FACULTE DES SCIENCES BEN M'SICK
UNIVERSITÉ HASSAN II DE CASABLANCA

PROJET DE FIN DE MODULE

Système de Gestion des résidences Etudiantes

MDSBD

Réalisé Par :

Amine BAQUOUCH

Examiné par :

Dr. Abdessamad BELANGOUR

Année universitaire : 2024/2025

Résumé :

L’objectif de ce projet est de développer une application web de gestion des résidences étudiantes, permettant de simplifier et centraliser les tâches administratives. Cette solution numérique vise à répondre aux défis rencontrés dans la gestion traditionnelle, tels que le suivi manuel des chambres, des paiements et des requêtes de maintenance.

L’application propose des fonctionnalités clés, telles que la gestion des chambres (ajout, modification, suppression), l’attribution des chambres aux résidents, le suivi des paiements avec génération de reçus, ainsi que le traitement des requêtes de maintenance signalées par les étudiants. Grâce à l’intégration des technologies modernes comme **Spring Boot**, **Thymeleaf**, **Bootstrap**, et une base de données **MySQL**, l’application offre une interface intuitive et une performance optimale.

Ce rapport détaille les différentes étapes de réalisation, depuis l’analyse des besoins jusqu’à l’implémentation et les tests, tout en proposant des perspectives d’amélioration comme l’ajout de paiements en ligne via Stripe. L'application a pour ambition d'améliorer l'efficacité de la gestion des résidences tout en offrant une expérience utilisateur fluide et moderne.

Abstract:

This project aims to develop a web application for managing student residences, simplifying and centralizing administrative tasks. This digital solution addresses challenges faced in traditional management, such as manual tracking of rooms, payments, and maintenance requests.

The application provides key features, including room management (add, update, delete), room assignment to residents, payment tracking with receipt generation, and handling maintenance requests submitted by students. By leveraging modern technologies such as **Spring Boot**, **Thymeleaf**, **Bootstrap**, and a **MySQL** database, the application delivers an intuitive interface and optimal performance.

This report details the various implementation stages, from requirements analysis to testing, while suggesting future enhancements, such as online payments via Stripe. The application aims to enhance the efficiency of residence management while providing a seamless and modern user experience.

Table de Matière :

Liste des abréviations

MVC	**Model View Control**
SQL	**Structured Query Language**
XSS	**Cross-Site Scripting**

Introduction Générale

Les ordinateurs jouent un rôle crucial dans notre quotidien. Ils simplifient les échanges, offrent l'accès à une multitude d'informations et rendent les tâches professionnelles et éducatives plus simples. Grâce à leurs capacités, les ordinateurs sont devenus des outils essentiels pour la productivité, le divertissement et la créativité. Il y a une grande variété de choix en matière d'ordinateurs personnels, ce qui peut rendre la recherche du modèle idéal pour une utilisation spécifique et un budget serré, tout en étant technique. Entre les différentes marques, les configurations variées et les spécifications techniques comme le processeur, la mémoire RAM, le stockage et la carte graphique, il est facile de se sentir submergé.

Cette abondance d'options nécessite une réflexion approfondie et une compréhension des besoins individuels pour naviguer dans ce marché complexe. C'est pourquoi dans ce projet nous avons décidé de créer une application web "PC4U" qui facilitera le processus de choix de son prochain PC en fournissant des résultats clairs et faciles à comprendre.

Ce projet a été rendu possible grâce à l'intelligence artificielle et à l'apprentissage automatique, utilisant des modèles de prédiction et des méthodes de traitement des données pour atteindre les résultats souhaités.

I. Contexte générale du projet

1.1 Introduction

Le contexte de ce projet est crucial pour comprendre son importance et sa capacité à résoudre les problèmes actuels de gestion des résidences étudiantes. Actuellement, les administrations sont confrontées à une fragmentation des outils, utilisant souvent des feuilles Excel, des emails ou des systèmes non intégrés, ce qui entraîne des incohérences dans les données et un manque de visibilité en temps réel sur des éléments clés comme les chambres disponibles ou les paiements en retard. De plus, le suivi inefficace des incidents de maintenance et la gestion manuelle fastidieuse, comme l'attribution manuelle des chambres, provoquent des retards et des erreurs. Ce projet propose une solution numérique centralisée, regroupant toutes les informations dans une base de données unique, accessible en temps réel, et intégrant des technologies modernes comme les notifications automatiques et des interfaces intuitives pour simplifier les tâches courantes. En permettant une analyse approfondie des données (paiements, taux d'occupation, historique des incidents), l'application offre une meilleure prise de décision stratégique et prépare l'administration à gérer efficacement un nombre croissant de résidences et d'étudiants à l'avenir, tout en optimisant les processus pour une évolutivité future.

1.2 Problématique du projet

La gestion des résidences étudiantes représente un défi majeur pour les administrations, en raison du volume croissant de données à traiter et des attentes élevées des résidents en matière de services modernes et accessibles. Les processus manuels ou les outils obsolètes conduisent souvent à des inefficacités, telles que :

- La difficulté à suivre l'occupation des chambres en temps réel.
- Une gestion complexe des paiements et des retards de loyer.
- Un traitement lent des requêtes de maintenance, entraînant l'insatisfaction des résidents.
- Un manque de centralisation des informations, rendant la collaboration entre les administrateurs laborieuse.

Dans ce contexte, il est impératif de développer une solution numérique moderne capable de centraliser les données, automatiser les processus clés, et offrir une expérience utilisateur fluide tant pour l'administration que pour les résidents. L'objectif est d'améliorer l'efficacité des opérations tout en répondant aux besoins spécifiques des résidences étudiantes.

1.3 Objectifs du projet

Les objectifs du projet ne sont pas seulement techniques, mais aussi stratégiques et organisationnels. Chaque objectif est lié à un bénéfice clair pour les utilisateurs.

- **Centralization des opérations** :
  - L'application regroupe toutes les fonctionnalités nécessaires en un seul endroit : gestion des chambres, paiements, incidents, et statistiques.
  - Cela réduit la dépendance à plusieurs outils non connectés (par exemple, feuilles Excel et échanges par email).
- **Amélioration de l'efficacité administrative** :
  - Les administrateurs peuvent rapidement voir les informations critiques, comme le statut des chambres ou les paiements en retard.
  - Les tâches répétitives, comme l'envoi de rappels de paiement, sont automatisées.
- **Expérience utilisateur fluide** :
  - Les résidents ont accès à une interface simple pour consulter leurs informations, signaler des incidents et effectuer des paiements.
  - Les administrateurs disposent d'outils de recherche et de suivi pour gérer rapidement les requêtes ou incidents.
- **Automatisation et réduction des erreurs** :
  - Les processus comme l'attribution des chambres ou le suivi des paiements sont standardisés et automatisés.
  - Cela diminue les risques d'erreurs humaines tout en assurant une meilleure traçabilité.
- **Suivi analytique** :
  - L'application offre des tableaux de bord permettant de visualiser des statistiques importantes (taux d'occupation, paiements en retard, incidents).
  - Ces données permettent à l'administration de prendre des décisions éclairées et d'anticiper les besoins (exemple : planification de travaux de maintenance).

Conclusion

Ce premier chapitre montre pourquoi l'application est essentielle et ce qu'elle doit accomplir. Les points abordés (contexte et objectifs) posent une base solide pour les chapitres suivants, en particulier la description détaillée du projet et les spécifications techniques.

II. Analyse et conception

2.1 Introduction

L'analyse et la conception constituent les fondements essentiels de tout projet, qu'il s'agisse de développement logiciel, d'ingénierie de systèmes ou de conception de produits. L'analyse implique l'examen approfondi des besoins, des objectifs et des contraintes du projet, tandis que la conception consiste à élaborer des solutions efficaces et optimales pour répondre à ces exigences. Ensemble, ces deux processus permettent de définir clairement le problème à résoudre et d'identifier les meilleures approches pour le résoudre de manière efficace et efficiente. En combinant une analyse approfondie avec une conception réfléchie, on peut élaborer des solutions robustes et adaptées aux besoins spécifiques du projet, jetant ainsi les bases d'une mise en œuvre réussie.

2.2 Cahier de Charge

2.2.1 Spécifications Fonctionnelles :

Les spécifications fonctionnelles décrivent ce que l'application doit faire. Chaque fonctionnalité est décomposée en actions précises.

Gestion des Chambres :

- Création : Ajouter une nouvelle chambre avec des champs spécifiques (taille, équipements, disponibilité).

- Modification et suppression : Mettre à jour ou supprimer une chambre.

- Affichage des chambres disponibles : Liste des chambres avec filtres (taille, équipements).

Gestion des Résidents :

- Inscription : Formulaire avec validation des champs obligatoires.

- Attribution des chambres : Automatique ou manuelle.

- Profil résident : Affichage des informations personnelles, statut de la chambre et historique des paiements.

Gestion des Paiements :

- Suivi des loyers : Montants dus, payés, en retard.

- Notifications : Rappels par email pour les paiements en retard.

- Reçus : Génération et téléchargement de reçus.

Gestion des Requêtes de Maintenance :

- Signalement des incidents : Formulaire accessible aux résidents.

- Suivi des requêtes : Statut (en attente, en cours, résolu).

- Historique : Consultable par résident et par chambre.

Statistiques et Recherche :

- Tableaux de bord : Visualisation des statistiques globales.

- Recherche avancée : Filtres par chambre, résident ou requête.

2.2.2 Spécifications Techniques :

- Langages de programmation :

- Backend : Java (Spring Boot).

- Frontend : Angular ou React.js.

- Base de données : MySQL ou PostgreSQL.

- Serveur : Apache Tomcat.

- Authentification et sécurité : Spring Security avec chiffrement des données sensibles.

- Notifications : Intégration avec SendGrid ou JavaMail.

- Architecture : MVC avec API RESTful.

2.2.3 Spécifications Non Fonctionnelles

- Performance : Réponse en moins de 2 secondes pour les opérations courantes.

- Sécurité : Protection contre les attaques courantes (injections SQL, XSS).

- Disponibilité : 99,9 % de temps de disponibilité.

- Compatibilité : Accessible sur les principaux navigateurs et appareils mobiles.

- Extensibilité : Architecture modulaire pour l'ajout de nouvelles fonctionnalités

2.3 Conception UML

UML (Unified Modeling Language) est un langage formel et normalisé en termes de modélisation objet. Son indépendance par rapport aux langages de programmation, aux domaines de l'application et aux processus, son caractère polyvalent et sa souplesse ont fait de lui un langage universel. En plus UML est essentiellement un support de communication, qui facilite la représentation et la compréhension de solution objet. Sa notation graphique permet d'exprimer visuellement une solution objet, ce qui facilite la comparaison l'évaluation des solutions. L'aspect de sa notation, limite l'ambigüité et les incompréhensions.

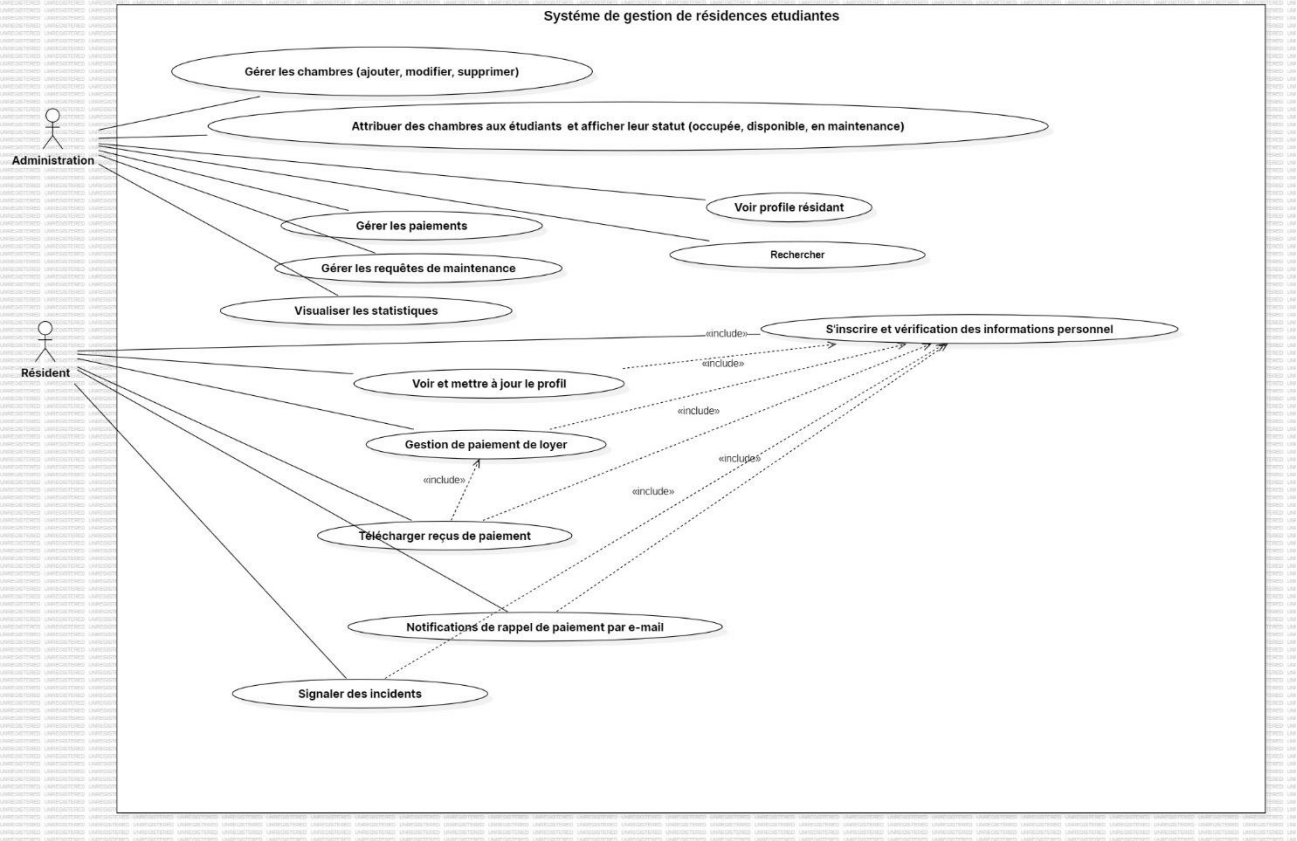
Voici les schémas que nous avons utilisés pour expliquer et visualiser notre application web :

- **Diagramme de cas d'utilisation**
- **Diagramme de classe**
- **Diagramme de scénario**

2.3.1 Diagramme de cas d'utilisation

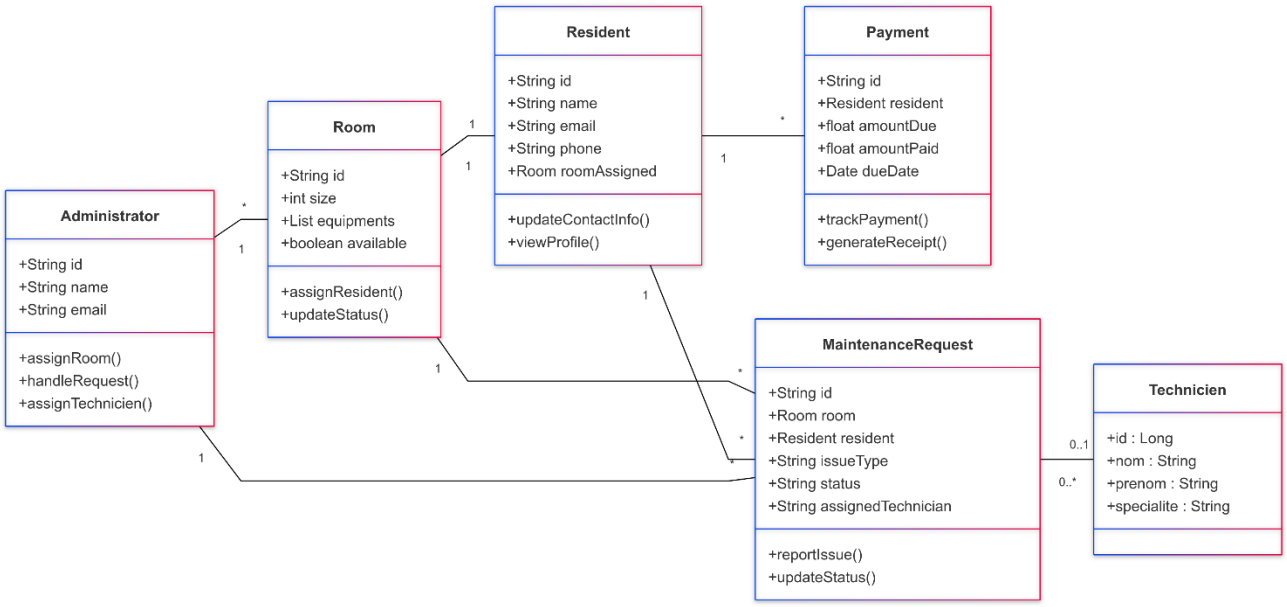
Un **diagramme de cas d'utilisation** (ou **use case diagram** en anglais) est un type de diagramme comportemental de la notation UML (Unified Modeling Language) qui décrit les interactions entre les utilisateurs (ou acteurs) et un système pour atteindre un objectif spécifique. Ce diagramme est utilisé pour capturer les exigences fonctionnelles d'un système et montrer les différentes manières dont les utilisateurs peuvent interagir avec le système.

Ce diagramme permet d'analyser d'organiser les besoins, et d'identifier les possibilités d'interaction entre le système et les acteurs.



2.3.2 Diagramme de classe

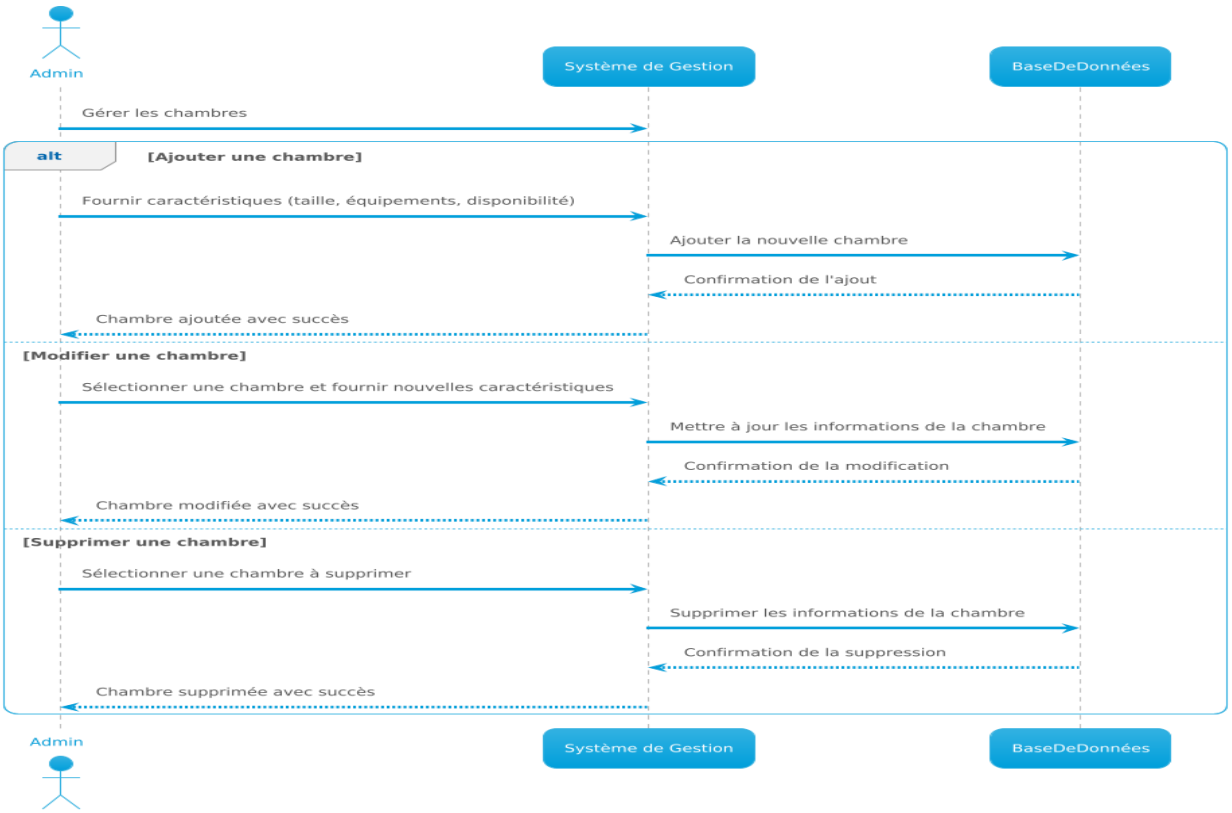
Un diagramme de classe est une représentation visuelle des classes et de leurs relations dans un système logiciel. Il présente les différentes entités (ou classes) dans un système, ainsi que les attributs, les méthodes et les associations entre ces classes. C'est un outil de modélisation utilisé pour planifier, concevoir et comprendre la structure statique d'une application.



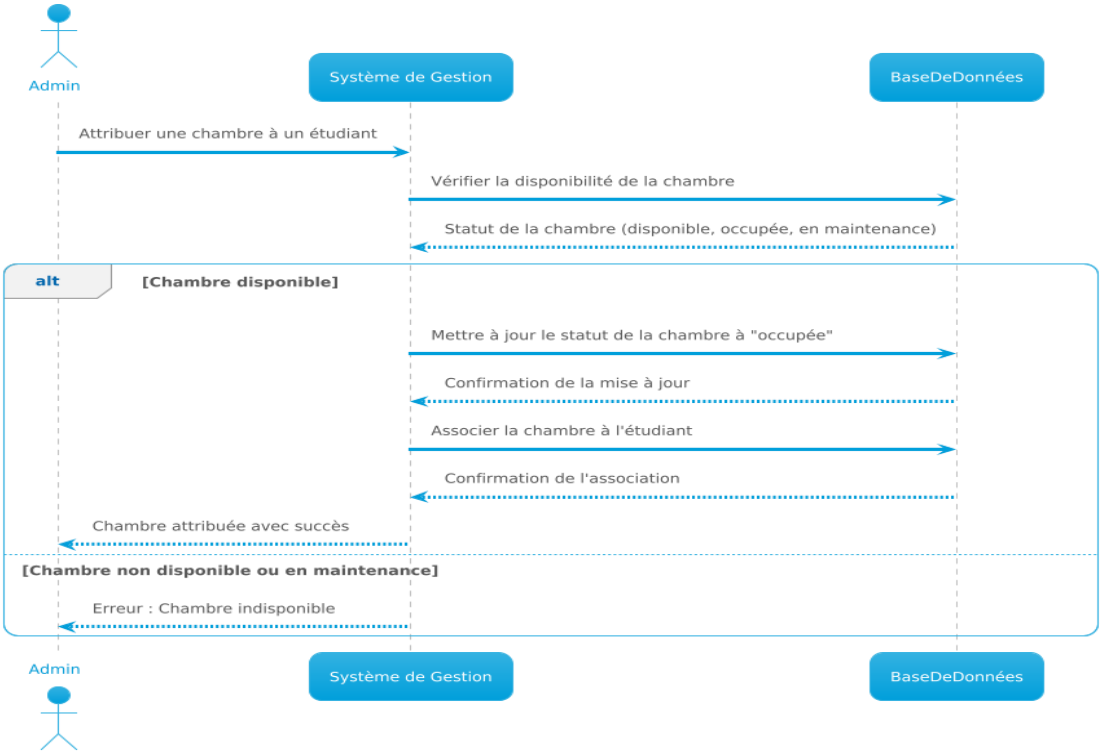
2.3.3 Diagramme de scénario

Un diagramme de scénario en UML est une représentation visuelle qui montre la manière dont les différents éléments d'un système interagissent dans une séquence chronologique pour réaliser une action spécifique. Il met en évidence les messages échangés entre les objets ou acteurs, permettant ainsi de comprendre le flux d'exécution d'un processus ou d'une fonctionnalité logicielle.

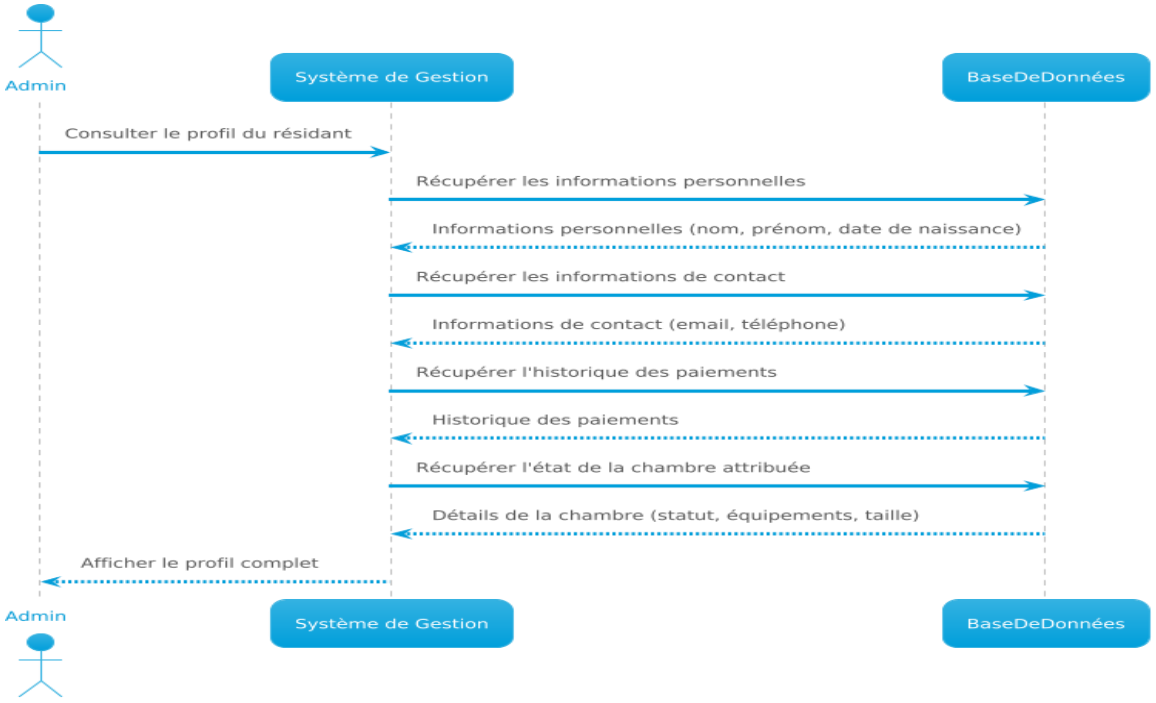
Gérer les chambres (ajouter, modifier, supprimer) :



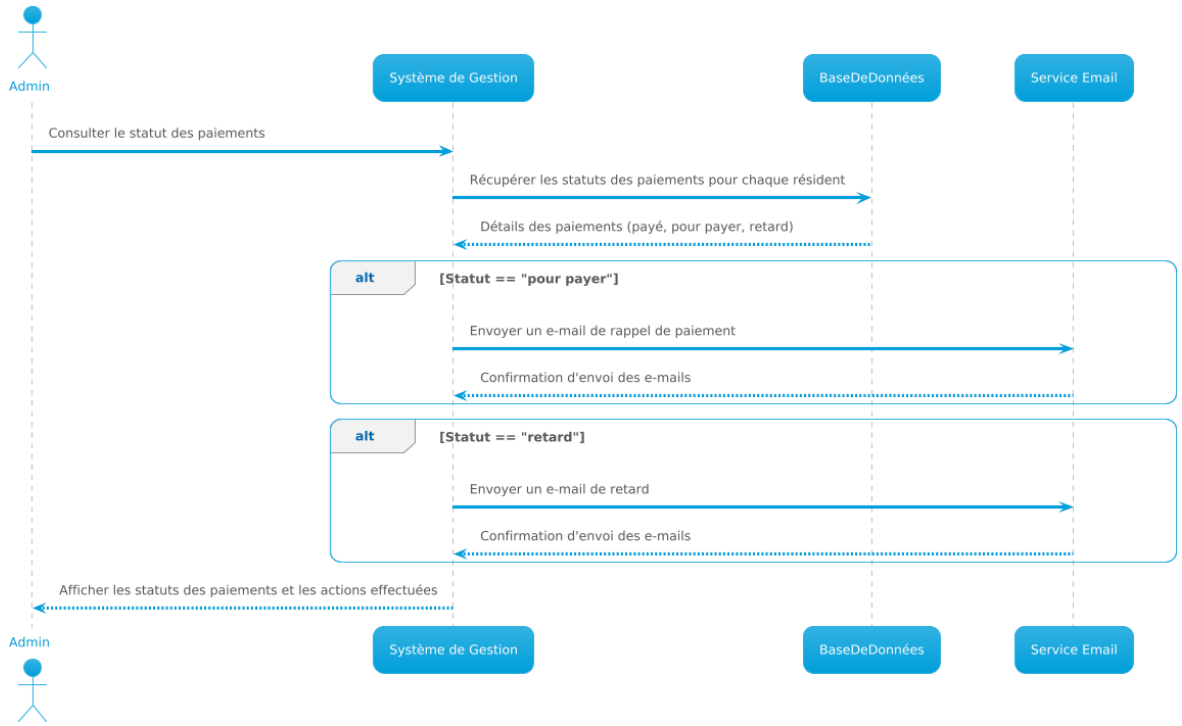
Attribuer des chambres aux étudiants et afficher leur statut (occupée, disponible, en maintenance) :



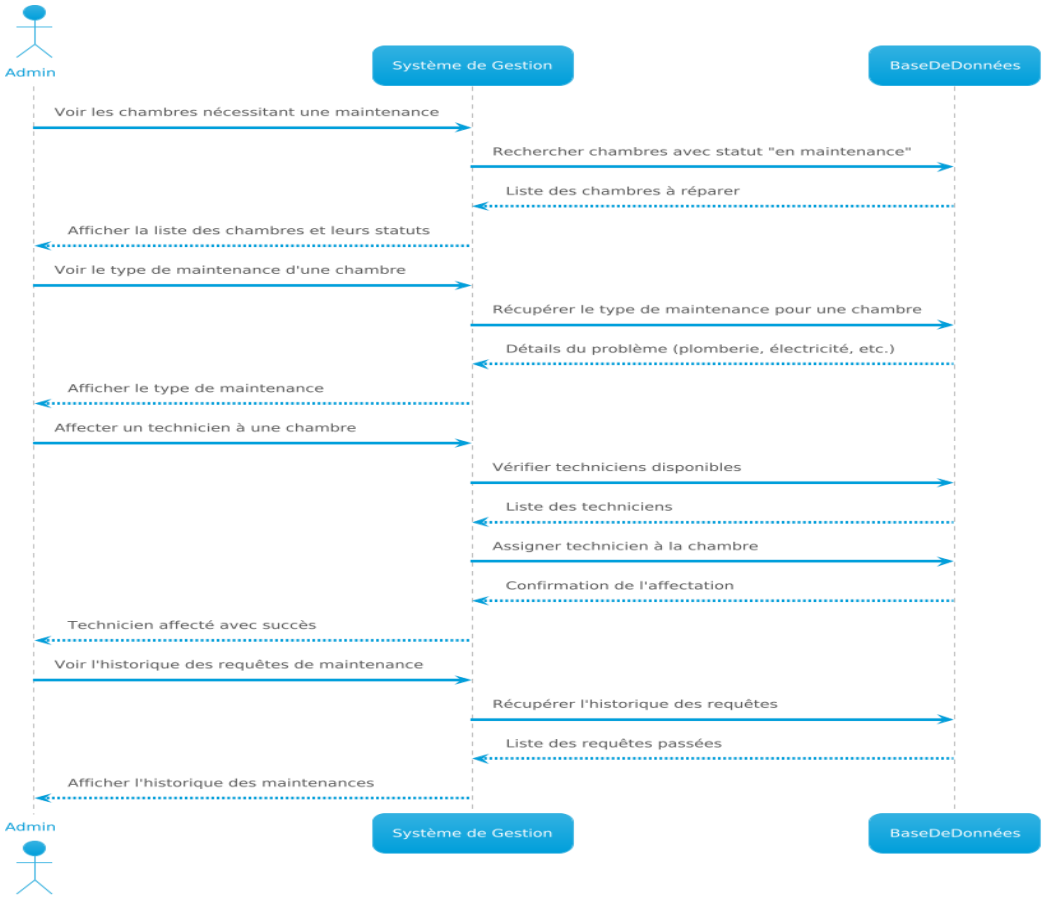
Voir profile résidant :



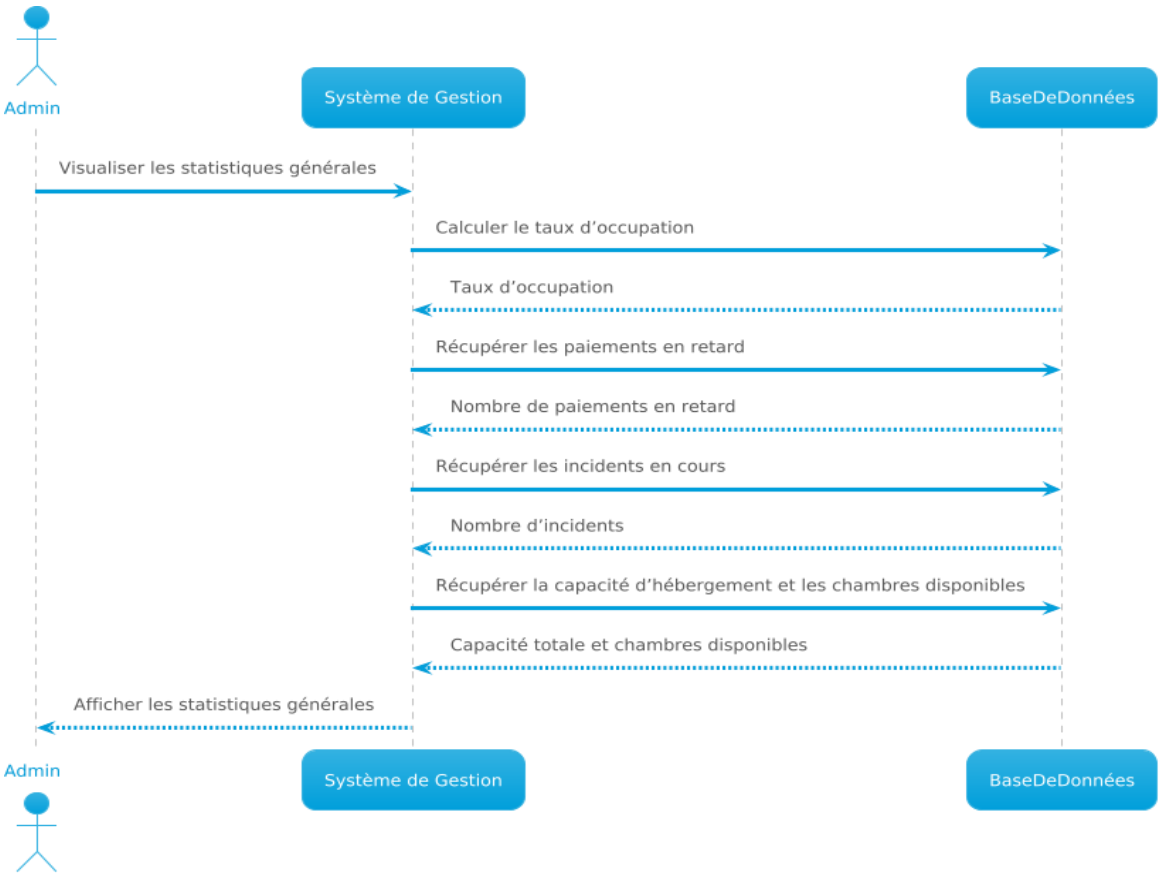
Gérer les paiements :



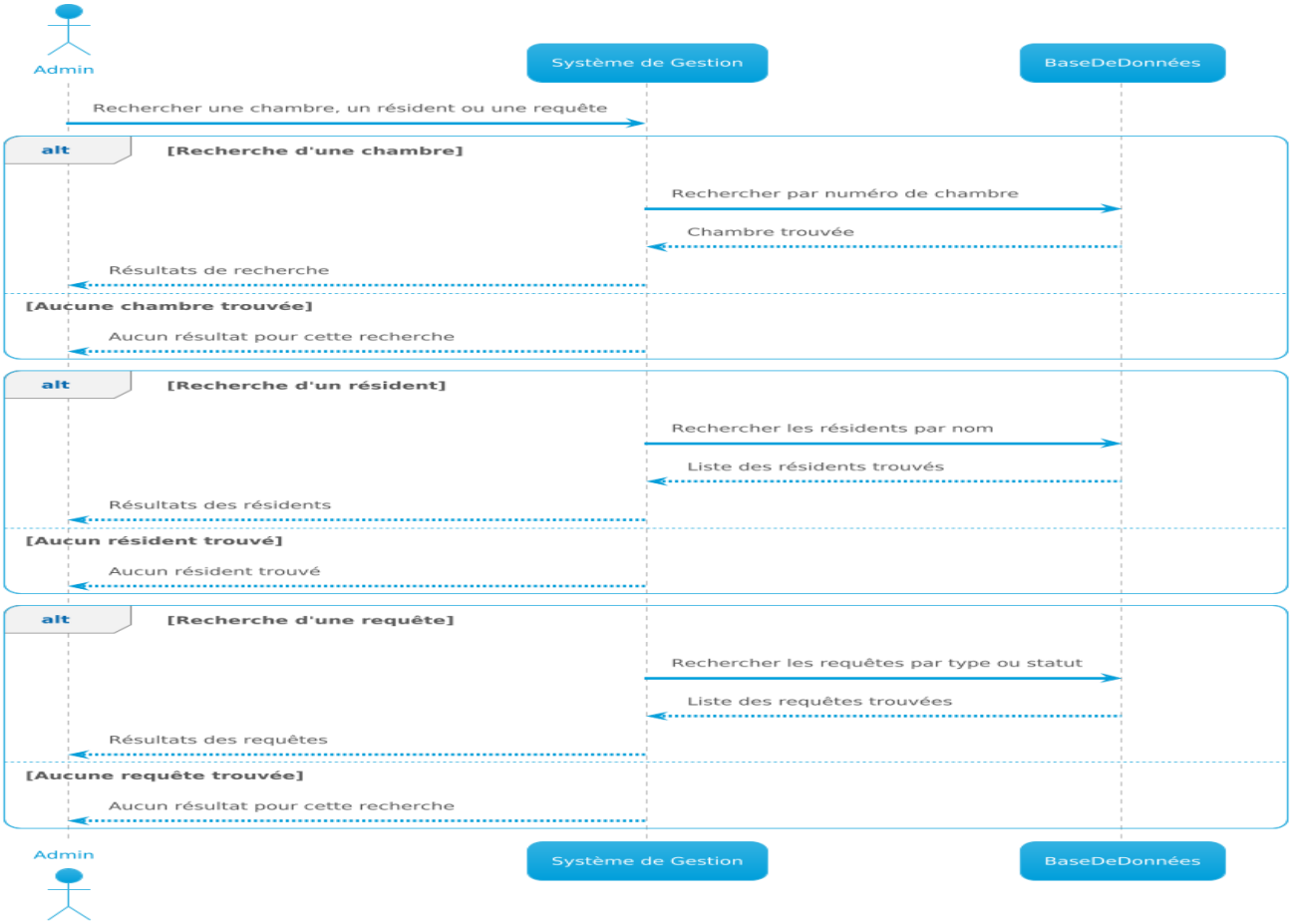
Gérer les requêtes de maintenance :



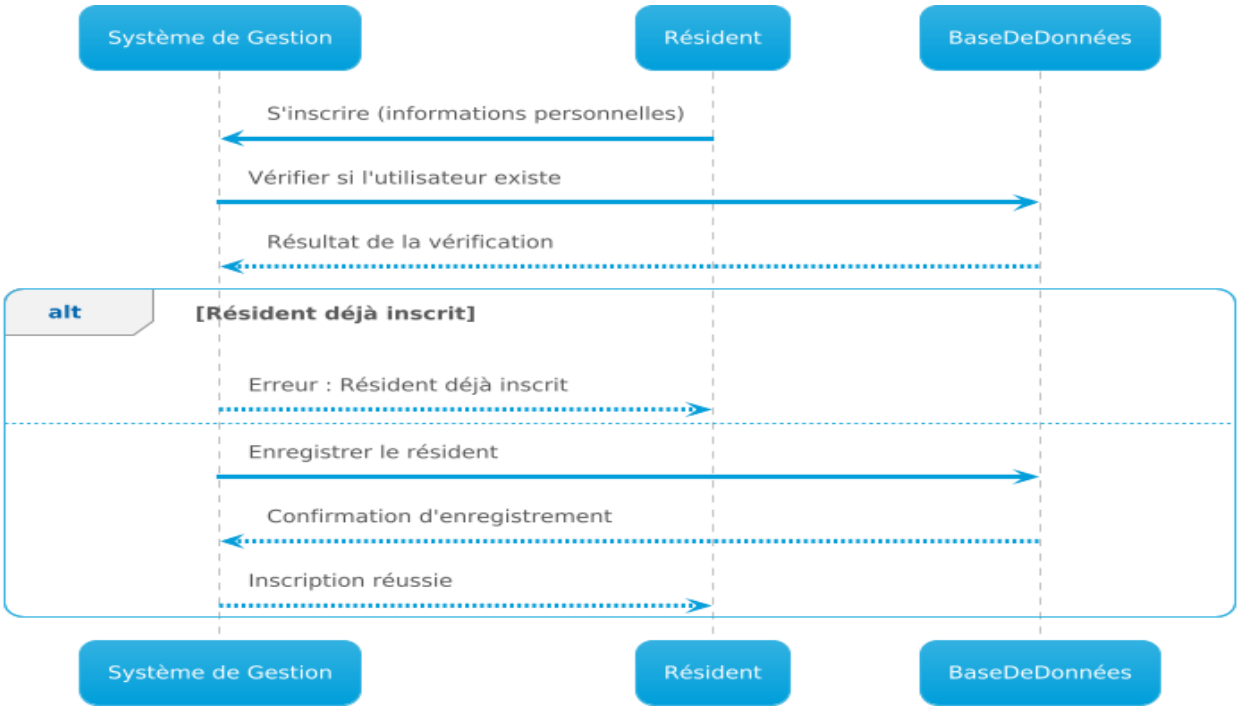
Visualiser les statistiques :


Admin
Système de Gestion
BaseDeDonnées
Visualiser les statistiques générales
Calculer le taux d'occupation
Taux d'occupation
Récupérer les paiements en retard
Nombre de paiements en retard
Récupérer les incidents en cours
Nombre d'incidents
Récupérer la capacité d'hébergement et les chambres disponibles
Capacité totale et chambres disponibles
Afficher les statistiques générales
Admin
Système de Gestion
BaseDeDonnées

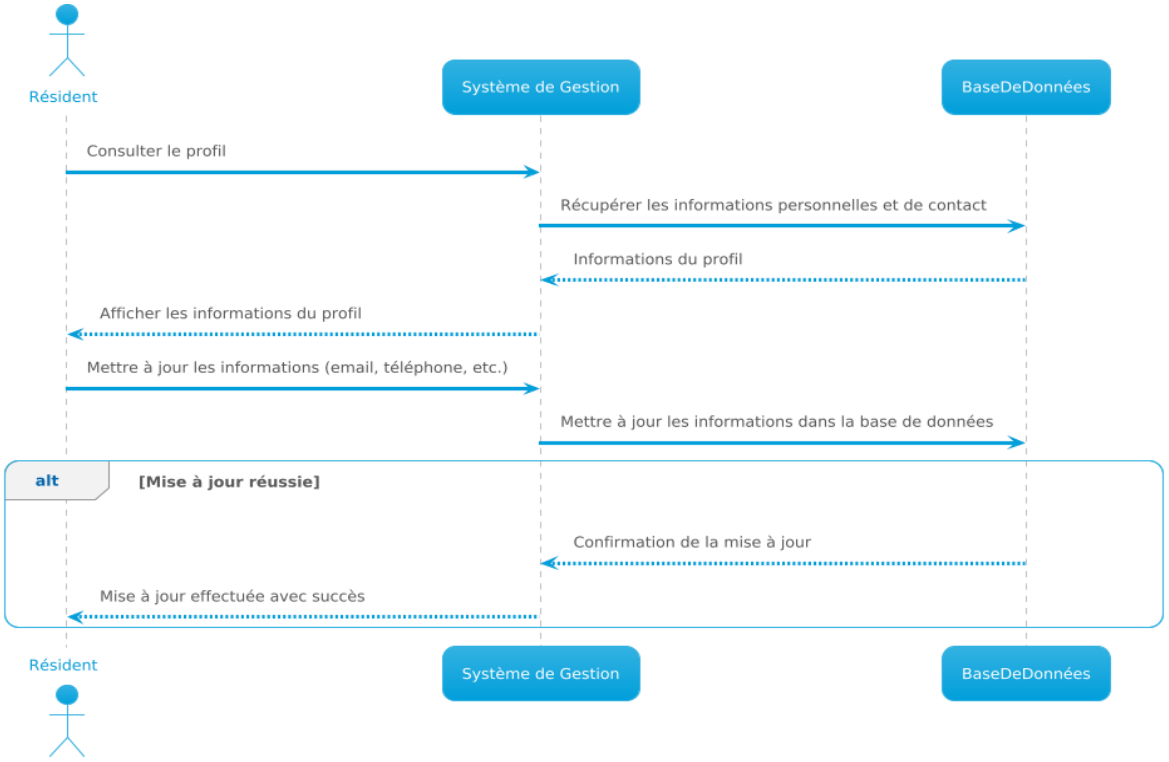
Rechercher :


Admin
Système de Gestion
BaseDeDonnées
Rechercher une chambre, un résident ou une requête
alt
[Recherche d'une chambre]
Rechercher par numéro de chambre
Chambre trouvée
Résultats de recherche
[Aucune chambre trouvée]
Aucun résultat pour cette recherche
alt
[Recherche d'un résident]
Rechercher les résidents par nom
Liste des résidents trouvés
Résultats des résidents
[Aucun résident trouvé]
Aucun résident trouvé
alt
[Recherche d'une requête]
Rechercher les requêtes par type ou statut
Liste des requêtes trouvées
Résultats des requêtes
[Aucune requête trouvée]
Aucun résultat pour cette recherche
Admin
Système de Gestion
BaseDeDonnées

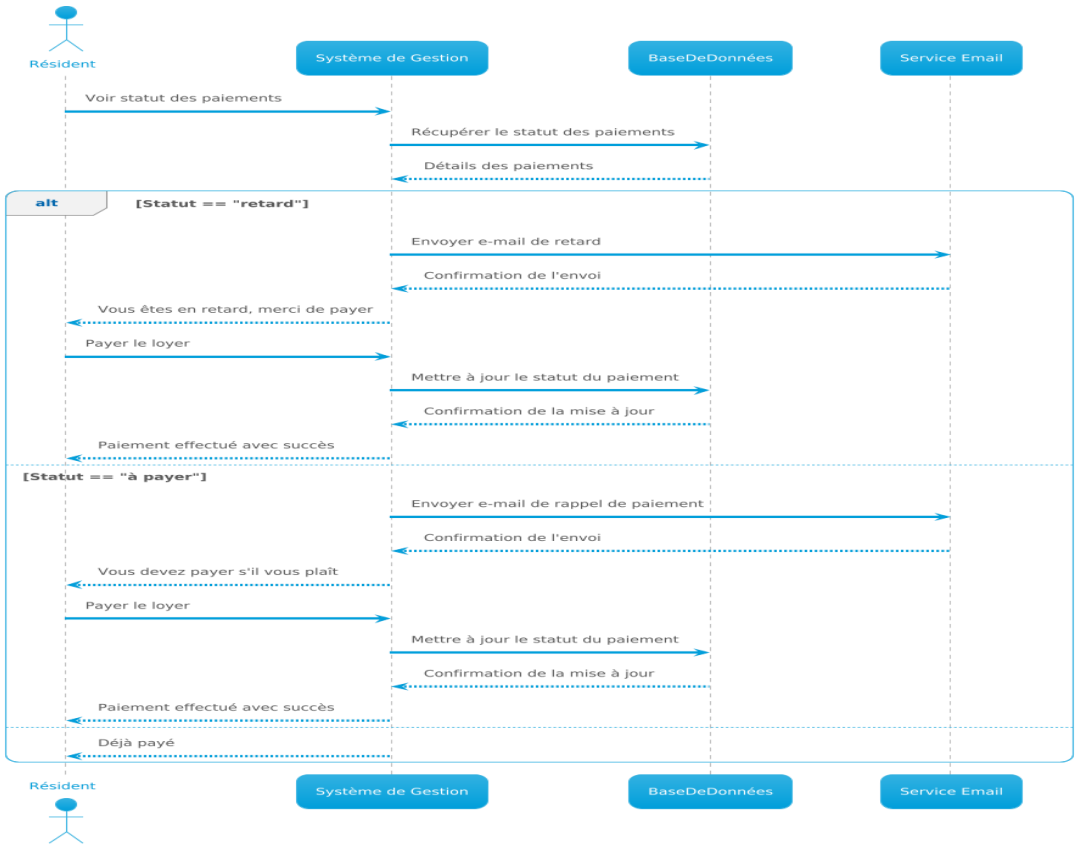
S'inscrire et vérification des informations personnel :



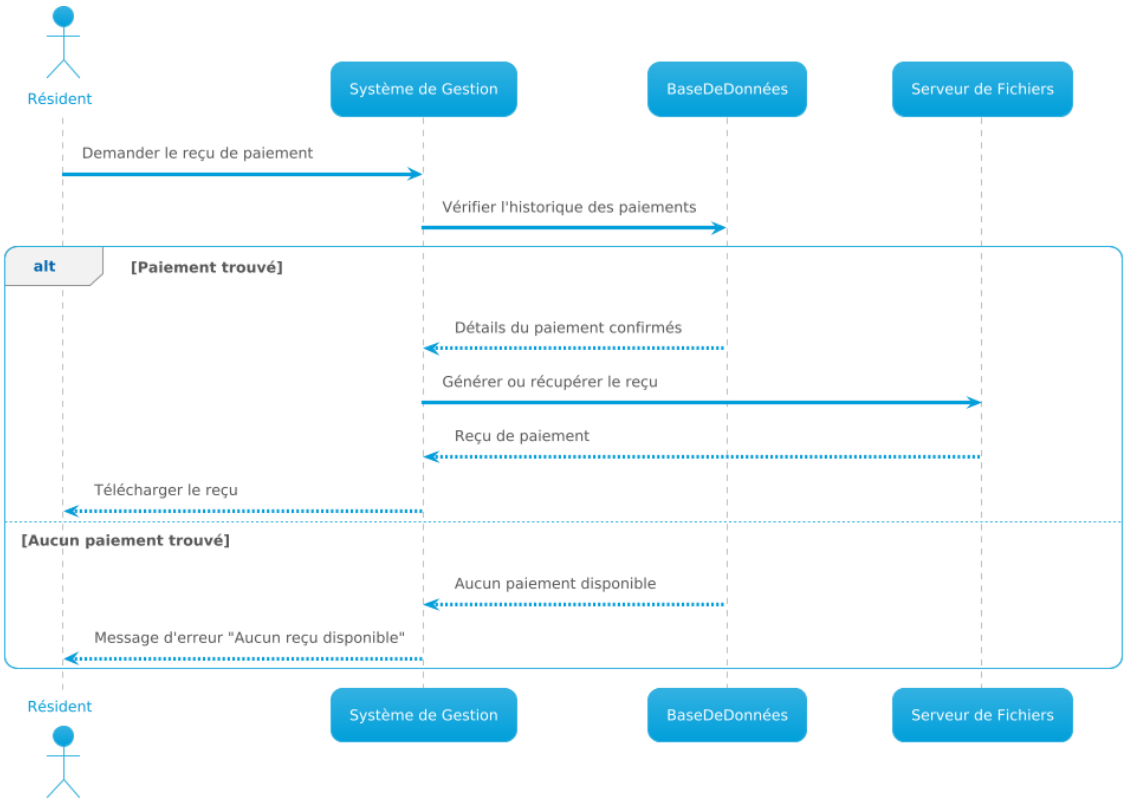
Voir et mettre à jour le profil :


Résident
Système de Gestion
BaseDeDonnées
Consulter le profil
Récupérer les informations personnelles et de contact
Informations du profil
Afficher les informations du profil
Mettre à jour les informations (email, téléphone, etc.)
Mettre à jour les informations dans la base de données
alt
[Mise à jour réussie]
Confirmation de la mise à jour
Mise à jour effectuée avec succès
Résident
Système de Gestion
BaseDeDonnées

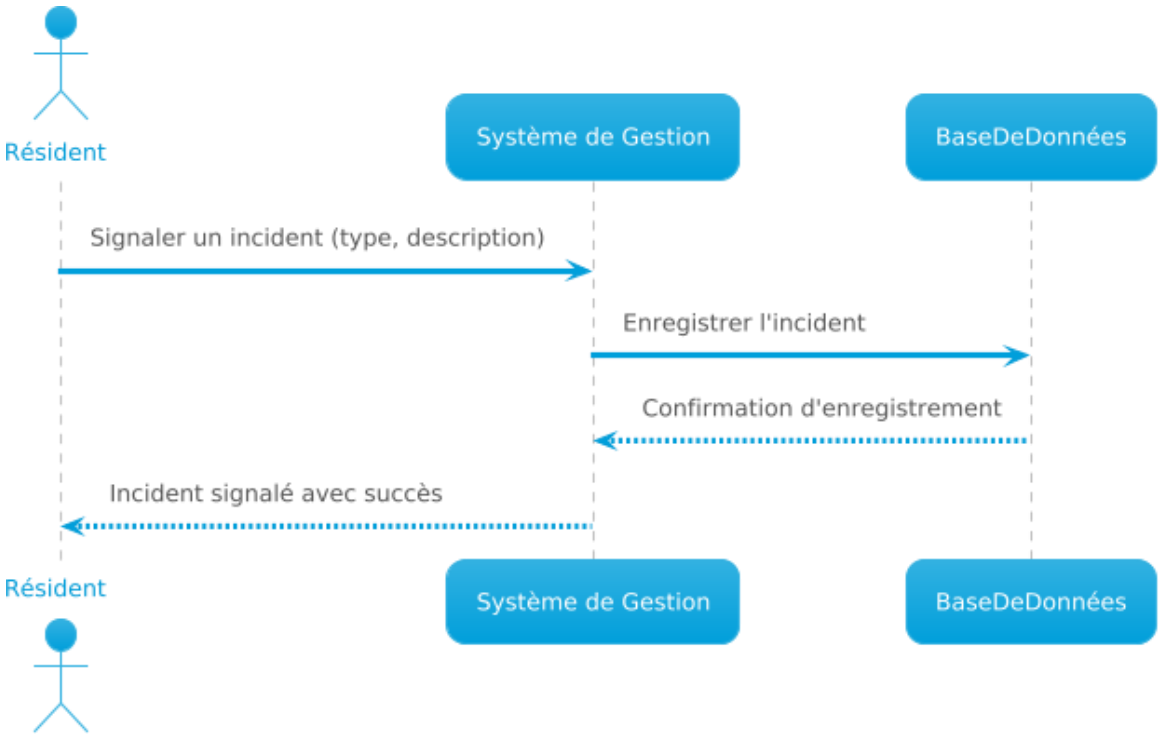
Gestion de paiement de loyer :



Télécharger reçus de paiement :



Signaler des incidents :



3 Choix des outils

Pour développer l'application, une sélection d'outils et de technologies robustes et éprouvées a été effectuée afin de garantir une performance optimale et une maintenance simplifiée :

1. **Backend** :
    - **Spring Boot** : Framework Java robuste permettant une création rapide d'applications web sécurisées et modulaires. Il simplifie l'intégration avec des bases de données et offre une large gamme de fonctionnalités prêtes à l'emploi.
2. **Frontend** :
    - **Thymeleaf** : Moteur de templates intégré à Spring Boot, permettant de générer des pages HTML dynamiques tout en simplifiant la gestion des données provenant du backend.
    - **Bootstrap** : Framework CSS populaire, utilisé pour concevoir une interface utilisateur moderne, responsive et intuitive, compatible avec tous les appareils.
3. **Base de données** :
    - **MySQL** : Système de gestion de bases de données relationnelles fiable et performant, utilisé pour stocker les informations essentielles comme les chambres, les résidents, les paiements et les requêtes de maintenance.
4. **Environnement de développement** :
    - **IntelliJ IDEA** : IDE utilisé pour coder et tester le projet en Java.
    - **Git/GitHub** : Pour le contrôle de version et la collaboration entre les développeurs.

Conclusion

En résumé, ce chapitre a conclu la phase d'analyse et de conception qui constitue une étape cruciale du projet, visant à définir précisément les besoins fonctionnels et non fonctionnels de l'application et à concevoir une architecture adaptée pour y répondre efficacement.

Dans un premier temps, une **analyse approfondie des besoins** a été réalisée pour identifier les fonctionnalités essentielles, comme la gestion des chambres, le suivi des paiements, la gestion des résidents et le traitement des requêtes de maintenance. Cette analyse s'est appuyée sur des outils comme les **diagrammes de cas d'utilisation UML** pour illustrer les interactions entre les utilisateurs (administrateurs et résidents) et le système.

Ensuite, la phase de **conception technique** a permis de définir une architecture basée sur une approche **3 tiers** (Frontend, Backend, Base de données). Le **diagramme de classes UML** a été utilisé pour modéliser les entités principales, telles que les chambres, les résidents et les paiements, tandis que les **diagrammes de séquence** ont illustré les scénarios d'interaction entre les différentes couches du système.

Enfin, des choix technologiques ont été faits pour garantir une implémentation robuste et moderne. L'application repose sur **Spring Boot** pour le backend, **Thymeleaf** et **Bootstrap** pour le frontend, et **MySQL** pour la gestion des données. Ces outils ont été choisis pour leur fiabilité, leur simplicité d'intégration et leur capacité à répondre aux besoins du projet.

Cette phase a posé les bases solides pour le développement, tout en garantissant que l'application respecte les exigences et offre une expérience utilisateur optimale.

Conclusion et perspective

Le développement de l'application de gestion des résidences étudiantes représente une avancée significative dans la centralisation et l'optimisation des tâches administratives. Avec des fonctionnalités robustes telles que la gestion des chambres, le suivi des paiements, et la prise en charge des requêtes de maintenance, cette solution permettra à l'administration de gagner en efficacité tout en offrant une meilleure expérience utilisateur aux résidents.

Cependant, l'évolution des besoins et les avancées technologiques ouvrent des perspectives intéressantes pour améliorer davantage l'application. Parmi ces axes d'amélioration, l'intégration de **Stripe** comme solution de paiement en ligne représente une opportunité majeure. Cette fonctionnalité apporterait des avantages tels que la sécurité accrue des transactions, la gestion automatisée des paiements récurrents, et une interface utilisateur intuitive pour régler les loyers en quelques clics. Bien qu'elle ne fasse pas partie des fonctionnalités immédiates, cette option pourrait être envisagée dans une version future de l'application, répondant ainsi à des besoins encore plus modernes et pratiques.

En résumé, ce projet marque une étape importante dans la transformation numérique de la gestion des résidences étudiantes. En restant à l'écoute des utilisateurs et en explorant des améliorations comme Stripe, l'application pourra évoluer pour devenir un outil encore plus performant et incontournable.

Bibliographie

Spirng : https://docs.spring.io/spring-boot/index.html

YouTube StatQuest: https://www.youtube.com/channel/UCtYLUTtgS3k1Fg4y5tAhLbw