v1.0 Projet CY-SIAO

FILIERE ING1-GI • 2024-2025
AUTEURS E.ANSERMIN – R. GRIGNON
E-MAILS eva.ansermin@cyu.fr – romuald.grignon@cyu.fr

DESCRIPTION DU PROJET

- Les **SIAO** sont des dispositifs départementaux qui assurent des missions comme l'hébergement d'urgence et l'orientation sociale pour des personnes en difficulté. Dans ce contexte, ce projet vous demande de développer une application graphique qui va permettre de gérer l'hébergement de personnes physiques dans un centre dédié.
- Votre application pourra donc créer la topologie des lits à disposition dans le centre, affecter les lits à des personnes, et fournir un état des lieux en temps réel de la capacité d'accueil.
- Toutes ces données seront stockées dans une base de données annexe. Il faudra donc mettre en place un tel serveur. Il n'est pas demandé que ce serveur soit distant, il peut être hébergé sur la machine sur laquelle est exécutée l'application.
- Votre application permettra de créer, modifier et/ou supprimer les différentes entrées de la base de données. L'utilisateur pourra également affecter des personnes aux lits disponibles.
- L'application fournira un système de proposition d'affectation : en entrant le nombre de personnes à héberger, les dates mini et maxi, les genres et ages des personnes, le système trouvera, si cela est possible, les affectations de lits possibles. Cette fonctionnalité évitera à l'opérateur de rechercher les disponibilités manuellement.
- Enfin graphiquement, il faudra que cette application puisse afficher la synthèse de la capacité d'hébergement, avec un code couleur pour chaque lit en fonction de la durée d'occupation. C'est à vous de définir la mise en page qui vous semble la plus adéquate pour répondre au cahier des charges.

INFORMATIONS GENERALES

- **Taille de l'équipe**
  Ce projet est un travail d'équipe. Il est autorisé de se réunir en groupe de 5 personnes. Si le nombre d'étudiants n'est pas un multiple de 5 et/ou si des étudiants n'arrivent pas à constituer des groupes, c'est aux chargés de projet de statuer sur la formation des groupes. Pensez-donc à anticiper la constitution de vos groupes.

- **Démarrage du projet et jalons**
  Vous obtiendrez de plus amples informations quant aux dates précises de rendu, de soutenance, les critères d'évaluation, le contenu du livrable, …, quand le projet démarrera officiellement.
  Quel que soit le planning initial du projet, vous veillerez à planifier plusieurs rendez-vous d'avancement avec votre chargé(e) de projet. C'est à votre groupe de prendre cette initiative. L'idéal étant de faire un point 1 ou 2 fois par semaine mais cette fréquence est laissée libre en fonction des besoins identifiés avec le tuteur de projet.

- **Versions de l'application**
  Pour éviter d'avoir un projet non fonctionnel à la fin, il vous est demandé d'avoir une version en ligne de commande fonctionnelle. Ceci vous permettra de tester votre modèle de données indépendamment de l'interface graphique. C'est la version avec l'interface graphique JavaFX qui sera bien entendu évaluée, mais dans le cas où certaines fonctionnalités ne seraient pas visibles avec cette interface, vous devez pouvoir présenter toutes les fonctionnalités de votre code Java en ligne de commande.

- **Dépôt de code**
  Vous devrez déposer la totalité des fichiers de votre projet sur un dépôt central Git. Il en existe plusieurs disponibles gratuitement sur des sites web comme ***github.com*** ou ***gitlab.com***. La fréquence des commits sur ce dépôt doit être au minimum de 1 commit / 2 jours.
- **Rapport du projet**
  Un rapport écrit est requis, contenant une brève description de l'équipe et du sujet. Il décrira les différents problèmes rencontrés, les solutions apportées et les résultats. L'idée principale est de montrer comment l'équipe s'est organisée, et quel était le flux de travail appliqué pour atteindre les objectifs du cahier des charges. Le rapport du projet peut être rédigé en français.
  Ce rapport contiendra en plus les éléments techniques suivants : un document de conception UML (diagramme de classe) de votre application, et un document montrant les cas d'utilisations.

- **Démonstration**
  Le jour de la présentation de votre projet, votre code sera exécuté sur la machine de votre chargé(e) de TD. La version utilisée sera la **dernière** fournie sur le dépôt Git **avant** la date de rendu. Même si vous avez une nouvelle version qui corrige des erreurs ou implémente de nouvelles fonctionnalités le jour de la démonstration, c'est bien la version du rendu qui sera utilisée.
  En parallèle, il vous faudra obligatoirement une deuxième machine avec votre application fonctionnelle car une partie de votre groupe aura des modifications de code à faire pendant que l'autre partie fera la présentation/démonstration de votre projet.

- **Organisation de l'équipe**
  Votre projet sera stocké sur un dépôt git (ou un outil similaire) tout au long du projet pour au moins trois raisons :
  - éviter de perdre du travail tout au long du développement de votre application
  - être capable de travailler en équipe efficacement
  - partager vos progrès de développement facilement avec votre chargé(e) de projet.

  De plus il est recommandé de mettre en place un environnement de travail en équipe en utilisant divers outils pour cela (Slack, Trello, Discord, …).

CRITERES GENERAUX

- Le **but principal** du projet est de fournir une **application fonctionnelle** pour l'utilisateur. Le programme doit correspondre à la description et aux fonctionnalités de ce document .
- Votre code sera généreusement **commenté**.
- Tous les éléments de **votre code** (variables, fonctions, commentaires) seront écrits **en anglais** obligatoirement.
- Votre code devra être commenté de telle manière que l'on puisse utiliser un outil de génération de documentation automatique de type **JavaDoc**. Un dossier contenant la doc générée par vos soins sera présent dans votre dépôt de code lors de la livraison.
- Votre projet doit être utilisable au clavier ou à la souris en fonction des fonctionnalités nécessaires.
- Votre application ne doit jamais s'interrompre de manière **intempestive** (crash), ou tourner en boucle indéfiniment, quelle que soit la raison.
  Toutes les erreurs doivent être gérées correctement. Il est préférable de d'avoir une application stable avec moins de fonctionnalités plutôt qu'une application contenant toutes les exigences du cahier des charges mais qui plante trop souvent.
  Une application qui se stoppe de manière imprévue à cause d'une exception, par exemple, sera un événement très pénalisant.
- Votre application devra être **modulée** afin de ne pas avoir l'ensemble du code dans un seul et même fichier par exemple. Apportez du soin à la conception de votre projet avant de vous lancer dans le code.
- Le livrable fourni à votre chargé(e) de TD sera simplement l'**URL** de votre **dépôt Git** accessible **publiquement**.

FONCTIONNALITES DU PROJET

- Votre programme devra gérer des personnes avec différentes informations les concernant (patronyme, genre, date et ville de naissance, numéro de sécurité sociale, une liste des adresses connues, …)
- Votre programme devra gérer des pièces de l'établissement (des salles) contenant chacune un ou plusieurs "lits". Chaque salle et chaque lit possède un identifiant unique, et possède un intitulé optionnel (exemple "Gymnase principal", "Salle Soleil", "Dortoir Ouest", … . Les lits possèdent en plus le nombre de places (1 ou 2 généralement) qu'ils peuvent accueillir.

- Chaque pièce peut avoir un accès restreint à des personnes d'un certain genre (par exemple des pièces pouvant accueillir seulement des femmes) et/ou avoir un accès restreint à une certaine plage d'âge (par exemple pour indiquer que certaines salles ne peuvent accueillir que des mineurs ou que des majeurs). Toutes les restrictions peuvent être combinées (par exemple une salle n'accueille que des hommes majeurs (ET logique) et une autre salle n'accueille que des enfants ou des femmes (OU logique).
- Pour tous les éléments fonctionnels (personnes, salles, lits, …) votre programme devra fournir le moyen de créer (ajouter), modifier ou effacer des éléments. Il est impossible d'effacer un élément s'il est référencé par un autre (exemple, effacer une personne si un planning référence cette dernière, ou effacer un lit s'il est lié à une salle).
- Si il y a une modification de restriction d'une salle, le programme doit vérifier que toutes les personnes liées aux lits contenus dans cette salle respectent ces critères : dans le cas contraire, la modification de restriction ne pourra pas être faite tant que ces personnes sont encore dedans.
- Votre programme devra gérer globalement un planning d'occupation des lits. Ce planning contiendra une liste d'informations comme par exemple l'identifiant d'un lit, l'identifiant d'une personne, la date de début et de fin d'occupation du lit, et une information pour indiquer la "sortie" de la personne de l'établissement. Cette dernière information permet d'indiquer si la période d'occupation a besoin d'être remise à jour ou non (date de fin inférieure à la date du jour par exemple, normalement l'information de sortie est à VRAI, dans le cas contraire il y a un souci de gestion des lits et des personnes).
- Les dates de début et de fin d'occupation d'un lit par une personne sont des dates incluses (c'est à dire qu'à partir du début le lit est occupé, et le jour de fin il est encore occupé : il sera disponible le lendemain de la date de fin).
- Dans le planning, chaque entrée peut être modifiée sous condition : les dates de début et de fin peuvent être modifiées si elles sont postérieures ou égales à la date du jour (indépendemment l'une de l'autre). Si les dates sont passées, on n'autorise plus les modifications.
- Votre programme doit proposer un moyen de trouver le créneau le plus proche pour héberer plusieurs personnes d'un certain age et genre pendant une certaine durée.
- Vous devez rentrer une plage de dates (jours/mois/année) puis le nombre de personnes à héberger. Une option permettra de choisir si vous voulez les héberger dans la même salle ou non (cela filtre encore plus la recherche). Enfin il faut renseigner pour chaque personne leur age et genre afin de faire une recherche qui utilise les restrictions d'accès aux salles.

  Cette fonctionnalité doit pouvoir proposer des créneaux de planning qui correspondent aux critères. Votre programme doit pouvoir alors valider ce créneau qui est enregistré dans le planning (lien automatique des personnes, dates, et lits).

  A vous de voir si vous inscrivez d'abord les personnes puis vous faites

une recherche à partir des critères de ces personnes ou bien si vous avez un formulaire spécifique pour faire la recherche AVANT inscription finale.

- Votre application doit pouvoir afficher l'état courant du centre, avec tous les lits/salles et leur état d'occupation avec un code couleur (par exemple : des lits occupés pour encore au moins 2 semaines en rouge, des lits occupés entre 1 et 2 semaines en jaune, moins d'une semaine en vert, et disponibles de suite en blanc).
  Ces durées et couleurs sont fixées ici dans l'exemple, mais il serait intéressant que votre application laisse la possibilité de les modifier dynamiquement, et/ou d'avoir un réel dégradé en fonction de la durée plutôt qu'un nombre discret de couleurs).
- Il est possible que des lits soient occupés pendant une durée relativement longue mais en incluant des "coupures" (exemple : un lit est prévu pour être occupé pendant 14 jours avec une coupure de 3 jours en plein milieu pendant laquelle ce lit est vacant). Il serait intéressant d'afficher ce lit avec la couleur décrite précédemment mais en y ajoutant un motif pour indiquer que la durée d'occupation n'est pas optimale (par exemple des hachures blanches).
  Toutes les propositions faites ici ne sont pas obligatoires : le seul critère est de différencier les durées d'occupation des lits pour avoir une vision claire de l'utilisation du centre.
- Toutes les données de l'application doivent être stockées dans une base de données (BDD). Votre programme devra donc faire l'interface avec la BDD. Vous créerez tous les liens et jointures entre les tables pour répondre aux exigences décrites précédemment.
- Lors du rendu final, vous devrez fournir un fichier d'export de votre BDD (structure : tables et des champs) ainsi qu'un export léger de quelques données utilisables (données des tables). Ces exports peuvent être fait directement depuis votre client de gestion de base de données.

RESSOURCES UTILES

- **Github**
  https://www.github.com
  https://docs.github.com/en/get-started/quickstart/hello-world
- **Patrons de conception**
  https://fr.wikipedia.org/wiki/Patron_de_conception
- **Sérialisation**
  https://fr.wikipedia.org/wiki/Sérialisation
- **Connexion à une base de données**
  https://docs.oracle.com/javase/tutorial/jdbc/basics/connecting.html
- **Les centres d'hébergement d'urgence**
  https://fr.wikipedia.org/wiki/Centre_d%27h%C3%A9bergement_d%27urgence