2025_2024

Université des Sciences

Technologiques Houari Boumedien

Faculté d'informatique

Rapport de Projet sur la Gestion d'une agence immobilier en language java

agence ismosphere

Team leader : Baouchi rekia 222231455014

Concepteur : Aribi sarah

Devloppeur : Boudiaf kawter 222231341302

Redacteur : Cherat zehour 222231472207

Introduction :

- Ce projet vise à développer une application de gestion d'agence immobilière, répondant aux besoins croissants du secteur immobilier en matière d'efficacité opérationnelle . Avec l'évolution rapide du marché immobilier, les agences sont confrontées à des défis de gestion de plus en plus complexes, nécessitant des solutions technologiques avancées pour rester compétitives. L'objectif principal de cette application est de fournir une plateforme centralisée et conviviale pour la gestion des biens immobiliers, des clients et des transactions.

- Dans cette introduction, nous présenterons brièvement les différentes composantes du projet. Tout d'abord, la base de données constitue le fondement de l'application, stockant de manière sécurisée et organisée toutes les informations relatives aux biens immobiliers, aux clients et aux transactions. Ensuite, les interfaces utilisateur joueront un rôle crucial en offrant une expérience intuitive et conviviale aux utilisateurs, qu'il s'agisse des agents immobiliers, des clients ou des administrateurs de l'agence. Enfin, le code Java sera utilisé pour développer la logique métier de l'application, assurant le bon fonctionnement des fonctionnalités clés telles que la gestion des biens, le suivi des transactions et la génération de rapports.

- En combinant ces éléments, nous aspirons à créer une solution complète et fonctionnelle, facilitant la gestion des biens immobiliers, la communication avec les clients et le suivi des transactions. Ce document servira de guide pour le développement de l'application, en définissant les objectifs à atteindre et les étapes à suivre pour aboutir à une application robuste et intuitive, répondant aux besoins spécifiques des agences immobilières tout en offrant une expérience utilisateur optimale.

description des classes:

1)class tableauxutulisateurs :

la classe **tableauxutulisateurs**fait partie du package `project_immobilier`.

a) Attributs de la classe

Ces variables d'instance (ou attributs) sont privées et représentent les caractéristiques de la propriété :

- `adresse` : L'adresse de la propriété.
- `prix` : Le prix de la propriété.
- `surface` : La surface de la propriété.
- `type` : Le type de la propriété (par exemple, appartement, maison).
- `disponibilite` : La disponibilité de la propriété (par exemple, disponible, vendu).

b)Constructeur

Le constructeur de la classe `Property` prend cinq paramètres et les assigne aux attributs de la classe. Cela permet de créer une nouvelle instance de `Property` avec des valeurs spécifiques pour chaque attribut.

c)Méthodes d'accès (getters)

Ces méthodes sont des "getters" qui permettent d'accéder aux valeurs des attributs privés de la classe. Chaque méthode retourne la valeur de l'attribut correspondant :

- `getAdresse()` retourne l'adresse de la propriété.
- `getPrix()` retourne le prix de la propriété.
- `getSurface()` retourne la surface de la propriété.
- `getType()` retourne le type de la propriété.
- `getDisponibilite()` retourne la disponibilité de la propriété.

d)Utilisation de la classe `Propert`

une instance de `Property` est créée avec des valeurs spécifiques, puis les valeurs des attributs sont affichées à l'aide des méthodes `get`.

2)class `pagemesprop`:

ce codz permet d'afficher les propriétés (données immobilières) dans une table en récupérant les données depuis une base de données Oracle. Voici une explication détaillée des éléments nécessaires de cette classe :

a)Importations

```
import java.awt.Color;

import java.awt.EventQueue;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import javax.swing.JFrame;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.table.DefaultTableModel;
```

Ces lignes importent les classes nécessaires pour l'interface graphique (Swing) et pour la connexion et la gestion des données de la base de données.

b)Attributs de la classe

- `frame` : La fenêtre principale de l'application.
- `table` : Le composant Swing pour afficher les données sous forme de tableau.
- `connection` : La connexion à la base de données.
- `statement` : L'objet utilisé pour exécuter des requêtes SQL

c)Méthode `main`

La méthode `main` est le point d'entrée de l'application. Elle crée une instance de `pagemesprop`, rend la fenêtre visible et exécute une requête SQL pour afficher les données de la table `despbien`.

d)Constructeur

Le constructeur appelle la méthode `initialize` pour initialiser l'interface utilisateur et établir la connexion à la base de données.

e)Méthode `initialize`

- Initialise la fenêtre principale avec les paramètres spécifiés.
- Ajoute un panneau de défilement (`JScrollPane`) qui contient une table (`JTable`).
- Charge le driver JDBC Oracle et établit une connexion à la base de données.
- Crée un objet `Statement` pour exécuter des requêtes SQL.

f)Méthode `executeSelectQuery`

- Exécute une requête SQL passée en paramètre.
- Appelle la méthode `displaySearchResults` pour afficher les résultats dans la table.
- Ferme le `ResultSet`, le `Statement`, et la `Connection` après l'exécution de la requête pour libérer les ressources.

g)Méthode `displaySearchResults`

- Récupère le nombre de colonnes dans le `ResultSet`.
- Crée un modèle de table (`DefaultTableModel`) et ajoute des colonnes correspondant aux noms des colonnes du `ResultSet`.
- Ajoute les lignes de données du `ResultSet` au modèle de table.
- Associe le modèle de table au composant `JTable`.

h)Méthode `getFrame`

- Renvoie l'objet `JFrame` pour permettre un accès extérieur à la fenêtre principale.

3)class jaicompte:

a)But de la Classe

La classe `pagejaicompte` représente une interface graphique pour la connexion des utilisateurs. Elle permet aux utilisateurs de saisir leur adresse e-mail et leur mot de passe, puis tente de vérifier ces informations dans une base de données Oracle. Si les informations sont correctes, l'utilisateur est redirigé vers une autre page; sinon, un message d'erreur est affiché.

b)Attributs Principaux

- `frame` : La fenêtre principale de l'application.
- `textField_1` : Champ de texte pour l'adresse e-mail.
- `textField_2` : Champ de texte pour le mot de passe.
- `connection` : Connexion à la base de données.

c)Méthode `main`

La méthode `main` est le point d'entrée de l'application. Elle crée une instance de la classe et rend la fenêtre visible

d)Constructeur et Initialisation de la Base de Données

Le constructeur appelle deux méthodes : `initialize` pour configurer l'interface utilisateur, et `initializeDBConnection` pour établir une connexion à la base de données.

e)Initialisation de l'Interface Utilisateur

La méthode `initialize` configure les composants de l'interface graphique (labels, text fields, boutons) et ajoute des écouteurs d'événements aux boutons.

f)Gestion du Bouton "Suivant"

Le bouton "Suivant" vérifie les informations de connexion dans la base de données.

- Lorsqu'on clique sur le bouton "Suivant", les valeurs des champs de texte sont récupérées.
- Une requête SQL est préparée pour vérifier si un utilisateur avec l'email et le mot de passe spécifiés existe dans la base de données.
- Si les informations sont correctes, un message de succès est affiché et la fenêtre actuelle est fermée pour ouvrir la fenêtre de l'utilisateur.
- Sinon, un message d'erreur est affiché.

g)Méthode `getFrame`

- Cette méthode renvoie l'objet `JFrame` pour permettre un accès extérieur à la fenêtre principale.

4)Classe `pagedespsup`:

a)Objectif de la Classe

La classe `pagedespsup` permet de supprimer une entrée d'un bien immobilier de la base de données en utilisant l'ID du bien. Elle fournit une interface utilisateur pour saisir l'ID et des boutons pour effectuer l'action de suppression ou retourner à la page précédente.

b)Attributs Principaux

- `connection` : Gère la connexion à la base de données.
- `frame` : Fenêtre principale de l'application.
- `textField` : Champ de texte pour entrer l'ID du bien à supprimer.

c)Méthode `main`

La méthode `main` initialise l'application et rend la fenêtre visibl

d)Constructeur et Initialisation

Le constructeur appelle la méthode `initialize` pour configurer l'interface utilisateur.

e)Méthode `initialize`

Cette méthode configure la fenêtre et ses composants, établit la connexion à la base de données, et définit les écouteurs d'événements pour les boutons.

- La fenêtre (`frame`) est configurée avec ses dimensions, son type, et sa couleur de fond.
- La connexion à la base de données est établie en utilisant le driver Oracle JDBC.
- Les boutons "Suivant" et "Retour" sont créés et configurés, et les écouteurs d'événements sont ajoutés.
- Les labels et le champ de texte sont ajoutés à la fenêtre.

f)Méthode `idsup`

Cette méthode traite la suppression du bien en fonction de l'ID saisi dans le champ de texte.

- L'ID du bien est récupéré à partir du champ de texte.
- Une requête SQL est préparée pour supprimer le bien correspondant à l'ID spécifié.
- Si la suppression réussit, un message de succès est affiché. Sinon, un message indiquant qu'aucun bien n'a été trouvé est affiché.
- En cas d'erreur SQL, un message d'erreur est affiché.

g)Méthode `getFrame`

- Cette méthode renvoie l'objet `JFrame`, permettant d'accéder à la fenêtre principale de l'extérieur de la classe

.

5) Classe `pagedescription:`

a)Objectif de la Classe

La classe `pagedescription` permet aux utilisateurs de télécharger des photos et d'ajouter une description pour un bien immobilier via une interface graphique. Elle affiche les photos téléchargées et fournit une zone de texte pour entrer la description.

b)Attributs Principaux

- `frame` : La fenêtre principale de l'application.
- `lblPhoto1` : Label pour afficher la première photo téléchargée.
- `lblPhoto2` : Label pour afficher la deuxième photo téléchargée.
- `textArea` : Zone de texte pour entrer la description du bien.

c)Méthode `main`

La méthode `main` initialise l'application et rend la fenêtre visible.

d)Constructeur

Le constructeur par défaut ne fait rien d'autre que créer une instance de la classe. Cela permet de suivre les bonnes pratiques de Java en ayant un constructeur explicite même s'il ne contient pas de code.

e)Méthode `initialize`

Cette méthode configure la fenêtre et ses composants, y compris les labels pour les photos, la zone de texte pour la description et les boutons pour télécharger les photos.

- `frame` est configurée pour avoir une couleur de fond, une taille spécifique, et un layout null pour positionner manuellement les composants.
- Les labels `lblPhoto1` et `lblPhoto2` sont initialisés pour afficher les photos téléchargées.
- `textArea` est une zone de texte où l'utilisateur peut entrer la description du bien immobilier.
- Les boutons `btnUploadPhoto1` et `btnUploadPhoto2` permettent de télécharger des photos et sont associés à des ActionListeners pour gérer les événements de clic.

f)Méthode `selectAndDisplayPhoto`

Cette méthode permet de sélectionner une image à partir du système de fichiers et de l'afficher dans le label spécifié.

- Utilise `JFileChooser` pour ouvrir une boîte de dialogue de sélection de fichier filtrée pour les fichiers d'image.

- Si un fichier est sélectionné, il est affiché dans le `JLabel` spécifié après avoir été redimensionné pour s'adapter aux dimensions du label.

g)Méthode `getFrame`

- Cette méthode semble incomplète et est prévue pour renvoyer l'objet `JFrame`. Cependant, actuellement, elle renvoie `null`. Elle devrait être modifiée pour renvoyer `frame`.

h)Remarque

La méthode `getFrame` doit être correctement implémentée pour retourner l'instance de `JFrame`

Cela permettrait à d'autres classes ou parties de l'application d'accéder à la fenêtre de cette classe.

6)classe pagecomteuser:

a)Objectif de la Classe

La classe `pagecompteuser` représente une interface graphique pour gérer les actions des utilisateurs dans une application immobilière. Cette interface offre plusieurs options : ajouter des propriétés, supprimer un propriétaire, afficher les propriétaires existants et passer à la prochaine page d'opérations.

b)Attributs Principaux

- `frame` : La fenêtre principale de l'application.

c)Méthode `main`

La méthode `main` initialise l'application et rend la fenêtre visible

d)Constructeur

Le constructeur appelle la méthode `initialize` pour configurer les composants de l'interface utilisateur.

e)Méthode `initialize`

Cette méthode configure la fenêtre (`JFrame`) et ajoute des composants tels que des boutons et des labels

- Boutons et Labels : Les boutons et labels sont créés avec des propriétés spécifiques (couleurs, polices, positions) et ajoutés au frame.
- Listeners d'Action : Des `ActionListener` sont associés aux boutons pour répondre aux événements de clic.

- `btnNewButton_1` : Ouvre une nouvelle fenêtre pour ajouter des propriétés et ferme la fenêtre actuelle.
- `btnNewButton_1_1` : Ouvre une nouvelle fenêtre pour supprimer un propriétaire et ferme la fenêtre actuelle.
- `btnNewButton_1_1_1` : Ouvre une nouvelle fenêtre affichant les propriétaires existants et exécute une requête pour récupérer les données.

f)Méthode `getFrame`

Cette méthode retourne l'instance de la fenêtre principale `JFrame`.

7)classe pagechoixcompte:

a)Objectif de la Classe

La classe `pagechoixcompte` représente une interface utilisateur graphique pour permettre aux utilisateurs de choisir entre se connecter à un compte existant, créer un nouveau compte ou retourner à une page précédente. Cette interface est une partie d'une application plus large qui gère des informations immobilières.

b)Attributs Principaux

- `frame` : La fenêtre principale de l'application.

c)Méthode `main`

La méthode `main` initialise l'application et rend la fenêtre visible.

d)Constructeur

Le constructeur appelle la méthode `initialize` pour configurer les composants de l'interface utilisateur.

e)Méthode `initialize`

Cette méthode configure la fenêtre (`JFrame`) et ajoute des composants tels que des boutons et des labels.

f)Description des Composants

- Label "Bienvenues :" : Affiche un message de bienvenue.
- Bouton "J'ai un compte" : Permet à l'utilisateur de se connecter à un compte existant. En cliquant sur ce bouton, la fenêtre actuelle se ferme et une nouvelle fenêtre de connexion (`pagejaicompte`) s'ouvre.

- Bouton "Créer un compte" : Permet à l'utilisateur de créer un nouveau compte. En cliquant sur ce bouton, la fenêtre actuelle se ferme et une nouvelle fenêtre de création de compte (`page_info_vc`) s'ouvre.
- Bouton "Retour" : Permet à l'utilisateur de revenir à la page précédente. En cliquant sur ce bouton, la fenêtre actuelle se ferme et une nouvelle fenêtre de choix (`page_de_choix`) s'ouvre.

g)Méthode `getFrame`

Cette méthode retourne l'instance de la fenêtre principale `JFrame`.

8)pagedescription:

Ce code définit une interface graphique en Java pour permettre aux utilisateurs de télécharger des photos et d'ajouter des descriptions pour des bien immobilier. Utilisant la bibliothèque Swing, l'interface est conçue pour être simple et intuitive.

a)Structure Générale

1. Classe `pagedescription` : Contient la définition de la fenêtre principale et des composants de l'interface utilisateur.
2. Méthode `main` : Point d'entrée de l'application. Utilise `EventQueue.invokeLater` pour s'assurer que l'interface graphique est créée et mise à jour sur le thread de dispatch des événements Swing
3. Constructeur `pagedescription` : Constructeur par défaut. Pour l'instant, il est vide mais peut être utilisé pour initialiser des variables d'instance si nécessaire
4. Méthode `initialize` : Configure et initialise l'interface utilisateur
5. Méthode `selectAndDisplayPhoto` : Ouvre un sélecteur de fichiers pour choisir une image, puis affiche cette image dans un label donné
6. Méthode `getFrame` : Pour l'instant, cette méthode n'est pas implémentée correctement. Elle devrait retourner le cadre principal (`frame`).

b)Composants de l'Interface Utilisateur

1. Fenêtre Principale (`JFrame`) :
    - `frame` : Fenêtre principale de l'application. Elle est configurée avec un fond orange, une taille de 800x599 pixels, et une fermeture par défaut pour quitter l'application.
2. Étiquettes (`JLabel`) :

- `lblPhoto1` et `lblPhoto2` : Étiquettes pour afficher les photos téléchargées. Initialement vides, elles sont positionnées en haut de la fenêtre.

3. Libellé de Description (`JLabel`) :
   - `lblDescription` : Libellé invitant l'utilisateur à ajouter une description du bien. Positionné sous les étiquettes des photos.
4. Zone de Texte (`JTextArea`) :
   - `textArea` : Zone de texte où l'utilisateur peut entrer la description du bien. Positionnée sous le libellé de description.
5. Boutons de Téléchargement de Photos (`JButton`) :
   - `btnUploadPhoto1` et `btnUploadPhoto2` : Boutons pour télécharger des photos. Ils sont associés à des écouteurs d'événements pour gérer le téléchargement et l'affichage des photos dans les étiquettes correspondantes.

9)pagechoixcompte:

Ce code définit une interface graphique en Java pour une page de choix de compte dans une application de gestion immobilière. Les utilisateurs peuvent choisir de se connecter s'ils ont déjà un compte, de créer un nouveau compte ou de revenir à la page précédente. La bibliothèque Swing est utilisée pour construire cette interface.

a)Structure Générale

1. Classe `pagechoixcompte` : Contient la définition de la fenêtre principale et des composants de l'interface utilisateur.
2. Méthode `main` : Point d'entrée de l'application. Utilise `EventQueue.invokeLater` pour s'assurer que l'interface graphique est créée et mise à jour sur le thread de dispatch des événements Swing.
3. Constructeur `pagechoixcompte` : Initialise l'interface utilisateur en appelant la méthode `initialize`.
4. Méthode `initialize` : Configure et initialise l'interface utilisateur.
5. Méthode `getFrame` : Retourne la fenêtre principale (`frame`). Cela permet à d'autres parties du programme d'accéder à la fenêtre principale.

b)Composants de l'Interface Utilisateur

1. Fenêtre Principale (`JFrame`) :
   - `frame` : Fenêtre principale de l'application. Elle est configurée avec un fond orange, une taille de 800x600 pixels, et une fermeture par défaut pour quitter l'application.
2. Étiquette de Bienvenue (`JLabel`) :
   - `lblNewLabel` : Étiquette affichant "Bienvenues :" en rouge foncé avec une police de grande taille. Positionnée en haut de la fenêtre.
3. Boutons (`JButton`) :
   - `btnNewButton_1` : Bouton "J'ai un compte" en rouge foncé avec une police de taille moyenne et un fond orange foncé. Positionné au milieu de la fenêtre. Il ouvre la page de connexion (`pagejaicompte`).

- `btnNewButton_1_1` : Bouton "Créer un compte" en rouge foncé avec une police de taille moyenne et un fond orange foncé. Positionné sous le bouton "J'ai un compte". Il ouvre la page de création de compte (`page_info_vc`).
- `btnRetour` : Bouton "Retour" en rouge foncé avec une police de petite taille et un fond orange foncé. Positionné en bas à gauche de la fenêtre. Il retourne à la page précédente (`page_de_choix`).

c)Listeners d'Événements

1. Listener pour le bouton "J'ai un compte" :
   - Ferme la fenêtre actuelle.
   - Crée une nouvelle instance de la page de connexion (`pagejaicompte`).
   - Rend la page de connexion visible.
2. Listener pour le bouton "Créer un compte" :
   - Ferme la fenêtre actuelle.
   - Crée une nouvelle instance de la page de création de compte (`page_info_vc`).
   - Rend la page de création de compte visible.
3. Listener pour le bouton "Retour" :
   - Ferme la fenêtre actuelle.
   - Crée une nouvelle instance de la page précédente (`page_de_choix`).
   - Rend la page précédente visible.

10)page_rechercheltabien:

Ce code définit une interface graphique Java pour afficher les résultats des propriétés immobilières en fonction de critères de recherche spécifiques. Il utilise Swing pour créer l'interface et JDBC pour interagir avec une base de données Oracle.

a)Ajout des Champs de Recherche

Les champs de recherche permettent à l'utilisateur de spécifier des critères tels que le prix, la surface et le nombre de pièces pour filtrer les résultats immobiliers.

b)Composants de l'Interface Utilisateur Ajoutés

1. Étiquettes et Champs de Texte pour les Critères de Recherche
   - Prix Minimum
   - Prix Maximum
   - Surface Minimum
   - Surface Maximum
   - Nombre de Pièces
2. Bouton de Recherche

3. Action Listener pour le Bouton de Recherche:

   Lorsque le bouton de recherche est cliqué, les critères de recherche sont récupérés et la fenêtre actuelle est fermée. Une nouvelle instance de `page_resultabien` est créée et les résultats sont affichés dans la nouvelle fenêtre

11)page_resultabien:

Ce code définit une interface graphique en Java pour afficher les résultats de propriétés immobilières dans un tableau (`JTable`). Les données sont récupérées à partir d'une base de données Oracle. La bibliothèque Swing est utilisée pour construire cette interface, et JDBC (Java Database Connectivity) est utilisé pour interagir avec la base de données.

a)Structure Générale

1. Classe `page_resultabien` : Contient la définition de la fenêtre principale et les composants de l'interface utilisateur, ainsi que la connexion à la base de données.
2. Méthode `main` : Point d'entrée de l'application. Utilise `EventQueue.invokeLater` pour s'assurer que l'interface graphique est créée et mise à jour sur le thread de dispatch des événements Swing.
3. Constructeur `page_resultabien` : Constructeur par défaut
4. Méthode `initialize` : Configure et initialise l'interface utilisateur et charge les données de la base de données dans le tableau
5. Méthode `getFrame` : Retourne la fenêtre principale (`frame`). Cela permet à d'autres parties du programme d'accéder à la fenêtre principale.

b)Composants de l'Interface Utilisateur

1. Fenêtre Principale (`JFrame`) :
   - `frame` : Fenêtre principale de l'application. Elle est configurée avec un fond orange, une taille de 809x600 pixels, et une fermeture par défaut pour quitter l'application.
   - Le conteneur de la fenêtre est configuré pour utiliser un gestionnaire de mise en page nul (`null`), ce qui signifie que les composants doivent être positionnés manuellement en utilisant les méthodes `setBounds`.
2. Tableau de Résultats (`JTable`) :
   - `table` : Tableau où les données des propriétés seront affichées. Il est configuré avec un fond couleur beige.

- `scrollPane` : Permet de faire défiler le tableau si les données dépassent la taille visible. Il contient le tableau et est ajouté à la fenêtre.

c)Connexion à la Base de Données

1. Chargement du Driver JDBC : Le driver JDBC pour Oracle est chargé dynamiquement.
2. Établissement de la Connexion : Connexion à la base de données Oracle en utilisant l'URL, le nom d'utilisateur et le mot de passe
3. Création de l'Instruction SQL : Création d'une instance de `Statement` pour exécuter des requêtes SQL
4. Exécution de la Requête SQL : Requête SQL pour récupérer les informations des propriétés immobilières. Les résultats sont stockés dans un `ResultSet`
5. Traitement des Résultats :

*Récupération du Nombre de Colonnes : Le nombre de colonnes est obtenu à partir des métadonnées du `ResultSet`

*Création du Modèle de Table (`DefaultTableModel`) : Un modèle de table par défaut est créé pour stocker les données.

*Ajout des Colonnes au Modèle : Les noms des colonnes sont ajoutés au modèle

*Ajout des Lignes au Modèle : Les données sont lues ligne par ligne depuis le `ResultSet` et ajoutées au modèle.

*Application du Modèle au Tableau : Le modèle de table est appliqué au tableau, ce qui affiche les données

12)page_info_vc:

Voici une explication détaillée de chaque partie du code et des méthodes:

a)import:

import java.awt.Color;

import java.awt.Component;

import java.awt.EventQueue;

import java.awt.Font;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JTextField;

Ces imports incluent les classes nécessaires pour créer l'interface utilisateur, gérer les événements, et interagir avec une base de données Oracle.

13)page_desbien:

Ce code Java crée une interface graphique pour une application de gestion immobilière, permettant aux utilisateurs d'ajouter des informations sur une propriété. Il inclut des champs pour saisir diverses informations et un bouton pour soumettre ces informations à une base de données Oracle. Voici une explication détaillée de chaque partie du code et des méthodes utilisées :

a)Attributs

- `private Connection connection;` et `private java.sql.Statement statement;`
  - Ces attributs gèrent la connexion à la base de données et l'exécution des requêtes SQL.
- `private JFrame frame;`
  - Cet attribut représente la fenêtre principale de l'interface utilisateur.

- `private JTextField wilayaV;,private JTextField municipalV;,private JTextField typeV;,private JTextField louerV;,private JTextField interfaceV;,private JTextField prix;`
    - Ces attributs sont des champs de texte pour saisir les informations sur la propriété.

b)Méthodes

`main(String[] args)`

- Le point d'entrée de l'application. Il utilise `EventQueue.invokeLater` pour s'assurer que la création de l'interface utilisateur se fait dans le thread de dispatching des événements de Swing.
- Crée une instance de `page_desbien` et rend la fenêtre (`frame`) visible.

`page_desbien()`

- Le constructeur de la classe. Il ne fait rien dans ce cas, mais il pourrait être utilisé pour initialiser des variables ou des paramètres.

`initialize()`

- Configure les composants de l'interface utilisateur, initialise la connexion à la base de données et ajoute les composants à la fenêtre (`JFrame`).
    - Configuration de la fenêtre (`JFrame`) :
        - `frame = new JFrame();`
        - `frame.setType(JFrame.Type.POPUP);`
        - `frame.setBounds(100, 100, 800, 600);`
        - `frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);`
        - `frame.getContentPane().setLayout(null);`
        - `frame.getContentPane().setBackground(new Color(255, 165, 0));`
    - Connexion à la base de données Oracle :
        - `Class.forName("oracle.jdbc.driver.OracleDriver");`
        - `connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system", "irekia15");`
    - Ajout des composants (labels, champs de texte, bouton) :
        - `JLabel lblNewLabel = new JLabel("Ajouter les information a propos de votre bien :");`
        - `lblNewLabel.setForeground(Color.WHITE);`
        - `lblNewLabel.setFont(new Font("Tempus Sans ITC", Font.PLAIN, 35));`
        - `lblNewLabel.setBounds(10, 11, 726, 62);`
        - `frame.getContentPane().add(lblNewLabel);`
        - De même pour les autres `JLabel`, `JTextField` et le `JButton`.
    - ActionListener pour le bouton "Suivant" :
        - Récupère les valeurs saisies.
        - Exécute une requête SQL pour insérer les données dans la base de données.

        - Affiche un message de confirmation.
        - Ouvre une nouvelle interface `pagedescription`.
- `getFrame()`
    - Retourne la fenêtre (`frame`) de l'interface utilisateur

14)Classe principale `page_info_vc:`

- `main` : L'entrée principale de l'application. Utilise `EventQueue.invokeLater` pour s'assurer que l'interface utilisateur est créée et mise à jour sur le thread de l'événement.
- Constructeur `page_info_vc` : Appelle la méthode `initialize` pour configurer l'interface utilisateur et établir la connexion à la base de données

a)Méthode `initialize`

- Chargement du pilote JDBC : Charge le pilote Oracle JDBC pour permettre la connexion à une base de données Oracle.
- Établissement de la connexion : Utilise `DriverManager.getConnection` pour se connecter à la base de données Oracle avec les informations d'identification fournies.
- Configuration de la fenêtre (`JFrame`) : Crée la fenêtre principale de l'application et configure son apparence

b)Ajout des composants à la fenêtre

- Labels (`JLabel`) et champs de texte (`JTextField`) : Ajoute des composants pour afficher des textes et pour permettre à l'utilisateur de saisir des informations comme le nom, l'email, le mot de passe, et le numéro de téléphone.

c)Ajout des boutons et gestion des événements

- Boutons `Suivant` et `Retour` : Crée deux boutons et les configure.
- Listeners pour les boutons : Ajoute des `ActionListener` pour définir ce qui se passe lorsque l'utilisateur clique sur les boutons.
    - `btnSuivant` : Appelle la méthode `saveData` pour enregistrer les données saisies.
    - `btnRetour` : Ferme la fenêtre actuelle et ouvre la fenêtre précédente (`pagechoixcompte`).

15)page_de_garde:

crée une interface graphique pour une page de garde d'un projet immobilier. Je vais le décomposer et expliquer chaque partie :

a)Importations

```
import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.Image;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.SwingConstants;
```

Ces lignes importent les classes nécessaires pour créer l'interface graphique, telles que `JFrame` pour la fenêtre principale, `JLabel` pour afficher du texte et des images, ainsi que des classes pour gérer les couleurs, les polices et les événements d'interface utilisateur

b)Classe `page_de_garde`

Cette classe représente la page de garde de l'application.

c)Attributs de la classe

- `frame` est la fenêtre principale de l'application.
- `lblImage` est un composant JLabel utilisé pour afficher une image.

d)Méthode `main`

Cette méthode est le point d'entrée de l'application. Elle crée une instance de la classe `page_de_garde` et appelle sa méthode `initialize` pour initialiser l'interface graphique.

e)Constructeur

Ce constructeur est vide car il n'effectue aucune initialisation spécifique.

d)Méthode `initialize`

Cette méthode initialise l'interface graphique en créant et configurant les composants, en définissant leur position et en les ajoutant à la fenêtre principale (`frame`).

e)Méthode `setImage`

Cette méthode charge une image à partir d'un chemin spécifié et l'affiche dans le composant `lblImage`.

f)Méthode `getFrame`

Cette méthode renvoie la fenêtre principale de l'application.

16)page_choix

Ce code Java crée une classe graphique pour une page de choix dans un projet immobilier. Voici une explication détaillée de chaque partie **:**

a)Importations

```
import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JButton;
import javax.swing.JFrame;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
```

Ces lignes importent les classes nécessaires pour créer une interface graphique (Swing), notamment les boutons, les couleurs, les polices et les écouteurs d'événements.

b)Classe `page_de_choix`

Cette classe représente la page de choix de l'application.

c)Attribut de la classe

C'est la fenêtre de l'interface graphique.

d)Méthode `main`

La méthode `main` est le point d'entrée de l'application. Elle crée une instance de `page_de_choix` et appelle sa méthode `initialize` pour démarrer l'interface graphique.

e)Constructeur

Ce constructeur est vide car il n'effectue aucune initialisation spécifique.

f)Méthode `initialize`

Cette méthode initialise l'interface graphique. Elle crée la fenêtre, ajoute des boutons avec des actions associées, et définit les propriétés de ces boutons (comme la couleur, la police, la taille et la position).

Dans cette méthode, trois boutons sont ajoutés :

1. Un bouton pour rechercher un bien.
2. Un bouton pour déposer une annonce.
3. Un bouton pour la gestion immobilière.

Chaque bouton a un actionneur (`ActionListener`) qui réagit lorsqu'il est cliqué. Lorsqu'un bouton est cliqué, il ferme la fenêtre actuelle (`frame.dispose()`) et ouvre une nouvelle fenêtre correspondant à l'action associée.

g)Méthode `getFrame`

Cette méthode renvoie la fenêtre de l'interface graphique.

16)add_photo:

Ce code Java représente une méthode statique appelée `uploadPhoto` qui permet à l'utilisateur de sélectionner un fichier image à partir de son système de fichiers local. Voici une explication détaillée :

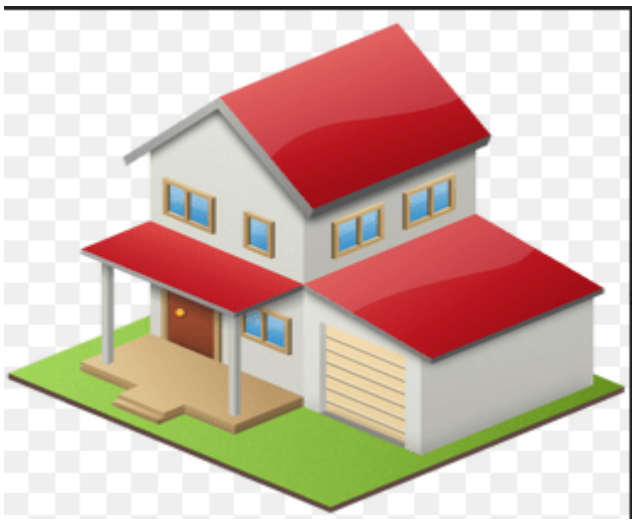
a)Méthode `uploadPhoto`

Cette méthode permet à l'utilisateur de sélectionner un fichier image à partir du système de fichiers local. Voici ce qu'elle fait :

1. Création du `JFileChooser`: Un objet `JFileChooser` est créé pour permettre à l'utilisateur de naviguer dans son système de fichiers et de choisir un fichier.
2. Configuration du filtre de fichier: Le filtre de fichier est configuré pour n'afficher que les fichiers d'images avec les extensions "jpg", "jpeg", "png", et "gif".
3. Affichage de la boîte de dialogue de sélection de fichier: La méthode `showOpenDialog(parentFrame)` affiche une boîte de dialogue permettant à l'utilisateur de choisir un fichier. Le paramètre `parentFrame` est utilisé pour spécifier la fenêtre parente de la boîte de dialogue.
4. Traitement de la sélection de fichier: Si l'utilisateur sélectionne un fichier et clique sur le bouton "Ouvrir" (ou équivalent), la méthode `showOpenDialog` retourne `JFileChooser.APPROVE_OPTION`, et nous récupérons alors le fichier sélectionné avec `fileChooser.getSelectedFile()`.
    - Si aucun fichier n'est sélectionné ou si l'utilisateur annule l'action, la méthode retourne `null`.
5. Retour du fichier sélectionné ou `null`: La méthode retourne le fichier sélectionné, s'il y en a un, ou `null` sinon.

b)Utilisation

Pour utiliser cette méthode, vous pouvez l'appeler depuis une autre classe en passant la fenêtre parente (`JFrame`) comme argument.



17)main_application:

ce code gère le processus d'affichage de deux fenêtres différentes dans une séquence chronologique. Voici une explication détaillée de chaque partie :

a)Package et Dépendances

```
package project_immobilier;
```

```
import java.awt.EventQueue;

import javax.swing.JFrame;

import javax.swing.Timer;
```

Ce code importe les classes nécessaires pour créer une interface graphique (Swing) ainsi que la classe Timer pour gérer le délai avant l'affichage de la deuxième fenêtre.

b)Classe `mainApplication`

Ce code importe les classes nécessaires pour créer une interface graphique (Swing) ainsi que la classe Timer pour gérer le délai avant l'affichage de la deuxième fenêtre

c)Classe `mainApplication`

- `frameDeGarde` représente la fenêtre de la page de garde.
- `frameDeChoix` représente la fenêtre de la page de choix.

d)Méthode `main`

- `frameDeGarde` représente la fenêtre de la page de garde.
- `frameDeChoix` représente la fenêtre de la page de choix.

e)Méthode `main`

Cette méthode initialise l'application en affichant d'abord la page de garde. Ensuite, elle crée un Timer qui après 3 secondes (3000 millisecondes), ferme la page de garde et ouvre la page de choix en appelant la méthode `openPageDeChoix`.

f)Méthode `openPageDeChoix`

Cette méthode crée une instance de la classe `page_de_choix`, initialise la fenêtre de la page de choix et la rend visible.