

# SYSTÈME DE GESTION D'EMPLOI DU TEMPS UNIVERSITAIRE

## Rapport Technique Détaillé

<b>Projet:</b>	Gestion Automatisée d'Emploi du Temps
<b>Version:</b>	2.0
<b>Date:</b>	26/01/2026
<b>Statut:</b>	■ Opérationnel
<b>Technologies:</b>	Python, Tkinter, JSON

# TABLE DES MATIÈRES

- 1. Introduction
- 2. Architecture du Système
- 3. Données et Structures
- 4. Interfaces Utilisateur
- 5. Logique Métier
- 6. Fonctionnalités Principales
- 7. Contraintes et Algorithmes
- 8. Statistiques du Projet
- 9. Conclusion

## 1. INTRODUCTION

Ce projet représente un système complet de gestion d'emploi du temps universitaire, conçu pour automatiser et optimiser la planification des cours, travaux dirigés (TD) et travaux pratiques (TP) dans un environnement académique complexe.

Le système gère l'allocation intelligente de ressources (salles, enseignants, groupes) tout en respectant de multiples contraintes temporelles et matérielles. Il offre des interfaces distinctes pour trois types d'utilisateurs : administrateurs, enseignants et étudiants.

### **Objectifs Principaux:**

- Automatiser la génération d'emplois du temps
- Gérer les réservations de salles en temps réel
- Optimiser l'utilisation des ressources disponibles
- Détecter et résoudre les conflits d'horaires
- Fournir des statistiques et rapports détaillés
- Offrir une interface intuitive pour tous les utilisateurs

## 2. ARCHITECTURE DU SYSTÈME

Le système adopte une architecture modulaire en couches, séparant clairement les responsabilités entre présentation, logique métier et données.

### Structure des Dossiers:

<b>Dossier</b>	<b>Description</b>
interfaces/	Interfaces graphiques (Admin, Enseignant, Étudiant)
logic/	Logique métier et algorithmes
DONNÉES PRINCIPALES/	Fichiers JSON de configuration
GESTION EDT/	Données générées (EDT, réservations)
DOCUMENTATION/	Documentation technique
backend/	Services backend (optionnel)
frontend/	Interface web (Vite.js)

### 3. DONNÉES ET STRUCTURES

<b>Ressource</b>	<b>Quantité</b>	<b>Description</b>
Salles	87	50 salles (Amphi, TP, TD, Cours)
Enseignants	60	Corps enseignant qualifié
Modules	186	6 modules par filière-année
Groupes	65	Groupes d'étudiants organisés
Filières	31	DEUST, Licence, Master, Ingénieur
Étudiants	1550	Total d'étudiants inscrits

#### Répartition des Salles:

- **3 Amphithéâtres** (400 places) - Cours magistraux DEUST
- **27 Salles TP** (30 places) - Travaux pratiques équipés
- **10 Salles Cours** (25 places) - Petits groupes
- **6 Salles TD** (50 places) - Travaux dirigés
- **4 Salles Cours** (90 places) - Grands groupes

## 4. INTERFACES UTILISATEUR

Le système propose trois interfaces distinctes, chacune adaptée aux besoins spécifiques de son type d'utilisateur.

### 4.1 Interface Administrateur

- **Tableau de Bord:** Vue d'ensemble des statistiques globales
- **Génération EDT:** Lancement de l'algorithme de placement automatique
- **Réservations:** Validation/rejet des demandes de réservation
- **Indisponibilités:** Gestion des absences enseignants
- **Occupation:** Visualisation en temps réel de l'occupation des salles
- **Statistiques:** Graphiques et analyses avancées
- **Exports:** Export PDF, Excel, CSV des emplois du temps
- **Disponibilités:** Blocage de créneaux pour maintenance
- **Données:** Consultation des salles, enseignants, modules

### 4.2 Interface Enseignant

- **Mon Emploi du Temps:** Visualisation personnalisée des cours
- **Réservations:** Demande de réservation de salles
- **Indisponibilités:** Déclaration d'absences planifiées
- **Statistiques:** Charge de travail et répartition horaire
- **Export:** Téléchargement de l'emploi du temps personnel

### 4.3 Interface Étudiant

- **Mon Emploi du Temps:** Consultation par filière et groupe
- **Filtrage:** Affichage par jour, semaine ou module
- **Informations:** Détails des cours (salle, enseignant, horaire)
- **Export:** Sauvegarde en PDF ou image

## 5. LOGIQUE MÉTIER

<b>Module</b>	<b>Fichier</b>	<b>Responsabilité</b>
Générateur EDT	edt_generator.py	Algorithme de placement des séances
Base de données	database.py	Gestion des fichiers JSON
Modèles	models.py	Classes POO (Salle, Enseignant, etc.)
Réservations	reservation_manager.py	Gestion des réservations
Générateur séances	seance_generator.py	Création des séances
Optimisation	optimization.py	Équilibrage de charge
Statistiques	stats_manager.py	Calculs statistiques
Export	exporter.py	Export CSV, Excel, PDF
Conflits	conflict_manager.py	Détection de conflits

## 6. FONCTIONNALITÉS PRINCIPALES

### Génération Automatique d'EDT

Algorithme intelligent qui place automatiquement toutes les séances en respectant les contraintes de disponibilité des enseignants, capacité des salles, et évite les conflits d'horaires.

### Gestion des Réservations

Les enseignants peuvent demander des réservations de salles. L'administrateur valide ou rejette après vérification de disponibilité.

### Déclaration d'Indisponibilités

Les enseignants déclarent leurs absences planifiées. Le système bloque automatiquement les créneaux concernés.

### Occupation en Temps Réel

Visualisation instantanée de l'état de toutes les salles (libre, occupée, réservée) pour un jour et horaire donnés.

### Statistiques Avancées

Graphiques de répartition par jour, taux d'occupation des salles, charge de travail des enseignants, plages horaires les plus demandées.

### Exports Multiformats

Export des emplois du temps en PDF, Excel, CSV, ou image PNG avec filtrage par filière ou enseignant.

### Détection de Conflits

Vérification automatique des conflits : salle occupée, enseignant indisponible, groupe en double, créneaux bloqués.

### Optimisation de Charge

Équilibrage intelligent de la charge de travail sur la semaine pour éviter les journées surchargées.





## 7. CONTRAINTES ET ALGORITHMES

### 7.1 Créneaux Horaires

<b>Jour</b>	<b>Créneaux</b>	<b>Total</b>
Lundi - Jeudi	09:00-10:30, 10:45-12:15, 12:30-14:00, 14:15-15:45, 16:00-17:30	5 × 4 = 20
Vendredi	09:00-10:30, 10:45-12:15, 14:15-15:45, 16:00-17:30	4
Samedi	09:00-10:30, 10:45-12:15	2
<b>Total Semaine</b>		<b>26 créneaux</b>

### 7.2 Contraintes Dures (Obligatoires)

- Un enseignant ne peut pas être à deux endroits en même temps
- Une salle ne peut accueillir qu'un seul cours à la fois
- Un groupe d'étudiants ne peut avoir qu'un cours à la fois
- La capacité de la salle doit être suffisante pour le groupe
- Respect des créneaux bloqués (indisponibilités, maintenance)
- Type de salle adapté au type de séance (TP nécessite salle TP)

### 7.3 Contraintes Douces (Optimisations)

- Équilibrage de la charge sur la semaine (éviter journées surchargées)
- Minimisation des salles sous-utilisées
- Regroupement des cours d'une même filière
- Préférence pour les salles du même bâtiment
- Éviter les créneaux tardifs quand possible

### 7.4 Algorithme de Placement

L'algorithme utilise une approche gloutonne avec backtracking:

1. **Tri par priorité:** Examens > Cours > TD > TP
2. **Pour chaque séance:**
  - a. Trier les jours par charge actuelle (équilibrage)
  - b. Pour chaque jour, chercher un créneau libre
  - c. Vérifier disponibilité enseignant et groupe

- d. Trouver salle adaptée (type, capacité, équipements)
  - e. Détecter les conflits
  - f. Si OK, placer la séance, sinon essayer jour suivant
3. **Si échec:** Proposer solution alternative ou signaler erreur
  4. **Sauvegarder:** EDT généré + rapport d'erreurs

## 8. STATISTIQUES DU PROJET

<b>Métrique</b>	<b>Valeur</b>
Lignes de code Python	~5021 lignes
Fichiers Python	42 fichiers
Modules logiques	10 modules
Interfaces graphiques	3 interfaces
Fichiers JSON de données	7 fichiers
Classes POO	~15 classes
Fonctions principales	~80 fonctions

### Technologies et Bibliothèques:

- **Python 3.x:** Langage principal
- **Tkinter:** Interface graphique desktop
- **JSON:** Stockage de données
- **Matplotlib:** Génération de graphiques statistiques
- **ReportLab:** Génération de rapports PDF
- **OpenPyXL:** Export Excel
- **Vite.js:** Interface web moderne (frontend)
- **Git:** Gestion de versions

## 9. CONCLUSION

Ce projet représente une solution complète et robuste pour la gestion automatisée d'emplois du temps universitaires. Le système démontre une architecture bien pensée, séparant clairement les responsabilités entre interfaces, logique métier et données.

### Points Forts:

- Architecture modulaire et maintenable
- Algorithme intelligent de placement avec gestion des contraintes
- Interfaces utilisateur intuitives et adaptées à chaque rôle
- Gestion complète du cycle de vie (génération, réservation, modification)
- Système de détection et résolution de conflits
- Exports multiformats et statistiques avancées
- Code bien documenté avec docstrings et commentaires

### Fonctionnalités Clés Implémentées:

- Génération automatique d'emplois du temps
- Gestion des réservations avec validation
- Déclaration d'indisponibilités enseignants
- Visualisation en temps réel de l'occupation
- Statistiques et analyses graphiques
- Exports PDF, Excel, CSV, Image
- Optimisation de la charge de travail

### Perspectives d'Évolution:

- Intégration d'une base de données SQL pour de meilleures performances
- Développement d'une API REST pour l'interface web
- Ajout de notifications par email
- Système de suggestions intelligentes basé sur l'historique
- Application mobile pour consultation en déplacement
- Intégration avec systèmes existants (Moodle, etc.)

Le système est actuellement **opérationnel** et prêt pour un déploiement en environnement de production après tests utilisateurs.