Document de synthèse

Gestion de biens immobiliers

projet réalisé par
Alfred MENGIN, Arthur GENET et Lilian CALAS

1 : Rendu de l'application

Le but principal de notre application est de gérer les transactions entre individus souhaitant vendre, louer ou acheter un bien. Les clients correspondent aux personnes à l'origine de la transaction (vendeurs, bailleurs) tandis que les particuliers correspondent aux individus qui acquièrent le bien (acheteurs, locataires). L'application comprend une page principale avec un menu. Le menu contient 4 boutons, un bouton pour les agents, un bouton pour les clients, un bouton pour le responsable et un dernier pour les particuliers. Les clients, les agents et le responsable peuvent accéder à leur espace personnel en s'authentifiant. Les clients peuvent également se créer un compte s'ils n'en possèdent pas encore un. Une candidature sera alors envoyée au responsable d'agence qui pourra accepter ou non cette candidature sur son espace personnel.

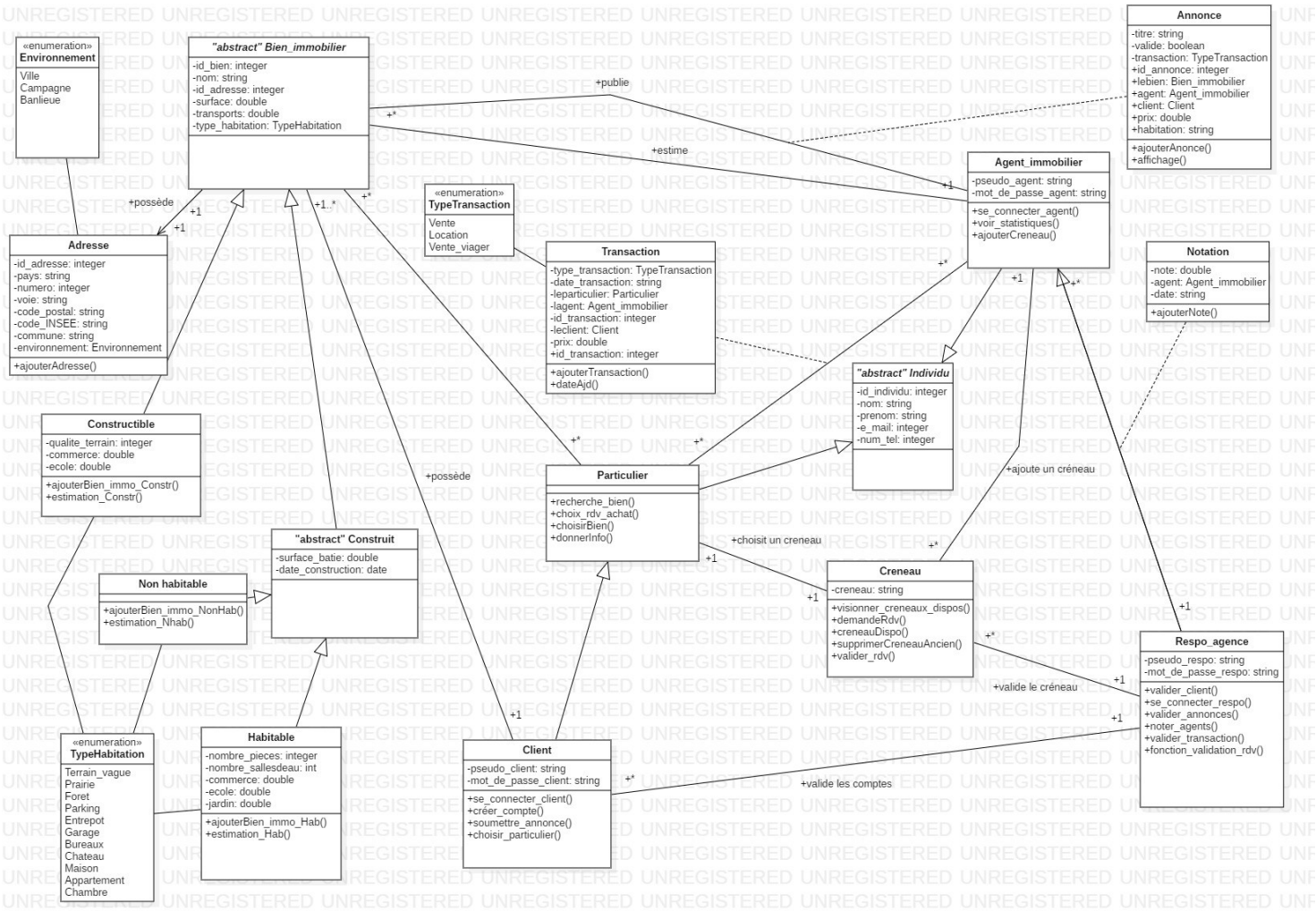
Une fois ce compte validé par le responsable d'agence, le **client** peut se connecter au gré de ses envies. Sur son espace personnel, le client peut soumettre un bien au responsable d'annonce, en rentrant diverses valeurs utiles. Lorsque des particuliers proposent une transaction pour un de ses biens, le client peut voir la liste des propositions et décider avec quel particulier faire la transaction.

Le **particulier** peut quant à lui prendre un rendez-vous ou acquérir un bien. Dans un premier temps, il doit faire une recherche selon les critères qui l'intéressent (type de bien, type d'habitation et type d'environnement). Une liste de biens s'affiche alors avec les différentes caractéristiques (surface, prix, distance aux commerces...). Le particulier regarde cette liste puis choisit celui qui lui plaît dans un menu déroulant. Il a ensuite le choix entre deux actions : prendre rendez-vous pour le bien ou proposer une transaction pour le bien. Des informations personnelles lui sont alors demandées (nom, prénom, numéro de téléphone, e mail). Si le nom et le prénom rentrés sont déjà présents dans la base de données, on demande au particulier si ses informations correspondent bien. Si il veut faire une transaction, le particulier a alors terminé. Si il souhaite prendre rendez-vous, il doit choisir un créneau parmi une liste. Les créneaux sont ajoutés par les agents mais si ceux-ci sont antérieurs à la date d'aujourd'hui, ils sont supprimés automatiquement. La demande de rendez-vous est alors envoyée au responsable d'agence.

Le **responsable d'agence** accède a plusieurs fonctionnalités une fois connecté. Tout d'abord, il peut valider ou refuser les demandes de rendez-vous des particuliers. C'est aussi lui qui valide les demandes de création de compte des clients. Enfin, il valide les demandes de création d'annonces et de transaction des clients. Quand il valide une annonce, le responsable doit aussi attribué un agent qui sera chargé du bien. Il peut voir de combien de biens chaque agent s'occupe déjà avant de prendre une décision. Sa dernière mission est de donner une note aux agents. Il choisit dans un premier temps l'agent qu'il veut noter parmi un choix dans un menu déroulant. Une boite de dialogue s'affiche avec le nombre de transaction qu'a effectué l'agent. Le responsable tape alors sa note sur vingt.

Une fois connecté, l'**agent immobilier** a alors le choix entre voir ses statistiques ou ajouter un créneau à la base de données. Si il décide de regarder ses statistiques, il peut voir la moyenne des notes données par le responsable d'agence ainsi que son nombre de transactions réalisées et sa prime. S'il décide d'ajouter un créneau, une boite de dialogue s'affiche alors lui demandant de rentrer l'horaire voulu. Celui-ci sera automatiquement ajouté à la base de données.

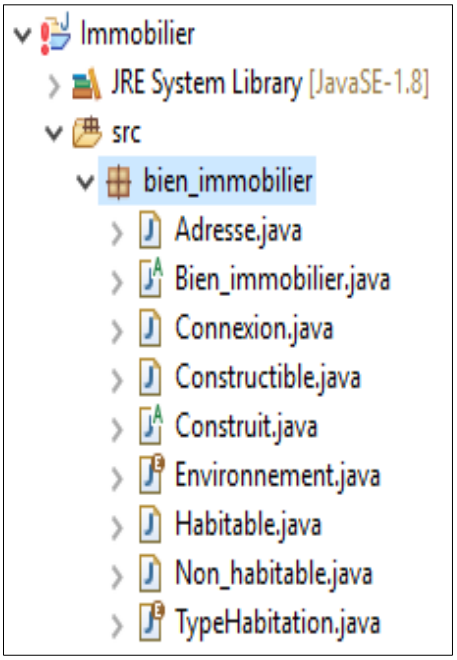
En progressant dans notre travail, nous avons modifié certaines choses dans notre façon de voir le sujet. Ces modifications ont pu être induites par l'apparition d'une meilleure façon de traiter le problème ou alors par des impasses au niveau du développement.
C'est pour cela que nous avons réalisé un nouveau diagramme de classe afin de montrer les quelques changements que nous avons dû opérer en cours de route.



Sur cette dernière version du diagramme de classe, de nombreuses méthodes ont été modifiées, supprimées ou ajoutées. Par exemple, dans la partie bien_immobilier, nous avons choisi d'enlever la méthode abstraite estimation et d'en créer trois propres aux classes de type de bien à cause de problèmes de développement.
Nous avons également enlevé la classe Rendez_vous car elle s'apparentait trop à la classe Creneau. La classe créneau a donc pris les deux rôles (de créneau horaire et de rendez-vous).

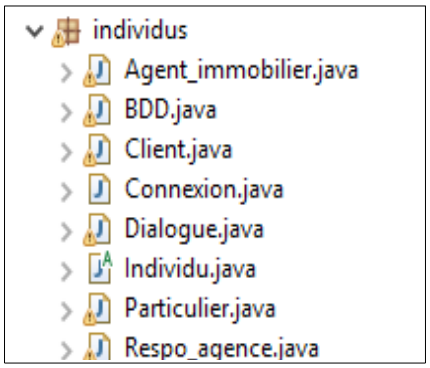
2 : Architecture du code et de la base de données

Nous avons décidé d'organiser le code en 3 grandes parties en créant les packages bien_immobilier, individus et interactions.



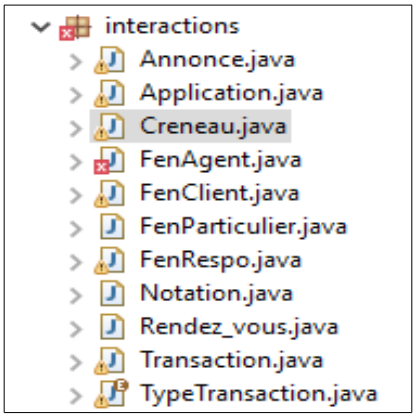
Le premier comprend les différents biens immobiliers ainsi que toutes les méthodes qui s'y rapportent.
On y retrouve la classe Bien_immobilier qui recense les attributs que l'on trouve dans tous les types de biens et qui établit la connection à la base de données. Puis, il y a les classes Constructible et Construit qui héritent de Bien_immobilier. Constructible traite de tous les biens où rien n'a encore été construit (terrain_vague, forêt, ..) et dispose d'une qualité du terrain qui indique si il est plus ou moins propice à la construction. Construit, quant à elle, fait référence aux biens qui ont déjà des infrastructures présentes qu'elles soient habitables ou non habitables. Habitable et Non_habitable sont alors les deux classes restantes. Pour Habitable, on trouve tous les biens où l'on peut vivre (maison, appartement, ..) avec ses caractéristiques (pieces d'eau, jardin, ..). Pour Non_habitable, ce sont des biens d'ordre pratique que l'on trouve (parking, bureaux, ..).

Chacune des classes Constructible, Habitable et Non_habitable possède une méthode qui permet l'ajout d'un bien dans la base de données avec toutes ses caractéristiques dans la table qui lui correspond. Elles possèdent également chacune une méthode d'estimation qui renvoie une valeur calculée à partir de la surface, de l'environnement et des autres attributs. Cette valeur diffère selon le type de transaction en renvoyant soit un prix de vente, soit un loyer, soit une rente.
A ces classes, se rajoutent les classes « énumération » TypeHabitation et Environnement qui stockent les différents types de bien et leur localisation possible avec des méthodes permettant d'accéder aux noms des objets et inversement. Enfin, on trouve la classe connexion qui rend possible la connection à la base de données.



Le second comprend les différents types d'individus et les méthodes qui permettent l'action de ces individus dans l'application. La classe individu est la classe mère la plus générale, elle comprend les attributs communs à tous les individus. La classe Particulier contient les personnes souhaitant prendre rendez-vous, candidater pour un bien ou encore seulement rechercher un bien. Les particuliers ne disposent pas d'un compte propre. Les clients, les agents immobiliers et le responsable disposent chacun d'un espace qui leur est propre , ils y accèdent en s'authentifiant. Le client peut soumettre un bien.
Il peut également choisir le particulier qui pourra jouir du bien. Le responsable a un rôle central dans l'application. Il valide les candidatures des clients, les annonces (et y associent un agent), les rendez-vous et les transactions. Il note également les agents. Les agents peuvent consulter leurs statistiques personnelles et créer des créneaux en fonction d'un modèle. Les classes Particulier, Agent_immobilier et Client héritent de la classe Individu, tandis que la classe Respo_agence hérite de Agent_immobilier. Le package contient également la classe Connexion permettant de créer le lien vers la base de données. La classe Dialogue est une classe contenant les fonctions qui créent les fenêtres de saisie (menus déroulants, cadre de saisie...), ainsi que certaines autres fonctions utiles. La classe BDD contient les méthodes auxiliaires effectuant des recherches dans la base de données.

Enfin, le troisième package permet de gérer les interactions entre les classes principales (Individu et ses classes filles, Bien_immobilier et ses classes filles). La classe Annonce contient l'annonce soumise par un client et validée par le responsable. Le classe Creneau représente une brique élémentaire de temps sur laquelle on peut placer un rendez-vous (date de début). La classe Transaction permet de dater une transaction dans le temps, en reliant le client, le particulier intéressé, l'agent impliqué, le montant, le type de transaction, la date et le bien. La classe Notation contient l'ensemble des notes des agents à une date précise.



La classe Application contient la fenêtre principale de l'application. Cette dernière lance le point de départ de l'application. Les classes FenClient, FenAgent et FenRespo correspondent aux fenêtres qui s'ouvrent une fois que l'individu s'est authentifié dans la bonne case. Ces fenêtres ont la personne connectée comme attribut (par exemple FrenClient a le client qui vient de se connecter comme attribut).

La base de données

Nous avons choisi le logiciel SQLite pour notre base de données car il nous a permis de travailler facilement sur nos ordinateurs personnels en dehors de l'école.
Pour les biens immobiliers, nous avons choisi de faire 3 tables (constructible, habitable et non_habitable) auxquelles se rajoute la table adresse (id_adresse associé à un id_bien). Nous avons fait le choix de séparer les trois types de bien gérés par l'agence afin d'éviter d'avoir à rentrer les attributs qui ne correspondent au type.
Pour les individus, il y a 4 tables (agent_immobilier, client, particulier et respo_agence) qui correspondent chacune à un type de personne susceptible d'utiliser l'application avec les informations qui le caractérisent (mot de passe, identifiant, nom, mail, ..). De la même façon que pour les biens, nous avons fait le choix de séparer les différents individus.
Puis, nous avons créé les tables propres au rendez-vous (creneau), à la vente du bien (annonce et latransaction) et l'évaluation des agents (notation).
Enfin, il y a les tables « reception » qui permettent de créer un dialogue entre les personnes et qui permettent au responsable de valider certaines informations

3 : Organisation du travail et limites

Organisation

Avant de nous lancer dans le projet , nous avons réalisé un diagramme de Gantt pour avoir une vision d'ensemble du travail à réaliser et structurer l'avancée de notre projet.
Durant la partie d'analyse, nous nous sommes tout d'abord mis d'accord sur l'architecture générale du projet en définissant les classes ainsi que les méthodes à créer.
Ensuite nous avons chacun réalisé un ou plusieurs diagrammes (avec explications) de notre côté.
Enfin, nous avons rédigé ensemble le rapport en mettant en commun et en discutant de nos diagrammes.
Pour le développement, nous nous sommes séparés le travail en 3 parties que sont les 3 packages. Chacun a implémenté les classes de son package indépendamment avec les attributs et méthodes que nous avions défini dans la partie analyse. A l'aide de Github, nous avons pu récupérer le travail effectué au fur et à mesure mais aussi nous entraider en modiant ou ajoutant des éléments dans les autres packages. Finalement, nous avons trouvé un moyen de créer une interface graphique avec JoptionPane en même temps que nous codions nos classes.
Cependant, nous avons dépassé le planning que nous nous étions fixé au départ en prolongeant la partie développement jusqu'au matin du 9 Mai. En effet, l'intégration et la mise en commun du travail avec notamment l'utilisation de la base de données ont été laborieuses durant les derniers jours.
Finalement, nous avons commencé la rédaction du document de synthèse le 8 Mai en parallèle des derniers réglages au niveau du code. Nous avons partagé la rédaction, la création du ReadMe et la préparation de la soutenance entre nous.

Limites

Nous sommes content du résultat final de notre projet. Bien qu'il y ait de nombreux petits ajouts qui permettraient d'ajouter des fonctionnalités ou de corriger quelques erreurs, les fonctions principales fonctionnent et l'application est utilisable en l'état.
Un de nos problèmes concerne peut-être plus la partie orienté objet de notre programme. Quand nous avons choisi ce sujet, nous savions qu'il demanderait un gros travail sur la partie base de données. N'étant pas encore très à l'aise avec ces outils, nous avons décidé de les prendre en main assez rapidement. A partir de là, nous nous sommes peut-être trop concentrés sur cet aspect et n'avons pas utilisé pleinement le potentiel de l'orienté objet.
Dans la classe Transaction, nous avons également rencontré un problème pour la récupération du prix d'estimation du bien immobilier. En effet « 0 € » s'affiche au lieu de la bonne valeur.
Nous aurions aimé pousser un peu plus la partie statistiques avec des critères supplémentaires, il en est de même pour le nombre de biens et leurs attributs et ainsi avoir une recherche de bien plus élaborée.
Nous avons aussi un problème avec la fonctionnalité « valider une demande de rendez-vous ». Malgré nos efforts, nous n'avons pas réussi à empêcher l'erreur « database is locked » qui nous a empêcher de modifier notre table « rendezvous » pour le valider ou le supprimer.

Conclusion :

Nous avons bien aimé réaliser ce projet qui nous a appris beaucoup sur le Java et les bases de données mais aussi sur comment gérer un projet informatique en groupe.