Manuel d'installation

Projet : Conception d'un outil d'expérimentation de scénarios et de génération d'emploi du temps

Arbaut Jean-Baptiste

Deschamps Kylian

Duez-Faurie Valentine

Eramil Kadir

Pilloud Aubry

Tropel Célia

Université Grenoble Alpes

Manuel d'installation

Conception d'un outil d'expérimentation de scénarios et de génération d'emploi du temps

Les informations d'identification du document :		*Les éléments de vérification du document :*	
Référence du document :		**Validé par :**	
Version du document :	1.1	**Validé le :**	
Date du document :	13/06/2025	**Soumis le :**	
Auteur(s) :	Arbaut Jean-Baptiste Deschamps Kylian Duez-Faurie Valentine Eramil Kadir Pilloud Aubry Tropel Célia	**Type de diffusion :**	Document électronique (.pdf)
		Confidentialité :	

Les éléments d'authentification :

Maître d'ouvrage:	Aurélie Landry	**Chef de projet :**	
Date / Signature :		**Date / Signature :**	

Sommaire

Table des tableaux et illustrations

1. Introduction

Ce document est le manuel d'installation de l'application Emploi du Temps - TER. Il rassemble l'ensemble des procédures nécessaires à la mise en place de l'application dans un environnement. Il est destiné à toute personne en charge de l'installation et du déploiement de l'outil sur un poste utilisateur.

Le manuel détaille étape par étape :

- Le matériel requis pour exécuter l'application dans de bonnes conditions.
- Le paramétrage du système (installation de Python).
- L'installation du logiciel (code source, bibliothèques).
- Le paramétrage et le lancement de l'interface.
- La gestion des données, des mises à jour et de l'arrêt / redémarrage de l'application.

L'objectif est de permettre une installation autonome, reproductible, et fiable sur n'importe quel système compatible.

1.1 Objectifs et méthodes

L'application développée s'inscrit dans le cadre d'un projet de TER (Travail d'Étude et de Recherche) et vise à proposer une solution complète et accessible pour la génération automatique d'emplois du temps dans les établissements scolaires de type collège. Cette application a pour objectif de réduire significativement le temps et la complexité liés à la création manuelle des plannings hebdomadaires, tout en garantissant la prise en compte de l'ensemble des contraintes pédagogiques, matérielles et humaines propres à chaque établissement.

Le projet repose sur deux composantes principales :

- Une interface web interactive, conçue avec le framework Dash (Python), permettant à l'utilisateur de renseigner ou importer les données nécessaires : enseignants, classes, disciplines, salles, contraintes horaires, options pédagogiques… Cette interface se veut claire, et accessible à des utilisateurs sans compétences en programmation. Elle permet également de visualiser, modifier et exporter les résultats.
- Un moteur de résolution de contraintes, basé sur la bibliothèque OR-Tools développée par Google, appelé "solver", qui modélise le problème de l'emploi du temps sous forme d'un problème d'optimisation à contraintes. Ce solver prend en compte les règles définies (volumes horaires, incompatibilités, préférences, simultanéités, etc.) et produit des plannings faisables et équilibrés, en minimisant les conflits.

Le développement a suivi une approche itérative en plusieurs étapes :

1. Recueil des besoins auprès d'exemples réels d'emplois du temps scolaires et définition des types de contraintes à prendre en charge.
2. Modélisation du problème et du plan de développement.
3. Implémentation du solver avec OR-Tools, incluant la gestion des sous-groupes, des salles, des volumes horaires par matière, et des conflits potentiels. Conception en parallèle de l'interface Dash en modules fonctionnels : saisie des données, gestion des contraintes, visualisation des résultats et export. Avec des fonctionnalités telles que le système de sauvegarde automatique.
4. Validation continue par des jeux de données réalistes et ajustements selon la faisabilité et la lisibilité des emplois du temps générés.
5. Création des tests et de la documentation associée.
6. Documentation : les commentaires du code source ainsi que la documentation interne ont été réalisés tout le long du projet. Le rapport final et les guides ont été rédigés en fin de projet.

L'ensemble du système est conçu pour fonctionner localement, sans dépendre d'un serveur distant, et pour pouvoir être adapté selon les spécificités des établissements. Le projet met l'accent sur la modularité, la fiabilité des résultats, et l'accessibilité pour des non-informaticiens. L'un des objectifs était de fournir un export compatible avec Monoposte d'Index Education, ce qui n'a pas pu être effectué.

1.2 Documents de référence

Le fichier "requirements.txt" du projet a été utile pour l'élaboration de ce document et est à utiliser pour l'installation du logiciel.

2. Guide de lecture

Ce document a été conçu pour répondre aux besoins de différents profils de lecteurs impliqués dans l'installation, l'utilisation ou la maintenance de l'application. Chaque section du manuel correspond à une étape clé de la mise en place du système.

2.1 Personne en charge de l'installation

Objectif : Installer et configurer l'application sur un poste utilisateur ou un serveur.

À lire en priorité :

- Section 4 à 7 : depuis l'installation du matériel jusqu'au lancement de l'interface.
- Section 9.1 : procédure de mise à jour.

Ce profil doit s'assurer que Python est bien installé, que les bibliothèques sont en place et que le projet est fonctionnel localement.

Cette personne peut être aussi l'utilisateur final, le maître d'ouvrage et M. Pellier pour son évaluation du projet.

2.2 Utilisateur final et maître d'ouvrage

Objectif : Utiliser l'interface pour configurer les données, générer et visualiser les emplois du temps.

À lire en priorité :

- Section 7.2 et 7.3 : comment utiliser et redémarrer l'application.
- Section 8 : gestion des données (saisie, import, export).
- Section 9.2 : sauvegarde et export des emplois du temps.

L'utilisateur final doit savoir comment interagir avec l'interface.

2.3 Développeur (maître d'œuvre)

Objectif : Comprendre, adapter ou améliorer le code source.

À lire en priorité :

- Section 6 : installation du projet via Git ou archive.

- Section 7 : structure et lancement de l'application.
- Section 8 : organisation des données JSON.
- Documents de référence (section 1.2).

3. Concepts de base

Cette section présente les notions essentielles à connaître pour comprendre le fonctionnement de l'application et suivre efficacement les étapes décrites dans ce document. Elle permet d'aborder la suite du manuel avec une vision claire des composants et mécanismes du système.

Certaines étapes d'installation ou d'utilisation (comme le lancement de l'application) nécessitent d'ouvrir un terminal (sous Linux ou macOS) ou une invite de commande (sous Windows). Il s'agit d'une interface en ligne de commande permettant d'exécuter des instructions système telles que "python app.py". Ce document fournit les indications nécessaires pour y accéder selon le système d'exploitation.

L'interface utilisateur est conçue avec Dash, un framework Python permettant de créer des interfaces web interactives. L'application s'ouvre dans un navigateur, ce qui permet à l'utilisateur d'interagir avec l'ensemble des fonctionnalités sans avoir besoin de connaissances en programmation. Toutes les saisies, modifications, visualisations et exports se font directement via cette interface.

Les données sont stockées sous forme de fichiers JSON (.json), un format standard lisible à la fois par les machines et par les humains, structuré en paires clé / valeur. Ce format est utilisé pour enregistrer les enseignants, les classes, les matières, les contraintes... Il n'est pas nécessaire de modifier ces fichiers à la main : l'interface gère leur création et leur mise à jour automatiquement.

Le cœur du moteur de génération repose sur OR-Tools, une bibliothèque d'optimisation développée par Google. Elle permet de formuler un ensemble de contraintes (Par exemple : « Un enseignant ne peut pas être assigné à deux classes au même moment. ») et de calculer une solution optimisée, correspondant à un emploi du temps cohérent et réaliste.

Enfin, l'application fonctionne en local, c'est-à-dire directement sur l'ordinateur de l'utilisateur, sans besoin de connexion à un serveur distant ou à une base de données en ligne. L'ensemble du traitement (saisie, calcul, affichage et export) est effectué localement.

4. Installation du matériel

L'application ne nécessite aucun matériel spécifique pour fonctionner.
Il est néanmoins recommandé d'installer le système sur un poste disposant de 8 Go de mémoire vive et d'au moins 2 Go d'espace libre sur le disque. Ces caractéristiques permettent d'assurer une exécution fluide, en particulier lors de l'utilisation du solveur de contraintes qui mobilise des ressources de calcul.
L'ordinateur doit être équipé d'un système d'exploitation récent tel que Windows 10, Ubuntu 20.04, ou macOS 12 ou supérieur. Il est aussi nécessaire d'avoir un accès à Internet pour l'installation initiale des dépendances, et d'un navigateur web à jour pour l'utilisation de l'interface.
Aucun périphérique particulier n'est requis, si ce n'est un clavier, une souris, et un écran d'une résolution suffisante (1280×720 ou plus) pour permettre un affichage lisible et confortable de l'interface web.

5. Paramétrage du système

L'application repose sur le langage Python, en version 3.10 ou ultérieure. Ce langage est utilisé à la fois pour l'interface web (Dash) et pour le moteur de calcul (solveur OR-Tools). Voici la procédure à réaliser pour obtenir Python dans une version compatible sur votre ordinateur.

5.1 Étape 1 : Vérifier si Python est déjà installé

Pour cela, il est nécessaire d'ouvrir le terminal de commande / l'invite de commande selon votre système :

Sytème	Comment ouvrir le terminal / la console
Windows	Appuyer sur "Windows + R", taper "cmd", puis appuyer sur "Entrée". (ou taper "Invite de commandes" dans le menu "Démarrer").
macOS	Ouvrir "Launchpad" → taper "Terminal" ou aller dans "Applications" > "Utilitaires" > "Terminal".
Linux	Appuyer sur "Ctrl + Alt + T" ou chercher "Terminal" dans le menu des applications.

Tableau 1 : Ouverture du terminal par système

Taper ensuite dans le terminal ou l'invite de commande :

python --version ou python3 --version

Si Python est déjà installé sur la machine, vous obtiendrez la réponse suivante : "python" suivi d'un numéro de version. Si la version est trop ancienne, il est nécessaire de poursuivre la suite des étapes. Si la version est 3.10 ou plus récente, passer directement à la partie 6. Installation du logiciel.

5.2 Étape 2 : Télécharger et installer Python

Si Python non installer ou que la version est inférieure à 3.10 :

Sytème	Procédure
Windows	1. Télécharger l'installeur "Windows Installer 64-bit" sur Python.org. 2. Cocher la case "Add Python to PATH". 3. Cliquer sur Install Now.
macOS	1. Télécharger l'installateur macOS 64-bit ("macOS 64-bit universal2

	installer") sur Python.org. 2. Ouvrir le fichier .pkg et suivre les étapes d'installation.
Linux	Dans le terminal écrire les commandes suivante : sudo apt update sudo apt install python3.10 python3.10-venv python3.10-distutils

Tableau 2 : Procédure de téléchargement de Python par système

Pour vérifier l'installation, il faut saisir dans le terminal de commande ou dans l'invite de commande :

pip --version ou pip3 --version

Veiller à bien utiliser pip3 sous macOS.

Un message similaire doit apparaître :

pip 23.3.1 from ... (python 3.10)

6. Installation du logiciel

Cette section détaille les étapes à suivre pour installer concrètement l'application sur un poste utilisateur ou un serveur local.

6.1 Étape 3 : Clonage ou récupération du projet

Ne réaliser qu'une des deux options suivantes suivant votre objectif :

1. **Option 1 : Cloner le projet : Github-Emploi-Du-Temps-TER**
   Cette option est à privilégier pour un développeur. Pour cela dans le terminal ou l'invite de commande, il faut saisir :

   ```
   cd /chemin/vers/le/dossier/souhaité
   git clone https://github.com/ValentineDz/Emploi-Du-Temps-TER.git
   cd Emploi-Du-Temps-TER/Projet
   ```

2. **Option 2 : Depuis une archive .ZIP :**
   Cette option est à privilégier pour un utilisateur souhaitant réaliser un emploi du temps. Pour cela :
   1. Télécharger l'archive du projet sur le Github-Emploi-Du-Temps-TER > Code > Download ZIP :

      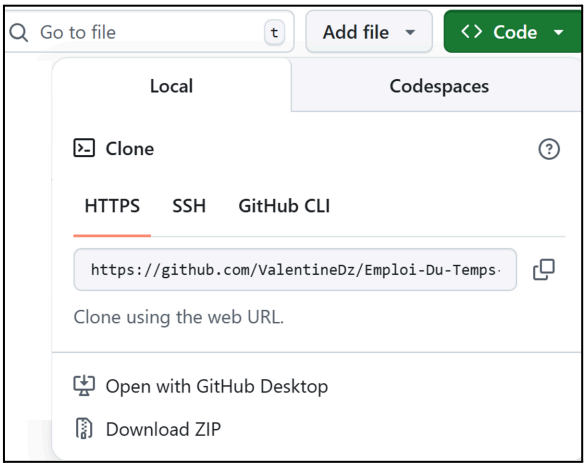

      Figure 1 : Télécharger le projet sur GitHub

   2. Extraire tous les fichiers dans le dossier souhaité.
   3. Ouvrir un terminal est utiliser la commande :

      ```
      cd /chemin/vers/le/dossier/Emploi-Du-Temps-TER/Projet
      ```

6.2 Étape 4 : Installation des dépendances

Les bibliothèques utilisées dans ce projet sont déclarées dans le fichier "requirements.txt". Ce fichier contient la liste des paquets nécessaires pour faire fonctionner l'interface et le solveur. Il faut donc dans le terminal ou l'invite de commande (dans le dossier du projet) écrire la commande suivante :

```
pip install -r requirements.txt     ou     pip3 install -r requirements.txt
```

7. Paramétrage du logiciel

7.1 Étape 5 : Lancement de l'application

Une fois les bibliothèques installées, pour lancer l'application, il faut écrire l'une des commandes suivantes dans le terminal ou l'invite de commande (toujours dans le dossier du projet) :

```
python app.py          ou          python3 app.py
```

Veiller à bien utiliser python3 sous macOS.

Le serveur Dash se lance localement, ouvrir le navigateur et aller à l'adresse :
http://127.0.0.1:8050.

7.2 Étape 6 : Utilisation et arrêt de l'application

L'application est maintenant fonctionnelle. Vous pouvez :

- Importer ou saisir les données (profs, classes, contraintes…)
- Générer l'emploi du temps via l'interface
- Visualiser les résultats
- Exporter les fichiers souhaités (PDF, HTML, etc.)

Pour arrêter l'application proprement, il suffit d'appuyer sur "CTRL + C" dans le terminal ou l'invite de commande.

7.3 Étape 7 : Redémarrer l'application plus tard

À chaque nouvelle session, il suffit de :
Ouvrir un terminal
Naviguer dans le dossier du projet et lancer à nouveau l'application :

```
cd /chemin/vers/le/dossier/Emploi-Du-Temps-TER/Projet
python app.py          ou          python3 app.py
```

Veiller à bien utiliser python3 sous macOS.

8. Installation des données

L'application ne repose sur aucune base de données relationnelle externe (comme MySQL ou PostgreSQL). Toutes les données sont uniquement stockées localement dans des fichiers au format JSON, lisibles et modifiables via l'interface. Il est cependant fortement déconseillé de modifier ces fichiers manuellement, afin d'éviter des erreurs de structure ou d'incompatibilité.
À la première utilisation, l'application affiche une interface web permettant de saisir ou d'importer l'ensemble des données : enseignants, classes, matières, contraintes, salles, options, langues, calendrier... Ces informations sont automatiquement enregistrées dans un fichier nommé "data_interface.json", situé dans le dossier "data".
L'utilisateur peut à tout moment réinitialiser les données depuis la page d'informations de l'interface, ou bien importer un fichier structuré selon le même format que "data_interface.json". Lorsque des données sont déjà présentes dans ce fichier, elles sont automatiquement chargées et affichées dans l'interface au démarrage. Elles peuvent être modifiées, complétées ou supprimées à tout moment par l'utilisateur.

9. Autres informations

9.1 Procédure de mise à jour

Pour mettre à jour le projet sans perte de données :

4. Arrêter l'application (en faisant "CTRL + C" dans le terminal ou l'invite de commande).
5. Sauvegarder le fichier "data/data_interface.json" si vous souhaitez conserver les données saisies, vous pourrez les réimporter depuis l'interface.
6. Télécharger la nouvelle version du projet via git pull ou une nouvelle archive ZIP.
7. Réinstaller les dépendances si le fichier "requirements.txt" a été modifié.
8. Relancer l'application avec :

python app.py ou python3 app.py

Veiller à bien utiliser python3 sous macOS.

9.2 Export et sauvegarde des données

L'utilisateur peut à tout moment exporter les résultats au format .pdf ou sauvegarder manuellement le fichier "data_interface.json" pour archivage ou réimport ultérieur.

9.3 Compatibilité avec les navigateurs

L'interface est compatible avec la plupart des navigateurs : Chrome, Firefox, Edge... Il est recommandé d'utiliser une version à jour pour éviter des problèmes d'affichage.

10. Glossaire

Ce glossaire définit les principaux termes techniques ou spécifiques utilisés dans ce document afin d'en faciliter la lecture pour tous les profils de lecteurs.

Terme	Définition
1. Application locale	Application qui fonctionne entièrement sur l'ordinateur de l'utilisateur, sans connexion à un serveur distant.
2. Archive .ZIP	Fichier compressé contenant l'ensemble des fichiers d'un projet ; peut être extrait sur n'importe quel système.
3. Bibliothèque (ou package)	Ensemble de fonctions et outils Python préprogrammés, installables via pip, utilisés pour étendre les capacités d'un projet.
4. CMD (Invite de commandes)	Terminal système sous Windows, permettant d'exécuter des commandes comme python, cd, etc.
5. Dépendance	Bibliothèque externe nécessaire au bon fonctionnement du programme (ex : Dash, OR-Tools).
6. Dash	Framework Python utilisé pour créer des interfaces web interactives et réactives.
7. Fichier JSON (.json)	Format texte structuré utilisé pour stocker les données de manière hiérarchique (enseignants, matières, contraintes…).
8. Framework	Ensemble cohérent d'outils permettant de structurer et développer plus facilement une application (ex : Dash pour l'interface).
9. Git / GitHub	Système de gestion de versions (Git) et plateforme de partage de code (GitHub), permettant de suivre l'évolution d'un projet et de le déployer.
10. Interface web	Interface utilisateur accessible via un navigateur internet, permettant de gérer l'application sans connaissance technique.
11. Navigateur web	Logiciel permettant d'accéder à l'interface utilisateur (ex : Chrome, Firefox, Edge…).
12. OR-Tools	Bibliothèque d'optimisation développée par Google, utilisée pour résoudre des problèmes complexes comme la génération d'un emploi du temps.
13. PATH (variable d'environnement)	Chemin d'accès utilisé par le système pour exécuter des programmes ; ajouter Python au PATH permet d'utiliser python dans le terminal.

14. pip	Gestionnaire de paquets Python permettant d'installer des bibliothèques (ex : pip install dash).
15. Projet TER	Travail d'Étude et de Recherche réalisé dans le cadre d'un cursus universitaire (ici : Master MIASHS).
16. Python	Langage de programmation utilisé pour le développement de l'application. Version 3.10 ou supérieure recommandée.
17. Résolution de contraintes	Méthode consistant à satisfaire un ensemble de conditions logiques, ici appliquée à la construction d'un emploi du temps.
18. Serveur local	Programme lancé sur l'ordinateur de l'utilisateur, permettant d'afficher l'application dans un navigateur via une adresse comme : http://127.0.0.1:8050.
19. Système d'exploitation (OS)	Logiciel principal d'un ordinateur : Windows, macOS ou Linux.
20. Terminal	Interface textuelle permettant d'interagir avec le système d'exploitation. Également appelée Invite de commandes sous Windows.
21. venv (environnement virtuel)	Environnement Python isolé dans lequel les bibliothèques sont installées indépendamment du reste du système.
22. Visualisation	Affichage graphique des résultats (emplois du temps), souvent accompagné de couleurs, de tableaux ou d'aperçus interactifs.

11. Références

Cette section présente l'ensemble des documents et ressources qui ont contribué à la conception de l'application, ainsi que des références utiles pour approfondir certains aspects techniques ou fonctionnels, que ce soit du point de vue de l'utilisateur ou du développeur.

- Documentation Python : Documentation officielle du langage Python, utile pour comprendre l'environnement, les commandes de base et la gestion des paquets.
- Documentation Dash : Référence officielle pour développer des interfaces interactives en Python avec Dash.
- Documentation OR-Tools (Google) : Documentation technique complète sur la bibliothèque OR-Tools, utilisée ici pour résoudre les contraintes d'emploi du temps.
- Dépôt GitHub du projet Emploi du Temps - TER : Dépôt source contenant le code complet de l'application, les fichiers de configuration, et les instructions d'installation.
- Cahier des charges fonctionnel : Document définissant les objectifs et contraintes.

12. Index

Voici la liste les mots-clés du document et où les trouver dans celui-ci :