Pour réussir notre projet, plusieurs étapes doivent être respectées. Tout d'abord, il est essentiel d'identifier les classes et leurs relations entre elles à l'aide d'un diagramme de classes UML. Cette première étape nécessite de définir les classes nécessaires au projet, ainsi que leurs attributs et méthodes respectifs. Cela implique une analyse minutieuse des exigences du projet pour comprendre les entités à modéliser et les fonctionnalités à inclure. Le diagramme de classes UML, un outil largement utilisé, permet de visualiser la structure des classes et leurs relations, telles que l'association, la composition ou l'héritage.

Ensuite, vient la phase de programmation et de préparation des tests. Une fois le diagramme de classes UML établi, la seconde étape consiste à concrétiser la conception en programmant les classes et en implémentant les fonctionnalités du logiciel. Cette étape implique la rédaction du code source dans un langage de programmation orienté objet, tel que JAVA dans notre cas. L'objectif est de mettre en œuvre le logiciel conformément à la conception établie lors de la première étape, tout en garantissant l'exactitude et la fonctionnalité de toutes les caractéristiques à travers des tests rigoureux.

Voici notre modèle :

Une image contenant diagramme, Plan, Dessin technique, schématique

Description générée automatiquement

Nous retrouvons les classes importantes comme les classes Hotel, Client, Reservation, Chambre, Sejour, Employé, Consommation, Produit ainsi que leur attributs.

**Génération automatique des méthodes de base**

Nous avons utilisé l'outil de génération de code d'Eclipse pour générer automatiquement les getters, setters, attributs et constructeurs pour toutes les classes du projet. Cela nous a permis de gagner du temps et de nous assurer que le code est cohérent et exempt d'erreurs.

**Gestion des chambres**

Le logiciel permet d'ajouter, modifier et supprimer des chambres, de suivre leur état et de les affecter aux réservations. Il est également possible de vérifier la disponibilité d'une chambre pour une période donnée et de marquer son statut comme nettoyée après le départ d'un client. La classe chambre permet de rechercher une chambre par 3 moyens : le numéro, le type et le nom

**Gestion des clients**

La création de profils clients permet d'enregistrer leurs informations et de suivre leurs réservations. L'utilisateur peut facilement créer des profils clients en saisissant les informations basiques telles que le nom, le prénom, le numéro de téléphone et l'adresse email.

**Gestion des consommations**

La classe Consommation permet à l’utilisateur de renter les produits que le clienta pu consommer en supplément.

**Gestion des employés**

Nous avons créé une classe mère Employe qui fournira les informations universels des employés : telles que le nom, le prénom, et les numéros. Ensuite, nous avons créé deux sous-classes : Réceptionniste et AgentEntretien. La classe AgentEntretien permet de suivre les chambres dont l'agent s'est occupé, tandis que la classe Réceptionniste permet d'associer les réceptionnistes aux réservations.

**Gestion des réservations**

Le logiciel offre un système de recherche performant pour trouver des chambres disponibles. Il permet également de créer, modifier et annuler des réservations, et de les affecter aux chambres correspondantes.

**Gestion des séjours**

L'enregistrement des arrivées et des départs des clients est simplifié grâce au logiciel. Il permet également de suivre les consommations durant le séjour. La classe Sejour permet de calculer le total du prix de sejour.

**Classes du contrôleur:**

Ces classes gèrent les interactions avec l'utilisateur et manipulent les données des classes du modèle. Elles implémentent souvent l'interface ActionListener pour répondre aux clics des utilisateurs sur les boutons ou les éléments de menu.

* BouttonEditeur (Bouton d'édition) : Cette classe crée un bouton personnalisé pour une cellule JTable qui permet de modifier une chambre de l'hôtel. Il supprime la chambre après confirmation.
* BouttonEditeurRes (Bouton d'édition de réservation) : Similaire à BouttonEditeur, cette classe modifie une réservation, la supprime ainsi que la chambre et le séjour associés si elle est confirmée.
* ButtonConsulSejour (Bouton Consulter Séjour) : Cette classe permet de consulter les détails d'un séjour en cliquant sur un bouton dans le JTable.
* ButtonEditeurSuppClient (Bouton Modifier/Supprimer Client) : Cette classe modifie ou supprime un client du système, en mettant également à jour les réservations associées au client.
* ControlAjout (Contrôle d'ajout) : Cette classe gère l'ajout d'une nouvelle chambre à l'hôtel.
* ControlAjoutReservation (Contrôle d'ajout de réservation) : Cette classe gère l'ajout d'une nouvelle réservation, y compris la recherche des chambres disponibles et la création d'un objet réservation.
* ControlRecherche (Contrôle de recherche) : Cette classe recherche les chambres disponibles en fonction des dates et du type de chambre fournis par l'utilisateur.
* ControlRechercheClient (Contrôle de recherche de client) : Cette classe recherche des clients en fonction d'une chaîne de requête saisie par l'utilisateur.
* ControleAjoutClient (Contrôle d'ajout de client) : Cette classe ajoute un nouveau client au système hôtelier.
* ControlAffecMenage (Affecter le ménage) : Cette classe semble être un espace réservé pour une fonctionnalité potentielle de gestion du ménage (non implémentée dans le code fourni).

**Classes de vue:**

Ces classes représentent les composants visuels de l'application.

* Fenetre (Fenêtre) : Cette classe représente la fenêtre principale de l'application, contenant des menus et un panneau de contenu qui affiche d'autres vues.
* BottonRendu (Rendu de bouton) : Cette classe crée un bouton personnalisé pour une cellule JTable.