

Cahier de Recette

*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

Cahier de recette

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 : 1.02	Validé le :
Date du document : 12/01/2025	Soumis le : 12/01/2025
Auteur(s) : Arbaut Jean-Baptiste Deschamps Kylian Duez-Faurie Valentine Eramil Kadir Pilloud Aubry Tropel Célia	Type de diffusion : Document électronique (.pdf)
	Confidentialité :
<i>Les éléments d'authentification :</i>	
Maître d'ouvrage: Aurélie Landry	Chef de projet :
Date / Signature : 13/01/2025	Date / Signature :
	

Sommaire

Sommaire.....	3
1. Introduction.....	4
1.1. Objectifs et méthodes.....	4
1.2. Périmètre de la recette.....	6
1.3. Documents de référence.....	6
2. Guide de lecture.....	8
2.1. Maîtrise d'œuvre.....	8
2.2. Maîtrise d'ouvrage.....	8
3. Concepts de base.....	9
4. Description de la fourniture.....	10
5. Moyen d'essai et outils.....	11
6. Conformité aux spécifications générales.....	13
7. Conformité aux spécifications fonctionnelles.....	15
7.1. Scénario 01 : Gestion des contraintes fortes.....	15
7.2. Scénario 02 : Exportation au format ".edt".....	15
7.3. Scénario 03 : Gestion des contraintes faibles.....	16
7.4. Scénario 04 : Multi-solution pour un emploi du temps.....	17
7.5. Scénario 05 : Modification post-génération.....	17
7.6. Scénario 06 : Gestion des droits d'accès.....	18
8. Conformité aux spécifications d'interfaces.....	19
8.1 Scénario 07 : Ergonomie de l'interface.....	19
8.2 Scénario 08 : Validation de la navigation entre modules.....	19
8.3 Scénario 09 : Accessibilité des interfaces.....	20
8.4 Scénario 10 : Ergonomie des formulaires.....	20
9. Conformité de la documentation.....	22
9.1 Scénario 11 : Validation du guide utilisateur.....	22
9.2 Scénario 12 : Validation de la documentation technique.....	22
10. Glossaire.....	24
12. Références.....	26
13. Index.....	27

1. Introduction

Ce cahier de recette a pour objectif de garantir la conformité des fonctionnalités de l'outil de génération d'emploi du temps aux exigences définies dans le cahier des charges. Il structure les étapes de validation nécessaires, de la gestion des contraintes fortes et faibles à la génération, l'optimisation et l'exportation des emplois du temps.

Le projet repose sur une méthodologie Agile, permettant des livraisons itératives et des ajustements en fonction des retours des utilisateurs et dans notre cas, de la cheffe de projet Mme. Landry. Parmi les étapes clés figurent l'analyse approfondie des besoins, la conception d'une base de données, l'adaptation du solveur OR-Tools pour gérer des contraintes complexes, et le développement d'interfaces utilisateur (ergonomiques).

Le périmètre de la recette inclut la vérification des fonctionnalités telles que la gestion des contraintes, la génération d'emplois du temps, les performances du solveur, et l'ergonomie des interfaces.

1.1. Objectifs et méthodes

Le développement du logiciel a été structuré en plusieurs étapes clés afin de garantir une réalisation efficace et conforme aux exigences du projet. Ces étapes s'appuient donc sur une méthodologie Agile, avec des livraisons progressives et des ajustements itératifs en fonction des retours des parties prenantes.

1. Analyse des besoins :

Cette partie a été réalisée durant la première phase du projet, en parallèle de la réalisation du cahier des charges. Nous avons :

- L'étude des besoins : Analyse approfondie des attentes exprimées par les parties prenantes (enseignants, responsables administratifs et utilisateurs finaux).
- La définition des contraintes : Identification et formalisation des contraintes fortes, moyennes et faibles pour les emplois du temps.
- La sélection de l'algorithme : Le solveur OR-Tools, capable de gérer des contraintes complexes et Open-Source.

2. Conception de la base de données :

Conception de l'architecture de la base de données :

- Création d'une base de données pour gérer les informations relatives aux professeurs, aux salles, aux classes, et aux contraintes.

- Définition d'une architecture permettant d'évoluer par la suite en cas de changement.

3. Développement du solveur :

- Adaptation au projet : Adaptation de l'algorithme / solveur OR-Tools pour notre projet et notre type de données.
- Optimisation des performances : Réduction des temps de calcul pour garantir une génération rapide des emplois du temps, même pour des établissements avec de nombreuses contraintes.

4. Interfaces utilisateur :

- Interface de connexion : en cas d'utilisation par plusieurs établissements, pour avoir accès seulement aux données de l'établissement souhaité.
- Interface de saisie des contraintes :
 - Conception intuitive permettant aux utilisateurs d'ajouter, de modifier ou de supprimer des données.
 - Gestion des priorités (contraintes fortes/moyennes/faibles) via une interface claire.
- Interface de visualisation : Affichage graphique des emplois du temps générés avec des indicateurs de qualité (pourcentage de complétion, contraintes non satisfaites, etc.).
- Exportation : Génération d'un fichier au format ".edt" compatible avec des logiciels de gestion comme Monoposte. Exportation des contraintes dites "classiques" qui peuvent être supportées par un logiciel de gestion d'emploi du temps.

5. Documentation :

- Guide utilisateur : Création d'un manuel expliquant les étapes de prise en main, depuis l'installation jusqu'à l'exploitation des résultats.
- Documentation technique : Description détaillée de l'architecture du logiciel, des algorithmes utilisés, et des formats d'échange de données.

6. Validation et recette :

Cette partie sera détaillée par la suite.

- Tests unitaires : Vérification de chaque composant pour s'assurer de son bon fonctionnement.
- Tests d'intégration : Validation de l'interopérabilité entre les différentes parties du logiciel.
- Scénarios de tests : Utilisation de données réelles pour simuler des cas d'utilisation concrets et vérifier la conformité avec les exigences.

7. Livraison :

Le logiciel est livré sous la forme d'un programme exécutable accompagné de ses fichiers de configuration et de sa documentation. Il est également hébergé sur une plateforme de gestion de version (GitHub ou similaire) pour permettre des mises à jour et un suivi collaboratif.

1.2. Périmètre de la recette

Voici les fonctionnalités que nous devons tester :

1. Gestion des contraintes :

- Ajout, modification et suppression de contraintes dans la base de données.
- Gestion des contraintes fortes (disponibilités des professeurs, capacités des salles, etc.).
- Gestion des contraintes faibles (réduction des heures de permanence, etc.).

2. Génération des emplois du temps :

- Prise en compte des contraintes saisies pour produire une solution.
- Affichage des différentes solutions proposées.

3. Interface utilisateur :

- Ergonomie et intuitivité.
- Gestion des droits d'accès par profil utilisateur (administrateur, enseignant, etc.).

4. Export et compatibilité :

- Export au format “.edt”.
- Importation des contraintes compatibles avec d'autres logiciels.

5. Performances et fiabilité :

- Capacité à traiter des volumes importants de données.
- Temps de calcul raisonnable pour la génération des solutions.

1.3. Documents de référence

Le cahier de recette s'appuie sur plusieurs ressources et documents essentiels pour son élaboration. Voici la liste des références utilisées :

- Cahier des charges : Version 2.02, datée du 12/01/2025, décrivant les besoins fonctionnels et techniques ainsi que les contraintes liées au projet.

- [Documentation technique du solveur OR-Tools](#) : Fournissant les spécifications et détails d'implémentation de l'algorithme utilisé pour la génération des emplois du temps.
- [Exemples de données](#) : Ensemble de données fournies par M. Laffond, directeur du collège, incluant des informations anonymisées pour tester et valider le logiciel dans des conditions réalistes.
- [Documentation Monoposte](#) : cette documentation décrit les fonctionnalités et formats supportés par le logiciel de gestion d'emploi du temps EDT-Monoposte.

2. Guide de lecture

Le guide de lecture permet d'aider les lecteurs à utiliser efficacement ce document en fonction de leur rôle et de leurs responsabilités. Chaque sous-partie est adaptée aux besoins spécifiques des différents types de lecteurs, leur offrant des instructions claires et adaptées à leurs missions.

2.1. Maîtrise d'œuvre

La maîtrise d'œuvre intervient dans la gestion technique du projet. De ce fait, ce document leur permet de suivre scrupuleusement les différents tests prévus afin de garantir la conformité technique des livrables aux exigences définies. En cas d'anomalies ou de dysfonctionnements identifiés lors des tests, il est essentiel de les documenter avec précision dans le but d'assurer leur correction rapide et efficace. Ce processus rigoureux vise à maintenir la qualité et la fiabilité des solutions techniques déployées.

2.2. Maîtrise d'ouvrage

La maîtrise d'ouvrage intervient principalement pour s'assurer que les solutions proposées répondent parfaitement aux besoins exprimés. À travers ce document, les membres de l'équipe maîtrise d'ouvrage peuvent valider les scénarios de test pour vérifier leur alignement avec les attentes initiales. Ils participent également aux tests utilisateurs, un moment crucial pour évaluer l'ergonomie et la pertinence des fonctionnalités. Leur contribution garantit que les solutions livrées sont en adéquation avec les objectifs du projet.

3. Concepts de base

Dans cette partie, les concepts fondamentaux nécessaires à la compréhension du document sont détaillés. L'objectif du projet est de concevoir un outil de génération d'emploi du temps sous contraintes, capable de répondre aux besoins spécifiques des établissements scolaires. Ces contraintes sont divisées en deux catégories principales : *fortes et faibles*.

Les *contraintes fortes* représentent des conditions obligatoires pour garantir la validité des emplois du temps. Elles incluent, par exemple, les indisponibilités des professeurs, les capacités des salles et le respect du nombre d'heures par matières fixé par le programme national. Ces contraintes doivent être entièrement respectées pour chaque solution générée. À l'inverse, les *contraintes faibles* sont des préférences ou des optimisations souhaitées, comme la réduction des heures de permanence ou la préférence pour des cours le matin. Bien qu'importantes, elles peuvent être partiellement satisfaites selon les priorités définies par les utilisateurs.

Pour traiter ces contraintes, un *solveur* sera utilisé. Il s'agit d'un *algorithme* spécialisé dans la résolution de problèmes sous contraintes, capable de proposer des solutions optimisées en fonction des priorités définies. Le solveur sélectionné pour ce projet est *OR-Tools* (proposé par Google), un outil Open Source performant et largement utilisé pour la gestion de contraintes.

L'application s'appuiera sur une *base de données*, conçue pour stocker toutes les informations nécessaires à la génération des emplois du temps, comme les listes de professeurs, les classes, les salles et leurs disponibilités. Cette base de données permettra une gestion dynamique des contraintes et offrira la possibilité de tester différents scénarios en fonction des besoins des établissements.

Enfin, l'outil sera composé d'*interfaces* pour permettre aux utilisateurs d'interagir facilement avec le système. Ces interfaces couvriront plusieurs fonctionnalités, telles que la saisie et la modification des contraintes, la visualisation des solutions générées, et l'exportation des emplois du temps au format “.edt”, compatible avec des logiciels tiers comme Monoposte.

4. Description de la fourniture

L'application sera livrée sous forme d'un programme exécutable compatible avec les systèmes d'exploitation les plus courants (Windows, Mac et Linux). Elle sera accompagnée des éléments suivants :

- La documentation utilisateur : Un guide utilisateur détaillé au format PDF, contenant des instructions sur l'installation, la configuration et l'utilisation de l'application. Les étapes seront aussi détaillées sur les interfaces.
- La documentation technique : Un document technique décrivant l'architecture du système, les principales fonctionnalités, les dépendances et les instructions pour les développeurs souhaitant maintenir ou faire évoluer l'outil. On retrouvera aussi les spécifications des interfaces et les formats supportés (notamment le format .edt).
- l'hébergement sur un dépôt distant : L'application, son code source et la documentation seront disponibles sur une plateforme de gestion de version GitHub, facilitant les mises à jour, les modifications et les retours des utilisateurs.

5. Moyen d'essai et outils

Cette section décrit les moyens et outils mobilisés pour vérifier la conformité du logiciel avec les exigences spécifiées dans le cahier des charges.

Dans un premier temps, voici les outils utilisés :

- Un IDE (Environnement de Développement Intégré) pour la création et la maintenance du code source (exemple : Visual Studio Code).
- Une Plateforme de gestion de version (GitHub) : Utilisée pour centraliser le code, gérer les versions, et collaborer efficacement entre les membres de l'équipe.

Cette liste d'outils est susceptible d'évoluer suivant les retours des premières semaines de développement.

Dans un second temps, la validation de l'application repose sur une méthodologie structurée inspirée des pratiques Agile. Cette approche combine des tests itératifs à chaque étape de développement et une validation finale basée sur des scénarios réels.

Voici les tests permettant de vérifier la conformité :

- Tests fonctionnels :
 - Objectif : Valider chaque fonctionnalité indépendamment pour garantir qu'elle répond aux spécifications.
 - Exemples : Vérification de la saisie des contraintes, de l'exportation des fichiers ".edt", de l'ajout de nouvelles données dans la base.
- Tests d'intégration :
 - Objectif : Vérifier l'interopérabilité entre les différents modules du logiciel (interface utilisateur, solveur, base de données).
 - Méthodologie : Test l'entièreté du processus pour s'assurer que les contraintes saisies sont bien intégrées au solveur et reflétées dans l'emploi du temps généré.
- Tests utilisateurs :
 - Objectif : Recueillir les retours des utilisateurs testeurs sur l'ergonomie, la facilité d'utilisation, et la pertinence des résultats.
 - Méthodologie : Créer des sessions de tests, observer les utilisateurs testeurs et leur faire compléter un questionnaire post-test sur l'ergonomie de l'outil (navigation, difficultés rencontrées, etc.).

- Tests de performance :
 - Objectif : Vérifier que l'application est rapide et fluide, même avec des volumes de données importants.
 - Mesures à tester : Temps de calcul du solveur, temps de chargement des interfaces et capacité à gérer simultanément plusieurs scénarios de contraintes.

6. Conformité aux spécifications générales

Dans cette section, nous allons décrire les vérifications nécessaires pour s'assurer que l'application respecte les spécifications générales définies dans le cahier des charges. Nous précisons les critères de validation et les méthodes employées pour évaluer la conformité du produit final.

1. Respect des contraintes fortes :

L'application doit garantir que toutes les contraintes fortes, telles que les indisponibilités des professeurs ou les capacités des salles, sont intégralement respectées dans chaque emploi du temps généré. Cette conformité sera vérifiée par une analyse systématique des solutions proposées pour s'assurer que ces contraintes sont appliquées sans exception.

2. Satisfaction des contraintes faibles :

Les contraintes faibles, telles que la réduction des heures de permanence, doivent être optimisées dans les scénarios proposés par l'algorithme. Un objectif minimum de satisfaction de 70 % est requis. Cette exigence sera vérifiée par une évaluation des solutions générées, en mesurant le taux de satisfaction des préférences définies par les utilisateurs.

3. Performance de génération :

Le temps de calcul pour générer un emploi du temps doit rester inférieur à 1 heure 30 pour un établissement de taille moyenne comprenant environ 30 enseignants et 500 élèves. Les temps de calcul seront mesurés lors des tests de performance réalisés avec différents volumes de données pour valider cet objectif.

4. Interface utilisateur :

L'interface doit être ergonomique, intuitive et accessible à des utilisateurs de niveaux techniques variés. Toutes les actions principales, telles que la gestion des contraintes, la génération des emplois du temps, et leur exportation, doivent être réalisables en trois clics maximum. Les critères de Bastien & Scapin devront être respectés. Ces exigences seront validées par des tests utilisateurs et une évaluation des parcours.

5. Exportation des emplois du temps :

Les fichiers générés au format ".edt" doivent être compatibles avec des logiciels tiers tels que "Monoposte" d'Index Education. Cette compatibilité sera confirmée en testant l'importation des fichiers dans plusieurs outils tiers pour garantir leur intégrité et leur bon fonctionnement.

6. Indicateurs de qualité :

L'application doit fournir des indicateurs clairs sur la qualité des solutions générées, incluant le pourcentage de complétion, le nombre de contraintes non respectées, et l'identification des contraintes restrictives. Ces indicateurs seront vérifiés à l'aide des rapports détaillés générés par l'interface de visualisation.

7. Conformité légale et réglementaire :

L'application doit respecter le Règlement Général sur la Protection des Données (RGPD) en garantissant l'anonymisation des données sensibles lors des tests et des démonstrations. Cette conformité sera vérifiée par un audit des données manipulées et une documentation claire des processus de protection des données.

8. Méthodes de vérification :

Pour valider les critères ci-dessus, les tests sont organisés en plusieurs phases :

- Les tests fonctionnels permettent de valider chaque fonctionnalité de manière isolée pour garantir qu'elle répond aux spécifications.
- Les tests d'intégration vérifient que les modules (solveur, interface utilisateur, base de données) interagissent correctement entre eux.
- Les tests de performance mesurent les temps de calcul et évaluent la robustesse du logiciel face à des volumes importants de données.
- Les tests utilisateurs recueillent les retours des parties prenantes sur l'expérience utilisateur et la pertinence des résultats.

7. Conformité aux spécifications fonctionnelles

Cette section décrit les scénarios de tests nécessaires pour s'assurer que l'application respecte le périmètre fonctionnel défini lors de la phase de spécification. Chaque scénario est présenté de manière détaillée pour valider les fonctionnalités clés du logiciel.

7.1. Scénario 01 : Gestion des contraintes fortes

Description : Ce test vise à garantir que les contraintes fortes (par exemple, les indisponibilités des professeurs) sont correctement appliquées et respectées dans les emplois du temps générés. Le test sera effectué dans un environnement simulé avec des données de test réalistes.

Contraintes : Ce scénario nécessite l'accès à l'interface de gestion des contraintes et au module de génération des emplois du temps.

Dépendances : La base de données doit être correctement initialisée avec les informations sur les professeurs, les salles, et les classes.

Procédure de test :

1. Se connecter avec un profil administrateur.
2. Ajouter une contrainte forte (exemple : indisponibilité d'un professeur X le mercredi matin).
3. Lancer la génération de l'emploi du temps.
4. Vérifier que la contrainte est respectée dans les solutions proposées.

Données en entrée : Indisponibilité du professeur X le mercredi matin.

Résultat attendu : Toutes les solutions générées respectent intégralement la contrainte forte (obligatoire).

7.2. Scénario 02 : Exportation au format “.edt”

Description : Ce test vérifie que les emplois du temps générés sont exportés dans un format compatible avec le logiciel "Monoposte" et d'autres outils similaires.

Contraintes : Un logiciel tiers compatible avec le format “.edt” doit être disponible pour vérifier l'importation.

Dépendances : La fonctionnalité de génération doit être opérationnelle avant d'effectuer l'exportation.

Procédure de test :

1. Générer un emploi du temps à partir d'un ensemble de contraintes.
2. Exporter l'emploi du temps au format ".edt".
3. Importer le fichier généré dans le logiciel "Monoposte".

Données en entrée : Emploi du temps généré.

Résultat attendu : Le fichier est importé sans erreur dans le logiciel tiers.

7.3. Scénario 03 : Gestion des contraintes faibles

Description : Ce test vérifie que les contraintes faibles sont optimisées dans les solutions proposées par le solveur, tout en respectant les contraintes fortes.

Contraintes : Nécessite des données définissant les contraintes faibles (exemple : minimisation des heures de permanence).

Dépendances : Les contraintes fortes doivent être validées avant ce test.

Procédure de test :

1. Accéder à l'interface de gestion des contraintes.
2. Ajouter une contrainte faible (exemple : réduction des heures de permanence).
3. Configurer le poids de la contrainte comme "faible".
4. Générer un emploi du temps.
5. Évaluer les solutions générées pour vérifier la prise en compte de la contrainte faible.

Données en entrée : Contrainte faible.

Résultat attendu : Les solutions proposées minimisent les heures de permanence tout en respectant les contraintes fortes.

7.4. Scénario 04 : Multi-solution pour un emploi du temps

Description : Ce test assure que l'outil peut proposer plusieurs emplois du temps viables en fonction des priorités définies par l'utilisateur.

Contraintes : Nécessite la saisie de contraintes avec des poids variables.

Dépendances : Les modules de génération et de visualisation des solutions doivent être pleinement fonctionnels.

Procédure de test :

1. Saisir plusieurs contraintes avec des priorités différentes.
2. Lancer la génération des emplois du temps.
3. Vérifier que plusieurs solutions distinctes sont proposées, avec des détails sur leur pourcentage de complétion.

Données en entrée : Ensemble de contraintes.

Résultat attendu : L'outil propose au moins deux solutions distinctes.

7.5. Scénario 05 : Modification post-génération

Description : Ce test vérifie que les contraintes peuvent être modifiées après une génération initiale et que le solveur peut produire une nouvelle solution adaptée.

Contraintes : Les modifications doivent être appliquées sur un emploi du temps existant.

Dépendances : Nécessite la fonctionnalité de gestion des contraintes et le solveur.

Procédure de test :

1. Générer un emploi du temps initial.
2. Modifier une contrainte forte (exemple : ajout d'une indisponibilité pour une salle).
3. Relancer la génération.
4. Vérifier que la nouvelle contrainte est respectée dans la solution mise à jour.

Données en entrée : Emploi du temps généré initialement et contrainte modifiée.

Résultat attendu : La nouvelle solution respecte la contrainte modifiée.

7.6. Scénario 06 : Gestion des droits d'accès

Description : Ce test vise à garantir que les utilisateurs disposent uniquement des accès correspondant à leur rôle : représentant d'un établissement et administrateur. Les droits d'accès doivent permettre de protéger les données sensibles et d'éviter les modifications non autorisées.

Contraintes : Ce scénario nécessite un environnement où plusieurs profils d'utilisateurs sont configurés avec des droits distincts.

Dépendances : La fonctionnalité de gestion des utilisateurs doit être opérationnelle avant de réaliser ce test.

Procédure de test :

1. Se connecter avec un profil utilisateur de type "Représentant d'un établissement".
2. Tenter d'accéder à des données non relatives à son établissement.
3. Se connecter avec un profil utilisateur de type "Administrateur".
4. Modifier une contrainte dans la base.

Données en entrée : Profils configurés pour chaque type d'utilisateur

Résultat attendu : L'utilisateur "Représentant d'un établissement" ne peut pas accéder à des données non relatives à son établissement. L'utilisateur "Administrateur" peut modifier n'importe quelle donnée de la base.

8. Conformité aux spécifications d'interfaces

Cette section détaille les scénarios de tests permettant de vérifier que les interfaces de l'application respectent les spécifications fonctionnelles et ergonomiques définies dans le cahier des charges. L'objectif est de garantir une expérience utilisateur fluide, intuitive, et accessible à tous les profils d'utilisateurs.

8.1 Scénario 07 : Ergonomie de l'interface

Description : Ce test vérifie que toutes les actions principales (gestion des contraintes, génération des emplois du temps, exportation) sont accessibles en un maximum de trois clics.

Contraintes : Nécessite l'utilisation de l'interface dans sa version finale avec toutes les fonctionnalités implémentées.

Dépendances : L'intégration de toutes les interfaces doit être terminée avant ce test.

Procédure de test :

1. Se connecter en tant qu'utilisateur.
2. Accéder à la gestion des contraintes.
3. Lancer la génération d'un emploi du temps.
4. Exporter un emploi du temps au format “.edt”.

Données en entrée : Connexion avec un profil utilisateur valide.

Résultat attendu : Toutes les actions peuvent être réalisées en un maximum de trois clics, sans confusion ni latence.

8.2 Scénario 08 : Validation de la navigation entre modules

Description : Ce test vise à garantir une navigation fluide entre les différents modules (gestion des contraintes, visualisation des solutions, exportation des résultats).

Contraintes : Nécessite une interface fonctionnelle et intégrée avec tous les modules opérationnels.

Dépendances : Les modules de gestion des contraintes, de génération, et d'exportation doivent être finalisés.

Procédure de test :

1. Accéder à l'interface de gestion des contraintes.
2. Passer à l'interface de génération d'un emploi du temps.
3. Naviguer vers l'interface d'exportation.
4. Revenir à la gestion des contraintes.

Données en entrée : Emploi du temps généré avec des contraintes définies.

Résultat attendu : La navigation entre les modules est fluide, avec un temps de transition inférieur à 2 secondes (sans compter la génération de l'emploi du temps).

8.3 Scénario 09 : Accessibilité des interfaces

Description : Ce test vérifie que l'application est accessible aux utilisateurs ayant des besoins spécifiques : mode contraste élevé.

Contraintes : Le mode contraste élevé doit être activé pour tester l'accessibilité.

Dépendances : Les fonctionnalités d'accessibilité doivent être implémentées dans toutes les interfaces.

Procédure de test :

1. Activez le mode contraste élevé dans les paramètres.
2. Effectuez des actions : ajout de contraintes, génération d'un emploi du temps, exportation.

Données en entrée : Paramètres d'accessibilité activés.

Résultat attendu : Les interfaces restent lisibles et utilisables en mode contraste élevé.

8.4 Scénario 10 : Ergonomie des formulaires

Description : Ce test s'assure que les formulaires de saisie (ajout de contraintes, modifications) sont clairs et intuitifs, avec des validations explicites et des messages d'erreur précis.

Contraintes : Les formulaires doivent être testés avec des saisies valides et invalides pour vérifier les validations.

Dépendances : Les modules de gestion des contraintes doivent être entièrement fonctionnels.

Procédure de test :

1. Ajouter une contrainte forte pour une salle en saisissant des données valides.
2. Tenter d'ajouter une contrainte avec des champs incomplets ou des valeurs incorrectes.
3. Ajouter une contrainte faible pour un professeur.
4. Vérifier les messages de confirmation et les éventuels messages d'erreur.

Données en entrée : Données de saisie valides et invalides.

Résultat attendu : Les formulaires sont validés sans ambiguïté, et les erreurs sont signalées clairement avec des instructions pour les corriger.

9. Conformité de la documentation

Cette section détaille les tests permettant de vérifier que la documentation fournie avec l'application est complète, claire et conforme aux besoins des utilisateurs et des développeurs. Elle inclut le guide utilisateur et la documentation technique, qui doivent être adaptés aux différents profils d'utilisateurs.

9.1 Scénario 11 : Validation du guide utilisateur

Description : Ce test vise à s'assurer que le guide utilisateur est suffisamment détaillé pour permettre à un utilisateur non technique de prendre en main l'application sans assistance.

Contraintes : Nécessite un environnement de test où l'utilisateur suit exclusivement les instructions du guide.

Dépendances : Le guide utilisateur doit être finalisé et couvrir l'ensemble des fonctionnalités du logiciel.

Procédure de test :

1. Suivre les instructions du guide utilisateur pour installer l'application.
2. Configurer une contrainte forte (exemple : indisponibilité d'un professeur).
3. Générer un emploi du temps.
4. Exporter l'emploi du temps au format “.edt”.

Données en entrée : Guide utilisateur et application installée.

Résultat attendu : Toutes les étapes doivent être réalisées sans confusion, et le guide répond à la quasi-totalité des questions des utilisateurs.

9.2 Scénario 12 : Validation de la documentation technique

Description : Ce test vérifie que la documentation technique est suffisante pour qu'un développeur tiers comprenne l'architecture de l'application et puisse effectuer des modifications ou des ajouts.

Contraintes : Nécessite un environnement de test où un développeur tiers effectue les modifications en s'appuyant uniquement sur la documentation technique.

Dépendances : La documentation technique doit être complète et inclure les sections sur l'architecture, les algorithmes, et les dépendances.

Procédure de test :

1. Analyser la section de la documentation décrivant le solveur.
2. Modifier une contrainte du programme national (contrainte forte et globale à tous les établissements).
3. Tester cette contrainte en générant un emploi du temps.

Données en entrée : Documentation technique complète et environnement de développement configuré.

Résultat attendu : La contrainte ajoutée est fonctionnelle, et la documentation permet de comprendre et d'implémenter les modifications sans ambiguïté.

10. Glossaire

Algorithme : Suite d'instructions ou de règles logiques permettant de résoudre un problème donné, comme la génération d'emplois du temps dans ce projet.

Base de données : Structure organisée pour stocker et gérer les informations nécessaires à l'application, notamment les données sur les professeurs, les salles, les classes, et les contraintes.

Contraintes fortes : Conditions incontournables pour la génération d'un emploi du temps valide, telles que les indisponibilités des professeurs ou les capacités des salles. Leur respect est obligatoire pour chaque solution générée.

Contraintes faibles : Préférences ou optimisations souhaitées dans un emploi du temps, comme la réduction des heures de permanence. Elles ne sont pas obligatoires mais sont prises en compte en fonction de leur priorité.

Critères de Bastien & Scapin : Les critères ergonomiques de Bastien et Scapin sont des principes définis pour évaluer et concevoir des interfaces utilisateur de manière ergonomique dans le but d'améliorer l'expérience utilisateur. Ils sont regroupés en huit grandes catégories : Guidage, Charge de travail, Contrôle explicite, Adaptabilité, Gestion des erreurs, Homogénéité / Cohérence, Signifiante des codes / dénominations et Compatibilité. Voici un [lien](#) expliquant chaque catégorie.

Emploi du temps : Organisation planifiée des cours pour un établissement scolaire, prenant en compte les disponibilités des ressources (salles, enseignants) et les contraintes pédagogiques.

Exportation : Action de sauvegarder les emplois du temps générés au format **.edt**, compatible avec des logiciels tiers comme Monoposte.

Indicateurs de qualité : Mesures fournies par l'application pour évaluer la qualité des emplois du temps générés, incluant le pourcentage de complétion et le nombre de contraintes respectées.

Interface utilisateur : Ensemble des écrans et outils interactifs permettant aux utilisateurs d'interagir avec l'application pour saisir des contraintes, visualiser les résultats et exporter les emplois du temps.

Monoposte : Logiciel tiers utilisé pour gérer les emplois du temps dans les établissements scolaires. L'application doit générer des fichiers compatibles avec ce logiciel.

OR-Tools : Solveur open source utilisé pour gérer les contraintes complexes et proposer des solutions optimisées dans le cadre de ce projet.

RGPD (Règlement Général sur la Protection des Données) : Règlement européen visant à protéger les données personnelles. L'application doit garantir la conformité à ces règles en anonymisant les données sensibles.

Scénario de test : Suite d'étapes prédéfinies permettant de vérifier que l'application répond aux spécifications et satisfait les critères de validation.

Solveur : Algorithme spécialisé utilisé pour résoudre des problèmes sous contraintes. Il génère des solutions optimales en fonction des priorités définies.

Temps de calcul : Durée nécessaire au solveur pour générer un emploi du temps à partir des contraintes saisies.

Test fonctionnel : Test permettant de valider qu'une fonctionnalité de l'application fonctionne indépendamment des autres.

Test d'intégration : Test visant à vérifier le bon fonctionnement et l'interaction entre les différents modules de l'application.

Test utilisateur : Phase de validation impliquant des utilisateurs finaux (enseignants, administrateurs) pour évaluer l'ergonomie et la pertinence des résultats.

Test de performance : Évaluation de la rapidité et de la fluidité de l'application, notamment la durée de calcul du solveur pour des volumes importants de données.

Visualisation : Fonctionnalité de l'application permettant d'afficher graphiquement les emplois du temps générés ainsi que les indicateurs de qualité associés.

12. Références

Les références bibliographiques apportant des informations complémentaires :

- Cahier des charges :
 - Version 2.02, datée du 12 janvier 2025.
 - Document définissant les besoins fonctionnels, les contraintes techniques, et les objectifs du projet.
- Plan de Développement :
 - Version 1.01, datée du 12 janvier 2025.
 - Document décrivant l'organisation, les méthodes, et les outils utilisés pour la réalisation du projet..
- [Documentation technique OR-Tools](#) : Spécifications et détails d'implémentation de l'algorithme utilisé pour la génération des emplois du temps.
- [Documentation Monoposte d'Index Éducation](#) : Référentiel décrivant les fonctionnalités et formats pris en charge par le logiciel de gestion d'emploi du temps Monoposte.
- Exemples de données fournies par M. Laffond : Jeux de données anonymisées pour tester et valider les fonctionnalités de l'outil dans des conditions réalistes (disponible en annexe du cahier des charges).
- Critères ergonomiques Bastien & Scapin :
 - Description : Ensemble de principes utilisés pour évaluer l'ergonomie des interfaces utilisateur.
 - Lien de la définition : [Critères Bastien & Scapin](#).
- Outils collaboratifs utilisés :
 - [GitHub](#) : Dépôt pour la gestion de version et la collaboration sur le code source du projet.
 - [Diagramme de Gantt](#) : Suivi des tâches et organisation.
 - [Trello](#) : outil de gestion de projet.

13. Index

Mot-clé	Pages	Sections
Algorithme	4, 9, 24	Introduction, Concepts de base, Glossaire
Base de données	4, 9, 10, 25	Concepts de base, Description de la fourniture, Glossaire
Cahier des charges	6, 25	Documents de référence, Références
Contrainte forte	9, 15, 24	Concepts de base, Scénarios, Glossaire
Contrainte faible	9, 16, 24	Concepts de base, Scénarios, Glossaire
Documentation Monoposte	6, 25	Documents de référence, Références
Documentation technique	10, 22, 25	Description de la fourniture, Validation de la documentation, Références
Emploi du temps	4, 12, 24	Introduction, Description de la fourniture, Glossaire
Ergonomie	15, 19, 20	Scénarios d'interfaces, Glossaire
Exportation	4, 15, 25	Introduction, Scénarios, Références
Indicateurs de qualité	13, 24	Spécifications générales, Glossaire
Interface utilisateur	9, 19, 24	Concepts de base, Scénarios d'interfaces, Glossaire
Monoposte	15, 24, 25	Scénarios, Glossaire, Références
OR-Tools	4, 9, 25	Introduction, Concepts de base, Références
RGPD	13, 24	Spécifications générales, Glossaire
Scénarios de test	15, 22, 24	Scénarios, Validation de la documentation, Glossaire
Solveur	4, 9, 25	Concepts de base, Références
Test fonctionnel	11, 24	Moyens d'essai, Glossaire
Test utilisateur	11, 24	Moyens d'essai, Glossaire