

Compte rendu final :

HENOCQUE Antoine

Au début du projet, nous nous sommes chacun donnés une tâche à faire en fonction de nos compétences.

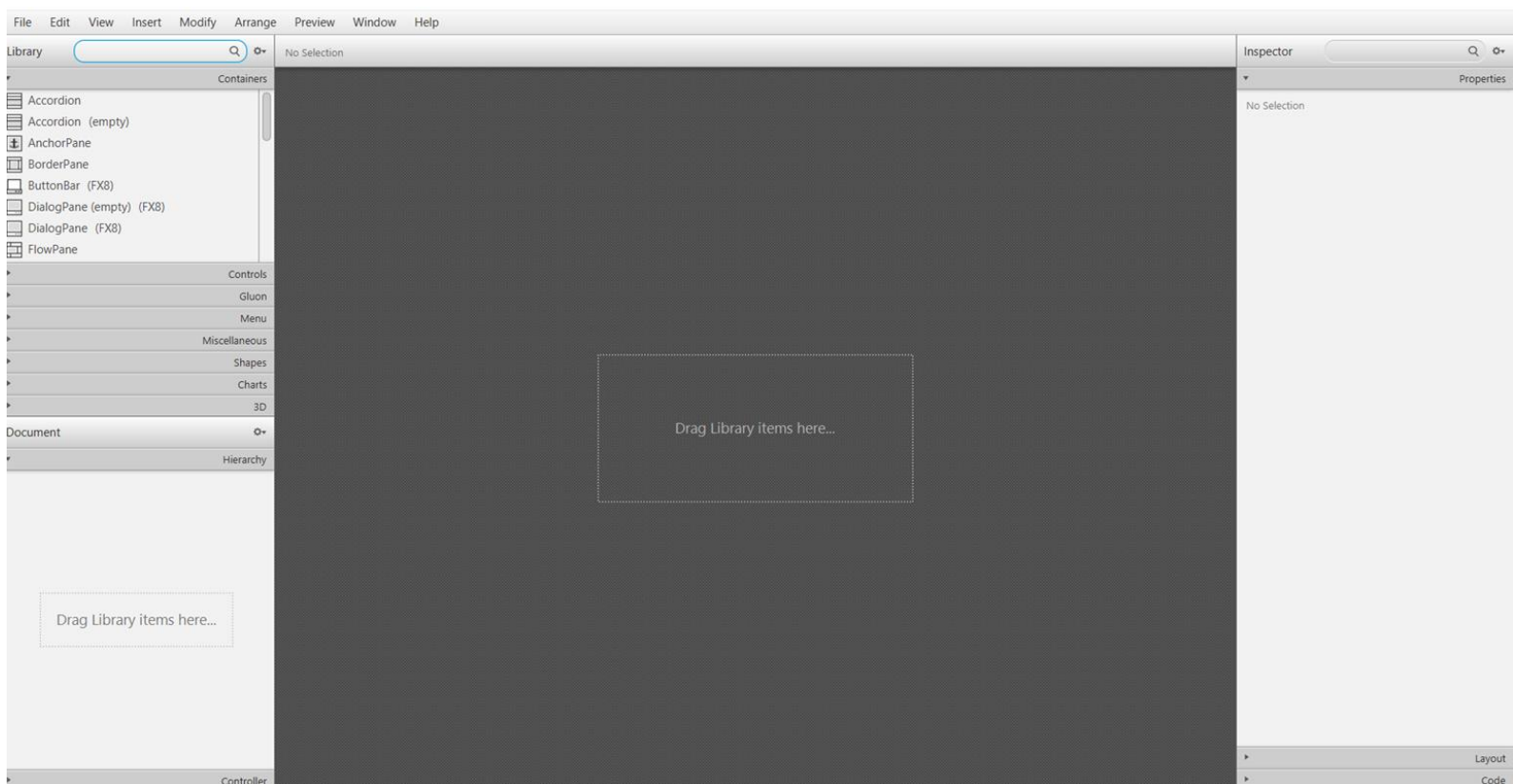
Ma tâche consistait à créer les pages « *Etat des frais engagé* » et « *Remboursement des frais engagé* ».

Pour ceci, j'ai utilisé l'application SceneBuilder, vivement recommandée par notre professeur.



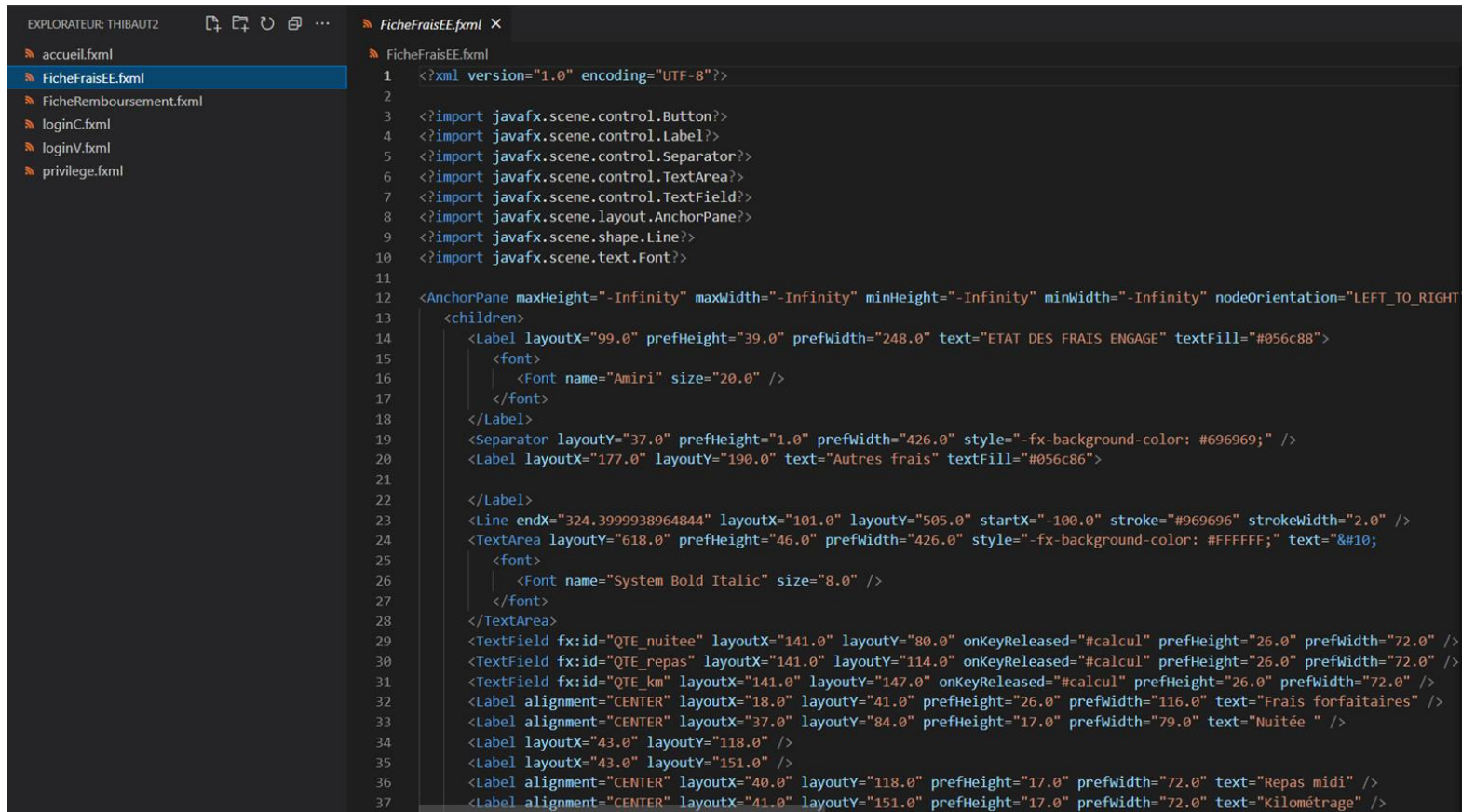
Dans un premier temps, j'ai dû me familiariser avec l'outil, afin de créer et d'optimiser au mieux les pages nécessaires.

Ci-dessous, voici le premier aperçu que l'on a en ouvrant l'application :



Pour créer les pages, il à été nécessaire de créer deux fichiers .fxml différents.

- Un fichier « FicheFraisEE.fxml » :

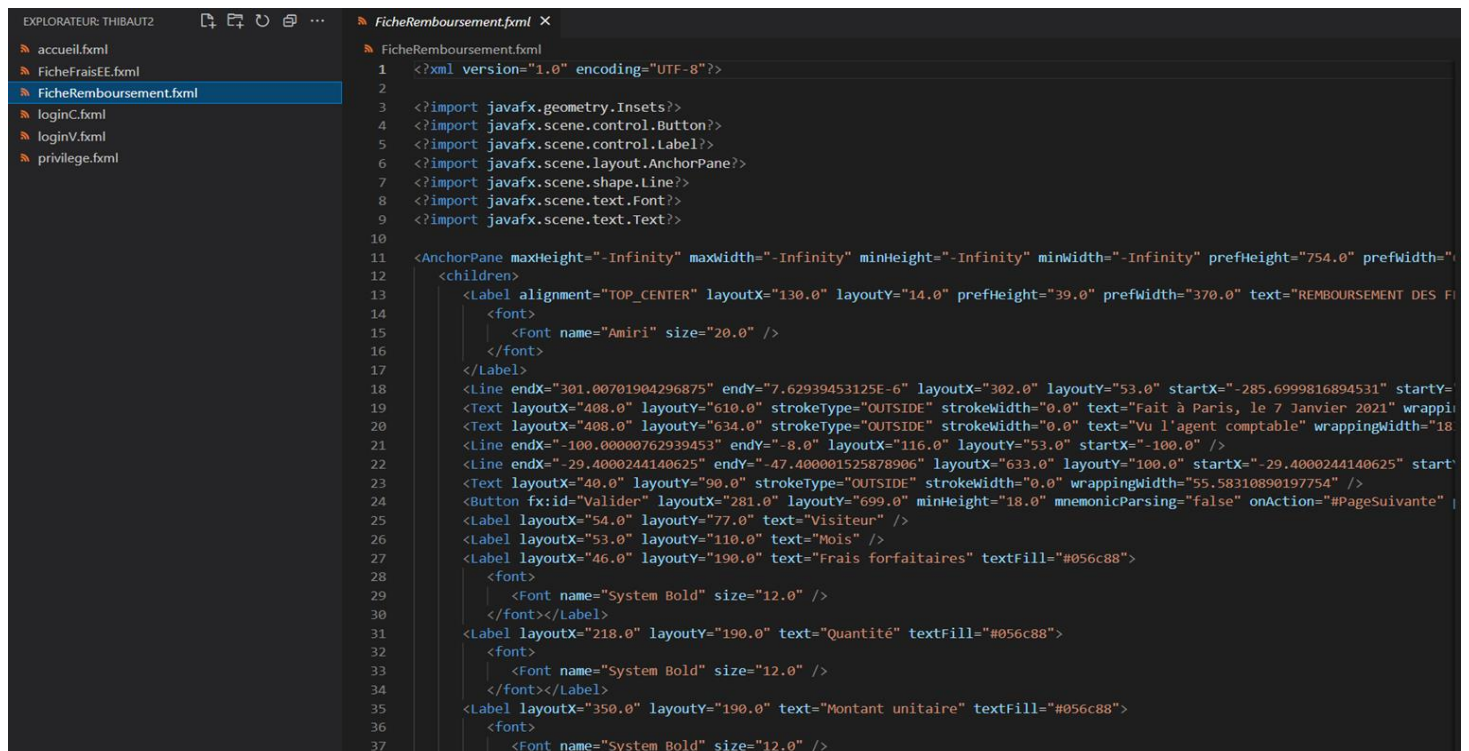


```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.scene.control.Button?>
4 <?import javafx.scene.control.Label?>
5 <?import javafx.scene.control.Separator?>
6 <?import javafx.scene.control.TextArea?>
7 <?import javafx.scene.control.TextField?>
8 <?import javafx.scene.layout.AnchorPane?>
9 <?import javafx.scene.shape.Line?>
10 <?import javafx.scene.text.Font?>
11
12 <AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" nodeOrientation="LEFT_TO_RIGHT">
13     <children>
14         <Label layoutX="99.0" prefHeight="39.0" prefWidth="248.0" text="ETAT DES FRAIS ENGAGE" textFill="#056c88">
15             <font>
16                 <Font name="Amiri" size="20.0" />
17             </font>
18         </Label>
19         <Separator layoutY="37.0" prefHeight="1.0" prefWidth="426.0" style="-fx-background-color: #696969;" />
20         <Label layoutX="177.0" layoutY="190.0" text="Autres frais" textFill="#056c86">
21
22         </Label>
23         <Line endX="324.3999938964844" layoutX="101.0" layoutY="505.0" startX="-100.0" stroke="#969696" strokeWidth="2.0" />
24         <TextArea layoutY="618.0" prefHeight="46.0" prefWidth="426.0" style="-fx-background-color: #FFFFFF;" text="#10;
25             <font>
26                 <Font name="System Bold Italic" size="8.0" />
27             </font>
28         </TextArea>
29         <TextField fx:id="QTE_nuitee" layoutX="141.0" layoutY="80.0" onKeyReleased="#calcul" prefHeight="26.0" prefWidth="72.0" />
30         <TextField fx:id="QTE_repas" layoutX="141.0" layoutY="114.0" onKeyReleased="#calcul" prefHeight="26.0" prefWidth="72.0" />
31         <TextField fx:id="QTE_km" layoutX="141.0" layoutY="147.0" onKeyReleased="#calcul" prefHeight="26.0" prefWidth="72.0" />
32         <Label alignment="CENTER" layoutX="18.0" layoutY="41.0" prefHeight="26.0" prefWidth="116.0" text="Frais forfaitaires" />
33         <Label alignment="CENTER" layoutX="37.0" layoutY="84.0" prefHeight="17.0" prefWidth="79.0" text="Nuitée " />
34         <Label layoutX="43.0" layoutY="118.0" />
35         <Label layoutX="43.0" layoutY="151.0" />
36         <Label alignment="CENTER" layoutX="40.0" layoutY="118.0" prefHeight="17.0" prefWidth="72.0" text="Repas midi" />
37         <Label alignment="CENTER" layoutX="41.0" layoutY="151.0" prefHeight="17.0" prefWidth="72.0" text="Kilométrage" />
```

- Un fichier « FicheRemboursement.fxml » :

C'est sur ces deux pages que les modifications effectuées sur SceneBuilders vont être prises en compte !

Pour terminer, après réflexions et conseils de mes collègues et d'autres groupes, je suis



```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.Insets?>
4 <?import javafx.scene.control.Button?>
5 <?import javafx.scene.control.Label?>
6 <?import javafx.scene.layout.AnchorPane?>
7 <?import javafx.scene.shape.Line?>
8 <?import javafx.scene.text.Font?>
9 <?import javafx.scene.text.Text?>
10
11 <AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="754.0" prefWidth="1280.0">
12     <children>
13         <Label alignment="TOP_CENTER" layoutX="130.0" layoutY="14.0" prefHeight="39.0" prefWidth="370.0" text="REMBOURSEMENT DES FRAIS" />
14         <font>
15             <Font name="Amiri" size="20.0" />
16         </font>
17     </Label>
18     <Line endX="301.00701904296875" endY="7.62939453125E-6" layoutX="302.0" layoutY="53.0" startX="-285.6999816894531" startY="53.0" />
19     <Text layoutX="408.0" layoutY="610.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Fait à Paris, le 7 Janvier 2021" wrappingWidth="180.0" />
20     <Text layoutX="408.0" layoutY="634.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Vu l'agent comptable" wrappingWidth="180.0" />
21     <Line endX="-100.00000762939453" endY="-8.0" layoutX="116.0" layoutY="53.0" startX="-100.0" />
22     <Line endX="-29.4000244140625" endY="-47.400001525878906" layoutX="633.0" layoutY="100.0" startX="-29.4000244140625" startY="100.0" />
23     <Text layoutX="40.0" layoutY="90.0" strokeType="OUTSIDE" strokeWidth="0.0" wrappingWidth="55.58310890197754" />
24     <Button fx:id="Valider" layoutX="281.0" layoutY="699.0" minHeight="18.0" mnemonicParsing="false" onAction="#PageSuivante" />
25     <Label layoutX="54.0" layoutY="77.0" text="Visiteur" />
26     <Label layoutX="53.0" layoutY="110.0" text="Mois" />
27     <Label layoutX="46.0" layoutY="190.0" text="Frais forfaitaires" textFill="#056c88">
28         <font>
29             <Font name="System Bold" size="12.0" />
30         </font></Label>
31     <Label layoutX="218.0" layoutY="190.0" text="Quantité" textFill="#056c88">
32         <font>
33             <Font name="System Bold" size="12.0" />
34         </font></Label>
35     <Label layoutX="350.0" layoutY="190.0" text="Montant unitaire" textFill="#056c88">
36         <font>
37             <Font name="System Bold" size="12.0" />
38         </font></Label>
39     </children>
40 </AnchorPane>
```

arrivé aux résultats suivants :

REMBOURSEMENT DES FRAIS ENGAGES

Visiteur
Label

Mois
Label

Frais forfaitaires	Quantité	Montant unitaire	Total
Nuitée	Label	Label	Label
Repas midi	Label	Label	Label
Véhicule	Label	Label	Label
Autres frais			
Date	Libellé	Montant	
Label	Label	Label	
Label	Label	Label	

Label	€
-------	---

Fait à Paris, le 7 Janvier 2021

Vu l'agent comptable

Valider

ETAT DES FRAIS ENGAGE

Frais forfaitaires	Quantité	Montant unitaire	Total
Nuitée	<input type="text"/>	<input type="text" value="80.00"/>	<input type="text"/>
Repas midi	<input type="text"/>	<input type="text" value="29.00"/>	<input type="text"/>
Kilométrage	<input type="text"/>	<input type="text" value="0.52"/>	<input type="text"/>

Autres frais

Date	Libelle	Montant
<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>

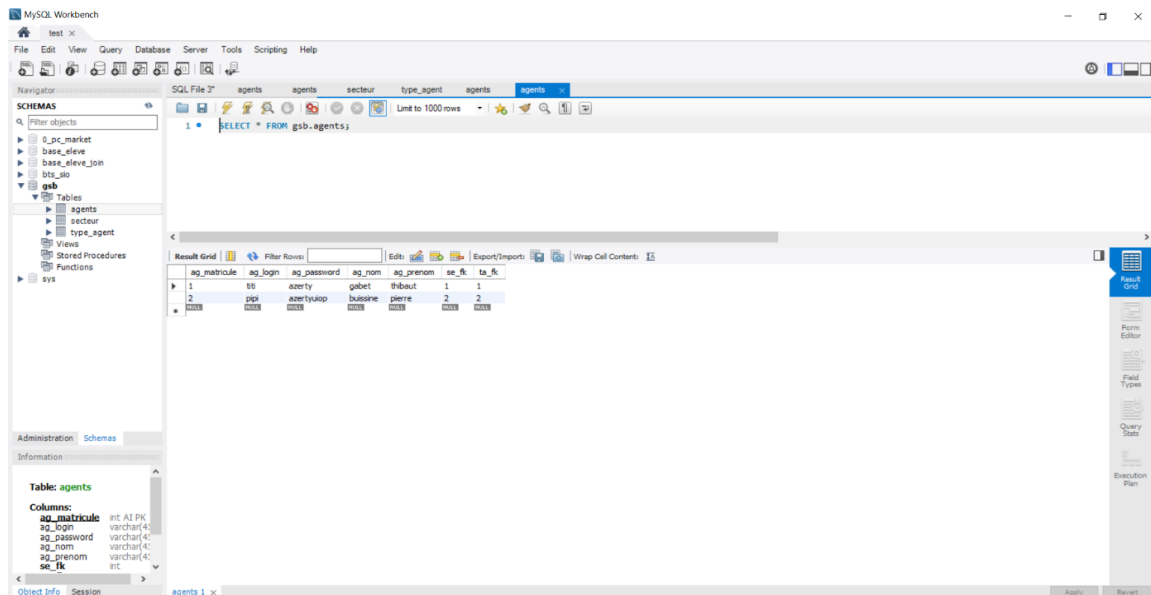
Valider fiche de frais

- Ce travail m’a permis de développer des compétences à travailler en groupe et sur une longue durée.
- J’ai appris à plus écouter les avis critiques négatifs que je n’acceptais pas avant.
- Ce fût une bonne première expérience de travail et j’ai pris beaucoup de plaisir à me sentir utile et à contribuer au bon développement de notre application malgré un manque de fonctionnalités.

GABET Thibaut

Formulaire de connexion :

Tout d'abord j'ai créé la base de données sur Worbench.



Ensuite j'ai créé les éléments nécessaires au formulaire avec SceneBuilder.

The screenshot shows a login form titled 'login'. It contains two input fields: 'user :' and 'mdp :'. Below the input fields is a button labeled 'connexion'.

Puis je suis passé aux codes du formulaire avec JavaFX.

Grace au code si dessous je récupéré le texte qu'a écrit l'utilisateur dans les casses login et mot de passe. S'il n'y a rien alors je ne me connecte pas à la bdd.

```

System.out.println( C_login.getText());
System.out.println( C_mdp.getText());

if (!C_login.getText().equals("") && !C_mdp.getText().equals("")) {

    String dbURL = "jdbc:mysql://localhost:3306/gsb";
    String username = "root";
    String password = "";

    try {

        Connection conn = DriverManager.getConnection(dbURL, username, password);

        if (conn != null) {
            System.out.println("Connected");

```

Je récupère les données de la Table agents pour les mettre dans des variable String.

```

//étape 3: créer l'objet statement
Statement stmt = conn.createStatement();
String sql = "SELECT ag_matricule, ag_login, ag_password, ag_prenom, ag_nom FROM agents";
ResultSet res = stmt.executeQuery(sql);

while(res.next()){
    //Récupérer par nom de colonne
    String matricule = res.getString("ag_matricule");
    String login = res.getString("ag_login");
    String mdp = res.getString("ag_password");
    String prenom = res.getString("ag_prenom");
    String nom = res.getString("ag_nom");

```

Ensuite je vérifie si les données de la Table agents concordent avec les données d'entrée du login et du mot de passe de l'utilisateur qui souhaite se connecter. Si c'est le cas l'utilisateur se connecte et change de page.

(j'ai décidé d'afficher les données pour pouvoir vérifier que tout fonctionne correctement)

```

if (C_login.getText().equals(login) && C_mdp.getText().equals(mdp)) {
System.out.print(", matricule: " + matricule);
System.out.print(", login: " + login);
System.out.print(", password: " + mdp);

```

```
System.out.print(", prénom: " + prenom);
System.out.print(", nom: " + nom);

App.setRoot("secondary");
```

	ag_matricule	ag_login	ag_password	ag_nom	ag_prenom	se_fk	ta_fk
▶	1	titi	azerty	gabet	thibaut	1	1

```
titi
azerty
Connected
, matricule: 1, login: titi, password: azerty, prénom: thibaut, nom: gabet
```

J'ai ensuite créé une classe `Donnee` pour pouvoir appeler les données de l'utilisateur sur une autre page de l'application.

```
package fr.thibaut2;

public class Donnee {
    public static String matricule;
    public static String login;
    public static String mdp;
    public static String prenom;
    public static String nom;
}
```

J'ai modifié ma requête sql pour pouvoir cibler les données que je souhaite récupérer en les comparant avec le login et le mot de passe entré par l'utilisateur. Cela permet de ne pas récupérer toutes les données de la table agents.

```
String sql = "SELECT ag_matricule, ag_login, ag_password, ag_prenom, ag_nom
FROM compta WHERE ag_login='"+C.login.getText()+"' AND ag_password='"+C.login.getText()+"'
IS NOT NULL";
```

J'ai fait en sorte qu'il y est un message d'erreur si l'utilisateur entre un mot de passe incorrect ou un mauvais login incorrect.

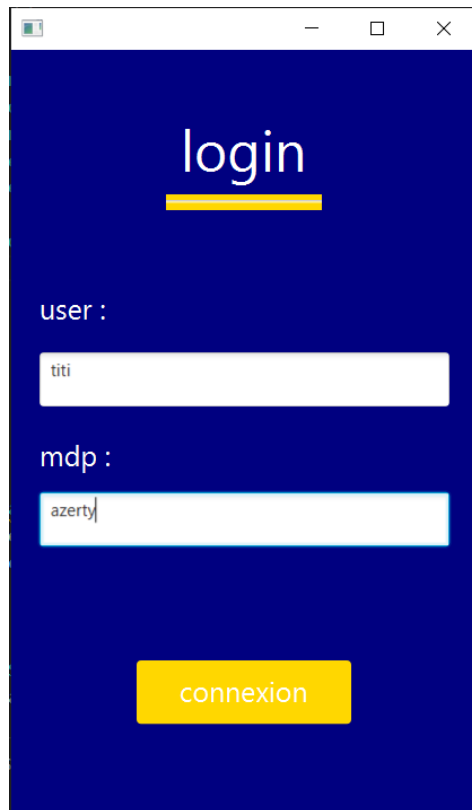
```
} else {
    Erreur.setText("identifiant incorrect");
}
```

Si le login et le mot de passe est correct alors l'utilisateur se connectera et changera de page.

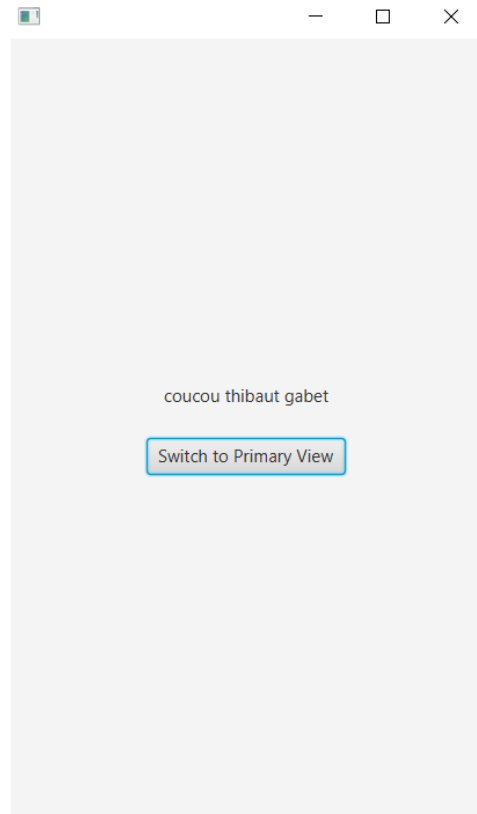
Sur la deuxième page un message s'affiche avec les données de l'utilisateur récupéré grâce à la classe `Donnee`.


```
Text_U.setText("coucou " + Donnee.prenom + " " + Donnee.nom);
```

Pour finir j'ai rajouté un peu de design à l'application.



The login screen has a dark blue background. At the top, the word "login" is written in white, underlined with a yellow bar. Below it, the text "user :" is followed by a white text input field containing "titi". Further down, the text "mdp :" is followed by a white password input field containing "azerty". At the bottom, there is a yellow button with the text "connexion" in white.



	ag_matricule	ag_login	ag_password	ag_nom	ag_prenom	se_fk	ta_fk
▶	1	titi	azerty	gabet	thibaut	1	1

Fiche de frais :

ETAT DES FRAIS ENGAGE			
Frais forfaitaires	Quantité	Montant unitaire	Total
Nuitée	<input type="text"/>	80.00	<input type="text"/>
Repas midi	<input type="text"/>	29.00	<input type="text"/>
Kilométrage	<input type="text"/>	0.52	<input type="text"/>
Autres frais			
Date	Libelle	Montant	
<input type="text"/>	<input type="text"/>	<input type="text"/>	
<input type="text"/>	<input type="text"/>	<input type="text"/>	
<input type="text"/>	<input type="text"/>	<input type="text"/>	
<input type="button" value="Valider fiche de frais"/>			

J'ai ajouté des id correspondant au casse dont je souhaite récupérer les valeurs d'entrer de l'utilisateur, que j'ai ensuite récupérer sur ma page FicheDeFraisEEController.

```
@FXML
private TextField MU_nuitee;

@FXML
private TextField QTE_nuitee;

@FXML
private TextField TOTAL_nuitee;

@FXML
private TextField MU_repas;

@FXML
private TextField QTE_repas;

@FXML
private TextField TOTAL_repas;

@FXML
private TextField MU_km;

@FXML
private TextField QTE_km;

@FXML
private TextField TOTAL_km;

@FXML
private Button Validation;
```

j'ai ensuite codé le calcul en java pour pouvoir changer la valeur du total et l'afficher au moment où l'utilisateur entre sa quantité dans les cases.

```
@FXML
void calcul(KeyEvent event) {

    //calcule nuitée

    //recuperation et transformation variable 1
    String quantiter_nuitee = QTE_nuitee.getText();
    int nbr_QTE_nuitee = Integer.parseInt(quantiter_nuitee);

    //recuperation et transformation variable 2
    String montant_unitaire_nuitee = MU_nuitee.getText();
    double nbr_MU_nuitee = Double.parseDouble(montant_unitaire_nuitee);

    //calcul et transformation resultat
    double TOTAL1 = nbr_QTE_nuitee * nbr_MU_nuitee;
    String TOTALnuitee = String.valueOf(TOTAL1);

    //affichage resultat
    TOTAL_nuitee.setText(TOTALnuitee);

    //-----//

    //calcule repas

    //recuperation et transformation variable 1
    String quantiter_repas = QTE_repas.getText();
    int nbr_QTE_repas = Integer.parseInt(quantiter_repas);

    //recuperation et transformation variable 2
    String montant_unitaire_repas = MU_repas.getText();
    double nbr_MU_repas = Double.parseDouble(montant_unitaire_repas);

    //calcul et transformation resultat
    double TOTAL2 = nbr_QTE_repas * nbr_MU_repas;
    String TOTALrepas = String.valueOf(TOTAL2);

    //affichage resultat
    TOTAL_repas.setText(TOTALrepas);

    //-----//

    //calcule km

    //recuperation et transformation variable 1
    String quantiter_km = QTE_km.getText();
    int nbr_QTE_km = Integer.parseInt(quantiter_km);
```

```

//recuperation et transformation variable 2
String montant_unitaire_km = MU_km.getText();
double nbr_MU_km = Double.parseDouble(montant_unitaire_km);

//calcul et transformation resultat
double TOTAL3 = nbr_QTE_km * nbr_MU_km;
String TOTALkm = String.valueOf(TOTAL3);

//affichage resultat
TOTAL_km.setText(TOTALkm);

//-----//
}

```

— □ ×

ETAT DES FRAIS ENGAGE

Frais forfaitaires	Quantité	Montant unitaire	Total
Nuitée	<input type="text" value="2"/>	<input type="text" value="80.00"/>	<input type="text" value="160.0"/>
Repas midi	<input type="text" value="3"/>	<input type="text" value="29.00"/>	<input type="text" value="87.0"/>
Kilométrage	<input type="text" value="100"/>	<input type="text" value="0.52"/>	<input type="text" value="52.0"/>

Autres frais

Date	Libelle	Montant
<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>

Valider fiche de frais

Pour finir j’ai écrit le code de la requête sql pour envoyer les données de quantité à la base de données.

```

@FXML
void ValidationEvent() throws IOException {

    if (!QTE_repas.getText().equals("") && !QTE_nuitee.getText().equals("") &&
!QTE_km.getText().equals("")) {

        String dbURL = "jdbc:mysql://localhost:3306/application";
        String username = "root";
        String password = "";

        try {

            Connection conn = DriverManager.getConnection(dbURL, username, password);

            if (conn != null) {
                System.out.println("Connected");

                // étape 3: créer l'objet statement
                Statement stmt = conn.createStatement();
                String sql = "INSERT INTO fiche_de_frais (repas_midi, kilometrage, nuitee) VALUES
('"+QTE_repas.getText()+"', '"+QTE_km.getText()+"', '"+QTE_nuitee.getText()+"')";
                stmt.executeUpdate(sql);
            }

        } catch (SQLException ex) {
            ex.printStackTrace();
        }

    }
}

```

	fiche_id	repas_midi	kilometrage	nuitee	fk_autre
▶	3	3	100	2	NULL
*	NULL	NULL	NULL	NULL	NULL

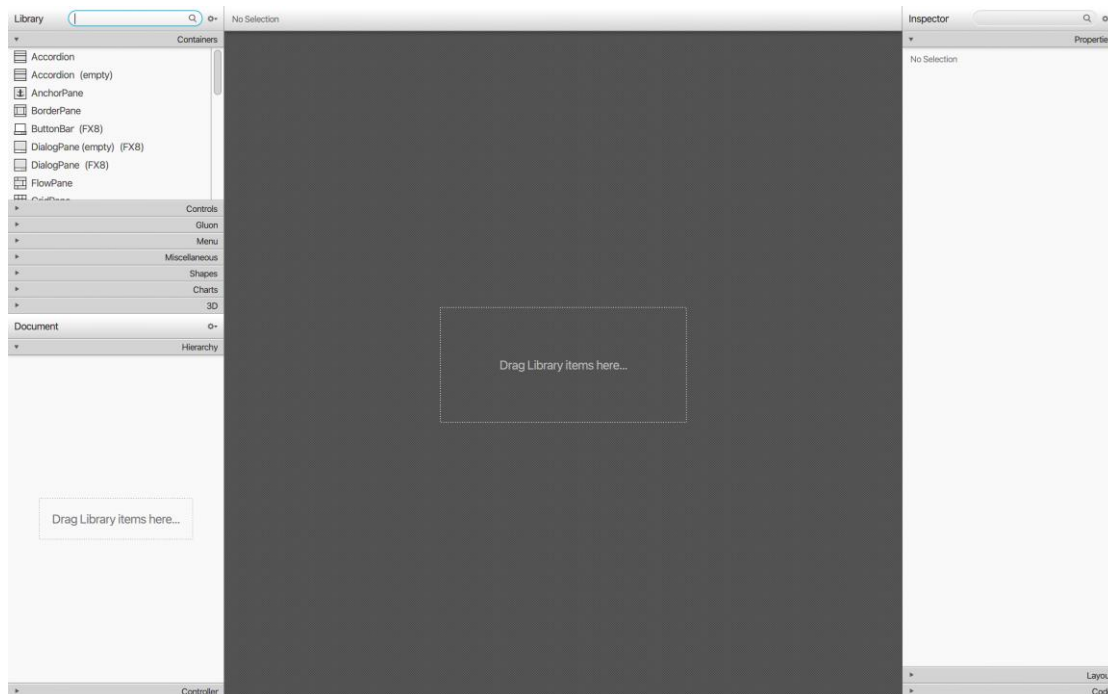
Ce projet en groupe fut très intéressant à développer et m'a permis d'acquérir de nouvelles compétences qui me seront très utiles et me permettront de finir cette application et d'en créer d'autres par la suite.

RAVAUT Simon

Tout d'abord à l'aide de l'application Scène Builder, ma tâche était de réaliser la page de connexion de l'application.

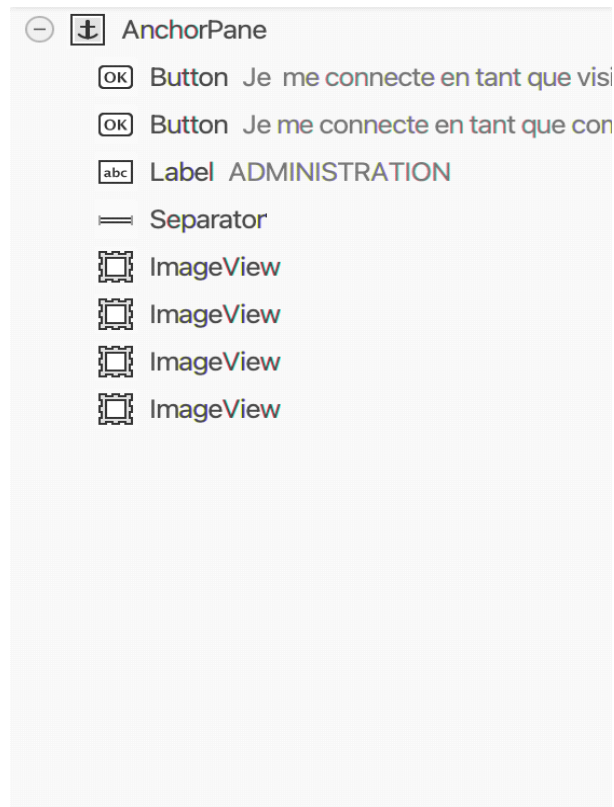
Cette page va permettre de s'identifier en tant que visiteur ou comptable. Selon le choix de connexion, la personne sera redirigée vers une autre page qui sera approprié au statut de la personne.

Voici ci-dessous l'application scène builder permettant de s'occuper de la partie graphique :



Ce logiciel permet d'exécuter des pages fxml .FXML est un format de données textuelles, dérivé du format XML, qui permet de décrire une interface utilisateur pour des applications conçus avec JavaFX .

Ci-dessous, on peut voir les objets que j'ai utilisés afin de conceptualiser la page de connexion. On y retrouve le type de page / bouton / label (texte) / ligne séparateur / ainsi que des images .



La page de connexion une fois terminée, sera reliée à la page d'accueil mais avant cela on va devoir la connecter à la base de données afin que les visiteurs ou les comptables puissent s'identifier avec de vrai login et mot de passe.

Sur Visual Studio code, la bibliothèque javafx nous crée des pages FXML qu'on va pouvoir modifier afin de rajouter des actions pour des boutons ou encore relier différentes pages FXML lors d'une clique sur un bouton. Cela va permettre de passer d'une page à une autre.

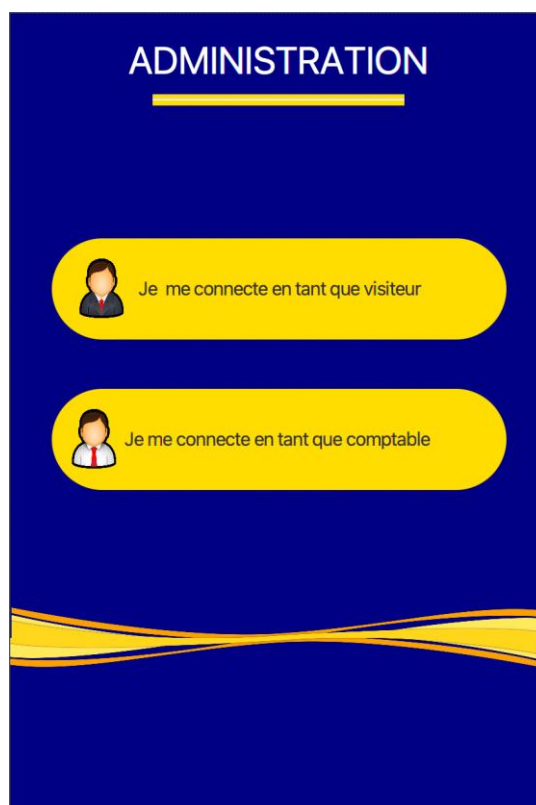
On peut voir ci-dessous la page FXML que j'ai édité afin que lorsque l'on clique sur un bouton, il nous renvoie sur une deuxième page.

```

src > main > java > fr > simon2 > PrimaryController.java > ...
1  package fr.simon2;
2
3  import java.io.IOException;
4
5  import javafx.event.ActionEvent;
6  import javafx.fxml.FXML;
7  import javafx.scene.control.Button;
8
9  public class PrimaryController {
10
11     @FXML
12     private Button Cbutton;
13
14     @FXML
15     private Button Vbutton;
16
17     @FXML
18     void SwitchToComptable(ActionEvent event) throws IOException {
19         App.setRoot("secondary");
20     }
21
22     @FXML void fr.simon2.PrimaryController.SwitchToVisiteur(ActionEvent event) throws IOException
23     void SwitchToVisiteur(ActionEvent event) throws IOException {
24         App.setRoot("secondary");
25     }
26
27 }
28

```

Ci- dessous on peut voir le résultat graphique de nôtre page de connexion :



Pour la fin de ce projet, je me suis occupé d'intégrer toutes les fiches de paiements donc la fiche de remboursements et les frais engagés. J'ai aussi vérifié lorsque l'on rentre les données dans la fiche de frais et que l'on clique sur le bouton valider, les informations sont bien envoyées dans la base de données. Pour cela j'ai dû remplacer l'ancienne base de données nommée gsb par la nouvelle base de données nommée application. Ensuite comme Thibaut s'occupait plus de la partie code et moi de la partie intégration ainsi que Scene Builder pour la création de l'interface et des fiches, j'aidais Thibaut dans la recherche de solution pour diverses choses qui nous été inconnu comme le convertisseur des valeurs par exemple.

Comme Thibaut s'y connaît beaucoup plus que nous dans le langage java j'étais vraiment là pour l'assister et essayé de l'aider. J'avais pour tâche aussi de vérifier le bon fonctionnement de l'application afin de regarder si tout marchait bien. Je vérifie la connexion au compte visiteurs ainsi qu'au compte comptable pour voir le bon fonctionnement des identifiants et mot de passe. L'envoi des données dans notre base de données, le fonctionnement du convertisseur de valeur ainsi que les actions lors d'un appui sur un bouton.

Je suis très fier de notre application même si elle est encore incomplète car cela nous a permis de travailler en équipe et d'apprendre la répartition des tâches par le biais de la communication.