\}$

$D = \{D(P_1) = \dots = D(P_p) = [1, p], D(F_1) = \dots = D(F_f) = [1, f], D(E_1) = \dots = D(E_e) = [1, e], D(S_1) = \dots = D(S_s) = [1, s], D(C_1) = \dots = D(C_c) = [1, c]\}$

$C = \{\text{les contraintes dures plus les contraintes faibles}\}.$

Les problèmes d'optimisation combinatoire et d'affectation sous contraintes ont des méthodes adaptables, développées par la recherche opérationnelle et sont basées sur un algorithme qui sont les méthodes approchées, cette dernière permet de trouver des solutions de bonne qualité en un temps d'exécution limité pour des problèmes réels de grande taille..

Chapitre 3 : Analyse et Conception

Ce chapitre a pour objectif de décrire les éléments d'analyse du sujet afin de présenter une approche originale et une modélisation de l'application.

3.1. Analyse et approche

Notre vie est plein des problèmes parmi aux celle d'affectation sous contrainte Et d'optimisation combinatoire dans tous les secteurs professionnelles : Administratif, transport, informatique et industriel. Et par conséquent la bonne gestion de temps reste la seule solution de la génération Automatique d'un emploi de temps. Cette solution consiste à effecteur un ensemble d'objets a un ensemble de ressources Dans l'espace de temps en satisfaisant un ensemble de contraintes. Donc le traitement du problème d'affectation se fait à partir d'une vue Recherche Opérationnelle (RO) et Intelligence Artificielle(IA). Le but de notre application est de répondre au besoin qui consiste au génération automatique d'un emploi du temps tout en évitant le maximum de conflits possibles.

3.2. Conception UML

3.2.1. Diagramme de cas d'utilisation

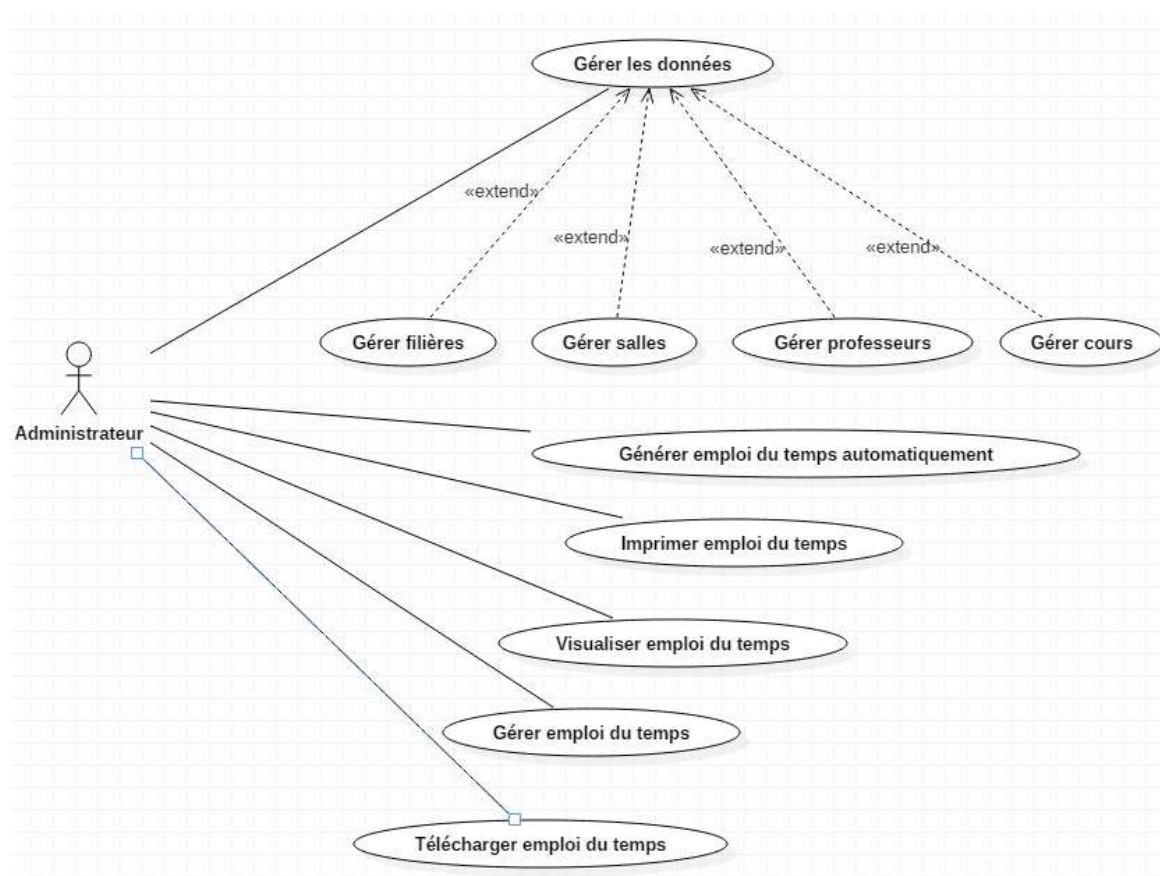


Figure 1 : Diagramme de cas d'utilisation

3.2.2. Diagramme des classes

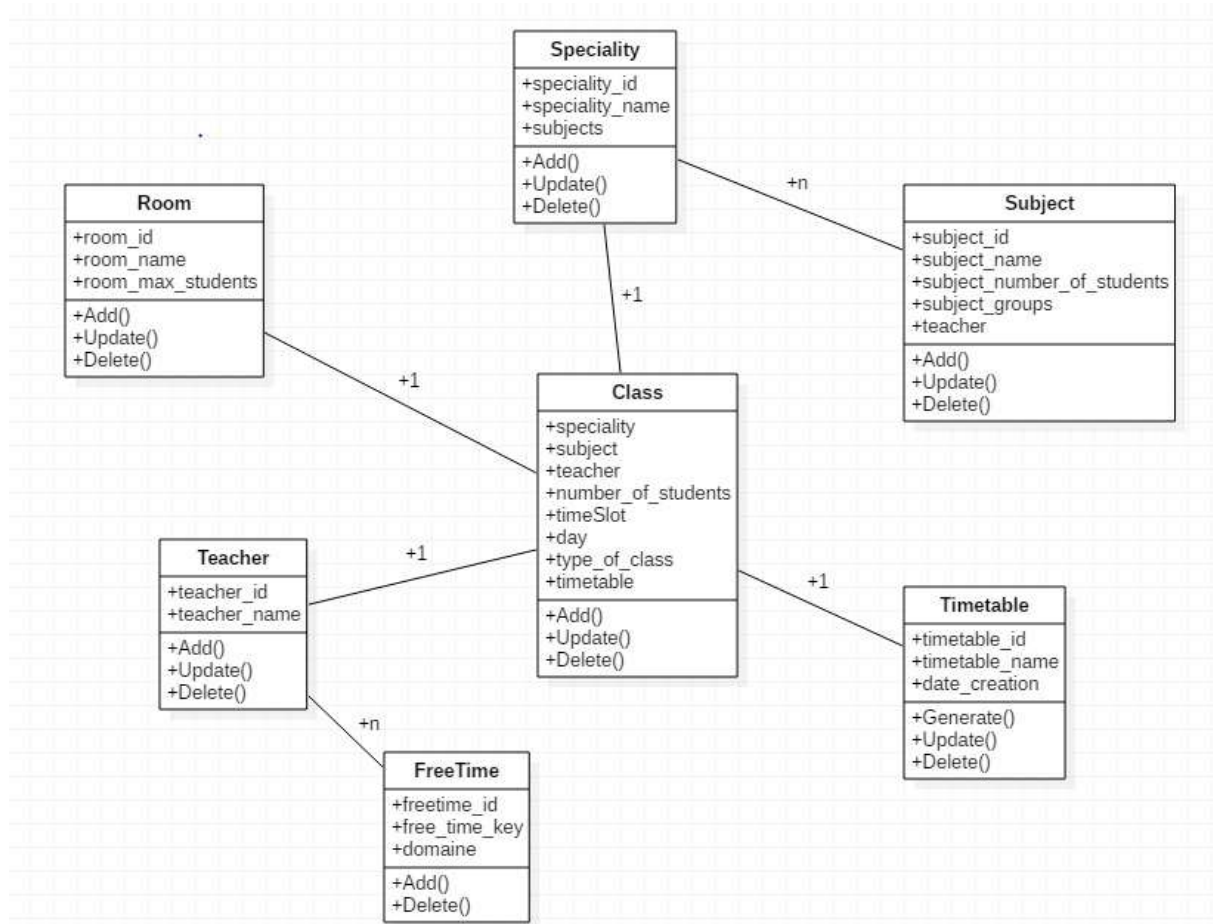


Figure 2 : Diagramme de classes

Dans ce chapitre, nous avons élaboré une analyse et une conception, décidé d'une approche crédible, et modélisé de l'application.

Chapitre 4 : Réalisation

Ce chapitre abordera les outils que nous avons choisis d'utiliser, et présentera le résultat final de l'application.

4.1. Backend

4.1.1. Django Framework :

Django est un framework Python de haut niveau, permettant un développement rapide de sites internet, sécurisés, et maintenables. Créé par des développeurs expérimentés, Django prend en charge la plupart des tracas du développement web, vous pouvez donc vous concentrer sur l'écriture de votre application sans avoir besoin de réinventer la roue. Il est gratuit, open source, a une communauté active, une bonne documentation, et plusieurs options pour du support gratuit ou non.

Django vous aide à écrire une application qui est:

- **Complète** : Django suit la philosophie "Piles incluses" et fournit presque tout ce que les développeurs pourraient vouloir faire. Comme tout ce dont vous avez besoin est une partie de ce "produit", tout fonctionne parfaitement ensemble, suivant des principes de conception cohérents.
- **Polyvalent** : Django peut être (et a été) utilisé pour créer presque tous les genres de sites — du gestionnaire de données aux wikis, jusqu'aux réseaux sociaux et aux sites d'actualités. Il peut fonctionner avec n'importe quelle infrastructure côté client, et peut renvoyer des données dans quasiment n'importe quel format (notamment HTML, RSS, JSON, XML, etc).
- **Sécurisé** : Django aide les développeurs à éviter les erreurs de sécurité classique en fournissant une infrastructure conçue pour "faire ce qu'il faut" pour protéger les sites internet automatiquement. Par exemple, Django fournit un moyen sécurisé pour gérer les comptes des utilisateurs ainsi que leurs mots de passe, évitant les erreurs classiques comme mettre des informations sur la session dans des cookies, où elles sont vulnérables (à la place les cookies contiennent seulement une clé, et les données sont stockées dans la base de données), ou directement stocker des mots de passe, au lieu de mot de passe haché. Un mot de passé haché est une valeur dont la longueur est fixée, créée en envoyant le mot de passe à travers une fonction de hachage cryptographique. Django peut vérifier si un mot de passe entré est correct en l'envoyant dans la fonction de hachage et en comparant le retour avec la valeur stockée dans la base de données. De ce fait, la nature unidirectionnelle de la fonction rend difficile pour un attaquant de retrouver le mot de passe d'origine, même si la valeur hachée est compromise. Django active par défaut la protection contre beaucoup de vulnérabilités, comme les injections SQL, le cross-site scripting, le cross-site request forgery et le clickjacking.
- **Scalable** : Django utilise une architecture composite "shared-nothing" (chaque composant de l'architecture est indépendant des autres, et peut ainsi être remplacé ou

changé si besoin). En ayant des séparations nettes entre les différentes parties, Django peut se scaler lors d'une hausse de trafic en ajoutant du hardware à tous les niveaux : serveurs cache, serveurs de base de données, serveurs d'application. Certains des sites les plus fréquentés ont réussi à scaler Django pour répondre à leur demande (par exemple, Instagram et Disqus pour ne nommer qu'eux deux).

- **Maintenable** : Les principes de design du code Django encouragent la création d'un code simple à maintenir et réutilisable. Il fait notamment appel à la philosophie du Ne Vous Répétez Pas (DRY pour Don't Repeat Yourself en anglais), afin d'éviter toute duplication superflue, réduisant la taille de votre code. Django promeut aussi le regroupement de fonctionnalités reliées entre elles en "applications" réutilisables et, à un plus bas niveau, regroupe des lignes de code dépendantes entre elles en modules (suivant les lignes du motif d'architecture Modèle-vue-contrôleur (MVC)).

- **Portable** : Django est écrit en Python, qui fonctionne sous diverses plateformes. Cela veut dire que vous ne serez plus contraint par une plateforme en particulier, et vous pourrez faire fonctionner vos applications sous autant de versions de Linux, Windows et Mac OS X que vous le souhaitez. De plus, Django est très bien supporté par plusieurs fournisseurs d'hébergement web, qui offrent souvent des infrastructures et de la documentation spécifiques pour héberger des sites Django.

4.1.2. Design pattern MVC/MVT

4.1.2.1. ORM

Avant de voir les spécificités des design pattern MVC et MVT, il convient de s'intéresser aux ORM. C'est un terme anglais signifiant Object Relational Mapping, littéralement Mappage objet-relationnel. Il s'agit d'un programme informatique qui se place entre la couche de stockage des données dans la base de données relationnelle et la couche applicative. Elle permet une abstraction des données et une représentation sous forme d'objets. Ainsi dans la conception de notre application, nos informations seront stockées sous forme d'objets informatiques, et l'ORM s'occupera tout seul de la correspondance avec la base de données.

4.1.2.2. MVC

Le design pattern Modèle-Vue-Contrôleur ou **MVC** est un type d'architecture logicielle destiné aux interfaces graphiques lancé en 1978 et très populaire pour les applications web. Le motif est composé de trois types de modules assurant différents rôles:

- *Le modèle (Model) représente, souvent sous forme d'objet, les données à utiliser par l'application et utilise un ORM pour l'interaction avec la base de données.

- * La vue (View) contient la présentation des données via une interface graphique.

* Un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur.

L'idée importante dans ce design pattern est de séparer distinctement les tâches effectuées par les différents éléments. Ainsi la conception et la maintenance sont simplifiées.

4.1.2.3. MVT

Django utilise l'architecture **MVT** (modèle-vue-template) qui s'inspire de MVC:

* Le modèle interagit avec une base de données via un ORM. Tous les modèles sont réunis dans un fichier python models.py.

* La vue reçoit une requête HTTP et renvoie une réponse HTTP convenable (par exemple si la requête est une interaction avec une base de données, la vue appelle un modèle pour récupérer les items demandés). Les vues se trouvent dans le fichier views.py

* Le template est un fichier HTML récupéré par la vue et envoyé au visiteur avec les données des modèles.

La figure ci-dessous montre comment les différents composants de l'architecture MVT de Django interagissent pour répondre à la requête d'un utilisateur. Ici le contrôleur ne correspond pas au contrôleur du MVC, mais à Django en lui-même qui gère en interne tout ce qui est lié au choix de la vue à laquelle envoyer la requête HTTP,...

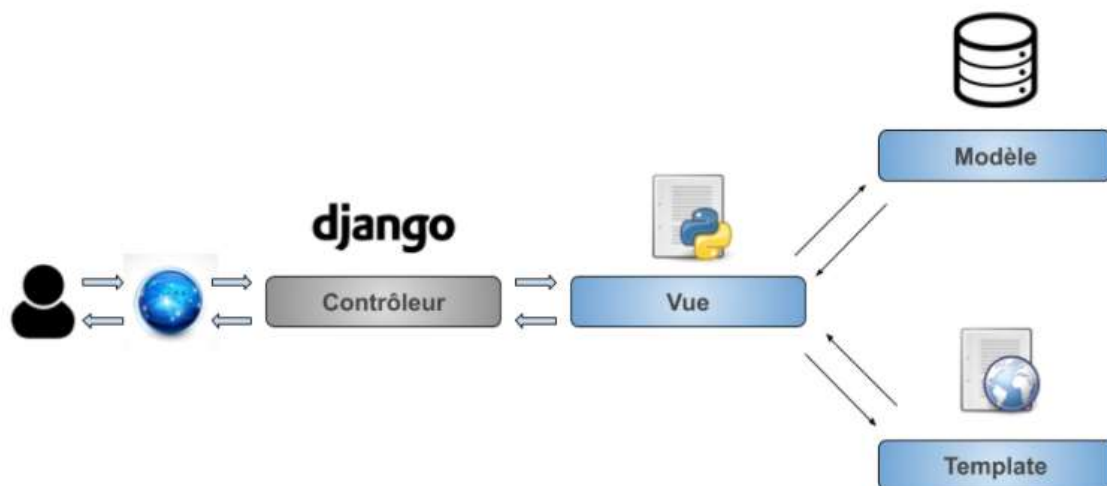


Figure 3 : Schéma de l'architecture MVT


4.2. Frontend



Figure 4 : Page accueil

| Timetable ID | TimetableName | Created On | view Timetable | Update name | Delete |
|--------------|------------------------|---------------------------|----------------|-------------|--------|
| 25 | my wonderful timetable | June 19, 2021, 12:44 a.m. | VIEW TIMETABLE | UPDATE NAME | REMOVE |
| 26 | None | June 20, 2021, 12:33 a.m. | VIEW TIMETABLE | UPDATE NAME | REMOVE |
| 27 | None | June 20, 2021, 12:35 a.m. | VIEW TIMETABLE | UPDATE NAME | REMOVE |
| 28 | None | June 20, 2021, 12:51 a.m. | VIEW TIMETABLE | UPDATE NAME | REMOVE |
| 29 | None | June 20, 2021, 12:52 a.m. | VIEW TIMETABLE | UPDATE NAME | REMOVE |
| 30 | None | June 20, 2021, 12:58 a.m. | VIEW TIMETABLE | UPDATE NAME | REMOVE |
| 31 | None | June 20, 2021, 1:04 a.m. | VIEW TIMETABLE | UPDATE NAME | REMOVE |

Figure 5 : Liste des emplois du temps



[Home](#)
[Subjects](#)
[Teachers](#)
[Classrooms](#)
[Specialties](#)

VIEW BY ▾

- Global
- View by teacher
- View by Speciality

| TIME | MONDAY | TUESDAY | WEDNESDAY | THURSDAY | FRIDAY |
|---------------|---|--|-----------|----------|--------|
| 08:30 - 09:50 | GL Ingénierie du Web Mahmoud EL Hamlaoui Lecture A1 | GL Systeme d'Informations Bouchaib Bourabou Practice 2.A2 GL vision et perception numerique fathi senae Practice 1.A1 GL Android Mounia Abik Practice 3.A5 | | | |
| 10:00 - 11:20 | GL Ingénierie du Web Mahmoud EL Hamlaoui Practice 1.A1 GL Design Patterns Mahmoud Nasser Practice 2.A3 GL Systeme d'Informations Bouchaib Bourabou Practice | GL Systeme d'Informations Bouchaib Bourabou Practice 3.A2 GL Android Mounia Abik Practice 1.A1 | | | |

Figure 6 : Emploi du temps d'une filière

Timeline - Personnel - Microsoft Edge

about:blank

| Time | Monday | Tuesday | Wednesday | Thursday | Friday |
|---------------|--------|---------|--|--|--|
| 08:30 - 09:50 | | | | Item JVM Bouchra Bernada Lecture L3 | Item JVM Bouchra Bernada Lecture L3 |
| 10:00 - 11:20 | | | | Item JVM Bouchra Bernada Practice 2.L3 | Item JVM Bouchra Bernada Practice 3.L2 |
| 11:40 - 13:00 | | | Item JVM Bouchra Bernada Practice 3.L3 | | |
| 13:30 - 14:50 | | | Item JVM Bouchra Bernada Lecture L3 | Item JVM Bouchra Bernada Practice 3.L3 | Item JVM Bouchra Bernada Practice 1.L3 |
| 15:00 - 16:20 | | | | Item JVM Bouchra Bernada Practice 2.L3 | Item JVM Bouchra Bernada Practice 2.L3 |
| 16:30 - 17:50 | | | Item JVM Bouchra Bernada Practice 1.L3 | Item JVM Bouchra Bernada Practice 1.L3 | |

Figure 7 : Télécharger emploi du temps


|  Home Subjects Teachers Classrooms Specialties | | | | | | |
|---|--------------------------------|------------------------|--------|-----------------------------|------------------------|------------------------|
| <input type="text" value="Search..."/> | | SEARCH | | ADD SUBJECT | | |
| Subject ID | Subject Name | number of students | groups | Teacher | Update | Delete |
| 1 | Ingénierie du Web | 60 | 3 | Mahmoud EL Hamlaoui | UPDATE | REMOVE |
| 2 | Design Patterns | 50 | 2 | Mahmoud Nassar | UPDATE | REMOVE |
| 3 | Datawarehouse | 40 | 4 | leila KJiri | UPDATE | REMOVE |
| 4 | Sql Server | 60 | 3 | Ahmed Ettalbi | UPDATE | REMOVE |
| 5 | Réseaux mobiles | 40 | 4 | Amine Berqia | UPDATE | REMOVE |
| 6 | Système d'informations | 50 | 5 | Bouchaib Bounabat | UPDATE | REMOVE |
| 7 | vision et perception numerique | 45 | 3 | fikhi sanae | UPDATE | REMOVE |

Figure 8 : Liste des matières



Figure 9 : Supprimer matière



The screenshot shows the 'ADD SUBJECT' form in the ENSIAS system. The form is located on the left side of the page, under a blue header bar that contains the ENSIAS logo and navigation links: Home, Subjects, Teachers, Classrooms, and Specialties. The form itself has a blue header with the text 'ADD SUBJECT'. Below this, there are four input fields: 'Subject name*', 'Subject number of students*', 'Subject groups*', and 'Teacher*'. Each field has a small asterisk indicating it is required. The 'Teacher*' field is a dropdown menu. Below the input fields is a blue button labeled 'SAVE SUBJECT'. To the right of the form is an illustration of three books (one blue, one green, one pink) and a thought bubble. Below the illustration, there is a text link: 'You Can View Your Added Course Here. [View Subjects](#)'.

Figure 10: Ajouter matière

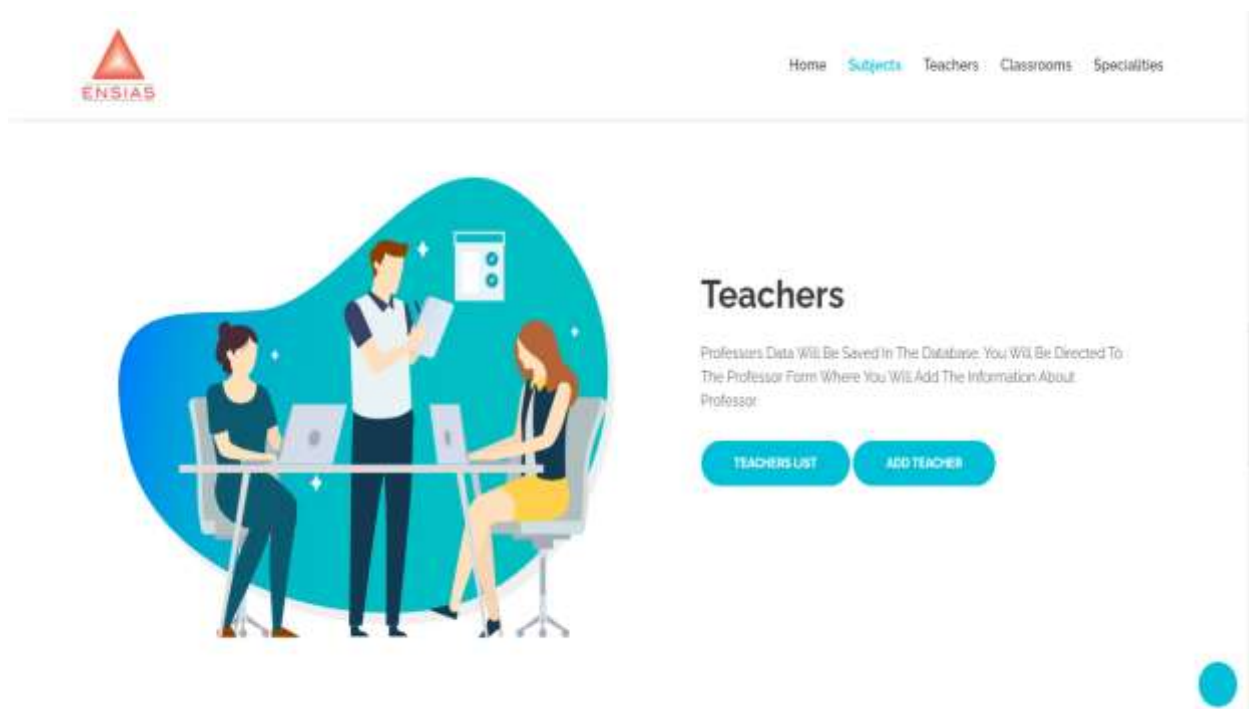
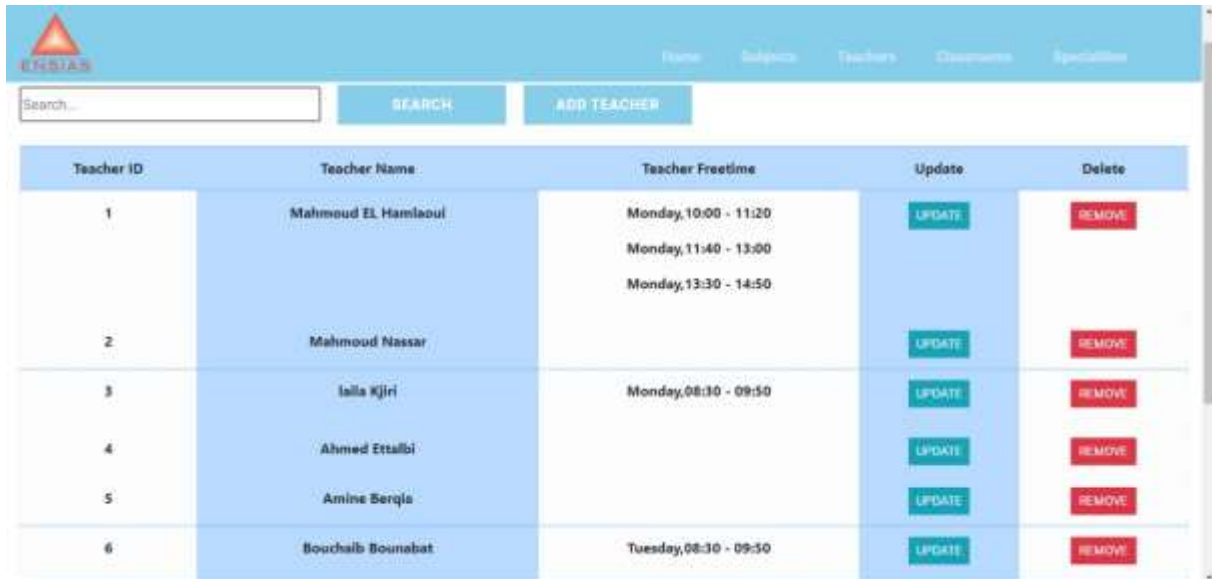


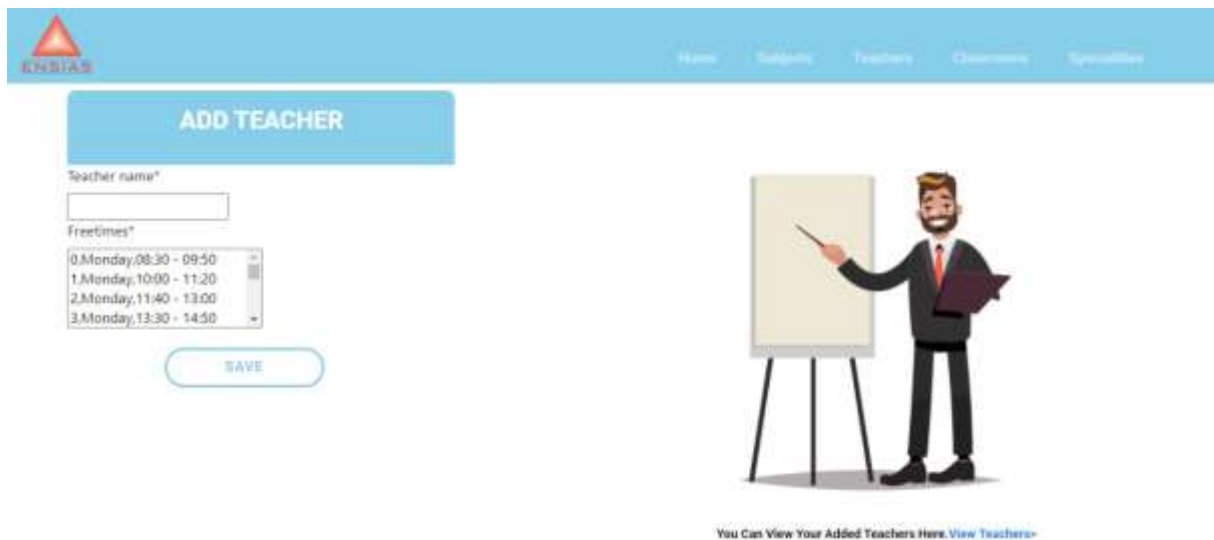
Figure 11 : Gestion des professeurs



The screenshot shows the 'Teachers' page of the ENSIA application. It features a navigation bar with links to Home, Subjects, Teachers, Classrooms, and Specialities. Below the navigation bar is a search bar and an 'ADD TEACHER' button. The main content is a table listing teachers with columns for Teacher ID, Teacher Name, Teacher Freetime, Update, and Delete.

| Teacher ID | Teacher Name | Teacher Freetime | Update | Delete |
|------------|---------------------|---|------------------------|------------------------|
| 1 | Mahmoud EL Hamlaoui | Monday, 10:00 - 11:20 Monday, 11:40 - 13:00 Monday, 13:30 - 14:50 | UPDATE | REMOVE |
| 2 | Mahmoud Nassar | | UPDATE | REMOVE |
| 3 | Iaila KJiri | Monday, 08:30 - 09:50 | UPDATE | REMOVE |
| 4 | Ahmed Ettalbi | | UPDATE | REMOVE |
| 5 | Amine Berqis | | UPDATE | REMOVE |
| 6 | Bouchaib Bounabat | Tuesday, 08:30 - 09:50 | UPDATE | REMOVE |

Figure 12 : Liste des professeurs



The screenshot shows the 'ADD TEACHER' form in the ENSIA application. It includes a navigation bar with links to Home, Subjects, Teachers, Classrooms, and Specialities. The form has a title 'ADD TEACHER' and two input fields: 'Teacher name*' and 'Freetimes*'. The 'Freetimes*' field is a dropdown menu showing a list of time slots. Below the input fields is a 'SAVE' button. To the right of the form is an illustration of a teacher standing next to a whiteboard. At the bottom, there is a link to 'View Teachers'.

Teacher name*


Freetimes*

0.Monday, 08:30 - 09:50
1.Monday, 10:00 - 11:20
2.Monday, 11:40 - 13:00
3.Monday, 13:30 - 14:50

[SAVE](#)

You Can View Your Added Teachers Here. [View Teachers](#)


Figure 13 : Ajouter un nouveau professeur



[Home](#)
[Subjects](#)
[Teachers](#)
[Classrooms](#)
[Specialities](#)

| Room ID | Room name | room max students | Update | Delete |
|---------|-----------|-------------------|---------------------------------------|---------------------------------------|
| 1 | L1 | 20 | <input type="button" value="UPDATE"/> | <input type="button" value="REMOVE"/> |
| 2 | L2 | 35 | <input type="button" value="UPDATE"/> | <input type="button" value="REMOVE"/> |
| 3 | L3 | 30 | <input type="button" value="UPDATE"/> | <input type="button" value="REMOVE"/> |
| 4 | A1 | 60 | <input type="button" value="UPDATE"/> | <input type="button" value="REMOVE"/> |
| 5 | A2 | 55 | <input type="button" value="UPDATE"/> | <input type="button" value="REMOVE"/> |
| 6 | L4 | 15 | <input type="button" value="UPDATE"/> | <input type="button" value="REMOVE"/> |
| 7 | A3 | 60 | <input type="button" value="UPDATE"/> | <input type="button" value="REMOVE"/> |
| 8 | A4 | 70 | <input type="button" value="UPDATE"/> | <input type="button" value="REMOVE"/> |


Figure 14 : Liste des salles



[Home](#)
[Subjects](#)
[Teachers](#)
[Classrooms](#)
[Specialities](#)


Room name*

Room max students*



You Can View Your Added Classroom Here, [View Rooms](#)

Figure 15 : Ajouter une salle




[Home](#) [Subjects](#) [Teachers](#) [Classrooms](#) [Specialities](#)

SEARCH
ADD SPECIALITY

| Speciality ID | Speciality Name | Speciality Subjects | Update | Delete |
|---------------|-----------------|---|--|--|
| 1 | GL | Ingénierie du Web Design Patterns Systeme d'informations vision et perception numerique Android | UPDATE | REMOVE |
| 2 | Iwlm | Ingénierie du Web Sql Server Réseaux mobiles java UML | UPDATE | REMOVE |

Figure 16 : Liste des filières



[Home](#) [Subjects](#) [Teachers](#) [Classrooms](#) [Specialities](#)


ADD SPECIALITY

Speciality name*

Subjects*

Ingénierie du Web
 Design Patterns
 Sql Server
 Réseaux mobiles

SAVE SUBJECT



[You Can View Your Added Specialities Here. View Specialities](#)

Figure 17 : Ajouter une filière

Conclusion

Le problème d'affectation et d'automatisation de l'emploi du temps est un problème complexe et très difficile à résoudre, même qu'il existe différentes approches d'affectation et de génération d'emplois du temps, reste toujours un champ de travail et de recherche d'actualité pour répondre à ce problème dans les mesures : faisabilité d'implémentation, temps de réponse d'affectation, optimisation de gain et ressources affectés. Dans notre projet, nous proposons l'affectation et la génération de l'emploi du temps par l'utilisation d'un algorithme qui s'inscrit dans l'ensemble de solutions proposées par la recherche opérationnelle. Afin d'une bonne conception de notre problème et de notre système d'information, l'UML (Unified Modeling Language) qui présente un langage de modélisation graphique à base de pictogrammes conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Dans ce projet, nous avons exploité nos efforts et nos connaissances pour pouvoir bien analyser le sujet de la génération d'emplois du temps. On espérant bien que ce travail peut soumettre pour résoudre le problème de l'automatisation de l'emploi du temps.

Webographie

- Documentation Django : [Django documentation | Django documentation | Django \(djangoproject.com\)](#)
- Conception d'un système pour l'emploi du temps universitaire,
Auteurs : Zouina Mallem, Mohamed-Khireddine Kholadi El-Oued
University, Sahraoui Sabrina Université Constantine 2.
[\(PDF\) Conception d'un système pour l'emploi du temps universitaire \(researchgate.net\)](#)
- Intégration de techniques CSP pour la résolution du problème WCSP,
auteur : Nicolas Paris.
- [Stack Overflow - Where Developers Learn, Share, & Build Careers](#)
- [GeeksforGeeks | A computer science portal for geeks](#)
- [Démarrez votre projet avec Python - OpenClassrooms](#)

