RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES HOUARI BOUMEDIENE
FACULTÉ D'INFORMATIQUE

Rapport de Projet ApplicationGestionAgence Immobilière

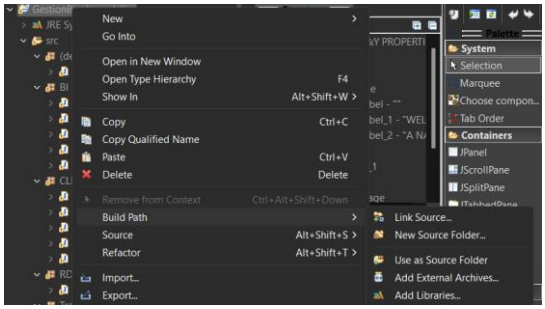
BELASSAM AKRAM 212134081260

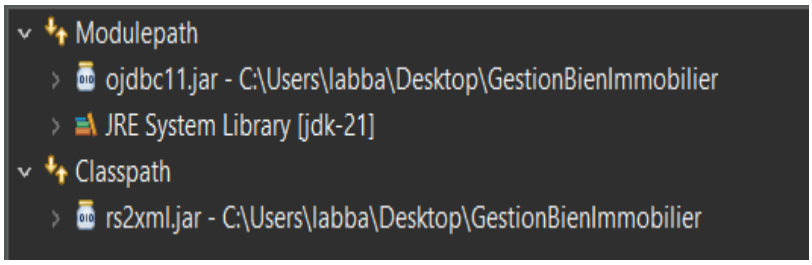
LABBAOUI YENNI 202031058828

BOUKECHICHE ISHAK 212137038994

ADDAOUD ABDERRAOUF 212139088236

Connexion des bibliothèques





- On connecte les JARS externes ‘’OJDBC’’ et ‘’RS2XML’’ qui vont servir a lire et a ecrire dans la base de données

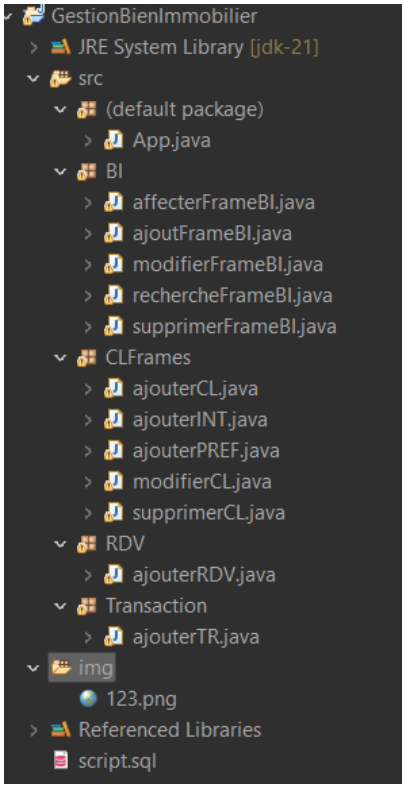
creation des packages

- On a créé 5 packages :

  . Default packages : qui contient la **JFRAME** (App.java) la classe principalede notre application et qui va contenir les differents **PANEL** de notre interface

  . BI : qui contient les **JFRAMES** de la gestion des biens immobiliers
  . CLFrames : qui contient les **JFRAMES** de la gestion des clients
  . RDV : qui contient les **JFRAMES** de la gestion des RDV
  . Transaction : qui contient les **JFRAMES** de la gestion des transactions

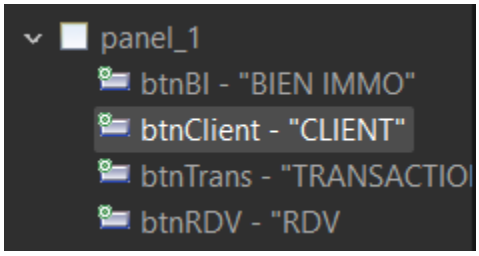
- On a créé un source folder : qui contient les images utilisés pour l’interface



Panels :

- On a organisé tous les **PANELS** dans la Jframe (App.java) qui va contenir :

. PANEL_1 : contient les 4 premiers boutons principaux (Bienimmobilier, client, RDV, Transaction)



. Layered Panel : qui contient plusieurs PANEL des 4 premiers boutons plus un Panel de la premiere page :

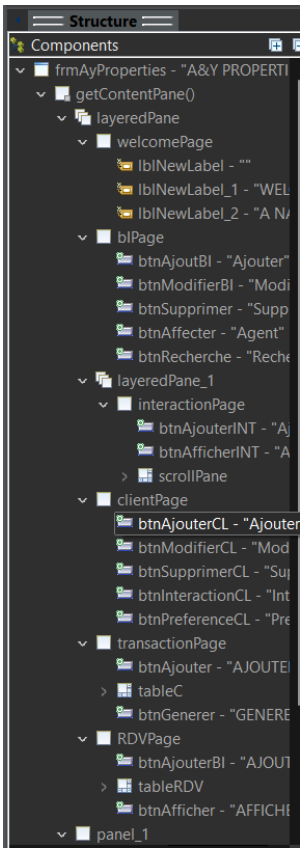
. WelcomePage : Contient la page de BIENVENUE (l'image et le nom de l'agence)

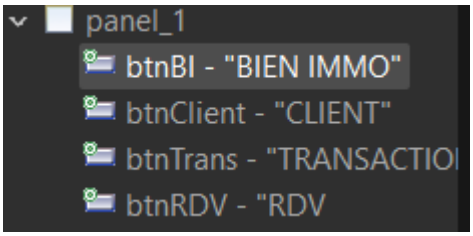
. BiPage : contient les differents boutons utilisés pour la gestion des bien immobiliers (Ajouter, Modifier, Supprimer ….)

. ClientPage : contient les differents boutons utilisés pour la gestion des clients (Ajouter, Modifier, Supprimer ….)

. TransactionPage : contient les differents boutons utilisés pour la gestion des Transactions

. RDVPage : contient les differents boutons utilisés pour la gestion des RDV





Code dans les JFRAMES :

- Dans chaque JFRAME on a importé les bibliothèques nécessaires tels que (java.util.sql.*…etc) en plus des packages pour avoir accés aux classes d'autres packages

```
import BI.*;

import CLFrames.*;
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 java.awt.Color;
import javax.swing.JPanel;
import java.awt.BorderLayout;
import javax.swing.JSplitPane;
import javax.swing.JLabel;
import javax.swing.SwingConstants;
import java.awt.FlowLayout;
import javax.swing.JSeparator;
import java.awt.GridLayout;
import java.awt.Image;

import javax.swing.BoxLayout;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.Font;
import java.awt.SystemColor;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.AbstractAction;
import javax.swing.Action;
import java.awt.Component;
import javax.swing.JLayeredPane;
import javax.swing.JOptionPane;

import java.awt.CardLayout;
import javax.swing.border.EmptyBorder;
import javax.swing.border.SoftBevelBorder;
import javax.swing.border.BevelBorder;
```

- Dans chaque JFRAME on a étalit la connexion a notre base de données dans la méthode (initilize()) et on l'a entouré avec un « try catch » qui retourne l'erreur ("LOGIN TO DATABASE DENIED ! \n username or password incorrect \n", "ERROR!") s'il y a des erreurs de connexion a la base de données

```
private void initialize() {


    try {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE","labbaoui","yenni");
    } catch (Exception e1) {

        JOptionPane.showMessageDialog(null, "LOGIN TO DATABASE DENIED ! \n username or password incorrect \n", "ERROR!",JOptionPane.ERROR_MESSAGE);

    }
```

App.java :

- Dans le code de cette JFRAME on a pu Initialisé les differents PANEL et les composants de cette JFRAME déjà cités au-dessus et configurer les boutons
- On a configuré les boutons dans les methodes (public void actionPerformed(ActionEvent e)) de chaque bouton ou on a etablit la connexion avec la JFRAME qui correspond a chaque un

```
JButton btnAjoutBI = new JButton("Ajouter");
btnAjoutBI.setForeground(new Color(255, 255, 255));
btnAjoutBI.setBackground(new Color(0, 255, 127));
btnAjoutBI.setBorder(null);
btnAjoutBI.setFont(new Font("Montserrat", Font.BOLD, 15));
btnAjoutBI.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        ajoutFrameBI ns = new ajoutFrameBI();
        ns.ajoutFrameBI();


    }
});
```

```
JLayeredPane layeredPane = new JLayeredPane();
layeredPane.setBounds(203, 0, 509, 470);
frmAyProperties.getContentPane().add(layeredPane);
layeredPane.setLayout(new CardLayout(0, 0));

JPanel welcomePage = new JPanel();
welcomePage.setBackground(Color.WHITE);
layeredPane.add(welcomePage, "name_25964551188300");
welcomePage.setLayout(null);

JLabel lblNewLabel = new JLabel("");
Image img = new ImageIcon(this.getClass().getResource("/123.png")).getImage();
lblNewLabel.setBounds(79, 11, 350, 350);
lblNewLabel.setIcon(new ImageIcon(img));
welcomePage.add(lblNewLabel);

JLabel lblNewLabel_1 = new JLabel("WELCOME TO A&Y PROPERTIES !");
lblNewLabel_1.setForeground(new Color(47, 79, 79));
lblNewLabel_1.setFont(new Font("Montserrat Black", Font.BOLD, 19));
lblNewLabel_1.setBounds(78, 359, 361, 24);
welcomePage.add(lblNewLabel_1);

JLabel lblNewLabel_2 = new JLabel("A NAME YOU CAN TRUST");
lblNewLabel_2.setForeground(new Color(47, 79, 79));
lblNewLabel_2.setFont(new Font("Montserrat", Font.BOLD, 13));
lblNewLabel_2.setBounds(168, 382, 173, 17);
welcomePage.add(lblNewLabel_2);
```

- On a configuré les 4 premiers boutons pour qu'ils affichent les PANELS superposés correspondants

```
JButton btnTrans = new JButton("TRANSACTION");
btnTrans.setFocusPainted(false);
btnTrans.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        layeredPane.removeAll();
        layeredPane.add(transactionPage);
        layeredPane.repaint();
        layeredPane.revalidate();
    }
});
btnTrans.setBorder(new TitledBorder(null, "", TitledBorder.LEADING, TitledBorder.TOP, null, null));
btnTrans.setBackground(Color.WHITE);
btnTrans.setFont(new Font("Montserrat Black", Font.PLAIN, 14));
btnTrans.setBounds(29, 257, 131, 58);
panel_1.add(btnTrans);

JButton btnRDV = new JButton("RDV\r\n");
btnRDV.setFocusPainted(false);
btnRDV.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        layeredPane.removeAll();
        layeredPane.add(RDVPage);
        layeredPane.repaint();
        layeredPane.revalidate();
    }
```

- On a ajouté une image sur le panel WelcomePage avec ce code :

```
JPanel welcomePage = new JPanel();
welcomePage.setBackground(Color.WHITE);
layeredPane.add(welcomePage, "name_25964551188300");
welcomePage.setLayout(null);

JLabel lblNewLabel = new JLabel("");
Image img = new ImageIcon(this.getClass().getResource("/123.png")).getImage();
lblNewLabel.setBounds(79, 11, 350, 350);
lblNewLabel.setIcon(new ImageIcon(img));
welcomePage.add(lblNewLabel);
```

JFRAMES du Package BI :

- Dans les JFRAMES de ce package on a fait l'ajout, la modification, la suppression… de sorte que quand on clique sur Ajouter La JFRAME AJOUTER apparait et on pourra ajouter nos informations

- On a créés 2 attributs **Connection** et **statement** pour la connection et l'execution des requettes SQL (on a utilisé SQLPLUS) comme gestionnaire de BDD)
- Pour saisir les informations souhaités on a recupérés les chaines de caractères entrés par l'utilisateur avec la fonction (getText()) chacun dans une variable qu'on a créé
- Ensuite on a declaré une variable de type STRING pour introduire la requette SQL
- on l'a executé en utilisant la fonction `statement.executeQuery(sql);`
- on affiche un message d'erreur ou reussite en utilisant `JOptionPane.showMessageDialog(null, "agent à été ajouté", "ADDED!",JOptionPane.INFORMATION_MESSAGE);`
- On entoure l'execution avec un (try catch) en cas d'erreur

```
JButton btnAjouterBI = new JButton("AJOUTER\r\n");
btnAjouterBI.setForeground(Color.WHITE);
btnAjouterBI.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String NomAG = getNomAG.getText();
        String PrenomAG = getPrenomAG.getText();
        String PhoneAG = getPhoneAG.getText();
        String EmailAG = getEmailAG.getText();

        String sql = "INSERT INTO AGENT VALUES ('"+NomAG+"','"+PrenomAG+"', '"+PhoneAG+"', '"+EmailAG+"')";
        try {
            statement = connection.createStatement();
            statement.executeQuery(sql);
            JOptionPane.showMessageDialog(null, "agent à été ajouté", "ADDED!",JOptionPane.INFORMATION_MESSAGE);
        } catch (Exception e1) {

            JOptionPane.showMessageDialog(null, "ERROR OCCURED ! \n make sure that the information entered is correct and no field is empty\n", "ERROR!",J
            // TODO Auto-generated catch block
        }
```

- On a fait presque la meme chose pour les autres fonctionalités tel que modifier ou supprimer

```
JButton btnRechercheBI = new JButton("SUPPRIMER\r\n");
btnRechercheBI.setForeground(new Color(255, 255, 255));
btnRechercheBI.setFont(new Font("Montserrat Black", Font.BOLD, 14));
btnRechercheBI.setBorderPainted(false);
btnRechercheBI.setBackground(new Color(220, 20, 60));
btnRechercheBI.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String searchBI = getSearch.getText();
        String sql = "DELETE FROM BienImmobilier BI WHERE BI.Num = '"+searchBI+"'";

        try {
            statement = connection.createStatement();
            statement.executeQuery(sql);
        } catch (Exception e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }

    JOptionPane.showMessageDialog(null, "le bien immobilier à été supprimé ", "DELETED!",JOptionPane.INFORMATION_MESSAGE);
```

- Et comme dans chaque JFRAME il faut initialiser les composants de cette JFRAME

JFRAMES de CLFrames

- Ce package concerne les fonctionalités de gestion des clients (ajouter, modifier, supprimer…etc
- Comme dans les autres JFRAMES :
- On initialise , on etablit la connection, on execute les requettes dont on a besoin dans les methodes d'action de chaque bouton (`public void actionPerformed(ActionEvent e))`
- On prend exemple de supprimer Client

```
JButton btnSupprimer = new JButton("Supprimer");
btnSupprimer.setForeground(new Color(255, 255, 255));
btnSupprimer.setBorder(null);
btnSupprimer.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String Nom = getNomS.getText();
        String Prenom = getPrenomS.getText();
        String sql = "DELETE FROM Client WHERE Nom = '"+Nom+"' AND Prenom = '"+Prenom+"'" ;

        try {
            statement = connection.createStatement();
            statement.executeQuery(sql);
            JOptionPane.showMessageDialog(null, "le bien immobilier à été supprimé ", "DELETED!",JOptionPane.INFORMATION_MESSAGE);
        } catch (Exception e1) {

            JOptionPane.showMessageDialog(null, "ERROR OCCURED ! \n make sure that the information entered is correct and no field is empty\n", "ERROR!",J
            // TODO Auto-generated catch block
        }


    }
```

Affichage :

- Pour afficher les table de notre base de données on ajoute un tableau sur notre PANEL
- On a créé une variable resultat ou on va mettre le resultat de lecture depuis notre base de données
- Puis on a utilisé `table_1.setModel(DbUtils.resultSetToTableModel(rs));` Pour transformer notre resultat en un tableau pour l'afficher

```
JButton btnGenerer = new JButton("GENERER \r\n");
btnGenerer.setForeground(new Color(255, 255, 255));
btnGenerer.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        String sql = "SELECT NomCL AS NOM, PrenomCL AS PRENOM, Type, NumBI, DateD, DateF, Montant FROM Transaction";
        try {
            statement = connection.createStatement();
            statement.executeQuery(sql);
            ResultSet rs = statement.executeQuery(sql);
            table_1.setModel(DbUtils.resultSetToTableModel(rs));


        } catch (Exception e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }
    }
```

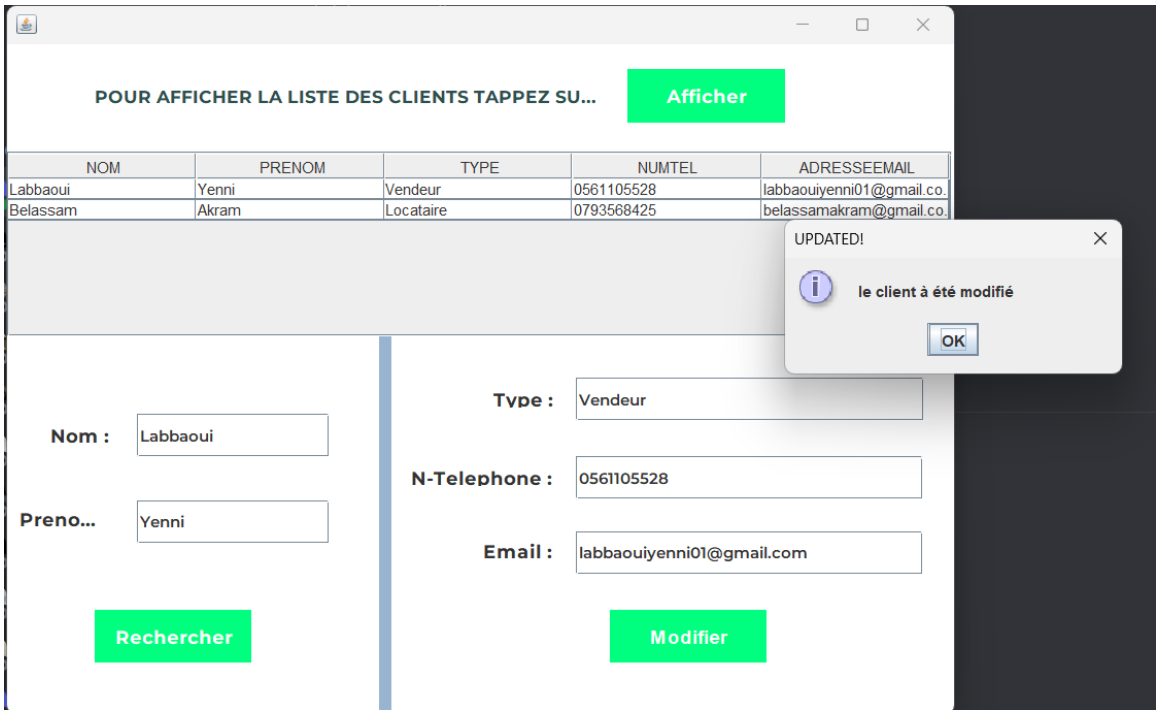
NOM	PRENOM	TYPE	NUMBI	DATED	DATEF	MONTANT
Labbaoui	Yenni	vente	001	2024-02-02 ...	2025-04-04 ...	500000

Modification :

- Pour la modification on fait l'affichage puis la rechreche puis
- On utilise

```
   while(rs.next()) {

        String NomCL1 = rs.getString("Nom");
        String PrenomCL1 = rs.getString("Prenom");
        String Type1 = rs.getString("Type");
        String Phone1 = rs.getString("NumTel");
        String Email1 = rs.getString("AdresseEmail");



        getTypeCL.setText(Type1);
        getPhoneCL.setText(Phone1);
        getEmailCL.setText(Email1);

}

rs.close();
```

Pour afficher les informations a afficher dans les champs d'écriture comme suit



JFRAMES de RDV :

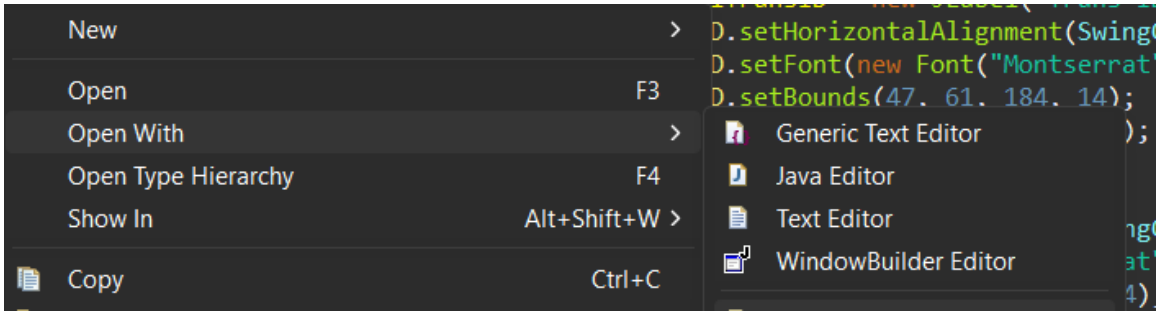
```
JButton btnAjouterBI = new JButton("AJOUTER\r\n");
btnAjouterBI.setFocusPainted(false);
btnAjouterBI.setForeground(new Color(255, 255, 255));
btnAjouterBI.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String NomCL = getNomCL.getText();
        String PrenomCL = getPrenomCL.getText();
        String Date = getDate.getText();
        String Desc = getDesc.getText();

        String query = "INSERT INTO RDV VALUES ('"+NomCL+"', '"+PrenomCL+"', to_date('"+Date+"', 'dd-mm-yyyy'), '"+Desc+"')";
        try {
            statement = connection.createStatement();
            statement.executeQuery(query);
            JOptionPane.showMessageDialog(null, "RDV à été ajouté", "ADDED!",JOptionPane.INFORMATION_MESSAGE);
        } catch (Exception e1) {

            JOptionPane.showMessageDialog(null, "ERROR OCCURED ! \n make sure that the information entered is correct and no field is empty \n note : make
            // TODO Auto-generated catch block
        }
```

JFRAMES de Transaction :

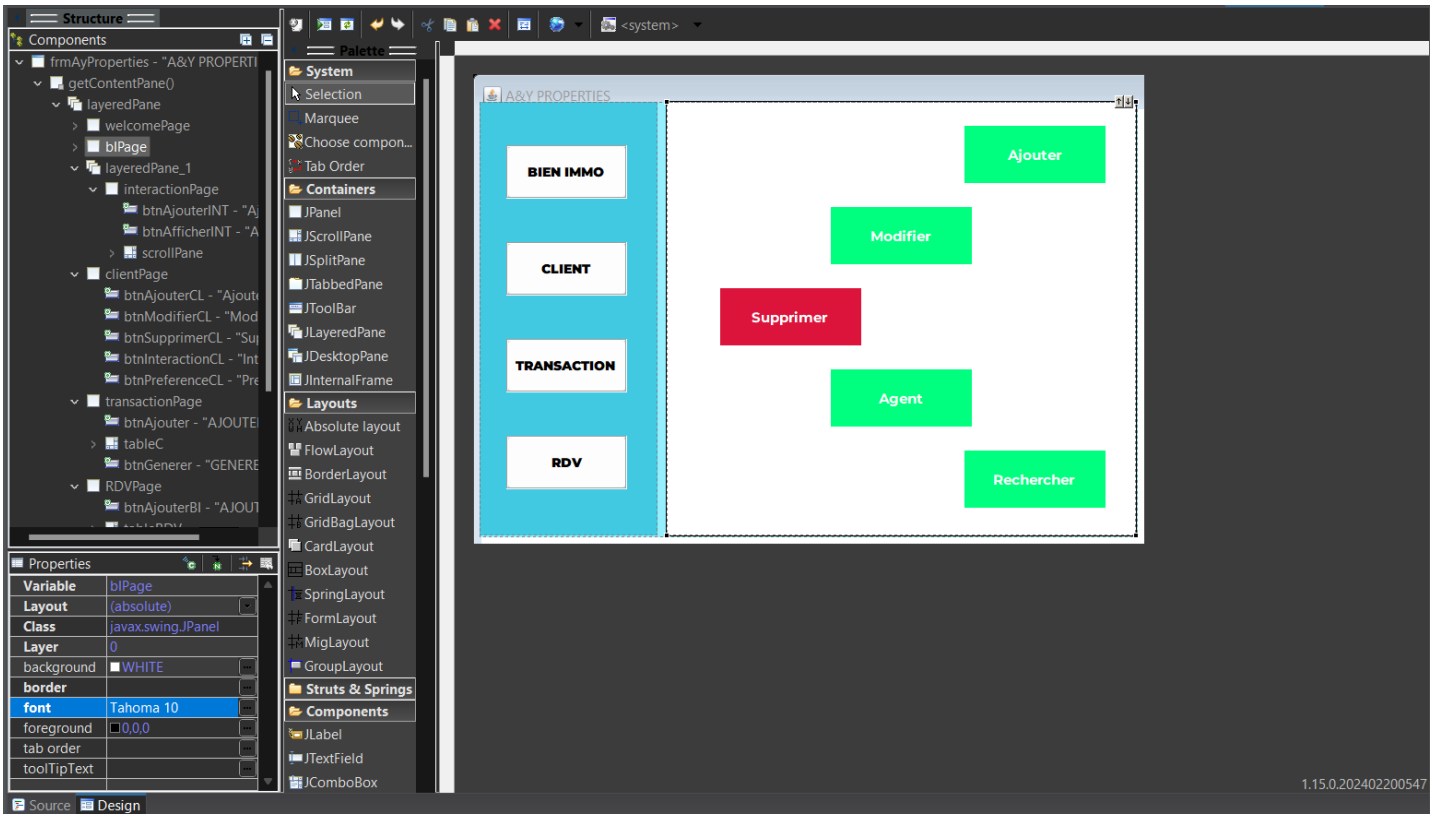
```
btnAjouterBI = new JButton("AJOUTER\r\n");
btnAjouterBI.setForeground(new Color(255, 255, 255));
btnAjouterBI.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String TransID = getTransID.getText();
        String NomCL = getNomCL.getText();
        String PrenomCL = getPrenomCL.getText();
        String NumBI = getNumBI.getText();
        String Type = getType.getText();
        String DateD = getDateD.getText();
        String DateF = getDateF.getText();
        String Montant = getMontant.getText();



        String sql = "INSERT INTO Transaction VALUES ('"+TransID+"', '"+NomCL+"', '"+PrenomCL+"', '"+Type+"', '"+NumBI+"', to_date('"+DateD+"','dd-mm-yyyy
        try {
            statement = connection.createStatement();
            statement.executeQuery(sql);
            JOptionPane.showMessageDialog(null, "transaction à été ajouté", "ADDED!",JOptionPane.INFORMATION_MESSAGE);
        } catch (Exception e1) {
            JOptionPane.showMessageDialog(null, "ERROR OCCURED ! \n make sure that the information entered is correct and no field is empty \n note : make
        }
```

Personalisation :

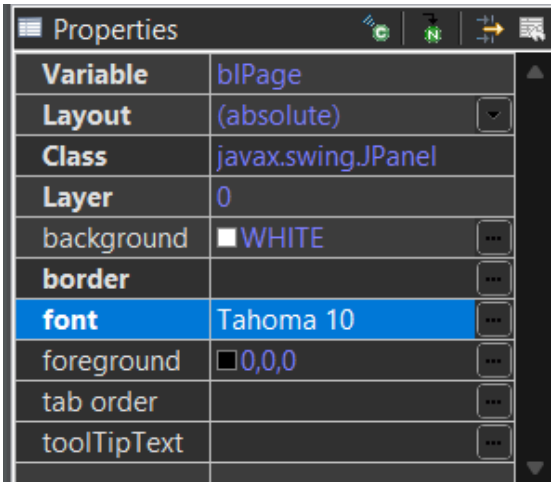
- Pour la personalisation on a utilisé l'extension (**WindowBuilder Editor**) qu'on a installé dans le marketplace de **Eclipse**



- Dans chaque JFRAME on a ajouté un PANEL ou on a pu par la suite ajouter des boutons, Jlabel, Text fields…etc
- On a attribué a chaque composant un nom unique pour ensuite ne pas y avoir d'ambiguité dans le code
- On a utilisé differentes fontes et couleurs de background et de foreground grace a l'editeur de design **WindowBuilder Editor**



- On personalise le nom et les fontes et les couleurs ici



Conception des tables de la base de données

BIENIMMOBILIER :

- Num (clé primaire)
- typeBI
- Surface
- Localisation
- Prix
- Description

AGENT

- NomAG
- PrenomAG
- Phone
- Email
- Clé primaire (NomAG, PrenomAG)

AFFECTATION

- NomAG * (clé étrangère refrences Agent)
- PrenomAG * (clé étrangère refrences Agent)
- NumBI * (clé étrangère refernces BienImmobilier)

CLIENT

- Nom
- Prenom
- Type
- NumTel
- AdresseEmail
- Clé primaire (Nom, Prenom)

PREFERENCE

- Nom * (clé étrangère refrences Client)
- Prenom *(clé étrangère refrences Client)
- Type
- Surface
- Prix_max
- Localisation

INTERACTION

- Nom * (clé étrangère refrences Client)
- Prenom * (clé étrangère refrences Client)
- Date_INT

- Description

TRANSACTION

- TransID (clé primaire)
- NomCL * (clé étrangère refrences Client)
- PrenomCL * (clé étrangère refrences Client)
- NumBI * (clé étrangère refrences BienImmobilier)
- DateD
- DateF
- Montant

RDV

- NomCL * (clé étrangère refrences Client)
- PrenomCL *(clé étrangère refrences Client)
- DateRDV

Script de creation des tables :

```
DROP TABLE Affectation
/
DROP TABLE AGENT
/
DROP TABLE Transaction
/
DROP TABLE RDV
/
drop table interaction
/
drop table preference
/
drop table client
/
DROP TABLE BienImmobilier
/
alter SESSION set nls_date_format = 'dd-mm-yyyy'
/

--Creation de la table bien immobilier
CREATE TABLE BienImmobilier (
    Num varchar2(20) PRIMARY KEY NOT NULL,
    typeBI varchar2(20) NOT NULL,
    Surface varchar2(20) NOT NULL,
    Localisation varchar2(20) NOT NULL,
    Prix varchar2(20) NOT NULL,
    Description varchar2(100)
    )
/


-- Création de la table client
CREATE TABLE client (
    nom VARCHAR2(20) NOT NULL,
    prenom VARCHAR2(20) NOT NULL,
    type VARCHAR2(20) NOT NULL,
    numTel VARCHAR2(20) NOT NULL,
    AdresseEmail VARCHAR2(30) NOT NULL,
    PRIMARY KEY (nom, prenom)
)
/
```

```
-- Création de la table preference
CREATE TABLE preference (
    nom VARCHAR2(20) NOT NULL,
    prenom VARCHAR2(20) NOT NULL,
    type VARCHAR2(20) NOT NULL,
    surface VARCHAR2(20) NOT NULL,
    prix_max VARCHAR2(20) NOT NULL,
    localisation VARCHAR2(50) NOT NULL,
    CONSTRAINT fk_preference_client FOREIGN KEY (nom, prenom)
        REFERENCES client (nom, prenom)
)
/

-- Création de la table interaction
CREATE TABLE interaction (
    nom VARCHAR2(20) NOT NULL,
    prenom VARCHAR2(20) NOT NULL,
    date_INT date NOT NULL,
    description VARCHAR2(100) NOT NULL,
    CONSTRAINT fk_interaction_client FOREIGN KEY (nom, prenom)
        REFERENCES client (nom, prenom)
)
/

CREATE TABLE Transaction(
    TransID varchar2(20) PRIMARY KEY NOT NULL,
    NomCL varchar2(20) NOT NULL,
    PrenomCL varchar2(20) NOT NULL,
    Type varchar2(20) NOT NULL,
    NumBI varchar2(20) NOT NULL,
    DateD date NOT NULL ,
    DateF date,
    Montant varchar2(20) NOT NULL,
    CONSTRAINT fk_Transaction_BI FOREIGN KEY (NumBI)
    REFERENCES BienImmobilier (Num),
    CONSTRAINT fk_Transaction_Client FOREIGN KEY (NomCL, PrenomCL)
    REFERENCES Client (Nom, Prenom)
)
/
```

```
CREATE TABLE Agent(
    NomAG varchar2(20) NOT NULL,
    PrenomAG varchar2(20) NOT NULL,
    Phone varchar2(20) NOT NULL,
    Email varchar2(30) NOT NULL,
    PRIMARY KEY (NomAG, PrenomAG)
)
/
CREATE TABLE Affectation(
    NomAG varchar2(20) NOT NULL,
    PrenomAG varchar2(20) NOT NULL,
    NumBI varchar2(20) NOT NULL,
    constraint fk_affectation_agent FOREIGN KEY (NomAG, PrenomAG)
    REFERENCES Agent(NomAG, PrenomAG),
    constraint fk_affectation_BI FOREIGN KEY (NumBI)
    REFERENCES BienImmobilier (Num)
)
/
CREATE TABLE RDV(
    NomCL varchar2(20) NOT NULL,
    PrenomCL varchar2(20) NOT NULL,
    DateRDV date NOT NULL,
    DESCRIPTION varchar2(100) NOT NULL,
    constraint fk_rdv_client FOREIGN KEY (NomCL, PrenomCL)
    REFERENCES Client(Nom, Prenom)
)
/
```