



Git du projet :

https://github.com/Denver-2024/Projet_genie_logiciel_groupe_16_CY-SIAO

David EBAKOU
Yacine KHELIL
Orlan MONJOLY
Ulrich RANAIVOJAONA
Abdelah SAIDJ

Sommaire

I.	Introduction et contexte.....	page 3
II.	Répartition des tâches.....	page 5
III.	Conception du projet.....	page 6
IV.	Architecture du projet.....	page 8
V.	Interface utilisateur.....	page 12
VI.	Gestion des données.....	page 15
VII.	Problèmes rencontrés et solutions apportées.....	page 16
VIII.	Conclusion et perspectives.....	page 17

Introduction et contexte

Choix du sujet :

Nous avons choisi le projet **CY-SIAO** dans le cadre de la gestion d'hébergement d'urgence et de l'orientation sociale pour des personnes en difficulté. Ces dispositifs départementaux ont pour tâches de centraliser les demandes d'hébergement, d'optimiser l'occupation des places disponibles ainsi que de faciliter l'accompagnement social des publics concernés.

Ce projet s'inscrit dans ce contexte en proposant le développement d'une application graphique permettant de gérer l'hébergement de personnes physiques dans un centre dédié. L'objectif est de concevoir un outil graphique intuitif modélisant la répartition des lits, d'attribuer les places disponibles selon certains critères (âge, genre, disponibilités...), de visualiser en temps réel la capacité d'accueil et de générer une synthèse graphique de l'état d'occupation.

Afin d'assurer le bon fonctionnement de l'application, une base de données hébergée dans le cloud est mise en place pour stocker les données nécessaires à la gestion des lits et des personnes hébergées. Des fonctionnalités d'édition des données ainsi qu'un système d'affectation sont également ajoutées pour simplifier le travail de l'utilisateur.

Outils et technologies utilisés :

Dans l'optique de développer cette application de gestion de l'hébergement, nous avons utilisé les outils et technologies suivants :

- **Langage de programmation :**

Le projet a été développé en Java, un langage orienté objet robuste et adapté aux applications métiers.

- **Interface graphique :**

Nous avons utilisé JavaFX, une bibliothèque d'interface utilisateur pour Java qui permet de concevoir des interfaces graphiques interactives.

- **Base de données :**

Les données de l'application sont stockées dans une base de données distante hébergée sur AlwaysData, un hébergeur permettant de déployer facilement des bases PostgreSQL à distance.

- **Accès aux données (DAO) :**

L'accès aux données s'appuie sur le modèle DAO (Data Access Object). Ce modèle consiste à créer des classes spécifiques chargées d'effectuer les

requêtes vers la base (ajout, suppression, mise à jour, consultation). Il permet ainsi de séparer le code qui manipule les données du reste de l'application.

- **Structure et architecture du projet :**

La structure et l'architecture du projet ont été préparées en amont à l'aide de diagrammes UML, incluant des diagrammes de cas d'utilisation et de classes.

- **Gestion de projet avec Maven :**

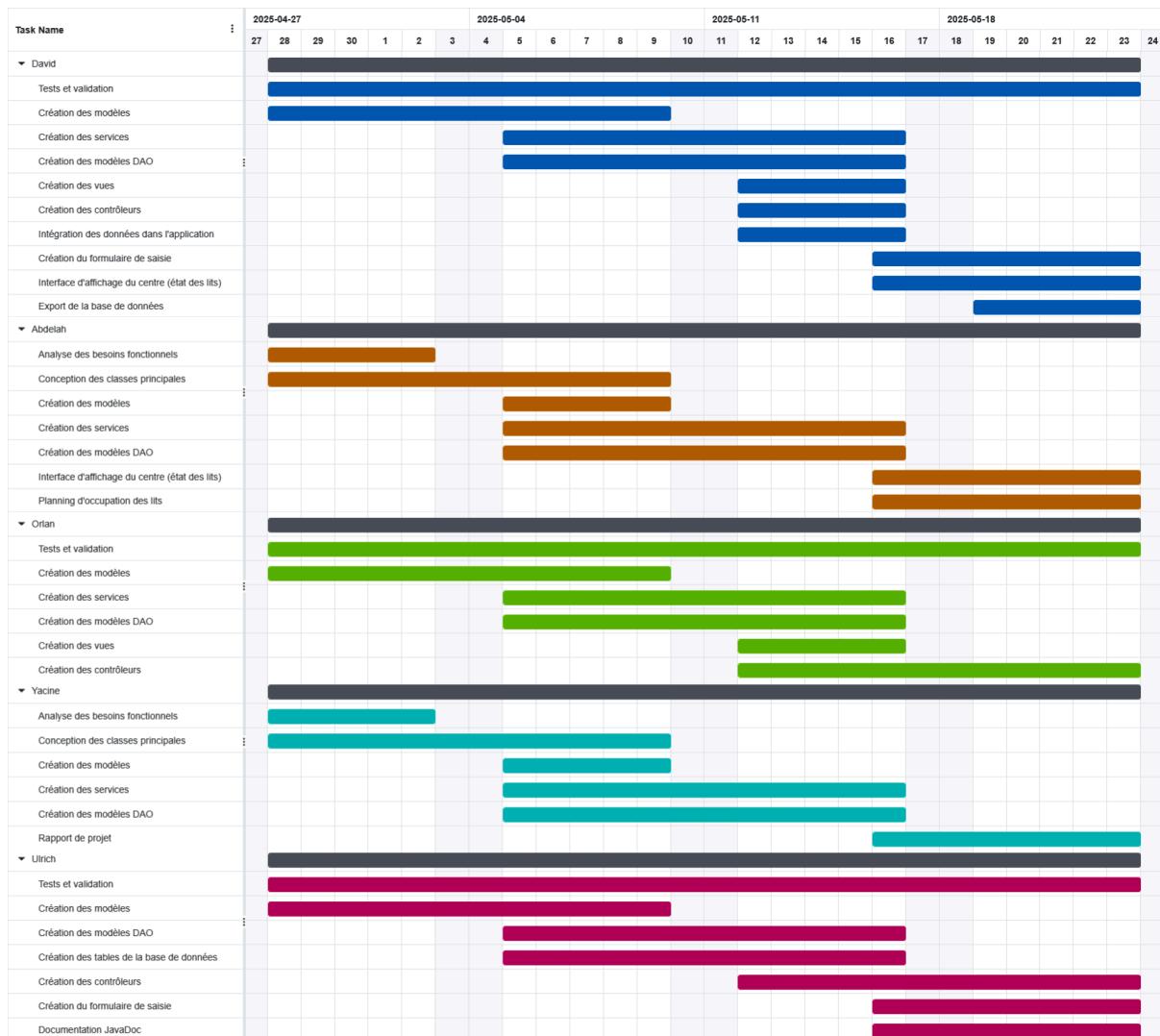
Nous avons utilisé Maven, un outil de gestion de projet Java, pour automatiser la compilation du code, gérer les dépendances, prendre en charge les plugins, standardiser la structure du projet et faciliter l'exécution de l'application.

- **Environnement de développement :**

Le développement a été réalisé à l'aide d'IntelliJ IDEA qui est un IDE dédié au projet Java, facilitant la gestion des dépendances, la compilation et le débogage.

Répartition des tâches

Nous avons réparti les différentes fonctionnalités du projet en plusieurs tâches distinctes afin d'organiser notre travail efficacement et de respecter les délais impartis. Chacun des membres de l'équipe a ainsi pu se concentrer sur ses propres objectifs pour mener à bien notre projet. Nous échangeons régulièrement afin de mettre en commun nos avancées. L'ensemble de ces tâches et leur planification sont présentés dans le **diagramme de Gantt** ci-dessous, qui illustre la chronologie et les dépendances entre les différentes étapes d'avancement.



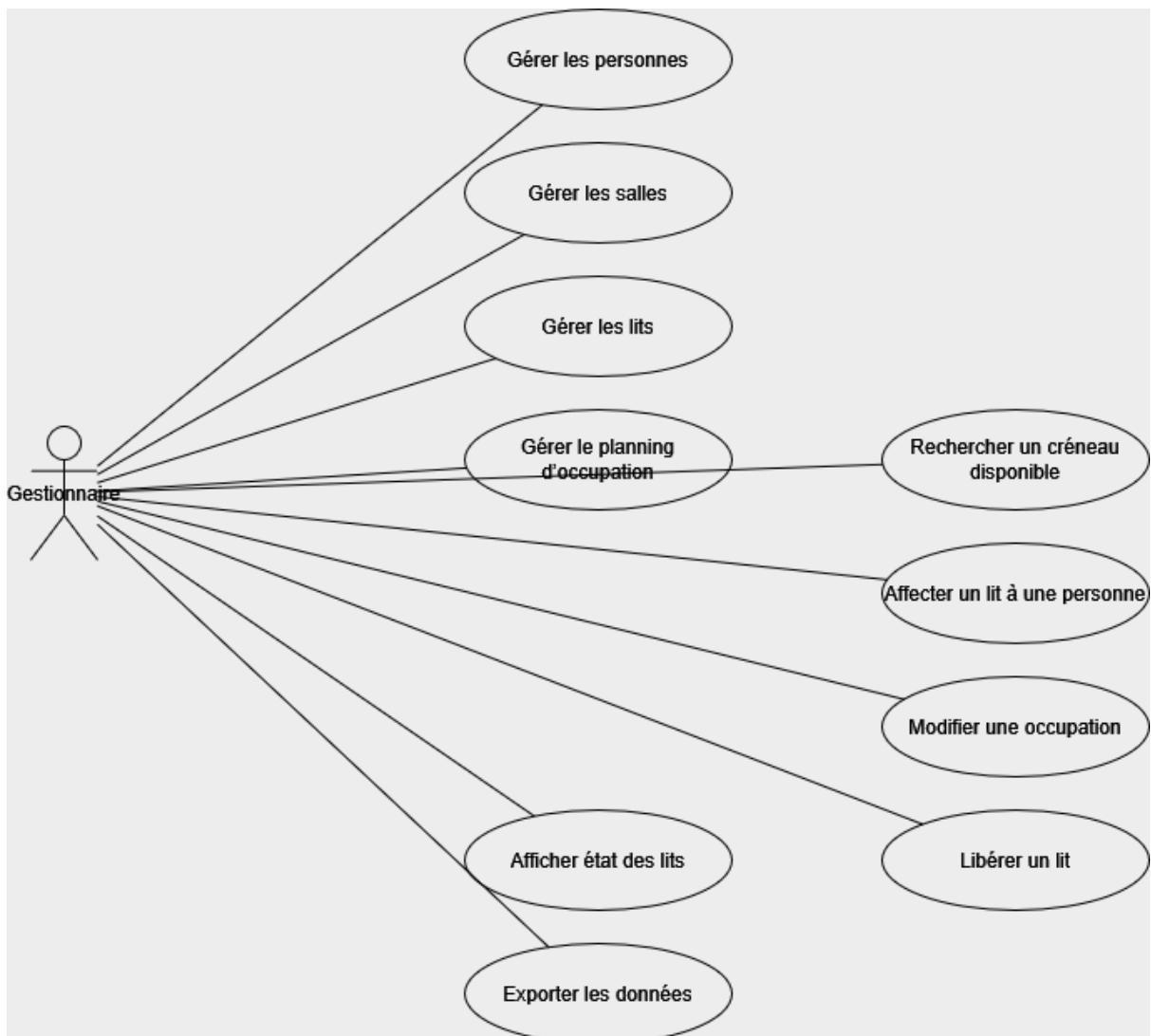
Conception du projet

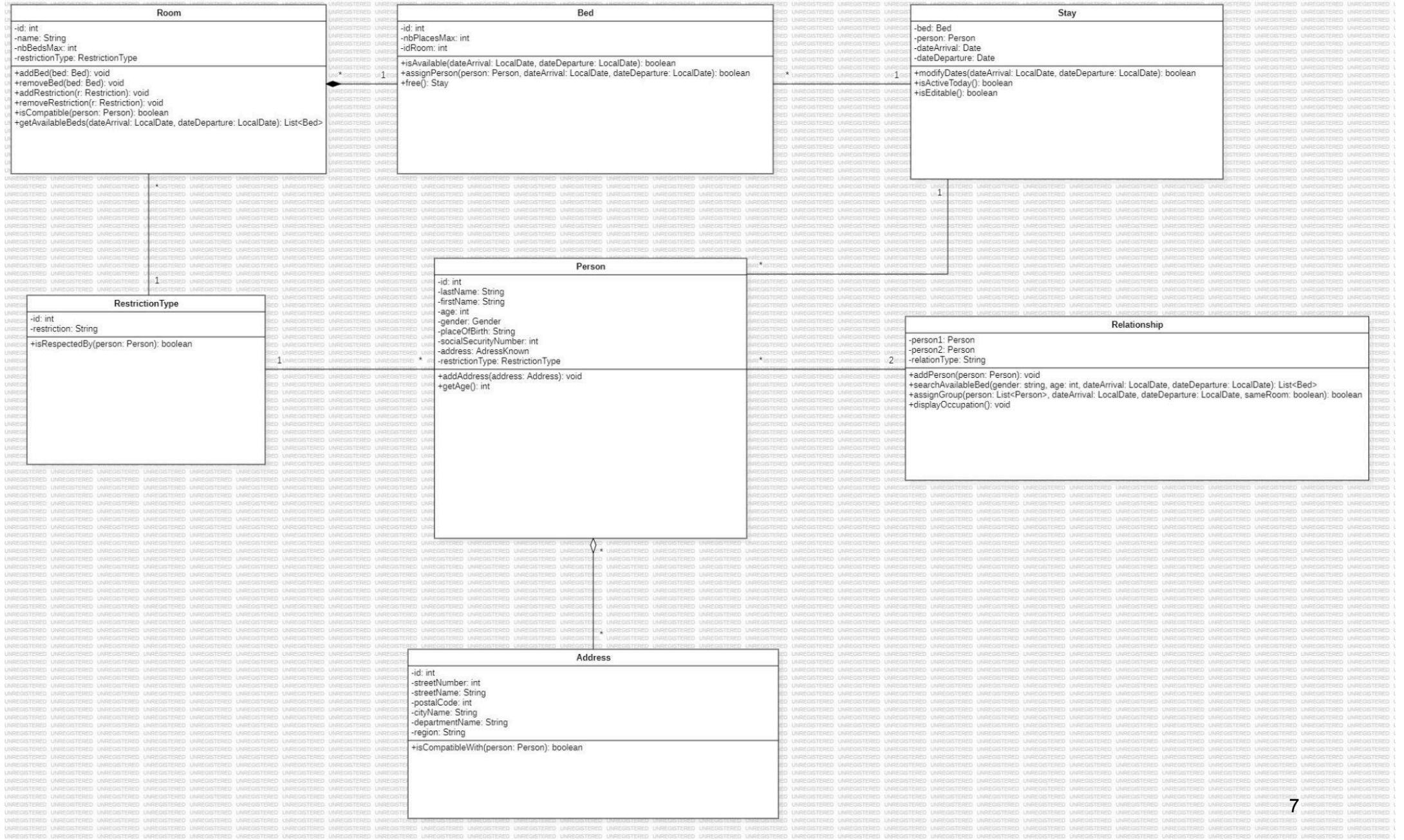
Avant d'entamer la phase de développement, une phase de conception a été réalisée pour poser les bases structurelles de l'application. Cette étape nous a permis de clarifier les besoins du système, d'identifier les entités principales à gérer et leurs interactions.

Deux types de diagrammes UML ont été utilisés pour cette modélisation :

- Le **diagramme de cas d'utilisation** qui présente les différentes fonctionnalités accessibles aux utilisateurs et les interactions possibles avec le système.
- Le **diagramme de classes** qui structure les données et représente les relations entre les entités majeures telles que les personnes, les salles, les lits ou encore le planning d'occupation.

Ces éléments ont servi de références tout au long du projet pour en garantir la cohérence et la robustesse.





Architecture du projet

L'architecture du projet repose sur une organisation claire et modulaire suivant le modèle MVC (Modèle-Vue-Contrôleur), enrichi par une couche de service ainsi que l'utilisation du pattern DAO pour la gestion de la base de données :

```
.  
|-- BDDprojetJava (2).sql  
|-- README.md  
|-- pom.xml  
|-- src  
|   |-- main  
|   |   |-- java  
|   |   |   `-- com  
|   |   |       `-- cy_siao  
|   |   |           |-- App.java  
|   |   |           |-- GuiApp.java  
|   |   |           |-- Main.java  
|   |   |           |-- controller  
|   |   |               |-- GUIController.java  
|   |   |               |-- cli  
|   |   |                   |-- BedController.java  
|   |   |                   |-- CLIController.java  
|   |   |                   |-- PersonController.java  
|   |   |                   |-- RoomController.java  
|   |   |               `-- StayController.java  
|   |   |       |-- gui  
|   |   |           |-- BedControllerFx.java  
|   |   |           |-- DashboardController.java  
|   |   |           |-- MainMenuController.java  
|   |   |           |-- PersonControllerFx.java  
|   |   |           |-- RoomControllerFx.java  
|   |   |               `-- StayControllerFx.java  
|   |   |-- dao
```

```
|   |   |   |   |-- AddressDao.java  
|   |   |   |   |-- BedDao.java  
|   |   |   |   |-- Dao.java  
|   |   |   |   |-- KnowsDao.java  
|   |   |   |   |-- PersonDao.java  
|   |   |   |   |-- RelationshipDao.java  
|   |   |   |   |-- RestrictionRoomDao.java  
|   |   |   |   |-- RestrictionTypeDao.java  
|   |   |   |   |-- RoomDao.java  
|   |   |   |   `-- StayDao.java  
|   |   |   |-- model  
|   |   |   |   |-- Bed.java  
|   |   |   |   |-- Knows.java  
|   |   |   |   |-- RestrictionRoom.java  
|   |   |   |   |-- RestrictionType.java  
|   |   |   |   |-- Room.java  
|   |   |   |   |-- Stay.java  
|   |   |   |   `-- person  
|   |   |   |       |-- Address.java  
|   |   |   |       |-- Gender.java  
|   |   |   |       |-- Person.java  
|   |   |   |       `-- Relationship.java  
|   |   |   |-- service  
|   |   |   |   |-- AddressService.java  
|   |   |   |   |-- BedService.java  
|   |   |   |   |-- EligibilityService.java  
|   |   |   |   |-- PersonService.java  
|   |   |   |   |-- RoomService.java
```

```
| | |     | `-- StayService.java  
| | |     |-- util  
| | |     | `-- DatabaseUtil.java  
| | |     `-- view  
| | |         |-- BedView.java  
| | |         |-- CLIView.java  
| | |         |-- DashboardView.java  
| | |         |-- GuiView.java  
| | |         |-- MainMenuView.java  
| | |         |-- PersonView.java  
| | |         |-- RoomView.java  
| | |         |-- StayView.java  
| | |         `-- ViewManager.java  
| | `-- resources  
| |     |-- Images  
| |     |   |-- BedslImage.png  
| |     |   |-- Projet_SIAO.png  
| |     |   |-- Projet_SIAO_logo.png  
| |     |   |-- fond - Copy.png  
| |     |   `-- fond.jpg  
| |     |-- bed_view.fxml  
| |     |-- config.properties  
| |     |-- dashboard_view.fxml  
| |     |-- mainMenu.fxml  
| |     |-- person_view.fxml  
| |     |-- room_view.fxml  
| |     |-- stay_view.fxml  
| |     |-- styles
```

```
| |   | `-- style.css
| |   '-- test.fxml
| '-- test
|   '-- java
|     '-- com
|       '-- cy_siao
|         '-- AppTest.java
`-- target
  |-- classes
  |
  |-- generated-sources
  |   '-- annotations
  '-- maven-status
    '-- maven-compiler-plugin
      '-- compile
        '-- default-compile
          |-- createdFiles.lst
          '-- inputFiles.lst
```

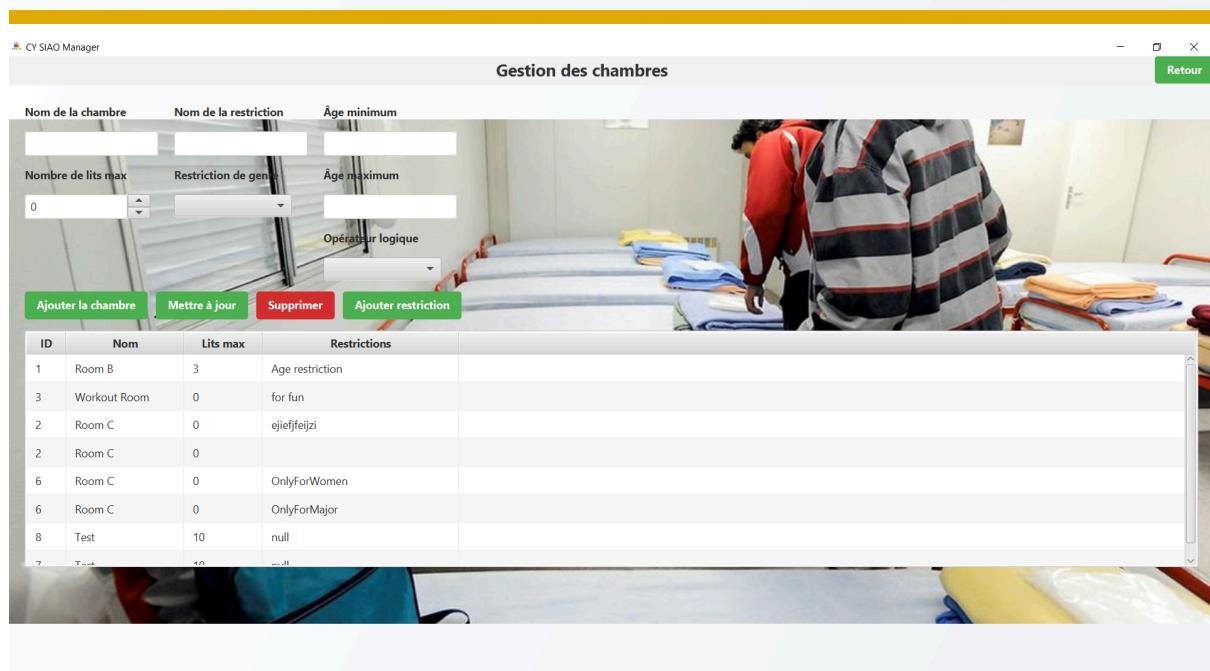
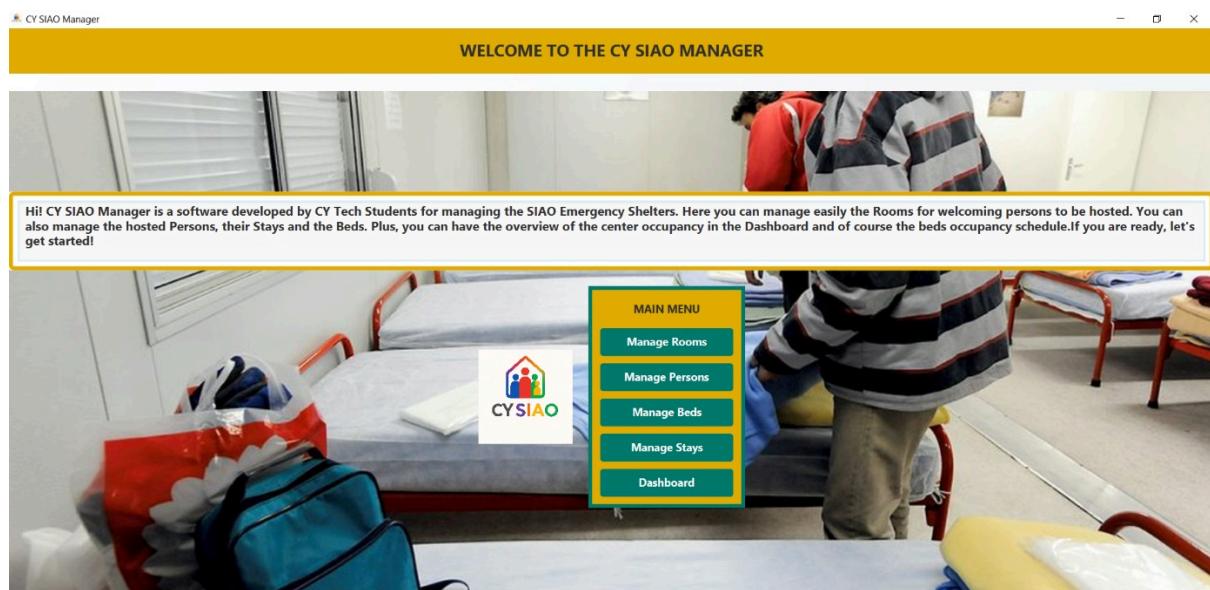
42 directories, 87 files

Interface utilisateur

L'interface utilisateur de l'application a été développée à l'aide de JavaFX, une bibliothèque graphique adaptée aux applications Java. Les différentes vues permettent à l'utilisateur de naviguer facilement entre les écrans de gestion des personnes, des chambres, des séjours ou encore des restrictions.

Chaque vue correspond à une fonctionnalité bien précise de l'application. Les actions utilisateur (ajout, modification, suppression, consultation) sont ainsi directement reliées aux éléments graphiques (boutons, champs, listes), facilitant l'interaction avec les données.

Page d'accueil :



ID	Nom	Lits max	Restrictions
1	Room B	3	Age restriction
3	Workout Room	0	for fun
2	Room C	0	ejiefifeizi
2	Room C	0	
6	Room C	0	OnlyForWomen
6	Room C	0	OnlyForMajor
8	Test	10	null
7	T...-a	10	...

CY SIAO Manager

Gestion des occupants



Prénom Lieu de naissance Numéro de rue Code postal

Nom	Numéro de Sécurité Sociale	Nom de rue	Ville

Âge Genre

--	--

[Ajouter](#) [Mettre à jour](#) [Supprimer](#) [Ajouter adresse](#)

Prénom	Nom	Âge	Genre	Lieu de naissance	N° Sécurité Sociale	Adresse
Victor	Doom	37	Male	Lunas	1234567891234	12 rue, 12563 Lest
Victor	Doom	37	Male	Lunas	1234567891234	15 Bon, 14523 Due
John	Doe	31	Male	Paris	0	10 Main Street, 75001 Lyon
Luc	Bon	18	Male	Mairie	1234567891234	25 Rue du bon, 95000 Cergy
Jane	Doe	28	Female		0	0 null, 0 null

CY SIAO Manager

Room ID :

Is the Bed Double ?

[Add Bed](#)

[Delete Selected Bed](#)

[Retour](#)

ID	Room ID	Nombre de place
1	1	2
3	1	1
5	7	1
6	7	1
7	7	1
8	7	1
9	7	1
14	6	2
15	6	1

[See the Beds Schedule](#)

CY SIAO Manager

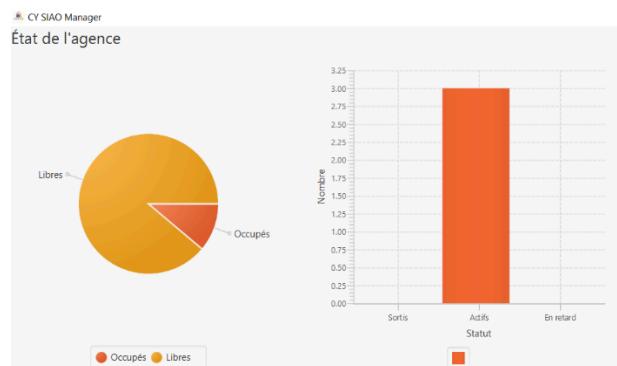
Gestion des Séjours

Date d'arrivée Personne

Date de départ Lit

Ajouter Mettre à jour Supprimer Rechercher

ID	Date arrivée	Date départ	Occupant	ID Lit
1	2025-05-21	2025-05-28	Doe	1
8	2025-05-24	2025-06-03	Doe	1
9	2025-05-24	2025-06-03	Doe	3



Gestion des données

La gestion des données s'appuie sur le modèle DAO (Data Access Object). Pour chaque entité du projet (comme Personne, Chambre ou Séjour), une classe DAO spécifique prend en charge les opérations de création, lecture, mise à jour et suppression (CRUD).

Les interactions avec la base de données s'effectuent via des connexions à une instance PostgreSQL hébergée à distance sur la plateforme AlwaysData. Cela permet de faciliter le travail en équipe et de tester l'application dans un environnement proche des conditions réelles d'utilisation.

La connexion à la base est centralisée dans une classe utilitaire **DatabaseUtil**, qui améliore la lisibilité du code et sa fiabilité.

Problèmes rencontrés et solutions apportées

Durant notre projet, nous avons rencontré plusieurs difficultés techniques. En premier lieu, des erreurs de connexion se sont manifestées concernant la base de données ce qui a nécessité de revoir les droits d'accès et les ports notamment. Sur GitHub, nous avons eu des conflits Git, ce qui nous a poussé à travailler sur plusieurs branches et en faisant des fusions planifiées. Deuxièmement, le réajustement en quelques reprises des modélisations (MCD, Diagramme de classe, ...) nous a pris un peu de temps. Les tests d'intégrité et les corrections de bugs nous ont également pris du temps. Cependant, nous avons utilisé les méthodes deux par deux : deux codent puis deux autres testent ou vérifient et vice-versa. Puis, la mise à jour des données en temps réel dans les interfaces JavaFX n'a pas toujours été fluide ce qui a nécessité l'utilisation d'observables et de méthodes *refresh()*. Enfin, la gestion ainsi que la répartition des tâches a parfois conduit à des réajustements afin de répondre aux exigences du cahier des charges.

Conclusion et perspectives

Ce projet nous a permis de mettre en pratique nos connaissances en Java orienté objet, en conception UML, ainsi qu'en gestion de base de données avec le modèle DAO notamment. Grâce à une répartition des tâches efficace et à l'utilisation d'outils adaptés tels que IntelliJ IDEA, Maven et GitHub, nous avons pu mener à bien le développement de notre application d'hébergement d'urgence.

Plusieurs axes d'amélioration sont envisageables comme l'ajout d'un système d'authentification utilisateur ou encore une interface web afin de rendre l'application accessible à distance.